Note to readers:
Please ignore these sidenotes; they're just hints to myself for preparing the index, and they're often flaky!

KNUTH

THE ART OF COMPUTER PROGRAMMING

VOLUME 4 PRE-FASCICLE 8A

HAMILTONIAN PATHS AND CYCLES

DONALD E. KNUTH *Stanford University*

ADDISON–WESLEY

May 8, 2026

Internet
Stanford GraphBase
MMIX

Internet page https://www-cs-faculty.stanford.edu/~knuth/taocp.html contains current information about this book and related books.

See also https://www-cs-faculty.stanford.edu/~knuth/sgb.html for information about *The Stanford GraphBase*, including downloadable software for dealing with the graphs used in many of the examples in Chapter 7.

See also https://www-cs-faculty.stanford.edu/~knuth/mmixware.html for downloadable software to simulate the MMIX computer.

See also https://www-cs-faculty.stanford.edu/~knuth/programs.html for various experimental programs that I wrote while writing this material (and some data files).

Copyright © 2026 by Addison–Wesley

All rights reserved. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form, or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior consent of the publisher, except that the official electronic file may be used to print single copies for personal (not commercial) use.

Zeroth printing (revision -48), 08 May 2026

BARRY
Internet

PREFACE

But that is not my point.
I have no point.
— DAVE BARRY (2002)

THIS BOOKLET contains draft material that I'm circulating to experts in the field, in hopes that they can help remove its most egregious errors before too many other people see it. I am also, however, posting it on the Internet for courageous and/or random readers who don't mind the risk of reading a few pages that have not yet reached a very mature state. *Beware:* This material has not yet been proofread as thoroughly as the manuscripts of Volumes 1, 2, 3, 4A, and 4B were at the time of their first printings. And alas, those carefully checked volumes were subsequently found to contain thousands of mistakes.

Given this caveat, I hope that my errors this time will not be so numerous and/or obtrusive that you will be discouraged from reading the material carefully. I did try to make the text both interesting and authoritative, as far as it goes. But the field is vast; I cannot hope to have surrounded it enough to corral it completely. So I beg you to let me know about any deficiencies that you discover.

To put the material in context, this portion of fascicle 8 previews Section 7.2.2.4 of *The Art of Computer Programming*, entitled "Hamiltonian paths and cycles." I haven't had time to write much of it yet — not even this preface!

* * *

The explosion of research in combinatorial algorithms since the 1970s has meant that I cannot hope to be aware of all the important ideas in this field. I've tried my best to get the story right, yet I fear that in many respects I'm woefully ignorant. So I beg expert readers to steer me in appropriate directions.

Please look, for example, at the exercises that I've classed as research problems (rated with difficulty level 46 or higher), namely exercises 161, 205, 210, 225, ...; I've also implicitly mentioned or posed additional unsolved questions in the answers to exercises 281, 369, 370, 375, ... Are those problems still open? Please inform me if you know of a solution to any of these intriguing questions. And of course if no solution is known today but you do make progress on any of them in the future, I hope you'll let me know.

I urgently need your help also with respect to some exercises that I made up as I was preparing this material. I certainly don't like to receive credit for things that have already been published by others, and most of these results are quite natural "fruits" that were just waiting to be "plucked." Therefore please tell me if you know who deserves to be credited, with respect to the ideas found in exercises

36, 41, 42, 53, 55, 62, 63, 65, 71, 73, 84, 100, 106, 135, 136, 137, 138, 156, 157, 158, 159, 163, 177, 185, 202, 208, 215, 216, 217, 218, 223, 242, 246, 247, 270, 271, 275, 279, 285, 286, 287(c), 290, 300, 302, 341, 343, 350, 360, 361, 369, 372(a, b, c), ... For example, exercise 290 is surely well known; do you know a reference? Furthermore I've credited exercises 79, ... to unpublished work of Nikolai Beluhov and .... Have any of those results ever appeared in print, to your knowledge?

Beluhov

While writing this section I also wrote numerous programs for my own edification. (I usually can't understand things well until I've tried to explain them to a machine.) Most of those programs were quite short, of course; but several of them are rather substantial, and possibly of interest to others. Therefore I've made a selection available by listing some of them on the following webpage:

`https://cs.stanford.edu/~knuth/programs.html`

In particular, prototypes of the main algorithms can be found there: Algorithm B (SSBIDIHAM, and SSDIHAM for the special case of digraphs); Algorithm C (HAMSAT); Algorithm E (DYNAHAM) and E^+ (DYNAHAMP); Algorithm F (HAM-EULER); Algorithm H (SSHAM); Algorithm W (BACK-WARNSDORF). A few other programs are also mentioned in the answers to certain exercises. If you want to see a program called FOO, look for FOO on that webpage. See also

SSDIHAM

`https://cs.stanford.edu/~knuth/programs/ham-benchmarks.tgz`

for the benchmark graphs in Table 1.

benchmark graphs

* * *

Special thanks are due to George Jelliss, Arnaud Lefebvre, Shisheng Li, Filip Stappers, Peter Weigel, Udo Wermuth, ... for their detailed comments on my early attempts at exposition, and to numerous other correspondents who've contributed crucial corrections. Andrej Krevl and Brian Roberts have helped me to utilize hundreds of core processors on powerful computers in Stanford's Infolab—quite a thrill!

Jelliss
Lefebvre
Li
Stappers
Weigel
Wermuth
Krevl
Roberts
Stanford's Infolab

* * *

I happily offer a "finder's fee" of \$2.56 for each error in this draft when it is first reported to me, whether that error be typographical, technical, or historical. The same reward holds for items that I forgot to put in the index. And valuable suggestions for improvements to the text are worth 32¢ each. (Furthermore, if you find a better solution to an exercise, I'll actually do my best to give you immortal glory, by publishing your name in the eventual book:–)

Cross references to yet-unwritten material sometimes appear as '00'; this impossible value is a placeholder for the actual numbers to be supplied later.

Happy reading!

Stanford, California D. E. K.
99 Umbruary 2016

Knuth
KNUTH
Ewing

I have twenty years' work ahead of me
to finish The Art of Computer Programming.
— DONALD E. KNUTH, letter to John Ewing (04 September 1990)

CONTENTS

Chapter 7 — Combinatorial Searching . . . . . [4A.1]

7.2. Generating All Possibilities . . . . . [4A.281]
7.2.1. Generating Basic Combinatorial Patterns . . . . . [4A.281]
7.2.2. Backtrack Programming . . . . . [4B.30]
7.2.2.1. Dancing links . . . . . [4B.65]
7.2.2.2. Satisfiability . . . . . [4B.185]
7.2.2.3. Constraint satisfaction . . . . . [4fasc7.1]
7.2.2.4. Hamiltonian paths and cycles 1
Hamiltonian paths in antiquity 2
A greedy heuristic 6
Path flipping 9
Searching exhaustively 11
A census of knight's tours 18
Dynamic enumeration 21
Directed and bidirected graphs 30
SAT encodings 37
*Color coding 42
History 43
Exercises 49

Answers to Exercises 72

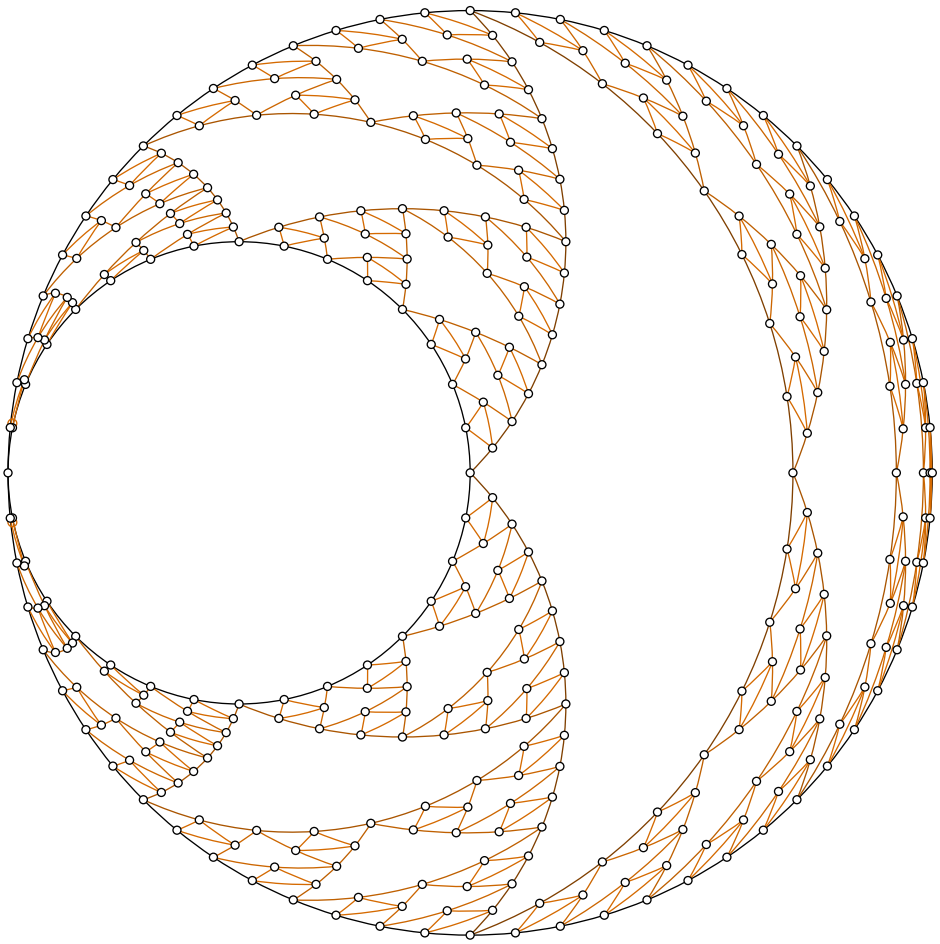
Index to Algorithms and Theorems . . . . . ??

Answers to Puzzles in the Answers 126

Index and Glossary 127

I always thought Volume 4 was a myth,
like the missing part of the Dead Sea scrolls.
— BILL GASARCH (blog post, 10 January 2008)

pinched gasket



May 8, 2026

A long train of consistent calculations opens itself out, for every result of which there is found a corresponding geometrical interpretation, in the theory of two of the celebrated solids of antiquity, alluded to with interest by Plato in the Timæus; namely, the Icosaedron, and the Dodecaedron.

— WILLIAM ROWAN HAMILTON (1856)

The total number of possible [knight's] tours that can be made is so vast that it is safe to predict that no mathematician will ever succeed in counting up the total.

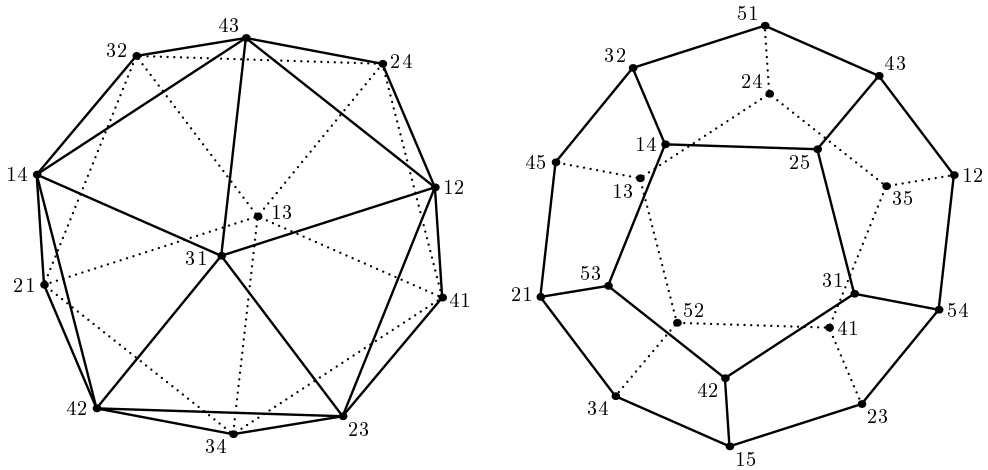
— ERNEST BERGHOLT (1915)

I'll show that this problem is susceptible to a very special analysis, which merits extra attention because it involves reasoning of a kind rarely used elsewhere. The excellence of Analysis is easy to see, but most people think that it's limited to traditional questions about Mathematics; hence it will always be quite important to apply Analysis to subjects that seem to make it out of reach, for it incorporates the art of reasoning in the highest degree. One cannot then extend the bounds of Analysis without justifiably expecting great advantages.

— LEONHARD EULER (1759)

Plato
HAMILTON
BERGHOLT
EULER
Hamilton
quaternions
Platonic solids
icosahedron
Icosian Game
planar graph
dual of a planar graph
dodecahedron

7.2.2.4. Hamiltonian paths and cycles. A path or cycle that touches every vertex of a graph is called "Hamiltonian" in honor of W. R. Hamilton, who began to ponder and publicize such questions shortly after discovering the quaternions. Hamilton was fascinated by Platonic solids such as the icosahedron, with its 20 triangular faces; and he introduced what he called the Icosian Game, based on paths that go from face to face in that solid. Equivalently (see Fig. 121), his game was based on paths from vertices to vertices along the edges of a dodecahedron.



(a) Icosahedron; (b) Dodecahedron.

Fig. 121. The icosahedron and dodecahedron, whose vertices, edges, and faces define "dual" planar graphs: The faces of one solid correspond to the vertices of the other. (The vertices have been named with two-digit codes that are discussed in exercise 3.)

It's convenient to redraw Fig. 121(b) as three concentric rings, without crossing edges, as shown in Fig. 122(a). Then it's easy to find a Hamiltonian cycle, such as the one indicated by bold edges in Fig. 122(b). (In fact, Hamilton proved that *every* such cycle on the dodecahedron is essentially the same as this one; see exercise 9.) Thus we can also redraw the dodecahedron's graph as shown in Fig. 122(c). From that diagram it's *obviously* Hamiltonian — that is, it obviously has a spanning cycle; but it's not obviously planar at first glance.

concentric rings
Hamilton
spanning cycle
chords
3-regular graph
trivalent graphs
cubic graph
NP
Ham paths, history of–
Graeco-Roman icosahedra
Greek alphabet
Michon
Louvre
Perdrizet
British Museum
author

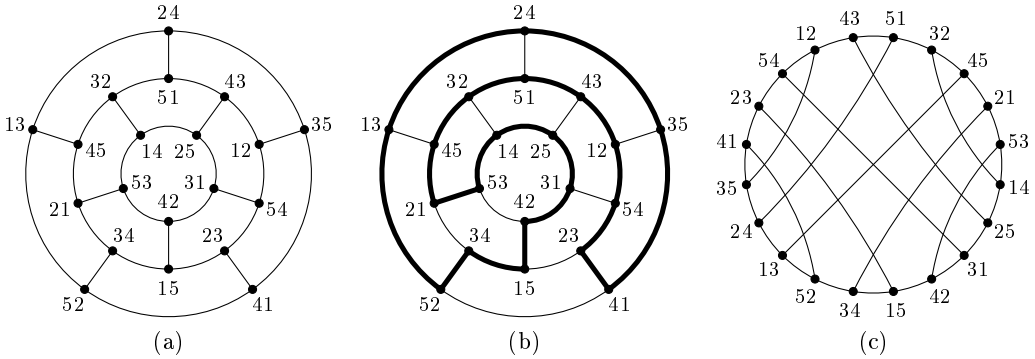


Fig. 122. Alternative views of a dodecahedron's vertices and edges.

Every Hamiltonian graph can clearly be drawn as a great big cycle, together with "chords" between certain pairs of vertices that aren't neighbors in the cycle. Thus a 3-regular graph can be specified compactly by listing only a third of its edges, if it is Hamiltonian and we number its vertices according to one of its cycles. (On the other hand, many trivalent graphs are *not* Hamiltonian. In fact, the task of deciding whether or not a given cubic graph is Hamiltonian turns out to be NP-complete; see exercise 14.)

Hamiltonian paths in antiquity. Let's take a moment to discuss the rich history of the subject before we consider techniques by which Hamiltonian paths and cycles can be found. A strong case can actually be made for the assertion that questions of this kind represent the birth of graph theory, in the sense that they were the first nontrivial graph problems to be investigated.

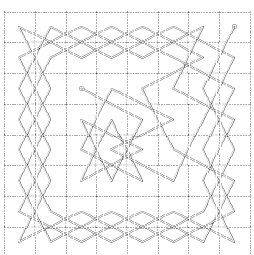
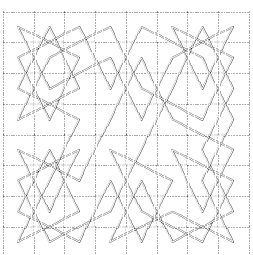
For example, museums in many parts of the world contain specimens of ancient icosahedral objects whose 20 faces are inscribed with the first twenty letters of the Greek alphabet. In most of these cases the alphabetical sequence A, B, Γ, Δ, ..., T, Y on such artifacts forms a Hamiltonian path between adjacent triangles. [E. Michon, in *Bulletin de la Société nationale des Antiquaires de France* (1897), 310 and (1904), 327–329, described an example in the Louvre, catalog number N 1532; P. Perdrizet, in *Bulletin de l'Institut français d'archéologie orientale* **30** (1930), 1–16, illustrated several others.]

In 2015, curators of the Egyptian antiquities at the British Museum kindly allowed the author to inspect the four icosahedra in their collection (EA 29418, EA 49738, EA 59731, EA 59732), of which the first three are Hamiltonian. The experience of rotating them by hand, slowly and systematically according to

alphabetic order, turned out to be unexpectedly delightful. Here are views of the largest one, EA 49738, centered at each of its twelve vertices:

It is made of steatite, 5.8 centimeters in diameter and 228 grams in weight, and was acquired in 1911. A similar example, smaller and with more beautiful letterforms, is object number UC 59254 in the nearby Petrie Museum of University College London [see W. M. F. Petrie, *Objects of Daily Use* (1927), #288]. What a pleasant coincidence that W. R. Hamilton himself would independently come up with the same concept some 1800 years later, and would proceed to find a closed cycle instead of just a path!

Now fast forward to the ninth century, when Hamiltonian paths and cycles of quite a different kind came into play. The game of Chaturanga or Shaṭranj — a predecessor of chess, having different rules for certain pieces, but with knights moving just as they do today — was becoming popular in Asia. And in A.D. 842 the current world champion, al-'Adlī ar-Rūmī, published a book about Shaṭranj. Complete copies of that work are lost; but we know from a subsequent treatise by Abū Zakarīyā Yaḥyā ibn Ibrāhīm al-Ḥakīm that al-'Adlī had presented a closed *knight's tour*: a Hamiltonian cycle on the chessboard. That same treatise also recorded an "open" knight's tour (a Hamiltonian path that can't be completed to a cycle), which was credited to an otherwise unknown author Ibn Manī'.

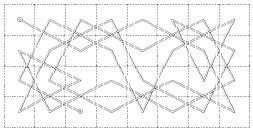
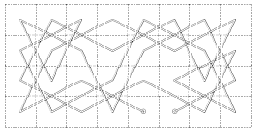
Ibn Manī':  al-'Adlī:  (1)

[See H. J. R. Murray, *A History of Chess* (Oxford, 1913), 175–176, 336.] These remarkable constructions are the earliest known solutions to what was destined to become a classic combinatorial problem. It seems likely that the first path was discovered before the first cycle, because there are so many more of the former.

alphabetic order
Pi, as written in Greece
University College London
Petrie
coincidence
Hamilton
reentrant knight's tour, see closed
Chaturanga
Shaṭranj
knights
al-'Adlī ar-Rūmī
Abū Zakarīyā Yaḥyā
knight's tour
Ibn Manī'
open versus closed tour
closed versus open tour
Murray

Rudraṭa
Ratnākara
slokas
fractured English

Remarkably, knight's tours on *half* of a chessboard, 4×8, had been published even earlier, by Kashmiri poets who were famous for their wordsmithing skills:

Rudraṭa (c. 815)  Ratnākara (c. 830)  (2)

Two copies of Rudraṭa's half-tour will make an open tour on the full board. And two copies of Ratnākara's will make a closed tour, if we rotate one copy by 180°.

Sanskrit poems traditionally consisted of verses called *slokas*, containing 32 syllables each. Here is sloka number 15 in chapter 5 of Rudraṭa's *Kāvyālaṅkāra*:

सेना लीलीलीना नाली लीनाना नानालीलीली ।
नालीनालीले नालीना लीलीली नानानानाली ॥१५॥

senā līlīlīnā nālī līnānā nānālīlīlī
nālīnālīle nālīnā līlīlī nānānānālī [15]

This enigmatic text, which speaks of military leadership, sounds almost like gibberish. But it cleverly represents a knight's tour, in the same way that his sloka 14 had represented a rook's tour: *When we read those 32 syllables in order of the left tour in* (2)*, we get exactly the same words!*

More precisely, consider the following two 4×8 arrays of syllables σ_j:

$$\begin{array}{cccccccc}\sigma_1&\sigma_{30}&\sigma_9&\sigma_{20}&\sigma_3&\sigma_{24}&\sigma_{11}&\sigma_{26}\\ \sigma_{16}&\sigma_{19}&\sigma_2&\sigma_{29}&\sigma_{10}&\sigma_{27}&\sigma_4&\sigma_{23}\\ \sigma_{31}&\sigma_8&\sigma_{17}&\sigma_{14}&\sigma_{21}&\sigma_6&\sigma_{25}&\sigma_{12}\\ \sigma_{18}&\sigma_{15}&\sigma_{32}&\sigma_7&\sigma_{28}&\sigma_{13}&\sigma_{22}&\sigma_5\end{array}\qquad\begin{array}{cccccccc}\sigma_1&\sigma_2&\sigma_3&\sigma_4&\sigma_5&\sigma_6&\sigma_7&\sigma_8\\ \sigma_9&\sigma_{10}&\sigma_{11}&\sigma_{12}&\sigma_{13}&\sigma_{14}&\sigma_{15}&\sigma_{16}\\ \sigma_{17}&\sigma_{18}&\sigma_{19}&\sigma_{20}&\sigma_{21}&\sigma_{22}&\sigma_{23}&\sigma_{24}\\ \sigma_{25}&\sigma_{26}&\sigma_{27}&\sigma_{28}&\sigma_{29}&\sigma_{30}&\sigma_{31}&\sigma_{32}\end{array}\qquad(3)$$

The subscripts on the left correspond to the first sequence of knight moves in (2), while the subscripts on the right have their natural order. Rudraṭa composed a verse with the amazing property that both arrays agree (with $\sigma_1=\sigma_1$, $\sigma_{30}=\sigma_2$, $\sigma_9=\sigma_3$, $\ldots$, $\sigma_5=\sigma_{32}$), by choosing $\sigma_1=$ से, $\sigma_2=$ ना, $\sigma_3=\sigma_4=\sigma_5=$ ली, etc.

Notice that the constraints forced him to use at most four different symbols, thereby throwing away most of the tour's structure. It turns out, in fact, that there are *two* knight's tours consistent with his sloka. Therefore nobody knows whether he was thinking of the tour in (2) and (3) or the tour in exercise 36(i).

Thousands of 4×8 knight's tours are possible, and if Rudraṭa had known more of them he could have written a much less ambiguous sloka that had twelve distinct syllables. For example, a "fractured English" verse that describes such a poet-friendly tour might go like this (see exercise 36(ii)):

Want a good, good time, lots of fun?
Now not time so good; now not time.
Foo. Ah, so! So now fun is lost.
Time not now good, so time not now. (4)

Ratnākara came up with a better idea a few years later. For his tour, illustrated at the right of (2), he composed two *different* slokas, both of which made sense as part of his overall poem. Their syllable patterns

$$\begin{array}{cccccccc}\sigma_{26}&\sigma_{11}&\sigma_{24}&\sigma_5&\sigma_{20}&\sigma_9&\sigma_{30}&\sigma_7\\ \sigma_{23}&\sigma_4&\sigma_{27}&\sigma_{10}&\sigma_{29}&\sigma_6&\sigma_{19}&\sigma_{16}\\ \sigma_{12}&\sigma_{25}&\sigma_2&\sigma_{21}&\sigma_{14}&\sigma_{17}&\sigma_8&\sigma_{31}\\ \sigma_3&\sigma_{22}&\sigma_{13}&\sigma_{28}&\sigma_1&\sigma_{32}&\sigma_{15}&\sigma_{18}\end{array}\qquad\begin{array}{cccccccc}\sigma_1&\sigma_2&\sigma_3&\sigma_4&\sigma_5&\sigma_6&\sigma_7&\sigma_8\\ \sigma_9&\sigma_{10}&\sigma_{11}&\sigma_{12}&\sigma_{13}&\sigma_{14}&\sigma_{15}&\sigma_{16}\\ \sigma_{17}&\sigma_{18}&\sigma_{19}&\sigma_{20}&\sigma_{21}&\sigma_{22}&\sigma_{23}&\sigma_{24}\\ \sigma_{25}&\sigma_{26}&\sigma_{27}&\sigma_{28}&\sigma_{29}&\sigma_{30}&\sigma_{31}&\sigma_{32}\end{array}\qquad(5)$$

would have allowed him to define a tour quite precisely using 32 *distinct* syllables. (See his *Haravijaya*, Chapter 43, slokas 145 and 146.) For example, here's an English rendition of his two-sloka scheme:

poetic license
Rudraṭa
Ratnākara
Bhoja
Deśika
Someshvara III
nonsense verse
doggerel
Yaḥyā ibn Ibrāhīm
abjad numerals

> Have some fun, watch this or that word—
> Great four lines, take out, each gives eight.
> Left; then two black; and just here white.
> Three rook steps make one knight move, right?
>
> One, two, three, four! Watch each word here;
> Or take some left steps and move eight.
> Just right gives this black rook great fun,
> Then have lines make out that white knight. (6)

We obtain the second verse by reading the first verse in knight's tour order, starting at the fifth syllable of the fourth line. (Ratnākara actually used only 24 different syllables. Furthermore, his choices for σ_5 and σ_{32} did not agree in the two slokas; this may be due to errors in transmission of the ancient text, or to "poetic license." In any case his remarkable poem clearly defined a knight's tour.)

Such wordplay had many devotees in medieval India. For example, Rudraṭa's tour of (2) was rendered in Ratnākara's two-sloka style by King Bhoja in his *Sarasvatī-kaṇṭhābharaṇa* (c. 1050), slokas 2.306 and 2.308; also by Vedānta Deśika in his devotional hymn *Pādukāsahasra* (1313), slokas 929 and 930.

A simpler scheme, capable of encoding knight's tours on the full 8×8 board, was used in slokas 5.623–632 of the encyclopedic Sanskrit work *Mānasollāsa* by King Someshvara III (c. 1130). He named each square of the board systematically by combining a consonant for the column with a vowel for the row; then an arbitrary tour was a nonsense verse of 64 syllables, which could be memorized if you wanted to impress your friends. For example, in English we could use the names

$$\begin{bmatrix} \text{bah} & \text{bay} & \text{bee} & \text{boe} & \text{boo} & \text{buh} & \text{bai} & \text{bao} \\ \text{dah} & \text{day} & \text{dee} & \text{doe} & \text{doo} & \text{duh} & \text{dai} & \text{dao} \\ \text{fah} & \text{fay} & \text{fee} & \text{foe} & \text{foo} & \text{fuh} & \text{fai} & \text{fao} \\ \text{hah} & \text{hay} & \text{hee} & \text{hoe} & \text{hoo} & \text{huh} & \text{hai} & \text{hao} \\ \text{lah} & \text{lay} & \text{lee} & \text{loe} & \text{loo} & \text{luh} & \text{lai} & \text{lao} \\ \text{mah} & \text{may} & \text{mee} & \text{moe} & \text{moo} & \text{muh} & \text{mai} & \text{mao} \\ \text{nah} & \text{nay} & \text{nee} & \text{noe} & \text{noo} & \text{nuh} & \text{nai} & \text{nao} \\ \text{sah} & \text{say} & \text{see} & \text{soe} & \text{soo} & \text{suh} & \text{sai} & \text{sao} \end{bmatrix} \tag{7}$$

to encode Someshvara's tour as the following (memorable?) quatrain:

> Sah nee soo nai lao fai bao duh, foe boo dee bah fay lah nay soe;
> nuh sao mai hao dai huh doo bee, dah hay mah say noe suh nao lai.
> Fao bai fuh dao buh doe bay fah, lay nah see noo sai mao hai foo?
> Luh moe hee loo mee hoe moo lee, hoo fee boe day hah may loe muh. (8)

Incidentally, Yaḥyā ibn Ibrāhīm had presented the two knight's tours in (1) by first stating two 64-word poems in Arabic, then copying the words of those poems into 8×8 diagrams, according to the knight's paths. Then he repeated the right-hand tour, using the Arabic words "first," "second," ..., together with Persian-style abjad numerals, in place of the words of the corresponding

John Rylands Library
Path diagrams
dalla Volpe
greedy
heuristic
greedy algorithms
Warnsdorf–

poem. [His work is preserved in a rare manuscript belonging to the John Rylands Library in Manchester: *Arabic MS. 766*, folio 39.] The latter convention, which corresponds to

60	11	56	7	54	3	42	1
57	8	59	62	31	64	53	4
12	61	10	55	6	41	2	43
9	58	13	32	63	30	5	52
34	17	36	23	40	27	44	29
37	14	33	20	47	22	51	26
18	35	16	39	24	49	28	45
15	38	19	48	21	46	25	50

(9)

in decimal notation, has been used by many subsequent authors to characterize particular knight's tours in an easy-to-understand way. Path diagrams such as (1) and (2), which provide complementary insights, weren't invented until much later, when Lelio dalla Volpe published a short book *Corsa del Cavallo per tutt' i scacchi dello Scacchiere* (Bologna, 1766), containing nineteen examples.

A greedy heuristic. Early in the 1800s, the knight's tour problem inspired an important new approach to combinatorial problems, based on making a sequence of locally optimum decisions. Such techniques, now known as "greedy algorithms," were unheard-of at the time. But H. C. von Warnsdorf, a high court official in Hesse who had challenged himself by spending many nights trying to construct long paths of a knight, hit on a simple idea that worked like magic: *At each step, move to a place that has the fewest remaining exits.* This principle has become famous as "Warnsdorf's rule."

For example, suppose we want to construct an open knight's tour on a 5×5 board, starting in a corner. Numbering the cells ij for $0 \le i, j < 5$, we can assume by symmetry that the first two steps are 00 — 12. From cell 12 we can move the knight to either 04, 24, 33, 31, or 20, from which it could then exit in either 1, 3, 3, 3, or 3 ways; Warnsdorf's rule tells us to choose 04, because $1 < 3$. (Indeed, this is our last chance to visit 04, unless the tour will end at that cell.) After 04 the knight must proceed to 23; and again we have five choices, namely 44, 42, 31, 11, or 02. The rule takes us to 44, then 32; then to 40, then 21; and we've completed a partial tour that looks like this:

1	*3*	*2*	*3*	**3**
3	*2*	**2**	*2*	*3*
2	**8**	*8*	**4**	*2*
3	*2*	**6**	*2*	*3*
7	*3*	*2*	*3*	**5**

(**Bold** numbers are the visited cells. *Italicized* numbers tell how many exits remain from the unvisited cells.) (10)

Four cells are now candidates for step **9**, and they're all currently marked '*2*'. So there's a four-way tie. In such cases, von Warnsdorf explicitly said that it's OK to choose arbitrarily, among all cells that have the fewest exits. Let us therefore proceed boldly to cell 33 (between **4**, **5**, and **6**). That makes a two-way tie; and we might as well go next to 41 (just to the right of **7**). From here we *don't* want to go to the middle square, which has just dropped from *8* to *7*, because our

other choice is a *1*. And now it's plain sailing, as von Warnsdorf leads us on a merry chase—ending gloriously with move **25** in the center cell 22.

SGB format
linear time
greed
seven deadly sins
morality
virtue

It's easy to implement Warnsdorf's rule, by representing the given graph in SGB format. (The reader should be familiar with this format; see, for example, Algorithm 7B and the remarks that precede it.) The node for each vertex v in Algorithm W below extends the basic format by including two utility fields, DEG(v) and TAG(v), which correspond to the *italic* and **bold** numbers in (10).

Algorithm W allows the user to specify "target" vertices $t_1, \ldots, t_r$, which are to be visited only when no other vertices are available. A similar mechanism was, in fact, used by von Warnsdorf himself, in the advanced examples of his original booklet that introduced the idea [*Des Rösselsprunges einfachste und allgemeinste Lösung* (Schmalkalden, 1823); see also *Schachzeitung* **13** (1858), 489–492].

Algorithm W (*Warnsdorf's rule*). Given a graph G, a source vertex s, and optional target vertices $t_1, \ldots, t_r$, this algorithm applies Warnsdorf's rule to find a (hopefully Hamiltonian) path $v_1, v_2, \ldots$ that begins with s. Let $n = \texttt{N}(G)$ be the number of vertices of G; let $v_0 = \texttt{VERTICES}(G)$ be G's initial vertex in memory.

W1. [Initialize.] For $0 \le k < n$ and $v \leftarrow v_0 + k$, do the following: Set $d \leftarrow 0$, $a \leftarrow \texttt{ARCS}(v)$; while $a \neq \Lambda$, set $d \leftarrow d + 1$ and $a \leftarrow \texttt{NEXT}(a)$; then set $\texttt{DEG}(v) \leftarrow d$ and $\texttt{TAG}(v) \leftarrow 0$. (Thus DEG($v$) is the degree of v.) Finally set $k \leftarrow 0$, $v \leftarrow s$, and $\texttt{DEG}(t_i) \leftarrow \texttt{DEG}(t_i) + n$ for $1 \le i \le r$.

W2. [Visit v.] Set $k \leftarrow k + 1$, $v_k \leftarrow v$, $\texttt{TAG}(v) \leftarrow k$, $a \leftarrow \texttt{ARCS}(v)$, and $\theta \leftarrow 2n$.

W3. [All arcs tested?] If $a = \Lambda$, go to W7. Otherwise set $u \leftarrow \texttt{TIP}(a)$, and go to W6 if $\texttt{TAG}(u) \neq 0$. (Vertex u is a neighbor of v_k and a candidate for v_{k+1}.)

W4. [Decrease DEG(u).] Set $t \leftarrow \texttt{DEG}(u) - 1$ and $\texttt{DEG}(u) \leftarrow t$.

W5. [Is DEG(u) smallest?] If $t < \theta$, set $\theta \leftarrow t$ and $v \leftarrow u$.

W6. [Loop over arcs.] Set $a \leftarrow \texttt{NEXT}(a)$ and return to W3.

W7. [Done?] If $\theta = 2n$, terminate with path $v_1 \ldots v_k$. Otherwise go to W2. ∎

Notice that the candidates for v_{k+1} are precisely the vertices u whose DEG needs to change when v_k leaves the active graph. Therefore this algorithm runs in linear time: Every arc is examined at most twice, once in step W1 and once in step W3.

The path chosen by Algorithm W depends on the ordering of arcs that lead out of each vertex in SGB format, because Warnsdorf's rule makes an arbitrary decision in case of ties. A simple change to step W5 will randomize the path properly, as if all orderings of the arcs were equally likely (see exercise 53).

Now that we understand Warnsdorf's rule, let's talk a little bit about greed. Greed is of course one of the seven deadly sins; hence we might well question the morality of ever using a greedy algorithm in our own work. However, greed is actually a *virtue*, when it enhances the environment and harms nobody.

In what sense is Algorithm W greedy? From the standpoint of *short-term* greed, also known as "instant satisfaction," the best choice for v_{k+1} would seem to be a vertex with *maximum* degree, not minimum, because that vertex will give us the most flexibility when choosing v_{k+2}. But from the standpoint of *long-term* greed, also known as "risk management" or maximizing our chance

de Jaenisch
Hamilton
dodecahedron graph
3-regular
n-cube
Gray binary codes
all_perms
permutations
change ringing
binary
binary trees
rotation
NP-complete
Pohl

of success, it's best to choose a vertex with *minimum* degree, as von Warnsdorf stipulated; that choice leaves us with the most arcs remaining for moves in the future. Indeed, short-term greed turns out to be very bad (see exercise 59).

How good is Warnsdorf's rule? It works so well for knight moves that von Warnsdorf naïvely believed it to be infallible, except perhaps on $m \times n$ boards with $m < 6$ or $n < 6$. He even thought that he had a proof of guaranteed success. His booklet exhibited many examples: 6×6, 6×7, ..., up to 10×10. Experiments by C. F. de Jaenisch [*Traité des applications de l'analyse mathématique au jeu des échecs* **2** (1862), 59] showed in fact that, on an ordinary 8×8 chessboard, one can basically choose the first 40 moves at random, and obtain a complete knight's tour by applying Warnsdorf's rule only to the last 24 steps!

The rule can fail, however. On a 6×6 board, it gives a complete tour about 97.2% of the time, yet it sometimes stops after only 32 or 34 steps if the starting position is one of the eight interior diagonal squares. On the 8×8 board it succeeds even more often (about 97.9%). Yet with probability 0.0000038 it might stop with a path of length 39, as shown in the answer to exercise 59.

Hamilton's dodecahedron graph (Fig. 122) is quite different from a graph of knight moves, because it is 3-regular. A partial path in a 3-regular graph can be extended in at most two ways, after we've selected the first two points, while a knight can have up to seven choices at every step. (Furthermore, all starting edges of the dodecahedron are equivalent.) Nevertheless, Algorithm W handles that graph well: It finds a Hamiltonian path $v_1v_2\ldots v_{20}$ with probability $\frac{31}{32} = .96875$. Furthermore, it finds a path with v_{20} — v_1 (hence a Hamiltonian cycle) with probability $\frac{15}{128} \approx .117$. That probability rises to $\frac{139}{256} \approx .543$ if we set t_1 to a neighbor of s; it's exactly $1/2$ if we set $\{t_1, t_2, t_3\}$ to the *three* neighbors of s.

It's not difficult to see that Algorithm W always works perfectly when G is the graph of a rectangular grid and s is a corner vertex (see exercise 62). With a bit more thought, we can even prove that it always succeeds when G is an n-cube, thereby finding many examples of the generalized Gray binary codes that we studied in Section 7.2.1.1 (see exercise 63). When G is the SGB graph *all_perms*$(5, 0)$ — whose vertices are the permutations of $\{0, 1, 2, 3, 4\}$, related by swapping adjacent digits — Warnsdorf's rule finds "change ringing" paths of length $5! - 1 = 119$ about 29% of the time. (See Algorithm 7.2.1.2P. This probability drops to less than 2%, however, with permutations of 6 elements, and to near zero with permutations of 7.) Another instructive example is the SGB graph *binary*$(10, 0, 0)$, whose vertices are the 16796 binary trees with 10 nodes, related by "rotation." Starting at the tree with all-null left links, Algorithm W finds a Hamiltonian path about 5.6% of the time. (See Algorithm 7.2.1.6L.)

Of course Algorithm W isn't a panacea. We can't expect any algorithm to solve the NP-complete Hamiltonian path problem in linear time! Warnsdorf's rule certainly has difficulty in critical cases; indeed, it can fail spectacularly even on small graphs (see exercise 65). But it's often a good first thing to try, when presented with a graph that we haven't seen before.

Ira Pohl [*CACM* **10** (1967), 446–449; **11** (1968), 1] has suggested breaking ties in Warnsdorf's rule by looking at the *sum of the degrees* of v_k's neighbors.

path flipping–
flipping paths–
path exchange, see path flipping
Euler
Bertrand
breedy
mutations
genetic algorithms
closed

Path flipping. Long before Warnsdorf's time, the great mathematician Leonhard Euler had already published a classic paper about knight's tours [*Mémoires de l'académie des sciences de Berlin* **15** (1759), 310–337], in which he showed how to discover long paths by a completely different method. (Euler credited this idea, at least in part, to his friend Louis Bertrand.) Instead of Warnsdorf's "greedy" algorithm, his approach might be called a "breedy" method, because it proceeded by simple mutations and adaptations of paths already known.

Suppose, for example, that we want to find a 3×10 knight's tour, and that Warnsdorf has already told us how to reach 28 of the 30 cells:

4	7	2	27	24	13	10	19	a	17
1	28	5	14	9	22	25	16	11	20
6	3	8	23	26	15	12	21	18	b

. (11)

We can't go from position 28 to an unvisited cell; but we needn't despair, because 28 is just one knight's move away from cell 23. Similarly, cell 1 is adjacent to 8. Therefore we can immediately deduce that two more equally long paths exist:

$$1\,..\,23, 28\,..\,24; \qquad 7\,..\,1, 8\,..\,28. \tag{12}$$

(Here '$x\,..\,y$' stands for the path from x to y that proceeds by unit steps ± 1.) Operating in the same fashion on the first of these yields three more,

$$1\,..\,5, 24\,..\,28, 23\,..\,6; \quad 1\,..\,15, 24\,..\,28, 23\,..\,16; \quad 7\,..\,1, 8\,..\,23, 28\,..\,24. \tag{13}$$

And, aha, one of these can be extended to a full tour $1\,..\,15, 24\,..\,28, 23\,..\,16, \mathrm{b}, \mathrm{a}$:

4	7	2	19	16	13	10	25	30	27
1	20	5	14	9	22	17	28	11	24
6	3	8	21	18	15	12	23	26	29

. (14)

Now the same subpath-flipping technique leads from (14) to additional tours

$$1\,..\,17, 30\,..\,18; \qquad 1\,..\,23, 30\,..\,24; \qquad 7\,..\,1, 8\,..\,30; \tag{15}$$

and we can continue to find tours galore:

$$1\,..\,13, 18\,..\,30, 17\,..\,14; \quad 1\,..\,5, 18\,..\,30, 17\,..\,6; \quad 7\,..\,1, 8\,..\,17, 30\,..\,18;$$
$$7\,..\,1, 8\,..\,23, 30\,..\,24; \quad 13\,..\,8, 1\,..\,7, 14\,..\,30; \quad 1\,..\,7, 14\,..\,17, 30\,..\,18, 13\,..\,8;$$

etc. Indeed, the latter is a Hamiltonian *cycle* — a *closed* tour — because 1 is adjacent to 8! A Hamiltonian cycle represents 30 different Hamiltonian *paths*, each of which leads to further flips, hence further paths and cycles.

If we start with (14) and keep flipping until no new paths arise, it turns out that we will have discovered all 16 of the Hamiltonian cycles of the 3×10 knight graph, as well as 2472 of its 2568 noncyclic Hamiltonian paths.

One of the 96 noncyclic Hamiltonian paths *not* derivable from (14) is

1	12	3	22	15	10	7	26	29	18
4	23	14	11	6	25	20	17	8	27
13	2	5	24	21	16	9	28	19	30

. (16)

It leads via flips only to three others, namely to $1\,..\,17, 30\,..\,18$; $13\,..\,1, 14\,..\,30$; $13\,..\,1, 14\,..\,17, 30\,..\,18$. We *wouldn't* have found a cycle, if we'd started with (16).

Let's formulate Euler's approach more precisely:

Euler
breadth-first search
update
canonical form
breadth-first search

Algorithm F (*Long paths by flipping*). Given a simple path $v_1 — v_2 — \cdots — v_t$ in a connected n-vertex graph, this algorithm repeatedly obtains new paths by reversing subpaths as explained above, until either exhausting all possibilities or finding a path that can be extended by a vertex $\notin \{v_1, v_2, \ldots, v_t\}$. An auxiliary table of vertex labels $w[v]$ is used to discover potential flips.

F1. [Initialize for breadth-first search.] Prepare a dictionary, initially empty, for storing paths of length $t-1$. Set $w[v] \leftarrow 0$ for each of the n vertices v of the graph. Set $q \leftarrow 0$ and perform *update*$(v_1, \ldots, v_t)$, where *update* is the subroutine defined below. Then set $d \leftarrow p \leftarrow p_1 \leftarrow p_2 \leftarrow 0$ and $p_0 \leftarrow q$.

F2. [Done with distance d?] (At this point we've entered q paths into the dictionary, and we've explored the successors of the first p paths. Exactly p_i of those paths were obtained by making $\le d-i$ flips, for $0 \le i \le 2$.) Go to F6 if $p = p_0$; otherwise set $p \leftarrow p+1$.

F3. [Explore path p.] Let $u_1 — u_2 — \cdots — u_t$ be the pth path that entered the dictionary, and set $w[u_k] \leftarrow k$ for $1 \le k \le t$. Go to F5 if $u_t — u_1$.

F4. [Process a noncyclic path.] For each vertex v such that $u_t — v$, do the following: Set $k \leftarrow w[v]$; terminate the algorithm if $k = 0$; otherwise call *update*$(u_1, \ldots, u_k, u_t, \ldots, u_{k+1})$. Then, for each v such that $u_1 — v$, do the following: Set $k \leftarrow w[v]$; terminate if $k = 0$; otherwise *update*$(u_{k-1}, \ldots, u_1, u_k, \ldots, u_t)$. Then return to F2.

F5. [Process a cyclic path.] (A cyclic path will be in the dictionary only if $t = n$; see below.) For $1 \le j \le t$ and for each v such that $u_j — v$, do the following: Set $k \leftarrow w[v]$ (which will be positive). If $k < j$, call *update*$(u_{j+1}, \ldots, u_t, u_1, \ldots, u_k, u_j, \ldots, u_{k+1})$ and *update*$(u_{k-1}, \ldots, u_1, u_t, \ldots, u_j, u_k, \ldots, u_{j-1})$; otherwise *update*$(u_{j+1}, \ldots, u_k, u_j, \ldots, u_1, u_t, \ldots, u_{k+1})$ and *update*$(u_{k-1}, \ldots, u_j, u_k, \ldots, u_t, u_1, \ldots, u_{j-1})$. Then return to F2.

F6. [Advance d.] Terminate if $p = q$ (we have found all the reachable paths). Otherwise set $d \leftarrow d+1$, $p_2 \leftarrow p_1$, $p_1 \leftarrow p_0$, $p_0 \leftarrow q$, and go back to F2. ∎

Algorithm F relies on a subroutine '*update*$(v_1, \ldots, v_t)$', whose purpose is to put the path $v_1 — \cdots — v_t$ into the dictionary unless it's already there. First the path is converted to a canonical form, so that equivalent paths are entered only once: If $v_t \not— v_1$, the canonical form is obtained by changing $(v_1, \ldots, v_t) \leftarrow (v_t, \ldots, v_1)$ if $v_1 > v_t$. On the other hand if $v_t — v_1$, the path is cyclic, and we terminate the algorithm if $t < n$. (The graph is connected, so there must be a vertex outside the cycle that is adjacent to a vertex of the cycle.) Finally, if $t = n$ and $v_n — v_1$, we obtain the canonical form by permuting the cycle cyclically so that v_1 is the smallest element; then we set $(v_1, v_2, \ldots, v_n) \leftarrow (v_1, v_n, \ldots, v_2)$ if $v_2 > v_n$. Once $(v_1, \ldots, v_t)$ is in canonical form, the *update* routine looks for it in the dictionary. If unsuccessful, *update* sets $q \leftarrow q+1$ and inserts it as the qth path.

The theory of breadth-first search tells us that $(v_1, \ldots, v_t)$ cannot match any path in the dictionary that was obtained with fewer than $d-1$ flips. (Otherwise the path $(u_1, \ldots, u_t)$ that led to it would have been seen before making d flips.)

Therefore step F6 can save dictionary space and lookup time by deleting all paths of index $\le p_2$ from the dictionary whenever p_2 increases. Exercise 73 discusses a simple trick that makes this deletion painless.

Algorithm F is amazingly versatile. For example, there are 9862 closed knight's tours on a 6×6 board, and 2963928 open tours. All of them will be found by Algorithm F, when given any single instance.

We began our search for a 3×10 knight's cycle by using the Warnsdorf-inspired path (11). But we could have started Algorithm F with $t = 1$, thus presenting it with only a single vertex v_1. Every time the algorithm finds a larger path, we can simply restart it, with t increased.

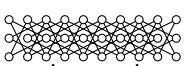
For example, the author tried the 3×10 problem 100 times, choosing v_1 at random and ordering the vertex neighbors randomly in steps F4 and F5. A Hamiltonian cycle was found in 82 cases, usually after making fewer than 100 calls on *update*. A stubborn Hamiltonian path like (16) was found in 6 cases. And the remaining 12 cases failed to reach $t = 30$; once t was even stuck at 22.

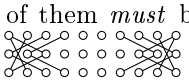
Of course that's a very small problem. When presented with the graph of permutations of $\{0, 1, 2, 3, 4, 5\}$, Algorithm F was able to find a "change ringing" cycle of length 720 in each of ten random trials, averaging less than 50,000 updates per trial. On the other hand it did *not* do well when trying to find a closed 3×100 knight's tour.

Searching exhaustively. Let's try now to design an algorithm that systematically finds *every* Hamiltonian cycle of a given graph. Such an algorithm will also find every Hamiltonian *path*, because exercise 2 shows that every Hamiltonian path of G corresponds to a Hamiltonian cycle of a related graph G'.

A Hamiltonian cycle involves every vertex. So we can start it at any convenient vertex v_1. Then there's an obvious way to grow all possible cycles via backtracking: For each v_2 with $v_1 — v_2$, we consider each $v_3 \ne v_1$ with $v_2 — v_3$, etc. We can also formulate the task as an XCC problem (see, for example, the "prime queen attacking problem" in Section 7.2.2.3).

But those approaches are overly specific; there's usually a much more efficient way to proceed. Instead of regarding our task as the assignment of appropriate labels 1, 2, ..., n to the n vertices of our graph, it's better to regard it as the task of choosing n edges in such a way that (i) every vertex is an endpoint of precisely two of those edges; (ii) the subgraph defined by those edges is connected. Indeed, a Hamiltonian cycle is nothing more nor less than an (unordered) set of n edges that form a single cycle.

Consider, for example, the 3×10 knight graph,  , which has 30 vertices and 50 edges. We can conveniently denote its vertices by two-digit numbers, $\{00, 01, \ldots, 09, 10, 11, \ldots, 19, 20, 21, \ldots, 29\}$, and write its edges compactly in two-line form: $\begin{smallmatrix} 00 & 00 & 01 & & & 17 & 17 & 18 & 19 \\ 12, & 21, & 20, & \cdots, & & 25, & 29, & 26, & 27. \end{smallmatrix}$

Notice that vertices 00, 09, 10, 11, 18, 19, 20, and 29 have degree 2 in this graph. Consequently the two edges that touch each of them *must* be present in any Hamiltonian cycle; in other words, the pattern  is already forced, before we begin to choose further edges.

Warnsdorf
permutations
change ringing
all Hamiltonian cycles
all Hamiltonian paths
backtracking
XCC problem
prime queen attacking problem
knight graph

In this pattern, vertex 12 belongs to the edges $\binom{00}{12}$ and $\binom{12}{20}$, which were forced from vertices 00 and 20. So 12 already has the two edges that it needs; and the other edges that touch it, namely $\binom{04}{12}$ and $\binom{12}{24}$, cannot ever be part of a Hamiltonian cycle. We might as well delete them. Similarly, we can delete edges $\binom{05}{17}$ and $\binom{17}{25}$.

Nothing else is obviously forced at this point, so we must make a choice. For example, we must use either the edge $\binom{02}{21}$ or the edge $\binom{13}{21}$, because those edges are the only ways for vertex 21 to reach its quota of two. In other words, our search tree for the 3×10 knight graph must begin with a binary branch.

Suppose we choose $\binom{02}{21}$. That edge implies in particular that we now have $01 — 20 — 12 — 00 — 21 — 02 — 10 — 22$ as a subpath of the final cycle. Consequently edges $\binom{13}{21}$, $\binom{02}{14}$, and $\binom{02}{23}$ can no longer be used. Nor can the edge $\binom{01}{22}$ (because it would complete a short cycle!).

Aha. Only one of the remaining edges touches 01, namely $\binom{01}{13}$. So that edge is now forced. And then we're confronted with another two-way branch. Exercise 108 discusses one sequence of reasonable initial choices, and the reader is strongly encouraged to study that scenario.

In general, as we're trying to visit all Hamiltonian cycles of a given graph, we'll have a partial solution consisting of a set of disjoint subpaths to be included, and a set of edges by which those subpaths might be extended until a complete cycle is obtained. The subpaths are defined by the edges that have been chosen so far. If there are t subpaths, $\{u_1 \dots v_1, \dots, u_t \dots v_t\}$, we say that the $2t$ endpoints $\{u_1, v_1, \dots, u_t, v_t\}$ are "outer" vertices; any vertex that lies on a subpath but is *not* an endpoint is called "inner"; and all other vertices are "bare." Every vertex begins bare, and is eventually clothed. If we reach a state where all but two of the vertices are inner, and if those two outer vertices are adjacent, we can complete a Hamiltonian cycle. Success!

Algorithm H below finds Hamiltonian cycles by essentially starting with a graph G and removing edges until only a cycle remains. It uses a sparse-set representation for G, because such structures are an especially attractive way to maintain the current status of a graph that is continually getting smaller.

The idea is to have two arrays, `NBR` and `ADJ`, with one row for each vertex v. If v has $d = \texttt{DEG}(v)$ neighbors in G, they're listed (in any order) in the first d columns of `NBR`$[v]$. And if `NBR`$[v]$`[`k`]` $= u$, where $0 \le k < d$, we have `ADJ`$[v]$`[`u`]` $= k$; in other words, there's an important invariant relation,

$$\texttt{NBR}[v][\texttt{ADJ}[v][u]] = u, \qquad \text{for } 0 \le u < n \text{ and } u — v, \tag{17}$$

where n is the number of vertices in G. Neighbors can be deleted by moving them to the right and decreasing d; neighbors can be undeleted by simply increasing d. Furthermore, if u is *not* a neighbor of v, `ADJ`$[v]$`[`u`]` has the impossible value ∞; thus the `ADJ` array functions also as an adjacency matrix.

The edges $u — v$ of G are considered to be pairs of arcs, $u \longrightarrow v$ and $v \longrightarrow u$, which run in opposite directions. In particular, we always have `ADJ`$[u]$`[`v`]` $= \infty$ if and only if `ADJ`$[v]$`[`u`]` $= \infty$. When an edge is deleted, however, we often need to delete only one of those arcs in the `NBR` array, because Algorithm H doesn't always need to look at both of them.

binary branch: A choice between two possibilitie
outer vertices
inner vertices
bare vertices
sparse-set representation
data structures
invariant relation
adjacency matrix

Algorithm H represents the vertices of G by the integers 0 to $n-1$, as we've seen in (17). Every vertex v has three fields: its current degree, DEG(v); its external name, NAME(v), used only to print the answers; and a special field called MATE(v). We have MATE(u) $= v$ and MATE(v) $= u$ when u and v are the outer vertices at the ends of a current subpath; and we have MATE(v) $= -1$ if and only if vertex v is bare. The value of MATE(v) is undefined when v is an inner vertex, but it must be nonnegative in that case. (See exercise 109.)

An inner vertex is essentially invisible to our algorithm, because we already know its context in the final cycle. We maintain an array VIS to list the visible vertices—those that are either bare or outer. VIS is a sparse-set representation, containing a permutation of the vertices, with the invisible ones listed last. The inverse permutation appears in a companion array called IVIS, so that we have

$$\texttt{VIS}[k] = v \qquad \Longleftrightarrow \qquad \texttt{IVIS}[v] = k. \tag{18}$$

Vertex v is visible if and only if IVIS[v] $<$ S, where S is a global variable. Thus v is inner if and only if IVIS[v] $\ge$ S. Here's how a vertex becomes invisible:

$$\text{makeinner}(v) = \begin{cases} \text{Set } \texttt{S} \leftarrow \texttt{S}-1,\ v' \leftarrow \texttt{VIS}[\texttt{S}],\ k \leftarrow \texttt{IVIS}[v]; \\ \text{set } \texttt{VIS}[\texttt{S}] \leftarrow v,\ \texttt{IVIS}[v] \leftarrow \texttt{S}, \\ \qquad \texttt{VIS}[k] \leftarrow v',\ \texttt{IVIS}[v'] \leftarrow k. \end{cases} \tag{19}$$

If u is a bare vertex whose degree decreases to 2, Algorithm H can make significant progress, because the two remaining edges that touch u must both be part of the cycle. Whenever such a u is discovered, we put it into a "trigger list" called TRIG. A global variable, T, holds the size of the trigger list. This behavior is implemented by using the following procedure to delete the arc $u \longrightarrow v$:

$$\text{remarc}(u,v) = \begin{cases} \text{Set } d \leftarrow \texttt{DEG}(u)-1,\ k \leftarrow \texttt{ADJ}[u][v],\ w \leftarrow \texttt{NBR}[u][d]. \\ \text{If } \texttt{MATE}(u) < 0 \text{ and } d = 2, \text{ set } \texttt{TRIG}[\texttt{T}] \leftarrow u,\ \texttt{T} \leftarrow \texttt{T}+1. \\ \text{Set } \texttt{NBR}[u][d] \leftarrow v,\ \texttt{NBR}[u][k] \leftarrow w, \\ \qquad \texttt{ADJ}[u][v] \leftarrow d,\ \texttt{ADJ}[u][w] \leftarrow k; \\ \text{set } \texttt{DEG}(u) \leftarrow d. \end{cases} \tag{20}$$

One might think that we've now defined a comprehensive set of data structures for implementing Algorithm H; but we aren't done yet. There's also a doubly linked list, maintained in arrays LLINK[v] and RLINK[v] for $0 \le v \le n$, with entries LLINK[n] and RLINK[n] serving as the list head. This list contains all of the current "outer" vertices. More precisely, suppose that there are t subpaths, and suppose that we have RLINK[n] $= v_1$, RLINK[v_j] $= v_{j+1}$ for $1 \le j < 2t$, and RLINK[v_{2t}] $= n$. Then the outer vertices are $\{v_1, v_2, \ldots, v_{2t}\}$; and we also have LLINK[n] $= v_{2t}$, LLINK[v_j] $= v_{j-1}$ for $1 < j \le 2t$, and LLINK[v_1] $= n$. Insertion and deletion are accomplished in the usual way:

$$\text{activate}(v) = \begin{cases} \text{Set } k \leftarrow \texttt{LLINK}[n], \\ \quad \texttt{LLINK}[n] \leftarrow \texttt{RLINK}[k] \leftarrow v, \\ \quad \texttt{LLINK}[v] \leftarrow k,\ \texttt{RLINK}[v] \leftarrow n. \end{cases} \tag{21}$$

$$\text{deactivate}(v) = \begin{cases} \text{Set } j \leftarrow \texttt{LLINK}[v],\ k \leftarrow \texttt{RLINK}[v], \\ \quad \texttt{LLINK}[k] \leftarrow j,\ \texttt{RLINK}[j] \leftarrow k; \\ \text{makeinner}(v). \end{cases} \tag{22}$$

degree
DEG(v)
NAME(v)
MATE(v)
outer vertices
bare
inner vertex
sparse-set representation
visible
invisible
makeinner(v)
trigger list
data structures
doubly linked list
LLINK[v]
RLINK[v]
list head
outer vertices
Insertion
deletion

stack with holes
n-cycle

The algorithm makes frequent use of the following subroutine:

$$\text{makemates}(u,w) = \begin{cases} \text{If } \texttt{ADJ}[w][u] < \texttt{DEG}(w), \\ \quad \text{remarc}(u,w) \text{ and } \text{remarc}(w,u); \\ \text{set } \texttt{MATE}(u) \leftarrow w \text{ and } \texttt{MATE}(w) \leftarrow u. \end{cases} \tag{23}$$

Vertices u and w are becoming endpoints. It removes the edge between u and w, if present, in order to prevent the formation of a short (non-Hamiltonian) cycle.

The current state at each level of the search tree is kept in two sequential stacks. `ACTIVE` is an array that remembers which vertices are outer; `SAVE` is an array that remembers the mates and degrees of the visible vertices. The `SAVE` stack is somewhat unusual because it's a "stack with holes": n slots are allocated to it at every level, but only the slots for visible vertices are actually used.

Level l of the search can in fact involve up to seven state variables: `CV`(l) is the outer vertex on which we're branching; `I`(l) identifies the neighbor of that vertex in the currently chosen edge; `D`(l) is the current degree of `CV`(l); `E`(l) is the number of edges chosen so far; `S`(l) is the number of vertices that are currently visible; `T`(l) and `A`(l) are the current sizes of `TRIG` and `ACTIVE`.

Algorithm H (*All Hamiltonian cycles*). Given a graph G on the n vertices $\{0,1,\ldots,n-1\}$, this algorithm uses the data structures discussed above to visit every subset of n edges that form an n-cycle. During every visit, the chosen edges are `EU`$[k]$ — `EV`$[k]$ for $0 \le k < n$.

H1. [Initialize.] Set up the `NBR` and `ADJ` arrays as described in (17). Set the global variables $a \leftarrow e \leftarrow i \leftarrow l \leftarrow \texttt{T} \leftarrow 0$. Also set `VIS`$[v] \leftarrow$ `IVIS`$[v] \leftarrow v$ and `MATE`$(v) \leftarrow -1$ for $0 \le v < n$. Set `LLINK`$[n] \leftarrow$ `RLINK`$[n] \leftarrow$ `S` $\leftarrow n$. Finally, for every vertex v with `DEG`$(v) = 2$, set `TRIG`$[$`T`$] \leftarrow v$ and `T` $\leftarrow$ `T`$+1$.

H2. [Choose the root vertex.] Let `CURV` be a vertex of minimum degree, and set $d \leftarrow$ `DEG(CURV)` $- 1$. If $d < 1$, terminate (there is no Hamiltonian cycle). If $d = 1$, set `CURV` $\leftarrow -1$ and go to H4.

H3. [Force a root edge.] Set `CURU` $\leftarrow$ `NBR[CURV]`$[d-i]$ (the last yet-untried neighbor of `CURV`), and set `EU[0]` $\leftarrow$ `CURU`, `EV[0]` $\leftarrow$ `CURV`, $e \leftarrow 1$. Then activate(`CURU`), activate(`CURV`), and makemates(`CURU`, `CURV`).

H4. [Record the state.] Set `CV`$(l) \leftarrow$ `CURV`, `I`$(l) \leftarrow i$, `D`$(l) \leftarrow d$, `E`$(l) \leftarrow e$, `S`$(l) \leftarrow$ `S`, `T`$(l) \leftarrow$ `T`. For $0 \le k <$ `S`, set $u \leftarrow$ `VIS`$[k]$ and `SAVE`$[nl+u] \leftarrow$ $\big($`MATE`(u), `DEG`$(u)\big)$ (thereby leaving "holes" in the `SAVE` stack). Then set $u \leftarrow$ `RLINK`$[n]$; while $u \ne n$, set `ACTIVE`$[a] \leftarrow u$, $a \leftarrow a+1$, $u \leftarrow$ `RLINK`$[u]$. Finally set `A`$(l) \leftarrow a$, and go to H6 if $l = 0$.

H5. [Choose an edge.] Set `CURU` $\leftarrow$ `NBR[CURV]`$[i]$, `CURT` $\leftarrow$ `MATE(CURU)`, `CURW` $\leftarrow$ `MATE(CURV)`, `EU`$[e] \leftarrow$ `CURU`, and $e \leftarrow e+1$. If `CURT` < 0 (`CURU` is bare), makemates(`CURU`, `CURW`), activate(`CURU`), and go to H6. Otherwise (`CURU` is outer), makemates(`CURT`, `CURW`). Call remarc$\big($`NBR[CURU]`$[k]$, `CURU`$\big)$ for k decreasing from `DEG(CURU)` $- 1$ to 0. Then deactivate(`CURU`).

H6. [Begin trigger loop.] Set $j \leftarrow 0$ if $l = 0$, else $j \leftarrow$ `T`$(l-1)$. Go to H10 if $j =$ `T`.

H7. [Clothe TRIG[j].] Set $v \leftarrow$ TRIG[j], and go to H9 if MATE(v) ≥ 0 (v is no longer bare). Otherwise go to H15 if DEG(v) < 2 (Hamiltonian cycle is impossible). Set $u \leftarrow$ NBR[v][0] and $w \leftarrow$ NBR[v][1]. Go to H15 if $w =$ MATE(u) and $e \neq n-2$ (cycle is too short). Set EU[e] $\leftarrow u$, EV[e] $\leftarrow v$, $e \leftarrow e+1$, EU[e] $\leftarrow v$, EV[e] $\leftarrow w$, $e \leftarrow e+1$, MATE(v) $\leftarrow v$, and makeinner(v). data structures

H8. [Take stock.] (We've just joined v to its only two neighbors, u and w, which aren't mates unless $e = n$.) Update the data structures as described in exercise 112, based on whether MATE(u)<0 and/or MATE(w)<0 (four cases).

H9. [End trigger loop?] Set $j \leftarrow j + 1$, and return to H7 if $j <$ T.

H10. [Enter new level.] Set $l \leftarrow l + 1$, and go to H13 if $e \geq n - 1$.

H11. [Choose vertex for branching.] Set CURV $\leftarrow$ RLINK[n], $d \leftarrow$ DEG(CURV), $k \leftarrow$ RLINK[CURV]. While $k \neq n$, if DEG(k) $< d$ reset CURV $\leftarrow k$ and $d \leftarrow$ DEG(k); set $k \leftarrow$ RLINK[k]. Go to H14 if $d = 0$. Otherwise set EV[e] $\leftarrow$ CURV and T $\leftarrow$ T($l-1$). (See exercise 129.)

H12. [Make CURV inner.] Call remarc(NBR[CURV][k], CURV) for $0 \leq k < d$ (thereby removing CURV from its neighbors' lists). Then deactivate(CURV), set $i \leftarrow 0$, and go to H4.

H13. [Visit a solution.] If $e < n$, set $u \leftarrow$ LLINK[n] and $v \leftarrow$ RLINK[n]; go to H14 if ADJ[u][v] $= \infty$; otherwise set EU[e] $\leftarrow u$, EV[e] $\leftarrow v$, $e \leftarrow n$. Now visit the n-cycle defined by arrays EU and EV. (See exercise 113.)

H14. [Back up.] Terminate if $l = 0$. Otherwise set $l \leftarrow l - 1$.

H15. [Undo changes.] Set $d \leftarrow$ D(l) and $i \leftarrow$ I(l) $+ 1$. Go to H14 if $i \geq d$. Otherwise set I(l) $\leftarrow i$, $e \leftarrow$ E(l), $k \leftarrow$ ($l > 0$? A($l-1$): 0), $a \leftarrow$ A(l), $v \leftarrow n$. While $k < a$, set $u \leftarrow$ ACTIVE[k], RLINK[v] $\leftarrow u$, LLINK[u] $\leftarrow v$, $v \leftarrow u$, $k \leftarrow k + 1$. Then set RLINK[v] $\leftarrow n$, LLINK[n] $\leftarrow v$, S $\leftarrow$ S(l), T $\leftarrow$ T(l). For $0 \leq k <$ S, set $u \leftarrow$ VIS[k] and (MATE(u), DEG(u)) $\leftarrow$ SAVE[$nl + u$]. Finally set CURV $\leftarrow$ CV(l). Go to H5 if $l > 0$.

H16. [Advance at root level.] Terminate if CURV < 0. Otherwise set CURU $\leftarrow$ MATE(CURV). (The previous edge CURU — CURV is gone.) Set LLINK[n] $\leftarrow$ RLINK[n] $\leftarrow n$, $a \leftarrow 0$, MATE(CURU) $\leftarrow$ MATE(CURV) $\leftarrow -1$, S $\leftarrow n$. (Everything is again bare.) If DEG(CURU) $= 2$, set TRIG[0] $\leftarrow$ CURU and T $\leftarrow 1$; otherwise set T $\leftarrow 0$. If DEG(CURV) $= 2$, set TRIG[T] $\leftarrow$ CURV and T $\leftarrow$ T $+ 1$. Go to H3 if T $= 0$. Otherwise set CV(0) $\leftarrow -1$, A(0) $\leftarrow e \leftarrow 0$, and go to H6. ∎

This marvelous algorithm has lots of steps, but it isn't terribly hard to understand. Its length arises mostly from the fact that a variety of data structures need to work together, combined with the fact that special provisions must be made at root level when no vertex has degree 2. In such cases, which are handled in steps H3 and H16, we choose a root vertex of minimum degree, and a root edge that touches it. We find all Hamiltonian cycles for which the root edge is present; then we discard that edge, and repeat the process. Eventually we will see a vertex of degree 2.

Table 1

A BAKER'S BAKER'S DOZEN OF EXAMPLE GRAPHS FOR ALGORITHM H

Graph	Description	Vertices	Edges	Degrees min .. max	Hamiltonian cycles	Running time (mems)	Mems per solution
A	*Anna Karenina*	49	138	3 .. 19	49152	414M	8429.1
B	binary trees	42	84	4 .. 4	14306485	8739M	610.8
C	concentric rings	144	216	3 .. 3	66770562	35G	524.4
D	disconnected	23	118	7 .. 22	0	65G	∞
E	expander	48	96	4 .. 4	107921396	70G	648.3
F	Fleischner G_3	57	126	3 .. 14	2	2723M	$1.4 \cdot 10^9$
G	giraffe tours	100	216	2 .. 6	4515918298	3232G	715.6
H	Halin from π	128	227	3 .. 11	10128654600	2913G	287.6
P	parity clash	82	145	2 .. 4	0	203G	∞
Q	5-cube	32	80	5 .. 5	906545760	248G	273.5
R	"random"	64	125	2 .. 6	9011601	7087M	786.4
S	Sierpiński simplex	34	96	3 .. 6	1165688832	310G	265.6
T	tripartite $K_{4,5,6}$	15	74	9 .. 11	207360000	40G	190.7
U	USA from `ME`	50	154	2 .. 48	68656026	181G	2641.5

tripartite graph, complete
benchmarks+
unstructured
Tolstoy
book
Stanford GraphBase
SGB
4-regular graph
regular graph
binary
binary trees
dodecahedron
concentric rings
Hamilton
components
tough
SGB
raman
Ramanujan graph
expander graph, see *raman*

One good way to start learning Algorithm H is to play through the steps by hand when G is a small graph like K_3 or K_4 or C_4. (See exercise 115.)

Algorithm H promises to produce interesting results galore, because the number of interesting graphs is enormous. We can get an idea of its performance in practice by studying the statistics of the benchmarks in Table 1, which reports on many different kinds of graphs. Each graph has been given an identifying letter for convenience.

• A, an "unstructured" graph based on Tolstoy's novel *Anna Karenina*. More precisely, this smallish graph arises from *book*(`"anna"`, 0, 6, 0, 0, 1, 1, 0) in the Stanford GraphBase (SGB) after repeatedly removing vertices whose degree is less than 3. Algorithm H finds its Hamiltonian cycles in a flash.

• B, by contrast, is a 4-regular graph with a strict mathematical structure. It's the SGB graph *binary*(5, 5, 0), which consists of the binary trees with 5 internal nodes; they're related by the "rotation" operation, 7.2.1.6–(12). (Algorithm 7.2.1.6L defines a Hamiltonian *path* in this graph, not a cycle.)

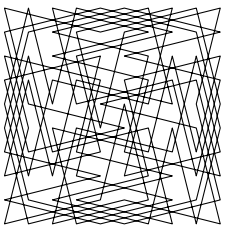
• C is one of the generalized dodecahedron graphs considered in exercise 11, with parameter $q = 36$ where Hamilton's original graph had $q = 5$.

• D is a contrived example that's obtained from six disjoint copies of K_3, together with a copy of K_5 whose five vertices are joined to everything else. It has no Hamiltonian cycles, because it breaks into six nonempty components when the five vertices of K_5 are removed. (In the terminology of exercise 117, graph D isn't "tough." Every Hamiltonian graph is tough.)

• E is the SGB graph *raman*(3, 47, 1, 1), a "Ramanujan graph of type 1." The vertices are $\{0, 1, \ldots, 46, \infty\}$, and the edges are described in exercise 119.

• F is a minor of the amazing 338-vertex graph introduced by H. Fleischner in 2013. His graph has 318 vertices of degree 4, 20 vertices of degree 14, and exactly one Hamiltonian cycle(!). (See exercise 120.)

• G is a graph of 10×10 "giraffe" moves, where a giraffe is like a knight in chess except that it's a $(4,1)$-leaper instead of a $(2,1)$-leaper. For example, the attractive tour shown here, which has 90°-rotational symmetry, was published by Maurice Kraïtchik in §67 of his pioneering book *Le Problème du Cavalier* (Paris: Gauthiers[sic]-Villars, 1927). Graph G requires the giraffe to make a special kind of tour whose diagram exhibits a "Cossack cross" in the four central squares. (A symmetrical example of such a tour appears in the answer to exercise 149.)



• H is a representative example of a large family of planar Hamiltonian graphs called "Halin graphs." (See exercises 122–125.)

• P is a non-Hamiltonian graph that's another kind of Achilles heel for Algorithm H. We obtain it by appending new vertices '!' and '!!' to the 8×10 grid graph $P_8 \mathbin{\square} P_{10}$, with three new edges u — ! — !! — v, where u and v are opposite corners of the grid. There's no Hamiltonian cycle; for if we collapse '!' and '!!' into a single vertex, we obtain an equivalent bipartite graph P' with 41 vertices in one part and 40 vertices in the other. Unfortunately, Algorithm H doesn't understand this. So it explores zillions of fruitless paths.

• Q is the familiar 5-cube, $P_2 \mathbin{\square} P_2 \mathbin{\square} P_2 \mathbin{\square} P_2 \mathbin{\square} P_2$, whose Hamiltonian cycles are the 5-bit "Gray cycles" that we investigated in Section 7.2.1.1.

Their total number, about 900 million, is just half of the value of $d(5)$ that was reported in Eq. 7.2.1.1–(26). Hmmm; was that a mistake? No: $d(5)$ considers the cycles $(v_0 \ldots v_{31})$ and $(v_{31} \ldots v_0)$ to be different, while Algorithm H does not.

• R is a graph obtained by adding 64 "random" edges to a 64-cycle. More precisely, it's the SGB graph whose official name is

$$\mathit{gunion}(\mathit{random\_graph}(64,64,0,0,0,0,0,1,1,3142),\mathit{board}(64,0,0,0,1,1,0),0,0).$$

It has only 125 edges, because three of the added edges were already present.

• S is the Sierpiński tetrahedron $S_3^{(4)}$, which was defined and illustrated in Section 7.2.2.3 (Fig. 114). Lots and lots of Hamiltonian cycles here.

• T is a special case of exercise 106.

• U, the graph that's last but not least in Table 1, is another unstructured example from "real life." It revisits the graph of the 48 contiguous states of the USA, 7.1.4–(133), augmented by two additional vertices '!' and '!!'; there are edges !! — ME, and ! — v for all $v \notin \{\texttt{ME}, !\}$. Thus its Hamiltonian cycles are the same as the Hamiltonian *paths* in 7.1.4–(133) from ME to any other state. (We used ZDD technology to treat those 68 million paths in Section 7.1.4.)

Fleischner
unique Hamiltonian cycle
giraffe
knight
leaper
90°-rotational symmetry
Kraïtchik
Cossack cross
cross
Halin graphs
grid graph
bipartite graph
5-cube
Gray cycles
random
gunion
random_graph
board
Sierpiński tetrahedron
unstructured
ZDD

census
knight's tours–
author
McKay
closed knight's tours
rotation–
reflection–
equivalence classes
bunches
wedge
al-'Adlī

A census of knight's tours. Soon after the author had first learned to program a computer in the 1950s, he wondered whether he'd be able to list all of the closed knight's tours on a chessboard. Alas, however, he quickly learned that the number of such tours is humongous — way too large to be computed by the slow machines of those days. So his hopes were dashed. In fact, nobody even had a good *estimate* for the total number of possibilities, until forty years later.

That ancient riddle was finally solved, hurray, by Brendan D. McKay, who proved (without actually constructing them) that the total number of closed knight's tours is exactly 13,267,364,410,532. [*Technical Report TR-CS-97-03* (Computer Science Department, Australian National University, 1997), 4 pages.]

Hmmm. Thirteen trillion is indeed a huge number. Yet it isn't completely out of reach. If we can visit one tour every microsecond, we can visit them all in 13 million seconds, which is about 5 months. Also, if we represent each tour as a 168-bit vector that shows which edges are used, we can store all the tours in about 279 terabytes; and we'll see later that further compression is possible.

Furthermore, the vast majority of knight's tours belong to sets of eight that are essentially the same, except for rotation and/or reflection of the board. Indeed, McKay found that there are 1,658,420,247,200 equivalence classes of size 8, and 608,233 equivalence classes of size 4 (see exercise 137); hence the total number of essentially different closed tours is the sum of those two numbers, namely 1,658,420,855,433. We could fit them all into at most 35 terabytes.

Instead of storing them all, however, we can actually *compute* them all, in a fairly short time, if we exploit parallelism. The idea is to partition the set of all tours into a large number of *bunches*, where the members of each bunch can be computed rapidly by Algorithm H. Every bunch is independent of the others. Therefore several bunches can be computed simultaneously, if we have a computer that has several processing units.

Suppose C is a closed knight's tour. Let's say that the *wedge* of C at cell (i,j) is the pair of edges that touch that cell in C. At most 8 edges touch any cell; hence there are at most $\binom{8}{2} = 28$ possible wedges at any cell, and it's convenient to give each of those possibilities a code letter, as shown in Fig. 123.

We shall partition the closed tours into 28^4 bunches, based on their wedges at the four central cells. More precisely, we shall number the rows and columns from top to bottom and left to right with the digits 0 to 7, and we shall place each tour into the bunch that corresponds to its wedges at cells 33, 34, 43, and 44.

A slightly tricky rule turns out to be a good way to give a four-letter *name* to every bunch, by writing down the code letter for the wedge at 34 after rotating the tour clockwise by 0°, 90°, 180°, and 270°, respectively. For example, let's look again at al-'Adlī's historic tour (1); it clearly has wedge `c` at cell 34. Rotating it 90° clockwise puts cell 33 into position 34, where we now see wedge `z`. Another 90° rotation yields wedge `b` at 34 (because `B` was originally at 43). And a final rotation gives us another `z`. Therefore cycle (1) belongs to bunch `czbz`.

Notice that the four equivalent tours obtained from (1) by rotation belong to bunches `czbz`, `zbzc`, `bzcz`, and `zczb`. In general, the tours of bunch $\alpha_1\alpha_2\alpha_3\alpha_4$

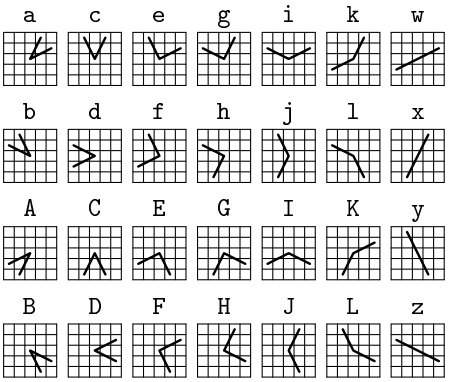


Fig. 123. The 28 possible wedges of a knight.

The angle θ of the narrowest wedges, {a, b, A, B}, is $\arctan\frac{3}{4} \approx 37°$; {c, d, C, D} are slightly wider, $90° - \theta = \arctan\frac{4}{3} \approx 53°$. Eight wedges, namely {e, f, E, F, g, h, G, H}, make a 90° turn. Then come {i, j, I, J}, at $90° + \theta \approx 127°$; and {k, l, K, L}, at $180° - \theta \approx 143°$. Finally, wedges {w, x, y, z} are straight. Notice that a 90° counterclockwise rotation changes a ↦ b ↦ A ↦ B ↦ a; ...; k ↦ l ↦ K ↦ L ↦ k; w ↦ y ↦ w; and x ↦ z ↦ x.

top-bottom reflection
multiplicity
canonical
ASCII
uppercase letters

are equivalent to the tours of bunches $\alpha_2\alpha_3\alpha_4\alpha_1$, $\alpha_3\alpha_4\alpha_1\alpha_2$, and $\alpha_4\alpha_1\alpha_2\alpha_3$ whenever each α_j is one of the 28 wedge codes.

Reflection also gives an equivalent tour, whose bunch depends only on the unreflected bunch name. For example, the top-bottom reflection of cycle (1) gives a cycle that belongs to bunch yByD. In general, let ρ and τ be the permutations of wedge codes that correspond to 90° rotation and to top-bottom reflection. Then $\alpha \mapsto \alpha\rho$ is the mapping discussed in Fig. 123, and we have

$$\begin{aligned} \alpha &= \texttt{a b c d e f g h i j k l A B C D E F G H I J K L w x y z}\,; \\ \alpha\rho &= \texttt{b A d C f E h G j I l K B a D c F e H g J i L k y z w x}\,; \\ \alpha\tau &= \texttt{B A C d G h E f I j l k b a c D g H e F i J L K z y x w}\,; \\ \bar\alpha = \alpha\tau\rho &= \texttt{a B D C H G F E J I K l A b d c h g f e j i k L x w z y}\,. \end{aligned} \tag{24}$$

Exercise 142 shows that top-bottom reflection maps $\alpha_1\alpha_2\alpha_3\alpha_4 \mapsto \bar\alpha_4\bar\alpha_3\bar\alpha_2\bar\alpha_1$.

The main consequence is that, if $\alpha_1\alpha_2\alpha_3\alpha_4$ is any one of the 28^4 bunches, its closed tours are equivalent to those of seven other bunches:

$$\alpha_2\alpha_3\alpha_4\alpha_1,\ \alpha_3\alpha_4\alpha_1\alpha_2,\ \alpha_4\alpha_1\alpha_2\alpha_3,\ \bar\alpha_4\bar\alpha_3\bar\alpha_2\bar\alpha_1,\ \bar\alpha_3\bar\alpha_2\bar\alpha_1\bar\alpha_4,\ \bar\alpha_2\bar\alpha_1\bar\alpha_4\bar\alpha_3,\ \bar\alpha_1\bar\alpha_4\bar\alpha_3\bar\alpha_2. \tag{25}$$

In most cases these eight bunches are distinct, and we say that $\alpha_1\alpha_2\alpha_3\alpha_4$ has multiplicity 8. For example, the equivalent bunches czbz, zbzc, bzcz, zczb, yByD, ByDy, yDyB, DyBy all have multiplicity 8. But sometimes all eight bunches are identical, and we say that the multiplicity is 1. (There are just four bunches of multiplicity 1, namely aaaa, llll, AAAA, and LLLL.) Exercise 146 shows that 30 canonical bunches have multiplicity 2; and 774 of the canonical bunches have multiplicity 4. The multiplicity is always equal to either 1 or 2 or 4 or 8.

A bunch is *canonical* if it is the lexicographically smallest of the bunches equivalent to it, where the lexicographic order uses ASCII code (so that uppercase letters *precede* lowercase letters). For example, ByDy is canonical; it's lexicographically smaller than bzcz and the other six equivalent bunches.

Although there are $28^4 = 614656$ bunches altogether, only 77245 of them are canonical. Furthermore, no knight's tour has 'a' anywhere in its bunch name; do you see why? This reduces the number of relevant canonical bunches to 66771. (See exercises 145 and 148.)

symmetry
parallelism
diverse knight's tours+

In order to carry out a census of all the closed knight's tours, it therefore suffices to solve subproblems of the form "Visit all of the tours in bunch $\alpha_1\alpha_2\alpha_3\alpha_4$," for 66,771 canonical names $\alpha_1\alpha_2\alpha_3\alpha_4$. And each of those 66,771 subproblems asks for the Hamiltonian cycles on a modified 8×8 knight graph, where eight particular edges are forced to be part of the cycle. Equivalently, 24 particular edges of that knight graph are forbidden. Every subproblem is in fact "Algorithm H friendly," because at least 16 of the remaining edges — 8 edges in the center, and 8 edges in the corners — are forced.

For instance, it turns out that bunch `ByDy` has exactly 31,905,973 tours; and Algorithm H needs only 16 Gμ (a few seconds) to visit them all. The same is true, of course, for bunch `czbz`; but we don't need that bunch in our census, because it's not canonical.

One easy way to carry out the census is to prepare ten shell scripts, each with 6677 or 6678 of the subproblems. Then run all the scripts simultaneously, on a machine with 10 processors. At the time this section was written, the job was thereby accomplished with off-the-shelf hardware in less than two days. Notice that this strategy, via bunches, saves a factor of 8 because of symmetry, and another factor of 10 because of parallelism.

Almost all of the 66,771 bunches contributed solutions; the only exceptions were 198 cases of the form α`1`β`1` or `1`α`1`β, and Algorithm H rejected them immediately. The smallest nonempty canonical bunch class was `CFgd`, with only 165,504 solutions (80 Mμ); the largest was `LLLL`, with 652,228,612 solutions (287 Gμ); the median was `Cflz`, with 17,440,101 solutions (8587 Mμ). To get the total number of tours, we simply compute the sum, over all bunches, of the number of solutions times the multiplicity. (Without multiplying by multiplicity, the total number of solutions over all bunches came to 1,671,517,634,718.)

The scheme just described has worked well, but we could have conducted the census in many other ways. For example, instead of defining bunches based on the 28^4 possible wedges at the four central cells, we could have based our definition on the 6^8 possible wedges at centrally located boundary cells. Or we could have used the 5^8 possible wedges at the cells that are a knight's move away from a corner. Exercises 151 and 152 explore those interesting alternatives.

The ability to conduct a reasonably quick census opens the door to the solution of many problems that were long thought to be out of reach, and it also raises new questions that are interesting in their own right. For example, how many of the 13 trillion possible tours involve each of the 28 possible wedges at least once? (Answer: 278,078,503,988.) How many of those tours involve each wedge at least twice? (Answer: 155,528.) And how many of *those* doubly diverse tours involve each wedge at most thrice? (Answer: 70,240.) In the latter tours, exactly eight of the 28 wedges occur three times, because $64 = 20\cdot2 + 8\cdot3$.

It turns out that Algorithm H is *not* the bottleneck when answering questions of this kind: The process of analyzing a tour tends to take longer than the process of generating the tour itself, because Algorithm H is so efficient. The total time is therefore roughly equal to the time to analyze 1.67 trillion tours, divided by the degree of parallelism.

topological type
dynamic enumeration–
induced subgraph
Hamilton
dodecahedron

Exercises 156–164 are devoted to a wide variety of questions that can be answered by a good census-taker, and Fig. A–19 in the answer pages is a gallery of knight's tours with unusual properties. For example, one of the 70,240 tours with maximally diverse wedges appears in Fig. A–19(a).

What's the maximum number of times that a knight can make the sharpest possible turn during a complete cycle, forming a tight angle of just $\theta \approx 37°$? (See Fig. 123.) Contrariwise, what's the maximum number of times that it can continue in the same direction as its previous move, making no turn at all? Exercise 158 clears up those riddles.

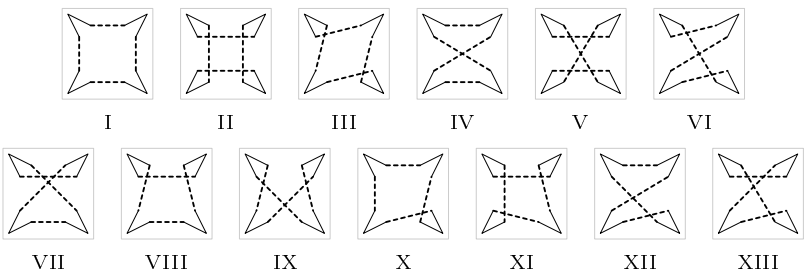


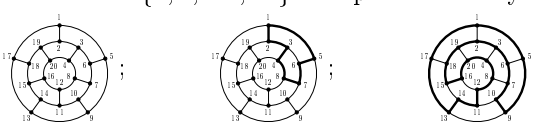
Fig. 124. The thirteen topological types of knight's cycles. Every Hamiltonian cycle on an $n \times n$ knight graph includes eight fixed edges, forming narrow wedges at each corner. The endpoints of those wedges can be connected up in 13 essentially different ways, modulo rotation and reflection, indicated by dashed lines in these diagrams. (The actual paths of interconnection can, of course, have wildly differing lengths and shapes.)

Figure 124 suggests a census-oriented question of a different kind, because it points out that there are 13 fundamentally different kinds of knight's cycles on a square board. How many tours are of each topological type? (See exercise 156.)

Dynamic enumeration. Let's switch gears now and focus on *counting*. Instead of trying to visit every Hamiltonian cycle of a given graph, we'll try only to figure out exactly *how many* such cycles exist.

Algorithm E below is, in fact, designed to solve a somewhat more general problem: Given a graph G on the vertices $\{1, 2, \ldots, n\}$, we'll determine the number of m-cycles in the induced subgraph $G_m = G \mid \{1, \ldots, m\}$, for $3 \le m \le n$. In particular, when $m = n$ we'll know the number of Hamiltonian cycles in G.

Algorithm E is easy to understand, once you understand it, but not so easy to explain. We shall study it by looking first at how it applies to Hamilton's original example, the vertices of a dodecahedron. To start, let's redraw Fig. 122(a) so that the vertices are named $\{1, 2, \ldots, 20\}$ in a particular way:



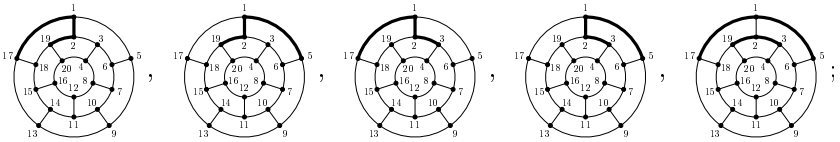
(26)

On the left is Hamilton's graph G; in the middle is an 8-cycle in G_8, clearly unique; on the right is the 20-cycle of Fig. 122(b).

m-config
m-frontier
outer
inner
bare
m-class

The key idea that underlies Algorithm E is the notion of an "m-config," which is a subset of the edges that satisfies three properties: (i) Every vertex $\le m$ appears in exactly two edges. (ii) No edge has both endpoints $> m$. (iii) There is no cycle of edges. One consequence of (i) and (iii) is that the edges of an m-config always form disjoint subpaths of the graph. One consequence of (ii) is that the only 0-config is the empty set.

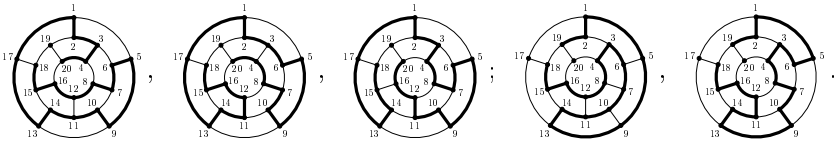
For example, Hamilton's graph obviously has just five 2-configs, namely


(27)

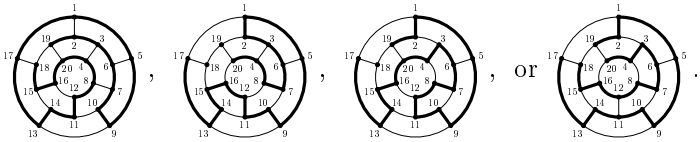
and they contain respectively 1, 1, 1, 1, 2 subpaths.

The "m-frontier" F_m of G is the set of vertices $> m$ that are reachable from $\{1,\ldots,m\}$. Building on our experience with Algorithm H, we classify each vertex of F_m in an m-config as either "outer" (an endpoint of a subpath), or "inner" (an intermediate vertex of a subpath), or "bare" (not in any subpath), according as its degree in the m-config is 1, 2, or 0. (There'll be exactly $2t$ outer vertices when there are t subpaths.) Two m-configs are *equivalent* if they have the same outer, inner, and bare vertices, and if the outer ones are paired up in the same way.

$F_2 = \{3,5,17,19\}$, and no two of the 2-configs in (27) are equivalent. But when m gets larger, we usually have fairly large equivalence classes. For instance, it turns out that Hamilton's graph has exactly 32 16-configs, including these five:


(28)

The first three of these are equivalent, and so are the last two. Furthermore, *every* 16-config turns out to be equivalent either to one of those or to one of


(29)

Algorithm E works by systematically discovering every "m-class," namely, each equivalence class of m-configs, while also computing all of the class sizes. So it's important to give an appropriate *name* to each m-class. When F_m has q elements $(u_1,\ldots,u_q)$, this name consists of q integers a_j, one for each element of the frontier: If u_j is inner, a_j is -1 (written '$\bar{1}$' for short). If u_j is bare, $a_j = 0$. And if u_j is outer, with mate $u_{j'}$ at the other end of its subpath and $j' > j$, we set a_j and $a_{j'}$ to the smallest positive integer not assigned to an outer vertex u_i with $i < j$.

For example, the frontier F_{16} is $(u_1,u_2,u_3,u_4) = (17,18,19,20)$. The 16-class at the left of (28) has a subpath from 18 to 20, with 17 inner and 19 bare; so its name is $\bar{1}101$. The class at the right has a subpath from 18 to 19 and leaves both 17 and 20 bare; so its name is 0110. The names of the four classes

full disclosure
extended frontier
notation $\mapsto_m$

in (29) are respectively $\bar1 11\bar1$, 1001, $101\bar1$, and 1212. (Check them!) Algorithm E determines not only that every 16-config belongs to one of those six classes, but also that the classes contain respectively 4, 2, 6, 6, 4, and 10 configs.

Confession: The statements above are almost true, but not really correct. Algorithm E actually works with an *extended* frontier $\widehat F_m = F_m \cup \{m+1\}$, instead of with F_m, for $0 \le m < n$; in particular, $\widehat F_0 = \{1\}$. This modification makes the program simpler. And it doesn't change the definition of equivalence classes, because vertex $m+1$ will always be bare if it wasn't already in F_m.

The precise ordering of vertices $(u_1, \ldots, u_q)$, where $q = |\widehat F_m|$, is important for naming the m-classes. A somewhat peculiar rule turns out to work best: We divide $\widehat F_m$ into two parts, $\widehat F_m^- = \big(\widehat F_{m-1} \cup \{m{+}1\}\big) \setminus \{m\}$ and $\widehat F_m^+ = \widehat F_m \setminus \widehat F_m^-$; then we place the elements of $\widehat F_m^-$ first, otherwise sorting into increasing order:

$$\{u_1, \ldots, u_{q_0}\} = \widehat F_m^-,\ \{u_{q_0+1}, \ldots, u_q\} = \widehat F_m^+, \text{ where } q_0 = |\widehat F_m^-|; \quad u_j < u_{j+1} \text{ for } 1 \le j < q \text{ and } j \ne q_0. \tag{30}$$

For example, the extended-and-ordered frontiers of Hamilton's graph are

$$\begin{array}{lll}
\widehat F_0 = (1); & \widehat F_7 = (8,9,17,19,20,10); & \widehat F_{14} = (15,16,17,19,20);\\
\widehat F_1 = (2,5,17); & \widehat F_8 = (9,10,17,19,20,12); & \widehat F_{15} = (16,17,19,20,18);\\
\widehat F_2 = (3,5,17,19); & \widehat F_9 = (10,12,17,19,20,13); & \widehat F_{16} = (17,18,19,20);\\
\widehat F_3 = (4,5,17,19,6); & \widehat F_{10} = (11,12,13,17,19,20); & \widehat F_{17} = (18,19,20);\\
\widehat F_4 = (5,6,17,19,8,20); & \widehat F_{11} = (12,13,17,19,20,14); & \widehat F_{18} = (19,20);\\
\widehat F_5 = (6,8,17,19,20,9); & \widehat F_{12} = (13,14,17,19,20,16); & \widehat F_{19} = (20).\\
\widehat F_6 = (7,8,9,17,19,20); & \widehat F_{13} = (14,16,17,19,20); &
\end{array} \tag{31}$$

Notice that $\widehat F_m$ always begins with $u_1 = m+1$.

Everything works nicely because we can readily enumerate all the m-classes once we know the names and sizes of all the $(m{-}1)$-classes. Indeed, the transition from $m-1$ to m means that vertex m gains respectively $(0,2,1)$ neighbors if it is (inner, bare, outer). And the state of vertex m in an $(m{-}1)$-config is the first digit of its class name; thus vertex m gains $(0,2,1)$ neighbors if and only if that name begins with $(\bar1, 0, 1)$, respectively.

When $m = 17$, for example, we know that Hamilton's 16-configs have the class names $\bar1101$, $\bar1 11\bar1$, 0110, 1001, $101\bar1$, 1212. In cases $\bar1101$ and $\bar1 11\bar1$, vertex 17 is already inner; so those cases are already 17-configs. In case 0110, the class fizzles out and leads to no 17-configs, because vertex 17 has only one neighbor in the frontier (namely vertex 18) and it cannot gain two. Cases 1001, $101\bar1$ and 1212 do lead to 17-configs, when 17 is joined to 18; see exercise 173.

Let's write '$\alpha \mapsto_m \beta$' if the $(m{-}1)$-class α can lead to the m-class β. Then we can verify, using (31), that the sequence

$$\begin{aligned}
&0 \mapsto_1 011 \mapsto_2 1221 \mapsto_3 01122 \mapsto_4 121233 \mapsto_5 123123 \mapsto_6 123312 \mapsto_7\\
&\quad 122313 \mapsto_8 121233 \mapsto_9 \bar1 12210 \mapsto_{10} 010221 \mapsto_{11} \bar1 01122 \mapsto_{12}\\
&\quad 012210 \mapsto_{13} \bar1 0\bar1 11 \mapsto_{14} 00\bar1 11 \mapsto_{15} 1\bar1 221 \mapsto_{16} \bar1 11\bar1 \mapsto_{17} 11\bar1 \mapsto_{18} C_{20}
\end{aligned} \tag{32}$$

takes us step-by-step from the left of (26) to the right, if '$\alpha \mapsto_m C_p$' means that the $(m{-}1)$-class α can be immediately followed by a p-cycle in $G\,|\,\{1, \ldots, p\}$.

In general, if C is *any* m-cycle in $G_m = G\,|\,\{1,\ldots,m\}$, where G is any graph with m or more vertices, there's a unique sequence of transitions

$$0 = \alpha_0 \mapsto_1 \alpha_1 \mapsto_2 \alpha_2 \;\cdots\; \alpha_{j-1} \mapsto_j C_m, \quad \text{for some } j < m. \tag{33}$$

For we obtain the j-class α_j by first removing all edges of C_m whose endpoints both exceed j; then we use the extended frontier $\widehat{F}_j$ to name the j-config that results. Conversely, any sequence (33) defines a unique m-cycle in G_m. This one-to-one correspondence is the basis of Algorithm E.

Notice that the size of a j-class α, that is, the number of equivalent j-configs that it contains, is the number of paths of length j from 0 to α in such a sequence of transitions. We're counting Hamiltonian cycles by counting paths in a (large) digraph of j-classes.

The main data structures for Algorithm E are two tries (see Section 6.3), one for classes of the $(m-1)$-configs already enumerated and one for classes of the newly seen m-configs that they spawn. When the extended frontier $\widehat{F}_m$ has size q, the trie for m-configs has q levels, representing successive digits of each class name. Then there's a "bottom" level of lieves, containing the class sizes.

tries
lieves
Lief: A leaf of a trie
Lieves: The plural of "Lief."
prefix
bignum
Null pointers

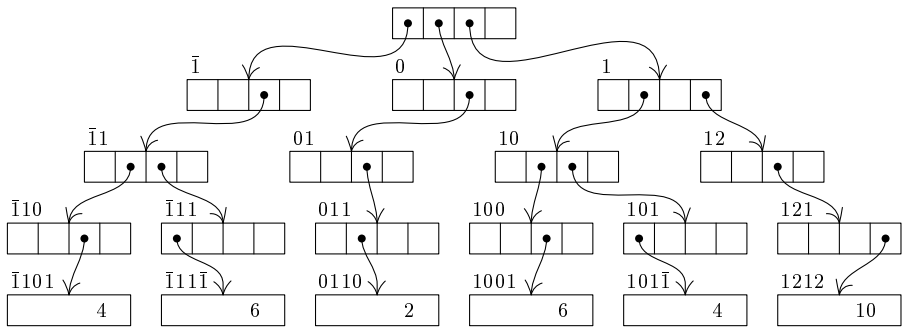


Fig. 125. A trie with $q = 4$, having $\Delta = 4$ fields in each node.

For example, the six 16-classes in (28) and (29) might be represented by the trie in Fig. 125. There's one lief for every class name $a_1 \ldots a_q$; and the path to that lief, from the root at level 0, implicitly specifies the digits $a_1, \ldots, a_q$ in turn. More precisely, every node on level l, for $0 \le l < q$, has Δ fields, representing the potential digits $(\bar{1}, 0, 1, \ldots)$ that might appear in a name; the field for a_{l+1} links to the node or lief at level $l+1$. In this way each node or lief on level l represents all classes whose name begins with a particular prefix $a_1 \ldots a_l$.

Algorithm E uses two arrays, `MEM` and `WT`, to represent a trie. Each element of `MEM` is a node that's capable of holding Δ pointers, where Δ has been chosen large enough to exceed $a_l + 1$ for any digit a_l. Each element of `WT` is a "bignum," a nonnegative integer that might be rather large; 128 bits or more are typically allocated for each bignum, depending on the input graph. Trie nodes live in `MEM`; trie lieves live in `WT`. For example, node 011 in Fig. 125 might be in `MEM`[9], and lief 0110 might be in `WT`[3]. Then we'd have `MEM`[9][1] = 3 and `WT`[3] = 2. (Null pointers, like `MEM`[9][0], are zero, and shown as blanks in this illustration.)

sparse-set data structure
FR
IFR
0-origin and 1-origin indexing
MATE table
BMATE table
basic mate table

A sparse-set data structure is ideal for maintaining the frontiers as m grows. There's an array FR, which contains a permutation of the vertices, and a companion array IFR for the inverse permutation. The first q elements of FR are the current frontier. More precisely, we have

$$\texttt{FR[IFR[}v\texttt{]]} = v, \text{ for } 1 \le v \le n; \quad \text{and} \quad 1 \le \texttt{FR[}k\texttt{]} \le n, \text{ for } 0 \le k < n. \tag{34}$$

Vertex v is part of $\widehat{F}_m$ if and only if $\texttt{IFR[}v\texttt{]} < q$. Vertex u_j in the discussion above corresponds to $\texttt{FR[}j{-}1\texttt{]}$ in the computer's internal representation. (These conventions intentionally mix 0-origin and 1-origin indexing. Algorithm E wants the vertices to be named $\{1, 2, \ldots, n\}$, not $\{0, 1, \ldots, n{-}1\}$, for ease in exposition.)

The main work of Algorithm E, which is to carry out the transitions from $(m{-}1)$-classes to m-classes, is greatly facilitated by the use of a MATE table somewhat like that of Algorithm H: $\texttt{MATE[}j\texttt{]} = (-1,\ 0,\ k > 0)$ means that u_j is respectively (inner, bare, mated to u_k). For example, class $\bar{1}101$ is equivalent to

$$\texttt{MATE[1]} = -1,\ \texttt{MATE[2]} = 4,\ \texttt{MATE[3]} = 0,\ \texttt{MATE[4]} = 2, \tag{35}$$

because both conventions mean that u_1 is inner, u_3 is bare, and that there's a subpath whose endpoints are u_2 and u_4. It's easy to convert from one convention to the other (see exercise 179).

The transition from $m-1$ to m is basically straightforward. But the details can be a bit tricky, because two frontiers and two MATE tables are involved. The $(m{-}1)$-classes are characterized by a table $\texttt{OMATE[}j\texttt{]}$ for $1 \le j \le q' = |\widehat{F}_{m-1}|$ based on the "old" frontier $\widehat{F}_{m-1} = (u'_1, \ldots, u'_{q'})$, while the m-classes are characterized by a table $\texttt{MATE[}j\texttt{]}$ for $1 \le j \le q = |\widehat{F}_m|$ that's based on the "current" frontier $\widehat{F}_m = (u_1, \ldots, u_q)$. Vertices that belong to both frontiers are represented by different indices in OMATE and MATE.

Consider, for example, the case $m = 8$ in graph (26). The old frontier $\widehat{F}_7$ is $(8, 9, 17, 19, 20, 10)$, while $\widehat{F}_8$, the current frontier, is $(9, 10, 17, 19, 20, 12)$, according to (31). Thus $u'_2 = u_1$ and $u'_6 = u_2$. A subpath from vertex 9 to vertex 17 is represented by $\texttt{OMATE[2]} = 3$ in a 7-config, but by $\texttt{MATE[1]} = 3$ in an 8-config.

In general, if we set $u_0 = m$, there's a one-to-one mapping σ such that

$$u'_j = u_{j\sigma}, \quad \text{for } 1 \le j \le q'; \quad 1\sigma = 0. \tag{36}$$

Going the other way, if we set $u'_0 = m + 1$, there's a one-to-one mapping τ with

$$u_k = u'_{k\tau}, \quad \text{for } 1 \le k \le q_0; \tag{37}$$

here q_0 is defined in (30). We have $1\tau = 0$ if and only if $q_0 = q'$ (see exercise 181).

Three main cases arise when we consider the m-classes that can follow a given $(m{-}1)$-class, depending on whether vertex m is inner, bare, or outer in that class. In other words, there are three cases, depending on whether $\texttt{OMATE[1]}$ is -1, 0, or > 1. The first case is easy, because the given $(m{-}1)$-class is already an m-class, and its MATE table is directly inherited from OMATE. We shall call this the BMATE table ("basic mate table"):

$$\texttt{BMATE[}k\texttt{]} = \begin{cases} \texttt{OMATE[}k\tau\texttt{]}\sigma, & \text{if } 1 \le k \le q_0; \\ 0, & \text{if } q_0 < k \le q; \end{cases} \qquad \begin{matrix} \texttt{OMATE[0]} = 0, \\ (-1)\sigma = -1,\ 0\sigma = 0. \end{matrix} \tag{38}$$

NBR
bignum
contribute()
try

In the other two cases, we start with the basic mate table, then add either two edges from m to non-inner vertices (if OMATE[1] $= 0$), or one edge from m to a non-inner vertex (if OMATE[1] > 1), in all possible ways. We set up an array called NBR, so that the r edges from vertex m to vertices $> m$ are represented as

$$m \mathrel{—} u_{\texttt{NBR}[0]},\ \ldots,\ m \mathrel{—} u_{\texttt{NBR}[r-1]}. \tag{39}$$

Algorithm E (*Enumerate Hamiltonian cycles*). Given a graph G on the vertices $\{1, 2, \ldots, n\}$, this algorithm computes CYC[m], the number of m-cycles in the induced graph $G \mid \{1, 2, \ldots, m\}$, for $3 \le m \le n$. As described above, it uses the arrays MEM, WT, OMEM, and OWT to represent tries; FR and IFR to represent frontiers; NBR to represent neighbors; and several other auxiliary arrays, which are described in various exercises that contain implementation details.

E1. [Initialize.] Set CYC[m] $\leftarrow 0$ (which is a "bignum"), for $3 \le m \le n$. Set FR[k] $\leftarrow k+1$ and IFR[k+1] $\leftarrow k$, for $0 \le k < n$. Set MEM[0][j] $\leftarrow$ OMEM[0][j] $\leftarrow 0$ for $0 \le j < \Delta$. Also set $m \leftarrow 1$, $q \leftarrow 1$, MEM[0][1] $\leftarrow 1$, and WT[1] $\leftarrow 1$ (a "bignum").

E2. [Establish the trie for $m-1$.] (At this point, FR and IFR represent the external frontier $\widehat{F}_{m-1}$, which has q elements. Arrays MEM and WT represent the trie of $(m-1)$-classes.) Set $q' \leftarrow q$, $p \leftarrow w \leftarrow 0$; also swap OMEM $\leftrightarrow$ MEM and OWT $\leftrightarrow$ WT. (Only the base addresses change. Thus OMEM and OWT now represent the $(m-1)$-classes. The previous contents of OMEM and OWT are now irrelevant; we'll construct the trie of m-classes in their place. That trie contains p nodes and w lieves as it is being built. It's now empty, because $p = w = 0$.) Change FR, IFR, q_0, and q so that they now represent $\widehat{F}_m$. (See exercise 187.)

E3. [Visit the first $(m-1)$-class.] Set $q'+1$ pointer variables $p'_0, \ldots, p'_{q'}$ so that OMEM[p'_l] is node $a'_1 \ldots a'_l$ for $0 \le l < q'$ and OWT[$p'_{q'}$] is lief $a'_1 \ldots a'_{q'}$, where $a'_1 \ldots a'_{q'}$ is the lexicographically smallest $(m-1)$-class. (See exercise 189.)

E4. [Prepare to process $a'_1 \ldots a'_{q'}$.] Set up the OMATE table and the BMATE table. (See exercise 179(a) and (36)–(38).) Go to step E5 if OMATE[1] < 0, to step E6 if OMATE[1] $= 0$, otherwise to step E7.

E5. [Contribute when m is inner.] Set MATE[k] $\leftarrow$ BMATE[k] for $1 \le k \le q$. Then call contribute() (exercise 191) and go to E8.

E6. [Contribute when m is bare.] Call the subroutine try(NBR[i], NBR[j]) for $0 \le i < j < r$ (see exercise 193), and go to E8.

E7. [Contribute when m is outer.] Call try(OMATE[1]σ, NBR[k]) for $0 \le k < r$.

E8. [Visit the next $(m-1)$-class.] Set $q'+1$ pointer variables $p'_0, \ldots, p'_{q'}$ so that OMEM[p'_l] is node $a'_1 \ldots a'_l$ for $0 \le l < q'$ and OWT[$p'_{q'}$] is lief $a'_1 \ldots a'_{q'}$, where $a'_1 \ldots a'_{q'}$ is the lexicographically smallest unvisited $(m-1)$-class, and return to E4. (See exercise 189.) If all of the $(m-1)$-classes have been visited, however, set $m \leftarrow m+1$. Return to E2 if $w > 0$; otherwise terminate. ∎

A superficial glance at this algorithm leads to a natural question: Where does it actually calculate the values CYC[3], CYC[4], ..., CYC[n], which are the desired outputs? The answer is that those values accumulate as m-cycles are discovered, during the calls of try(i, j) in steps E6 and E7.

complete graph
OEIS
bignum
frontiers
colexicographic order

It's quite instructive to watch Algorithm E in action when G is the complete graph on n vertices. That's when we get the most cycles. (See exercise 183.)

On the other hand, we've developed Algorithm E by considering a toy problem that has only a few Hamiltonian cycles. Indeed, when we apply it to the little graph (26), the results are that `CYC[8]` $=$ `CYC[14]` $= 1$, `CYC[17]` $= 2$, `CYC[20]` $= 30$, and `CYC[`m`]` $= 0$ otherwise. Ho hum. What's the point? We obviously could have counted those cycles much faster by just visiting them directly.

But Algorithm E yields truly impressive results when we apply it to many other graphs. For example, suppose G is the graph of knight moves on an 8×32 board, with vertices ordered column by column from left to right. How many closed tours are possible? Answer:

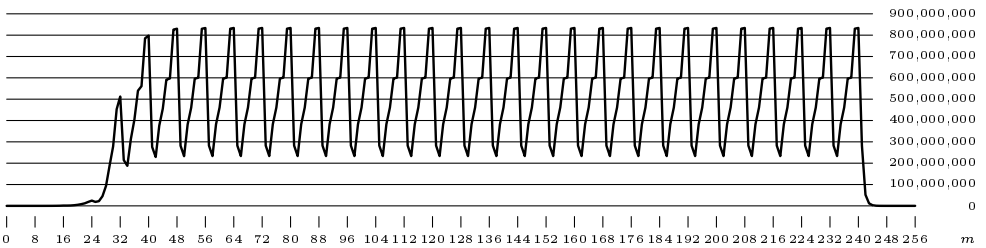
$$2{,}989{,}043{,}104{,}279{,}785{,}843{,}506{,}369{,}864{,}414{,}419{,}975{,}166{,}020,\\ 125{,}721{,}505{,}674{,}144{,}076{,}449{,}194{,}991{,}145{,}270{,}100. \tag{40}$$

Almost 3 quinvigintillion (see OEIS A193055)! That's `CYC[256]`. Of course, while computing this bignum, Algorithm E also discovers our old friend `CYC[64]`, the number of closed 8×8 tours, and many other interesting numbers along the way (see exercise 196). And the running time for the entire calculation was just 66.5 teramems — about 2.5 teramems to go from $8 \times k$ to $8 \times (k{+}1)$ boards.

Let's look more closely at the calculations that led to (40). The extended-and-ordered frontiers of the 8×32 knight graph start small, but they rapidly fall into a pattern in which each $\widehat{F}_m$ has size 16 or 17:

$$\begin{aligned}
\widehat{F}_0 &= (00);\\
\widehat{F}_1 &= (10, 21, 12);\\
\widehat{F}_2 &= (20, 21, 12, 31, 02, 22);\\
\widehat{F}_3 &= (30, 21, 31, 02, 12, 22, 01, 41, 32);\\
\widehat{F}_4 &= (40, 01, 21, 31, 41, 02, 12, 22, 32, 11, 51, 42);\\
\widehat{F}_5 &= (50, 01, 11, 21, 31, 41, 51, 02, 12, 22, 32, 42, 61, 52);\\
\widehat{F}_6 &= (60, 01, 11, 21, 31, 41, 51, 61, 02, 12, 22, 32, 42, 52, 71, 62);\\
\widehat{F}_7 &= (70, 01, 11, 21, 31, 41, 51, 61, 71, 02, 12, 22, 32, 42, 52, 62, 72);\\
\widehat{F}_8 &= (01, 11, 21, 31, 41, 51, 61, 71, 02, 12, 22, 32, 42, 52, 62, 72);\\
\widehat{F}_9 &= (11, 21, 31, 41, 51, 61, 71, 02, 12, 22, 32, 42, 52, 62, 72, 13);\\
\widehat{F}_{10} &= (21, 31, 41, 51, 61, 71, 02, 12, 22, 32, 42, 52, 62, 72, 13, 03, 23);\\
\widehat{F}_{11} &= (31, 41, 51, 61, 71, 02, 12, 22, 32, 42, 52, 62, 72, 03, 13, 23, 33);\\
\widehat{F}_{12} &= (41, 51, 61, 71, 02, 12, 22, 32, 42, 52, 62, 72, 03, 13, 23, 33, 43);\\
\widehat{F}_{13} &= (51, 61, 71, 02, 12, 22, 32, 42, 52, 62, 72, 03, 13, 23, 33, 43, 53);\\
\widehat{F}_{14} &= (61, 71, 02, 12, 22, 32, 42, 52, 62, 72, 03, 13, 23, 33, 43, 53, 63);\\
\widehat{F}_{15} &= (71, 02, 12, 22, 32, 42, 52, 62, 72, 03, 13, 23, 33, 43, 53, 63, 73);\\
\widehat{F}_{16} &= (02, 12, 22, 32, 42, 52, 62, 72, 03, 13, 23, 33, 43, 53, 63, 73);\\
\widehat{F}_{17} &= (12, 22, 32, 42, 52, 62, 72, 03, 13, 23, 33, 43, 53, 63, 73, 14);\\
\widehat{F}_{18} &= (22, 32, 42, 52, 62, 72, 03, 13, 23, 33, 43, 53, 63, 73, 14, 04, 24);\\
\widehat{F}_{19} &= (32, 42, 52, 62, 72, 03, 13, 23, 33, 43, 53, 63, 73, 04, 14, 24, 34);
\end{aligned} \tag{41}$$

and so on. (The vertices have row-column names (00, 10, ..., 70, 01, 11, ..., 71, 02, ...) here, instead of (1, 2, ..., 256).) When $7 \le m \le 232$, $\widehat{F}_{m+8}$ turns out to be the same as $\widehat{F}_m$, except that the vertices are shifted one column to the right.



linear recurrence
generating function
growth ratio
periodic knight's tours

Fig. 126. The number of lieves, C_m, in the tries for m-classes in the 8×32 knight graph.

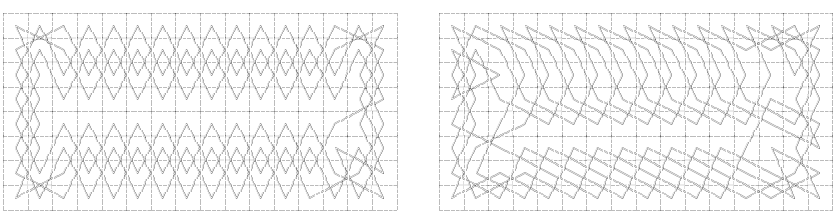
The m-classes also fall into a pattern that's periodic modulo 8; but they don't stabilize until m gets somewhat larger. It turns out that, when $m \bmod 8$ is $(0, 1, \ldots, 7)$, the number C_m of m-classes is respectively (282609677, 233377701, 382538097, 461026164, 596486159, 601036842, 830339355, 833813266), for $72 \le m \le 240$; thus it reaches its peak when $m \bmod 8 = 7$ and we're closing out a column. Similarly, the number P_m of nonlief trie nodes turns out to be respectively (531992432, 470709142, 834186552, 1115100721, 1320322736, 1343754219, 1779294552, 1798400809), which is about (1.88, 2.02, 2.18, 2.42, 2.21, 2.24, 2.14, 2.16) nodes per class. Further statistics are discussed in exercise 198.

This periodic pattern proves that the number `CYC[`$8n$`]` of $8 \times n$ knight's cycles satisfies a linear recurrence, and has a generating function that's a quotient of (huge) polynomials. Empirically, the numbers are very close to $.000000002465 \cdot 526.46^n$, for $16 \le n \le 32$.

Periodicity mod 8 suggests that we can also construct "periodic knight's tours," by finding classes $\alpha_0, \alpha_1, \ldots, \alpha_7$ such that

$$\alpha_0 \mapsto_m \alpha_1, \quad \alpha_1 \mapsto_{m+1} \alpha_2, \quad \alpha_2 \mapsto_{m+2} \alpha_3, \quad \ldots, \quad \alpha_7 \mapsto_{m+7} \alpha_0. \tag{42}$$

Such sequences of transitions occur, for example, when $m \bmod 8 = 0$ and α_0 is 1234214300000000 or $01\bar{1}2314505004023$ (see exercise 200). If we can also find transitions from 0 to α_0, and from α_0 to C_p for some p with $p \bmod 8 = 0$, we obtain complete knight's cycles with 8 rows and an unbounded number of columns:



(43)

These 8×16 tours become $8 \times (k+5)$ tours when the m-class α_0 occurs k times, for every $k \ge 1$.

It's fascinating to follow the course of the knight in labyrinthine tours like this. Starting, for example, at the left of the left-hand tour, the knight will hop all the way to the right, then left, then right, then left, right, left, right, and left again. A knight traversing the right-hand tour will reverse direction ten times!

memory+
RAM
Hamiltonian *paths*
∞
Hamilton
OEIS
open knight's tours
Chernov
Stertenbrink
growth ratio

Algorithm E is remarkably fast, but it needs *lots* of memory. Although the frontier sets in (41) are rather large, the algorithm succeeds only because they aren't *too* large. No frontier has more than 17 elements; hence each trie node needs to hold only 10 pointers (see exercise 195). And we've seen that no trie has more than 1.8 billion nodes; hence each pointer needs to occupy only four bytes. Therefore, with 40 bytes in each trie node and 48 bytes in each bignum, the total bytewise memory requirement is roughly 2 billion times 40, plus 1 billion times 48, namely 128 gigabytes per trie. (Exercises 203 and 204 show, however, that this memory requirement can be significantly reduced: The trie of $(m-1)$-classes can be greatly compressed, and the trie nodes of m-classes can be greatly shortened, if we use more sophisticated schemes to access those tries.)

By making only a few changes to Algorithm E, we can extend it to "Algorithm E$^+$," which counts the Hamiltonian *paths* of each induced subgraph $G\,|\,\{1,\ldots,m\}$ instead of counting the Hamiltonian cycles. The idea is simply to imagine a new vertex '∞', following vertex n, with v — ∞ for $1\le v\le n$. Hamiltonian paths of $G\,|\,\{1,\ldots,m\}$ are then equivalent to Hamiltonian cycles of $G\,|\,\{1,\ldots,m,\infty\}$. (See exercises 2 and 33.) The new vertex '∞' becomes a new member of every frontier. Exercise 207 has the details.

For example, let's go back to Hamilton's original graph G in (26). We get a Hamiltonian m-path in $G_m = G\,|\,\{1,\ldots,m\}$ for $m\le 8$ by taking an appropriate subpath of 4 — 3 — 2 — 1 — 5 — 6 — 7 — 8 — 4, which is the 8-cycle in the middle of (26). These are the *only* m-paths for $m<8$, except that 4 — 3 — 6 — 5 — 1 — 2 also works when $m=6$. By omitting any one edge of the 8-cycle, we get eight 8-paths for $m=8$; and there are two more, one of which is 5 — 1 — 2 — 3 — 6 — 7 — 8 — 4. The total number of Hamiltonian m-paths in G_m, for $m=(2,\,3,\,\ldots,\,20)$, turns out to be respectively (1, 1, 1, 1, 2, 1, 10, 3, 12, 3, 16, 6, 32, 7, 44, 84, 120, 276, 1620).

Wow! Algorithm E$^+$ allows us to go way beyond (40) and to obtain the exact number of *open* knight's tours of size 8×32:

$$20{,}279{,}590{,}726{,}014{,}132{,}421{,}141{,}646{,}018{,}182{,}968{,}888{,}437{,}777,$$
$$268{,}855{,}614{,}013{,}516{,}231{,}312{,}347{,}225{,}658{,}967{,}818{,}240. \qquad (44)$$

(See OEIS A389760 and exercise 209.) It's roughly 6785 times as big as (40). While computing this value, which is `PATH[256]`, hundreds of smaller totals were of course also found, including `PATH[64]`:

$$9{,}795{,}914{,}085{,}489{,}952. \qquad (45)$$

This is the number of open knight's tours on a normal chessboard, first computed by A. Chernov and G. Stertenbrink in 2014 by taking a weighted sum of 136 individual counts for different choices of where to start and stop the tour.

The author's computation of (44) was a bit of an adventure, lasting about 33 days on a machine with 2 terabytes of RAM. Again, periodic behavior began to kick in when m passed 80, as in Fig. 126; but now the tries were about 8.61 times larger. Empirically, the number `PATH[8n]` of $8\times n$ open knight's tours is fairly well approximated by $(.000000028+.0000001n+.000000015n^2)\cdot 526.458^n$ for $16\le n\le 32$. (See exercise 210.)

directed graphs–
football scores
Stanford GraphBase
Ivy League
4-cube

Directed and bidirected graphs. So far we've been discussing Hamiltonian paths and cycles only with respect to garden-variety (undirected) graphs. But they're quite important in *directed* graphs too.

Let's look, for example, at an 8-vertex digraph that can easily be analyzed by hand. It's based on the following matrix of football scores:

	Brown	Columbia	Cornell	Dartmouth	Harvard	Penn	Princeton	Yale
Brown	·	17−0	7−34	0−29	37−52	24−17	23−27	21−27
Columbia	0−17	·	0−41	20−34	6−9	6−21	17−15	7−31
Cornell	34−7	41−0	·	6−11	20−17	21−15	17−14	41−31
Dartmouth	29−0	34−20	11−6	·	17−0	6−16	23−6	27−17
Harvard	52−37	9−6	17−20	0−17	·	20−24	23−20	19−34
Penn	17−24	21−6	15−21	16−6	24−20	·	20−34	10−27
Princeton	27−23	15−17	14−17	6−23	20−23	34−20	·	7−34
Yale	27−21	31−7	31−41	17−27	34−19	27−10	34−7	·

(46)

(The Stanford GraphBase includes data for the scores of all games played between the top 120 college football teams in the USA during the 1990 season — a particularly memorable year. This matrix shows the "Ivy League" games. For example, Brown beat Columbia, 17 to 0, but lost to Cornell, 7 to 34.)

We get a digraph with $\binom{8}{2}$ arcs from (46) by observing who beat whom:

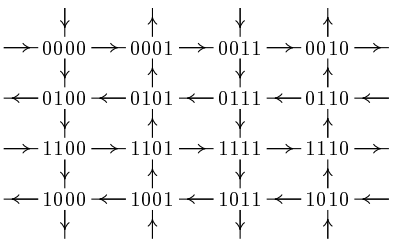
Brown → Columbia, Brown → Penn, Columbia → Princeton, Cornell → Brown, Cornell → Columbia, Cornell → Harvard, Cornell → Penn, Cornell → Princeton, Cornell → Yale, Dartmouth → Brown, Dartmouth → Columbia, Dartmouth → Cornell, Dartmouth → Harvard, Dartmouth → Princeton, Dartmouth → Yale, Harvard → Brown, Harvard → Columbia, Harvard → Princeton, Penn → Columbia, Penn → Dartmouth, Penn → Harvard, Princeton → Brown, Princeton → Penn, Yale → Brown, Yale → Columbia, Yale → Harvard, Yale → Penn, Yale → Princeton. (47)

Does this digraph have a Hamiltonian cycle? If so, it will have to include the arc Columbia → Princeton, because that was Columbia's only victory. It will also have to include Penn → Dartmouth → Cornell, because those were Dartmouth's and Cornell's only defeats. Then Cornell → Yale is also forced, because Yale's only other loss was to Dartmouth. ... We soon discover two solutions, one of which is

Yale → Harvard → Columbia → Princeton → Brown → Penn → Dartmouth → Cornell → Yale

and the other is the answer to exercise 211.

A vast number of interesting digraphs arises when we take an undirected graph and assign an orientation to each edge. Consider, for example, the 4-cube,



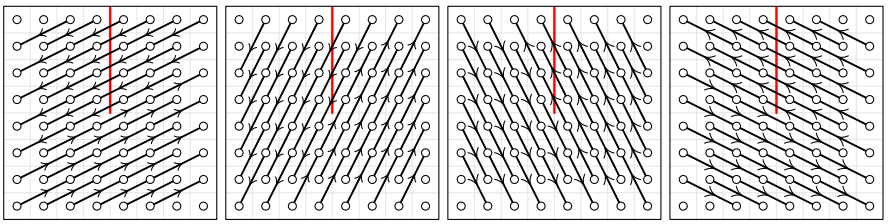
, (48)

whose vertices are the 16 binary vectors of length 4, adjacent when they differ in just one bit position. (As in exercise 7.2.1.1–17, the 4-cube is represented here as a torus, $C_4 \mathbin{\square} C_4$, with arcs wrapping around from top to bottom and left to right.) There are 64 Hamiltonian cycles with this orientation, one of which is

$$0000 \longrightarrow 0001 \longrightarrow 0011 \longrightarrow 0010 \longrightarrow 1010 \longrightarrow 1011 \longrightarrow 1001 \longrightarrow 1101 \longrightarrow 0101 \longrightarrow 0100 \longrightarrow 0110 \longrightarrow 0111 \longrightarrow 1111 \longrightarrow 1110 \longrightarrow 1100 \longrightarrow 1000 \longrightarrow 0000 . \quad (49)$$

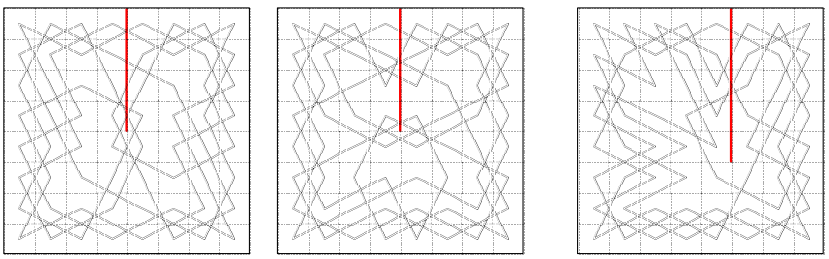
(We know from Eq. 7.2.1.1–(26) that the *non*-oriented 4-cube has 1344 of them.)

A *knight* graph can be oriented in a particularly interesting way, if we insist that the knight always moves counterclockwise with respect to some pivot point. For example, the knight will continually whirl around the center of an ordinary chessboard if and only if its moves have the following orientations:



. (50)

Do such whirling knight's tours exist? Yes indeed; but they're rare. Even rarer are the tours that whirl around a pivot that lies *southeast* of the center(!):



(51)

Only 1120 tours of the former type exist, and only 4 of the latter. The left-hand example is one of 16 that have central symmetry.

It turns out that all 1120 of the Hamiltonian cycles of the digraph defined by (50) have the property that they whirl around the center exactly seven times. Equivalently, exactly seven of their arcs cross the "plumb line" that extends vertically from the center. In fact, the tour shown in the middle example of (51) maintains a fairly steady angular velocity: The number of steps between its seven plumb-line crossings always differs from 64/7 by less than 2. (This example, and its left-right reversal, are the *only* such whirling knight's tours. See exercise 228.)

The concept of whirling tours adds a new, relatively unexplored dimension to the age-old study of the movements of chess pieces. Exercises 227–236 explore some of the most basic questions related to them. Whirling kings are almost as interesting as whirling knights! Many beautiful patterns arise.

4-cube
torus
knight graph
counterclockwise
anticlockwise: see counterclockwise
pivot point
whirling knight's tours
angular velocity
chess pieces
Whirling kings

bidirected graph
oriented signed graph, see bidirected
bidirectional vs bidirected
arrows
angle brackets
introverted edge
extroverted edge
bidirected walk
trail
simple
bidirected cycle
bidirected path

We will soon be discussing Algorithm B, a variant of Algorithm H that quickly finds patterns such as those in (49) and (51). But it turns out that Algorithm B actually does considerably more: It visits all of the Hamiltonian cycles in an arbitrary *bidirected graph*, a vast family of graph structures that includes directed graphs as a very special case. Therefore it behooves us to become familiar with bidirectedness, a concept that cries out to be better known.*

The idea is actually quite simple and intuitive: The arcs of a bidirected graph essentially have *two* arrows in place of one. If u and v are vertices, four kinds of bidirected relationships are therefore possible: either $u \rangle\rangle v$ or $u \langle\langle v$ or $u \rangle\langle v$ or $u \langle\rangle v$. The first two cases are called *directed edges*, and they correspond to the familiar relations $u \longrightarrow v$ and $u \longleftarrow v$. (Our notation uses angle brackets '$\langle$' and '$\rangle$' instead of arrows, because brackets look fairly nice and don't take up much space.) The third case, $u \rangle\langle v$, is called an *introverted edge*; and the remaining case, $u \langle\rangle v$, is called an *extroverted edge*. In words, not symbols, we can verbalize these four cases by saying "u to v" or "u from v" or "u intro v" or "u extro v."

A single edge between u and v actually gives us only three different relations between those vertices, despite having four cases, because $u \rangle\rangle v$ means the same as $v \langle\langle u$. Similarly, $u \rangle\langle v$ means the same as $v \rangle\langle u$, and $u \langle\rangle v$ means the same as $v \langle\rangle u$. On the other hand, sixteen different relations between u and v are actually possible, when $u \neq v$, because we might have any subset of the four cases. For example, we might have $u \langle\rangle v$ and $u \langle\langle v$, but not $u \rangle\rangle v$ and not $u \rangle\langle v$.

A *bidirected walk* is obtained from a sequence of vertices joined by edges, just as in an ordinary graph or directed graph; but we require the arrows next to intermediate vertices to be identical. For example,

$$u \langle\langle x \langle\langle y \langle\rangle z \rangle\rangle y \rangle\rangle x \rangle\rangle v \tag{52}$$

is a bidirected walk of length 6 between u and v. Contrariwise, a sequence like $u \langle\langle x \rangle\langle y$ isn't part of any legal walk; the arrows that surround x don't match.

A walk is said to be a *trail* if its edges are distinct. In an ordinary graph, a walk between two vertices always implies the existence of a trail between those vertices; but (52) shows that we can't always make such an inference in a bidirected graph, because it includes the edge $x \langle\langle y$ twice.

More formally, a sequence

$$u = t_0 \,\mathbb{l}_0\mathbb{l}_1\, t_1 \,\mathbb{l}_1\mathbb{l}_2\, t_2 \,\mathbb{l}_2 \cdots \mathbb{l}_{l-1}\, t_{l-1} \,\mathbb{l}_{l-1}\mathbb{l}_l\, t_l = v, \tag{53}$$

where each $\mathbb{l}_j$ is either $\langle$ or $\rangle$, is a bidirected walk of length l between u and v. It's called *simple* if $\{t_0, \ldots, t_{l-1}\}$ are distinct and $\{t_1, \ldots, t_l\}$ are distinct. A simple bidirected walk is called a *bidirected cycle* if $t_0 = t_l$ and $\mathbb{l}_0 = \mathbb{l}_l$; otherwise it's called a *bidirected path*.

When all of the arcs of a bidirected graph are directed edges, it's exactly the same as a directed graph. When all of the arcs are introverted, the structure

* None of the textbooks on graph theory known to the author at the time this section was written even mention the concept. Furthermore, some of them have unfortunately said that a digraph is "bidirectional" if it has the property that $u \longrightarrow v$ implies $v \longrightarrow u$.

isn't really interesting, because all walks have length 1. But we *can* get really interesting structures when all of the arcs are either introverted or extroverted.

Indeed, suppose G and H are any two graphs on the same set of vertices. Define $u \rangle\langle v$ if u and v are adjacent in G; also define $u \langle\rangle v$ if u and v are adjacent in H. Then introverted and extroverted edges must alternate in any bidirected walk. In particular, there can't be any cycles of odd length.

For example, suppose G is the 4×4 grid on the 16 vertices $\{00, 01, \ldots, 33\}$, and let H be the 4×4 knight graph on those same vertices. Then we have $00 \rangle\langle 01$, $00 \rangle\langle 10$, $01 \rangle\langle 02$, $01 \rangle\langle 11$, $\ldots$, $32 \rangle\langle 33$ from G; and $00 \langle\rangle 12$, $00 \langle\rangle 21$, $01 \langle\rangle 13$, $01 \langle\rangle 20$, $\ldots$, $23 \langle\rangle 31$ from H. This bidirected graph has 180 Hamiltonian cycles, strictly alternating between knight moves and grid moves, one of which is

$$00 \langle\rangle 12 \rangle\langle 22 \langle\rangle 03 \rangle\langle 02 \langle\rangle 23 \rangle\langle 13 \langle\rangle 32 \rangle\langle 33 \langle\rangle 21 \rangle\langle 31 \langle\rangle 10 \rangle\langle 11 \langle\rangle 30 \rangle\langle 20 \langle\rangle 01 \rangle\langle 00. \tag{54}$$

Alternating patterns like this actually appeared very early in the history of knight's tours. A prominent Turkic polymath named Abū Bakr Muḥammad ibn Yaḥyā ibn al-'Abbās al-Ṣūli wrote a book on chess, a few decades after others had devised the tours that we saw in (1) and (2) at the beginning of this section. Besides discussing strategy, he presented several closed tours, including these:

32	35	30	25	8	5	50	55
29	24	33	36	51	56	7	4
34	31	26	9	6	49	54	57
23	28	37	12	1	52	3	48
38	13	22	27	10	47	58	53
19	16	11	64	61	2	43	46
14	39	18	21	44	41	62	59
17	20	15	40	63	60	45	42

37	14	16	35	33	18	24	31
15	36	34	17	19	32	30	25
13	38	48	11	21	26	28	23
39	12	10	49	27	20	22	29
9	42	40	47	61	50	52	63
43	8	46	41	51	60	62	53
45	6	4	59	57	2	64	55
7	44	58	5	3	56	54	1

49	42	40	51	9	34	36	11
47	52	54	45	39	12	14	33
41	50	48	43	37	10	8	35
55	44	46	53	15	32	38	13
61	22	16	63	5	26	28	7
19	56	58	21	31	64	2	25
17	62	60	23	29	6	4	27
59	20	18	57	3	24	30	1

. (55)

[His original treatise has been lost, but these tours were quoted by later authors. See H. J. R. Murray, *A History of Chess* (Oxford, 1913), 171–172, 335–337.] The knight's cycle at the left is remarkable because, as observed by George Jelliss, moves 26–40 are the top-down reflection of moves 11–25. The other two cycles are even more remarkable, because they introduced a completely new idea, with knight moves alternating with the moves of other ancient chesspieces called "fers" and "alfil." (A fers moves one step diagonally; an alfil hops twice as far.) Three astonishing *tours de force*, constructed more than 1100 years ago!

The purpose of Algorithm B below is to visit every Hamiltonian cycle of a given bidirected graph. But what exactly does that mean? In a bidigraph, $a \langle\langle b \langle\rangle c \rangle\langle a$ is quite different from $a \langle\rangle b \rangle\rangle c \rangle\langle a$. So we can't just "visit the cycle (abc)"; we've got to specify also the relations between consecutive vertices.

In general, Algorithm B is supposed to visit every subset of edges that can be expressed as a bidirected cycle having the form (53), where l is the total number of vertices. A similar rule characterizes Hamiltonian paths: They're the subsets that can be expressed as bidirected *paths* according to (53), where $l + 1$ (not l) is the total number of vertices. (See exercise 242.)

grid
knight graph
wazir moves, see grid moves
strictly alternating
grid moves
history
Abū Bakr Muḥammad ibn Yaḥyā ibn al-'Abbās
Murray
Jelliss
top-down reflection
vertical symmetry
alternating moves
fers
alfil
Hamiltonian cycle of a bidigraph
bidigraph: Short for bidirected graph
Hamiltonian path of a bidigraph

directed graphs
Tarjan
Karp
Hamiltonian matching

Any bidirected graph B on n vertices $\{v_1,\ldots,v_n\}$ can be represented as an ordinary graph G on $3n$ vertices $\{v_1^-,v_1,v_1^+,\ldots,v_n^-,v_n,v_n^+\}$, where the edges of G are $v_k^- \mathrel{---} v_k \mathrel{---} v_k^+$ for $1\le k\le n$ together with

$$\begin{aligned} &v_j^- \mathrel{---} v_k^+, \quad \text{if } v_j \langle\langle\ v_k \text{ in } B; && v_j^+ \mathrel{---} v_k^+, \quad \text{if } v_j \rangle\langle\ v_k \text{ in } B;\\ &v_j^- \mathrel{---} v_k^-, \quad \text{if } v_j \langle\rangle\ v_k \text{ in } B; && v_j^+ \mathrel{---} v_k^-, \quad \text{if } v_j \rangle\rangle\ v_k \text{ in } B. \end{aligned} \tag{56}$$

It's easy to see that the Hamiltonian cycles in B correspond one-to-one with the Hamiltonian cycles in G. (This idea, in the special case of directed graphs, is due to R. E. Tarjan; see the famous paper by R. M. Karp, in *Complexity of Computer Computations* (Plenum Press, 1972), pages 98 and 101.) Therefore we don't need a new algorithm; we could use our trusty old Algorithm H to visit all the Hamiltonian cycles of B.

But we can do better than that, because every "unsigned" vertex v_k is always surrounded by v_k^- and v_k^+ when it's in a Hamiltonian cycle. We can dispense with those unsigned vertices by slightly reformulating the problem.

Let $G(B)$ be the (ordinary) graph on $2n$ vertices $\{v_1^-,v_1^+,\ldots,v_n^-,v_n^+\}$ whose edges are specified by (56). A *Hamiltonian matching* of $G(B)$ is a set of n disjoint edges for which the addition of the n further edges $v_k^- \mathrel{---} v_k^+$ for $1\le k\le n$ will form a $2n$-cycle. Hamiltonian matchings of $G(B)$ correspond naturally to Hamiltonian cycles of B; and we can visit them all by using almost the same method that worked before, by simply adapting Algorithm H to the new setup.

Indeed, Algorithm B almost writes itself, because most of the former steps need little or no change. As usual, it's instructive to work out the details.

Let's state Algorithm B first, and discuss its fine points later.

The bidirected graph input to Algorithm B, like the undirected graph G input to Algorithm H, has vertices v identified by integers, $0\le v<n$. But Algorithm B actually works with the undirected graph $G(B)$, which has $2n$ vertices; vertex v of B corresponds to two vertices, $v^-=2v$ and $v^+=2v+1$, of $G(B)$.

Algorithm B (*All directed or bidirected Hamiltonian cycles*). Given a bidirected graph B on the n vertices $\{0,1,\ldots,n-1\}$, this algorithm uses the data structures discussed above to visit every Hamiltonian matching of the related graph $G(B)$. During every visit, the chosen edges are `EU[`k`]` $\mathrel{---}$ `EV[`k`]` for $0\le k<n$.

B1. [Initialize.] Set up the `NBR` and `ADJ` arrays (see exercise 250). Set the global variables $a\leftarrow e\leftarrow i\leftarrow l\leftarrow$ `T` $\leftarrow 0$. Also set `VIS[`v`]` $\leftarrow$ `IVIS[`v`]` $\leftarrow v$ and `MATE(`v`)` $\leftarrow -1$ for $0\le v<2n$. Set `LLINK[`$2n$`]` $\leftarrow$ `RLINK[`$2n$`]` $\leftarrow$ `S` $\leftarrow 2n$. Finally, for every vertex v with `DEG(`v`)` $=1$, set `TRIG[T]` $\leftarrow v$ and `T` $\leftarrow$ `T`$+1$.

B2. [Choose the root vertex.] Let `CURV` be a vertex of minimum degree, and set $d\leftarrow$ `DEG(CURV)`. If $d<1$, terminate (there is no Hamiltonian cycle). If $d=1$, set `CURV` $\leftarrow -1$, do step B4, and go to B6.

B3. [Force a root edge.] Set `CURU` $\leftarrow$ `NBR[CURV][`$d-1-i$`]` (the last yet-untried neighbor of `CURV`), and set `EU[0]` $\leftarrow$ `CURU`, `EV[0]` $\leftarrow$ `CURV`, $e\leftarrow 1$. Then get started with `CURU` joined to `CURV`, as explained in exercise 251, and go to B6.

B4. [Record the state.] Set $\texttt{CV}(l) \leftarrow \texttt{CURV}$, $\texttt{I}(l) \leftarrow i$, $\texttt{D}(l) \leftarrow d$, $\texttt{E}(l) \leftarrow e$, $\texttt{S}(l) \leftarrow \texttt{S}$, $\texttt{T}(l) \leftarrow \texttt{T}$. For $0 \le k < \texttt{S}$, set $u \leftarrow \texttt{VIS}[k]$ and $\texttt{SAVE}[2nl+u] \leftarrow \big(\texttt{MATE}(u), \texttt{DEG}(u)\big)$ (thereby leaving "holes" in the SAVE stack). Then set $u \leftarrow \texttt{RLINK}[2n]$; while $u \neq 2n$, set $\texttt{ACTIVE}[a] \leftarrow u$, $a \leftarrow a+1$, $u \leftarrow \texttt{RLINK}[u]$. Finally set $\texttt{A}(l) \leftarrow a$.

B5. [Choose an edge.] Set $\texttt{CURU} \leftarrow \texttt{NBR}[\texttt{CURV}][i]$, $\texttt{CURT} \leftarrow \texttt{MATE}(\texttt{CURU})$, $\texttt{CURW} \leftarrow \texttt{MATE}(\texttt{CURV})$, $\texttt{EU}[e] \leftarrow \texttt{CURU}$, $e \leftarrow e+1$. Call $\text{remarc}\big(\texttt{NBR}[\texttt{CURU}][k], \texttt{CURU}\big)$ for k decreasing from $\texttt{DEG}(\texttt{CURU})-1$ to 0. If $\texttt{CURT} < 0$ (CURU is bare), makeinner(CURU), activate(CURU$\oplus$1), and makemates(CURU$\oplus$1, CURW). Otherwise (CURU is outer), makemates(CURT, CURW) and deactivate(CURU).

B6. [Begin trigger loop.] Set $j \leftarrow 0$ if $l = 0$, else $j \leftarrow \texttt{T}(l-1)$. Go to B10 if $j = \texttt{T}$.

B7. [Clothe $\texttt{TRIG}[j]$.] Set $v \leftarrow \texttt{TRIG}[j]$, and go to B9 if $\texttt{MATE}(v) \ge 0$ or $\texttt{IVIS}[v] \ge \texttt{S}$ (v isn't bare). Otherwise go to B15 if $\texttt{DEG}(v) = 0$ (a Hamiltonian cycle is impossible). Set $u \leftarrow \texttt{NBR}[v][0]$, $\texttt{EU}[e] \leftarrow u$, $\texttt{EV}[e] \leftarrow v$, $e \leftarrow e+1$, makeinner(v), and activate($v \oplus 1$). Call $\text{remarc}\big(\texttt{NBR}[u][k], u\big)$ for k decreasing from $\texttt{DEG}(u) - 1$ to 0, except when $\texttt{NBR}[u][k] = v$.

B8. [Take stock.] (We've just joined v to its only neighbor, u.) Update the data structures as described in exercise 252, based on whether $\texttt{MATE}(u) < 0$.

B9. [End trigger loop?] Set $j \leftarrow j+1$, and return to B7 if $j < \texttt{T}$.

B10. [Enter new level.] Set $l \leftarrow l+1$, and go to B13 if $e \ge n-1$.

B11. [Choose vertex for branching.] Set $\texttt{CURV} \leftarrow \texttt{RLINK}[2n]$, $d \leftarrow \texttt{DEG}(\texttt{CURV})$, $k \leftarrow \texttt{RLINK}[\texttt{CURV}]$. While $k \neq 2n$, if $\texttt{DEG}(k) < d$ reset $\texttt{CURV} \leftarrow k$ and $d \leftarrow \texttt{DEG}(k)$; set $k \leftarrow \texttt{RLINK}[k]$. Go to B14 if $d = 0$. Otherwise set $\texttt{EV}[e] \leftarrow \texttt{CURV}$ and $\texttt{T} \leftarrow \texttt{T}(l-1)$.

B12. [Make CURV inner.] Call $\text{remarc}\big(\texttt{NBR}[\texttt{CURV}][k], \texttt{CURV}\big)$ for $0 \le k < d$ (thereby removing CURV from its neighbors' lists). Then deactivate(CURV), set $i \leftarrow 0$, and go to B4.

B13. [Visit a solution.] Set $u \leftarrow \texttt{LLINK}[2n]$ and $v \leftarrow \texttt{RLINK}[2n]$. Go to B14 if $\texttt{ADJ}[u][v] = \infty$. Otherwise set $\texttt{EU}[e] \leftarrow u$, $\texttt{EV}[e] \leftarrow v$, $e \leftarrow n$. Now visit the n-cycle defined by arrays EU and EV. (See exercise 253.)

B14. [Back up.] Terminate if $l = 0$. Otherwise set $l \leftarrow l-1$.

B15. [Undo changes.] Set $d \leftarrow \texttt{D}(l)$ and $i \leftarrow \texttt{I}(l)+1$. Go to B14 if $i \ge d$. Otherwise set $\texttt{I}(l) \leftarrow i$, $e \leftarrow \texttt{E}(l)$, $k \leftarrow (l > 0 ?\ \texttt{A}(l-1) : 0)$, $a \leftarrow \texttt{A}(l)$, $v \leftarrow 2n$. While $k < a$, set $u \leftarrow \texttt{ACTIVE}[k]$, $\texttt{RLINK}[v] \leftarrow u$, $\texttt{LLINK}[u] \leftarrow v$, $v \leftarrow u$, $k \leftarrow k+1$. Then set $\texttt{RLINK}[v] \leftarrow 2n$, $\texttt{LLINK}[2n] \leftarrow v$, $\texttt{S} \leftarrow \texttt{S}(l)$, $\texttt{T} \leftarrow \texttt{T}(l)$. For $0 \le k < \texttt{S}$, set $u \leftarrow \texttt{VIS}[k]$ and $\big(\texttt{MATE}(u), \texttt{DEG}(u)\big) \leftarrow \texttt{SAVE}[2nl+u]$. Finally set $\texttt{CURV} \leftarrow \texttt{CV}(l)$. Go to B5 if $l > 0$.

B16. [Advance at root level.] Set $\texttt{CURU} \leftarrow \texttt{EU}[0]$; remarc(CURV, CURU) and remarc(CURU, CURV). (The previous edge CURU — CURV disappears.) If $\texttt{DEG}(\texttt{CURU}) = 1$, set $\texttt{TRIG}[0] \leftarrow \texttt{CURU}$ and $\texttt{T} \leftarrow 1$; otherwise set $\texttt{T} \leftarrow 0$. If $\texttt{DEG}(\texttt{CURV}) = 1$, set $\texttt{TRIG}[\texttt{T}] \leftarrow \texttt{CURV}$ and $\texttt{T} \leftarrow \texttt{T}+1$. Go to B3 if $\texttt{T} = 0$. Otherwise set $\texttt{CV}(0) \leftarrow -1$, $e \leftarrow 0$, and go to B6. ∎

makeinner
remarc
activate
deactivate
makemates
perfect matchings
three-letter words
SSDIHAM

Algorithm B invokes the miniroutines makeinner, remarc, activate, deactivate, and makemates, which were defined in (19)–(23) for use in Algorithm H. However, two changes are necessary: The condition '$d = 2$' in (20) should become '$d = 1$'; and the value 'n' in (21) should become '$2n$' (three times).

The main novelty here is the shift of focus to finding a Hamiltonian *matching* of the special graph $G(B)$, which has two vertices v^- and v^+ for each vertex of B. This matching problem replaces the original goal of Algorithm H, which was to find a Hamiltonian *cycle* of an ordinary graph.

For example, let's consider a simple case where the given graph B has just three vertices $\{0, 1, 2\}$ and six bidirected edges,

$$0 \langle\langle 1, \quad 0 \langle\rangle 1, \quad 0 \rangle\rangle 2, \quad 0 \rangle\langle 2, \quad 1 \langle\langle 2, \quad 1 \rangle\rangle 2. \tag{57}$$

In this case, by (56), $G(B)$ has six vertices $\{0^-, 0^+, 1^-, 1^+, 2^-, 2^+\}$ and six (undirected) edges, which happen to form a cycle:

$$0^- — 1^+ — 2^- — 0^+ — 2^+ — 1^- — 0^-. \tag{58}$$

A 6-cycle always has two perfect matchings, and they turn out to be Hamiltonian; but Algorithm B doesn't know this, so let's watch how it discovers the two solutions. No vertex of $G(B)$ has degree less than 2; therefore step B3 chooses a "root edge" to build upon. We shall assume that the root edge is '$0^- — 1^-$'. The final cycle, after we add the required edges $0^- — 0^+$, $1^- — 1^+$, $2^- — 2^+$ to our Hamiltonian matching, will therefore contain the subpath

$$0^+ — 0^- — 1^- — 1^+. \tag{59}$$

This is the partial solution that is present when the algorithm first reaches step B6. At that point 0^- and 1^- are inner vertices; the endpoints 0^+ and 1^+ are outer vertices (and they are mates). The remaining vertices $\{2^-, 2^+\}$ are currently bare. However, vertex 2^+ has degree 1, because the edge $2^+ — 1^-$ that joins it to an inner vertex has gone away. Therefore 2^+ goes onto the trigger list, and step B7 soon extends (59) to

$$2^- — 2^+ — 0^+ — 0^- — 1^- — 1^+. \tag{60}$$

The algorithm now gets to level $l = 1$, and goes to step B13. Aha — the endpoints of (60) complete a cycle! Hence $\{2^+ — 0^+,\ 0^- — 1^-,\ 1^+ — 2^-\}$ is the first Hamiltonian matching found. (See exercise 254 for the sequel.)

In practice, Algorithm B is almost always applied to the special case in which the input is just an ordinary directed graph, without introverted or extroverted edges. In that case steps B1 and B13, which govern input and output, can be considerably simplified, and the interface with users becomes much more intuitive. (Exercise 256 has the details.) It's therefore wise to have two implementations: one for directed graphs only, and another for bidirected graphs in general.

arcowokidubayetajugnughimoownileicelfezipareveexisigia

— SSDIHAM (26 October 2025)

SAT encodings. We've mostly been studying algorithms that visit *every* Hamiltonian cycle of a given graph, or algorithms that count their total number.

But sometimes our goal is simply to find a *single* such cycle, or to prove that no such cycle exists. In such cases the powerful tools of SAT technology (Section 7.2.2.2) are often the method of choice.

SAT–relaxing
cycle cover
counterexample-guided abstraction refinement
CEGAR
lazy clause generation
asymmetry
at-least-one

It's not difficult to formulate sets of Boolean clauses that are satisfiable if and only if a given graph is Hamiltonian. For example, we can encode the fact that each vertex of an n-vertex graph corresponds to an integer k, with $0 \le k < n$, and that vertices k and $(k+1) \bmod n$ are adjacent.

But an indirect approach turns out to work much better on difficult cases. And fortunately, this alternative method is quite simple and instructive. It begins by *relaxing* the constraints: Instead of using a SAT solver to find a *single* cycle that involves every vertex, we ask it only to find a "cycle cover," namely a set of one *or more* cycles in which every vertex appears exactly once. The cycle cover problem has a particularly straightforward encoding into SAT clauses, and such a cover can usually be found quickly.

We'll study the precise details of cycle cover encoding shortly. Its Boolean variables are the pairs uv, one for each arc $u \longrightarrow v$ in the graph. (As usual, every edge $u \longrightarrow v$ of our graph is represented as a pair of arcs, $u \longrightarrow v$ and $v \longrightarrow u$.)

If there is no cycle cover, the graph is certainly not Hamiltonian. And if we've found a cycle cover with just one cycle, the graph certainly *is* Hamiltonian. Furthermore — and this is the point — a cycle cover with two or more cycles gives us information about how to construct further clauses, which must necessarily be satisfied if we do have a Hamiltonian graph. We can append those new clauses to the former ones and ask the solver to try again. (This approach has become known as "*counterexample-guided abstraction refinement*," or CEGAR for short.)

Indeed, if C is a cycle that doesn't include every vertex, every Hamiltonian cycle must include an arc from a vertex in C to a vertex that's not in C. So we shall tell our SAT solver to add the clause

$$\operatorname{cut}(C) \;=\; \bigvee \{uv \mid u \in C,\; v \notin C,\; u \longrightarrow v\} \tag{61}$$

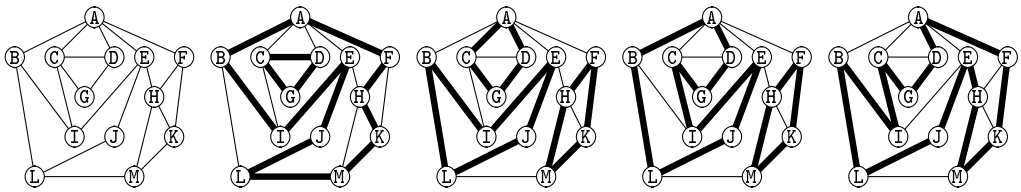
to the constraints that it already has. This will ensure that subsequent cycle covers never include C, nor will they include any other cycle on those same vertices.

Notice that we can imagine a huge set of clauses,

$$(\text{clauses that define a cycle cover}) \wedge \operatorname{cut}(C_1) \wedge \operatorname{cut}(C_2) \wedge \cdots \wedge \operatorname{cut}(C_N), \tag{62}$$

where C_1, C_2, $\ldots$, C_N are all the subsets of 3 to $n-3$ vertices that occur in a cycle that's part of some cycle cover. The solutions to this gigantic problem (62) are precisely the Hamiltonian cycles of our graph. Therefore the CEGAR method that we're using has also been called "lazy clause generation."

We shall use three kinds of clauses to define a cycle cover, in such a way that a set C of arcs will be a cycle cover if and only if the corresponding Boolean values $[uv \in C]$ satisfy all of the clauses of each kind. (i) The *asymmetry* clauses say that we cannot have both uv and vu, for any arc $u \longrightarrow v$ of the graph. (ii) The *at-least-one* clauses say that at least one arc leads out from some vertex v, or



(a) (b) (c) (d) (e)

at-most-one 2-cycles

Fig. 127. An example graph and four of its cycle covers.

that at least one arc leads in to some vertex v. (iii) The *at-most-one* clauses are similar, but they say that no two such arcs exist.

For example, consider the graph in Fig. 127(a), which has 13 vertices and 21 edges, hence 42 arcs. The 42 Boolean variables are AB, AC, AD, AE, AF, BA, BI, BL, ..., ML. There are 21 asymmetry clauses, which prevent cycles of length 2:

$$(\overline{\mathtt{AB}} \vee \overline{\mathtt{BA}}) \wedge (\overline{\mathtt{AC}} \vee \overline{\mathtt{CA}}) \wedge \cdots \wedge (\overline{\mathtt{KM}} \vee \overline{\mathtt{MK}}) \wedge (\overline{\mathtt{LM}} \vee \overline{\mathtt{ML}}). \tag{63}$$

There are 26 at-least-one clauses, two for each vertex, namely

$$\begin{aligned} (\mathtt{AB} \vee \mathtt{AC} \vee \mathtt{AD} \vee \mathtt{AE} \vee \mathtt{AF}) \ \wedge\ (\mathtt{BA} \vee \mathtt{CA} \vee \mathtt{DA} \vee \mathtt{EA} \vee \mathtt{FA});\\ (\mathtt{BA} \vee \mathtt{BI} \vee \mathtt{BL}) \ \wedge\ (\mathtt{AB} \vee \mathtt{IB} \vee \mathtt{LB});\\ \cdots\\ (\mathtt{MH} \vee \mathtt{MK} \vee \mathtt{ML}) \ \wedge\ (\mathtt{HM} \vee \mathtt{KM} \vee \mathtt{LM}). \end{aligned} \tag{64}$$

(Exercise 290 shows that only half of these are really necessary; but we get better results by requiring them all.) Finally, for the at-most-one clauses we shall use $(d-1)d$ binary clauses for each vertex of degree d:

$$\begin{aligned} &(\overline{\mathtt{AB}} \vee \overline{\mathtt{AC}}) \wedge (\overline{\mathtt{AB}} \vee \overline{\mathtt{AD}}) \wedge (\overline{\mathtt{AB}} \vee \overline{\mathtt{AE}}) \wedge (\overline{\mathtt{AB}} \vee \overline{\mathtt{AF}}) \wedge (\overline{\mathtt{AC}} \vee \overline{\mathtt{AD}}) \wedge\\ &\qquad (\overline{\mathtt{AC}} \vee \overline{\mathtt{AE}}) \wedge (\overline{\mathtt{AC}} \vee \overline{\mathtt{AF}}) \wedge (\overline{\mathtt{AD}} \vee \overline{\mathtt{AE}}) \wedge (\overline{\mathtt{AD}} \vee \overline{\mathtt{AF}}) \wedge (\overline{\mathtt{AE}} \vee \overline{\mathtt{AF}});\\ &(\overline{\mathtt{BA}} \vee \overline{\mathtt{CA}}) \wedge (\overline{\mathtt{BA}} \vee \overline{\mathtt{DA}}) \wedge (\overline{\mathtt{BA}} \vee \overline{\mathtt{EA}}) \wedge (\overline{\mathtt{BA}} \vee \overline{\mathtt{FA}}) \wedge (\overline{\mathtt{CA}} \vee \overline{\mathtt{DA}}) \wedge\\ &\qquad (\overline{\mathtt{CA}} \vee \overline{\mathtt{EA}}) \wedge (\overline{\mathtt{CA}} \vee \overline{\mathtt{FA}}) \wedge (\overline{\mathtt{DA}} \vee \overline{\mathtt{EA}}) \wedge (\overline{\mathtt{DA}} \vee \overline{\mathtt{FA}}) \wedge (\overline{\mathtt{EA}} \vee \overline{\mathtt{FA}});\\ &(\overline{\mathtt{BA}} \vee \overline{\mathtt{BI}}) \wedge (\overline{\mathtt{BA}} \vee \overline{\mathtt{BL}}) \wedge (\overline{\mathtt{BI}} \vee \overline{\mathtt{BL}}) \ \wedge\ (\overline{\mathtt{AB}} \vee \overline{\mathtt{IB}}) \wedge (\overline{\mathtt{AB}} \vee \overline{\mathtt{LB}}) \wedge (\overline{\mathtt{IB}} \vee \overline{\mathtt{LB}});\\ &\qquad\qquad \cdots\\ &(\overline{\mathtt{MH}} \vee \overline{\mathtt{MK}}) \wedge (\overline{\mathtt{MH}} \vee \overline{\mathtt{ML}}) \wedge (\overline{\mathtt{MK}} \vee \overline{\mathtt{ML}}) \ \wedge\ (\overline{\mathtt{HM}} \vee \overline{\mathtt{KM}}) \wedge (\overline{\mathtt{HM}} \vee \overline{\mathtt{LM}}) \wedge (\overline{\mathtt{KM}} \vee \overline{\mathtt{LM}}). \end{aligned} \tag{65}$$

(Exercise 291 discusses other ways to enforce the at-most-one constraints.)

When clauses (63)–(65) are presented to a SAT solver such as Algorithm 7.2.2.2C, it might find the solution

$$\begin{array}{llllllllllllll} \overline{\mathtt{AB}} & \overline{\mathtt{AC}} & \overline{\mathtt{AD}} & \overline{\mathtt{AE}} & \mathtt{AF} & \mathtt{BA} & \overline{\mathtt{BI}} & \overline{\mathtt{BL}} & \overline{\mathtt{CA}} & \overline{\mathtt{CD}} & \mathtt{CG} & \overline{\mathtt{CI}} & \overline{\mathtt{DA}} & \mathtt{DC} \\ \overline{\mathtt{DG}} & \overline{\mathtt{EA}} & \overline{\mathtt{EH}} & \mathtt{EI} & \overline{\mathtt{EJ}} & \overline{\mathtt{FA}} & \mathtt{FH} & \overline{\mathtt{FK}} & \overline{\mathtt{GC}} & \mathtt{GD} & \overline{\mathtt{HE}} & \overline{\mathtt{HF}} & \mathtt{HK} & \overline{\mathtt{HM}} \\ \mathtt{IB} & \overline{\mathtt{IC}} & \overline{\mathtt{IE}} & \mathtt{JE} & \overline{\mathtt{JL}} & \overline{\mathtt{KF}} & \overline{\mathtt{KH}} & \mathtt{KM} & \overline{\mathtt{LB}} & \mathtt{LJ} & \overline{\mathtt{LM}} & \overline{\mathtt{MH}} & \overline{\mathtt{MK}} & \mathtt{ML}, \end{array} \tag{66}$$

which gives us the cycle cover in Fig. 127(b). It corresponds to two cycles, $C_1 = (\mathtt{AFHKMLJEIB})$ and $C_2 = (\mathtt{CGD})$; so we add two more clauses,

$$\mathrm{cut}(C_1) = (\mathtt{AC} \vee \mathtt{AD} \vee \mathtt{IC}), \tag{67}$$

$$\mathrm{cut}(C_2) = (\mathtt{CA} \vee \mathtt{CI} \vee \mathtt{DA}), \tag{68}$$

and start the solver again. (The solver isn't being restarted "from scratch"; we allow it to remember any other clauses that it has already learned when processing (63)–(65), if it chooses to do so.)

On this second round the solver will not find Fig. 127(b) again; but it might come up with Fig. 127(c), which has *three* cycles. Notice that a solution with t cycles might be discovered in 2^t ways, because each cycle can be traversed in two directions. Let's assume for definiteness that cycles $C_1 = (\texttt{ACGD})$, $C_2 = (\texttt{BLJEI})$, and $C_3 = (\texttt{FKMH})$ have been found.

At this point we could launch round 3 by adding the clauses $\text{cut}(C_1)$, $\text{cut}(C_2)$, and $\text{cut}(C_3)$. But there's a better way! Cycles C_1 and C_2 can be *merged* into the single cycle $C_1' = (\texttt{ABLJEICGD})$ shown in Fig. 127(d), by swapping out the arcs $\{\texttt{AC}, \texttt{IB}\}$ and replacing them by $\{\texttt{AB}, \texttt{IC}\}$. In practice, merging of cycles is often possible, and it leads to significant speedups. We can merge successfully whenever two adjacent vertices of one cycle (in this case `A` and `C`) are respectively adjacent to two vertices of another cycle (in this case `B` and `I`).

merged
SUCC[v]
PRED[v]
CID[v]
sparse-set representation
CYC[0]
CLOC[c]
HEAD[c]

Let's therefore add two further clauses

$$\text{cut}(C_1') = (\texttt{AF} \vee \texttt{EH} \vee \texttt{LM}), \tag{69}$$

$$\text{cut}(C_3) = (\texttt{FA} \vee \texttt{HE} \vee \texttt{ML}), \tag{70}$$

and fire up the solver a third time. If we're lucky, it will now find the desired Hamiltonian cycle, Fig. 127(e), which happens to be unique. End of quest, we're done. (See exercise 292 for other scenarios.)

What data structures make it relatively easy to deal with cycle covers such as this? If there are n vertices, numbered 0 to $n-1$, it turns out to be convenient to have two arrays, SUCC[v] and PRED[v], for $0 \le v < n$, indicating the next or previous vertex in v's cycle, as well as an array CID[v] to tell the number of the cycle to which v belongs. (Cycles are numbered 1, 2, ..., t.) For example, the three cycles found in round 2 of the example above would be represented thus:

v =	0	1	2	3	4	5	6	7	8	9	10	11	12
NAME[v] =	A	B	C	D	E	F	G	H	I	J	K	L	M
SUCC[v] =	2	11	6	0	8	10	3	5	1	4	12	9	7
PRED[v] =	3	8	0	6	9	7	2	12	4	11	5	1	10
CID[v] =	1	2	1	1	2	3	1	3	2	2	3	2	3

(71)

Two arrays provide a sparse-set representation of the currently active cycles, which are CYC[0], ..., CYC[$t-1$]; CLOC[c] is the current location of cycle c in the CYC array. We also keep track of HEAD[c], an arbitrary vertex in cycle c.

The merging process absorbs one cycle into another, after which the absorbed cycle essentially disappears. For example, (71) will be accompanied by the values CYC[0] = 1, CYC[1] = 2, CYC[2] = 3, $t = 3$, CLOC[1] = 0, CLOC[2] = 1, CLOC[3] = 2, HEAD[1] = 0, HEAD[2] = 1, HEAD[3] = 5, before C_1 and C_2 are combined to form C_1'. But afterwards we'll have CYC[0] = 1, CYC[1] = 3, $t = 2$, CLOC[1] = 0, CLOC[3] = 1, SUCC[0] = 1, SUCC[8] = 2, PRED[1] = 0, PRED[2] = 8; and all occurrences of '2' in the CID array will have been changed to '1'.

If the given graph has m edges, there are $2m$ Boolean variables in our SAT clauses. It will be convenient to number them from 2 to $2m+1$, not from 1 to $2m$ and not from 0 to $2m-1$, because (i) 0 is useful as a special "sentinel" value; (ii) we can arrange things so that variable uv is number k if and only if variable vu is number $k \oplus 1$.

0-origin indexing
sentinel

We shall work with an adjacency matrix ADJ, whose rows and columns are indexed by vertices of the graph. If u is not adjacent to v, ADJ[u][v] $= 0$; otherwise ADJ[u][v] is the number of the Boolean variable uv that corresponds to the arc $u \longrightarrow v$.

adjacency matrix

Recall that in Section 7.2.2.2 the literals for Boolean variable k were represented internally by the numbers $2k$ and $2k+1$, where literal $2k$ was "positive" and literal $2k+1$ was "negative." (See 7.2.2.2–(57).) Thus the $4m$ possible literals are numbered from 4 to $4m+3$.

positive
negative

OK! With these data structures in mind, we're ready to pin down the details of the cut clause method:

Algorithm C (*CEGAR cuts*). Given a graph G and a SAT solver, this algorithm either finds a Hamiltonian cycle or proves that G doesn't have any.

C1. [Initialize.] Set ADJ[u][v] $\leftarrow 0$ for $0 \le u, v < n$, where n is the number of vertices. Then set $N \leftarrow 2$ and, for each $u \longrightarrow v$ with $u < v$, set ADJ[u][v] $\leftarrow N$, ADJ[v][u] $\leftarrow N+1$, $N \leftarrow N+2$. (The Boolean variables are $[2\,..\,N)$.)

asymmetry clauses

C2. [Create basic clauses.] Contribute the asymmetry clauses, the at-least-one clauses, and the at-most-one clauses to the solver, as described above, thus specifying the constraints of a cycle cover. (See exercise 296.)

at-least-one clauses
at-most-one clauses
cycle cover

C3. [Solve.] Run the solver. Stop (unsuccessfully) if the clauses are unsatisfiable.

C4. [Find the cycles.] For each true variable uv in the solution, set SUCC[u] $\leftarrow v$ and PRED[v] $\leftarrow u$. Also set CID[v] $\leftarrow 0$ for $0 \le v < n$. Then set $t \leftarrow v \leftarrow 0$, and do the following while $v < n$: "If CID[v] $= 0$, $t \leftarrow t+1$, CYC[$t-1$] $\leftarrow t$, CLOC[t] $\leftarrow t-1$, HEAD[t] $\leftarrow v$, CID[v] $\leftarrow t$, $u \leftarrow$ SUCC[v], and repeatedly set CID[u] $\leftarrow t$, $u \leftarrow$ SUCC[u] until $u = v$. Then set $v \leftarrow v+1$."

C5. [Done?] Stop (successfully) if $t = 1$, because SUCC defines an n-cycle.

C6. [Merge?] Use the algorithm of exercise 298 to merge adjacent cycles and reduce t, until no further merging is possible.

C7. [Done?] Stop (successfully) if $t = 1$, because SUCC defines an n-cycle.

C8. [Add cut clauses.] Contribute the clauses $\mathrm{Cut}(C_j)$ to the solver for $0 \le j < t$, where C_j denotes the vertices of CYC[j]. If $t > 2$, also contribute $\mathrm{Cut}(\sim C_j)$, for $0 \le j < t$. (See (61) and exercise 297.) Stop (unsuccessfully) if any of those clauses have size less than 2. Otherwise return to step C3. ∎

randomization

A SAT solver generally uses randomization, so the behavior of Algorithm C depends significantly on its source of random numbers. For example, when it was applied to the 16×16 knight graph with nine different random seeds, it produced cycle covers with the following nine sets of cycle lengths: $\{46, 154, 4, 10, 4, 6, 10, 18, 4\}$, $\{56, 182, 4, 4, 6, 4\}$, $\{232, 14, 10\}$, $\{198, 16, 26, 4, 4, 8\}$, $\{244, 4, 8\}$, $\{54, 20, 44, 120, 8, 6, 4\}$, $\{64, 4, 164, 6, 6, 4, 4, 4\}$, $\{192, 16, 20, 14, 6, 8\}$, $\{76, 134, 8, 14, 24\}$.

In all nine cases, step C6 was able to merge these cycles into a single cycle of length 256; hence there was immediate success at step C7. The running time was about 201 kilomems to get started and to set up the basic clauses in steps C1 and C2, followed by between 52 Kμ and 107 Kμ (median 64 Kμ) to run the solver in step C3, followed by between 4 Kμ and 8 Kμ (median 6.5 Kμ) to do the merging in steps C4–C6. (By contrast, Algorithm H spends about 75 Kμ to initialize its data structures for the 16×16 knight graph in step H1, then another 88 Kμ to search, until finding its first solution in step H13.)

SGB graph
all_perms
permutations
adjacent interchanges
change-ringing
Warnsdorf's rule
Fleischner
unique Hamiltonian cycle
yottamems
Grinberg
petamems
4-cycles
Heule

Of course Algorithm C was not especially intended for use with knight graphs, because knight's tours are so abundant. But there are plenty of graphs that have zillions of Hamiltonian cycles, yet Algorithm H spins its wheels endlessly before finding a single one of them. For example, consider the SGB graph *all_perms*$(7, 0)$, whose 5040 vertices are the permutations of seven elements and whose 15120 edges represent adjacent interchanges. Its Hamiltonian cycles solve the "change-ringing" problem of Section 7.2.1.2, and we've already observed that Warnsdorf's rule (Algorithm W) has a tough time with this graph. But Algorithm H is much, much worse: It's typically working at level 3600 or so; and even after a teramem of computation, it still hasn't backtracked to level 2192, in order to reconsider its initial choices of 2193 edges! On the other hand, Algorithm C finds a solution quickly, in about 400 megamems. (More precisely, nine runs of Algorithm C with different random seeds took respectively (723, 375, 384, 138, 395, 399, 395, 396, 378) Mμ. The cycle covers always had about 80 cycles before merging. Then merging reduced them to a single cycle, except in the fifth run — which ended with a 5036-cycle and a 4-cycle, generating two cut clauses of size 16 before trying again and succeeding on round 2.)

Fleischner's graph, a 338-vertex supergraph of the graph F in Table 1, is another stunning success story for Algorithm C. As mentioned above, its vertices all have degree 4 or 14, and it has a unique Hamiltonian cycle. Random sampling indicates that Algorithm H will need more than 10^{25} mems to find that cycle — yes, more than 10 yottamems! But nine random runs of Algorithm C succeeded in just (93, 111, 114, 127, 99, 105, 107, 85, 77) *mega*mems, using (74, 82, 79, 86, 81, 74, 77, 69, 75) rounds of SAT solving and generating (1190, 1206, 1220, 1244, 1210, 1132, 1168, 1120, 1198) cut clauses of average length about 18.

Grinberg's cubic graph on 268 vertices, discussed in the answer to exercise 21, is another noteworthy example. Algorithm H is able to prove that it has no Hamiltonian cycles, but only after a weeks-long computation — 1.27 petamems. Algorithm C doesn't find it easy either. This graph has no 4-cycles; therefore step C6 is never able to merge two cycles together. [*Temporary note:* I'm currently trying to apply Algorithm C to Grinberg's recalcitrant graph, in order to see if this task too will need weeks! After 50 teramems and 700,000 rounds, it still shows no signs of success; and it ran out of memory, because cut clauses accumulate on each round and I'd allocated only the default memsize of 2^{26}. I'll try again. But meanwhile I've learned that a *different* SAT-based method, due to Marijn Heule, handles it nicely. So I'll report on that soon.]

Exercise 303 discusses Algorithm C's behavior on the benchmarks of Table 1.

***Color coding.** color coding–

Who knows what I might eventually decide to say next? Please stay tuned.

History. There's something inherently satisfying about a path that "hits all the bases." For example, the ancient artist who carved the pattern of Fig. 777 in a mammoth's tusk might well have been trying to create a Hamiltonian-like path in an implicit grid graph, before the dawn of recorded history!

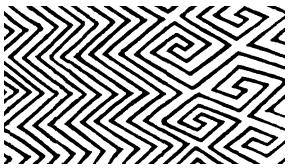


Fig. 777. A pattern from the Old Stone Age. This detail from a Paleolithic ivory carving, found near the present-day village of Mizyn in northern Ukraine, illustrates interesting ways to cover a grid, rotated 45°, with a nonintersecting path.

Of course we cannot read the minds of people who lived c. 15,000 B.C.; but we certainly can admire the wonderful sophistication that's evident in this fascinating artifact. [See M. Rudyns'kyj, *Industrie en os de la station paléolithique de Mizyn, interprétée par Fedir Vovk* (Kyjiv: Vseukraïns'ka Akademiîa Nauk, Kabinet Antropolohiĭ im. F. Vovka, 1931), 66 pages, 32 plates.]

Fast forward now to 750 B.C., by which time "meander friezes" had become well developed in many cultures, especially in Greek pottery (see exercise 360).

A few hundred years later, another kind of Hamiltonian pattern appeared on icosahedra, as we saw near the beginning of this section. And considerable research on knight's tours began shortly after 800 A.D., as we've seen in (1) and (2).

But let's jump to the computer age. The first reasonably successful algorithm for visiting all Hamiltonian cycles of a given graph was published by S. M. Roberts and B. Flores in *CACM* **9** (1966), 690–694; **10** (1967), 201–202. They actually considered *directed* graphs; but of course their method also handled undirected graphs, because each edge u — v can be represented by a pair of arcs, $\{u \longrightarrow v,\ v \longrightarrow u\}$. Their procedure was an early instance of straightforward backtracking: At each stage of the computation, a partial path $v_1 \longrightarrow \cdots \longrightarrow v_k$ was extended by trying all possible successors to v_k that hadn't yet appeared.

Mark B. Wells, in §4.2.4 of his book *Elements of Combinatorial Computing* (1971), presented a similar method, but for the task of visiting all Hamiltonian *paths*, in *undirected* graphs. He explained how to detect certain impossible cases early in the search, by backtracking whenever a partial path v_1 — $\cdots$ — v_k wipes out all chances to reach a yet-untouched vertex v, namely when all of v's neighbors belong to $\{v_1, \ldots, v_k\}$. He also considered the untouched v that have exactly *one* neighbor in $\{v_1, \ldots, v_k\}$: There must not be three such vertices; and special restrictions apply when there are exactly two of them.

But Geoffrey R. Selby, in his Ph.D. thesis at Imperial College (University of London, 1970), 264 pages, realized that an edge-oriented approach would be much better. Instead of assigning vertices sequentially, he developed a "link algorithm" for undirected graphs that has much in common with Algorithm H above. Selby's method was not as symmetrical as it could have been — he still retained the concept of a "main" partial path v_1 — $\cdots$ — v_k; but he augmented that path with a separate set of edges that it *implies*. Such edges, which form additional partial paths, arise when an untouched vertex belongs to only two available edges.

Historical notes
grid graph
Rudyns'kyj
Vovk
Ukraine
Stone Age
Paleolithic
Mizyn (Cyrillic 'Mi1zin')
meander friezes
Greek pottery
icosahedra
knight's tours
all Hamiltonian cycles
Roberts
Flores
directed graphs
backtracking
partial path
Wells
all Hamiltonian *paths*
undirected graphs
Selby
Imperial College
University of London
edge-oriented approach
link algorithm

Christofides
multi-path algorithm
Martello
MRV heuristic
Warnsdorf's rule
Rubin
Selby
Wells
Hakimi
Kocay
articulation point
bipartite
Chalaturnyk
data structures
Knuth
HAMDANCE
data structures
dancing links

Selby's co-advisor at Imperial College, Nicos Christofides, worked out a similar "multi-path algorithm" for *directed* graphs, and presented it in §10.2.3 of his book *Graph Theory: An Algorithmic Approach* (1975). Then S. Martello improved the algorithm further by incorporating the MRV heuristic, when selecting the initial vertex v_1 and when rank-ordering the neighbors of v_k that are candidates for v_{k+1}. [*ACM Trans. on Mathematical Software* **9** (1983), 131–138.] (The MRV heuristic had also been mentioned by Wells, who pointed out that neighbor ranking makes no difference to the total running time when we are visiting *all* of the solutions, because we are going to consider all choices of v_{k+1} anyway. However, just as with Warnsdorf's rule, MRV tends to find the *first* solution much faster.) Curiously none of these authors realized that it would be much better to use MRV symmetrically on the set of *all* endpoints of the current partial paths or directed paths, as in Algorithm H or Algorithm B, instead of always extending a "main" path by choosing a neighbor for v_k at every stage. (Selby did sometimes extend his main path at the left, if vertex v_1 had a forced neighbor v_0.)

Frank Rubin [*JACM* **21** (1974), 576–580], unaware of Selby's or Wells's work, but inspired in part by S. L. Hakimi [*IEEE Region Six Conf. Record* (1966), 635–643], proposed a similar algorithm that was in some ways weaker and in other ways stronger. His method repeatedly extended a single path at the right, while marking certain off-path edges as "required" and other edges as "deleted." (For example, edges that touch a vertex of degree 2 were "required.") But again, there was only a single main path. He also tested reachability between the path vertices and the remaining vertices; and he discussed preprocessing, whereby a graph could sometimes be partitioned into subgraphs whose Hamiltonian cycles could be pieced together to obtain the overall cycles.

William Kocay [*Discrete Math.* **101** (1992), 171–188] realized that Selby and Christofides's multi-path algorithm could be made symmetrical, by giving equal status to every subpath. (Curiously, however, he still distinguished the left and right endpoints of subpaths. Instead of using MRV, the vertex that he chose for branching was a right endpoint of *maximum* degree(!)—see exercise 129.)

Kocay also went further, by backtracking when the current subproblem could not be completed to a Hamiltonian cycle because the current subgraph either had an articulation point or was bipartite in an impossible way. (See exercise 130.) This test was costly, but it could save considerable time in many cases.

Andrew Chalaturnyk (Master's thesis, University of Manitoba, 2008, vi + 123 pages) made Kocay's algorithm significantly faster by designing data structures that allow it to backtrack efficiently. He improved the method also by invoking tests for articulation points or bipartiteness only at judicious intervals, when chances for effective pruning of the search tree seemed most likely. Without such pruning (which was optional), his program was rather similar to Algorithm H, although considerably more complicated.

In unpublished experiments during 2001, Donald E. Knuth had developed a symmetrical edge-based algorithm for Hamiltonian cycles in undirected graphs that was comparatively simple. He called it HAMDANCE, because it used data structures analogous to the dancing links of Algorithm 7.2.2.1X. Algorithm H,

which is called SSHAM because it uses sparse-set structures instead, was devised in 2024, and followed by Algorithm B (SSBIDIHAM) in 2025.

Counting of knight's tours on rectangular boards began with a 4-page note by J. J. Duby, *Etude #8* (Paris: IBM France, 22 October 1964), stating that the 6×6 board has 9862 cycles. Subsequent early work is summarized in *The Games and Puzzles Journal* **2**,15 (December 1997), 265.

David Singmaster [*International Series of Numerical Mathematics* **29** (Basel: Birkhäuser, 1975), 117–130] discussed Hamiltonian enumeration and gave a heuristic ballpark estimate for the total number of 8×8 knight's cycles: $10^{23\pm3}$.

Martin Löbbing and Ingo Wegener [*Electronic J. Combinatorics* **3**,1 (1996), #R5, 1–4 and comment] tried to count the 8×8 cycles by applying extended BDD methods to each of roughly 380 million subproblems. Unfortunately something went wrong, because the answer they got — more than 33 trillion — was not a multiple of 4. This error stimulated Brendan D. McKay to compute the correct value, as mentioned above, but without actually visiting the solutions.

The idea of a census is due to Günter C. Stertenbrink, who formulated the "corner wedge" approach of exercise 152 in 2003, thereby gaining a factor of 8 because of symmetry. Andrew Chalaturnyk, as part of his thesis work in 2008, generated the cycles for each of Stertenbrink's 41790 canonical corner-bunches. Working off and on, at times together with Yann Denef, Stertenbrink was able in 2023 to compile a compressed database that contains representatives of all 13 trillion tours, occupying fewer than three terabytes of SSD storage. (See `http://magictour.free.fr`.) A census based on *central* wedges was independently devised by the author in 2010; see *FGbook*, pages 494 and 495.

Algorithm E is based on an approach that is often called the "transfer-matrix method" by mathematicians and physicists, or "dynamic programming" by computer scientists. Those ideas were first applied to Hamiltonian cycles only in very special cases, such as the grid graphs $P_m \mathbin{\square} P_n$; see, for example, Robert Stoyan and Volker Strehl, *J. Combin. Math. and Combin. Computing* **21** (1996), 109–127. But Ville H. Pettersson [*Electronic J. Combinatorics* **21** (2014), #P4.7, 1–15] explained how to adapt the same methods to an arbitrary graph.

Indeed, the dissertation of André Pönitz (Dr. rer. nat., Tech. Univ. Freiberg, 2004) developed a highly general approach to graph computations that included not only Hamiltonian cycles but also colorings, independent sets, acyclic orientations, and other problems galore. His "composition" framework solves such problems dynamically by building up a graph one vertex and one edge at a time. [See *Operations Research Proceedings* (2002), 383–388.]

Algorithm E was also inspired by work on knight's tours. Early in 1994, Noam Elkies and Donald E. Knuth independently obtained generating functions for the number of closed $3 \times n$ knight's tours. They learned of each other's work during a chance encounter in Berkeley, but didn't publish the results at that time. Knuth [*FGbook*, Chapter 42] eventually extended this analysis to open $3 \times n$ tours, and to tours with various kinds of symmetry.

Euler had proved in 1759 that there are no closed tours on a $4 \times n$ board. Johan de Ruiter realized that $m \times n$ boards for fixed $m > 4$ were computationally

SSHAM
sparse-set
SSBIDIHAM
Duby
Singmaster
Löbbing
Wegener
BDD methods
McKay
Stertenbrink
corner wedge
Chalaturnyk
Denef
author
transfer-matrix method
dynamic programming
grid graphs
Stoyan
Strehl
Pettersson
border structure, see marked involution
Pönitz
colorings
independent sets
acyclic orientations
knight's tours
Elkies
Knuth
generating functions
symmetry
Euler
de Ruiter

feasible too; he obtained the results for $m = 5$ and $m = 6$ in 2010 (see OEIS A175855, A175881). In the following year Yi Yang and Zhao Hui Du extended the calculations to $m = 7$ and $m = 8$ (see OEIS A193054, A193055). They made the key observation that it's far better to use prior results by increasing the size of the board by one cell at a time, not by one column at a time; so their method was quite close to that of Algorithm E, in the special case of knight graphs.

Pettersson's algorithm was sort of a dual to Algorithm E: Instead of working with the frontiers $F_m = \{v \mid v > m \text{ and } v \text{ — } w \text{ for some } w \le m\}$, he worked with the "anti-frontiers" $B_m = \{v \mid v \le m \text{ and } v \text{ — } w \text{ for some } w > m\}$. As in Algorithm E, he passed from $m-1$ to m by appropriately combining the sizes of equivalence classes, characterized by mate tables. But instead of using tries, he devised methods to compute the index of a weight directly from the mate table of its class, in several families of highly structured graphs. For example, he successfully enumerated not only the 26×26 rook's tours (Hamiltonian cycles of $P_{26} \mathbin{\square} P_{26}$), but also the 16×16 king's tours (Hamiltonian cycles of $P_{16} \boxtimes P_{16}$), and the tours on a triangular grid with 20 vertices on each side (Hamiltonian cycles of $\mathit{simplex}(19, 19, 19, 19, 0, 0, 0)$).

An interesting precursor to Algorithm E$^+$ appears implicitly in OEIS sequence A083386, which enumerates the open knight's tours on a $5 \times n$ board for $n \le 50$. It was contributed by A. Pönitz in 2003, as part of his thesis research, curiously many years before the *closed* $5 \times n$ tours had been counted.

Whirling knight's tours were first considered by E. W. Bennett [*Fairy Chess Review* **6**, 105 (February 1947), 72, problem 7159; 106 (April 1947), 82]. He found a Hamiltonian path in the digraph (50), but was unable to construct a Hamiltonian cycle. Almost 50 years went by before G. P. Jelliss discovered the symmetrical cycle at the left of (51), as well as the three other classes of whirling cycles that have circular symmetry [*J. Recreational Mathematics* **28** (1997), 234].

Bidirected graphs were named by Jack Edmonds, in the notes of some influential lectures that he presented at the University of Michigan ["An introduction to matching" (Summer 1967), 41–42; `https://web.eecs.umich.edu/~pettie/matching/Edmonds-notes.pdf`]. The concept was actually implicit in a series of papers by Anton Kotzig [*Matematicko-Fyzikálny Časopis Slovenskeij Akadémie Vied* **9** (1959), 73–91, 136–159; **10** (1960), 205–215], who considered the structure of graphs that contain a perfect matching: The vertices of such graphs can be named $\{v_1, v_1', \ldots, v_{n/2}, v_{n/2}'\}$, where n is even and v_j — v_j' for $1 \le j \le n/2$; and the nonmatched edges essentially define a bidirected graph on $n/2$ vertices.

Computational progress was significantly stimulated by the "FHCP Challenge Set," a collection of 1001 Hamiltonian graphs curated from a wide variety of sources by Michael Haythorpe as part of the Flinders Hamiltonian Cycle Project [*Bulletin of the Institute of Combinatorics and its Applications* **83** (2018), 98–107]. These graphs, with up to 9528 vertices, were intentionally difficult. In fact, the challenge began as a year-long competition, in which \$1001 was offered for the first team of researchers that could solve them all; and nobody did! But David Coudert and Nathann Cohen were able to find Hamiltonian cycles in all but 16. (See `https://interstices.info/le-defi-des-1001-graphes/`.)

OEIS
Yang
Du
frontiers
anti-frontiers
equivalence classes
mate tables
tries
rook's tours
grid graphs
king's tours
triangular grid
simplex
OEIS
Pönitz
Whirling knight's tours
Bennett
Jelliss
Bidirected graphs
Edmonds
University of Michigan
Kotzig
perfect matching
FHCP Challenge Set
Haythorpe
Flinders Hamiltonian Cycle Project
Coudert
Cohen

Several attempts have been made to encode the Hamiltonian cycle problem into SAT language, and the method of cut clauses (Algorithm C) has turned out to be much better than any of the others with respect to the 1001 FHCP benchmarks. [See R. Ohashi, T. Soh, D. Le Berre, H. Nabeshima, M. Banbara, K. Inoue, and N. Tamura, *LIPIcs* **341** (2025), 24:1–24:10.] For example, when Algorithm C is implemented as an extension of Algorithm 7.2.2.2C, it solves 16 of them in less than 1 megamem, 220 of them in less than 1 gigamem, 844 of them in less than 1 teramem, and ??? (computations are in progress!).*

Ohashi
Soh
Le Berre
Nabeshima
Banbara
Inoue
Tamura

* I should be able to say "985 of them in less than 1 petamem," because the system of Ohashi et al. is able to resolve all but the hardest 16 in less than two days. I plan to have an exercise that discusses these empirical results, when the computations reach a stable state.

[This is a page-filler so that the exercises will begin on a right-hand page.]

EXERCISES

spanning
Hamiltonian
fourfold symmetry
symmetry
planar
icosahedron
concentric rings
4-cube
Gray binary code
draw the graph
unit-distance graph
graph drawing
dodecahedron
Hamiltonian path
generalized Petersen graph
Petersen graph
$\mathrm{GP}(n,k)$
concentric-ring graph
enumeration of Ham cycles
one-in-three satisfiability problem
NP-complete
cubic graph
satisfiable
win
$K_{2,1,1}$
induced subgraph
Wheatstone bridge
literals
clause gadget

1. [*15*] We could save ourselves three syllables and three letters by saying "spanning cycle" and "spanning path" instead of "Hamiltonian cycle" and "Hamiltonian path." Textbooks on graph theory could save lots of paper. Why doesn't everybody do that?

▸ **2.** [*17*] Join every vertex of graph G to a new vertex, obtaining $G' = G \mathbin{—} K_1$. True or false: G has a Hamiltonian path if and only if G' is Hamiltonian.

3. [*M22*] Reverse-engineer the rules by which Fig. 121's vertices have been named.

4. [*M30*] The Hamiltonian cycle in Fig. 122(b) doesn't look symmetrical. Show, however, that it has fourfold symmetry when drawn on an undistorted dodecahedron.

5. [*M20*] A *second* glance at the graph depicted in Fig. 122(c) reveals that it actually *is* obviously planar. Why?

6. [*22*] Draw the graph of the icosahedron in the style of Fig. 122(a), arranging the vertices in three concentric rings.

7. [*20*] Draw the graph of the 4-cube in the style of Fig. 122(c), using Gray binary code as the Hamiltonian cycle.

8. [*HM25*] Show that it's possible to redraw the graph of the dodecahedron, Fig. 122, in such a way that all lines between adjacent vertices have the same length.

▸ **9.** [*M21*] A Hamiltonian cycle on a planar cubic graph, such as the dodecahedron in Fig. 121(b), can be described as a sequence of Ls and Rs denoting "left turn" and "right turn" at each vertex encountered during the cyclic journey.

a) Prove that no Hamiltonian cycle on the dodecahedron can contain any of the following subsequences: (i) LLLL; (ii) LRRL; (iii) LRLRLRL; (iv) LLRLRLL; (v) LLRLRR; (vi) LLRLL; (vii)–(xii), subsequences (i)–(vi) with L ↔ R swapped.

b) Therefore there is essentially only one cycle (and its dual obtained by L ↔ R).

10. [*24*] For which vertices v of Fig. 122(a) is there a Hamiltonian path from 12 to v?

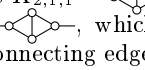
▸ **11.** [*M32*] The *generalized Petersen graph* $\mathrm{GP}(n,k)$ is an interesting cubic graph with $2n$ vertices $\{0, 1, \ldots, n-1, 0', 1', \ldots, (n-1)'\}$ and $3n$ edges

$$\{i \mathbin{—} (i+1) \bmod n,\ i \mathbin{—} i',\ i' \mathbin{—} (i+k)' \bmod n \mid 0 \le i < n\}.$$

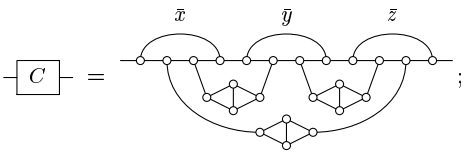
Figure 122(a) is the special case $q = 5$ of a general *concentric-ring graph* $\mathrm{GP}(2q, 2)$.

Exactly how many Hamiltonian cycles exist in the graph $\mathrm{GP}(n, 2)$, when $n \ge 1$?

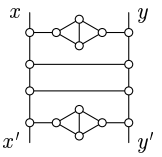
14. [*22*] The one-in-three satisfiability problem of exercise 7.2.2.2–517 is NP-complete. For every such problem F, we shall construct a cubic graph G that is Hamiltonian if and only if F is satisfiable. Every edge of G corresponds to a Boolean variable; values of the variables for which the true edges form a Hamiltonian cycle will be called a *win*.

a) A cubic graph that contains $K_{2,1,1} =$ as an induced subgraph also contains the "Wheatstone bridge"  , which has two edges that connect to other vertices. Show that those connecting edges must be true in every win.

b) For every clause $C = (x \vee y \vee z)$ of F, where x, y, and z are literals, include the "clause gadget" below as part of G. Show that $x + y + z = 1$ in every win.



;



.

c) If two edges x and y of G are replaced by the "XOR gadget" $x \oplus y$ above, show that $x = x'$, $y = y'$, and $x = \bar{y}$ in every win.

d) Suppose the clauses of F are $\{C_1, \ldots, C_m\}$. Use the gadgets above to construct the desired graph G, starting with —[C_1]—$\cdots$—[C_m]—.

16. [*29*] What's the smallest connected cubic graph that is *not* Hamiltonian?

18. [*M20*] True or false: If a planar graph has a Hamiltonian cycle, so does its dual.

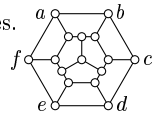
▸ **20.** [*M30*] (T. P. Kirkman, 1856.) Let G be a planar graph with n vertices and with exactly α_k k-sided faces for $k \geq 3$ (including the unbounded exterior face). For example, the graph of the dodecahedron, Fig. 122, has $n = 20$ and $\alpha_k = 12[k = 5]$.

a) If G is Hamiltonian, prove that integers a_k exist such that $0 \leq a_k \leq \alpha_k$ and $\sum_{k=3}^{n}(k-2)a_k = n - 2$. (For example, the dodecahedron has $a_k = 6[k = 5]$.)

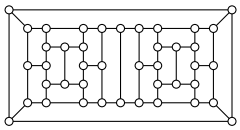
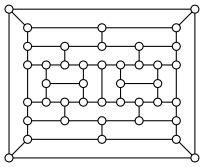
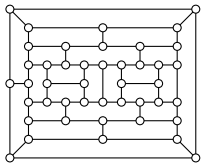
b) In a similar way, prove that the dodecahedron has no cycle of length 19.

c) Furthermore its vertices can't be completely covered by two *disjoint* cycles.

d) Use (a) to prove that every Hamiltonian cycle in the planar 16-vertex cubic graph G shown here must include the edges a—b, c—d, e—f.



e) Use (a) to decide whether any of the following graphs are Hamiltonian:

(i)  ; (ii)  ; (iii)  .

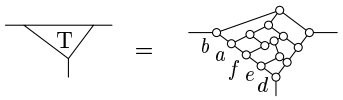
21. [*M25*] Large graphs that contain no Hamiltonian cycles can often be useful benchmarks. Construct infinitely many cubic planar graphs that fail to satisfy exercise 20(a).

24. [*M28*] A cubic graph is called *perfectly Hamiltonian* if its edges can be 3-colored in such a way that the edges of any two colors form a Hamiltonian cycle.

a) Which of the cubic graphs on 8 vertices are Hamiltonian? Perfectly Hamiltonian?

b) Prove that a *planar* cubic graph can be perfectly Hamiltonian only if there are nonnegative integers (a_k, b_k, c_k, d_k) for all $k \geq 3$ such that $a_k + b_k + c_k + d_k = \alpha_k$ is the number of k-faces as in exercise 20(a), and $\sum_k (k-2)a_k = \sum_k (k-2)b_k = \sum_k (k-2)c_k = \sum_k (k-2)d_k = (n-2)/2$, where n is the number of vertices.

27. [*M23*] The *Tutte gadget* is a useful 15-vertex graph fragment

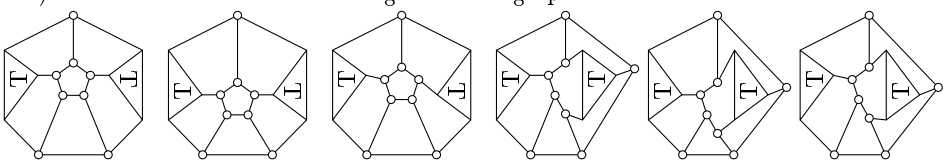


that can be obtained by removing vertex c from the 16-vertex graph in exercise 20(d).

a) Prove that every Hamiltonian cycle in a graph that contains the gadget must use the edge at the bottom of the T.

b) Prove that no Hamiltonian cycle in the pentagonal prism GP(5, 1), , includes the edges of two nonconsecutive "spokes."

c) Therefore none of the following 38-vertex graphs are Hamiltonian:



XOR gadget
gadgets
planar graph
dual
Kirkman
planar graph
faces
dodecahedron
cycle cover
benchmarks
perfectly Hamiltonian
edge coloring
decomposition, Hamiltonian
cubic graphs
Tutte gadget
pentagonal prism
generalized Petersen graphs

d) Are any two of those six graphs isomorphic to each other?

isomorphic

30. [*20*] Each letter in a Græco-Roman icosahedron can be placed three ways within its triangular face, depending on the choice of "bottom edge" (except that Δ and O are symmetric). From this standpoint, the fact that Π and Y share the *same* bottom edge, in the text's example from the British Museum, is a bit disconcerting.

Redesign that layout for the 21st century, so that (i) Roman letters `A`, `B`, ..., `T` replace the Greek ones; (ii) the bottom edge of a letter's successor is always the upper left or upper right edge of the current letter; and (iii) `T` is adjacent to `A`, completing a cycle.

Græco-Roman icosahedron
British Museum

▸ **33.** [*M20*] Suppose G is an n-vertex graph that has H Hamiltonian cycles and h Hamiltonian paths that aren't cycles. (Thus, there are H sets of n edges whose union is a cycle, and h sets of $n-1$ edges whose union is a path but not a cycle.) Let $G' = \{*\} \mathbin{—\!—} G$ be the $(n+1)$-vertex graph obtained from G by adjoining a new vertex that's adjacent to all the others. How many Hamiltonian cycles does G' have?

Hamiltonian paths that aren't cycles

35. [*M25*] A close look at (1) shows that al-'Adlī's closed tour is traced by a "thread" that weaves alternately over and under itself at each crossing, forming a "knot."

a) Prove that *every* closed knight's tour can be drawn as such a knot.

b) On the other hand, the over-under rule is violated four times in Ibn Manī''s open tour. Prove that every drawing of his tour must necessarily have at least four such exceptions to the rule.

al-'Adlī
path diagrams
weaves
knot
alternating knot diagrams
Ibn Manī'

36. [*22*] Find 4×8 knight's tours that (i) preserve the syllables of Rudraṭa's sloka, but differ from (2); (ii) preserve the fractured English syllables of (4).

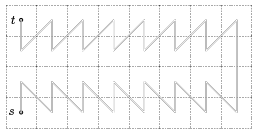
Rudraṭa
sloka
fractured English

37. [*22*] How many 4×8 knight's tours are possible?

38. [*25*] Write a two-verse English poem for Rudraṭa's 4×8 tour, analogous to (6).

poem

40. [*25*] The variant of Chaturanga played in Rudraṭa's day used a curious piece called an *elephant* (gaja) instead of a chess bishop. This piece had only five moves: one step forward or one step diagonally, representing the elephant's trunk and its four legs. For example, an elephant can tour a 4×8 board by following the (s,t)-path illustrated here.



Represent this half-tour with a two-verse poem in English.

Chaturanga
elephant
(s,t)-path: A path from vertex s to vertex t.

▸ **41.** [*M32*] This exercise classifies all elephant's tours on an $m \times n$ board, for $m, n \geq 2$.

a) Let E_{mn} be the $m \times n$ elephant digraph. How many arcs does it have?

b) Does E_{mn} have a *closed* tour (a Hamiltonian *cycle*), for some values of m and n?

c) The open elephant's tour in exercise 40 begins at the bottom left corner of E_{48}. Show that there's also an open tour that begins at the *top* left corner of E_{48}.

d) Prove that every elephant's tour must begin or end in the top row, when $m > 2$.

e) Similarly, prove that every such tour must begin or end in the bottom row.

f) Characterize all $m \times n$ elephant's tours that begin in the top row.

g) Characterize all $m \times n$ elephant's tours that begin in the bottom row.

h) Explain how to compute the number of Hamiltonian paths of E_{mn} that begin at a given vertex s and end at a given vertex t.

elephant's tours
elephant digraph
trunk moves

42. [*30*] Find a cycle of elephant moves on the 8×8 chessboard that (i) visits all but two of the cells, and (ii) has the fewest "trunk moves" among all such cycles.

44. [*18*] Using the syllables (7), construct a knight's tour quatrain that *rhymes*.

quatrain
nonsense verse

46. [*19*] Draw Someshvara's tour (8) in the style of (1).

Someshvara
path diagrams

50. [*19*] The text describes only one scenario for moves **9**, **10**, ..., that might extend the partial tour (10). What other paths are consistent with Warnsdorf's rule?

Warnsdorf's rule

51. [*21*] What paths does Algorithm W construct when G is the graph of knight moves on a 5×5 board, s is cell 00, $r = 1$, and t_1 is cell 44 (the corner opposite 00)?

52. [*20*] What is the behavior of Algorithm W if $t_i = t_j$ for some $i \neq j$?

▸ **53.** [*M21*] *Randomize* Algorithm W, by changing step W5 so that each candidate u is chosen with probability $1/q$ when there's a q-way tie for the minimum number of exits. *Hint:* There's a nice way to do this "on the fly" without building a table of candidates.

55. [*20*] How many of the 63 moves in the historic knight's tour (1) by Ibn Manī' agree with Warnsdorf's rule? Consider also the closed tour of al-'Adlī, with the same opening moves v_1 and v_2, as well as the open tour of Someshvara in exercise 46.

56. [*20*] Algorithm W sometimes moves to a "dead end" vertex (from which there's no exit), even though it could prolong the path by making a different choice. Discuss.

▸ **57.** [*21*] Design an algorithm to compute the tree of all possible paths that might be computed by Algorithm W, given G, s, and $\{t_1, \ldots, t_r\}$. Also compute, for each path, the probability that it would be obtained by the algorithm of exercise 53.

59. [*21*] What are the longest and shortest paths obtainable in the 8×8 knight graph when the *anti-Warnsdorf* rule is used? (Move always to a cell with the *most* exits.) Compare those results to the behavior of Algorithm W.

60. [*22*] Study empirically the behavior of Algorithm W on the concentric-ring graphs $R_q = \mathrm{GP}(2q, 2)$ of exercise 11, for $6 \le q \le 10$. What is the probability of obtaining (a) a Hamiltonian path? (b) a Hamiltonian cycle, when no target vertex is specified? (c) a Hamiltonian cycle, when there's a single target vertex with s —— t_1?

62. [*16*] Prove that Algorithm W always finds a Hamiltonian path when $G = P_m \mathbin{\square} P_n$ is the $m \times n$ grid graph, $s = (0, 0)$ is a corner vertex, and $r = 0$.

▸ **63.** [*M30*] Prove that Algorithm W always finds a Hamiltonian path in the special case when $G = P_2 \mathbin{\square} P_2 \mathbin{\square} \cdots \mathbin{\square} P_2$ is an n-cube and $r = 0$.

▸ **65.** [*25*] Is there a Hamiltonian graph for which Algorithm W always *fails* to find a Hamiltonian path, regardless of the starting point and the ordering of arcs?

70. [*11*] Show that step F4 sometimes calls '$update(u_1, \ldots, u_t)$', which does nothing.

71. [*M20*] Euler believed that his method for discovering tours was "safe" and "infallible"; but (16) is a case where it fails to find a cycle. Construct arbitrarily large Hamiltonian graphs for which Algorithm F can in fact get stuck with paths of length 10.

73. [*21*] Discuss implementing the dictionary of Algorithm F with a hash table based on linked lists. If the entry for each path links to the number of the previous path that belongs to the same list, step F6 can regard all links $\le p_2$ as null.

75. [*M23*] (*One-sided flips.*) Simplify Algorithm F so that it flips subpaths only at the right, and doesn't distinguish between paths and cycles; call the resulting procedure "Algorithm F$^-$." (More precisely, Algorithm F$^-$ never goes to step F5; it omits the second *update* in step F4; and it doesn't put paths into canonical form.)

a) Suppose Algorithm F$^-$ is applied to a Hamiltonian path v_1 —— $\cdots$ —— v_n in a *cubic* graph G. Show that it constructs a *cycle* of Hamiltonian paths, each beginning with v_1, and illustrate this cycle when G is the 3-cube.

b) Furthermore the number of *cyclic* Hamiltonian paths in that cycle is even.

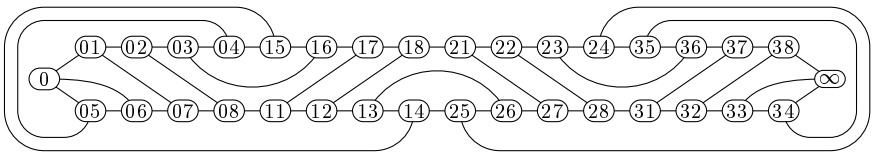
c) Moreover, the number of Hamiltonian cycles containing any given edge is even.

d) Every cubic Hamiltonian graph therefore has at least three Hamiltonian cycles.

e) Every cubic graph with exactly three Hamiltonian cycles is *perfectly* Hamiltonian.

Randomize
Ibn Manī'
al-'Adlī
Someshvara
anti-Warnsdorf
$m \times n$ knight graph: The SGB graph $board(m$
concentric-ring graphs
generalized Petersen graphs
Warnsdorf's rule
grid graph
n-cube
$update(u_1, \ldots, u_t)$
Euler
hash table
linked lists
null
One-sided flips
canonical form
lollipop method, see one-sided flips
cubic
3-cube
perfectly

77. [*M26*] The *Cameron graph* C_n *of order* n is a planar cubic graph on the $8n+2$ vertices $\{ij \mid 0 \le i < n, 1 \le j \le 8\} \cup \{0, \infty\}$ defined by the relations $i7 - i1 - i2 - i3 - i4 - i5 - i6 - i7 - i8 - i2$, $i3 - (i{+}1)6$, $i4 - (i{+}1)5$, and $i8 - (i{+}1)1$ for all integers i; replace all vertices ij for $i < 0$ by 0, and all ij for $i \ge n$ by ∞. For example,

$C_4 =$ 

Cameron graph
planar cubic graph
involution: perm of order 2
flips
mate
concentric-ring graphs
generalized Petersen graphs
Beluhov
equivalent
Forcibly Hamiltonian degrees
degree seq of graph
complete graph
flips
degree sequence
Nash-Williams
r-regular

a) Prove that the involution $ij \leftrightarrow (n{-}1{-}i)(9{-}j)$ is an automorphism of C_n.
b) Prove that C_n has exactly three Hamiltonian cycles (one of which is the "obvious" cycle $\alpha_n = 0 - 01 - 02 - 03 - \cdots - \infty - \cdots - 07 - 06 - 05 - 0$).
c) Compute the number c_n of one-sided flips needed to go from α_n to its mate β_n with respect to $0 - 01$, in the sense of answer 75(c), for $1 \le n \le 9$.
d) Surprise! Exactly $c_{n-2} + 10$ flips go from α_n to β_n with respect to $01 - 0$.
e) How many flips go from α_n to its mate γ_n with respect to (i) $0 - 05$? (ii) $05 - 0$?

78. [*22*] Study empirically the behavior of Algorithm F on the concentric-ring graphs $R_q = \mathrm{GP}(2q, 2)$ of exercise 11, for $6 \le q \le 10$ and $q = 100$. Let $t = 1$, and choose v_1 at random; also randomize the order in which a vertex's neighbors are examined. Estimate the probability of obtaining (a) a Hamiltonian path; (b) a Hamiltonian cycle. How many nontrivial calls of *update* are typically needed, before succeeding?

79. [*M32*] (N. Beluhov, 2019.) Say that two Hamiltonian paths or cycles are *equivalent* if they can be transformed into each other by Algorithm F.
a) Find a graph with two inequivalent cycles.
b) Can a graph have arbitrarily many pairwise inequivalent cycles?

80. [*M20*] For which $q_1, \ldots, q_s, t$ is the graph $(K_{q_1} \oplus \cdots \oplus K_{q_s}) - K_t$ Hamiltonian?

81. [*M27*] (*Forcibly Hamiltonian degrees.*) Sometimes we can conclude that a graph is Hamiltonian just by knowing that it has lots of edges. If $n > 2$ and the vertices of G have respective degrees $d_1 \le d_2 \le \cdots \le d_n$, we shall prove that G is Hamiltonian whenever

$$1 \le k < n/2 \quad \text{and} \quad d_k \le k \quad \text{implies} \quad d_{n-k} \ge n - k. \tag{$*$}$$

a) If G satisfies $(*)$ and has $m < \binom{n}{2}$ edges, so that G is not the complete graph K_n, prove that G contains two nonadjacent vertices $\{u, v\}$ with $\deg(u) + \deg(v) \ge n$.
b) Continuing (a), let $G_0 = G$; and let $G_{k+1} = G_k \cup \{u_k - v_k\}$, where $u_k \not- v_k$ and $\deg(u_k) + \deg(v_k) \ge n$ in G_k, for $0 \le k < \binom{n}{2} - m$. Explain how to construct a Hamiltonian cycle in G_k, given a Hamiltonian cycle in G_{k+1}. (Since $G_{\binom{n}{2}-m} = K_n$ is Hamiltonian, so too is G_0.) *Hint:* Use flips as in Algorithm F.

82. [*M25*] If condition $(*)$ fails in G for at least one value of k, show that there's a non-Hamiltonian graph G' whose degree sequence $d'_1 \le d'_2 \le \cdots \le d'_n$ satisfies $d_1 \le d'_1$, $d_2 \le d'_2$, $\ldots$, $d_n \le d'_n$. (In this sense exercise 81 is the best possible result of its kind.)

83. [*M30*] (C. S. A. Nash-Williams.) Let G be an r-regular graph with $2r+1$ vertices.
a) Prove that r is even.
b) Prove that G has a Hamiltonian path $u_0 - u_1 - \cdots - u_{2r}$.
c) If G isn't Hamiltonian, show that $u_0 - u_j \iff u_{j-1} \not- u_{2r}$, for $1 < j < 2r$.
d) If G isn't Hamiltonian, show that it has a cycle $v_1 - v_2 - \cdots - v_{2r} - v_1$.
e) Conclude that G *is* Hamiltonian. *Hint:* Suppose $v_0 - v_j \iff j$ is odd.

84. [*M28*] What's the smallest number of edges for which condition (∗) in exercise 81 forces an n-vertex graph to be Hamiltonian?

85. [*HM21*] (P. Erdős, 1962.) Let $f(n,k)=\binom{n-k}{2}+k^2$, and $g(n,k)=\max(f(n,k), f(n,\lfloor(n-1)/2\rfloor))$. Prove that if $1\le k<n/2$, there's a non-Hamiltonian graph with $g(n,k)$ edges and n vertices, where every vertex has degree $\ge k$. But every such graph with *more* than $g(n,k)$ edges is Hamiltonian. *Hint:* When is $f(n,k)\ge f(n,k+1)$?

▸ **86.** [*HM25*] A graph is called *traceable* if it has a Hamiltonian path. Continuing exercise 85, determine the largest possible number of edges in a nontraceable n-vertex graph for which the degree of every vertex is k or more. *Hint:* Let the function $\hat f(n,k)=\binom{n-1-k}{2}+k(k+1)$ play the role of $f(n,k)$ in that exercise.

88. [*M27*] The *length* of a graph is the number of edges in its longest path. (For example, the 4×4 knight graph has length 14.)

a) Let G be a connected graph whose n vertices each have degree k or more, where $k<n/2$. Prove constructively that the length of G is at least $2k$.

b) Prove that an n-vertex graph of length l has at most $nl/2$ edges.

c) Exhibit an n-vertex graph of length l and at least $nl/2-(l+1)^2/8$ edges.

▸ **89.** [*M31*] The *circumference* of a graph is the number of edges in its longest cycle. (For example, the 4×4 knight graph has circumference 14.)

a) Let G be a biconnected graph whose n vertices each have degree k or more, where $1<k\le n/2$. Prove constructively that the circumference of G is at least $2k$.

b) Prove that an n-vertex graph of circumference c has at most $(n-1)c/2$ edges.

c) If $c>2$, exhibit an n-vertex graph of circumference c and $\ge nc/2-(c+1)^2/8$ edges.

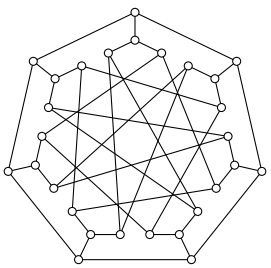
90. [*16*] True or false: The length of G is two less than the circumference of $K_1\mathbin{—}G$.

93. [*M25*] (J. W. Moon, 1965.) A graph that has a Hamiltonian path between every pair of vertices $u\ne v$ is called *Hamiltonian-connected*.

a) True or false: Every vertex of a Hamiltonian-connected graph has degree ≥3.

b) Construct a Hamiltonian-connected graph on $n\ge4$ vertices that has the smallest possible number of edges (for example, 8 edges when $n=5$; 9 edges when $n=6$).

▸ **95.** [*M28*] The *Coxeter graph* is a remarkable cubic graph whose 28 vertices $\{a_j,b_j,c_j,d_j\mid0\le j<7\}$ are connected by the edges $a_j\mathbin{—}d_j$, $b_j\mathbin{—}d_j$, $c_j\mathbin{—}d_j$, $a_j\mathbin{—}a_{j+1}$, $b_j\mathbin{—}b_{j+2}$, $c_j\mathbin{—}c_{j+3}$, for $0\le j<7$. (All subscripts are treated modulo 7. Vertices $a_0,\ldots,a_6$ form the "outer ring" of the illustration.)



a) Determine its automorphisms, by finding a Sims table as in Section 7.2.1.2. (Use the ordering $(a_6,b_6,c_6,d_6,\ldots,a_0,b_0,c_0,d_0)$; thus, for example, the permutations of $S_{n-2}=S_{26}$ will fix the final vertex d_0.) *Hint:* There will be a surprise!

b) Show that it is a *vertex-transitive graph*: Given any vertices v and v', there's an automorphism that takes $v\mapsto v'$. ("All vertices are alike.")

c) Show that it's also an *edge-transitive graph*: Given any edges $u\mathbin{—}v$ and $u'\mathbin{—}v'$, there's an automorphism that takes $\{u,v\}$ into $\{u',v'\}$. ("All edges are alike.")

d) Furthermore, it's a *hypohamiltonian graph*: It has no Hamiltonian cycle, yet it does become Hamiltonian when any vertex is removed.

▸ **100.** [*HM30*] Analyze the cycle covers of the flower snark graph J_q, for $q>2$ (see exercise 7.2.2.2–176). How many of them have exactly k cycles?

▸ **103.** [*M25*] If a graph G has a Hamiltonian cycle H, show that there's a very easy way to test whether or not G is planar. *Hint:* See Algorithm 7B.

Erdős
traceable
nontraceable
length
knight graph
circumference
biconnected graph
Moon
Hamiltonian-connected
Coxeter graph
cubic graph
automorphisms
Sims table
vertex-transitive graph
edge-transitive graph
hypohamiltonian graph
cycle covers
flower snark graph
planar

105. [*M18*] Exactly how many Hamiltonian cycles are present in (a) the complete graph K_n? (b) the complete bipartite graph $K_{m,n}$?

106. [*M26*] Continuing exercise 105, enumerate the Hamiltonian cycles of $K_{l,m,n}$.

▸ **108.** [*24*] According to the discussion in the text, every Hamiltonian cycle that contains edge $\{{}^{02}_{21}\}$ of the 3×10 knight graph also contains the edge $\{{}^{01}_{13}\}$.

a) Given those edges, show that either $\{{}^{07}_{28}\}$ or $\{{}^{16}_{28}\}$ must be chosen.
b) And if that choice is $\{{}^{16}_{28}\}$, another edge is forced.
c) Continuing (b), show that edge $\{{}^{03}_{22}\}$ leads to a contradiction.
d) Continuing (c), consider the consequences of now choosing $\{{}^{03}_{15}\}$.

109. [*15*] After Algorithm H deduces the starting pattern for the 3×10 knight graph, what are the values of `MATE(0)`, `MATE(1)`, `MATE(2)`, `MATE(3)`, `MATE(4)`?

111. [*23*] How much space should be allocated for the arrays `TRIG`, `ACTIVE`, and `SAVE` in Algorithm H so that no memory bounds will be exceeded?

▸ **112.** [*26*] Exactly what changes to the data structures should be made in step H8 of Algorithm H when we have (a) `MATE(`u`)` < 0 and `MATE(`w`)` < 0? (b) `MATE(`u`)` < 0 and `MATE(`w`)` ≥ 0? (c) `MATE(`u`)` ≥ 0 and `MATE(`w`)` < 0? (d) `MATE(`u`)` ≥ 0 and `MATE(`w`)` ≥ 0?

113. [*20*] Design an algorithm to "unscramble" the cycle defined by arrays `EU` and `EV` in step H13: It should find a permutation such that $v_1 — v_2 — \cdots — v_n — v_1$.

115. [*M23*] Describe the search tree of Algorithm H when G is the complete graph K_n.

116. [*20*] Find a Hamiltonian cycle in the graph *binary*$(4, 4, 0)$ (see Table 7.2.1.6–3).

117. [*M22*] (V. Chvátal, 1973.) The *toughness* $t(G)$ of graph G is $\min |U|/k(G \setminus U)$, where the minimum is taken over all sets of vertices U such that $G \setminus U$ is disconnected, and k denotes the number of components. (If G is the complete graph K_n, no set U disconnects it, and we have $t(K_n) = \infty$.) We say that G is "tough" if $t(G) \geq 1$.

a) True or false: $t(G) = 0$ if and only if G isn't connected.
b) Show that $t(G \setminus e) \leq t(G)$ for all edges e.
c) Prove that every Hamiltonian graph is tough.
d) Evaluate $t(K_{m,n})$ when $m \leq n$.
e) What's $t(G)$ when G is the Petersen graph (which isn't Hamiltonian)?

118. [*M27*] Continuing exercise 117, determine $t(K_m \mathbin{\square} K_n)$, when $m, n > 1$.

119. [*M24*] Read the SGB source code of *raman* and explain the edges of graph E.

120. [*M27*] Let G_0 be the graph with vertices $\{i, j, k, u, v, w, x, y, z, U, V, W, X, Y, Z\}$ and the following 24 edges: $i — j — k — i$; $u — i — U$, $v — j — V$, $w — k — W$; $u — U — x — X — v — V — y — Y — w — W — z — Z — u$; $x — Y$, $y — Z$, $z — X$.

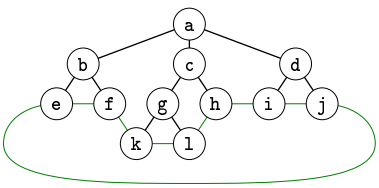
a) Prove that G_0 has twelve automorphisms and exactly two Hamiltonian cycles.
b) Let $(p_1, p_2, p_3) = (i, j, k)$; $(q_1, q_2, q_3) = (u, v, w)$; $(Q_1, Q_2, Q_3) = (U, V, W)$; and $(P_1, P_2, P_3) = (Z, X, Y)$. For $1 \leq t \leq 3$, let $G_0^{(t)}$ be a copy of G_0 with vertices $i^{(t)}, \ldots, Z^{(t)}$. Obtain graph G_t by appending $G_0^{(t)}$ to G_{t-1} and then doing this: (i) Remove vertex q_t; (ii) remove the edges $q_t — Q_t$ and $x^{(t)} — X^{(t)}$; (iii) replace the edges $p_t — q_t$ and $P_t — q_t$ by $p_t — x^{(t)}$ and $P_t — X^{(t)}$; (iv) add edges from Q_t to all the vertices of $G_0^{(t)}$ except $i^{(t)}$, $j^{(t)}$, $k^{(t)}$. (Graph G_t therefore has $15+14t$ vertices and $24 + 34t$ edges.) Prove that G_t has exactly two Hamiltonian cycles.

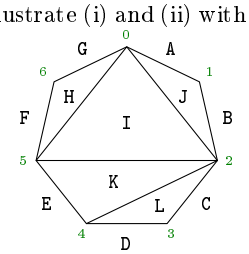
complete graph
K_n
complete bipartite graph
$K_{m,n}$
$K_{l,m,n}$
tripartite graph, complete
3×10 knight graph
`MATE`
`TRIG`
`ACTIVE`
`SAVE`
memory bounds
unscramble
complete graph K_n
binary
Chvátal
toughness
disconnected
cutsets: sets of vertices that disconnect a graph
components
complete graph
K_n
$K_{m,n}$
Petersen graph
rook graph $K_m \mathbin{\square} K_n$
SGB
raman
automorphisms

Halin graph
preorder
regular q-gon
nonintersecting chords
chords
planar
isomorphic
uniformly Hamiltonian
Hamiltonian, uniformly
nonintersecting chords
pi, as random source
regular 100-gon
Halin graphs
Hamiltonian path
articulation point
planar
MRV heuristic
branching
benchmarks
current graph
visible vertices

▸ **122.** [*M27*] A *Halin graph* can be defined in two complementary ways: (i) Let T be a tree in which no node has degree 1, and the root has degree ≥ 3. Let the leaves of T be $x_0x_1\ldots x_{q-1}$ in preorder, and let C be the cycle x_0 — x_1 — $\cdots$ — x_{q-1} — x_0. Then $H = T \cup C$ is a Halin graph. (ii) Let C be a regular q-gon with vertices 0, 1, ..., $q-1$. Let $\{i_1$ — j_1, ..., i_t — $j_t\}$ be nonintersecting chords of C; they partition the interior of C into $t+1$ regions. Let H be the graph of order $q+t+1$ whose vertices are the sides of C, together with those regions. Two sides are adjacent in H if they are consecutive; a region is adjacent to the sides on its boundary and to the regions with which it shares a boundary. Then H is a Halin graph. Notice that, under either definition, a Halin graph must be planar, and each of its vertices must have degree 3 or more.

For example, here are structures that respectively illustrate (i) and (ii) with $q = 7$:





a) Prove that the corresponding graphs are isomorphic, by finding a correspondence between vertices $\{\texttt{a}, \texttt{b}, \ldots, \texttt{l}\}$ and vertices $\{\texttt{A}, \texttt{B}, \ldots, \texttt{L}\}$ that preserves adjacency.
b) If H satisfies definition (i), prove that it also satisfies definition (ii).
c) If H satisfies definition (ii), prove that it also satisfies definition (i).

123. [*M30*] How many nonisomorphic Halin graphs have n vertices, for $n \leq 1000$?

124. [*M25*] A graph is *uniformly Hamiltonian* if, for every edge e, it contains a Hamiltonian cycle C^+ with $e \in C^+$ as well as a Hamiltonian cycle C^- with $e \notin C^-$. Prove that every Halin graph is uniformly Hamiltonian.

125. [*20*] Use the decimal digits $(\pi_0.\pi_1\pi_2\ldots)_{10}$ of π to define t nonintersecting chords $(i_1$ — j_1, ..., i_t — $j_t)$ of a regular 100-gon for $0 \leq t < 98$, by letting $i_k = \pi_{4r}\pi_{4r+1}$ and $j_k = \pi_{4r+2}\pi_{4r+3}$, where r is as small as possible with respect to previous chords. For example, the sequence begins (31 — 41, 59 — 26, 53 — 58, 97 — 93, 23 — 84, 62 — 64, ...); the next chord cannot be 33 — 83, because that one overlaps 31 — 41. These chords define Halin graphs $H_\pi^{(t)}$ with $101 + t$ vertices, by exercise 122.

What is the final chord, i_{97} — j_{97}?

127. [*20*] There's obviously no Hamiltonian path in the graph 7.1.4–(133) from ME to any of the other states of New England (NH, VT, MA, CT, RI), because NY is an articulation point. Is there a Hamiltonian path from ME to every *other* state?

▸ **128.** [*20*] Which of the 14 benchmark graphs in Table 1 are planar?

▸ **129.** [*24*] The MRV heuristic used in step H11 to choose a vertex for branching prefers small degree d, because the search tree has a d-way branch. On the other hand, one can argue that large d is actually good, because step H12 removes d edges — and that might force a contradiction, or it might cause more vertices to become clothed.

Experiment with a modified step H11, which *maximizes* d in cases where $d < 2$ is impossible, by testing the modified algorithm on the benchmarks of Table 1.

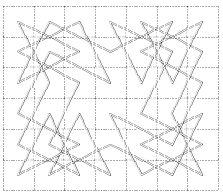
130. [*20*] At the beginning of step H11, define the "current graph" G' to be the graph whose vertices are the currently visible vertices, and whose edges are (i) the edges of G that haven't been deleted, and (ii) the edges v — MATE(v) for all outer vertices v. Prove that we could safely jump to step H14 if G' isn't Hamiltonian.

133. [*M20*] Can a knight's cycle on an $n \times n$ board have diagonal symmetry?

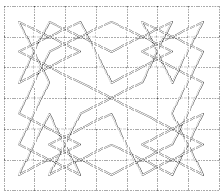
134. [*M20*] Continuing exercise 133, show that a knight's cycle on an $m \times n$ board can have vertical symmetry only if $m/2$ and n are odd. (We require (i, j) —— (i', j') to be an edge of the cycle if and only if $(m-1-i, j)$ —— $(m-1-i', j')$ is also an edge.)

135. [*22*] Given m and n, with $m/2$ and n odd, construct a graph whose Hamiltonian cycles correspond to the vertically symmetric $m \times n$ knight's cycles.

▸ **136.** [*23*] Centrally symmetric $m \times n$ knight's cycles can be surprisingly subtle:



1	40	11	36	25	38	27
12	9	42	39	28	35	24
41	2	13	10	37	26	29
8	5	16	31	34	23	20
3	14	7	18	21	30	33
6	17	4	15	32	19	22



7	4	19	16	33	2	31
20	15	6	3	30	17	34
5	8	21	18	1	32	29
14	11	42	25	22	35	38
9	26	13	40	37	28	23
12	41	10	27	24	39	36

On the left, the symmetry shows up because the step numbers of opposite cells always differ by 21: $|1-22| = |40-19| = |11-32| = \cdots = |29-8| = 21$. (This 6×7 tour begins in one corner, travels to the opposite corner in 21 steps, then repeats its motions — but rotated 180°.) The symmetry on the right, however, is quite different, although most of the edges are the same: The step numbers of opposite cells now *sum* to 43: $7+36 = 4+39 = 19+24 = \cdots = 29+14 = 43$. (It has to be seen to be believed!) After the right-hand tour has gone halfway, it moves *backwards* over the paired cells.

Given m and n, with m even and n odd, construct a graph whose Hamiltonian cycles correspond to the centrally symmetric $m \times n$ knight's cycles. *Hint:* See exercise 135.

▸ **137.** [*28*] Continuing exercise 136, show that centrally symmetric $m \times n$ knight's cycles are even *more* subtle when m and n are both even. How can all of them be found, with the help of a suitable graph G? Explore the case $m = n = 8$ in detail.

138. [*24*] Use the results of exercises 134–137 to compute the exact number of symmetrical $m \times n$ knight's cycles, when $m \bmod 4 = 2$, n is odd, and $mn < 100$.

139. [*21*] Find all the 10×10 giraffe tours that are symmetric under 90° rotation.

141. [*16*] Exactly how many 8×8 arrays like (9) define a closed knight's tour?

142. [*M22*] If C is a closed knight's tour in bunch $\alpha_1\alpha_2\alpha_3\alpha_4$, what bunch contains (a) C's top-bottom reflection? (b) C's left-right reflection? (c) C's transpose?

143. [*16*] What bunch contains the closed knight's tour formed from Ratnākara's half-tour (2)? What is its canonical bunch?

144. [*12*] True or false: `abAB` is a canonical bunch of multiplicity 4.

145. [*15*] Why can't '`a`' appear in the name of a closed knight's tour's bunch?

146. [*20*] List all canonical bunches whose multiplicity is 2.

148. [*M21*] Use "Burnside's Lemma" to determine the number of canonical bunches.

▸ **149.** [*25*] Explain how to visit every closed *giraffe's tour* on a 10×10 board.

151. [*25*] The knight's census described in the text is based on the wedges formed at cells $\{33, 34, 43, 44\}$ of an 8×8 board, where ij denotes the cell in row i and column j for $0 \le i, j < 8$. Explain how to carry out a similar census, based instead on the wedges formed at $\{03, 04, 30, 37, 40, 47, 73, 74\}$.

▸ **152.** [*25*] Do exercise 151, but with the wedges formed at $\{12, 15, 21, 26, 51, 56, 62, 65\}$.

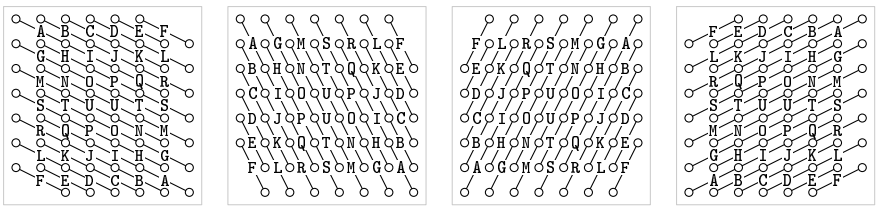
diagonal symmetry
vertical symmetry
axial symmetry
Centrally symmetric
giraffe tours
symmetric under 90° rotation
path diagrams
knight's tour
bunch
top-bottom reflection
left-right reflection
transpose
Ratnākara
canonical bunch
bunch
Burnside's Lemma
canonical bunches
giraffe's tour

154. [*19*] To which of the thirteen topological types in Fig. 124 does al-'Adlī's classic tour (1) belong? *Hint:* See (9).

155. [*25*] The classification of knight's cycles in Fig. 124 applies only to square boards. Show that additional topological types arise on $m \times n$ boards when $m < n$.

156. [*24*] Compute the exact numbers of 8×8 knight's cycles of each type in Fig. 124.

▸ **157.** [*24*] The set of all knight moves on a chessboard — that is, the set of all edges on the 8×8 knight graph — is partitioned into 21 equivalence classes of size 8, when we say that two edges are equivalent if rotation and/or reflection takes one into the other. Each move can therefore be given a label from `A` to `U`, indicating its class:



Use a census to determine how many 8×8 knight's cycles (a) have at least one move of each class; (b) have at least two moves of each class; (c) have all eight of the moves in six different classes. (Every cycle contains all eight of the class `A` moves.)

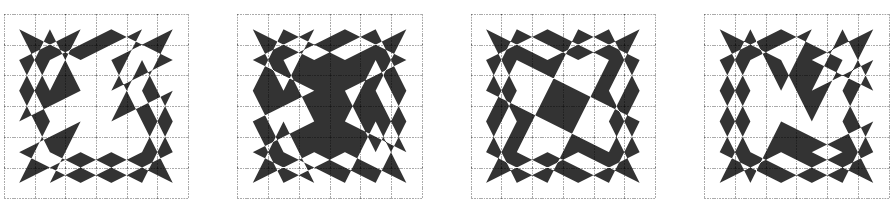
▸ **158.** [*29*] (G. P. Jelliss, 1976.) According to Fig. 123, six different angles $\{\theta, 90°{-}\theta, 90°, 90°{+}\theta, 180°{-}\theta, 180°\}$ can occur in a wedge. For every such angle α, determine the maximum and minimum number of times α can occur among the 64 moves of a closed knight's tour. Determine also the maximum and minimum *sum* of all 64 angles. Furthermore, discover exactly how many tours achieve those maxima and minima.

159. [*34*] Every knight's move "tarnishes" the two cells that it jumps over, by invading their space. A corner call cannot be tarnished; and the other 24 cells on the border of a chessboard can be tarnished at most twice. The 36 interior cells can each be tarnished at most seven times (not eight!). Use a census to discover as many interesting facts as you can about the multisets of 128 tarnishments that can arise in closed tours.

160. [*39*] Knight moves can intersect each other in essentially four ways: (i) perpendicular, '$\perp$'; (ii) 1:3 cuts, '$\asymp$'; (iii) 1:2 cuts, '$\asymp$'; and (iv) 1:1 cuts, '$\times$', also called X cuts. Perpendicular cuts are asymmetrical, with one arm having a cut ratio of 3:2 while the ratio in the other is 1:4. The other types are symmetrical, having the stated cut ratios; the angles at the crossing point are θ and $180° - \theta$ for (iii), but $90° \pm \theta$ for (ii) and (iv). Notice that every knight move has exactly one "conjugate," with which it makes an $\times$. What interesting facts about intersections can you turn up, censuswise?

161. [*46*] Can a closed $n \times n$ knight's tour have fewer than $12n + O(1)$ intersections?

▸ **163.** [*40*] Closed tours can also be depicted by *changing color* when the path is crossed:



(i) (ii) (iii) (iv)

topological types
al-'Adlī
knight graph
equivalence classes
rotation and/or reflection
census
diverse knight's tours
Jelliss
angles of a knight move
tarnished
1:3 cuts
1:2 cuts
X cuts
self-intersections
Perpendicular cuts
conjugate
crossings, see intersections

In such harlequinesque patterns, we cross the knight's path an odd number of times when we travel from a shaded region to the edge of the board. (Graphic designers know this as the "eofill" operation, short for "even-odd filling.")

Examples (i) and (ii) are the 6×6 tours with minimum and maximum shaded area, $\frac{547}{60} \approx 9.11667$ and $\frac{303}{20} = 15.15$, out of the total conceivable area of $5 \times 5 = 25$. (The extremal tours that achieve those limits are in fact unique, up to symmetry.)

Suppose C is an oriented cycle that divides the plane into regions when it crosses itself. The *winding number* of a point p with respect to C, when $p \notin C$, is the net number of times by which C encircles p in the counterclockwise direction. All points of a region have the same winding number. The shaded area of C is the sum of the areas of regions whose winding number is odd.

The *swept area* of C is the integral of the winding number over all $p \notin C$; equivalently, it's the sum, over all regions, of the area of that region times the winding number of that region. Example (iii) is a cycle whose regions have winding numbers $\{0, 1, 2, 3, 4, 5\}$ (or $\{0, -1, -2, -3, -4, -5\}$, depending on which way we traverse that cycle). It's the unique cycle whose swept area achieves the maximum value (namely 61); its shaded area is $\frac{181}{15} \approx 12.067$. On the other hand, the cycle in example (iv) has a swept area of zero. (Its regions have winding numbers $\{-2, -1, 0, 1, 2\}$.) Among all $8 \cdot 13$ such cycles, it uniquely has the smallest shaded area: $\frac{32}{3} \approx 10.667$.

Use a census to explore these aspects of 8×8 knight's cycles. How large and small can the shaded area be? What is the maximum swept area? How many of the 49 internal corner points can have winding number zero? And so on.

harlequinesque patterns
eofill
even-odd filling
winding number
swept area
swept area
Parmentier
Hamilton's graph
1-configs
m-configs
dodecahedron graph
m-classes
involution
marked involution
recurrence relation

164. [*M30*] Prove that the swept area A of an $m \times n$ knight's cycle is always an integer, and it can be computed in at least two ways:

a) A is the sum of the winding numbers at the $(m-1)(n-1)$ internal corner points.

b) $A = \frac{1}{2}(i_0 j_1 - i_1 j_0 + i_1 j_2 - i_2 j_1 + \cdots + i_{mn-1} j_{mn} - i_{mn} j_{mn-1})$, when the tour is (i_0, j_0) — (i_1, j_1) — $\cdots$ — $(i_{mn}, j_{mn}) = (i_0, j_0)$.

165. [*22*] (T. Parmentier, 1891.) Every knight move has four possible slopes, namely $-1/2$, -2, $+2$, and $+1/2$, as exhibited in exercise 157. Is there a knight's tour that has exactly 16 moves of each slope?

170. [*20*] What 14-configs of Hamilton's graph (26) belong to class $11\bar{1}00$?

171. [*10*] How many 1-configs does a graph have?

172. [*M15*] Describe the $(n-1)$-configs of an n-vertex graph.

173. [*20*] According to the text, Algorithm E discovers that the dodecahedron graph (26) has exactly six 16-classes, namely $(\bar{1}101, \bar{1}11\bar{1}, 0110, 1001, 101\bar{1}, 1212)$, of respective sizes $(4, 6, 2, 6, 4, 10)$. What then are the 17-classes, and their sizes?

174. [*15*] Explain the last step, '$11\bar{1} \mapsto_{18} C_{20}$', of (32).

▸ **176.** [*M20*] An *involution* is a permutation whose cycles all have length 1 or 2. A *marked involution* is similar, but each 1-cycle is either "marked" or "unmarked." For example, the marked involutions of order 2 are $(1)(2)$, $(1)(2)'$, $(1)'(2)$, $(1)'(2)'$, and (12).

a) Let t_n and T_n be the number of involutions and marked involutions of order n. (The first few values are $(t_0, \ldots, t_9) = (1, 1, 2, 4, 10, 26, 76, 232, 764, 2620)$ and $(T_0, \ldots, T_9) = (1, 2, 5, 14, 43, 142, 499, 1850, 7193, 29186)$.) Show that $T_n = \sum_k \binom{n}{k} t_k$.

b) Prove the recurrence relation $T_n = 2T_{n-1} + (n-1)T_{n-2}$, for $n \geq 2$.

c) Suppose $q = |\widehat{F}_m|$ is the number of elements in the extended m-frontier of a graph. Show that the total number of m-classes in that graph is at most T_q.

177. [*HM28*] Study the asymptotic behavior of T_n, by deriving a formula for marked involutions that's analogous to Eq. 5.1.4–(53) for ordinary involutions.

178. [*18*] Why is it wise to use marked involutions, encoded in the form $a_1 \dots a_q$, as class names, instead of using a `MATE` table directly?

179. [*20*] In a `MATE` table such as (35), `MATE[`j`]` $= (-1,\ 0,\ k > 0)$ means that u_j is respectively (inner, bare, mated to u_k).

a) Convert a given marked involution $a_1 \dots a_q$ to an equivalent `MATE` table.

b) Conversely, convert a given `MATE` table to an equivalent marked involution.

181. [*20*] Explain (37) by considering two cases: (i) $m{+}1 \in \widehat{F}_{m-1}$; (ii) $m{+}1 \notin \widehat{F}_{m-1}$.

183. [*20*] List (by hand) all relations $\alpha \mapsto_m \beta$ that are valid for the complete graph K_5.

184. [*M23*] Find all α such that $\alpha \mapsto_m C_p$, when G is the graph K_n and $n \ge p$.

185. [*M25*] Suppose G is the complete graph K_n. What is $F(m, r, s, t)$, the size of an m-class for which $(r, s, 2t)$ elements of the extended frontier $\widehat{F}_m = (m+1, \dots, n)$ are respectively (inner, bare, outer)? (Here $m + r + s + 2t = n$.)

187. [*24*] Give details of how Algorithm E moves from $\widehat{F}_{m-1}$ to $\widehat{F}_m$ when it updates `FR`, `IFR`, q_0, and q in step E2. Also compute σ and τ for (36)–(38); r and `NBR` for (39).

189. [*20*] Explain how Algorithm E can traverse its "old" trie in steps E3 and E8.

▸ **191.** [*21*] Design a subroutine 'contribute()' for use by Algorithm E. It should insert the m-class defined by the `MATE` table into the current trie of m-classes, if that class isn't already present in the trie; and it should add `OWT[`$p'_{q'}$`]` to that class's current size. *Note:* As stated in step E2, the trie has p nodes and w lieves.

▸ **192.** [*M20*] True or false: If `OMATE[1]` > 0 in step E4, then `BMATE[OMATE[1]`σ`]` $= 0$.

▸ **193.** [*30*] Design a subroutine 'try(i, j)' for use by Algorithm E. It should contribute() if we can legitimately connect u_i with u_j within each m-config in the class of the `BMATE` table. It should also update `CYC[`m'`]`, if that connection would close a suitable m'-cycle.

▸ **195.** [*20*] How large should Δ be, when Algorithm E works on the 8×32 knight graph?

196. [*20*] When the cells of a chessboard are ordered columnwise as in (41), the first 26 cells make a curious sub-board, which consists of two rows of length 4 above six rows of length 3. Find, by hand, a knight's cycle on that sub-board.

▸ **197.** [*20*] If you use the Stanford GraphBase to create the 8×32 knight graph for Algorithm E, should you make *board*$(8, 32, 0, 0, 5, 0, 0)$ or *board*$(32, 8, 0, 0, 5, 0, 0)$?

198. [*24*] Watching Algorithm E, answer the following about the computation of (40):

a) Every m-class enters its trie via the contribute() subroutine of exercise 191. Some classes are contributed once; others are contributed many times. How often was that subroutine called, as a function of $m \bmod 8$, assuming that $72 < m \le 240$?

b) Continuing (a), how often did the subroutine try(i, j) of exercise 193 update `CYC[`m'`]` instead of calling contribute()?

c) What is the smallest weight of an m-class when (i) $m = 32$? (ii) $m = 64$? (iii) $m = 96$? (iv) $m = 128$?

d) What is the largest weight of an m-class, for those m?

e) What's the largest m for which some m-class has weight 1?

f) What's the smallest m for which some m-class has weight $\ge 10^9$?

g) What is the lexicographically smallest $(72 + r)$-class, for $0 \le r < 8$?

▸ **199.** [*35*] Find an ordering of the vertices of the 8×32 knight graph for which the size of every extended frontier is at most 16.

asymptotic
marked involutions
involutions
`MATE` table
complete graph
complete graph K_n
frontier
`FR`
`IFR`
`NBR`
traverse
trie
contribute()
`MATE` table
try(i, j)
basic mate table
`BMATE` table
Δ
Stanford GraphBase
$m \times n$ knight graph
board graphs
contribute()
try(i, j)
lexicographically smallest
knight graph
frontier

200. [*23*] Consider the eight transitions in the cycle (42). What will α_1, α_2, ..., α_7 be, when (a) $\alpha_0 = 1234214300000000$? (b) $\alpha_0 = 01\bar{1}2314505004023$?

201. [*21*] What 8-classes $a_1 \ldots a_{16}$ of the 8×32 knight graph have $a_1, \ldots, a_{16} > 0$?

▸ **202.** [*25*] Construct a periodic knight's tour, analogous to those of (43), in which the knight changes direction *sixteen times* as it traverses the cycle.

▸ **203.** [*25*] Notice that the trie in Fig. 125 can be reconstructed by just knowing the null-versus-nonnull patterns of the node fields: Writing 0 for a null link and 1 otherwise, the root's pattern is 1110, and its leftmost child's pattern is 0010, and so on; the patterns in preorder are 1110, 0010, 0110, 0010, 1000, 0010, 0010, 0100, 0101, 0110, 0010, 1000, 0010, 0001. From these patterns, we can deduce the names of all six lieves, and we can visit the lieves in lexicographic order — needing no link fields whatsoever! (See Theorem 2.3.1A.) In particular, the space needed in Algorithm E's `OMEM` for the calculation of (40) can be reduced from 40 bytes per node to just 10 *bits* per node.

What revisions to Algorithm E will exploit this compression scheme?

▸ **204.** [*26*] Furthermore, the trie of Fig. 125 has surplus space, because four fields appear in every node. For example, the rightmost field of the root node must always be null, and so must the rightmost fields of the nodes for prefixes $\bar{1}$ and 0, because no key of an m-class can begin with '2' or '$\bar{1}$2' or '02'.

a) In fact, show that 23 of the $4 \cdot 14$ fields in that trie are necessarily null.

b) Design an algorithm that builds more economical tries for Algorithm E, by using a *variable* number of fields in every node instead of a fixed number Δ.

205. [*HM46*] Analyze the average total number of pointers, when the method of exercise 204 builds a compact trie from n random keys $a_1 \ldots a_q$ (that is, from n independently generated marked involutions as in exercise 176, with repetitions allowed).

▸ **206.** [*30*] When a trie with P nodes is implemented as in Fig. 125, every node contains Δ link fields that hold integers in $[0\,..\,P]$. Therefore, if $P > 2^{32}$, each of those fields must have more than 32 bits (and will typically be an octabyte with 64 bits).

Suppose $P \approx 2^{35}$. Devise a way to represent P-node tries whose link fields fit in 32 bits, thereby needing only about half as much RAM. *Hint:* Use randomization.

207. [*26*] Design Algorithm E^+, a modification of Algorithm E that computes the numbers `PATH[`m`]` for $2 \le m \le n$, where `PATH[`m`]` is the number of Hamiltonian paths in the induced graph $G \mid \{1, \ldots, m\}$ and G is a given graph on vertices $\{1, \ldots, n\}$.

208. [*20*] Let G be the Petersen graph, GP(5, 2) in the notation of exercise 11, with its vertices arranged in the order (0, 1, 2, 3, 4, $0'$, $1'$, $2'$, $3'$, $4'$). How many m-cycles and m-paths of G are found by (a) Algorithm E? (b) Algorithm E^+?

▸ **209.** [*21*] The text observes that 256 gigabytes of RAM were needed for the computation of (40). Discuss the memory requirements for computing (44).

210. [*HM46*] Let $S_{m,n}$ be the number of closed $m \times n$ knight's tours, namely the number of Hamiltonian cycles in the graph of knight moves on an $m \times n$ board; and let $S_m(z) = \sum_{n \ge 0} S_{m,n} z^n$ be the corresponding generating function for boards with m rows. The periodic nature of Algorithm E proves that $S_m(z) = P_m(z)/Q_m(z)$ for certain (huge) relatively prime polynomials $P_m(z)$ and $Q_m(z)$.

Similarly, if $S^+_{m,n}$ is the number of *open* $m \times n$ knight's tours (the number of Hamiltonian *paths*), with generating function $S^+_m(z) = \sum_{n \ge 0} S^+_{m,n} z^n$, the periodic nature of Algorithm E^+ shows that $S^+_m(z) = P^+_m(z)/Q^+_m(z)$.

Prove or disprove that $Q^+_m(z)$ is a multiple of $Q^m(z)^3$ when $m \ge 5$.

periodic knight's tour
preorder
trie, compressed
compression
analysis of algorithms
compact trie
marked involutions
octabyte
data structures
RAM
Hamiltonian paths
induced graph
Petersen graph
RAM
memory, random access
graph of knight moves
generating function
open

211. [*18*] What is the other Hamiltonian cycle of the digraph (47)?

212. [*21*] The digraph (47) is an example of a *tournament* (an orientation of K_n). True or false: Any tournament for which no vertex has in-degree 0 or out-degree 0 has a Hamiltonian cycle.

213. [*35*] The 8×8 matrix (46) is just part of a 120×120 matrix in the Stanford GraphBase, which contains all of the scores of the American 1990 college football season. When the full matrix is converted to a digraph on 120 vertices, there clearly is no Hamiltonian cycle — because, for example, Fullerton won no games.

We do get a plausible digraph if we include two-way arcs for the *close* games, by saying that $u \longrightarrow v$ also when the difference between their scores is less than 10. (For example, (47) would gain 11 more arcs: Brown → Princeton, Brown → Yale, Columbia → Harvard, Cornell → Dartmouth, Harvard → Cornell, Harvard → Penn, Penn → Brown, Penn → Cornell, Princeton → Columbia, Princeton → Cornell, Princeton → Harvard.) The resulting digraph, `football-tol10.gb`, is available online.

Prove that `football-tol10.gb` has a Hamiltonian path but no Hamiltonian cycle.

215. [*21*] One of many interesting ways to orient the edges of the n-cube, illustrated for $n = 4$ in (48), is the following: Let $v \longrightarrow w$, where $v = v_1 \dots v_n$ and $w = v_1 \dots v_{j-1} \bar{v}_j v_{j+1} \dots v_n$ is the same as v but with the jth component complemented, if and only if v either has even parity (that is, $v_1 + \cdots + v_n$ is even) and j is even, or v has odd parity and j is odd.

Use Algorithm B to enumerate the Hamiltonian cycles for this orientation when $n = 5$. How does this compare to the undirected case (graph Q in Table 1)?

216. [*M20*] Another orientation of the n-cube stipulates that $v \longrightarrow w$ if and only if v either has even parity and $j = k$, or v has odd parity and $j \neq k$, where k is fixed.

Prove that the number of 2^n-cycles with this orientation is exactly twice the number of 2^{n-1}-cycles in the *un*oriented $(n-1)$-cube.

▸ **217.** [*21*] Consider the graph Π_n whose vertices are the $n!$ permutations of $\{1, \dots, n\}$, and whose edges v — w connect permutations that differ by interchanging adjacent elements: $v = v_1 \dots v_n$ and $w = v_1 \dots v_{j-1} v_{j+1} v_j v_{j+2} \dots v_n$ for some j, $1 \le j < n$.

As in exercise 215, we can orient Π_n by letting $v \longrightarrow w$ if and only if v either is an even permutation and j is even, or v is an odd permutation and j is odd.

Find, by hand, a Hamiltonian cycle of Π_4 with this orientation.

218. [*27*] Continuing exercises 216 and 217, we can also orient Π_n by letting $v \longrightarrow w$ when v either is even and $j = k$, or v is odd and $j \neq k$, for fixed k.

a) Prove by hand that Π_4 has no Hamiltonian cycle with this orientation when $k = 1$.

b) Find by computer the number of Hamiltonian cycles of Π_5 when k is (i) 1; (ii) 2.

220. [*32*] (F. Stappers, 2025.) Chess moves lead to interesting digraphs in yet another way: Place either a bishop (`B`), king (`K`), knight (`N`), queen (`Q`), or rook (`R`) on each cell of a board, and create a cycle where (i) every piece can capture its successor; and (ii) no piece captures a piece of the same kind.

Consider, for example, the following three placements on a 5×5 board:

$\mathtt{B}_{00}\ \mathtt{K}_{01}\ \mathtt{N}_{02}\ \mathtt{Q}_{03}\ \mathtt{R}_{04}$	$\mathtt{B}_{00}\ \mathtt{R}_{01}\ \mathtt{B}_{02}\ \mathtt{Q}_{03}\ \mathtt{B}_{04}$	$\mathtt{B}_{00}\ \mathtt{N}_{01}\ \mathtt{Q}_{02}\ \mathtt{N}_{03}\ \mathtt{Q}_{04}$
$\mathtt{K}_{10}\ \mathtt{N}_{11}\ \mathtt{Q}_{12}\ \mathtt{R}_{13}\ \mathtt{B}_{14}$	$\mathtt{R}_{10}\ \mathtt{B}_{11}\ \mathtt{N}_{12}\ \mathtt{B}_{13}\ \mathtt{Q}_{14}$	$\mathtt{B}_{10}\ \mathtt{N}_{11}\ \mathtt{Q}_{12}\ \mathtt{N}_{13}\ \mathtt{R}_{14}$
$\mathtt{N}_{20}\ \mathtt{Q}_{21}\ \mathtt{R}_{22}\ \mathtt{B}_{23}\ \mathtt{K}_{24}$;	$\mathtt{R}_{20}\ \mathtt{N}_{21}\ \mathtt{K}_{22}\ \mathtt{N}_{23}\ \mathtt{Q}_{24}$;	$\mathtt{K}_{20}\ \mathtt{K}_{21}\ \mathtt{R}_{22}\ \mathtt{Q}_{23}\ \mathtt{R}_{24}$.
$\mathtt{Q}_{30}\ \mathtt{R}_{31}\ \mathtt{B}_{32}\ \mathtt{K}_{33}\ \mathtt{N}_{34}$	$\mathtt{R}_{30}\ \mathtt{N}_{31}\ \mathtt{K}_{32}\ \mathtt{N}_{33}\ \mathtt{Q}_{34}$	$\mathtt{Q}_{30}\ \mathtt{K}_{31}\ \mathtt{R}_{32}\ \mathtt{R}_{33}\ \mathtt{B}_{34}$
$\mathtt{R}_{40}\ \mathtt{B}_{41}\ \mathtt{K}_{42}\ \mathtt{N}_{43}\ \mathtt{Q}_{44}$	$\mathtt{R}_{40}\ \mathtt{K}_{41}\ \mathtt{K}_{42}\ \mathtt{K}_{43}\ \mathtt{Q}_{44}$	$\mathtt{K}_{40}\ \mathtt{K}_{41}\ \mathtt{B}_{42}\ \mathtt{B}_{43}\ \mathtt{N}_{44}$

tournament
football scores
Stanford GraphBase
online
orient the edges
n-cube
parity
permutations
Stappers
Chess moves
bishop
king
knight
queen
rook

The first example yields 2,016,000 cycles, and (B00 K33 B23 K01 N02 B14 Q03 N43 K24 N34 K42 R31 N11 B32 Q21 R22 Q12 R13 K10 N20 B41 Q30 R40 Q44 R04 B00) is lexicographically least.

a) Find one of the 29201 cycles for the middle example.

b) Can you find the unique cycle for the rightmost example?

221. [*22*] Continuing the previous exercise, construct (by hand) a 4×4 placement that (i) uses each of the five allowable pieces at least once, and (ii) has a unique cycle.

222. [*24*] Let $NE(m,n)$ be the digraph whose vertices are ij for $0 \le i < m$ and $0 \le j < n$ and whose arcs represent "north" or "east" or "northeast" on the $m \times n$ torus: $ij \longrightarrow ((i-1) \bmod m)j$, $ij \longrightarrow i((j+1) \bmod n)$, $ij \longrightarrow ((i-1) \bmod m)((j+1) \bmod n)$. (Every vertex has out-degree 3 and in-degree 3.)

a) Sketch the nonisomorphic Hamiltonian cycles of $NE(3,4)$.

b) Count the Hamiltonian cycles of $NE(m,n)$ for $2 \le m, n \le 8$.

▸ **223.** [*28*] (*Shift and save or bump.*) Let $SB(m,n)$ be the digraph whose vertices are the m-ary strings $x_1x_2\ldots x_n$ with $0 \le x_k < m$ for $1 \le k \le n$ and whose arcs are $x_1x_2\ldots x_n \longrightarrow x_2\ldots x_nx_1$, $x_1x_2\ldots x_n \longrightarrow x_2\ldots x_nx_1^+$, where $x^+ = (x+1) \bmod m$. (Notice that there are self-loops whenever $x_1 = x_2 = \cdots = x_n$.)

a) True or false: Every vertex of $SB(m,n)$ has in-degree 2.

b) Find a connection between the Hamiltonian cycles of $SB(m,n)$ and something else that has been studied in Section 7.2.1.1.

c) Prove that $SB(m,2)$ has no Hamiltonian cycles when $m > 2$.

d) Find a Hamiltonian cycle of $SB(3,3)$.

e) Let C be a Hamiltonian cycle of $SB(m,n)$. Prove that there's a set S of m-ary strings $y_1\ldots y_{n-1}$ such that $x_1x_2\ldots x_n \longrightarrow x_2\ldots x_nx_1$ is in C if and only if $x_2\ldots x_n \in S$.

f) How many Hamiltonian cycles are in $SB(m,3)$, for $2 \le m \le 7$?

▸ **224.** [*M35*] Prove that $SB(m,n)$ isn't Hamiltonian when $m \bmod 4 = 3$ and n is even.

225. [*46*] Construct a Hamiltonian cycle in $SB(m,3)$ for all $m > 1$.

227. [*M20*] When does a move from cell (i,j) to cell (i',j') of a rectangular board go "counterclockwise" with respect to a given pivot point (p,q)? State the answer as an algebraic relation in terms of i, j, i', j', p, and q.

228. [*17*] How can the middle example in (51) be "uniquely steady" (in the sense of nearly equal distances between plumb-line crossings) when it isn't symmetrical?

229. [*33*] An $m \times n$ *whirling knight's tour* is a Hamiltonian cycle on the following digraph: There are $mn - [mn \text{ is odd}]$ vertices (i,j) for $0 \le i < m$ and $0 \le j < n$, omitting $(\frac{m-1}{2}, \frac{n-1}{2})$ when mn is odd. The arcs are $(i,j) \longrightarrow (i',j')$ when we have $(i-i')^2 + (j-j')^2 = 5$ and (i',j') is counterclockwise from (i,j) with respect to the pivot point $(\frac{m-1}{2}, \frac{n-1}{2})$, in the sense of exercise 227. The cycle has c *coils* if it crosses a plumb-line above the pivot point exactly c times. Let $W_{m,n} = \sum_{c \ge 0} W_{m,n}^{(c)}$, where $W_{m,n}^{(c)}$ is the number of $m \times n$ whirling knight's tours with c coils.

a) Can there be a knight move for which neither $(i,j) \longrightarrow (i',j')$ nor $(i',j') \longrightarrow (i,j)$?

b) Prove that $W_{m,n} = 0$ when $n \ge 2m - 1$.

c) If $m \le n$, prove that $W_{m,n}^{(c)} = 0$ when $c < m/2$ or $c > m$.

d) Use Algorithm B to compute $W_{m,n}^{(c)}$ for $m \le n \le 10$ and $n/2 \le c \le n$.

230. [*15*] True or false: The transpose of a whirling knight's tour is also a whirling knight's tour, when traveled in the opposite direction.

unique cycle
$NE(m,n)$
$m \times n$ torus
Shift and save or bump
$SB(m,n)$
m-ary strings
self-loops
counterclockwise
pivot point
plumb-line
symmetrical
angular velocity
whirling knight's tour
coils

231. [*37*] Find $n \times n$ whirling knight's tours that have exactly (a) $n/2$ (b) n coils. (The computations of exercise 229 show that no such tours exist when $n < 12$.)

▸ **232.** [*30*] Find infinitely many solutions to problem 231(b).

235. [*M30*] An $m \times n$ *whirling king's tour* is a Hamiltonian cycle on a digraph like the one in exercise 229, except that king moves replace knight moves; more precisely, the statement '$(i-i')^2+(j-j')^2 = 5$' is replaced by '$1 \le (i-i')^2+(j-j')^2 \le 2$'. Let $X_{m,n}$ and $X^{(c)}_{m,n}$ count whirling king's tours, by analogy with $W_{m,n}$ and $W^{(c)}_{m,n}$ in exercise 229.

a) Prove that $X_{n,n} > 0$ for all $n > 1$. (When $n = 1$ there are no vertices!)

b) Furthermore $X^{(c)}_{m,n} \neq 0$ and $X^{(c')}_{m,n} \neq 0$ implies $c = c'$.

c) Furthermore $X_{m,n} = 0$ when m is odd and $n > m+1$.

d) Furthermore $X_{m,n} = X_{m,n-2}$ when $n \ge 2m$.

236. [*21*] Continuing exercise 235, compute $X_{m,n}$ for $m \le n \le 8$. Look for symmetries!

240. [*05*] True or false: A bidirected path or cycle is always a trail.

241. [*16*] Construct a bidirected graph that has a trail between u and v but no path.

▸ **242.** [*M22*] The "complete bidirected graph" on vertices $\{v_1, \ldots, v_n\}$ has $v_i \langle\langle v_j$, $v_i \langle\rangle v_j$, $v_i \rangle\langle v_j$, and $v_i \rangle\rangle v_j$ for $1 \le i < j \le n$.

a) How many Hamiltonian (i) paths (ii) cycles does it have?

b) How many does it have if we omit the directed edges?

c) How many does it have if we omit the extroverted edges?

▸ **243.** [*M22*] Notice that the 32 "fers edges" $2 \text{—} 3$, $4 \text{—} 5$, $\ldots$, $64 \text{—} 1$ in the middle diagram of (55) make a simple pattern of 16 X's. And so do the 32 "alfil edges" in the right-hand diagram (although the X's overlap in that case).

Prove that a Hamiltonian cycle of alternating knight+fers moves on an $m \times n$ board is possible only if both m and n are even; furthermore, the fers moves always make a fixed pattern of $(m/2)(n/2)$ X's. Also prove a similar statement for knight+alfil tours.

244. [*20*] Using Algorithm B, determine the number of closed tours on an 8×8 board that alternately make the moves of a knight and (a) a fers; (b) an alfil. (See (55).)

245. [*17*] True or false: The graph $G(B)$ in Algorithm B is bipartite $\Longleftrightarrow$ B is directed.

▸ **246.** [*25*] Extend the trick of (54) from undirected graphs to directed graphs: Given any two directed graphs G and H on the same vertices, construct a bidirected graph B whose Hamiltonian cycles are in one-to-one correspondence with the Hamiltonian cycles that strictly alternate between arcs of G and arcs of H. (For example, suppose the vertices are $\{0, 1, \ldots, 9\}$ and the arcs of G are $v \to (v+1) \bmod 10$ and the arcs of H are $v \Rightarrow (v+3) \bmod 10$. Then there are two strictly alternating Hamiltonian cycles: $0 \to 1 \Rightarrow 4 \to 5 \Rightarrow 8 \to 9 \Rightarrow 2 \to 3 \Rightarrow 6 \to 7 \Rightarrow 0$ and $0 \Rightarrow 3 \to 4 \Rightarrow 7 \to 8 \Rightarrow 1 \to 2 \Rightarrow 5 \to 6 \Rightarrow 9 \to 0$.) *Hint:* B can have more vertices than G and H.

247. [*21*] Find all of the "whirling" 8×8 cycles that strictly alternate between knight moves and grid moves (as in (54)), always counterclockwise with respect to the center.

250. [*22*] The Stanford GraphBase represents a directed graph by specifying, for each vertex v, a linked list of the arcs from v to its successor vertices. Every element of this list is an "arc node," and each arc node a has fields `TIP(`a`)` and `NEXT(`a`)`. If the successors of v are w_1, w_2, $\ldots$, w_d, the list contains arc nodes a_1, a_2, $\ldots$, a_d, where

$$\texttt{ARCS(}v\texttt{)} = a_1,\ \texttt{TIP(}a_1\texttt{)} = w_1,\ \texttt{NEXT(}a_1\texttt{)} = a_2,\ \texttt{TIP(}a_2\texttt{)} = w_2,\ \ldots,\ \texttt{TIP(}a_d\texttt{)} = w_d,\ \texttt{NEXT(}a_d\texttt{)} = \Lambda.$$

(See, for example, 7–(31) and Algorithm 7B, near the beginning of Chapter 7.)

whirling king's tour
symmetries
bidirected path
trail
bidirected graph
complete bidirected graph
extroverted edges
fers
alfil
digraph
directed graphs
strictly alternating Hamiltonian cycles
whirling
counterclockwise
Stanford GraphBase
digraph, representation of
linked list
arc node
`TIP(`a`)`
`NEXT(`a`)`

Arc nodes also contain more. For example, there's a LEN field: An arc of length l from v to w is represented by an arc node with TIP(a) $= w$ and LEN(a) $= l$.

To represent a *bidirected graph*, we can use the two low bits of each LEN field, by representing a bidirected edge between v and w as an arc of length l from v to w, where

$$v \rangle\rangle w \iff l \bmod 4 = 3; \qquad v \rangle\langle w \iff l \bmod 4 = 2;$$
$$v \langle\langle w \iff l \bmod 4 = 1; \qquad v \langle\rangle w \iff l \bmod 4 = 0.$$

(Since $v \rangle\rangle w$ means the same as $w \langle\langle v$, we could represent that edge also as an arc with $l \bmod 4 = 1$ from w to v. Both arcs could, in fact, be specified. Note that LEN(a) has no connection whatsoever with the LEN field of Algorithm 7.2.2.1X and its cousins.)

Explain how to initialize the NBR and ADJ arrays in step B1 of Algorithm B, when the bidirected input graph B is represented in this way.

▸ **251.** [*25*] How should the data structures be set up in step B3 of Algorithm B so that the first edge of the partial Hamiltonian matching is CURU — CURV?

252. [*22*] Exactly what changes to the data structures should be made in step B8 of Algorithm B when we have (a) MATE(u) < 0? (b) MATE(u) ≥ 0?

▸ **253.** [*26*] When Algorithm B discovers a Hamiltonian matching in step B13, what's a good way for it to "unscramble" the corresponding Hamiltonian cycle of the input graph and to print it in a format like (53)? (Compare with exercise 113.)

254. [*20*] The text sketches the steps that lead to (60) and the first Hamiltonian matching of (57). What happens next?

256. [*21*] Customize Algorithm B to the special case where the input is simply an ordinary *directed* graph, by showing that the answers to exercises 250 and 253 can be significantly simplified in that case.

▸ **270.** [*22*] If G is a digraph on n vertices, define $\widehat{G}$ by adding two vertices s and t, with $2n$ additional arcs $s \longrightarrow v$ and $v \longrightarrow t$ for all v in G; also $t \longrightarrow s$.

a) Show that G has a Hamiltonian path if and only if $\widehat{G}$ has a Hamiltonian cycle.
b) Prove that if G has no cycles, it has at most one Hamiltonian path.
c) True or false: Algorithm B will handle $\widehat{G}$ in linear time when G is acyclic.

271. [*M25*] If G is a digraph with m arcs, n vertices, and p cycles, show that we need at most $O(n^p(m+n))$ steps to test whether or not G has a Hamiltonian path.

272. [*22*] How can Algorithm B be used to visit all Hamiltonian *paths* of a bidigraph?

▸ **275.** [*41*] Develop a dynamic enumeration algorithm, analogous to Algorithm E, that counts the number of Hamiltonian cycles in $B \mid \{1, \ldots, m\}$ for all $1 \leq m \leq n$, when B is a given *bidirected* graph on n vertices.

276. [*21*] The 4×4 tour (54), in which knight moves alternate with grid moves, is sometimes called a "knight + wazir tour," because the "wazir" was an early chess piece that moved only one step horizontally or vertically. Such tours seem to outnumber knight's tours; for example, there are 5350996 of them on a 6×6 board, compared to just 9862 closed tours of a knight alone. Thus we can expect an 8×8 board to have a huge number of knight + wazir tours, many more than we could hope to visit with Algorithm H.

Use exercise 275 to count the closed $8 \times n$ knight + wazir tours for $n \leq 32$.

277. [*20*] The knight + wazir tour in (54) is a Hamiltonian cycle of a bidirected graph. Show that it's also a Hamiltonian cycle of an appropriate *directed* graph.

length l
LEN(a)
bidigraph, representation of
LEN field
Hamiltonian matching
unscramble
Hamiltonian path
Hamiltonian *paths* of a bidigraph
dynamic enumeration
grid moves
knight + wazir tour

complemented brackets, see switching equivalenc
switching equivalent
canonical
directed
cyclic
bireachable
strongly connected
equivalence relation
Oriented grids
balanced *digraph*
always-turn
maybe-turn
parity

279. [*27*] If v is a vertex of a bidirected graph B, let $B \oplus v$ be the bidirected graph in which the brackets next to v in all edges have been complemented. More precisely, every edge $v \langle\langle w$, $v \langle\rangle w$, $v \rangle\langle w$, or $v \rangle\rangle w$ involving v and another vertex becomes respectively $v \rangle\langle w$, $v \rangle\rangle w$, $v \langle\langle w$, or $v \langle\rangle w$; also $v \langle\langle v$ becomes $v \rangle\rangle v$ (no change), $v \langle\rangle v$ becomes $v \rangle\langle v$, $v \rangle\langle v$ becomes $v \langle\rangle v$. (Thus $v^+ \leftrightarrow v^-$ in the graph $G(B)$ of (56).)

There's an obvious isomorphism between the walks (53) of B and the walks of $B \oplus v$. Therefore if B has n vertices and $S = \{v_1, \ldots, v_k\}$ is any subset of those vertices, we get up to 2^n bidirected graphs $B \oplus S = B \oplus v_1 \oplus \cdots \oplus v_k$, which all have essentially the same path structure. (Sometimes $B \oplus S = B$.) The bidigraphs B and $B \oplus S$ are said to be *switching equivalent*.

Design an algorithm that finds a *canonical* equivalent $[B]$ of any given bidirected graph B; that is, $[B] = [B']$ if and only if B and B' are switching equivalent. Furthermore $[B]$ should be directed if and only if B is directed.

▸ **281.** [*30*] In a bidirected graph B, say that an edge is *cyclic* if it is part of at least one bidirected cycle. Also say that vertex v is *bireachable* from vertex u, written '$u \Rightarrow v$', if $u = v$ or there exist bidirected paths of the forms $u \langle \cdots v$ and $u \rangle \cdots v$, where each '$\cdots$' can independently be filled in with some sequence of cyclic edges. Furthermore, u and v are said to be *strongly connected*, written $u \Leftrightarrow v$, if $u \Rightarrow v$ and $v \Rightarrow u$.

a) For example, if B has five vertices $\{a, b, c, d, e\}$ and six edges, where all of the edges occur in the two cycles $a \langle\langle b \langle\rangle c \rangle\rangle d \rangle\langle a$, $a \rangle\langle d \langle\langle c \langle\langle e \langle\rangle a$, show that $b \Leftrightarrow e$.

b) If $u \Rightarrow v$ and $v \langle\langle w$ is a cyclic edge of B, prove that $u \Rightarrow w$.

c) Let $u \mathrel{—} v$ if u and v are connected by a sequence $u = u_0, \ldots, u_l = v$ of cyclic edges $u_{j-1} \mathbin{\mathbb{I}_j \mathbb{I}'_j} u_j$ for $1 \le j \le l$. Prove that $u \Rightarrow v$ and $v \mathrel{—} w$ implies $u \Rightarrow w$.

d) Prove that '$\Leftrightarrow$' is an equivalence relation.

e) In fact, $u \Leftrightarrow v$ if and only if $u \mathrel{—} v$.

282. [*22*] If the definition of bireachability in exercise 281 had allowed '$\cdots$' to contain *arbitrary* edges of B, not only the cyclic edges, show that '$\Leftrightarrow$' wouldn't necessarily have been an equivalence relation.

283. [*22*] Let B be a strongly connected bidigraph; in other words, $u \Leftrightarrow v$ for all u and v, in the sense of exercise 281. We can deduce further structure by defining

$$\begin{aligned}
u \langle\sim\rangle v &\iff \text{there is no bidirected path } u\langle \cdots \rangle v \text{ of cyclic edges;}\\
u \langle\sim\langle v &\iff \text{there is no bidirected path } u\langle \cdots \langle v \text{ of cyclic edges;}\\
u \rangle\sim\rangle v &\iff \text{there is no bidirected path } u\rangle \cdots \rangle v \text{ of cyclic edges;}\\
u \rangle\sim\langle v &\iff \text{there is no bidirected path } u\rangle \cdots \langle v \text{ of cyclic edges.}
\end{aligned}$$

By convention, all four of these relations are true when $u = v$.

a) Prove that '$\langle\sim\rangle$' and '$\rangle\sim\langle$' are equivalence relations.

b) Prove that $u \langle\sim\rangle v$ and $v \langle\sim\langle w$ implies $u \langle\sim\langle w$.

▸ **285.** [*M32*] (*Oriented grids.*) The infinite 4-regular graph of all integer points ij on the plane, with $(i \pm 1)j \mathrel{—} ij \mathrel{—} i(j \pm 1)$ for all i and j, can be converted to an infinite balanced *digraph* in at least two interesting ways, "always-turn" and "maybe-turn":

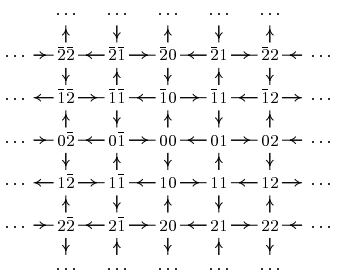
direction	arc		always-turn	maybe-turn
north	$ij \longrightarrow (i-1)j$	$\iff$	$i+j$ even	j even
south	$ij \longrightarrow (i+1)j$	$\iff$	$i+j$ even	j odd
east	$ij \longrightarrow i(j+1)$	$\iff$	$i+j$ odd	i even
west	$ij \longrightarrow i(j-1)$	$\iff$	$i+j$ odd	i odd

A vehicle driving from one vertex to another in the directed *always*-turn grid must make a 90° turn at every intermediate point. But in the directed *maybe*-turn grid, one

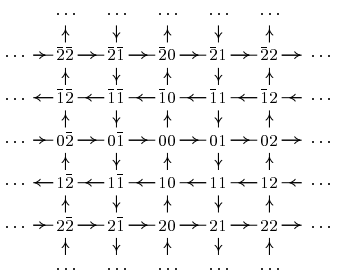
of the two options is always to continue in the same direction. (The latter digraph is often called the *Manhattan digrid*, because it is similar to many of the one-way street patterns in midtown Manhattan.)

Here's how those digrids look in the vicinity of vertex 00 (using '$\bar{1}$' for '-1', etc.):

always-turn



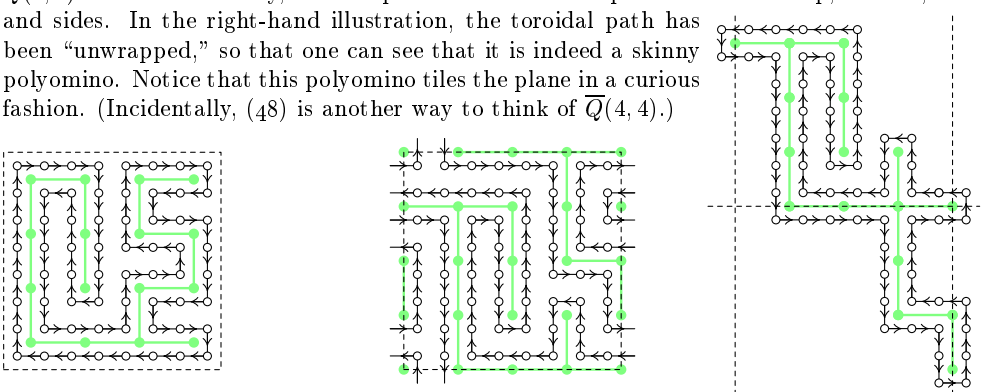
maybe-turn



a) Find a formula for the distance from 00 to ij in the always-turn digrid.
b) Find a formula for the distance from 00 to ij in the Manhattan digrid.
c) Show that the Manhattan digrid can be regarded as the *arc digraph* of the always-turn digrid: Every always-turn arc corresponds to a maybe-turn vertex, and always-turn trails correspond to maybe-turn paths. *Hint:* Rotate one grid by 45°.
d) Find a Hamiltonian path on the (infinite) Manhattan digrid. (By (c), this path corresponds to an infinite *Eulerian trail* on the always-turn digrid!)

▸ **286.** [*M26*] Let $Q(m,n)$ be the digraph that's obtained when the $m \times n$ grid $P_m \,\square\, P_n$ is given the Manhattan maybe-turn orientation of exercise 285. Also let $\overline{Q}(m,n)$ be the analogous digraph obtained by orienting the edges of the $m \times n$ torus $C_m \,\square\, C_n$. Both m and n should be even. It turns out that the Hamiltonian cycles of $Q(m,n)$ and $\overline{Q}(m,n)$ form the boundaries of special kinds of "skinny polyominoes"; typical examples with $m = n = 8$ are shown below. Shaded dots and lines have been inserted into the "spines" of these polyominoes in order to make the patterns clearer.

The directed cycle in $Q(8,8)$ is easy to understand. But the directed cycle in $\overline{Q}(8,8)$ is somewhat tricky, because paths on a torus "wrap around" at the top, bottom, and sides. In the right-hand illustration, the toroidal path has been "unwrapped," so that one can see that it is indeed a skinny polyomino. Notice that this polyomino tiles the plane in a curious fashion. (Incidentally, $(_4^8)$ is another way to think of $\overline{Q}(4,4)$.)



a) Show that the number of Hamiltonian cycles in $Q(m,n)$ is $c(P_{m/2} \,\square\, P_{n/2})$, the number of spanning trees in an $(m/2) \times (n/2)$ grid. (See exercise 7.2.1.6–106.)
b) Show that the number of Hamiltonian cycles in $\overline{Q}(m,n)$ is $2c(C_{m/2} \,\square\, C_{n/2})$, twice the number of spanning trees in an $(m/2) \times (n/2)$ torus.

Manhattan digrid
digrids
$\bar{1}$
trails
paths
Hamiltonian path
infinite Hamiltonian path
Eulerian trail
grid
torus
skinny polyominoes
polyominoes
pi, as source
spanning trees

▸ **287.** [*HM37*] The purpose of this exercise is to determine the asymptotic growth rate of the cycle counts in the previous exercise.

a) Let $f(x,y)$ be a doubly periodic function: $f(x,y)=f(x+1,y)=f(x,y+1)$ for all real x and y. Assume also that the double integral $\int_0^1\int_0^1 f(x,y)\,dy\,dx$ exists. Show that $\int_{-1}^1(\int_{\max(0,x^3)}^{\min(1,(x+1)^3)}f(x,y)\,dy)\,dx=\int_0^1\int_0^1 f(x,y)\,dx\,dy$.

b) True or false? $\int_0^1\int_0^1 f(5x,8y)\,dy\,dx=\int_0^1\int_0^1 f(x,y)\,dy\,dx$.

c) If $g(s,t)=f(s+t,-s+t)$, show that $\int_0^1\int_0^1 g(s,t)\,dt\,ds=\int_0^1\int_0^1 f(x,y)\,dy\,dx$.

d) Show that the number of cycles in exercise 286(b) can be expressed in terms of the trigonometric product $P(m,n)=\prod_{j=1}^{m-1}\prod_{k=1}^{n-1}(4\sin^2\frac{j\pi}{m}+4\sin^2\frac{k\pi}{n})$.

e) Let $p(x,y)$ be the doubly periodic function $\ln(\sin^2\pi x+\sin^2\pi y)$, and integrate it to get $I=\int_0^1\int_0^1 p(x,y)\,dy\,dx$. Prove that the growth ratio $P(m,n)^{1/mn}$ is asymptotically equal to $\exp\big(\ln 4+I+O(\frac{1}{m})+O(\frac{1}{n})\big)$ as $m\to\infty$ and $n\to\infty$.

f) Show that $I=4G/\pi-\ln 4$, where G is Catalan's constant $\sum_{k=0}^{\infty}(-1)^k/(2k+1)^2$.

290. [*M20*] The at-least-one and at-most-one clauses illustrated in (64) and (65) guarantee that every vertex has exactly one successor and exactly one predecessor.

a) Would it be sufficient to require only that every vertex has exactly one successor?

b) Would it be sufficient to require only that every vertex has at least one successor and at most one predecessor?

291. [*21*] Many ways are known by which the at-most-one condition can be encoded in clauses for SAT. (See, for example, *langford*(n), *langford*$'(n)$, *langford*$''(n)$, and *langford*$'''(n)$ in Section 7.2.2.2.) Is the simple scheme illustrated in (65) a good choice?

▸ **292.** [*22*] In the text's example of CEGAR cuts, show that the cycles of Fig. 127(c) could have been merged in another way, leading directly to a Hamiltonian cycle without a third round of SAT solving. On the other hand, the process might have been unlucky, because a cycle cover different from Fig. 127(e) also satisfies (63)–(65) and (67)–(70).

294. [*15*] If literal l in Algorithm C represents $\overline{uv}$, what does literal $l\oplus 3$ represent?

296. [*20*] Spell out the low-level details of step C2 (cycle cover clauses).

297. [*20*] Spell out the low-level details of step C8 (cut clauses).

▸ **298.** [*30*] Spell out the low-level details of step C6 (cycle merging).

▸ **300.** [*M30*] A graph is *mergeable* if every cycle cover with more than one cycle contains two cycles that can be merged. (Thus, Algorithm C will never call the solver twice.)

a) Prove that every complete k-partite graph is mergeable.

b) True or false: The $m\times n$ grid graph $P_m\,\square\,P_n$ is mergeable for all $m,n\ge 2$.

c) Prove that, for $n\ge 4$, the triangular grid of order n is not mergeable.

d) Prove that, for $n\ge 4$, the n-cube isn't mergeable.

▸ **302.** [*M25*] Let $N_{t,m}$ be the "non-tough" graph $(t+1)K_m$ — K_t. (For example, graph D in Table 1 is the special case $N_{5,3}$.)

a) Describe all cycle covers of $N_{t,m}$ for which no two cycles can be merged.

b) Describe all cut clauses that might arise when Algorithm C is applied to $N_{t,m}$.

c) What are the maximum and minimum number of rounds needed by Algorithm C to prove that $N_{t,m}$ has no Hamiltonian cycle?

303. [*24*] Compare Algorithm C to Algorithm H on the fourteen graphs of Table 1, when Algorithm H has been modified to stop after finding one solution.

doubly periodic function
asymptotically
Catalan's constant
at-least-one
at-most-one
langford(n)
CEGAR cuts
cycle cover
merging
mergeable
complete k-partite graph
grid graph
triangular grid
n-cube
non-tough

340. [*20*] Given a bipartite graph G with n vertices in each part, construct an exact cover problem with $3n$ primary items u^-, u^+, v: two for each vertex u in the first part, and one for each vertex v in the second part. Let the options be 'u^- v w^+', for all triples with $u \mathrel{—} v \mathrel{—} w$ and $u \neq w$.

a) What do the solutions to this exact cover problem represent?

b) Experiment with this construction when G is the 6×6 knight graph.

▸ **341.** [*27*] How many 8×8 closed knight's tours have the property that moves k and $32 + k$ occupy the same column, for $1 \le k \le 32$? *Hint:* Define an exact cover problem.

343. [*24*] Find a knight's tour whose step matrix has

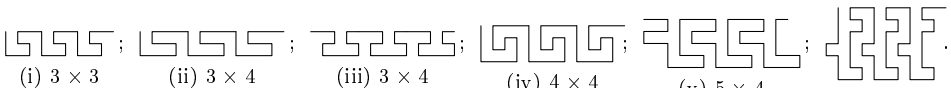
$$a_{11} = 1,\ \ a_{16} = 16,\ \ a_{32} = 64,\ \ a_{52} = 32,\ \ a_{71} = 33,\ \ a_{83} = 34,$$

and such that $1 \le a_{ij} \le 18$ implies $a_{(9-i)(9-j)} = 50 - a_{ij}$. (The latter condition means that moves 1, 2, ..., 18 are rotated 180° from moves 49, 48, ..., 32.)

▸ **344.** [*24*] (G. E. Carpenter, 1881.) Find a knight's tour for which the top row of the step matrix is '1 4 9 16 25 36 49 64'. How many such tours exist?

▸ **350.** [*M27*] Exactly how many Hamiltonian cycles are possible in the *Sierpiński gasket graph* $S_n^{(3)}$? (See Fig. 113, near 7.2.2.3–(69).) *Hint:* There is a fairly simple formula.

▸ **360.** [*25*] An $m \times n$ *meander frieze* is a Hamiltonian cycle of $P_m \mathbin{\square} C_n$ that isn't also a Hamiltonian cycle of $P_m \mathbin{\square} P_n$. For example, a few of the possibilities are



(i) 3×3 (ii) 3×4 (iii) 3×4 (iv) 4×4 (v) 5×4 (vi) 7×3

(Such friezes were often used as ornamentation on Greek vases of the "late Geometric" period, c. 750 B.C. For example, the famous Dipylon Amphora was decorated in part with the unusual frieze (vi) and several copies of (i) and (v).)

a) The *margins* of a meander frieze are the sequences $v_1 \ldots v_{m-1}$ and $h_1 \ldots h_n$, where there are v_i edges between rows $i-1$ and i and h_j edges between columns $j-1$ and j. For example, the margins of (iv) are $v_1 v_2 v_3 = 242$ and $h_1 h_2 h_3 h_4 = 1331$. True or false: v_i is always even and h_j is always odd.

b) Prove that no $m \times n$ meander frieze has m even and n odd.

c) A meander frieze is *reduced* if it doesn't have $v_i = v_{i+1} = n$ or $h_j = h_{j+1} = m$ for any i or j. (For example, (ii) reduces to (i) because it has $h_2 = h_3 = 3$.) Find an example of an unreduced frieze for which $v_2 = v_3 = n$.

d) Draw all the meander friezes with $m \le 8$ and $n = 3$. Are any of them symmetric?

e) An $m \times n$ meander frieze is *periodic* if it's the same as d copies of an $m \times (n/d)$ meander frieze, for some proper divisor of n. Draw all of the nonperiodic, reduced meander friezes with $m = 3$ and $n \le 8$. Which of them are symmetric?

f) How many automorphisms does the graph $P_m \mathbin{\square} C_n$ have, when $m, n \ge 3$?

g) Count the nonisomorphic, nonperiodic, reduced meander friezes with $3 \le m, n \le 7$.

h) How many of the friezes in (g) are symmetrical?

i) Draw all of the friezes in (g) that have fourfold symmetry.

361. [*M21*] Describe the nonisomorphic Hamiltonian cycles of the torus $C_3 \mathbin{\square} C_4$.

369. [*22*] (*3D knight's tours.*) The $l \times m \times n$ knight graph has vertices ijk for $0 \le i < l$, $0 \le j < m$, $0 \le k < n$ and edges $ijk \mathrel{—} i'j'k'$ where $(i-i')^2 + (j-j')^2 + (k-k')^2 = 5$.

bipartite
exact cover problem
knight graph
step matrix
rotated 180°
near symmetry
Bergholtian symmetry, partial
Carpenter
Sierpiński gasket graph
meander frieze
frieze
Greek vases
vases
Geometric
Dipylon Amphora
margins
parity
reduced
symmetric
periodic
automorphisms
fourfold symmetry
torus
3D knight's tours
knight graph

If $a, b, c \in \{0, 1\}$, say that the *abc-complement* of ijk is

$$ijk^{abc} = (a?\ l-1-i:\ i)\,(b?\ m-1-j:\ j)\,(c?\ n-1-k:\ k),$$

and the *abc*-complement of edge $ijk \mathrel{—} i'j'k'$ is $(ijk \mathrel{—} i'j'k')^{abc} = ijk^{abc} \mathrel{—} i'j'k'^{abc}$.

a) A cycle C in the $l \times m \times n$ knight graph has *central symmetry* if $(u \mathrel{—} v)^{111} \in C$ whenever $u \mathrel{—} v \in C$. Find a centrally symmetric $2 \times 3 \times 4$ knight's cycle.

b) Do centrally symmetric $4 \times 4 \times 4$ knight's cycles exist?

c) Find a symmetry of the $4 \times 4 \times 4$ knight's cycle (000 012 031 133 213 023 121 331 233 113 323 221 202 003 011 032 130 210 020 122 101 300 312 232 033 021 002 100 220 010 112 131 330 322 301 203 123 313 211 001 103 223 013 111 132 333 321 302 200 120 310 212 231 030 022 102 303 311 332 230 110 320 222 201).

d) Indeed, 48 of the 64 edges of that cycle form a set S that's "fully symmetric," in the sense that every *abc*-complement of every edge in S is also present in S.

e) How many $4 \times 4 \times 4$ knight's cycles include all of the edges in that set S?

370. [*M30*] The *Grabarchuk graph* is the graph on $4^3 = 64$ vertices xyz, $0 \le x, y, z < 4$, where $xyz \mathrel{—} x'y'z' \iff$ the Euclidean distance between (x, y, z) and (x', y', z') is 3.

a) Prove that the Grabarchuk graph is bipartite, and regular of degree 6.

b) What are its symmetries (automorphisms)?

c) Find three Hamiltonian cycles that, together, contain all of its edges.

▸ **371.** [*M38*] When r is even, an r-regular graph is called *Hamilton decomposable* when its edges can be partitioned into $r/2$ Hamiltonian cycles. Such graphs are not at all rare.

a) Show that K_{2n+1} is Hamilton decomposable.

b) Show that $C_m \mathbin{\square} C_n$ is Hamilton decomposable for all $m, n \ge 3$.

c) If the edges of G can be decomposed into two Hamiltonian cycles, prove that the edges of $G \mathbin{\square} C_n$ can be decomposed into three Hamiltonian cycles, for all $n \ge 3$.

d) Suppose G and G' are Hamilton decomposable, where G is r-regular and G' is r'-regular and $r \le r' \le 2r$. Prove that $G \mathbin{\square} G'$ is also Hamilton decomposable. *Hint:* Use (c) and the distributive law $(G \cup G') \mathbin{\square} H = (G \mathbin{\square} H) \cup (G' \mathbin{\square} H)$.

e) Show that $C_{m_1} \mathbin{\square} \cdots \mathbin{\square} C_{m_t}$ is Hamilton decomposable, when $m_j \ge 3$ for $1 \le j \le t$.

372. [*M29*] Recall that every vertex of an r-regular digraph D has in-degree and out-degree r. By analogy with exercise 371, we say that D is Hamilton decomposable if its arcs can be partitioned into r directed Hamiltonian cycles.

a) Show that $\vec{C_n} \mathbin{\square} \vec{C_n}$ is Hamilton decomposable for all $n \ge 2$.

b) Show that the symmetric digraph $K^{\leftrightarrow}_{3,3}$ is Hamilton decomposable, where '$\leftrightarrow$' means that every edge $u \mathrel{—} v$ has been replaced by two arcs $u \longrightarrow v$ and $v \longrightarrow u$.

c) Show that $\vec{C_3} \mathbin{\square} \vec{C_3} \mathbin{\square} \vec{C_3}$ is Hamilton decomposable.

d) But $\vec{C_2} \mathbin{\square} \vec{C_2} \mathbin{\square} \vec{C_2}$ is *not* Hamilton decomposable.

e) On the other hand, there *is* a Hamiltonian decomposition of $\vec{C_2} \mathbin{\square} \vec{C_2} \mathbin{\square} \vec{C_2} \mathbin{\square} \vec{C_2} \mathbin{\square} \vec{C_2}$.

375. [*20*] (R. Wolf, 1894.) Choose a uniformly random square of the chessboard and mark it '1'. After a square has been marked 'k', choose a uniformly random unmarked square that's a knight's move away, and mark it 'k+1'; but stop if there's no such square.

Empirically estimate the probability that (a) the final square is marked 'k', given $1 \le k \le 64$; (b) the final square is in row i and column j, given $0 \le i, j < 8$; (c) the final square is a knight's move away from the starting square.

999. [*M00*] this is a temporary exercise (for dummies)

complement
central symmetry
fully symmetric
Grabarchuk graph
Euclidean distance
unit-distance graph, 3D
bipartite
regular
automorphisms
Hamiltonian decomposition
r-regular graph
Hamilton decomposable
K_{2n+1}
torus grid
distributive law
regular digraph
symmetric digraph
5-cube
Wolf
chessboard

[This is a page-filler so that the answers will begin on a left-hand page.]

After [this] way of Solving Questions, a man may steale a Nappe, and fall to worke again afresh where he left off.

— JOHN AUBREY, *An Idea of Education of Young Gentlemen* (c. 1684)

AUBREY
Hamilton paths
Rote
inscribed cubes and tetrahedra
Brückner
historical notes
Kowalewski
Du Val
threefold symmetries
stereographic projection
sphere

SECTION 7.2.2.4

1. Established conventions promote communication, so they outweigh convenience. [And we could save even more syllables by saying "HC" and "HP." Many authors now save two syllables by saying just "Hamilton cycles" and "Hamilton paths."]

2. True (except in the trivial case where G has a single vertex). In fact, the number of Hamiltonian paths in G is the number of Hamiltonian cycles in G'; the number of Hamiltonian paths between u and $v \neq u$ in G is the number of Hamiltonian cycles in G' that include the edges by which u and v are joined to the new vertex.

3. The 12 vertices of Fig. 121(a) are named ij for $i \neq j$ and $1 \leq i, j \leq 4$. We define $12 — 23$, $12 — 24$, and $12 — 43$. If $ij — i'j'$ then $ji — j'i'$ and $(i\alpha)(j\alpha) — (i'\alpha)(j'\alpha)$, where α is any even permutation of $\{1, 2, 3, 4\}$. These rules define all of the edges.

The 20 vertices of Fig. 121(b) are named ij for $i \neq j$ and $1 \leq i, j \leq 5$. We define $12 — 35$, $12 — 43$, and $13 — 24$. If $ij — i'j'$ then $ji — j'i'$ and $(i\sigma)(j\sigma) — (i'\sigma)(j'\sigma)$, where σ is the permutation (12345). These rules define all of the edges.

We can get from the dodecahedron graph of Fig. 121(b) to the icosahedron graph of Fig. 121(a) by first removing the eight vertices whose label includes '5'. Each of the twelve vertices that remain can then be joined to its five nearest neighbors, which were at distance ≤ 2 in the original graph. (This attractive labeling scheme for the icosahedron was suggested by G. Rote in 2025.)

Delightful patterns are abundant here! For example, if $1 \leq l \leq 5$, exactly eight of the dodecahedron's vertices have a label containing l; and those eight vertices actually are the corners of an inscribed cube. In fact, the four vertices with left coordinate i are equidistant, and they're the corners of the ith "inscribed left tetrahedron"; similarly, the four with right endpoint j define the jth inscribed *right* tetrahedron. Thus vertex ij is the intersection of two inscribed tetrahedra.

[See M. Brückner, *Vielecke und Vielflache: Theorie und Geschichte* (Leipzig, 1900), §105; A. Kowalewski, *Sitz. Akad. Wiss. Wien* (IIa), **126** (1917), 67–90, 963–1007; P. Du Val, *Homographies Quaternions and Rotations* (Oxford, 1964), §2.11. See also exercise 7.2.2.1–136 for a 3D-geometry-based representation scheme.]

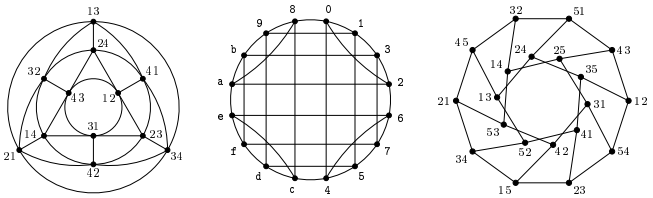
4. It remains unchanged after 180° rotation about any of the following lines: (i) from $\frac{13+45}{2}$ to $\frac{31+54}{2}$; (ii) from $\frac{14+53}{2}$ to $\frac{41+35}{2}$; (iii) from $\frac{15+34}{2}$ to $\frac{51+43}{2}$.

There also are remarkable threefold symmetries of a different kind: Color the edges of the cycle alternately red and green; color the other edges blue. Then a 120° rotation about any of the lines from 21 to 12, 23 to 32, 24 to 42, or 25 to 52 will permute the colors cyclically(!). That will yield green-blue and blue-red cycles (see exercise 24).

5. We can redraw the edges $\{12 — 35,\ 51 — 24,\ 45 — 13,\ 21 — 34,\ 53 — 42\}$ so that they lie outside the circle. Alternatively, via stereographic projection we can regard a planar graph as a graph embeddable on the surface of a sphere. In this sense Figure 122(c) shows the Hamiltonian cycle on the equator; we can imagine that half of the other edges lie in the hemisphere below. [A cubic planar graph with a Hamiltonian cycle can always be drawn as a circle, with some of the unused $n/2$ noncrossing edges inside and the others entirely outside. See exercise 103 for more about planarity.]

complex plane
golden ratio ϕ
Gerbracht

6, 7, 8.



To determine the distances and angles needed for the third drawing above, assuming that vertex '12' is point 1 in the complex plane and that '35' is point $re^{i\theta}$, solve $|1-re^{i\theta}|^2 = |1-e^{i\pi/5}|^2 = |re^{i\theta} - re^{i(\theta-2\pi/5)}|^2$. The answer (see exercise 1.2.8–19) is $r = 5^{-1/4}\phi^{-1/2} \approx .5257$, $\theta = \frac{\pi}{4} - \frac{1}{2}\arctan\frac{1}{2} \approx .5536$. [E. H.-A. Gerbracht, *Kolloquium über Kombinatorik*, Universität Magdeburg (15 November 2008).]

9. (a) We can assume by symmetry that the cycle begins at vertex 12, having just come from 35. Case (i) takes us to 54, 23, 41, 35; oops! In case (ii) it's 54, 31, 25, 14; now 43 is stranded. In case (iii) the moves to 54, 31, 42, 53, 21, 45, 13 force the cyclic path 51 — 43 — 25 — 14 — 32 — 51. The opening moves 54, 23, 15, 34 in cases (iv)–(vi) force the ending to be ..., 51, 24, 13, 52, 41, 35; so those cases are ruled out.

(b) The only remaining possibilities are $(\mathrm{LLLRRRLRLR})^2$ and $(\mathrm{RRRLLLRLRL})^2$.

10. All but 23, 24, 25, 31, 41, and 51. (There are 20 Hamiltonian paths from 12 to 35, in spite of the "uniqueness" of exercise 9. There are only six such paths from 12 to 21.)

11. (Notice that GP$(n,2)$ is a multigraph when $n \in \{1,2,4\}$.) The answer, X_n, is (0, 0, 3, 8, 0) when $n = (1, 2, 3, 4, 5)$, since parallel edges are distinct. Assume that $n \ge 6$.

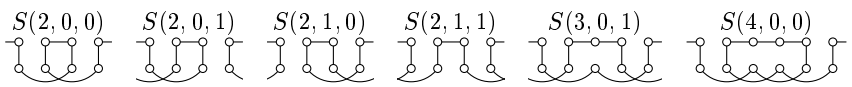
For $0 \le j < n$, let a_j = [edge j — $j{+}1$ not taken], b_j = [edge j' — $(j{+}2)'$ not taken], s_j = [edge j — j' taken]; all arithmetic here is mod n. Then $s_j = a_{j-1} + a_j = b_{j-2} + b_j$; in particular we can't have $a_{j-1} = a_j = 1$ or $b_{j-2} = b_j = 1$. We have $a_0 + \cdots + a_{n-1} = b_0 + \cdots + b_{n-1} = t$ and $s_0 + \cdots + s_{n-1} = 2t$ for some $t > 0$. It's easy to see that $t = 1$ works only in the case $n = 3$ already considered. Hence $t \ge 2$.

Assume that $a_0 = a_p = 1$ and $a_j = 0$ for $0 < j < p$. Then $s_0 = s_1 = s_p = s_{p+1} = 1$ and $s_j = 0$ for $1 < j < p$. If $p \ge 4$ we also have $b_j = 0$ for $0 \le j < p$; $b_p = b_{p+1} = 1$.

If p is odd, there's a contradiction (short cycle) when $p \ge 5$. Hence $p = 3$; and in this case b_1 must be 1 to avoid a short cycle, so $b_0 = b_2 = b_3 = 0$ and $b_4 = 1$.

The remaining case, $p = 2$, has four subcases depending on b_0 and b_1.

Every Hamiltonian cycle is a cyclic sequence $S(p_1, x_1, y_1) \ldots S(p_t, x_t, y_t)$, where each $S(p, x, y)$ is a subgraph with $2p$ vertices, having a and b sequences as just described:



The a sequence of $S(p, x, y)$ is 10^{p-1}; the b sequence begins with xy. More precisely, the b sequence when $p = 2$ is xy; when $p = 3$ it's 010; and when $p > 3$ is even, it's 0^p. And the b sequence immediately following must begin $\bar{x}\bar{y}$ when p is even, otherwise 01.

Hence consecutive subgraphs in the cycle are severely restricted; they must be

$$S(e,0,0)S(2,1,1),\ S(2,0,1)S(2,1,0),\ S(2,1,0)S(2,0,1),$$
$$S(2,1,0)S(3,0,1),\ S(2,1,1)S(e,0,0),\ \text{or}\ S(3,0,1)S(2,0,1),$$

where e is even. Furthermore $S(2,0,1)$ and $S(2,1,0)$ can't be used more than once, or the edges will be disconnected. The only solutions are cyclic rotations of $S(3,0,1)^{n/3}$, n odd; $S(2,1,0)S(3,0,1)^{(n-4)/3}S(2,0,1)$; $S(e_1,0,0)S(2,1,1)\ldots S(e_{t-1},0,0)S(2,1,1)$ with $e_1 + 2 + \cdots + e_{t-1} + 2 = n$. Hence $X_n = n$ when $n \bmod 6 = 1$; $X_n = 3$ when

$n \bmod 6 = 3$; $X_n = 0$ when $n \bmod 6 = 5$. And when n is even we find via generating functions that $X_n = 2L_{n/2} - 2 + n[n \bmod 6 = 4]$, where $L_q = F_{q+1} + F_{q-1}$ is the qth Lucas number. [A. J. Schwenk, *J. Combinatorial Theory* **B47** (1989), 53–59.]

generating functions
Lucas number
Fibonacci numbers
Schwenk

14. (a) In fact, let H be *any* induced subgraph whose vertices all have degree 3 except for exactly two vertices of degree 2. The other edges from those two must be true in every win, or we'd have a cycle entirely within H.

(b) The connecting edges are 1, and so are many of the internal edges. Thus internal cycles will appear when $x + y + z$ is 0, 2, or 3. (But if $x + y + z = 1$ we easily have a path through all the internal vertices.)

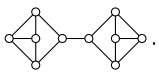
(c) The long horizontal edges must be true, because consecutive true vertical edges would yield a short cycle. Hence the edge values at the left and right are x, $\bar{x}$, x, $\bar{x}$, x and y, $\bar{y}$, y, $\bar{y}$, y. If $x = y$ we'd have a 4-cycle or two 8-cycles.

(d) Hook C_m to C_1. Then insert XOR gadgets to ensure that all appearances of the same variable have consistent values. [This construction can be extended so that G is not only cubic but planar, and triconnected, with at least five sides on every face. See M. R. Garey, D. S. Johnson, and R. E. Tarjan, *SICOMP* **5** (1976), 704–714.]

planar
triconnected
Garey
Johnson
Tarjan

16. (1, 2, 5, 19) connected cubic graphs on (4, 6, 8, 10) vertices are essentially distinct (not isomorphic); we'll study how to generate them in Section 7.2.3. They all are Hamiltonian except for two of order 10. One of the latter is the famous Petersen graph (Fig. 2(e) near the beginning of Chapter 7), which also is nonplanar.

isomorphic
Petersen graph

The other "smallest" non-Hamiltonian example is actually planar:  . [Arun Giridhar verified in 2015 that a 16-vertex variant of this graph, consisting of three 5-vertex diamonds joined to a central vertex, is (uniquely) the smallest cubic graph that has no Hamiltonian *path*.]

Giridhar
Hamiltonian *path*

18. False. For example, consider $0 — 1 — 2 — 3 — 4 — 5 — 0$, $0 — 2 — 4 — 0$.

20. (a) The condition holds when a_k is the number of k-sided faces *inside* the n-cycle. For it's certainly true when there's just one such face ($a_k = [k = n]$). And if a new chord is added to the graph, breaking an inner p-face into a q-face and an r-face where $q + r = p + 2$, $\sum_k (k-2)a_k$ changes by $(q-2) + (r-2) - (p-2) = 0$.

[The number $a'_k = \alpha_k - a_k$ of k-faces *outside* the cycle is also a solution to Kirkman's conditions. Indeed, we always have $\frac{1}{2}\sum_k (k-2)\alpha_k = \frac{1}{2}\sum_k k\alpha_k - \sum_k \alpha_k = \langle\text{edges}\rangle - \langle\text{faces}\rangle = \langle\text{vertices}\rangle - 2$ in a connected planar graph.]

(b) We can assume that the missing vertex is outside the cycle; and $3a_5$ can't equal $19 - 2$. (The dodecahedron does have cycles of lengths $\{5, 8, 9, 10, \ldots, 17, 18\}$.)

(c) We can assume that neither cycle is inside the other. A cycle that contains exactly a pentagons has length $3a + 2$; and $(3a + 2) + (3b + 2)$ can't equal 20.

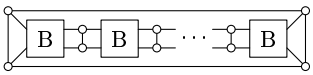
(d) Let G' be G without the edge $a — b$, and with two additional vertices of degree 2: one inserted between a and f, another between b and c. Any Hamiltonian cycle in G that omits $a — b$ corresponds to a Hamiltonian cycle in G'. But G' isn't Hamiltonian, because $\alpha'_k = [k = 4] + 7[k = 5] + [k = 11]$ and $2a'_4 + 3a'_5 + 9a'_{11} \neq 18 - 2$.

(e) Graph (i) has $\alpha_k = [k = 4] + 20[k = 5] + 2[k = 11]$; graph (ii) has $\alpha_k = [k = 4] + 18[k = 5] + 4[k = 8]$. So both of them fail Kirkman's test (a). Graph (iii), with $\alpha_k = 18[k = 5] + 3[k = 6] + 3[k = 8]$, passes the test only if $a_6 = 0$ or 3. But the three 6-edged faces can't all be inside or outside the cycle, since they share a common point.

[*Historical notes:* Kirkman actually studied full-length cycles in convex polyhedra slightly before Hamilton began to toy with such ideas. [See *Philosophical Transactions* **146** (1856), 413–418; *Philosophical Magazine* (4) **12** (1856), 446.] Graphs (ii) and (iii) in part (e) are due to È. Ya. Grinberg, *Latviĭskiĭ Matematicheskiĭ Ezhegodnik* **4** (1968), 51–58, who rediscovered Kirkman's long-forgotten criterion. Graph (i) was found as part of an exhaustive computer search by G. B. Faulkner and D. H. Younger, *Discrete Math.* **7** (1974), 67–74, who also established that Grinberg's (iii) is the unique smallest cubic planar graph that is *cyclically 5-connected*: It cannot be broken into two components each containing a cycle unless at least five edges are removed. (Graph (ii) clearly has four automorphisms; and graph (iii), obtained by adding a single edge, actually has six, although that isn't obvious from the diagram. If we add *another* edge at the right, in the mirror-image position, we get a 46-vertex graph with 36 Hamiltonian cycles. Of course each of those cycles uses both of the edges that were added to (ii).)]

Historical notes
Hamilton
Grinberg
Faulkner
Younger
cyclically 5-connected
automorphisms
gadget
Kirkman
Faulkner
Younger
Grinberg
3-cube
Grinberg
Historical notes
r-regular graph
"Star of David" graph

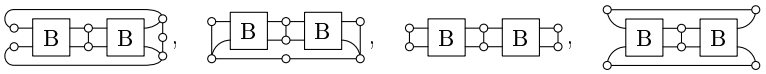
21. Among many possibilities, the simplest are perhaps the $(20n+2)$-vertex graphs

FY_n = 

made from n copies of an 18-vertex gadget

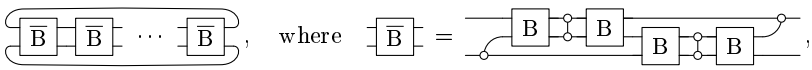
, where graph (i) in exercise 20(e) is FY_2 and

is illustrated there. In general, FY_n has $\alpha_k = [k=4] + 10n[k=5] + 2[k=5n+1]$; so it fails Kirkman's test whenever $n \bmod 3 = 2$. Further analysis, based on the fact that each of the four graphs



also fails Kirkman's test, shows that FY_n is actually non-Hamiltonian for *all* $n > 1$.

(Faulkner and Younger went on to show that the $78m$-vertex graphs



are not only non-Hamiltonian, they can't be covered by fewer than m disjoint cycles.)

Grinberg's paper of 1968 described non-Hamiltonian cubic planar graphs having $(14 \cdot 4^s - 10 \cdot 3^s)(3t-1)$ vertices, for any $s, t > 0$. His graphs are noticeably harder than FY_n for a computer to analyze; even the case $(s,t) = (2,1)$ is quite a challenge.

24. (a) $K_4 \oplus K_4$ is disconnected (and K_4 is perfectly Hamiltonian). The others are at least Hamiltonian, and we can number the vertices so that $0 — 1 — \cdots — 7 — 0$. There are five nonisomorphic possibilities: *Case 1,* $0 — 2$, $1 — 3$, $4 — 6$, $5 — 7$*:* 16 auts, 4H. [Translation: 16 automorphisms and 4 Hamiltonian cycles.] *Case 2,* $0 — 2$, $1 — 5$, $3 — 6$, $4 — 7$*:* 12 auts, 6H, perfect (two sets of three). *Case 3,* $0 — 2$, $1 — 5$, $3 — 7$, $4 — 6$*:* 4 auts, 3H, planar, perfect. *Case 4,* $0 — 3$, $1 — 6$, $2 — 5$, $4 — 7$*:* 48 auts, 6H, planar [the 3-cube]. *Case 5,* $0 — 4$, $1 — 5$, $2 — 6$, $3 — 7$*:* 16 auts, 5H.

(b) Let a_k be the number of k-faces inside none of the three Hamiltonian cycles; let b_k, c_k, d_k be the number that are inside cycles $\{1,2\}$, $\{1,3\}$, $\{2,3\}$, respectively. Then Kirkman's criterion for cycle 1 is satisfied by $b_k + c_k$ and $a_k + d_k$, the number of faces respectively inside or outside. Similarly, it's satisfied for cycle 2 by $b_k + d_k$ and $a_k + c_k$; for cycle 3 by $c_k + d_k$ and $a_k + b_k$. Let $A = \sum_k (k-2)a_k$, …, $D = \sum_k (k-2)d_k$; we've shown that $A+B = A+C = A+D = B+C = n-2$. [See Grinberg's paper in answer 20.]

[Similarly, an r-regular graph is perfectly Hamiltonian if its edges can be r-colored in such a way that all $\binom{r}{2}$ pairs of colors yield Hamiltonian cycles. (The 4-regular 6-vertex "Star of David" graph $L(K_4)$ is a good example; so is the 5-regular K_6.) Such

graphs are also called P1F, "perfectly 1-factorable," because two 1-factors — also known as perfect matchings — are called perfect if they yield a Hamiltonian cycle, and because an r-regular graph with $\chi(L(G)) = r$ is called 1-factorable. The pioneering explorations of A. Kotzig (see *Theory of Graphs and its Applications*, ed. by M. Fiedler (1964), 63–82) have led to a large literature with many provocative problems still unsolved (see A. Kotzig and J. Labelle, *Annales des Sciences Math. du Québec* **3** (1979), 95–106); for an excellent survey see A. Rosa, *Mathematica Slovaca* **69** (2019), 479–496. Answer 124 below, Case 2, proves incidentally that cubic Halin graphs are perfectly Hamiltonian. So are the generalized Petersen graphs GP$(6q + 3, 2)$.]

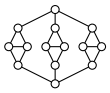
P1F
1-factors
perfect matchings
Kotzig
Fiedler
Labelle
strongly H graphs, see perfectly H graphs
Rosa
Halin graphs
generalized Petersen graphs
Historical notes

27. (a) This follows directly from exercise 20(d). (In general, we get a "forcing" gadget from *any* graph that has a "forced" edge, by removing any vertex of that edge.)

(b) Insert a degree-2 vertex into each of those spokes and apply 20(a).

(c) Two nonconsecutive spokes are forced to be in any Hamiltonian cycle.

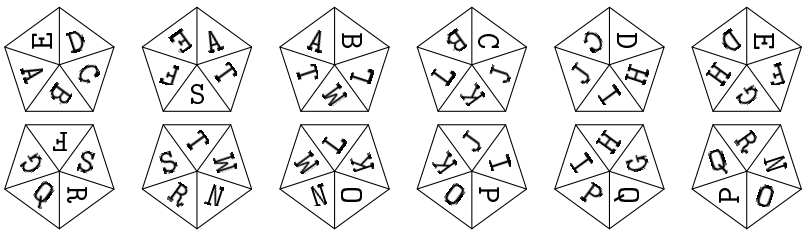
(d) No. One of the five 4-faced regions touches the unique 9-faced region. Its other neighbors have respectively $\{5,8,8\}$, $\{5,7,7\}$, $\{5,7,8\}$, $\{7,8,8\}$, $\{7,7,7\}$, $\{7,7,8\}$ faces.



Historical notes: A cubic graph that can be disconnected by removing one edge is clearly non-Hamiltonian. Many cyclically *2-connected* planar cubic graphs, such as the example shown, also have no Hamiltonian cycle. However, P. G. Tait investigated numerous cyclically *3-connected* planar cubic graphs — the "true" polyhedra — and found Hamiltonian cycles easily. So he conjectured that such cycles always exist, and he pointed out that the famous "Four Color Theorem" would then follow. (See §15 and §16 of his paper cited in answer 35.) Tait's conjecture was believed for many years, until W. T. Tutte [*J. London Math. Soc.* **21** (1946), 98–101] found a 46-vertex counterexample by putting together three Tutte gadgets. The smaller graphs in (c) were found independently in 1964 by D. Barnette, J. Bosák, and J. Lederberg; those graphs are the *only* counterexamples with fewer than 40 vertices [see D. A. Holton and B. D. McKay, *J. Combinatorial Theory* **B45** (1988), 305–319]. Every counterexample has a face with more than 6 vertices [F. Kardoš, *SIAM J. Discrete Math.* **34** (2020), 62–100]; in particular, all "fullerene graphs" are Hamiltonian.

cyclically *2-connected*
bridge, wheatstone
Tait
cyclically *3-connected*
Four Color Theorem
Tutte
Barnette
Bosák
Lederberg
Holton
McKay
Kardoš
fullerene graphs

30. We can use the Ls and Rs of answer 8 as a guide:



(The author cherishes a 3D-printed object like this, received as a surprise gift in 2016.)

author
3D-printed

33. $nH + h$ — one for every Hamiltonian path in G (cyclic or not). (Thus an algorithm that finds all Hamiltonian cycles can readily be adapted to find all Hamiltonian paths.)

all Hamiltonian paths

35. (a) The tour lines divide the plane into regions. Every such region can be assigned a rank, representing its distance from the outside. (More precisely, the rank is the minimum number of tour lines crossed by any path in the plane that goes from a point in the region to a point outside the chessboard, without passing through any of the tour's intersection points.) Then, as you walk along the tour, make your thread go on top at an intersection if and only if the region on your left has odd rank. [See P. G. Tait, *London, Edinburgh, and Dublin Philosophical Magazine* **17** (1884), 30–46, §19.]

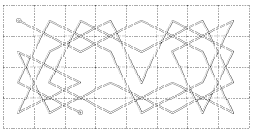
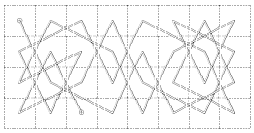
rank
Tait

(b) One endpoint is in the outside region, but the other is in a region of rank 4. Any artificial path that connects the two, and crosses k tour lines, will lead to a drawing with k exceptions when the artificial path is removed. Conversely, the exceptional tour segments in any drawing can be crossed by an artificial connection path together with zero or more artificial cycles; so there must be at least 4 exceptions.

(There's no problem in (2), because each endpoint lies in the outer region.)

Flye Sainte-Marie
open
Rudraṭa
poetic license
Murray
silver general
shōgi
Japanese chess

36. In (i), $\sigma_{22} \leftrightarrow \sigma_{24}$ and $\sigma_{25} \leftrightarrow \sigma_{27}$. In (ii), 'lots' matches 'lost':

(i)  ; (ii)  .

37. [C. Flye Sainte-Marie, *Bull. Soc. Math. de France* **7** (1877), 144–150.] Starting at cells (1, 2, 3, 4) of row 1 we obtain respectively (7630, 2740, 2066, 3108) tours. Starting at cells (1, 2, 3, 4) of row 2 we obtain none. Thus there are exactly $4 \cdot (7630 + 2740 + 2066 + 3108) = 62{,}176$ tours, all of which are open because they begin and end in the top or bottom row. (Among all such tours, 1904 cannot be represented by a single Rudraṭa-style sloka because all 32 syllables of such a sloka would have to be identical! Only the example in answer 36(ii), and its reversal after 180° rotation, are representable by a sloka that has 12 distinct syllables.)

38. One knight jumps like three rookwise steps.
Past sore too mean; so, just for free,
Hops here, turns there, flies each goose now.
Can't place last word? Won't find the sea.
One, two, three, four! See each word here:
Jumps so wise now find their place passed.
Terns can't soar, like flies the free rook;
Goose steps just won't mean knight hops last.

40. Meet me, you fool; trip some word up;
Eat, see if autumn is a mess.
To forgo this ordeal, I cheat:
Won three games like dice, card-trick, chess.
One, two, three, four! Games go like this:
Dice or card deal, tricky chess cheat,
Mess up; a word is some dumb trip.
Awful if you see me eat meat.

[Sloka 16 in Rudraṭa's *Kāvyālaṅkāra* can be interpreted as two poorly joined quarter-tours of an elephant. See Murray's *History of Chess*, pages 54 and 55.]

(A "silver general" in shōgi (Japanese chess) has the same moves as an elephant.)

41. (a) There are $(m-1)n$ "trunk" arcs from (i, j) to $(i-1, j)$, plus $4(m-1)(n-1)$ "leg" arcs from (i, j) to $(i \pm 1, j \pm 1)$; total $(m-1)(5n-4)$.

(b) Yes: If and only if $m = 2$ and n is even. (Use just two trunk moves.)

(c) In fact, the solution in Fig. A–18(a) is the *only* such Hamiltonian path on E_{48}.

(d) If not, every vertex of row 1 is of type A ( ) or B ( ) or C ( ) or D () within the path. There's a B at the left and an A at the right. The adjacent pairs AD, BD, CA, CB are not permitted, nor are the near-adjacent pairs B∘A, B∘C, C∘D,

D∘A, D∘C (with one vertex intervening). Furthermore the substrings B(CD)*A = {BA, BCDA, BCDCDA, ...} are forbidden, because these are closed cycles and $m > 2$.

But no string of A's, B's, C's, and D's obeys all of those restrictions.

(e) The same proof works, with types A (↘↕), B (↕↙), C (↘↗), and D (↖↙).

(f) The vertices of (d) are joined by one of type E (↙) or F (↘). The leftmost is either B or F; the rightmost is either A or E. Cases B∘E, D∘E, F∘A, F∘C are excluded, in addition to those of (d). Exactly n [n even] such sequences are possible, having the forms F(CD)*A, B(CD)*ED(CD)*A, B(CD)*CF(CD)*A, or B(CD)*E.

Each of these has exactly one unsaturated vertex in row 2. Thus there are one or two possible moves to row 3, and we've effectively reduced m to $m-2$.

(g) Now the vertices of (e) are joined by one of type E (↖) or F (↗) or G (↕). The leftmost is either B, F, or G; the rightmost is A, E, or G. The new forbidden substrings are AE, BE, CG, FA, FB, GD, C∘E, F∘D. Six species of solutions exist, namely GA*(CD)*A, B(CD)*B*G, B*FD(CD)*A, BCD(CD)*B*FD(CD)*A, B(CD)*CEA*, and B(CD)*CEA*(CD)*CDA. The solutions containing A* or B* work when n is odd.

Again we reduce m to $m-2$ and continue. (By induction, m must be even.)

(h) Let $A^{(m)}$ be the $n \times n$ matrix where $a_{ij}^{(m)}$ is the number of Hamiltonian paths from $(1,i)$ to (m,j); let $B^{(m)}$ be analogous, where $b_{ij}^{(m)}$ counts paths from (m,i) to $(1,j)$. (These matrices are symmetric about both diagonals, because the left-right and top-bottom reflections of any elephant path are elephant paths, possibly reversed.) We have

$$a_{ij}^{(2)} = ([i=j=1] + [i=j=n] + [|i-j|=1 \text{ and } \max(i,j) \text{ odd}])[n \text{ even}],$$

$$b_{ij}^{(2)} = \begin{cases} [i \text{ odd}][j \text{ even}][i<j] + [j \text{ odd}][i \text{ even}][j<i], & \text{if } n \text{ is even,} \\ [i \text{ odd}][j \text{ odd}]\max([i=1],[i=n],[i\neq j]), & \text{if } n \text{ is odd,} \end{cases}$$

by (f) and (g). Moreover, by considering moves between two-row subgraphs,

$$A^{(m+m')} = A^{(m)}XA^{(m')} \text{ and } B^{(m+m')} = B^{(m)}(X+I)B^{(m')}, \text{ where } x_{ij} = [|i-j|=1].$$

For example, there are $\sum A^{(4)} + \sum B^{(4)} = 14 + 120 = 134$ tours on a 4×8 board.

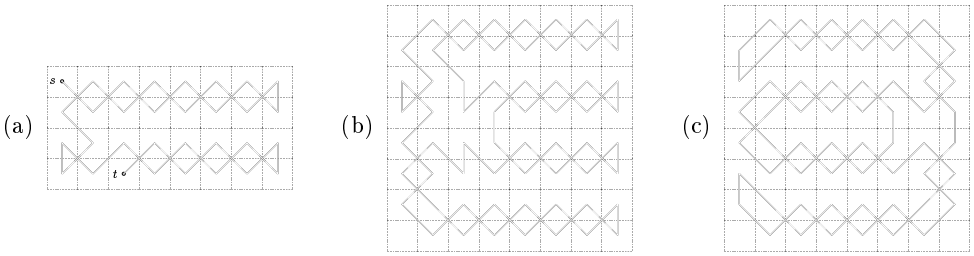


Fig. A–18. Noteworthy paths and cycles for elephants.

directed graphs
ZDD
oriented cycles
generating function

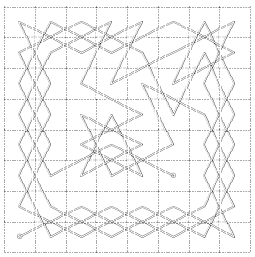
42. The technology of exercise 7.1.4–226 can be extended to directed graphs in a straightforward way. It constructs a ZDD of about 1.3 meganodes for all oriented cycles in the 8×8 elephant digraph, and shows that there are exactly 277,906,978,347,470 of them. The generating function by cycle length is $98z^2 + 205z^4 + 698z^6 + 3853z^8 + \cdots + 50128559z^{60} + 6544z^{62}$. If we say that trunk moves have weight 2 while other moves have weight 3, we find (by computing maximum-weight cycles) that exactly four of the 6544 62-cycles have only eight trunk moves. All four of these solutions are equivalent under reflection to Fig. A–18(b). (Figure A–18(c) shows an interesting symmetrical 60-cycle that omits the corners and has just *four* trunk moves.)

44. We can use Rudraṭa's half-tour twice:

Bah dee boo dai hao fuh hoe fay, bee doo bai fao huh foe hay dah?
Fee hah day boe foo hai dao buh, fai bao duh hoo doe bay fah hee.
Lah mee loo mai sao nuh soe nay, lee moo lai nao suh noe say mah?
Nee sah may loe noo sai mao luh, nai lao muh soo moe lay nah see!

Rudraṭa
open tour
closed tours
Ibn Manī‘
Murthy
Rudraṭa
Ratnākara
Deśika
de Jaenisch
von Warnsdorf

(This is an open tour. See exercise 199 for *closed* tours that rhyme just as perfectly.)



46. This tour is rather like that of Ibn Manī‘ in (1):

[G. S. S. Murthy, in *Resonance* **25** (2020), 1095–1116, comments further on the work of Someshvara, and gives excellent translations of the Sanskrit verses by Rudraṭa, Ratnākara, and Vedānta Deśika.]

50. In fact, the text's "merry chase" need not end at cell 22; by swapping **23** ↔ **25** we get a tour that ends at 02. The other seven choices of **9** and **10** can lead, similarly, to either 22 or one of $\{02, 11, 13, 20, 24, 31, 33\}$. Each case completes an open tour.

[See page 280 of de Jaenisch's book, for his analysis of 5×5 paths.]

51. Again $v_1v_2v_3v_4$ = 00 12 04 23 without loss of generality. Now $t_1 = 44$ forces $v_5 = 31$, and there's a tie for v_6. If $v_6 = 43$, the path continues $v_7 \ldots v_{15}$ = 24 03 11 30 42 34 13 01 20; and v_{16} is 32 or 41. The former case forces $v_{17} = 40$, hence "shutting out" 44; it leads to four paths, each ending at v_{24}. But the latter case leads to three paths, two of which end with $v_{25} = 44$ (yea) and one that ends with $v_{21} = 44$ (boo).

On the other hand if $v_6 = 10$ we get $v_7 \ldots v_{13}$ = 02 14 33 41 20 01 13 and then $v_{14}v_{15}v_{16}$ = 21 40 32 or 32 40 21. Either case shuts 44 out. Ten continuations are possible, each of which involves 24 vertices — all but cell 44 (close but no cigar).

(The randomized algorithm of exercise 53 will yield a Hamiltonian path with probability $\frac{3}{16}$. If we set $\{t_1, t_2, t_3\} = \{23, 32, 44\}$ and $r = 3$, this probability rises to $\frac{3}{8}$.)

52. The algorithm acts just as if a double-target vertex t has been entirely removed from the graph, because `DEG(`t`)` will never be less than $2n$ in step W5.

53. W5′. [Is `DEG(`u`)` smallest?] If $t < \theta$, set $\theta \leftarrow t$, $v \leftarrow u$, $q \leftarrow 1$. Otherwise, if $t = \theta$, set $q \leftarrow q + 1$, then set $v \leftarrow u$ with probability $1/q$.

55. Ibn Manī‘ broke Warnsdorf's rule first when choosing $v_{14} = 41$ instead of 06. His choices for v_{26}, v_{27}, v_{38}, v_{39}, v_{45}, v_{48}, and v_{54} also broke the rule. But altogether, his "hug the edge" strategy followed it $\frac{55}{63} \approx 87\%$ of the time, so he probably had some of the same intuition that von Warnsdorf acquired later. Similarly, Someshvara deviated only eight times. But al-‘Adlī broke the rule 15 times, clearly thinking other thoughts.

56. If there's no remaining exit from u, there's no remaining entrance to u. Therefore Algorithm W will not find a Hamiltonian path unless it moves to u. We might as well do that, if our goal is simply to find a Hamiltonian path. But maybe we really want to find as long a path as possible, via Warnsdorf-like rules; then we can do better.

backtrack algorithm
convex triangular grid graphs
SGB
simplex
dual
triangular faces
convex polygon
Gray binary code
delta sequence
ruler function ρ

Vertex u is a dead end if and only if `DEG(`u`)` is 0 or n. Suppose v_k has just two neighbors, u and u', where `DEG(`u`)` $= 0$ and `DEG(`u'`)` $= n$. We presumably should choose $v_{k+1} = u'$ in such a case, because u' is one of the designated targets.

Thus we can improve the algorithm by changing '$t < \theta$' in step W5 to '$f(t) < f(\theta)$', where $f(0) = 2n + 1$, $f(n) = 2n$, $f(t) = t + 2$ for $t \geq 2n$, and $f(t) = t$ otherwise.

57. This is a simple backtrack algorithm, following the outline of Algorithm 7.2.2B. Leaves of the search tree correspond to the paths of Algorithm W. The probability of each node is the probability of its parent, divided by the family size.

59. Build 64 anti-Warnsdorf trees, as in exercise 57. There are 8 paths $v_1 v_2 \dots v_{20}$ of length 19 (equivalent to each other via rotation/reflection), each occurring with probability $2^{-9}3^{-6} \approx .0000027$; they actually turn out to be cycles of length 20. At the other extreme, there are 24 anti-Warnsdorf paths as long as 53, of which 8 have probability $p = 2^{-20}3^{-5} \approx .0000000039$ and 16 have probability $3p$; representatives are shown below. The mean path length is ≈ 34.8, and the standard deviation is ≈ 5.3.

By contrast, Algorithm W will find a path that's longer than *every* anti-Warnsdorf path, more than 99.8% of the time. Its worst case has length 39; there are eight such paths, equivalent to the one shown, each obtained with probability 2^{-21}. (And if we use the modification of answer 56, the worst case length rises to 46; there are 40 such paths, of which the one shown is the most likely — its probability is $2^{-18}3^{-4}$. The total probability of all 40 cases for length 46 is $169 \cdot 2^{-23}3^{-3} \approx .00000075$.)

	20						
		18	1				
19		11	8	5	2		
	17	14	3	10	7		
	12	9	6	15	4		
		16	13				

	32	1	54	29		43	
	53	30	33	44	41	28	
31	34	15	2	7	10	45	42
52	21	8	5	12	3	40	27
35	16	23	14	9	6	11	46
22	51	20	17	4	13	26	39
	36	49	24	1	38	47	
50		18	37	48	25		

	32	1	42	29		53	
	43	30	33	52	41	28	
31	34	15	2	7	10	51	54
44	21	8	5	12	3	40	27
35	16	23	14	9	6	11	50
22	45	20	17	4	13	26	39
	36	47	24	1	38	49	
46		18	37	48	25		

		45	32	49			
46	31	48	1	20	33	50	37
17	44	19	10	5	36	21	34
30	47	16	7	2	9	38	51
43	18	27	4	11	6	35	22
26	29	12	15	8	3	52	39
	42	25	28	13	40	23	54
		14	41	24	53		

2	19	4	21			14	
5	22	1		15			
18	3	20					13
23	6		16				
32	17	30				12	
7	24	33		29			38
34	31	26	9	36	39	28	11
25	8	35	40	27	10	37	

3	36	5				23	
6	43	2		24			
37	4	35					22
44	7	42	1		25		
41	38	45	34			21	26
8	31	10	39	46	29	18	15
11	40	33	30	13	16	27	20
32	9	12	47	28	19	14	17

60. A tree constructed as in exercise 57 has two different forms, depending on whether s is a variable of type $\{a, d\}$ or $\{b, c\}$, because R_6 doesn't have as much symmetry as R_5.

(a) $\approx (.86, .74, .65, .57, .50)$ when $s = a_0$; $\approx (.90, .81, .71, .63, .56)$ when $s = b_0$. [For $q = 100$ these probabilities drop to about .000044 and .000050.]

(b) $\approx (.06, .05, .05, .04, .04)$ when $s = a_0$; $\approx (.07, .05, .05, .04, .04)$ when $s = b_0$.

(c) $\approx \big((.33, .29, .29, .24, .22), (.26, .26, .24, .19, .17)\big)$ when $s = a_0$, $t_1 = (b_0, a_1)$; and $\approx \big((.27, .23, .23, .18, .17), (.28, .21, .18, .15, .13)\big)$ when $s = b_0$, $t_1 = (a_0, c_0)$.

62. There are two choices for v_2; and the next moves must hug the edge. Thus the graph reduces either to $P_{m-1} \,\square\, P_n$ or $P_m \,\square\, P_{n-1}$, and we succeed by induction.

[It's necessary to require $r = 0$; for example, the algorithm can fail when $m = n = 4$, $s = (0, 0)$, $t_1 = (0, 3)$. The start vertex must also be a corner; consider $m = 4$, $n = 5$, $s = (0, 2)$. A similar proof shows that Algorithm W never fails on the convex triangular grid graphs produced by SGB's generator $simplex(n, n_1, n_2, n_3, 0, 0, 0)$, when starting at a corner. It also succeeds on the *dual* of that graph, provided that $n_1, n_2, n_3 < n$ and that a suitable starting vertex is specified; this is the graph whose vertices are the triangular faces inside the polygon, all of which have degree 2 or 3.]

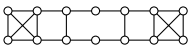
63. Notice first that the standard Gray binary code $g(0)$, $g(1)$, ..., $g(2^n - 1)$ is one of the paths that satisfy Warnsdorf's rule. For example, when $n = 3$ and $s = 000$, this path is 000, 001, 011, 010, 110, 111, 101, 100; and it has the delta sequence $\delta_0 \dots \delta_6 = 0102010$ of the ruler function ρ (see 7.2.1.1–(24)). Warnsdorf's rule for the 3-cube allows any $\delta_0 \in \{0, 1, 2\}$; if $\delta_0 = 0$, it allows $\delta_1 \in \{1, 2\}$; if $\delta_0\delta_1 = 01$, it forces $\delta_2\delta_3 = 02$; if $\delta_0\delta_1\delta_2\delta_3 = 0102$, it allows $\delta_4 \in \{0, 1\}$; and if $\delta_0 \dots \delta_4 = 01020$, it forces $\delta_5\delta_6 = 10$.

Warnsdorf's rule
next-to-rightmost 1
Warnsdorf paths
Li
author
random 32-bit weight
long hash code
3-cube
mates

In general one can prove that if $\delta_{k-1} = \rho(k)$ for $1 \le k < r$, then Warnsdorf's rule allows δ_r to be any coordinate in the interval $W_r = [\rho(r)\,..\,\rho'(r))$, where $\rho'(r) = \rho(r \,\&\, (r-1))$ is the index of the next-to-rightmost 1 in the binary representation of r. (If r is a power of 2, $\rho'(r)$ is undefined; in such cases we use n instead of $\rho'(r)$ in the formula for W_r.) For example, if $r = (1010010)_2$ and $n \ge 7$, we have $\rho(r) = 1$ and $\rho'(r) = 4$, hence $W_r = [1\,..\,4) = \{1, 2, 3\}$.

Moreover, the residual graph of unvisited vertices, when it is time to choose δ_r, is always symmetrical with respect to every coordinate in W_r. Therefore we can choose $\delta_r = \rho(r)$ without loss of generality; all other Warnsdorf paths can be mapped into the standard path by permuting coordinates. For example, the Warnsdorf paths for the 3-cube have the delta sequences 0102010, 0102101, 0201020, 0201202, 1012101, 1012010, 1210121, 1210212, 2021202, 2021020, 2120212, 2120121.

[Algorithm W always succeeds in the graph $P_3 \,\square\, P_3 \,\square\, P_3$ when $s = (0, 0, 0)$; but not always in $P_4 \,\square\, P_4 \,\square\, P_4$. It sometimes fails in $P_3 \,\square\, P_3 \,\square\, P_3 \,\square\, P_3$ when $s = (0, 0, 0, 0)$.]

65. Yes:  . (Shisheng Li has shown that this is the *smallest* example.)

70. It happens when $k = t - 1$ in the first call, and when $k = 2$ in the second call. (A little time can be saved by detecting these special cases. Similarly, '*update*$(u_1, \ldots, u_t)$' arises in step F5 when $k = j \pm 1$. Such updates weren't counted in the author's tests.)

71. Let the edges be $1 \mathrel{—} 2 \mathrel{—} \cdots \mathrel{—} n \mathrel{—} 1$ and $1 \mathrel{—} 5$, $3 \mathrel{—} (n{-}2)$, $(n{-}4) \mathrel{—} n$; consider the path $2 \mathrel{—} 1 \mathrel{—} 5 \mathrel{—} 4 \mathrel{—} 3 \mathrel{—} (n{-}2) \mathrel{—} (n{-}3) \mathrel{—} (n{-}4) \mathrel{—} n \mathrel{—} (n{-}1)$.

73. Assign a random 32-bit weight to each edge of the graph, and let each path have a "long hash code" H that's the sum of its edge weights (modulo 2^{32}). Let there be 2^b hash lists; a path with long hash H will go into list $H \bmod 2^b$. If c vertex names can be packed into an octabyte, the dictionary entry for $(v_1, \ldots, v_t)$ will occupy $B = 1 + \lceil t/c \rceil$ octabytes: one for the link and H, the others for the vertices (which are examined in detail during a search only when the long hash code is correct). Store the qth path in positions $(qB + s) \bmod M$ of a large array of octabytes, for $0 \le s < B$, where M is a large power of 2. (Overflow occurs if we try to write into position $(p_2 B + B) \bmod M$.)

75. (a) Let H be the graph whose vertices are the Hamiltonian paths of G that start with v_1, adjacent if they differ only by flipping a subpath. Since G is cubic, every vertex of H has degree 2. So H consists of cycles, and Algorithm F$^-$ constructs the cycle that begins with the given path. For example, when $G = P_2 \,\square\, P_2 \,\square\, P_2$, we can represent the vertices (000, 001, ..., 111) by (0, 1, ..., 7), and H has two cycles: `01326754` — `01326457` — `01375462` — `02645731` — `02645137` — `02673154` — `04513762` — `04513267` — `04576231` — `01326754`; `04623751` — `01573264` — `01546247` — `01546732` — `02376451` — `02315467` — `02315764` — `04675132` — `04623157` — `04623751`.

(b) Let $\alpha = v_1 \mathrel{—} v_2 \mathrel{—} \cdots \mathrel{—} v_n$ be a Hamiltonian path with $v_n \mathrel{—} v_1$. One of its two neighbors in H is $v_1 \mathrel{—} v_n \mathrel{—} \cdots \mathrel{—} v_2$, which is the reflected cycle α^R. The other neighbor will have v_2 unchanged. So the cyclic paths come in pairs.

(c) Consider maximal segments of the cycle in H whose paths begin with $v_1 \mathrel{—} v_2$. The first and last of these paths are Hamiltonian cycles, which we can regard as *mates* of each other. (For example, the mate of `01326754` with respect to `0` — `1` is `01375462`.)

(d) If α is a Hamiltonian cycle containing the edge e, its mate β is a Hamiltonian cycle containing an edge $e' \notin \alpha$. Hence γ, the mate of β with respect to e', is a third.

(e) Color the n edges of α alternately red and green; color the other $n/2$ edges blue. The blue edges of β and γ are the same; suppose there are $n/2 - x$ of them. Let β have r red and g green; hence γ has $n/2 - r$ red and $n/2 - g$ green. We have

$r+g+n/2-x=(n/2-r)+(n/2-g)+(n/2-x)=n$; hence $x=0$. Therefore no two consecutive edges of α can appear in β or γ; they must all be the same color.

[*Historical notes:* The theorem in (c) was discovered via algebraic reasoning by C. A. B. Smith, about 1940, but not published until later. See *Combinatorial Mathematics and its Applications* (Oxford conference, 1969), 259–283; W. T. Tutte, *Graph Theory As I Have Known It* (1998), 18, 48, 94. A. G. Thomason, in *Annals of Discrete Mathematics* **3** (1978), 259–268, introduced one-sided flips and used them to give an algorithmic proof of Smith's theorem.]

Historical notes
Smith
Tutte
Thomason
Cameron
Krawczyk
Stappers
generating function
Zhong

77. (a) Easily verified. (This is the *only* non-identity automorphism, when $n>1$.)

(b) Any Hamiltonian cycle containing $05-0-01$ mustn't contain $0-06$; hence $07-06-05$ but not $05-04$ or $01-07$; hence $03-04-15$, $01-02$, $08-07-06$, not $02-08$; hence $11-08$. Replacing all '$0j$' by '1' yields a Hamiltonian cycle containing $15-1-11$ in a graph isomorphic to C_{n-1}. By induction, it's α_n. Similarly, the only Hamiltonian cycles containing $06-0-01$ and $05-0-06$ are

$$\begin{aligned}\beta_n &= 0-01-07-08-02-03-16-\cdots-15-04-05-06-0;\\ \gamma_n &= 0-06-07-01-02-08-11-\cdots-16-03-04-05-0.\end{aligned}$$

(c) (11, 65, 265, 1005, 3749, 13927, 51683, 191735, 711243). [But an appropriate sequence of only $4n$ *two-sided* flips will take us from α_n to β_n, or β_n to γ_n, or γ_n to α_n.]

(d) When $n>2$, the first five flips yield $01-0-05-06-07-08-02-03-04-15-16-17-11-12-18$ followed by a sequence $21-22-\cdots$ that's the same as the suffix $01-02-\cdots$ of the second path obtained with respect to $0-01$, *except* that all entries are increased by 20, and '$14-13$' appears between 25 and 26. The next $c_{n-2}-1$ flips mimic (c); then six more flips give the reverse of β_n.

(e) When $n>1$, the number is $c_{n-1}+5$ in both cases(!), proved as in (d).

[The graphs C_n were introduced by K. Cameron, *Discrete Math.* **235** (2001), 69–77, who simplified a similar construction by A. Krawczyk and proved that $c_n\ge 2^n$. In 2020, Filip Stappers discovered that the generating function $c(z)=\sum_n c_n z^n$ is $p(z)/q(z)$, where $q(z)=(1-z)(1-3z-2z^2-2z^3-z^4-z^5)$ and $p(z)=z(1+z)(11+10z+6z^2+4z^3+z^4)$. He also proved that the number of one-sided flips to go from β_n to its mate γ_n, with respect to either $0-06$ or $06-0$, is $\hat c_n$, where $\sum_n \hat c_n z^n=\hat p(z)/q(z)$ and $\hat p(z)=2z(3+2z+z^2-2z^3)$. Consequently the actual limiting ratio c_{n+1}/c_n is $\rho\approx 3.709398$, the real root of $z^5=3z^4+2z^3+2z^2+z+1$. Asymptotically, $c_n\sim c\rho^n-8$ and $\hat c_n\sim\hat c\rho^n$, where $c\approx 5.349$ and $\hat c_n/c_n\sim\hat p(\rho)/p(\rho)\approx 0.3959$. In *Bull. Aust. Math. Soc.* **98** (2018), 18–26, L. Zhong introduced a family of graphs on $16n$ vertices for which the number of flips to get from a certain Hamiltonian path to its mate with respect to $0-1$ is *exactly* $6\cdot 2^n-10$. However, that number with respect to $1-0$ is only 4(!).]

78. (a, b) In contrast to exercise 60, success occurs with probability $\approx 100\%$ when $q\le 10$. Furthermore, a Hamiltonian cycle is usually found soon after finding the first Hamiltonian path. The average number of updates before that first cycle, observed in 100 runs for each q, was $\approx$ (81, 141, 146, 240, 295).

But $q=100$ was a different story. Here a 400-cycle was successfully found in only six of ten cases — sometimes after as few as 18 thousand updates, sometimes after as many as 8.3 million. In one of the other cases, millions of Hamiltonian paths (not cycles) were found; but memory overflow, with more than 2 million paths in the dictionary, aborted the run. Memory overflow also arose in the three other cases, once before achieving any paths longer than 365.

We conclude that Algorithm F can have wildly eccentric behavior, and it should probably be restarted if it spins its wheels too long.

Bondy
Murty
isolated vertices
Chvátal
Bondy
flip
degree sequence
forcibly Hamiltonian
graphical

79. (a) Let there be 12 vertices $\{0,\ldots,11\}$, with k — $(k{+}1)$ for $0\le k<12$ and $3k$ — $(3k{+}5)$ for $0\le k<4$ (modulo 12). This graph has two equivalence classes, each containing one Hamiltonian cycle and 12 Hamiltonian paths that aren't cycles.

(b) Let there be $13n$ vertices ij for $0\le i<n$ and $-1\le j<12$, with ik — ik' whenever k — k' in (a); also $i\overline{1}$ — $i0$ and $i1$ — $((i{+}1) \bmod n)\overline{1}$. This graph has 2^n equivalence classes, each containing one cycle and $17n$ noncycles.

80. If and only if $s\le t$ (except when $s=t=q_1=1$). [See J. A. Bondy and U. S. R. Murty, *Graph Theory* (2008), Theorem 18.1.]

81. (a) Choose nonadjacent $\{u,v\}$ with $\deg(u)\le\deg(v)$ so that $\deg(u)+\deg(v)$ is maximum, and assume that $\deg(u)+\deg(v)<n$. Let $k=\deg(u)$. Then $k>0$, because there are no isolated vertices when $d_{n-1}=n-1$. Exactly $n-1-\deg(v)\ge k$ vertices $\ne v$ are nonadjacent to v; these must all have degree $\le k$, by maximality. Similarly, exactly $n-k\ge\deg(v)$ vertices are nonadjacent to u, and they all have degree $\le\deg(v)$.

But $d_s\le t$ if and only if at least s vertices have degree $\le t$. Hence we have proved that $1\le k<n/2$, $d_k\le k$, and $d_{n-k}\le\deg(v)<n-k$, contradicting $(*)$.

(b) Each G_k satisfies $(*)$, so G_{k+1} exists. Let $(w_0w_1\ldots w_{n-1})$ be a cycle in G_{k+1} that's not also a cycle in G_k. We can assume that $w_0=u_k$ and $w_{n-1}=v_k$. There are $\deg(u_k)$ values of j with w_0 — w_{j+1} in G_k. And w_{n-1} — w_j for at least one such j, because $\deg(w_{n-1})\ge n-\deg(w_0)$. Thus $(w_0\ldots w_jw_{n-1}\ldots w_{j+1})$ is a cycle in G_k.

[Condition $(*)$ was discovered by V. Chvátal, *J. Comb. Theory* **B12** (1972), 163–168. This proof is due to J. A. Bondy and V. Chvátal, *Discr. Math.* **15** (1976), 111–135.]

82. Let G' be the graph $(kK_1\oplus K_{n-2k})$ — K_k, whose degree sequence has $d'_j=k$ for $0<j\le k$, $d'_j=n{-}1{-}k$ for $k<j\le n{-}k$, $d'_j=n{-}1$ for $n{-}k<j\le n$. (See exercise 80.)

83. (a) There are $(2r+1)r/2$ edges. (b) Use exercises 2 and 81.

(c) If u_0 — u_j and u_{j-1} — u_{2r}, a flip will create a cycle. So r vertices u_{j-1} cannot be adjacent to u_{2r}; the remaining r candidates must be u_{2r}'s neighbors.

(d) If the neighbors of u_0 are u_1, …, u_r and the neighbors of u_{2r} are u_r, …, u_{2r-1}, we have a $(2r+1)$-cycle. Otherwise let j be minimum such that $u_0\not-u_j$ and u_0 — u_{j+1}. Then u_{j-1} — u_{2r}, and we have a $2r$-cycle that excludes $v_0=u_j$.

(e) Assuming the hint, we can make a cycle v_1 — $\cdots$ — v_{2k-1} — v_0 — v_{2k+1} — $\cdots$ — v_1 that excludes v_{2k}, for any k; hence v_{2k} — v_j for all odd j. But then v_1 has degree $r+1$. [This result was announced in *Lecture Notes in Math.* **186** (1971), 201.]

Notice that Hamiltonicity is *not* implied by exercise 81, even though that exercise is "best possible" according to exercise 82. No efficient way is known to test whether all graphs with a given degree sequence are forcibly Hamiltonian.

84. Let $t=\lceil n/2\rceil$, and consider 2^{t-2} cases $a_1\ldots a_t$ where $a_1=a_t=0$ and $a_k=[d_k\le k]$ for $1\le k<t$. Then it's easy to see that the minimum $d_1+\cdots+d_n$ in case $a_1\ldots a_t$ occurs when $k<t$ and $a_k=0$ implies $d_k=k+1$, $d_{n-k}=d_{n-k-1}$; $a_k=1$ implies $d_k=d_{k-1}$, $d_{n-k}=n-k$; also $d_n=d_{n-1}$. (For example, if $n=11$ and $a_1\ldots a_6=010110$, we have $d_1\ldots d_{11}=22444677999$.) Let $s(a_1\ldots a_t)$ denote this minimum sum.

Suppose j is minimum with $a_j=1$, and k is minimum with $k>j$ and $a_k=0$. One can show without difficulty that $s(0^{k-1}a_k\ldots a_t)<s(0^{j-1}1^{k-j}a_k\ldots a_t)$, except that the inequality is reversed when n is odd and $j=t-1$. Consequently the overall minimum sum occurs uniquely for $d_1\ldots d_n=23\ldots(t{-}1)t^{t+2}$ when n is even, $23\ldots(t{-}1)(t{-}1)t^t$ when $n>3$ is odd. Increase d_n by 1 if the sum is odd.

The resulting sequence of degrees is graphical, by exercise 7–105. Hence the answer turns out to be $\lfloor(3n^2+6n)/16\rfloor$ when n is even; $\lfloor(3n^2+8n-3)/16\rfloor$ when n is odd.

85. The quadratic function $f(n,k)$ satisfies $f(n,k) \ge f(n,k+1)$ if and only if $k < \frac{n-1}{3}$. Thus $g(n,k) = \max_{k\le t<n/2} f(n,t)$. Every graph $(tK_1 \oplus K_{n-2t}) \mathbin{-\!\!-} K_t$ for $k \le t < n/2$ is non-Hamiltonian, with degree sequence $t^t(n-1-t)^{n-2t}(n-1)^t$ and $f(n,t)$ edges. Furthermore, every graph with $d_t \le t$ has at most t^2 edges that involve its first t vertices and at most $\binom{n-t}{2}$ edges that don't. Hence a graph with $d_1 \ge k$ and more than $g(n,k)$ edges must have $d_t > t$ for $k \le t < n/2$. And exercise 81 calls it Hamiltonian. [*Magyar Tudományos Akadémia Matematikai Kutató Int. Közl.* **7** (1962), 227–228.]

degree sequence components

86. Every graph $((t+1)K_1 \oplus K_{n-1-2t}) \mathbin{-\!\!-} K_t$, for $k \le t < (n-1)/2$, is untraceable. So we can achieve $\hat g(n,k) = \max(f(n,k), f(n,\lfloor n/2\rfloor - 1))$ edges, when $0 \le k < \lfloor n/2\rfloor$.

On the other hand, by exercises 2 and 81, a graph is traceable whenever its degree sequence $d_1 \le \cdots \le d_n$ satisfies the following condition:

$$1 \le t < (n+1)/2 \quad\text{and}\quad d_t < t \quad\text{implies } d_{n+1-t} \ge n-t. \tag{+}$$

In particular, a graph with minimum degree $d_1 \ge \lfloor n/2\rfloor$ is always traceable. If $k < \lfloor n/2\rfloor$ and (+) fails for some t, we have $d_s \le t-1$ for $1 \le s \le t$; $d_s \le n-t-1$ for $t < s \le n+1-t$; and $d_s \le n-1$ for $n+1-t < s \le n$. Hence $(d_1 + \cdots + d_n)/2 \le \hat f(n, t-1) \le \hat g(n,k)$. (The last inequality holds because $k \le t-1 \le \lfloor n/2\rfloor - 1$.)

88. (a) Let $v_0 \mathbin{-\!\!-} \cdots \mathbin{-\!\!-} v_l$ be a longest path, and assume that $l < 2k$. We will prove first that there's actually an l-cycle, using the fact that all neighbors of v_0 and v_l must lie on that path. Indeed, let $\{v_i \mid i \in I\}$ be the neighbors of v_0, and let $\{v_{j-1} \mid j \in J\}$ be the neighbors of v_l. Then I and J are subsets of $\{1,\ldots,l\}$. They can't be disjoint, because $|I| \ge k$ and $|J| \ge k$. Therefore there's some $j \in I \cap J$; and we have the cycle $v_0 \mathbin{-\!\!-} \cdots \mathbin{-\!\!-} v_{j-1} \mathbin{-\!\!-} v_l \mathbin{-\!\!-} \cdots \mathbin{-\!\!-} v_j \mathbin{-\!\!-} v_0$.

But there can't *be* an l-cycle! Since $l \le n-2$ and G is connected, there must be vertices w and w' not on the cycle, with $v_j \mathbin{-\!\!-} w \mathbin{-\!\!-} w'$ for some j. So there's a longer path.

(b) The result clearly holds for $n \le l+1$, because the number of edges is $\le \binom{n}{2} \le nl/2$. Also for larger n, if G isn't connected; for if there are r components, with n_j vertices and m_j edges in component j, each n_j is less than n. By induction, the number $m_1 + \cdots + m_r$ of edges is at most $(n_1 + \cdots + n_r)l/2 = nl/2$.

Assume therefore that $n > l+1$ and G is connected. Let $k = \lfloor l/2\rfloor + 1$. Then $2k > l$, so there's no path of length $2k$. Hence by (a), there's a vertex v of degree $< k$, unless $n = 2k = l+2$. And v exists even in that case; otherwise exercise 81 tells us there would be a cycle of length $2k$, hence a path of length $2k-1 > l$.

Now $G \setminus v$ has at most $(n-1)l/2$ edges; so G has at most $(n-1)l/2 + k - 1 \le nl/2$.

(c) $\lfloor n/(l+1)\rfloor K_{l+1} \oplus K_{n \bmod (l+1)}$, a graph with $\lceil n/(l+1)\rceil$ components. (The same number of edges is achieved by the much more interesting graph $K_{l/2} \mathbin{-\!\!-} \overline{K}_{n-l/2}$, if l is even and $n > l$ and $n \bmod (l+1) \in \{l/2, l/2+1\}$!)

89. (a) Let l be the length of G, and consider a longest path $v_0 \mathbin{-\!\!-} v_1 \mathbin{-\!\!-} \cdots \mathbin{-\!\!-} v_l$ where $v_l \mathbin{-\!\!-} v_p$ and p is as small as possible. The resulting cycle has length $c = l+1-p$; so we assume that $c < 2k$. A vertex v_q will be called "bounded" if its neighbors all belong to the cycle. We shall prove that v_q is bounded whenever $p < q \le l$.

The idea will be to construct a longest path $v_0 \mathbin{-\!\!-} \cdots \mathbin{-\!\!-} v_p \mathbin{-\!\!-} v'_{p+1} \mathbin{-\!\!-} \cdots \mathbin{-\!\!-} v'_l$, where $\{v'_{p+1},\ldots,v'_l\} = \{v_{p+1},\ldots,v_l\}$ and $v'_l = v_q$. Then v_q must be bounded, because l is maximum and p is minimum. Vertex v_l is clearly bounded; so is vertex v_{p+1}.

Suppose v_{q+1} is bounded, and let the neighbors of v_{p+1} and v_{q+1} be $\{v_i \mid i \in I\}$ and $\{v_j \mid j \in J\}$. Then $I \cup J \subseteq \{v_{p+1},\ldots,v_{l+1}\}$, where we set $v_{l+1} = v_p$. Also $|I|, |J| \ge k$.

If $i \in I$ and $i \le q$ and $i-1 \in J$, let $v_p v'_{p+1} \ldots v'_l = v_{l+1} \ldots v_{q+1} v_{i-1} \ldots v_{p+1} v_i \ldots v_q$. If $i \in I$ and $q < i \le l$ and $i+1 \in J$, let $v_p v'_{p+1} \ldots v'_l = v_{l+1} \ldots v_{i+1} v_{q+1} \ldots v_i \mathbin{-\!\!-}$

articulation point components
articulation point bicomponent
traceable
Historical notes
Erdős
Gallai
Faudree
Schelp
Woodall
cubic graphs
perfectly Hamiltonian
Coxeter
miracle

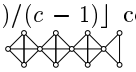
$v_{p+1} \ldots v_q$. One of these constructions must work; otherwise we'd have ruled out at least $k-1$ of the c potential elements of J, and we also have $q+1 \notin J$.

But $\{v_{p+1}, \ldots, v_l\}$ can't all be bounded! If $p = 0$, the graph G would be disconnected; otherwise vertex v_p would be an articulation point.

(b) The result clearly holds for $n \le c$, because the number of edges is $\le \binom{n}{2} \le (n-1)c/2$. Also for larger n, if G isn't connected; for if there are r components, with n_j vertices and m_j edges in component j, each n_j is less than n. By induction, the number $m_1 + \cdots + m_r$ of edges is at most $((n_1-1)+\cdots+(n_r-1))c/2 < (n-1)c/2$.

Assume therefore that $n > c$ and G is connected. If G isn't biconnected, there's an articulation point v that divides G into a bicomponent G' containing v and a connected graph $(G \setminus G') \cup v$. If G' has n' vertices, G has $\le (n'-1)c/2 + (n-n')c/2$ edges.

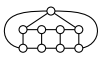
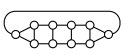
Finally, assume that G is biconnected and $n > c$. The proof follows as in exercise 88(b), because there exists a vertex whose degree is less than $k = \lfloor c/2 \rfloor + 1$.

(c) $K_1 \multimap (\lfloor (n-1)/(c-1) \rfloor K_{c-1} \oplus K_{(n-1) \bmod (c-1)})$. (The same number of edges is achieved by a traceable graph: Put $\lfloor (n-1)/(c-1) \rfloor$ copies of K_c and a $K_{1+(n-1) \bmod (c-1)}$ in a row, then paste them together;  if $(n,c) = (12,4)$.)

Historical notes: These results and those of the previous exercise are due to P. Erdős and T. Gallai [*Acta Mathematica Academiæ Scientiarum Hungaricæ* **10** (1959), 337–356]. R. J. Faudree and R. H. Schelp [*J. Combinatorial Theory* **B19** (1975) 150–160] proved that the lower bound of exercise 88(c) is sharp: The upper bound in 88(a) can be replaced by the size of those graphs. Similarly, D. R. Woodall [*Acta Math. Acad. Sci. Hung.* **28** (1976), 77–80] proved that the lower bound in (c) is sharp.

90. True, except when G has no edges (and length 0). See exercise 2.

93. (a) True, unless there are fewer than 4 vertices.

(b) Graphs like  and  for $n = 9$ and $n = 10$ work in general.

[*Mathematical Gazette* **49** (1965), 40–41. These cubic graphs for even n are also perfectly Hamiltonian. A more symmetrical graph, whose edges are $k \multimap (k+1)$ and $k \multimap (k+n/2)$ (modulo n), can also be used when n is a multiple of 4.]

95. (a) Powers of the "obvious" permutation $\sigma = (a_0a_1a_2a_3a_4a_5a_6)(b_0b_1b_2b_3b_4b_5b_6)$ $(c_0c_1c_2c_3c_4c_5c_6)(d_0d_1d_2d_3d_4d_5d_6)$ will take $d_6 \mapsto d_j$ for any j. There's also a "surprise," $\rho = (a_0b_0)(a_1b_2)(a_2d_2)(a_3c_2)(a_4c_5)(a_5d_5)(a_6b_5)(b_1b_6)(b_3d_6)(b_4d_1)(c_1d_4)(c_6d_3)$; one can verify that $u\rho \multimap v\rho$ whenever $u \multimap v$. (Notice that c_0, c_3, c_4, and d_0 are fixed by ρ. Coxeter called this "an apparent miracle.") When ρ is premultiplied and postmultiplied by appropriate powers of σ, we can take d_0 into any desired vertex.

The mapping $a_j \mapsto b_{5j}$, $b_j \mapsto c_{5j}$, $c_j \mapsto a_{5j}$, $d_j \mapsto d_{5j}$, namely the permutation $\tau = (a_0b_0c_0)\ (a_1b_5c_4a_6b_2c_3)\ (b_1c_5a_4b_6c_2a_3)\ (c_1a_5b_4c_6a_2b_3)\ (d_1d_5d_4d_6d_2d_3)$, is another automorphism that fixes d_0. When d_0 is fixed, we must take its neighbor c_0 into a neighbor; hence we can let $S_{26} = \{(), \tau, \tau^2\}$. And when c_0 is also fixed, we can let $S_{25} = \{(), \rho\}$, because b_0 must map to itself or a_0. Clearly $S_{24} = \{()\}$.

Finally, can we move anything else when d_0, c_0, b_0, a_0 are all fixed? Aha—there's just one possibility, namely τ^3, which swaps $a_j \leftrightarrow a_{-j}$, $b_j \leftrightarrow b_{-j}$, $c_j \leftrightarrow c_{-j}$, $d_j \leftrightarrow d_{-j}$, for $0 < j < 7$. Thus $S_{23} = \{(), \tau^3\}$ and $S_{22} = \cdots = S_1 = \{()\}$.

(b) Part (a) explains how to map $v \mapsto d_0 \mapsto v'$.

(c) In fact, part (a) shows that $u \multimap v \multimap w$ can be mapped to any $u' \multimap v' \multimap w'$.

(d) Algorithm H quickly shows that there are no Hamiltonian cycles. But there are 12 cycles, such as $a_1 \multimap a_0 \multimap a_6 \multimap a_5 \multimap a_4 \multimap a_3 \multimap a_2 \multimap d_2 \multimap c_2 \multimap c_6 \multimap$

d_6 — b_6 — b_1 — b_3 — d_3 — c_3 — c_0 — c_4 — d_4 — b_4 — b_2 — b_0 — b_5 — d_5 — c_5 — c_1 — d_1 — a_1, that omit (say) d_0.

Historical notes: The Coxeter graph was first discussed in print by W. T. Tutte [*Canadian Math. Bulletin* **3** (1960), 1–5], who proved it non-Hamiltonian. Eventually H. S. M. Coxeter wrote about "his graph" [*Proc. London Math. Society* (3) **46** (1983), 117–136], identifying its vertices with the $\binom{8}{2} = 28$ unordered pairs $\{x, y\}$ of the set $D = \{0, 1, 2, 3, 4, 5, 6, \infty\}$. His new names for vertices a_0 through d_7 were respectively 25, 36, 04, 15, 26, 03, 14; 34, 45, 56, 06, 01, 12, 23; 16, 02, 13, 24, 35, 46, 05; 0∞, 1∞, 2∞, 3∞, 4∞, 5∞, 6∞ (abbreviating $\{x, y\}$ by xy). If $0 \le x < y < 7$, the neighbors of xy are $\{2x - y, 3x - 2y\}$, $\{2y - x, 3y - 2x\}$, and $\{4x + 4y, \infty\}$, using arithmetic mod 7. He showed that the $7^3 - 7 = 336$ automorphisms correspond to the mappings $\{x, y\} \mapsto \{f(x), f(y)\}$, where f is a fractional linear transformation on D; that is, $f(x) = (ax + b)/(cx + d)$, where $0 \le a, b, c, d < 7$ and $(ad - bc) \bmod 7 \neq 0$ and either $c = 1$ or $(c, d) = (0, 1)$. (In this computation, $x/\infty = 0$, $x/0 = \infty$, and $f(\infty) = a/c$. The automorphisms σ, ρ, τ above correspond respectively to $f(x) = x + 1$, $1/x$, $5x$.)

Historical notes
Tutte
Coxeter
fractional linear transformation

100. (Using ideas of N. Beluhov.) When C is a cycle cover, let $s_j = 4[t_j \text{—} t_{j+1} \in C] + 2[v_j \text{—} w_{j+1} \in C] + [w_j \text{—} v_{j+1} \in C]$ encode its edges between indices j and $j + 1$ modulo q. A simple case analysis shows that $s_j \neq 0$; $s_j \in \{1, 2, 4\} \implies s_{j+1} = 7$; $s_j = 3 \implies s_{j+1} \in \{5, 6\}$; $s_j = 5 \implies s_{j+1} \in \{3, 5\}$; $s_j = 6 \implies s_{j+1} \in \{3, 6\}$; $s_j = 7 \implies s_{j+1} \in \{1, 2, 4\}$; and that the sequence $s_1 s_2 \ldots s_q$ completely determines C.

Thus there are two kinds of covers: Type A, where s_j is alternately 7 and an element of $\{1, 2, 4\}$; or type B, where each s_j is an element of $\{3, 5, 6\}$. Type A covers arise only when q is even, and they have $k + 1$ cycles when there are k occurrences of $s_j = s_{j+2} \neq 7$. Type B covers always have exactly 2 cycles.

Let $g(w, z) = \sum w^{[a_0 = a_1] + \cdots + [a_{n-1} = a_n]} z^n [a_0 = a_n]$, summed over all ternary sequences $a_0 a_1 \ldots a_n$, and let $h(w, z)$ be similar but requiring $a_0 \neq a_n$. Then $g(w, z) = 3 + wzg(w, z) + zh(w, z)$ and $h(w, z) = 2zg(w, z) + (1 + w)zh(w, z)$. So we find $g(w, z) = 3(1 - (1 + w)z)/((1 - (w - 1)z)(1 - (w + 2)z)) = 2/(1 - (w - 1)z) + 1/(1 - (2 + w)z)$. Consequently the number of type A covers with k cycles is $2\,[w^{k-1} z^{q/2}]\, g(w, z) = 4\binom{q/2}{k-1}(2^{q/2-k} - (-1)^{q/2-k})$ when q is even. (In particular, the number of Hamiltonian cycles is $4(2^{q/2-1} + (-1)^{q/2})$.)

Turning to type B, let there be f_{xyn} sequences $a_0 \ldots a_n$ with $a_0 = x$, $a_n = y$, and each $a_j \in \{3, 5, 6\}$, having no consecutive 33 or 56 or 65. We find by induction that $f_{xyn} = (2^n - (-1)^n)/3 + \delta_{xyn}$, where $\delta_{xyn} = 1$ when n is even and $x = y$, $\delta_{xyn} = -1$ when n is odd and $xy \in \{33, 56, 65\}$, otherwise $\delta_{xyn} = 0$. Hence there are $f_{33q} + f_{55q} + f_{66q} = 2^q + 2[q \text{ even}]$ covers of type B.

103. Let $H = v_0 \text{—} v_1 \text{—} \cdots \text{—} v_n = v_0$. Every edge of $G \setminus H$ has the form $e_{ij} = v_i \text{—} v_j$ for some $0 \le i < j < n$. When G is drawn in the plane with no crossing edges, two edges e_{ij} and $e_{i'j'}$ with $i < i' < j < j'$ cannot both lie inside H, nor can they both lie outside H. Therefore the graph E whose vertices are the e_{ij}, with e_{ij} adjacent to $e_{i'j'}$ when $i < i' < j < j'$, must be bipartite. Conversely, if E is bipartite, G is clearly planar. (And bipartiteness is readily tested by Algorithm 7B.)

Historical notes: This criterion for planarity was discovered by G. Demoucron, Y. Malgrange, and R. Pertuiset [*Revue Française de Recherche Opérationnelle* **8** (1964), 33–47] and independently by W. Bader [*Archiv für Elektrotechnik* **49** (1964), 2–12], at the time when planarity of printed circuits began to be important. A graph is planar if and only if its blocks (biconnected components) are planar; and in practice, a block that

Historical notes
Beluhov
generating functions
ternary sequences
bipartite
Historical notes
Demoucron
Malgrange
Pertuiset
Bader
printed circuits
blocks
biconnected components

isn't a single edge is almost always Hamiltonian. Notice that the nonplanar graphs K_5 and $K_{3,3}$ have Hamiltonian cycles, and the corresponding graphs E aren't 2-colorable.

105. (a) $[n\ge3]\,n!/(2n)$, one for every pair $\{\pi,\pi^-\}$ where π is a cyclic permutation.
(b) $[m=n\ge2]\,n!^2/(2n)$.

106. See *AMM* **134** (2027), solution to problem 12561, to appear.

108. (a) Those are the only remaining ways to include vertex 28 in the cycle.
(b) $\binom{08}{16}$ would form a short cycle; so 08 must be covered by $\binom{08}{27}$.
(c) 14 must be covered by $\binom{06}{14}$ and $\binom{14}{26}$; but then 26 — 14 — 06 — 18 — 26.
(d) We must choose $\binom{14}{22}$ to avoid the contradiction, after which $\binom{03}{15}$ leaves 23 — 04 — 25 and 05 — 24 — 16 as the only ways to cover 04 and 24. Then $\binom{07}{15}$ is forced, and almost everything is nailed down. Hence there are two ways to complete a cycle after the choices of (a), (b), and (d): either $\{\binom{05}{13},\binom{06}{25},\binom{14}{26}\}$ or $\{\binom{05}{26},\binom{06}{14},\binom{13}{25}\}$.

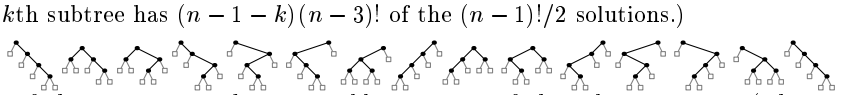
109. (In this graph we have `NAME(0)` = 00, `NAME(1)` = 01, ..., `NAME(29)` = 29.) `MATE(0)` is ≥0; `MATE(1)` = 21; `MATE(2)` = 22; `MATE(3)` = 23; `MATE(4)` = -1.

111. `TRIG` needs at most n locations, because no vertex can be a trigger more than once (when its degree drops to 2). `ACTIVE` needs at most $\binom{n+1}{2}$ locations, because at most $n-l$ vertices are outer in level l. `SAVE` needs at most n^2 slots, where each "slot" holds a mate and a degree, because there can be at most n levels.

112. (a) activate(u); activate(w); remarc(u,v); remarc(w,v); and makemates(u,w).
(b) activate(u); remarc(u,v); purge(w,v); makemates(`MATE(`w`)`,u); deactivate(w).
Here 'purge(w,v)' means 'remarc(`NBR[`w`][`k`]`,w) for k decreasing from `DEG(`w`)`-1 down to 0, except when `NBR[`w`][`k`]` = v'; it removes w from the lists of its non-v neighbors.
(c) activate(w); remarc(w,v); purge(u,v); makemates(`MATE(`u`)`,w); deactivate(u).
(d) Do nothing if $e=n$. Otherwise purge(u,v); purge(w,v); makemates(`MATE(`u`)`, `MATE(`w`)`); deactivate(u); deactivate(w).

113. Set $x[k]\leftarrow-1$ for $0\le k<n$. Then do this for $0\le k<n$: If x[`EU[`k`]`] <0, set x[`EU[`k`]`] $\leftarrow$ `EV[`k`]`; else set y[`EU[`k`]`] $\leftarrow$ `EV[`k`]`. If x[`EV[`k`]`] <0, set x[`EV[`k`]`] $\leftarrow$ `EU[`k`]`; else set y[`EV[`k`]`] $\leftarrow$ `EU[`k`]`. Finally set $v_1\leftarrow0$, $v_2\leftarrow x[0]$, and for $3\le k\le n$ set $v_k\leftarrow(v_{k-2}=x[v_{k-1}]?\ y[v_{k-1}]:\ x[v_{k-1}])$.

115. Assume that $n>4$. The root has degree $n-2$. The kth subtree, for $1\le k<n-2$, is $T(n-1-k,n-3,n-2,\ldots,2)$; and the last subtree is $T(n-3,n-2,\ldots,2)$. Here $T(d_0,\ldots,d_{r-1})$ denotes the complete tree with d_l-way branching at level l for $0\le l<r$. (The kth subtree has $(n-1-k)(n-3)!$ of the $(n-1)!/2$ solutions.)

116. 
is one of the six ways to do 14 suitable rotations of those binary trees. (These are also the Hamiltonian cycles of an *associahedron*; see exercise 7.2.1.6–29.)

117. (a) True. ($t(G)=0$ if and only if $U=\emptyset$ disconnects G.)
(b) If $|U|/k(G\backslash U)\ne|U|/k(G\backslash e\backslash U)$, edge e joins two components of $G\backslash e$; hence $k(G\backslash U)=k(G\backslash e\backslash U)-1$ (and the term for this U leaves the 'min' if $k(G\backslash U)=1$).
(c) This follows from the monotonicity proved in (b), since $t(C_n)\ge1$.
(d) After cutting out m' vertices of the smaller part and n' vertices of the larger part, the residual graph that's left is connected unless $m'=m$ or $n'=n$. The smallest ratio $(m'+n')/(m-m'+n-n')$ is m/n in such cases. (See exercise 105.)
(e) 4/3, by cutting 4 independent vertices. (Let N be the "net" graph, ; the smallest non-Hamiltonian tough graph is K_1 — N. A 42-vertex non-Hamiltonian graph

K_5
$K_{3,3}$
2-colorable
cyclic permutation
`NAME(`v`)`
purge(w,v)
complete tree
rotations
binary trees
associahedron
"net" graph

with $t(G) = 2$ was (surprisingly) constructed by D. Bauer, H. J. Broersma, and H. J. Veldman, in *Discrete Applied Math.* **99** (2000), 317–321; it's tough for Algorithm H!)

(The graph $N_{t,m}$ in exercise 302 is non-tough because $t(N_{t,m}) \le t/(t+1)$. Chvátal's original paper about toughness appeared in *Discrete Math.* **5** (1973), 215–228.)

Bauer
Broersma
Veldman
Chvátal
gadget
Fleischner
Frucht graph
automorphisms
identity
historical notes
Halin

118. (Solution by V. Chvátal.) Let the vertices of $G = K_m \mathbin{\square} K_n$ be $V \times W$, where $|V| = m$ and $|W| = n$. Given $p > 1$, the smallest set U that makes $k(G \setminus U) = p$ has the form $(V \times W) \setminus (V_1 \times W_1 \cup \cdots \cup V_p \times W_p)$, where $V_1 \cup \cdots \cup V_p$ and $W_1 \cup \cdots \cup W_p$ are set partitions of V and W. Hence $|U| = mn - m_1 n_1 - \cdots - m_p n_p$, where the sizes $|V_j| = m_j$ and $|W_j| = n_j$ are positive integers that sum to m and n. It is minimized when $m_1 = m + 1 - p$, $n_1 = n + 1 - p$, and all other m_j and n_j are 1; in other words, the smallest such $|U|$ is $mn - (p-1) - (m+1-p)(n+1-p) = (p-1)(m+n-p)$. Hence $t(G) = \min_{p=2}^{\min(m,n)} (p-1)(m+n-p)/p = (m+n-2)/2$.

119. They take $v \mapsto (\frac{av+b}{cv+d}) \bmod 47$, where $(a,b,c,d) = (20,15,17,27)$, $(20,17,15,27)$, $(31,19,21,16)$, and $(31,21,19,16)$. (We have $1/\infty = 0$, $1/0 = \infty$, and $\infty \mapsto \frac{a}{c} \bmod 47$.)

120. (a) The automorphisms are generated by $(i,j,k,u,v,w,x,y,z,U,V,W,X,Y,Z) \mapsto (j,k,i,V,W,U,X,Y,Z,v,w,u,z,x,y)$ or $(i,j,k,U,V,W,Z,X,Y,u,v,w,y,z,x)$. The cycles x — U — i — u — Z — y — V — j — v — X — z — W — k — w — Y — x and Z — u — i — U — x — X — v — j — V — y — Y — w — k — W — z — Z are quickly found by Algorithm H (just 50 nodes in the search tree).

(b) Since $G_0^{(t)}$ has a unique Hamiltonian path from $x^{(t)}$ to $X^{(t)}$, this construction uses it as a "gadget" to prevent any of the edges between Q_t and $G_0^{(t)}$ from appearing in any Hamiltonian cycle of G_t. The proof relies on the fact that q_t — p_t — Q_t — q_t — P_t.

(See H. Fleischner, *J. Graph Theory* **75** (2014), 167–177, Lemma 1. He goes on to define G_4 and G_5, thereby removing the degree-3 vertices Y and z in a similar way; those reductions introduce many more cycles, yet only one of them includes the edge u — U. Another trick removes y and Z, in a graph G_6 that's half of his tour-de-force!)

122. (a) For example, the triangles $\{\mathtt{b},\mathtt{e},\mathtt{f}\}$, $\{\mathtt{g},\mathtt{k},\mathtt{l}\}$, $\{\mathtt{d},\mathtt{i},\mathtt{j}\}$ must correspond somehow to the triangles $\{\mathtt{A},\mathtt{B},\mathtt{J}\}$, $\{\mathtt{F},\mathtt{G},\mathtt{H}\}$, $\{\mathtt{C},\mathtt{D},\mathtt{L}\}$. Hence the other vertices $\{\mathtt{a},\mathtt{c},\mathtt{h}\}$ and $\{\mathtt{E},\mathtt{I},\mathtt{K}\}$ must also correspond to each other in some order. The solution is

$$(\mathtt{a},\mathtt{b},\mathtt{c},\mathtt{d},\mathtt{e},\mathtt{f},\mathtt{g},\mathtt{h},\mathtt{i},\mathtt{j},\mathtt{k},\mathtt{l}) \leftrightarrow (\mathtt{I},\mathtt{J},\mathtt{K},\mathtt{H},\mathtt{A},\mathtt{B},\mathtt{L},\mathtt{E},\mathtt{F},\mathtt{G},\mathtt{C},\mathtt{D}).$$

[It's unique, because this happens to be the "Frucht graph," one of the smallest cubic graphs that has no automorphisms except the identity. See R. Frucht, *Canadian J. Math.* **1** (1949), 365–378. Halin graphs were introduced by R. Halin, *Combinatorial Mathematics and its Applications* (Oxford conference, 1969), 129–136.]

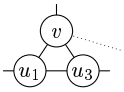
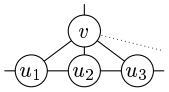
(b) For each nonleaf of T except the root, introduce the chord i — $((j+1) \bmod q)$ when its descendant leaves are $x_i \ldots x_j$. (The chords for $\mathtt{b}$, $\mathtt{c}$, $\mathtt{d}$, $\mathtt{g}$ in the example are 0 — 2, 2 — 5, 5 — 0, 2 — 4, because $x_0 \ldots x_6 = \mathtt{efklhij}$.)

(c) Choose any region to be the root. The other regions and sides will form a tree T, when we ignore the adjacencies between sides of C, because the other adjacencies form no cycles. The children of each region in T are its adjacent regions and sides, except for the parent, in (say) clockwise order. For instance, if we choose root $\mathtt{I}$ in the example, the children of $\mathtt{I}$ might be $(\mathtt{J},\mathtt{K},\mathtt{H})$; the children of $\mathtt{J}$ are $(\mathtt{A},\mathtt{B})$; the children of $\mathtt{K}$ are $(\mathtt{L},\mathtt{E})$; etc.; we could also have decided to let the children of $\mathtt{I}$ be $(\mathtt{K},\mathtt{H},\mathtt{J})$ or $(\mathtt{H},\mathtt{J},\mathtt{K})$. Or we could have chosen root $\mathtt{K}$, with children $(\mathtt{E},\mathtt{I},\mathtt{L})$ or $(\mathtt{I},\mathtt{L},\mathtt{E})$ or $(\mathtt{L},\mathtt{E},\mathtt{I})$; then the children of $\mathtt{I}$ would be $(\mathtt{H},\mathtt{J})$, etc.

123. The answers for $4 \le n \le 16$ are 1, 1, 2, 2, 4, 6, 13, 22, 50, 106, 252, 589, 1475, computed by using definition (ii). See A. Howroyd, OEIS A346779 and A380362 (2025).

124. By induction on the size of T in exercise 122(i). The result is clear when tree T has depth 1. Otherwise some nonroot vertex $v \in T$ has $d \ge 2$ children $u_1 \dots u_d$, all leaves.

Case 1: $d > 2$. Let H_0 be the Halin graph obtained by deleting leaf u_2. Then

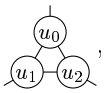
H_0 has the subgraph  where H has the subgraph  .

Since u_2 has degree 3, the Hamiltonian cycles of H have three possible forms:

$$u_2 — v — \alpha — u_1 — u_2, \quad u_2 — v — \alpha — u_3 — u_2, \quad u_2 — u_1 — \alpha — u_3 — u_2,$$

where the middle part is a Hamiltonian path of H_0. And such paths in H_0 arise from Hamiltonian cycles that respectively include the edges $v — u_1$, $v — u_3$, $u_1 — u_3$. So H has cycles that include/exclude $u_2 — u_1$, $u_2 — u_3$, $u_2 — v$. Furthermore, a Hamiltonian cycle of H_0 that excludes $u_1 — u_3$ must include $u_1 — v — u_3$; so we obtain Hamiltonian cycles in H that include/exclude $v — u_1$, $v — u_3$, and the edges from u_1 and u_3 to their anonymous neighbors. It's easy to include/exclude the other edges.

Case 2: $d = 2$. Now obtain H_0 by changing v to a leaf; in this case

H_0 has the subgraph where H has the subgraph  , letting $u_0 = v$.

By threefold symmetry, the Hamiltonian cycles of H that avoid edge $u_i — u_j$ correspond precisely to the Hamiltonian cycles of H_0 that avoid the "opposite" edge from v.

[Considerably more is also true; see Z. Skupień, *Contemporary Methods in Graph Theory* (Mannheim, 1990), 537–555. Uniform Hamiltonicity was introduced by C. A. Holzmann and F. Harary in *SIAM J. Applied Math.* **22** (1972), 187–193.]

125. $i_{97} — j_{97} = 58 — 54 = \pi_{78004}\pi_{78005} — \pi_{78006}\pi_{78007}$.

127. `GA` is impossible, because the arcs `AL` — `FL` — `GA` and `GA` — `SC` — `NC` are forced.

128. C, H, P, and U. (See exercise 103.)

129. (In the modified step H11, we can terminate the loop immediately if we encounter a vertex of degree 0 or 1.) The modified algorithm works surprisingly well: It wins convincingly on graphs A, G, and U (58 Mμ, 2762 Gμ, and 27 Gμ); it ties or does slightly better on graphs C, H, and T. It's slightly slower on graphs B, P, and Q; and it's more than 25% slower on graphs D, E, F, R, S.

On the 8×8 knight graph, min-remaining-values takes about 6.0 petamems to find all 13 billion solutions, compared to 6.7 petamems for max-remaining-values.

130. In other words, if the present state of the computation can lead to a Hamiltonian cycle C, the current graph G' must have a Hamiltonian cycle C'. Indeed, that C' can be exhibited by replacing each subpath in C by the corresponding virtual edge of G'.

(Conversely, every Hamiltonian cycle of G' that actually uses every virtual edge corresponds to a unique Hamiltonian cycle C of G. There might, however, be other Hamiltonian cycles in G'. This graph G' was defined by W. Kocay in his paper of 1992, but he doesn't seem to have realized its full potential for pruning the search.)

One can, for example, discover whether or not G' has an articulation point by using Hopcroft and Tarjan's efficient depth-first algorithm for bicomponents (Algorithm 7.4.1.2B, or BOOK_COMPONENTS in *The Stanford GraphBase*).

Howroyd
OEIS
Skupień
Holzmann
Harary
historical notes
8×8 knight graph
Kocay
pruning the search
articulation point
Hopcroft
Tarjan
depth-first
bicomponents
Stanford GraphBase

central symmetry
Jelliss
Bergholt
Euler
Hamiltonian *paths*

133. No! Let C be a cycle $(0,0)=v_0$ — v_1 — $\cdots$ — $v_{n^2}=(0,0)$ for which u — v is in C if and only if $u\alpha$ — $v\alpha$ is also in C, where $(i,j)\alpha=(j,i)$ denotes reflection about the main diagonal. Let $k>0$ be minimum with v_k on the diagonal; that is, $v_k=v_k\alpha$. Then $v_{k+1}=v_{k-1}\alpha$, because $v_{k-1}\alpha$ — $v_k\alpha$ is the other edge of C that touches v_k. Similarly, $v_{k+2}=v_{k-2}\alpha$, and so on; we find $v_{2k}=v_0$. Hence $2k=n^2$, and only two elements of the diagonal are in C. Contradiction.

The same argument shows more generally that no nontrivial automorphism α of a rectangular board can be a symmetry of a knight's cycle when α has fixed points.

134. Let $(i,j)\alpha=(m-1-i,j)$, and let the cycle begin $(0,0)=v_0$ — v_1 — $\cdots$ — $v_k=(m-1,0)$. Notice that m is even; otherwise $((m-1)/2,0)$ would be a fixed point of α. Therefore k is odd. *Case 1:* $v_1\alpha=v_{k+1}$. Then $v_{k+2}=v_2\alpha$, ..., and $v_{2k}=v_k\alpha=v_0$. So $mn/2=k$ is odd. *Case 2:* $v_{k-1}=v_1\alpha$. Then $v_{k-j}=v_j\alpha$ for $0\le j\le k=2l+1$. But we can't have both v_{l+1} — v_l and $v_{l+1}=v_l\alpha$.

[Similar conditions apply to central symmetry, as we'll see in exercise 136. These results are due to G. P. Jelliss, *Chessics* **2**, 22 (Summer 1985), 64.]

135. Let graph G have $N=mn/2$ vertices, one for each pair $\{xy,\bar{x}y\}$ of equivalent cells; here $0\le x<m/2$, $0\le y<n$, and $\bar{x}=m-1-x$. The neighbors of $\{xy,\bar{x}y\}$ in G are $\{x'y',\bar{x}'y'\}$ for all knight moves xy — $x'y'$ with $0\le x'<m$ and $0\le y'<n$. (For example, when $m=10$ we have $\{30,60\}$ — $\{41,51\}$, since 30 — 51 and $\{51,41\}=\{41,51\}$.)

Given a Hamiltonian cycle $\{00,\bar{0}0\}=v_0$ — v_1 — $\cdots$ — $v_N=\{00,\bar{0}0\}$ in G, there's a unique knight path $00=x_0y_0$ — x_1y_1 — $\cdots$ — x_Ny_N with $x_ky_k\in v_k$ for $0\le k\le N$. We must have $x_Ny_N=\bar{0}0$, because N is odd. Therefore we get an $m\times n$ knight's cycle by defining $x_{N+k}y_{N+k}=\bar{x}_ky_k$ for $0\le k\le N$.

136. We'll need names for these two kinds of symmetry. The right-hand species of symmetry is called *Bergholtian*, because it was discovered by Ernest Bergholt [*British Chess Magazine* **38** (1918), 104, 195; see also *The Games and Puzzles Journal* **2**, 14 (16 December 1996), 234]. The left-hand species is called *Eulerian*, because Leonhard Euler gave many examples of such cycles in §25–§34 of his 1759 memoir.

As in answer 135, we define a graph G with $N=mn/2$ vertices; but this time the vertices represent pairs $\{xy,\overline{xy}\}$, where $\overline{xy}=(m-1-x)(n-1-y)$. The neighbors of $\{xy,\overline{xy}\}$ are, similarly, the vertices $\{x'y',\overline{x'y'}\}$ obtained from knight moves xy — $x'y'$. Now, however, there's a slight problem: There are two "self-loops," because we can have xy — $x'y'$. (More precisely, we have u_0 — u_0 and u_1 — u_1, where $u_0=\{(\frac{m-2}{2})(\frac{n-3}{2}),(\frac{m}{2})(\frac{n+1}{2})\}$ and $u_1=\{(\frac{m-2}{2})(\frac{n+1}{2}),(\frac{m}{2})(\frac{n-3}{2})\}$.) It may seem best to simply disallow those self-loops; after all, a self-loop can't be in a Hamiltonian cycle.

But further analysis reveals that the Bergholtian solutions correspond precisely to the Hamiltonian *paths* between u_0 and u_1. Indeed, from a path $u_0=v_0$ — $\cdots$ — $v_{N-1}=u_1$ in G, we get x_0y_0 — $\cdots$ — $x_{N-1}y_{N-1}$ with each $x_ky_k\in v_k$, where $x_0y_0=(\frac{m-2}{2})(\frac{n-3}{2})$. Then x_0y_0 — $\cdots$ — $x_{N-1}y_{N-1}$ — $\overline{x_{N-1}y_{N-1}}$ — $\cdots$ — $\overline{x_0y_0}$ — x_0y_0 is a Bergholtian cycle.

On the other hand, a Hamiltonian *cycle* in G, say $\{00,\overline{00}\}=v_0$ — $\cdots$ — $v_N=\{00,\overline{00}\}$, will lead similarly to $00=x_0y_0$ — $\cdots$ — x_Ny_N. And it will yield an Eulerian cycle if and only if $x_Ny_N=\overline{00}$, which happens if and only if N is odd.

We conclude that if $n \bmod 4=2$, we should add the special edge u_0 — u_1 to G. Then its Hamiltonian cycles will correspond precisely to all of the centrally symmetric $m\times n$ knight cycles; they're Bergholtian if the special edge is used, Eulerian otherwise.

But if $n \bmod 4 = 0$, there aren't any $m \times n$ Eulerian cycles. We get the Bergholtian ones by adding *two* special edges, u_0 — ! — u_1, where '!' is a new vertex.

multigraph de Jaenisch

137. Again we construct G with $N = mn/2$ vertices $\{xy, \overline{xy}\}$. But there's a new complication: G is a *multigraph*, with four edges that occur twice! Indeed, when $x = \frac{m-4}{2}$, $y = \frac{n-4}{2}+k$, $x' = \frac{m-2}{2}$, and $y' = \frac{n-4}{2}+k'$, where $0 \le k < 4$ and $k' = (k+2) \bmod 4$, we have both xy — $x'y'$ and xy — $\overline{x'y'}$. Hence $\{xy, \overline{xy}\}$ — $\{x'y', \overline{x'y'}\}$ is a double edge.

For example, G has 32 vertices when $m = n = 8$, namely $\binom{00}{77}, \binom{01}{76}, \ldots, \binom{07}{70}, \binom{10}{67}, \binom{11}{66}, \ldots, \binom{17}{60}, \ldots, \binom{37}{40}$. (We write '$\binom{wx}{yz}$' as a convenient shorthand for vertex '$\{wx, yz\}$'.) The double edges for this case turn out to be $\binom{22}{55}$ — $\binom{34}{43}$, $\binom{23}{54}$ — $\binom{35}{42}$, $\binom{24}{53}$ — $\binom{32}{45}$, $\binom{25}{52}$ — $\binom{33}{44}$; and G also has 76 single edges. Algorithm H needs fewer than 800 megamems to visit each of G's 2,451,830 Hamiltonian cycles, one of which is

00 21 35 30 26 07 15 36 20 01 13 34 24 05 17 25 04 16 37 32 11 03 22 14 06 27 31 10 02 23 33 12 00
77 56 42 47 51 70 62 41 57 76 64 43 53 72 60 52 73 61 40 45 66 74 55 63 71 50 46 67 75 54 44 65 77 .

This cycle doesn't use any of the double edges; so we can uniquely extract a corresponding knight path that begins at 00, proceeding from left to right:

00 21 42 30 51 70 62 41 20 01 13 34 53 72 60 52 73 61 40 32 11 03 22 14 06 27 46 67 75 54 33 12 00 .

Hmmm. Bad luck. Only 32 cells have been touched before the knight has returned to its starting point, 00; hence this Hamiltonian cycle of G doesn't correspond to a valid knight's cycle of the full 8×8 board. (Its complement tours the other 32 cells.)

Let's try again. Here's another Hamiltonian cycle that's double-move free:

00 21 35 30 26 07 15 36 20 01 22 14 06 27 31 10 02 23 04 16 37 25 17 05 13 34 24 03 11 32 33 12 00
77 56 42 47 51 70 62 41 57 76 55 63 71 50 46 67 75 54 73 61 40 52 60 72 64 43 53 74 66 45 44 65 77 .

This one brings better news when we extract the corresponding knight path:

00 21 42 30 51 70 62 41 20 01 22 14 06 27 46 67 75 54 73 61 40 52 60 72 64 43 24 03 11 32 44 65 77 ;

aha, it ends in 77! We get a full knight's cycle by appending the complementary steps.

Consider now a Hamiltonian cycle of G that *does* use one of the double edges:

00 21 35 30 26 07 15 36 20 01 13 05 17 25 04 16 37 32 11 03 24 34 22 14 06 27 31 10 02 23 33 12 00
77 56 42 47 51 70 62 41 57 76 64 72 60 52 73 61 40 45 66 74 53 43 55 63 71 50 46 67 75 54 44 65 77 .

(The culprit is '$\binom{34}{43}$ — $\binom{22}{55}$', aka '$\binom{22}{55}$ — $\binom{34}{43}$'.) Knight-path extraction is now ambiguous,

00 21 42 30 51 70 62 41 20 01 13 05 17 25 04 16 37 45 66 74 53 34 $*$ 22 14 06 27 46 67 75 54 33 12 00 ,

because 34 can be followed by either 22 or 55. We'd better choose 55; that will complement all of the subsequent steps, and we'll end up with 77 as desired.

Next let's look at the path in G that corresponds to a famous knight's cycle that C. F. de Jaenisch [*Traité des applications de l'analyse math. au jeu des échecs* **2** (1862), 35–37] proudly called "seven-fold reentrant":

00 21 33 32 24 03 11 30 35 23 04 16 37 25 17 05 13 01 20 36 15 07 26 34 22 10 02 14 06 27 31 12 00
77 56 44 45 53 74 66 47 42 54 73 61 40 52 60 72 64 76 57 41 62 70 51 43 55 67 75 63 71 50 46 65 77 .

This one has *three* double edges, hence $2^3 = 8$ ways to resolve its ambiguities:

00 21 33 45 $*$ 24 03 11 30 42 $*$ 23 04 16 37 25 17 05 13 01 20 41 62 70 51 43 $*$ 22 10 02 14 06 27 46 65 77 .

Four of those eight will produce 77 at the right.

Can all four of the double edges participate? Yes, but such cases are much rarer:

00 21 35 23 04 16 37 32 24 03 11 30 26 07 15 27 06 14 02 10 31 34 22 01 20 36 17 05 13 25 33 12 00
77 56 42 54 73 61 40 45 53 74 66 47 51 70 62 50 71 63 75 67 46 43 55 76 57 41 60 72 64 52 44 65 77 .

Eight of the sixteen knight-path extractions are therefore fruitful in

00 21 42 $*$ 23 04 16 37 45 $*$ 24 03 11 30 51 70 62 50 71 63 75 67 46 34 $*$ 22 01 20 41 60 72 64 52 $*$ 33 12 00 .

Altogether the Hamiltonian cycles of G include exactly 1076876 without double edges, of which 536360 are unlucky; plus (978316, 341706, 52192, 2740) that have respectively (1, 2, 3, 4) doubles. That makes $1076876 - 536360 + 978316 + 2 \cdot 341706 + 4 \cdot 52192 + 8 \cdot 2740 = 2432932$ centrally symmetric tours, which form 608233 sets of 4.

138. (Each value of m is accompanied by the totals for $n = 3, 5, 7, \ldots$.)

Vertical symmetry. $m = 6$: 0, 4, 530, 20582, 994660, 45129332, 2082753196. $m = 10$: 4, 2266, 18480426, 56275825112. $m = 14$: 24, 722396, 539780910056. $m = 18$: 276, 238539296. $m = 22$: 2604. $m = 26$: 25736. $m = 30$: 248816.

Eulerian symmetry. $m = 6$: 0, 0, 526, 22210, 1477090, 100121632, 6606415888. $m = 10$: 0, 1212, 16330492, 49470226538. $m = 14$: 16, 498926, 529843978930. $m = 18$: 124, 167812624. $m = 22$: 1404. $m = 26$: 12824. $m = 30$: 126696.

Bergholtian symmetry. $m = 6$: 0, 0, 38, 3724, 363594, 19156740, 1265006728. $m = 10$: 4, 494, 3346312, 19308979910. $m = 14$: 8, 123028, 101557666784. $m = 18$: 152, 47966908. $m = 22$: 1200. $m = 26$: 12912. $m = 30$: 122120.

(We might as well also record here the other cases of Bergholtian symmetry. $m = 8$: 0, 22, 21968, 17072474, 8868635684. $m = 12$: 0, 8858, 452675596. $m = 16$: 48, 3145086. $m = 20$: 352. $m = 24$: 3752. $m = 28$: 34768. $m = 32$: 346128.)

(Algorithm H's running time for these graphs G is roughly 500 mems per solution. The totals for $(m, n) = (6, 15)$ and $(14, 7)$ were obtained by Algorithm E.)

Vertical symmetry
Eulerian symmetry
Bergholtian symmetry
dynamic enumeration
self-loops
transpose
Marlow
census
central wedges
canonical bunches

139. Let G be a graph with 25 vertices, one for each class of four cells $\{xy, y\bar{x}, \bar{x}\bar{y}, \bar{y}x\}$ that are rotationally equivalent, where $\bar{x} = 9-x$. Adjacency is defined by giraffe moves; we must omit the self-loops from $\{23, 37, 76, 62\}$ and $\{32, 26, 67, 73\}$ to themselves. Furthermore, we remove one of the two edges between $\{22, 27, 77, 72\}$ and $\{33, 36, 66, 63\}$.

This 51-edge graph has 56 Hamiltonian cycles (found in just 33 Kμ); but the actual number is 100, because 44 of those cycles include the double edge. That yields 100 ways to cover a 10×10 board with symmetrical Hamiltonian cycles.

A cycle and its transpose are both counted. Hence there are exactly 50 essentially distinct solutions. [They were first discovered by T. W. Marlow, shortly after he had enumerated the 415902 essentially distinct 10×10 *knight* cycles with 90° symmetry. See *The Games and Puzzles Journal* **2**, 16 (15 May 1999), 288–291.]

141. Multiply the number of Hamiltonian cycles of the 8×8 knight graph ($\approx$ 13 trillion) by 64 (to place '1') and by 2 (to place '2'): 1,698,222,644,548,096.

142. (a) If β is the wedge at 44 in C, then $\beta\tau$ is the wedge at 34 in the reflection. And $\alpha_4 = \beta\rho$. So $\beta\tau = \alpha_4\rho\rho\rho\tau = \alpha_4\tau\rho = \bar{\alpha}_4$. Continuing in this way we obtain $\bar{\alpha}_4\bar{\alpha}_3\bar{\alpha}_2\bar{\alpha}_1$.

(b) $\bar{\alpha}_2\bar{\alpha}_1\bar{\alpha}_4\bar{\alpha}_3$. (c) $\bar{\alpha}_3\bar{\alpha}_2\bar{\alpha}_1\bar{\alpha}_4$.

143. `dDdD` reflects to `cCcC`; the canonical bunch is `CcCc`, by (25).

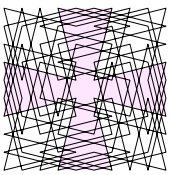
144. False. In its equivalence class {`abAB`, `bABa`, `ABab`, `BabA`}, the smallest is `ABab`.

145. It would force a 4-cycle with two edges at the nearby corner.

146. $\alpha\alpha\alpha\alpha$ for $\alpha \in$ {`B`, `C`, `D`, `E`, `F`, `G`, `H`, `I`, `J`, `K`, `w`, `y`}; $\alpha\beta\alpha\beta$ for $\alpha\beta \in$ {`AL`, `Aa`, `Al`, `Bb`, `Cd`, `Dc`, `Eh`, `Fg`, `Gf`, `He`, `Ij`, `Ji`, `Kk`, `La`, `Ll`, `al`, `wx`, `yz`}.

148. For example, the fixed point $\alpha_1\alpha_2\alpha_3\alpha_4 = \bar{\alpha}_3\bar{\alpha}_2\bar{\alpha}_1\bar{\alpha}_4$ occurs if and only if $\alpha_1 = \bar{\alpha}_3$, $\alpha_2 = \bar{\alpha}_2$, and $\alpha_4 = \bar{\alpha}_4$ ($28 \cdot 4 \cdot 4$ cases). Summing over all eight fixed points yields the answer $(28^4 + 28 + 28^2 + 28 + 28^2 + 28 \cdot 4 \cdot 4 + 28^2 + 28 \cdot 4 \cdot 4)/8$. Similarly, without '`a`', it's $(27^4 + 27 + 27^2 + 27 + 27^2 + 27 \cdot 3 \cdot 3 + 27^2 + 27 \cdot 3 \cdot 3)/8$.

149. A census based on the 28^4 possible central wedges works well, as it did for knights. (Notice that a giraffe's wedge `a` subtends the angle $\theta' = \arctan\frac{15}{8} \approx 61.93°$, which is significantly *wider* than the angle $90°-\theta'$ subtended by its wedge `c`.) As with knights, we deal with 66771 canonical bunches; but this time we exclude code `A` instead of code `a`. It turns out that 33975 of those canonical bunches — more than half! — have no solutions, often because of subtle constraints that lead to nontrivial search trees. Of the remaining



Cossack cross

multiplicities
al-'Adlī
reflection
lexicographically smallest
canonical bunches
al-'Adlī
reflection
lexicographically smallest
transposition
reflection about a diagonal
Jelliss

32796 cases, bunch `CdCd` has the fewest solutions (110); bunch `Baby` has a median number of solutions (847479); and bunch `aaaa` has the most (≈ 4.5 billion). (Bunch `aaaa`, which has multiplicity 8, also happens to be the graph G in Table 1 whose central wedges define a Cossack cross, one of which is pictured here. Bunch `CdCd` has multiplicity 4.) The total number of tours, taking multiplicities into account, is 1,018,865,516,976.

151. Now a bunch is defined by a sequence of *eight* wedge codes $\alpha_1\beta_1\alpha_2\beta_2\alpha_3\beta_3\alpha_4\beta_4$, where α_1 and β_1 are the wedges at 03 and 04; then α_2 and β_2 determine the wedges at 30 and 40 in a similar way, after we rotate the diagram 90° clockwise, etc. Therefore $\alpha_1\beta_1\alpha_2\beta_2\alpha_3\beta_3\alpha_4\beta_4$, $\alpha_2\beta_2\alpha_3\beta_3\alpha_4\beta_4\alpha_1\beta_1$, $\alpha_3\beta_3\alpha_4\beta_4\alpha_1\beta_1\alpha_2\beta_2$, $\alpha_4\beta_4\alpha_1\beta_1\alpha_2\beta_2\alpha_3\beta_3$ are equivalent bunches. (The only codes that can appear in the top row are {`A`, `B`, `C`, `E`, `G`, `I`}.)

For example, al-'Adlī's closed tour (1) belongs to bunch `EGABCAIG`, which is equivalent to `ABCAIGEG`, `CAIGEGAB`, and `IGEGABCA`, as well as to `EGEIBCAB`, `EIBCABEG`, `BCABEGEI`, and `ABEGEIBC` after reflection. These bunches all have 83,205,370 solutions; the census looks only at their canonical (lexicographically smallest) bunch, `ABCAIGEG`.

There are 210,771 canonical bunches altogether. But 29,984 of them have no solutions, usually for obvious reasons. For example, $\alpha_1\beta_2 =$ `AB` forces a 4-cycle; $\alpha_1\beta_2\alpha_3\beta_4 =$ `IIII` forces a 6-cycle; $\alpha_1\beta_2\alpha_3\beta_4 =$ `CCCC` forces a 12-cycle, for three choices of each of β_1, α_2, β_3, and α_4. Canonical bunch `CCCECECE` has a *unique* solution; and so does `CCCECCGE`! At the other extreme, `EGEGEGEG` has a whopping 3,046,049,272 solutions. The median canonical bunch, `AEEIECGC`, has 859,162. (1,676,968,941,608 solutions are visited.)

152. Again we'll have eight wedge codes $\alpha_1\beta_1\alpha_2\beta_2\alpha_3\beta_3\alpha_4\beta_4$ for the eight designated wedges. We'll base $\alpha_1\beta_1$ on the wedges at 15 and 26; then $\alpha_2\beta_2$ will define the wedges at 21 and 12, after rotating the board 90° so that $21 \mapsto 15$ and $12 \mapsto 26$; and so on. Bunch $\alpha_1\beta_1\alpha_2\beta_2\alpha_3\beta_3\alpha_4\beta_4$ will then be equivalent to bunch $\alpha_2\beta_2\alpha_3\beta_3\alpha_4\beta_4\alpha_1\beta_1$, as well as to $\bar\beta_4\bar\alpha_4\bar\beta_3\bar\alpha_3\bar\beta_2\bar\alpha_2\bar\beta_1\bar\alpha_1$ under reflection. The edges 15 — 07 — 26 are always present; therefore the possible wedges at 15 are (`D`, `F`, `i`, `K`, `w`) and the possible wedges at 26 are (`c`, `g`, `J`, `k`, `x`), in increasing order of their angles. Reflection takes `D` ↦ `c`, `F` ↦ `g`, etc.

For example, al-'Adlī's closed tour (1) belongs to bunch `KcFgFxiJ`, which is equivalent to `FgFxiJKc`, `FxiJKcFg`, and `iJKcFgFx`, as well as to `iJwgFgDk`, `wgFgDkiJ`, `FgDkiJwg`, and `DkiJwgFg` after reflection. These bunches all have 11,550,362 solutions; the census looks only at their canonical (lexicographically smallest) bunch, `DkiJwgFg`.

There are 5^8 bunches, of which 49225 are canonical. However, it's easy to see that a bunch with $\alpha_j\beta_j =$ `Kk` forces a 4-cycle; we might as well omit all such cases. That leaves us with 24^4 bunches, of which 41790 are canonical (see exercise 148).

Among those 41790, bunch `wxwxwxwx` has the fewest solutions, with only 2112; bunch `iJiJiJiJ` has the most, with 5,609,440,068; bunch `DkFgFkig` is a median, with 11,856,607. Altogether 1,692,674,826,245 solutions are visited.

154. The interconnecting steps of (9) are 2 . . 14, 16 . . 49, 51 . . 59, and 61 . . 64. Rotating the diagram by 180° shows that this is type XII.

155. Only types I, II, III, X, and XI are unchanged by transposition (reflection about a diagonal). The other eight types must be split into two subtypes: IV and IV^T, . . . , XIII and XIII^T, yielding 21 altogether. [The original 13-type classification in Fig. 124 is due to G. P. Jelliss, *The Games and Puzzles Journal* **2**, 16 (15 May 1999), 288.]

156. (357732461664, 166744766276, 483660455968, 498605611352, 333697459256, 812965778520, 1547585659448, 986042635376, 1513974300904, 1183196364192, 806039244560, 2491945752744, 2085173920272) for types (I, II, . . . , XIII).

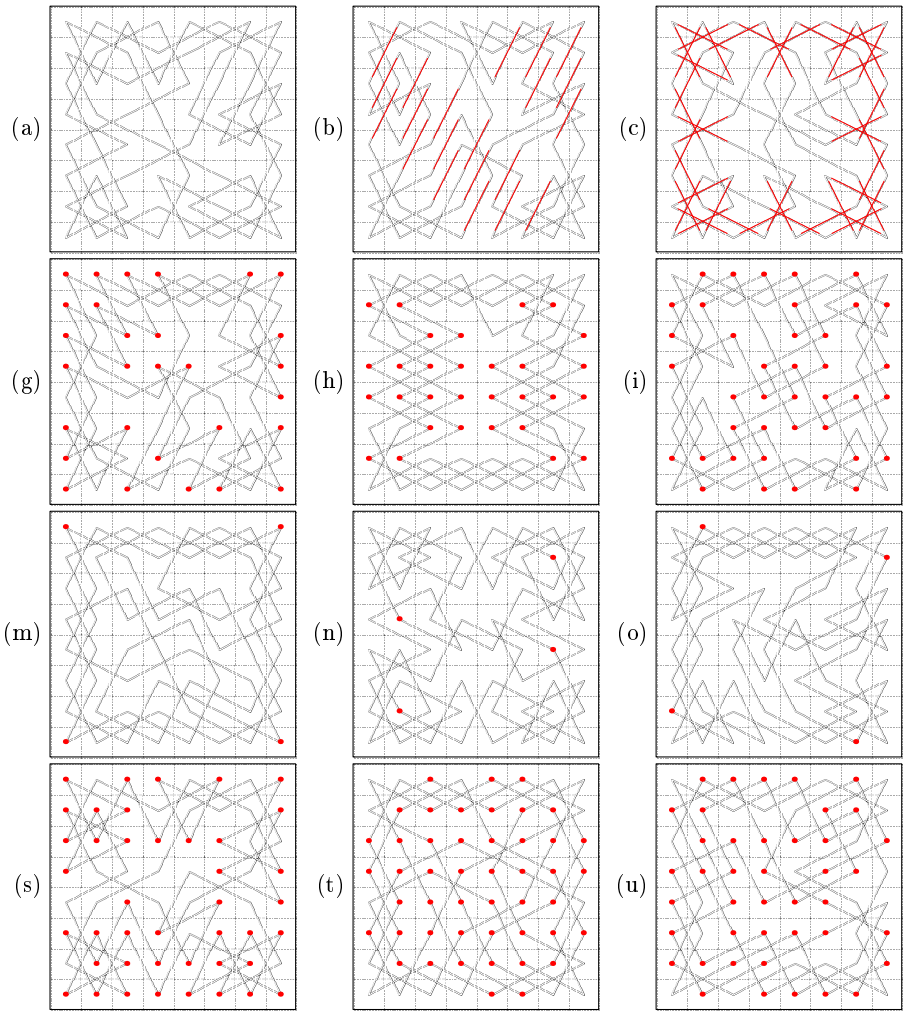


Fig. A–19. A gallery of

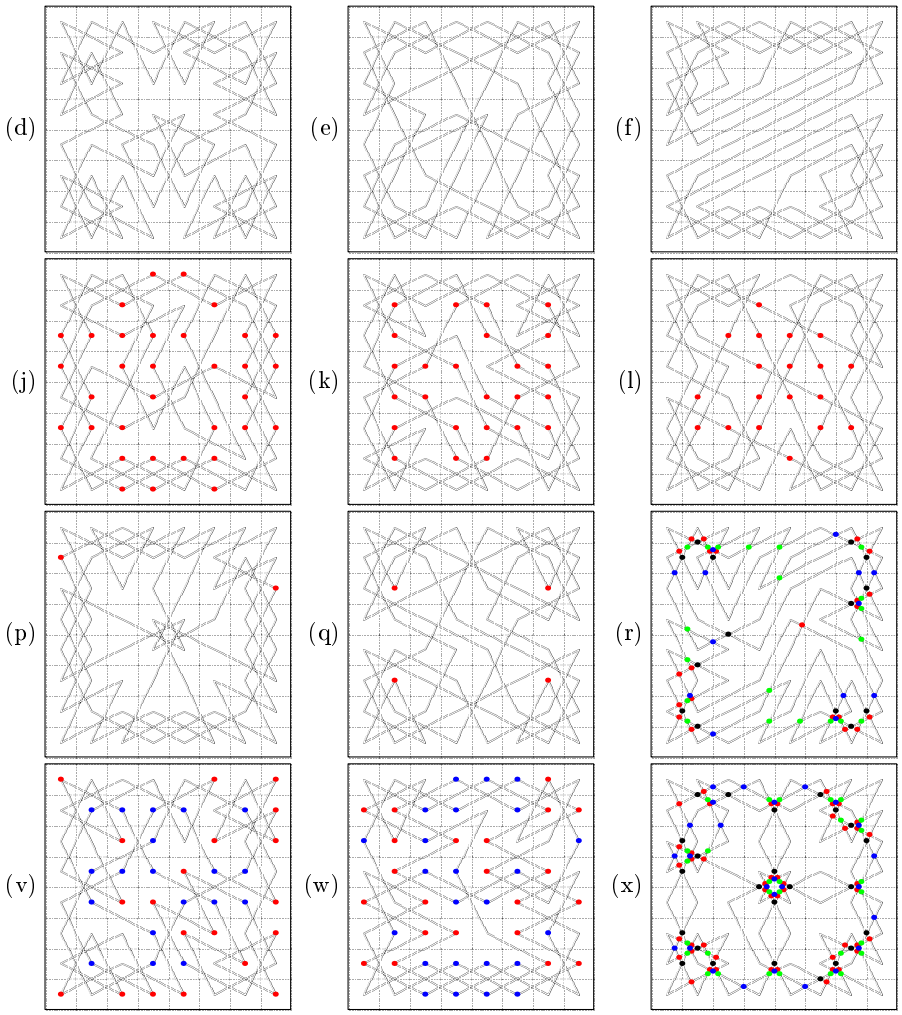
k-fold symmetry
Jelliss
magician

[These results are unexpected. Why does Type II occur less than half as often as Type I? But the totals are otherwise roughly in line with the prediction that a type with k-fold symmetry will occur about $1/k$ times as often as the unsymmetrical types XII and XIII. (The respective values of k are (8, 8, 4, 4, 4, 2, 2, 2, 2, 2, 2, 1, 1).)]

157. (a) 431,873,707,240. (b) 0! (No explanation for this lack of solutions is known.)

(c) Although six full classes force 48 of the 64 edges, there actually are 6720 solutions. For example, Fig. A–19(c) has all the edges of classes `A`, `C`, `E`, `F`, `G`, `S`.

[G. P. Jelliss devised Fig. A–19(b), whose moves of slope +2 solve (a), in *Chessics* **2**, 22 (Summer 1985), 66. He observed that a magician who memorizes a single solution to (a) can perform the following trick: A spectator places a white knight and a black knight anywhere on the board, a knight's move apart; the magician then captures the black knight with the white knight, the slow way, *after* first visiting every other square.]



exceptional knight's cycles.

158. We conduct *two* censuses, first to determine the maxima and unknown minima and then to count the extreme solutions. Since extreme solutions are rare, the second census needs to examine fewer than 2000 bunches. Most of the minima are obvious and findable by hand; Jelliss proved (surprisingly) that at least two 90° moves are needed.

Proof. Suppose there's a tour without any right-angle moves. We must make the eight moves of class `A` in exercise 157. Edge 01 — 22 is also forced; otherwise we'd have 20 — 01 — 13. Similarly, all eight edges of class `G` are forced. Then 03 — 15 is forced, because we can't have 11 — 03 — 24; we must have all of class `D`. Hence 02 — 14 (and all of class `C`) is forced. The central wedges are now determined, giving us all of class `U`. We must also have class `B`, because 13 — 32 makes a right angle. That forces class `M`, which forces class `E`. It's a nice kaleidoscopic pattern, with 8-fold symmetry; but it's not a tour! Sharper analysis shows that a single 90° angle is also impossible.

Each of the six possible angles α can occur surprisingly often in a single tour. Their maxima, (29, 30, 39, 33, 25, 19), are achieved respectively (136, 432, 48, 176, 32, 112) times among the 13 trillion solutions, and examples appear in (g), (h), (i), (j), (k), (l) of Fig. A–19. On the other hand the minima, namely (4, 0, 2, 0, 0, 0), aren't difficult to achieve, except for Jelliss's construction in Fig. A–19(p); they occur respectively (4073251792, 193895168, 1152, 316388348, 312777068, 196464725912) times.

It's also interesting to group angles together into *acute* angles ($< 90°$), *obtuse* angles ($> 90°$), and *orthogonal* angles ($90°$ or $180°$). These groups can occur as many as (42, 47, 42) times, while their minima are (4, 4, 4). (See Fig. A–19(m, n, o) and (s, t, u).) The maxima occur (56, 464, 7128) times, and the minima are also fairly rare: (28068, 4, 400624). Indeed, Fig. A–19(n) is essentially unique.

Connoisseurs also group together the *diagonal* angles $\{\theta, 180° - \theta\}$ and the *axial* angles $90° \pm \theta$, which occur at least (4, 4) and at most (39, 46) times. Those extremes, achieved in (300312, 1964, 344, 80) ways, appear in Appendix E and Fig. A–19(q, v, w).

The least and greatest sums of all angles are $52 \cdot 90° - 6\theta \approx 4458.8°$ and $84 \cdot 90° + 10\theta \approx 7928.7°$, illustrated in Fig. A–19(d, e), achievable in just 88 and 64 ways.

Historical notes: Problems 1 and 2 in Jelliss's magazine [*Chessics* **1**, 1 (Walton on Naze, March 1976), 2] asked only for the six maxima and minima; now at last we can ask and solve the more detailed questions. Only two of the maxima had been known before 2025. Prior to that, the best published constructions were (26, 38, 30, 22) for (θ, $90°$, $90°+\theta$, $180°-\theta$) [G. P. Jelliss, in *Chessics* **1**, 5 (July 1978), 4–5; *J. Recreational Math.* **27** (1995), 237], and 26 for $90° - \theta$ [J. J. Secker, in *Chessics* **1**, 7 (March 1979), 10]. Astonishingly, a tour achieving the correct value 19 for $180°$ had already been given by V. Onitiu in *The Problemist: Fairy Chess Supplement* **1**, 12 and 13 (June and August, 1932), pages 74 and 82! And H. J. R. Murray, on page 79 of an unpublished manuscript [*The Knight's Problem* (Oxford: Bodleian Library, 1942), viii + 283 pages], had found the "herringbone" tour of Fig. A–19(h), actually in another context.

159. (Solution by Filip Stappers.) The maximum number of cells that can be tarnished t times turns out to be respectively (20, 32, 46, 24, 20, 12, 6, 4) for $t = (0, 1, \ldots, 7)$. Figure A–20 exhibits champion tours that achieve those maxima, symmetrically when possible. Such winners are rare gems, especially when $1 \le t \le 5$: They occur only (9748, 16, 8, 56, 4, 28, 372348, 904604) times, respectively, among the 13 trillion possible tours. (Indeed, the solutions for $t = 2$ and $t = 4$ are *unique*, except for rotation and reflection.)

A cell that's tarnished by seven of its neighbors is called a *star*, and 4-star cycles have an interesting history. One of the first major treatises on knight's tours, Ballière de Laisement's 74-page *Essai sur les Problèmes de Situation* (Rouen, 1782), presented the 4-star tour of Fig. A–20(j) as his second example of how to construct a solution "mechanically" (pages 16–20). He also found a different 4-star tour (Planche A#5). C. F. de Jaenisch [*Traité des applications de l'analyse mathématique au jeu des échecs* **2** (1862), §96; Pl. IV, Fig. 7] presented Fig. A–20(h), the first known *symmetrical* example. And F. Hansson [*Fairy Chess Review* **6**, 111 (February 1948), solution (iv) to problem 7531] presented Fig. A–20(i), a symmetrical example with only two of the four stars adjacent to a corner. The four cells adjacent to a corner are always tarnished at least four times. It turns out that 2517414323 tours have *no* cell tarnished more than four times, among which 2213509 have *only* those four cells quadruply tarnished.

The sum of all tarnish counts is 128 in every knight's cycle; hence the mean is always exactly 2. What about the variance? *Answer:* The sum of squares is always at least 308 and at most 478 — achieved in 152 and 64 ways, such as Fig. A–20(k) and (l).

gallery of knight's tours
acute angles
obtuse angles
orthogonal angles
diagonal angles
axial angles
lateral angles, see axial
Historical notes
Secker
Onitiu
Murray
Stappers
unique
star
history
de Laisement
Laisement
de Jaenisch
Hansson
variance

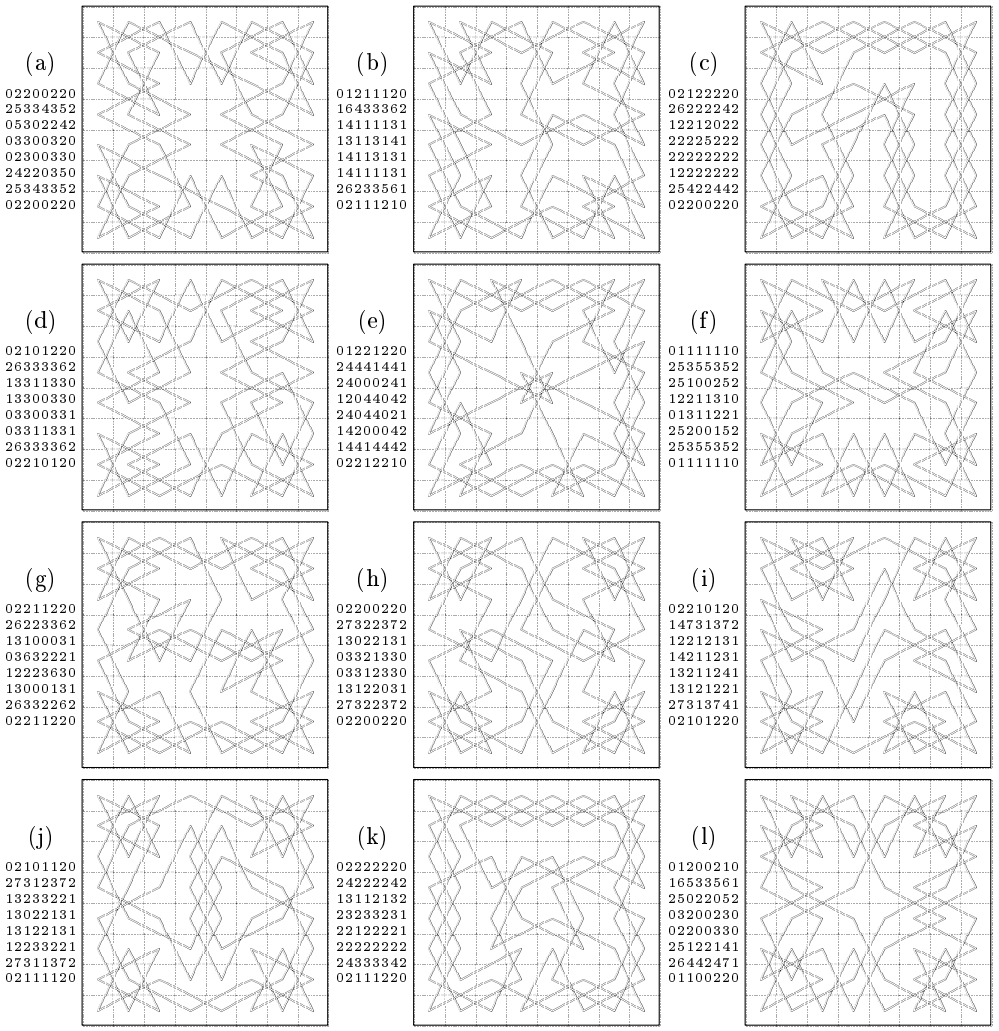


Fig. A–20. Record-breaking tarnish counts of closed 8×8 tours.

The *minimum* number of tarnish counts equal to $t = (0, 1, \ldots, 7)$ is respectively $(4, 0, 7, 0, 0, 0, 0, 0)$; and Fig. A–20 shows that each of those minima does occur. Such examples aren't very interesting except when $t = 0$ (that is, when all but the corners are tarnished) or $t = 2$ (because the lower bound 7 is a surprise). When $t > 2$ they're not at all rare, having respectively (40666596356, 80536, 960, 40696972, 26645983660, 523634871024, 4873809930916, 11539340580216) exemplars.

160. George Jelliss, in *Chessics* **2**, 19 (Autumn 1984), 25–26, exhibited a tour in which 10 moves are unintersected. He also showed that some move must be intersected at least four times. When the author asked him in 1992 about the fewest total intersections, he responded with a tour that has only 76 — but said that such a problem was definitely "a task for [your] computers." Our computers are now ready for this challenge.

Jelliss author

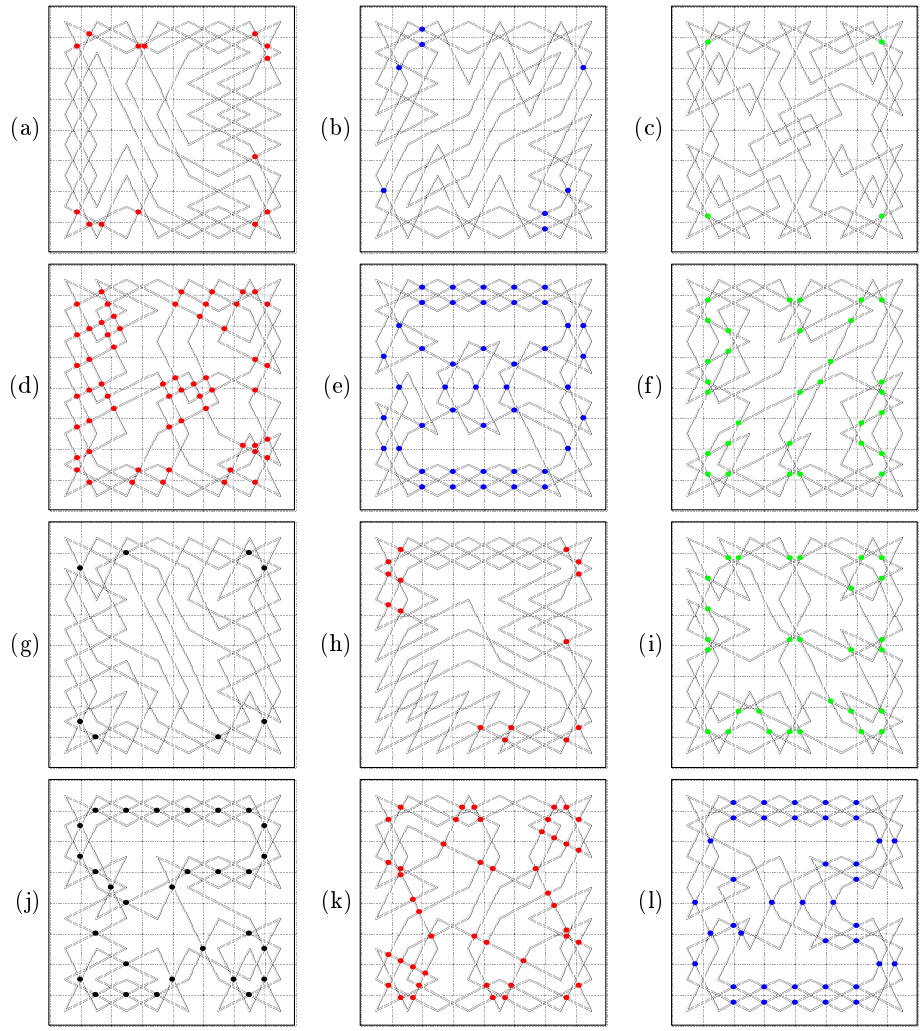


Fig. A–21. Record-breaking intersection statistics of closed 8×8 tours.

bitwise trick
trick
Stappers

Every knight move can be intersected by at most nine others, and by at most seven others in any given tour. (See *FGbook* page 497.) To speed up the census, we want a fast way to discover all of a tour's self-intersections. The obvious way does $\binom{64}{2}$ table-lookups; but there's a nice bitwise trick that needs only 64: The edges of any given tour can be represented in four 64-bit words called `NW`, `NE`, `SW`, `SE`, where each of those words has 16 bits from each of the four diagrams in exercise 157. (Edges of class `U` appear in all four of those words; edges of classes $\{\texttt{C},\texttt{D},\texttt{I},\texttt{J},\texttt{O},\texttt{P},\texttt{S},\texttt{T}\}$ appear in two of them.) Given an edge u —— v, we can assume that v is not one of the four central cells. Then if v is in the upper right quadrant, say, the number of edges that cross u —— v is $\nu(\texttt{NE} \mathbin{\&} m_v)$, for some 64-bit mask m_v with $\nu(m_v) \le 9$.

A census conducted by Filip Stappers has uncovered many surprising facts about intersections. For example, there's an essentially unique cycle that achieves 16(!) un-

intersected moves. There are $8 \cdot 5$ cycles that have only 69(!) total intersections. And—again uniquely(!)—it's possible to have a total of 126. (See Fig. A–19(f, r, x). The tour with 126 crossings had been known [*FGbook* page 495], but not its uniqueness.)

author
author
unique
Beluhov
Besa
Johnson
Mamano
Osegueda
Williams
Jelliss
mosaics
author

Every 8×8 knight's cycle has intersections of all four types, indeed at least 14 $\boxplus$'s, 8 $\diagup\!\!\diagup$'s, 4 $\diagup\!\!\diagup$'s, and 8 $\times$'s. Examples of those minima, which are attained in (376, 40, 896, 8384) ways, appear in Fig. A–21(a, b, c, g). On the other hand, Fig. A–21(d, e, f, j) exhibits remarkable (and even more rare) tours where each flavor of intersection is maximized, namely (59, 44, 31, 30) intersections, in just (16, 120, 1160, 16) ways.

Figure A–21(j) is particularly striking, because all but four of its 64 moves are half of an $\times$! (The author had conjectured, in *FGbook* page 502, that such a tour was essentially unique; this, however, is the other solution. Incidentally, N. I. Beluhov [arXiv:1310.3450 [math.CO] (2013), 7 pages] had proved that no $m \times n$ knight's tour consists entirely of $\times$ moves.)

Figure A–21(l) is perhaps even more startling: All but four of *its* moves are part of at least one $\diagup\!\!\diagup$! And all but ten of the moves in Fig. A–21(i) are part of at least one $\diagup\!\!\diagup$! Moreover, Fig. A–21(k) goes all the way: *Every move in that cycle is part of at least one* $\boxplus$ *intersection*, indeed sometimes three or four! Altogether (688, 1864, 10408) tours achieve those remarkable feats of Fig. A–21(l), (i), and (k).

Finally, Fig. A–21(h) is one of 48 cycles for which only 23 moves are part of a $\boxplus$. (Instances of the 40 and 896 cycles for which only 16 and 8 moves are part of an $\diagup\!\!\diagup$ and part of an $\diagup\!\!\diagup$, respectively, are left to the reader's imagination.)

161. J. J. Besa, T. Johnson, N. Mamano, M. C. Osegueda, and P. Williams [*Theoretical Computer Science* **902** (2022), 1–20] achieve that bound with an attractive construction, and prove that at least $4n - O(1)$ intersections are necessary.

163. Each cell of the board can be partitioned into 21 subregions, and we can compute the winding number of each subregion by choosing an appropriate point in that subregion and counting how often the tour crosses a straight line to the left of that point. (Downward counts $+1$; upward counts -1.) The area of each subregion is a multiple of $\frac{1}{120}$, so the calculation can work entirely with smallish integers.

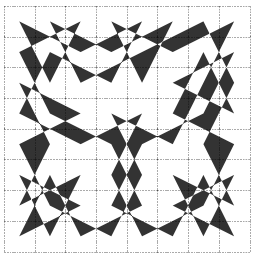
[See the online program SSHAM-WINDING-PREP. This way to represent tours by shaded regions was discovered by George Jelliss, who called them "knight's tour mosaics" and communicated his idea to the author on 26 December 1992. In that same letter he asked if the minimum shaded area could be computed. Yes, now it can!]

The fascinating extremal results are exhibited in Fig. A–22, where tours (a) and (b) attain the minimum and maximum shaded area ($\frac{1772}{120}$ and $\frac{3942}{120}$), while (c) attains the maximum *swept* area (150). All three of those solutions are *unique*, except for rotation and/or reflection. The 49 individual winding numbers at interior corners, shown below each figure, yield the total swept area when we add them up, as proved in exercise 164. Figure A–22(d) shows one of the 254,652 tours for which those 49 numbers take only two distinct values (possibly all 1 and 2). If we restrict consideration to the 129,937,524,256 tours whose swept area is zero, the min and max shaded area ($\frac{1838}{120}$ and $\frac{3828}{120}$) occur uniquely in tours (e) and (f). Tour (g) is one of 3,378,536 cases where the interior winding numbers vary over a range of ten digits (in this case -5 through $+4$). And the amazing and unique example (h) has only six nonzero interior winding numbers!

164. In fact this is true of *any* oriented polygonal cycle C whose vertices are a subset of the midpoints of square cells, provided that none of the lines between consecutive vertices goes exactly through a corner between cells. (See *The American Mathematical*

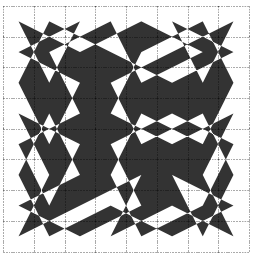
historical note

(a)



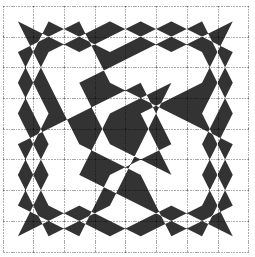
1	2	0	2	1	$\bar{1}$	1
1	0	2	0	0	0	0
$\bar{2}$	0	0	0	0	1	0
0	$\bar{1}$	0	0	0	$\bar{2}$	0
1	0	0	0	$\bar{2}$	$\bar{2}$	$\bar{1}$
0	1	0	$\bar{2}$	$\bar{2}$	$\bar{1}$	0
1	0	0	0	$\bar{2}$	0	1

(b)



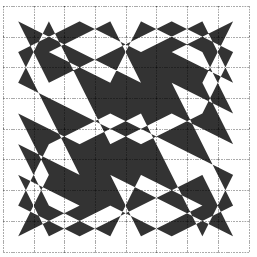
1	0	$\bar{1}$	1	1	2	1
$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	3	2	1
$\bar{1}$	$\bar{1}$	$\bar{1}$	1	1	1	1
$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$
$\bar{2}$	$\bar{3}$	$\bar{1}$	$\bar{3}$	$\bar{3}$	$\bar{3}$	$\bar{2}$
$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{3}$	$\bar{4}$	$\bar{1}$
$\bar{1}$	0	1	$\bar{1}$	$\bar{2}$	$\bar{1}$	$\bar{1}$

(c)



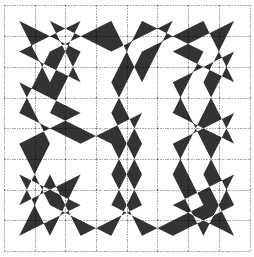
1	2	1	2	2	2	1
2	4	4	4	4	4	2
2	4	5	6	5	3	2
1	4	5	7	5	4	1
2	4	5	6	5	4	2
2	4	4	3	4	4	2
1	2	1	2	1	2	1

(d)



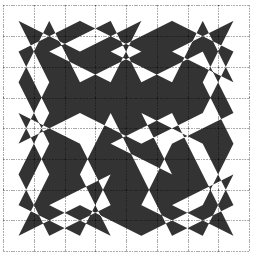
1	1	0	0	0	0	1
0	1	0	0	1	1	0
0	0	1	1	1	0	1
1	1	0	1	0	1	1
1	0	1	1	1	0	0
0	1	1	0	0	1	0
1	0	0	0	0	1	1

(e)



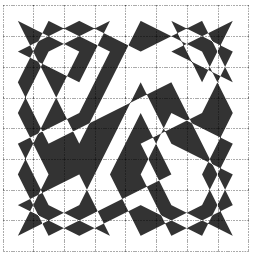
1	0	1	0	$\bar{2}$	1	1
0	2	0	$\bar{1}$	0	0	2
2	0	0	0	0	2	0
0	1	0	0	0	0	0
$\bar{1}$	0	0	0	0	$\bar{2}$	0
0	$\bar{1}$	0	0	0	$\bar{1}$	$\bar{2}$
$\bar{1}$	0	0	0	0	$\bar{1}$	$\bar{1}$

(f)



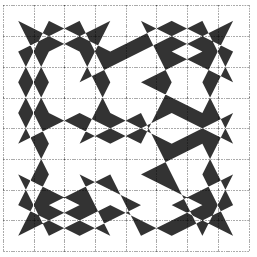
1	$\bar{1}$	0	1	1	1	$\bar{1}$
0	1	1	1	1	1	0
1	1	1	1	1	1	1
0	0	0	1	1	0	$\bar{1}$
$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$
0	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	$\bar{1}$	0
$\bar{1}$	0	1	$\bar{1}$	$\bar{1}$	0	$\bar{1}$

(g)



1	2	2	0	$\bar{1}$	2	1
1	4	2	0	0	0	2
2	2	1	0	$\bar{1}$	0	$\bar{1}$
1	0	$\bar{1}$	$\bar{2}$	$\bar{2}$	$\bar{3}$	$\bar{1}$
$\bar{2}$	$\bar{1}$	$\bar{1}$	$\bar{3}$	$\bar{5}$	$\bar{4}$	$\bar{1}$
$\bar{1}$	$\bar{2}$	$\bar{2}$	$\bar{3}$	$\bar{3}$	$\bar{4}$	$\bar{2}$
1	$\bar{2}$	$\bar{2}$	0	$\bar{1}$	$\bar{2}$	$\bar{1}$

(h)



1	0	$\bar{1}$	0	0	0	$\bar{1}$
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0
$\bar{1}$	0	0	0	0	$\bar{1}$	1

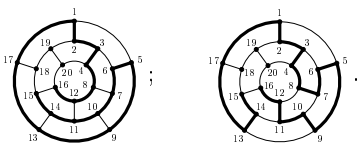
Fig. A–22. Record-breaking winding number patterns of closed 8×8 tours.

Monthly **101** (1994), 682–683; **104** (1997), 669; the proof consists of showing that such cycles can nicely be "spliced together.")

165. Yes; in fact, a census shows that there are 103,361,177,080 solutions(!). The maximum number of moves with a given slope, in an 8×8 knight's cycle, is 34; there are 4116 such tours, one of which is Fig. A–19(f). The minimum number is 2, achievable in 59124 ways (and easily findable by removing 40 edges from the knight graph).

[Parmentier's early survey of knight's tours was published by Association française pour l'avancement des sciences as a supplement to the proceedings of their Congrès de Marseille (1891), 24 pages and xi plates; Fig. 23 on plate iii exhibited an open tour with 36 moves of slope $-1/2$.]

170. Vertices 15 and 16 are endpoints; 17 is inner; 18, 19, 20 are bare. That forces a lot:



(When Algorithm E proceeds to 15-configs, these two answers yield 17-cycles of G_{17}.)

171. $\binom{d}{2}$, when vertex 1 has degree d. (They're the possible wedges of vertex 1.)

172. There aren't any, unless $n = 1$. (The only possible endpoint is 'n'.)

173. From $\bar{1}101$ and $\bar{1}11\bar{1}$ we get 101 and $11\bar{1}$. (The classes of 17-configs have three-digit names, because $\widehat{F}_{17} = (18, 19, 20)$.) From 0110 we get nothing. Class 1001 yields additional members of 101; class $101\bar{1}$ yields additional members of $11\bar{1}$; class 1212 yields an additional class, $\bar{1}11$. Each of the three 17-classes therefore has size 10. (And ultimately they'll account for the thirty 20-cycles, as in the next exercise.)

174. In the 17-class $11\bar{1}$, vertices 18 and 19 are endpoints of a subpath, while vertex 20 is inner. Joining 18 — 19 completes a cycle of G_{20}. (Similarly, $101 \mapsto_{18} 11 \mapsto_{19} C_{20}$.)

176. (a) Let k in the sum be the number of unmarked elements.

(b) $2T_{n-1}$ ways for n to be in a 1-cycle; $(n-1)T_{n-2}$ ways for it to be in a 2-cycle.

(c) There are exactly T_q possible names. When $q = 2$, for example, the five possible names $\bar{1}\bar{1}$, $\bar{1}0$, $0\bar{1}$, 00, 11 correspond naturally to the five possible marked involutions.

[See V. H. Pettersson, *Electronic J. Combinatorics* **21** (2014), #P4.7, Theorem 13. His §2.4.1 gave methods for ranking and unranking the nth order marked involutions. Marked involutions occur also in many other contexts; see OEIS A005425.]

177. The methods of Section 5.1.4 apply, with $T_n(k) = 2^{n-2k}t_n(k)$. For example, '$2(k+1)$' becomes '$8(k+1)$' and '$\frac{1}{2}(n-\sqrt{n})$' becomes '$\frac{1}{2}n-\sqrt{n}$' in Eqs. 5.1.4–(43) and (44). Appropriate changes to the subsequent formulas lead to

$$T_n = \frac{1}{\sqrt{2}} n^{n/2} e^{-n/2+2\sqrt{n}-1}\big(1 + \tfrac{5}{6}n^{-1/2} + O(n^{-3/4})\big),$$

a bit more than $\sqrt{n!}$. One can also use the saddle point method as in exercise 7.2.1.5–51.

178. The largest possible digit a_j of a marked involution is $\lfloor q/2 \rfloor$, while the largest possible digit of a `MATE` table is q.

179. (a) (Assume that error checking is unnecessary.) Set `HIT[`k`]` $\leftarrow 0$ for $1 \le k \le q/2$. Then do this for $k = 1, \ldots, q$: Set $j \leftarrow a_k$; if $j \le 0$, set `MATE[`k`]` $\leftarrow j$; otherwise if `HIT[`j`]` $= 0$, set `HIT[`j`]` $\leftarrow k$; otherwise set `MATE[HIT[`j`]]` $\leftarrow k$, `MATE[`k`]` $\leftarrow$ `HIT[`j`]`.

(b) Set $t \leftarrow 0$ and do this for $k = 1, \ldots, q$: Set $j \leftarrow$ `MATE[`k`]`; if $j \le 0$, set $a_k \leftarrow j$; otherwise if $j > k$, set $t \leftarrow t+1$, $a_j \leftarrow a_k \leftarrow t$.

181. (i) Suppose $m+1 = u'_j$, where $j > 0$. Then $(u_1, \ldots, u_{q_0})$ is a permutation of $\{u'_2, \ldots, u'_{q'}\}$, $1\tau = j$, and $q_0 = q' - 1 = |\widehat{F}_{m-1} \cap \widehat{F}_m|$. (ii) In this case $m+1 = u'_0$, $(u_2, \ldots, u_{q_0})$ is a permutation of $\{u'_2, \ldots, u'_{q'}\}$, $1\tau = 0$, and $q' = q_0$.

183. The only 0-class is '0'. Figure A–23, too big to find by hand in a reasonable time, shows all relations $\alpha \mapsto_m \beta$ that hold in K_6; we get the relations for K_5 by ignoring all class names that don't end with 0 (and by erasing the final zero when they do).

There also are cases where $\alpha \mapsto_m C_p$: $11000 \mapsto_2 C_3$; $01100 \mapsto_2 C_4$; $1100 \mapsto_3 C_4$; $0\bar{1}11 \mapsto_3 C_6$; $01\bar{1}1 \mapsto_3 C_6$; $011\bar{1} \mapsto_3 C_6$; $0110 \mapsto_3 C_5$; $1\bar{1}10 \mapsto_3 C_5$; $11\bar{1}0 \mapsto_3 C_5$; $011 \mapsto_4 C_6$; $1\bar{1}1 \mapsto_4 C_6$; $11\bar{1} \mapsto_4 C_6$; $110 \mapsto_4 C_5$; $11 \mapsto_5 C_6$. (They account for the

wedges
Pettersson
border structure, see marked involution
ranking
unranking
OEIS
saddle point method
notation $\mapsto_m$

consecutive 1s
Gessel
sparse-set manipulations
insertion sort

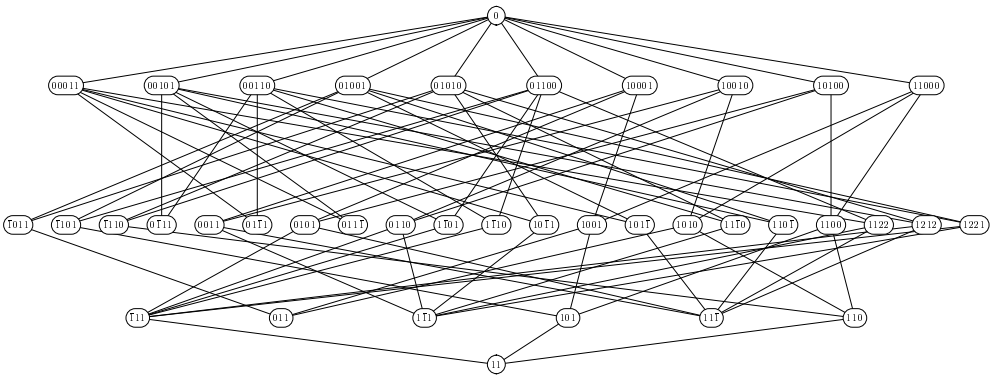


Fig. A–23. Transitions between 0-classes, ..., 4-classes when Algorithm E does K_6.

facts that the number of Hamiltonian $(3,4,5,6)$-cycles is $(1;1{+}2;2{+}2{+}2{+}6;2{+}2{+}2{+}6{+}12{+}12{+}24) = (1,3,12,60)$, in agreement with exercise 105(a).)

184. The $(m{-}1)$ class α has two forms: (i) $0\bar1^a1\bar1^b1\bar1^c0^{n-m-a-b-c-2}$, where $a,b,c\ge0$ and $p=m{+}a{+}b{+}c{+}2$. (ii) $1\bar1^a1\bar1^b0^{n-m-a-b-1}$, where $a,b\ge0$ and $p=m{+}a{+}b{+}1$.

185. Encoding vertex v by '$[v>m]$', we see that $F(m,r,s,t)=m!\,r!\,G(m,r,t)$, where $G(m,r,t)$ is the number of solutions to the following problem: "Construct t binary strings from m 0s and r 1s, where each string begins and ends with 0 and has no two consecutive 1s." Equivalently (after replacing '10' by '1'), "Construct t binary strings from $m-r$ 0s and r 1s, where each string begins with 0." Equivalently, "Construct t binary strings from $m-r-t$ 0s and r 1s, where each string might be empty." Equivalently, "Construct a binary string of length $m-t$, containing exactly r 1s, and factor it into t possibly empty substrings." Hence $G(m,r,t)=\binom{m-t}{r}\binom{m-1}{t-1}$.

[The classes in exercise 184 have $r=a{+}b{+}c$ and $t=1$, hence size $\binom{m-2}{r}(m{-}1)!\,r!$. If we fix p, the contributions to C_p from (i) are therefore $f(p)=\sum_{r=0}^{p-4}(\binom{p-r-4}{r}\times(p{-}r{-}3)!\,(r{+}2)!)/2$; from (ii) they are $g(p)=\sum_{r=0}^{p-3}(\binom{p-r-3}{r}(p{-}r{-}2)!\,(r{+}1))$. Neither f nor g seems to have a simple closed form. But the fact that $f(p)+g(p)=(p-1)!/2$ leads to the identity $\sum_{k=1}^{n-1}(n-k)!\,k!\,(\binom{n-k}{k-1}+\binom{n-k-1}{k-1})=n!$. Ira Gessel observes that the summand is $(n-(k-1))!\,(k-1)!\binom{n-k}{k-1}-(n-k)!\,k!\binom{n-k-1}{k}$, which telescopes.]

187. Set $\texttt{OFR}[k]\leftarrow\texttt{FR}[k]$ for $1\le k<q$. (At this point we always have $q=q'$.)

Set $t\leftarrow\texttt{IFR}[m{+}1]$ and $q\leftarrow q-1$. If $t<q$ (that is, if $m+1$ is in $\widehat F_{m-1}$ but isn't the last), set $x\leftarrow\texttt{FR}[q]$, $\texttt{FR}[0]\leftarrow m+1$, $\texttt{IFR}[m{+}1]\leftarrow0$, $\texttt{FR}[q]\leftarrow m$, $\texttt{IFR}[m]\leftarrow q$, $\texttt{FR}[t]\leftarrow x$, $\texttt{IFR}[x]\leftarrow t$. Otherwise set $\texttt{FR}[0]\leftarrow m+1$, $\texttt{IFR}[m{+}1]\leftarrow0$, $\texttt{FR}[t]\leftarrow m$, $\texttt{IFR}[m]\leftarrow t$; and if $t\ne q$, set $q\leftarrow q+1$ (thereby retaining the last element of $\widehat F_{m-1}$).

Set $q_0\leftarrow q$. For all vertices $v>m$ such that $m\mathrel{\text{---}}v$, do this: Set $t\leftarrow\texttt{IFR}[v]$; if $t\ge q$, set $x\leftarrow\texttt{FR}[q]$, $\texttt{FR}[q]\leftarrow v$, $\texttt{IFR}[v]\leftarrow q$, $\texttt{FR}[t]\leftarrow x$, $\texttt{IFR}[x]\leftarrow t$, $q\leftarrow q+1$.

Now do a simple insertion sort to establish (30): For $k=2,\ldots,q_0{-}1$, sortin$(k,0)$; for $k=q_0+1,\ldots,q-1$, sortin(k,q_0). Here 'sortin(k,l)' means "If $\texttt{FR}[k]<\texttt{FR}[k-1]$, do this: Set $t\leftarrow\texttt{FR}[k]$, $j\leftarrow k-1$; repeatedly set $\texttt{FR}[j+1]\leftarrow\texttt{FR}[j]$, $\texttt{IFR}[\texttt{FR}[j]]\leftarrow j+1$, and $j\leftarrow j-1$ until $j<l$ or $\texttt{FR}[j]<t$; set $\texttt{FR}[j+1]\leftarrow t$ and $\texttt{IFR}[t]\leftarrow j+1$."

To compute σ and τ we use arrays SIG and TAU, where $j\sigma=\texttt{SIG}[j+1]$ for $j\ge-1$: Set $\texttt{SIG}[0]\leftarrow-1$, $\texttt{SIG}[1]\leftarrow\texttt{SIG}[2]\leftarrow0$, $\texttt{TAU}[1]\leftarrow0$; $\texttt{SIG}[j+1]\leftarrow$

$1+$ IFR[OFR[$j-1$]] and TAU[SIG[$j+1$]] $\leftarrow j$, for $1<j\le q'$. (Step E4 uses those arrays by setting BMATE[k] $\leftarrow$ SIG[$1+$ OMATE[TAU[k]]] for $1\le k\le q_0$.)

Set $r\leftarrow 0$. Then, for all vertices $v>m$ such that m — v, set NBR[r] $\leftarrow 1+$IFR[v] and $r\leftarrow r+1$. (At this point we should also set up FMAP; see answer 193.)

BMATE
overflow
basic mate table
OEIS

189. In E3, set $t\leftarrow l\leftarrow 0$ and do this loop: Set $p'_l\leftarrow t$ and $j\leftarrow 0$; exit the loop if $l=q'$; while OMEM[t][j] $=0$, set $j\leftarrow j+1$; set $t\leftarrow$ OMEM[t][j], $l\leftarrow l+1$, $a'_l\leftarrow j-1$; repeat.

In E8, begin the following loop with $l\leftarrow q'$: Set $t\leftarrow p'_{l-1}$ and $j\leftarrow a'_l+2$; while $j<\Delta$ and OMEM[t][j] $=0$ set $j\leftarrow j+1$; if $j<\Delta$, set $a'_l\leftarrow j-1$, $t\leftarrow$ OMEM[t][j], and exit to the E3 loop; otherwise set $l\leftarrow l-1$, and repeat this loop if $l>0$.

191. Use exercise 179(b) to compute the class name $a_1\ldots a_q$. If $p=0$, set MEM[0][0] $\leftarrow$ MEM[0][1] $\leftarrow$ MEM[0][2] $\leftarrow 0$ and $p\leftarrow 1$. Set $t\leftarrow 0$, $l\leftarrow 1$, and do this loop: Set $t'\leftarrow$ MEM[t][a_l+1]; exit if $l=q$; if $t'>0$, set $t\leftarrow t'$, otherwise set MEM[p][j] $\leftarrow 0$ for $0\le j<\Delta$, MEM[t][a_l+1] $\leftarrow p$, $t\leftarrow p$, and $p\leftarrow p+1$; set $l\leftarrow l+1$ and repeat. Then if $t'>0$, add OWT[$p'_{q'}$] to WT[t']; otherwise set $w\leftarrow w+1$, MEM[t][a_q+1] $\leftarrow w$, WT[w] $\leftarrow$ OWT[$p'_{q'}$]. (In practice we should also ensure that $a_l+1<\Delta$ for $1\le l\le q$, and that p and w don't overflow memory bounds.)

192. True. Suppose OMATE[1] $=k>0$. Then $1<k\le q'$; and $u'_k=u_{k\sigma}=u'_{k\sigma\tau}$ by (36) and (37), because $1\le k\sigma\le q_0$. Hence BMATE[$k\sigma$] $=$ OMATE[$k\sigma\tau$]$\sigma=$ OMATE[k]σ by (38). And OMATE[k] $=1$. ["The mate of m in an $(m-1)$-config becomes bare in the basic mate table of the associated m-config."]

193. (This is the "heart" of Algorithm E.) Set MATE[k] $\leftarrow$ BMATE[k] for $1\le k\le q$. Do nothing if MATE[i] <0 or MATE[j] <0. (Vertices u_i and u_j must not be inner.)

Case 1: MATE[i] $=$ MATE[j] $=0$. If $i=j$, do the cycle test below. (That can happen in step E7! See exercise 192.) Otherwise set MATE[i] $\leftarrow j$, MATE[j] $\leftarrow i$, and contribute().

Case 2: MATE[i] $=0<$ MATE[j] $=k$. Set MATE[i] $\leftarrow k$, MATE[k] $\leftarrow i$, MATE[j] $\leftarrow -1$, and contribute(). (Vertex u_j becomes inner.)

Case 3: MATE[j] $=0<$ MATE[i] $=k$. Set MATE[j] $\leftarrow k$, MATE[k] $\leftarrow j$, MATE[i] $\leftarrow -1$, and contribute(). (Vertex u_i becomes inner.)

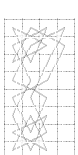
Case 4: MATE[i] $=k>0$ and MATE[j] $=l>0$. If $j=k$, do the cycle test below. Otherwise set MATE[k] $\leftarrow l$, MATE[l] $\leftarrow k$, MATE[i] $\leftarrow$ MATE[j] $\leftarrow -1$, and contribute().

Cycle test: A cycle cannot be in an m-config; but the new connection between u_i and u_j might complete an m'-cycle in $G_{m'}$. The latter occurs if and only if (i) all vertices $\le m'$ are inner; and (ii) all vertices $>m'$ are bare. To implement this test, first set MATE[i] $\leftarrow$ MATE[j] $\leftarrow -1$. Then find the smallest $k\ge 1$ such that either $k>q_0$ or MATE[k] ≥ 0 or FR[$k-1$] $\ne m+k$. If FR[$k-1$] $=m+k$, and if MATE[k'] $=0$ for $k\le k'\le q$, add OWT[$p'_{q'}$] to CYC[$m+k-1$].

(Proof sketch: Frontier vertices u_k for $q_0<k\le q$ cannot be in the cycle, because their only neighbor $\le m$ is m itself. Consequently we must have FR[$k-1$] $=m+k$ and MATE[k] $=-1$ for $1\le k\le m'-m$; also MATE[k] $=0$ for $m'-m<k\le q$.)

195. In general, the digits a_j in a class name belong to the set $\{\bar1,0,\ldots,\lfloor q/2\rfloor\}$ when the frontier has size q. So we need $\Delta\ge\lfloor q/2\rfloor+2$, unless we're lucky enough to have a graph for which the digit $\lfloor q/2\rfloor$ is never needed. The 8×32 knight graph has $q\le 17$; hence $\Delta\ge 10$ is sufficient. (And necessary, as seen in exercise 198.)

196. There are four solutions, one of which is shown. (Consequently Algorithm E sets CYC[26] $\leftarrow 4$. These are the smallest cycles that it finds. It also sets CYC[28] $\leftarrow 12$, CYC[30] $\leftarrow 212$, CYC[32] $\leftarrow 0$, CYC[34] $\leftarrow 50$, CYC[36] $\leftarrow 4525$, CYC[38] $\leftarrow 101730$, CYC[40] $\leftarrow 44202$, CYC[42] $\leftarrow 66034$, CYC[44] $\leftarrow 2408624$, CYC[46] $\leftarrow 69362264$, CYC[48] $\leftarrow 55488142$, etc.; see OEIS A383664.)



197. *board*$(p, q, 0, 0, 5, 0, 0)$ is a graph of knight moves with pq vertices named '$i\,.\,j$', and they appear in lexicographic order: `0.0`, `0.1`, ..., `0.`$(q-1)$, `1.0`, ..., $(p-1)$`.`$(q-1)$. So the frontier $\widehat{F}_{q-1}$ has $2q+1$ vertices, namely `0.`$(q-1)$ together with `1.`j and `2.`j for all j. Furthermore, all frontiers have at most $2q+1$ vertices, regardless of p.

Therefore Algorithm E fares poorly when $q > p$. Conversely, with the graph *board*$(32, 8, 0, 0, 5, 0, 0)$, we do get the desired frontiers (41) — but we must realize that the vertex for row i and column j has the SGB name '$j\,.\,i$', not '$i\,.\,j$'.

parity
Haanpää
Östergård
trie
gluing together
180° rotation
canonical notation
class names

198. (a) For $m \bmod 8 = (0, 1, \ldots, 7)$, respectively (1187898716, 411619845, 565860079, 1335885051, 1525183477, 1964090779, 2084942265, 1977893280); that's about (4.20, 1.76, 1.48, 2.90, 2.56, 3.27, 2.51, 2.37) contributions per class.

(b) Only rarely: (2, 0, 0, 4, 5, 2, 4, 2), respectively(!). For example, the $(8m+2)$-class $0\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar100000$ updates `CYC[`$8m+14$`]`; so does the $(8m+7)$-class $1\bar1\bar1\bar1\bar110000000000$, while $1\bar1\bar1\bar1\bar1\bar1\bar100000000$ updates `CYC[`$8m+16$`]`.

(c) (i) 1, occurring first in class $\bar1\bar1\bar1\bar1\bar1\bar1\bar10011\bar1\bar122$; (ii) 1, occurring first in class 1233245165788764; (iii) 655752828068, occurring first in class 1233456146788725; (iv) 51496469055038078292944, occurring first in class 1233456146788725.

(d) (i) 2320688; (ii) 225507902921136492; (iii) 17497529967689449592414967040; (iv) 13407965795037920355931071432775863398 20; all in class $0\bar1\bar1\bar1\bar1\bar1001000010$.

(e) $m = 71$, occurring first in class 12345316507684287.

(f) $m = 40$, class $0\bar1\bar1\bar1\bar1\bar1001000010$ has weight 1643851174.

(g) $\bar1\bar1\bar1\bar1\bar1\bar1\bar10\bar1\bar10\bar1\bar111$; $\bar1\bar1\bar1\bar1\bar1\bar10\bar1\bar1\bar1\bar1\bar1101$; $\bar1\bar1\bar1\bar1\bar10\bar1\bar1\bar1\bar1\bar110100$; $\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar110$-$010$; $\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar1000101$; $\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar1000\bar1011$; $\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar10\bar110011$; $\bar1\bar1\bar1\bar1\bar1\bar1\bar1\bar10\bar1\bar10\bar1101$.

199. We can begin with 8 even-parity cells, then 16 odd-parity, then 16 even-parity, ..., then 16 odd-parity, then 8 even-parity: 00, 20, 40, 60, 71, 51, 31, 11, 70, 50, 30, 10, 01, 21, 41, 61, 72, 52, 32, 12, 03, 23, 43, 63, 02, 22, 42, 62, ..., 74, 54, 34, 14, ... [See H. Haanpää and P. R. J. Östergård, *Math. Comp.* **83** (2014), 979–995.]

Note: With this ordering we can rapidly compute the total number `CYC[64]` of 8×8 knight cycles by stopping at the 24-classes, of which there are 95345608. (The trie for this class has 512961475 pointers, and the maximum weight is 79173; the frontier is $\widehat{F}_{24} = \{02, 22, 42, 62, 73, 53, 33, 13, 04, 24, 44, 64, 55, 35, 15, 75\}$.) Every 64-cycle can be obtained by "gluing together" its 24-config with the 24-config of its 180° rotation. For example, the historic tour (9) belongs to the 24-class $\bar1\bar1\bar1\bar1012331244000$, and its 180° rotation belongs to $\bar1\bar1\bar1\bar12324\bar11430000$. We can count all cycles by gluing together pairs of compatible 24-classes, summing the products of their weights; the entire computation involves only 102 gigamems! Similarly, the $(16k-8)$-classes suffice to determine `CYC[`$32k$`]`; to compute `CYC[256]` we can glue together pairs of 120-classes.

200. (a) 1231432004000000; 12342100430000000; 12314403200000000; 12344321000-000000; 12332140040000000; 12213400430000000; 11234204300000000. (The canonical notation for class names is *not* intuitive! For exercises such as this, it would be better to keep track of the endpoints of the four subpaths: If α_0 is $abcdbadc000000$, then α_1 is $bcdbadc00a000000$, and α_2 is $cdbadc00ab0000000$, etc.)

(b) $1\bar123145056040236$; $\bar112345056340126 00$; 12345056340120600; 12340452306-105006; $12303\bar112045050040$; $1202\bar1310450500403$; $101\bar12304505004023$.

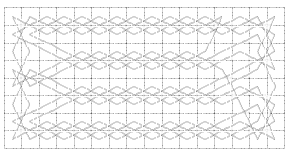
201. 1234156473762858, 1234156743267858, 1234567483812765, 1234567843218765.

202. If $\alpha_0 = a_1 \ldots a_{16}$ and $t = \max\{a_1, \ldots, a_{16}\}$, a periodic tour based on α_0 will change direction $2t$ times, because it crosses column $m/8$ in both directions, for each of t subpaths. Thus we want $t = 8$, as in exercise 201.

X cuts
recursive procedure
mini-sparse-set
sparse-set

But none of the 8-classes in answer 201 is suitable for α_0; they have no intermediate $\alpha_1, \ldots, \alpha_7$. The only way to sustain a class with $t = 8$ is to make eight horizontal X cuts (see exercise 160), and this requirement severely restricts α_0.

There *is* a suitable 16-class, namely $\alpha_0 = 1212343456567878$; and we can use it to obtain the solution shown.



Many other suitable classes exist. We might, for example, have $\alpha_0 = 1234562178654387$. But that one is difficult to reach — it arises first as a 40-class! In any case all solutions will look the same: They're mostly X cuts, except for the patterns at the very left and the very right.

203. Let OMEM now be simply an array of octabytes. We shall use a completely different way to set OMEM and OWT in step E2, after setting $q' \leftarrow q$, $p \leftarrow w \leftarrow 0$: Set $p' \leftarrow w' \leftarrow$ PACK $\leftarrow s \leftarrow 0$; here PACK is an octabyte, and s is the number of bits that have been packed into PACK. Call compress$(0, 0)$, where compress(l, p) is the following recursive procedure: If $l = q$, set OWT$[w']$ $\leftarrow$ WT$[p]$ and $w' \leftarrow w' + 1$. Otherwise set $b \leftarrow \sum_{k=0}^{\Delta-1} 2^k[\texttt{MEM}[p]\,[k] \neq 0]$; if $s + \Delta > 64$, set OMEM$[p']$ $\leftarrow$ PACK, PACK $\leftarrow b$, $s \leftarrow \Delta$, and $p' \leftarrow p' + 1$; else set PACK $\leftarrow$ PACK $+ (b \ll s)$ and $s \leftarrow s + \Delta$; then for $0 \le k < \Delta$, if MEM$[p]\,[k] \neq 0$, call compress$(l + 1,$ MEM$[p]\,[k])$. When compress$(0, 0)$ finishes, set OMEM$[p']$ $\leftarrow$ PACK. Conclude step E2 by changing FR, IFR, q_0, and q as before.

Steps E3–E8 are effectively replaced by an inverse process, which is invoked by setting $s \leftarrow p' \leftarrow w' \leftarrow 0$, PACK $\leftarrow$ OMEM[0], and calling uncompress(0). Procedure uncompress(l) sets up the address digits $a'_1 \ldots a'_{q'}$ and controls the activities as follows: If $l < q'$, set $b \leftarrow$ PACK $\gg s$, $s \leftarrow s + \Delta$; if $s + \Delta > 64$, set $p' \leftarrow p' + 1$, PACK $\leftarrow$ OMEM$[p']$, and $s \leftarrow 0$; then for $0 \le k < \Delta$, if $b \,\&\, 2^k \neq 0$, set $a_l \leftarrow k - 1$ and call uncompress$(l + 1)$. On the other hand if $l = q'$, uncompress(l) goes to step E4 (which transfers to either E5, E6, or E7). Our new algorithm no longer needs $p'_0, \ldots, p'_{q'}$; instead, we use OWT$[w']$ where the former algorithm used OWT$[p'_{q'}]$. When control reaches the former step E8, we simply set $w' \leftarrow w' + 1$, and exit from uncompress(q').

204. (a) The rightmost field '2' is needed only in nodes 1, 12, and 121. The leftmost fields, '$\bar{1}$' and '0', aren't needed in nodes 12, $\bar{1}10$, 100, 121. The other field, '1', isn't needed in nodes $\bar{1}11$, 011, 101, 121.

(b) Notice that, for example, the node for prefix $120\bar{1}32$ needs fields only for '$\bar{1}$', '0', '1', '3', '4'; and '4' is unnecessary if the key length, q, is 8 or 9; '$\bar{1}$' and '0' are unnecessary if $q = 8$ (the next codes in that case must be either '13' or '31').

A mini-sparse-set TMAP will conveniently keep track of the possibilties: Let TMS be the number of endpoint codes that have appeared once but not twice, and let TMX be the smallest positive code that hasn't yet been seen. The first TMS elements of TMAP will be the codes c for which the next trie node definitely needs a pointer field; and we'll maintain the condition ITMAP[TMAP$[k]$] $= k$ for $0 \le k <$ TMS.

The elements of MEM are now simply pointers, not nodes made up of fields.

We need a new version of the trie-building algorithm in exercise 191: Suppose a class name $a_1 \ldots a_q$ has been given. If $p = 0$, we create the trie root and initialize TMAP by setting MEM[0] $\leftarrow$ MEM[1] $\leftarrow$ MEM[2] $\leftarrow$ TMS $\leftarrow 0$, TMX $\leftarrow 1$, $p \leftarrow 3$. Then we set $t \leftarrow l \leftarrow 1$ and do this loop: If $a_l \le 0$, set $t \leftarrow t + a_l$. Otherwise if $a_l =$ TMX, set TMAP[TMS] $\leftarrow$ TMX, ITMAP[TMX] $\leftarrow$ TMS, TMS $\leftarrow$ TMS $+ 1$, TMX $\leftarrow$ TMX $+ 1$, $t \leftarrow t +$ TMS. Otherwise set TMS $\leftarrow$ TMS $- 1$, $k \leftarrow$ TMAP[TMS], $k' \leftarrow$ ITMAP$[a_l]$, TMAP$[k']$ $\leftarrow k$, ITMAP$[k]$ $\leftarrow k'$, $t \leftarrow t + 1 + k'$. Then set $t' \leftarrow$ MEM$[t]$. Exit the loop if $l = q$. Otherwise, if $t' = 0$, we will allocate d new pointers for level $l + 1$, and there are two cases. If

$l+$TMS $= q$ (tight), set $d \leftarrow$ TMS, $t' \leftarrow$ MEM$[t] \leftarrow p-1$; otherwise set $t' \leftarrow$ MEM$[t] \leftarrow p+1$ and $d \leftarrow$ TMS $+ (l +$ TMS $= q - 1?\ 3:\ 2)$. Then set MEM$[p+k] \leftarrow 0$ for $0 \le k < d$, and $p \leftarrow p + d$. Finally, with $t' > 0$, we set $t \leftarrow t'$, $l \leftarrow l + 1$, and repeat the loop.

Finish contributing as in answer 191, but change '$\texttt{MEM}[t]\,[a_q+1]$' to just '$\texttt{MEM}[t]$'.

The bit-compression scheme of exercise 203 can also be adapted to these compact tries, to create and traverse an OMEM of octabytes in step E3, with recursive procedures compress(l, p, d) and uncompress(l, d). Details are in the online program DYNAHAM.

(When this strategy is applied to the 8×32 knight's cycle problem of (41), we're lucky because the total number of pointers per m-class never reaches 2^{32}; hence the entries of MEM are just tetrabytes. There are respectively (1110667308, 1011042161, 2175000572, 2703920480, 3124552186, 3098250128, 4094603229, 4053632879) pointers, for $72 \le m \le 240$ and $m \bmod 8 = (0, 1, \ldots, 7)$; it comes to approximately (2.1, 2.1, 2.6, 2.4, 2.4, 2.3, 2.3, 2.3) pointers per node. And the max OMEM size per class is about 65 million octabytes. So the RAM requirement goes down from 256 to about 112.5 gigabytes, mostly for WT and OWT. The total runtime increases to about 72.2 teramems.)

205. (The simpler problem for n fixed at 1 should presumably be solved first, with its asymptotics when $q \to \infty$. The average number of trie nodes (distinct prefixes of n random keys) is also of interest.)

206. Let h be a hash function that takes partial keys $a_1 \ldots a_l$ for $0 \le l < q$ and $a_j \ge \bar{1}$ into $[0\,..\,8]$, say, and use nine trielike structures stored in MEM[0], MEM[1], ..., MEM[8], WT[0], WT[1], ..., WT[8]. To find the weight for a given key $a_1 \ldots a_q$, start with $t \leftarrow l \leftarrow 0$ and do this loop while $l < q$: Set $t \leftarrow \texttt{MEM}[h(a_1 \ldots a_l)]\,[t]\,[a_{l+1}+1]$, $l \leftarrow l + 1$. The desired weight is then $\texttt{WT}[h(a_1 \ldots a_q)]\,[\texttt{t}]$.

With high probability the set of needed prefixes $a_1 \ldots a_l$ with $h(a_1 \ldots a_l) = k$ will be approximately $2^{35}/9$, for $0 \le k \le 8$. And $2^{35}/9$ is comfortably less than 2^{32}.

207. The new vertex ∞, mentioned in the text, is considered to be vertex $n+1$, adjacent to all previous vertices. We put it into the first position of every extended frontier; for example, the extended frontiers (31) become $\widehat{F}_0 = (21, 1)$, $\widehat{F}_1 = (21, 2, 5, 17)$, $\widehat{F}_2 = (21, 3, 5, 17, 19)$, $\widehat{F}_3 = (21, 4, 5, 17, 19, 6)$, ..., $\widehat{F}_{19} = (21, 20)$, with $u_1 = n + 1$ and $u_2 = m + 1$. Thus we have

E1$^+$. [Initialize.] Set PATH$[m] \leftarrow 0$ (which is a "bignum"), for $2 \le m \le n$. Set FR$[0] \leftarrow n + 1$, IFR$[n + 1] \leftarrow 0$, and FR$[k] \leftarrow$ IFR$[k] \leftarrow k$ for $1 \le k \le n$. Set MEM$[0]\,[j] \leftarrow$ MEM$[1]\,[j] \leftarrow$ OMEM$[0]\,[j] \leftarrow 0$ for $0 \le j < \Delta$. Also set $m \leftarrow 1$, $q \leftarrow 2$, MEM$[0]\,[1] \leftarrow 1$, MEM$[1]\,[1] \leftarrow 1$, WT$[1] \leftarrow 1$ (a "bignum"), NBR$[0] \leftarrow 1$.

The other steps, similarly, have only minor alterations: In answer 187, change '$\texttt{FR}[0] \leftarrow m + 1$, $\texttt{IFR}[m+1] \leftarrow 0$' to '$\texttt{FR}[1] \leftarrow m + 1$, $\texttt{IFR}[m+1] \leftarrow 1$'; also '$\text{sortin}(k, 0)$' becomes '$\text{sortin}(k, 1)$'. To compute σ and τ, set SIG$[0] \leftarrow -1$, SIG$[1] \leftarrow$ TAU$[2] \leftarrow 0$, SIG$[2] \leftarrow$ TAU$[1] \leftarrow 1$; SIG$[j + 1] \leftarrow 1 +$ IFR[OFR$[j-1]]$ and TAU[SIG$[j + 1]] \leftarrow j$, for $2 < j \le q'$. And the final paragraph of answer 187 now begins with $r \leftarrow 1$.

Steps E4$^+$ and E7$^+$ use OMATE[2] instead of OMATE[1].

Finally, the cycle test of answer 193 begins by setting $\texttt{MATE}[i] \leftarrow \texttt{MATE}[j] \leftarrow -1$. It does nothing if MATE$[1] \ge 0$ (that is, if ∞ isn't inner). Otherwise it finds the smallest $k \ge 2$ such that either $k > q_0$ or MATE$[k] \ge 0$ or FR$[k-1] \ne m+k-1$. If FR$[k-1] = m+k-1$, and if MATE$[k'] = 0$ for $k \le k' \le q$, it adds OWT$[p'_{q'}]$ to PATH$[m+k-2]$.

(We could also make the cycle test update CYC when MATE$[1] = 0$ (∞ is bare).)

208. (a) One 5-cycle ($0 — 1 — 2 — 3 — 4 — 0$); two 9-cycles ($0 — 1 — 1' — 3' — 0' — 2' — 2 — 3 — 4 — 0$ and $0 — 0' — 2' — 2 — 1 — 1' — 3' — 3 — 4 — 0$).

online
tetrabyte: A 4-byte (32-bit) quantity.
hash function
∞
bignum

[The Petersen graph is known to be the smallest "hypohamiltonian graph," as in exercise 95(c). See D. A. Holton and J. Sheehan, *The Petersen Graph* (1993), Chapter 7.]

(b) (1, 1, 1, 5, 2, 1, 4, 30, 120) for $m = (2, 3, 4, 5, 6, 7, 8, 9, 10)$. [The 7-path is $0' — 0 — 4 — 3 — 2 — 1 — 1'$.]

hypohamiltonian graph
Holton
Sheehan

209. Since each frontier gains an element, we need $\Delta = 11$ to handle the frontiers of size 18 (see exercise 195). And the number P of nonlief trie nodes will now be roughly an order of magnitude larger than before; hence P will exceed 2^{32}, and each address of a trie or lief node must now be a 64-bit integer. (We were lucky in (40) to have $P \le 1798400809$, comfortably less than 2^{31}.) So trie nodes will now occupy 88 bytes, not 40.

The maximum P_m during the computation of (44) turned out to be approximately 15.5 billion, and the maximum class size C_m was about 7.9 billion. Hence RAM usage per trie was about 1.7 terabytes. However, the compression scheme of exercise 203 reduces the total to 1.9 terabytes for both tries; adding also exercise 204 yields 0.96 TB!

compression

210. In the case $m = 3$, an analogous proof is on page 532 of *FGbook*.

211. Yale → Harvard → Brown → Columbia → Princeton → Penn → Dartmouth → Cornell → Yale. (If Penn had beaten Brown, this would've been the *only* such cycle.)

Camion

212. False: Consider $a \longrightarrow b \longrightarrow c \longrightarrow a$, $x \longrightarrow y \longrightarrow z \longrightarrow x$, $\{a, b, c\} \longrightarrow \{x, y, z\}$. [See P. Camion, *Comptes Rendus Acad. Sci.* **249** (Paris, 1959), 2151–2152.]

Fischetti
Salvagnin
Ivy League
Patriot League
integer programming

213. (Solution by M. Fischetti and D. Salvagnin.) A Hamiltonian cycle is impossible because the 14 teams in the Ivy League plus the Patriot League (Bucknell, Colgate, Fordham, Holy Cross, Lafayette, Lehigh) dominate only 13 teams (all but Holy Cross).

But here's a Hamiltonian path, found by integer programming:

Pittsburgh → Louisville → San Jose State → Washington → Arizona State → Baylor → Texas Tech → Arkansas → Tulsa → New Mexico State → UTEP → Air Force → Hawaii → Brigham Young → Miami → Syracuse → Michigan State → Rutgers → Boston College → Navy → Akron → Fullerton → Long Beach → UNLV → Pacific → Utah State → Fresno → Utah → Colorado State → Oregon → Oregon State → Arizona → UCLA → Stanford → Notre Dame → Colorado → Oklahoma → Kansas → Oklahoma State → Iowa State → Missouri → Texas Christian → Southern Methodist → Rice → Houston → Texas A&M → Texas → Penn State → Florida State → Florida → Mississippi State → Memphis State → Tulane → Louisiana State → Miami of Ohio → Eastern Michigan → Kent State → Ohio University → Toledo → Western Michigan → Bowling Green → Cincinnati → West Virginia → Virginia Tech → East Carolina → Temple → Wisconsin → Ball State → Central Michigan → Kentucky → North Carolina → Maryland → Louisiana Tech → Southwestern Louisiana → Northern Illinois → Kansas State → New Mexico → San Diego State → Wyoming → Washington State → California → Southern California → Ohio State → Indiana → Auburn → Southern Miss → North Carolina State → Georgia Tech → Nebraska → Minnesota → Iowa → Purdue → Northwestern → Illinois → Michigan → Mississippi → Georgia → Alabama → Tennessee → Virginia → Clemson → South Carolina → Duke → Wake Forest → Vanderbilt → Army → Holy Cross → Fordham → Harvard → Cornell → Lafayette → Penn → Dartmouth → Yale → Brown → Columbia → Princeton → Bucknell → Lehigh → Colgate.

(Integer programming systems have been highly tuned for the traveling salesrep problem, of which this is a special case. By contrast, Algorithm B is hopelessly inefficient when given `football-tol10.gb`; it needs a good way to reject bad partial solutions.)

traveling salesrep problem

215. The 145152 solutions are found in 69125228 mems (476.3 mems per cycle).

216. We may assume by symmetry that $k = 1$. The 2^n-cycles are $0\alpha_0 \longrightarrow 1\alpha_0 \longrightarrow 1\alpha_1 \longrightarrow 0\alpha_1 \longrightarrow \cdots \longrightarrow 1\alpha_{2^{n-1}-1} \longrightarrow 0\alpha_{2^{n-1}-1} \longrightarrow 0\alpha_{2^{n-1}} \longrightarrow 0\alpha_0$, where $(\alpha_0\alpha_1\ldots\alpha_{2^{n-1}-1})$ is an $(n-1)$-bit Gray cycle. (See Section 7.2.1.1.)

Gray cycle

217. The lexicographically smallest of 16 solutions is $1234 \longrightarrow 1324 \longrightarrow 1342 \longrightarrow 1432 \longrightarrow 1423 \longrightarrow 1243 \longrightarrow 2143 \longrightarrow 2413 \longrightarrow 2431 \longrightarrow 2341 \longrightarrow 3241 \longrightarrow 3421 \longrightarrow 4321 \longrightarrow 4231 \longrightarrow$

4213 ⟶ 4123 ⟶ 4132 ⟶ 4312 ⟶ 3412 ⟶ 3142 ⟶ 3124 ⟶ 3214 ⟶ 2314 ⟶ 2134 ⟶ 1234. (Notice that every other step swaps the *middle* elements. This is dramatically different from the sequence 7.2.1.2–(3) of "plain changes"!)

plain changes
Dillon
Farrell
Rodgers
puzzle
automorphisms
OEIS

218. (a) Twelve steps 1234 ⟶ 2134, ..., 4321 ⟶ 3421 are forced (one for each even permutation). The other twelve steps are binary choices that form four groups of three: $P_1 \iff 1243 \longrightarrow 1423 \iff 1324 \longrightarrow 1234 \iff 1432 \longrightarrow 1342$; $P_2 \iff 2134 \longrightarrow 2314 \iff 2341 \longrightarrow 2431 \iff 2413 \longrightarrow 2143$; $P_3 \iff 3142 \longrightarrow 3412 \iff 3214 \longrightarrow 3124 \iff 3421 \longrightarrow 3241$; $P_4 \iff 4123 \longrightarrow 4213 \iff 4231 \longrightarrow 4321 \iff 4312 \longrightarrow 4132$.

If $i \neq j$ we must have $P_i \vee P_j$; otherwise there's a 4-cycle. (For example, $\neg P_1$ and $\neg P_2$ implies 1234 ⟶ 2134 ⟶ 2143 ⟶ 1243 ⟶ 1234.) Hence if, say $\neg P_1$, we must have $P_2 \wedge P_3 \wedge P_4$. But then 4321 ⟶ 3421 ⟶ 3241 ⟶ 2341 ⟶ 2431 ⟶ 4231 ⟶ 4321.

(b) (i) 4144380 cycles, found in 6.7 Gμ; (ii) 6364081880 cycles, found in 6.1 Tμ.

220. (a) It appears to be difficult to find any of them without computer help. The lexicographically least is ($\mathtt{B}_{00}$ $\mathtt{K}_{22}$ $\mathtt{B}_{11}$ $\mathtt{N}_{33}$ $\mathtt{K}_{41}$ $\mathtt{R}_{30}$ $\mathtt{K}_{32}$ $\mathtt{N}_{21}$ $\mathtt{B}_{02}$ $\mathtt{R}_{20}$ $\mathtt{N}_{23}$ $\mathtt{B}_{04}$ $\mathtt{R}_{40}$ $\mathtt{Q}_{44}$ $\mathtt{K}_{43}$ $\mathtt{Q}_{34}$ $\mathtt{R}_{01}$ $\mathtt{Q}_{03}$ $\mathtt{B}_{13}$ $\mathtt{Q}_{24}$ $\mathtt{K}_{42}$ $\mathtt{N}_{31}$ $\mathtt{R}_{10}$ $\mathtt{Q}_{14}$ $\mathtt{N}_{12}$ $\mathtt{B}_{00}$); the lexicographically greatest is ($\mathtt{B}_{00}$ $\mathtt{Q}_{44}$ $\mathtt{R}_{40}$ $\mathtt{K}_{43}$ $\mathtt{Q}_{34}$ $\mathtt{R}_{01}$ $\mathtt{Q}_{03}$ $\mathtt{N}_{23}$ $\mathtt{K}_{42}$ $\mathtt{N}_{33}$ $\mathtt{Q}_{14}$ $\mathtt{K}_{41}$ $\mathtt{R}_{30}$ $\mathtt{K}_{32}$ $\mathtt{N}_{21}$ $\mathtt{B}_{02}$ $\mathtt{Q}_{24}$ $\mathtt{B}_{13}$ $\mathtt{N}_{31}$ $\mathtt{R}_{10}$ $\mathtt{N}_{12}$ $\mathtt{B}_{04}$ $\mathtt{K}_{22}$ $\mathtt{B}_{11}$ $\mathtt{R}_{20}$ $\mathtt{B}_{00}$).

(b) Again, solution by hand doesn't seem easy, although Algorithm B's search tree has only 99 nodes: ($\mathtt{B}_{00}$ $\mathtt{N}_{44}$ $\mathtt{Q}_{23}$ $\mathtt{R}_{14}$ $\mathtt{N}_{13}$ $\mathtt{B}_{34}$ $\mathtt{Q}_{12}$ $\mathtt{N}_{03}$ $\mathtt{R}_{24}$ $\mathtt{Q}_{04}$ $\mathtt{K}_{40}$ $\mathtt{Q}_{30}$ $\mathtt{K}_{41}$ $\mathtt{B}_{42}$ $\mathtt{R}_{33}$ $\mathtt{B}_{43}$ $\mathtt{K}_{21}$ $\mathtt{B}_{10}$ $\mathtt{N}_{01}$ $\mathtt{K}_{20}$ $\mathtt{N}_{11}$ $\mathtt{R}_{32}$ $\mathtt{K}_{31}$ $\mathtt{R}_{22}$ $\mathtt{Q}_{02}$ $\mathtt{B}_{00}$). Only $\mathtt{Q}_{04} \longrightarrow \mathtt{K}_{40} \longrightarrow \mathtt{Q}_{30}$ is forced.

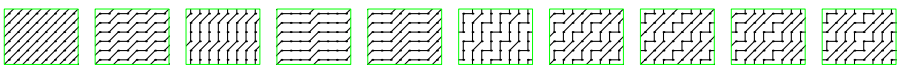
[This exercise was inspired by a puzzle game introduced by D. S. Dillon, J. Farrell, and T. Rodgers, *Homage to a Pied Puzzler* (A K Peters, 2009), 125–128.]

Stappers has also constructed examples with pieces belonging to two players, uppercase and lowercase; in this variant, each move must capture an opponent's piece. A nice two-player puzzle with three pieces of each kind is shown.

$$\begin{array}{cccccc}\mathtt{n}_{00} & \mathtt{q}_{01} & \mathtt{B}_{02} & \mathtt{b}_{03} & \mathtt{K}_{04} & \mathtt{k}_{05}\\ \mathtt{N}_{10} & \mathtt{q}_{11} & \mathtt{q}_{12} & \mathtt{N}_{13} & \mathtt{B}_{14} & \mathtt{N}_{15}\\ \mathtt{K}_{20} & \mathtt{R}_{21} & \mathtt{k}_{22} & \mathtt{R}_{23} & \mathtt{K}_{24} & \mathtt{Q}_{25}\\ \mathtt{Q}_{30} & \mathtt{r}_{31} & \mathtt{b}_{32} & \mathtt{n}_{33} & \mathtt{B}_{34} & \mathtt{r}_{35}\\ \mathtt{R}_{40} & \mathtt{Q}_{41} & \mathtt{r}_{42} & \mathtt{n}_{43} & \mathtt{k}_{44} & \mathtt{b}_{45}\end{array}$$

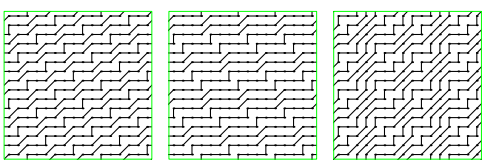
221. For example, Algorithm B needs no backtracking to solve this gem: $\begin{array}{cccc}\mathtt{B}_{00} & \mathtt{Q}_{01} & \mathtt{R}_{02} & \mathtt{B}_{03}\\ \mathtt{K}_{10} & \mathtt{K}_{11} & \mathtt{K}_{12} & \mathtt{K}_{13}\\ \mathtt{N}_{20} & \mathtt{N}_{21} & \mathtt{N}_{22} & \mathtt{N}_{23}\\ \mathtt{N}_{30} & \mathtt{N}_{31} & \mathtt{N}_{32} & \mathtt{N}_{33}\end{array}$.

222. (a) Each Hamiltonian path of $NE(m,n)$ has potentially mn automorphisms, by adding a constant $(\bmod m, \bmod n)$ to each entry; or $2mn$, when $m = n$, by optionally interchanging north and east. The 74 solutions for $m = 3$ and $n = 4$ divide into $1\cdot1 + 2\cdot1 + 3\cdot1 + 4\cdot2 + 12\cdot5$, where the ten essentially different solutions



have respectively (12, 6, 4, 3, 3, 1, 1, 1, 1, 1) automorphisms.

(b) The answer $X_{m,n}$ satisfies $X_{m,n} = X_{n,m}$, so we list $(X_{m,2}, \ldots, X_{m,m})$ for $2 \le m \le 8$: (6; 12, 18; 24, 74, 272; 48, 192, 560, 3900; 96, 408, 3596, 29017, 129024; 192, 1372, 17928, 137301, 1204399, 11050620; 384, 3834, 54412, 472684, 6880196, 58009240, 1211010880). Here are some 8×8 examples with 8, 4, and 2 automorphisms:



the total 1211010880 breaks down into $16\cdot32 + 32\cdot24 + 64\cdot1323 + 128\cdot9460351$.

(Clearly $X_{2,n} = 3\cdot2^{n-1}$. See OEIS A391478 for $X_{3,n}$.)

de Bruijn cycle
equivalence classes
automorphisms
Li
even
necklaces
bracelets
Burnside's Lemma
collinear

223. (a) True: $x_1\ldots x_{n-1}x_n$ is preceded by $x_nx_1\ldots x_{n-1}$ and $x_n^-x_1\ldots x_{n-1}$.

(b) Every Hamiltonian cycle of $SB(m,n)$ corresponds naturally to an m-ary de Bruijn cycle (see 7.2.1.1–(54)). The correspondence is in fact one-to-one when $m=2$.

(c) $00\longrightarrow01\longrightarrow11\longrightarrow12\longrightarrow\cdots\longrightarrow(m{-}1)(m{-}1)\longrightarrow(m{-}1)0\longrightarrow00$ is forced.

(d) (000100201202210211011121222) is lexicographically least, where this notation means that $000\longrightarrow001\longrightarrow010\longrightarrow\cdots\longrightarrow122\longrightarrow222\longrightarrow220\longrightarrow200\longrightarrow000$.

(e) If $x_1x_2\ldots x_n\longrightarrow x_2\ldots x_nx_1$ is in C then $x_1^-x_2\ldots x_n\longrightarrow x_2\ldots x_nx_1$ is not in C, hence $x_1^-x_2\ldots x_n\longrightarrow x_2\ldots x_nx_1^-$ is in C.

(f) Using Algorithm B we find (2, 12, 88, 7510, 675714, 459086712). (If C is any cycle, we obtain another by adding 1 to each digit, mod m. We can also go from $x_1\ldots x_n\longrightarrow y_1\ldots y_n$ to $\bar y_n\ldots\bar y_1\longrightarrow\bar x_n\ldots\bar x_1$. Thus we obtain equivalence classes whose size is a divisor of $2m$. It can be shown that no C has the property that $x_1x_2\ldots x_n\longrightarrow x_2\ldots x_nx_1\iff x_1^+x_2^+\ldots x_n^+\longrightarrow x_2^+\ldots x_n^+x_1^+$. But nontrivial automorphisms do exist; for example, when $m=4$ the cycle

(000100200310320331332303013023120121232202213210211310111222333)

is unchanged if we add 2 to each entry, mod 4, then reverse and complement. In this sense there are $4\cdot8+8\cdot7$ solutions for $m=4$, and $6\cdot275+12\cdot56172$ for $m=6$.)

224. (Solution by Shisheng Li.) A Hamiltonian cycle is a cyclic permutation ϕ of the n-tuples such that $(x\alpha)\phi=\alpha f(x,\alpha)$ for each $(n{-}1)$-tuple α, where $f(x,\alpha)=(x+[\alpha\notin S])\bmod m$ and S is defined in exercise 223(e). We can write $\phi=\sigma\psi$, where $(x\alpha)\sigma=\alpha x$ and $(\alpha y)\psi=\alpha f(y,\alpha)$. Since m^n is odd, a permutation is even if and only if it has an odd number of cycles. Hence ϕ is even; and so is ψ, whose cycles all have length 1 or m. Consequently $N_k(n)$, the number of cycles of σ, must be odd.

The cycles of σ are the m-ary *necklaces* of length n, namely the equivalence classes of m-ary n-tuples under cyclic rotation. If we also allow equivalence under reversal, we obtain the m-ary *bracelets* of length n, of which there are $N_k(n)/2+(m+1)m^{n/2}$ by "Burnside's Lemma." Therefore $N_k(n)$ is even. Contradiction.

225. More generally, construct one in $SB(m,n)$ for all $m>1$ and odd $n>2$.

227. $(i{-}p)(j'{-}q)>(i'{-}p)(j{-}q)$. (Equality occurs when the three points are collinear.)

228. Distances between crossings aren't preserved under rotation. For example, the north-pointing line has nearly equal distances $\{8,8,9,9,9,10,11\}$, but the distances for the south-pointing line are $\{7,7,9,9,10,11,11\}$. The distances for lines pointing east or west are $\{6,7,8,8,11,12,12\}$.

229. (a) Yes, if and only if m and n are sufficiently large and at least one of them is odd. (If $m+n$ is odd, one knight move actually *straddles* the pivot.)

(b) The in-degree of $(0,0)$ is zero when $n\ge2m-1$.

(c) Examine carefully the ways that a knight can cross the plumb-line.

(d) To count coils, give the length 1 to arcs that cross the plumb-line, otherwise length 0; modify Algorithm B so that it reports the total length of each cycle. When $m<6$ there's just one solution: $W_{3,3}^{(3)}=1$. When $m=6$, we have $W_{6,6}^{(5)}=2$, $W_{6,7}^{(5)}=18=2{\cdot}1{+}4{\cdot}4$, $W_{6,8}^{(5)}=72=2{\cdot}2{+}4{\cdot}17$, $W_{6,9}^{(5)}=22=2{\cdot}1{+}4{\cdot}5$, $W_{6,10}^{(5)}=32=2{\cdot}2{+}4{\cdot}7$, and all the tours have 5 coils. When $m=7$ the nonzero cases are $W_{7,7}^{(5)}=142=2\cdot1+4\cdot1+8\cdot17$; $W_{7,8}^{(7)}=8=4\cdot2$; $W_{7,10}^{(5)}=20=4\cdot5$; $W_{7,10}^{(7)}=48=2\cdot4+4\cdot10$. When $m=8$, $W_{8,8}^{(7)}=1120$; $W_{8,9}^{(7)}=324$; and $W_{8,10}^{(c)}=(4,1340,57784)$ for $c=(5,6,7)$. When $m=9$, $W_{9,9}^{(7)}=146=2\cdot1+4\cdot2+8\cdot17$; $W_{9,10}^{(c)}=(32924,499220,3070788)$ for $c=(7,8,9)$. And when $m=10$, $W_{10,10}^{(c)}=(226436,5196594,72217878)$ for $c=(7,8,9)$ (found in

121 Gμ). Examples of highly symmetric tours appear in Fig. A–24. (Incidentally, $W^{(c)}_{11,11} = (5936420, 436960600, 23419337498, 12215200)$ for $c = (7, 8, 9, 10)$.)

reflection

230. True. (For example, take $p = q = 0$ and $(i, j) \leftrightarrow (i', j')$ in answer 227.) The same is true of left-right or top-bottom reflection. (Hence, as in the undirected case, the 8×8 tours form equivalence classes of sizes 4 and 8; we have $1120 = 4 \cdot 4 + 8 \cdot 138$.)

Beluhov

integer programming problem

cycle covers

Li

90° symmetry

231. (Solution by N. Beluhov.) To get c coils, we can set up an integer programming problem that finds *cycle covers* of the digraph — subsets of the arcs for which every vertex has in-degree 1 and out-degree 1 — together with additional constraints that force every "ray" from the center to be crossed exactly c times, unless that ray runs through a vertex. When a short cycle C appears in a solution, add further constraints to eliminate all cycles that are isomorphic to C. Keep doing this until there are no more solutions, or time runs out, or a Hamiltonian cycle is found. A solution to (b) for $n = 12$, shown in Fig. A–24, was obtained in this way on the 48th trial.

On the other hand, no cycle covers for (a) were found when $n \bmod 8 = 4$ or $n \bmod 8 = 6$; indeed, Shisheng Li has proved [to appear] that such cycles do not exist. Several successes with $n/2$ coils *were* obtained for $n = 16$ and $n = 18$, including the spectacular 18×18 example with 90° symmetry that appears in Fig. A–24. (In such tours the knight's polar angle must increase slowly.)

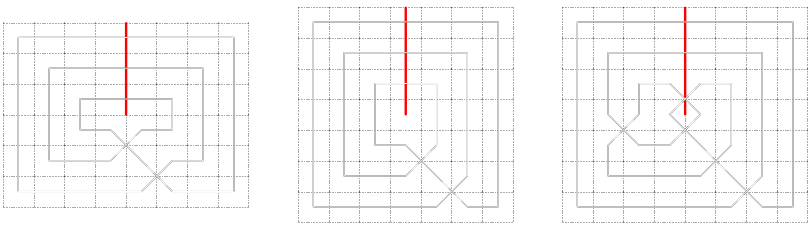
Stappers

braids

232. (Solution by N. Beluhov and F. Stappers.) If $n \geq 28$ is a multiple of 4, it's possible to start with a cycle that consists of n disjoint cycles that form $n/4$ "braids," then to stitch them together cleverly while preserving counterclockwise motion. The 32×32 example in Fig. A–24 exhibits the general pattern.

Beluhov

235. (a) The following constructions are readily generalized to show in fact that $X_{2m,2n} > 0$, $X_{2m+1,2m+1} > 0$, and $X_{2m+3,2m+4} > 0$, for $0 < m \leq n$:



(b) (Solution by N. Beluhov.) Assume that $m \leq n$. If m is odd, every coil must pass through exactly one of the vertices $\{(\frac{m-1}{2}, k) \mid \frac{n-1}{2} < k < n\}$; hence $c = \lfloor n/2 \rfloor$. If n is odd, every coil must pass through exactly one of $\{(k, \frac{n-1}{2}) \mid 0 \leq k < \frac{m-1}{2}\}$; hence $c = \lfloor m/2 \rfloor$. When both are even, every coil must pass through exactly one of $\{(k, \frac{m+n-2}{2} - k) \mid 0 \leq k < \frac{m}{2}\}$; hence $c = m/2$.

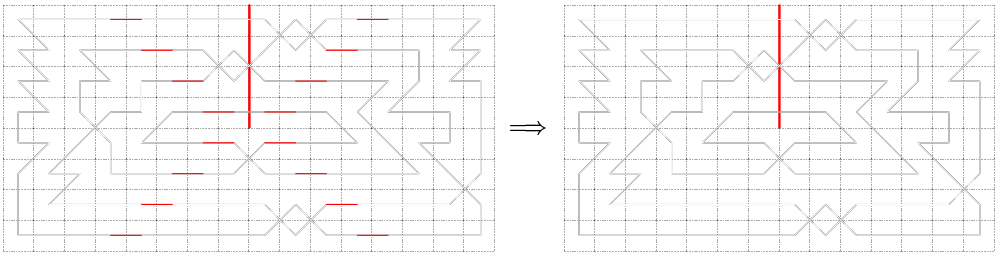
(c) This is a consequence of (b): If n is odd, we must have $m = n$. Otherwise $c = \frac{n}{2} \leq \frac{m+1}{2}$, because the plumb-line can be crossed only from $\frac{m+1}{2}$ vertices in column $\frac{n}{2}$.

(d) Let m be even. Every whirling king's tour must include the arcs $v(k, j) \longrightarrow v(k, j) - (0, 1)$ for $0 \leq k < m/2$ and $0 \leq j < \lfloor n/2 \rfloor - m/2 - 1 - k$, where $v(k, j) = (k, n - 2k - 1 - j)$. The proof is by induction on k: Vertex $v(0, j)$ is in the top row and has out-degree 1, so its move can only go west (left). When $k > 0$, the move from $v(k, j)$ can't go northeast or north to a vertex whose predecessor is already known. Nor can it go northwest, except perhaps when j has its maximum value; but that move is also impossible, because the northwest vertex must belong to a coil for smaller k.

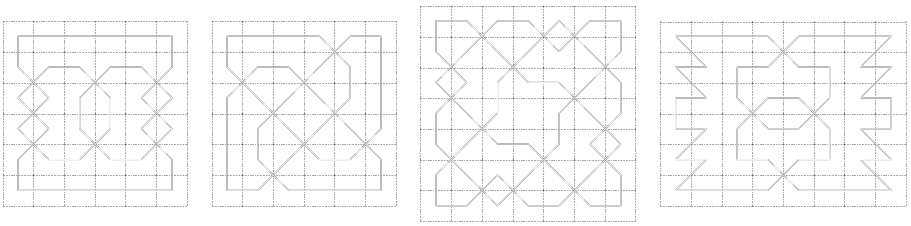
Similarly, the arcs from $(k, 2k + 1 + j)$ must go west. And symmetrically, the arcs from $(m - 1 - k, n - 2k - 2 - j)$ and $(m - 1 - k, 2k + 1 + j)$ must go east.

pi, as random source
introverted edge
Beluhov
perfect matching
introverted
extroverted

Now we obtain an $m \times (n-2)$ tour by removing two arcs in each row, namely the two forced arcs that are nearest the center. We also shift all of the "extreme" vertices in a row (those that don't lie between the two removed arcs), one column towards the center, with their arcs., For example, here's how a "random" 8×16 tour becomes 8×14:



236. The nonzero values are $X_{3,3} = 1$; $X_{4,4} = 8 = 4 \cdot 2$, $X_{4,5} = 72 = 2 \cdot 4 + 4 \cdot 16$, $X_{4,6} = X_{4,8} = 50 = 2 \cdot 9 + 4 \cdot 8$, $X_{4,7} = 200 = 2 \cdot 8 + 4 \cdot 46$; $X_{5,5} = 128 = 8 \cdot 16$, $X_{5,6} = 144 = 2{\cdot}12{+}4{\cdot}30$; $X_{6,6} = 6480 = 2{\cdot}6{+}4{\cdot}63{+}8{\cdot}777$, $X_{6,7} = 545278 = 2{\cdot}611{+}4{\cdot}136014$, $X_{6,8} = 308637 = 1 \cdot 25 + 2 \cdot 898 + 4 \cdot 76704$; $X_{7,7} = 552960 = 2 \cdot 4 + 4 \cdot 94 + 8 \cdot 69072$, $X_{7,8} = 9210632 = 2 \cdot 2146 + 4 \cdot 2301585$; $X_{8,8} = 237059136 = 4 \cdot 12708 + 8 \cdot 29626038$. Here are some of the attractive examples with different kinds of 4-way symmetry:



240. True. The edges of (53) are clearly distinct when the vertices are distinct, unless perhaps $l = 2$ and $t_0 = t_2$. But even then, $t_0\ \mathbb{I}_0\mathbb{I}_1\ t_1$ is never the same as $t_1\ \mathbb{I}_1\mathbb{I}_2\ t_0$.

241. Let the edges be $u\ \mathbb{I}_0\mathbb{I}_1\ x\ \mathbb{I}_1\mathbb{I}_2\ y\ \mathbb{I}_2\mathbb{I}_3\ x\ \mathbb{I}_3\mathbb{I}_4\ v$, with arbitrary arrows $\mathbb{I}_0, \ldots, \mathbb{I}_4$.

242. (a) (i) $2^{n-1}n!$; (ii) $2^{n-1}(n-1)!$.
(b) (i) $n!$; (ii) $(n-1)! \cdot [n \text{ even}]$.
(c) (i) $(n{+}1)!/2$ (at most one introverted edge); (ii) $(n{-}1)!$ (no introverted edges).

243. (Solution by N. Beluhov.) The fers moves make a perfect matching of the $m \times n$ fers graph. That graph has a unique matching when $m \bmod 2 = n \bmod 2 = 0$, else none.

The $m \times n$ alfil graph is the union of four fers graphs, of sizes $\lfloor m/2 \rfloor \times \lfloor n/2 \rfloor$, $\lfloor m/2 \rfloor \times \lceil n/2 \rceil$, $\lceil m/2 \rceil \times \lfloor n/2 \rfloor$, $\lceil m/2 \rceil \times \lceil n/2 \rceil$. So it has at most one perfect matching. We conclude that *a Hamiltonian cycle of alternating anypiece+alfil moves on an $m \times n$ board is possible only if $m \bmod 4 = n \bmod 4 = 0$; furthermore, the alfil moves of any such cycle always make a fixed pattern of $(m/4)(n/4)$ groups of four overlapping X's.*

244. Create bidirected graphs without directed edges, as in (54). Then problem (a) has 20293176 solutions (found in 8.7 Gμ); (b) has 127119280 (found in 52.1 Gμ).

245. Half true: A directed graph B (having no introverted or extroverted edges) always leads to a bipartite $G(B)$, with the natural partition into positive and negative vertices.

But $G(B)$ can also be bipartite in many other cases — such as when B is obtained as in (54) from graphs G and H that are themselves bipartite.

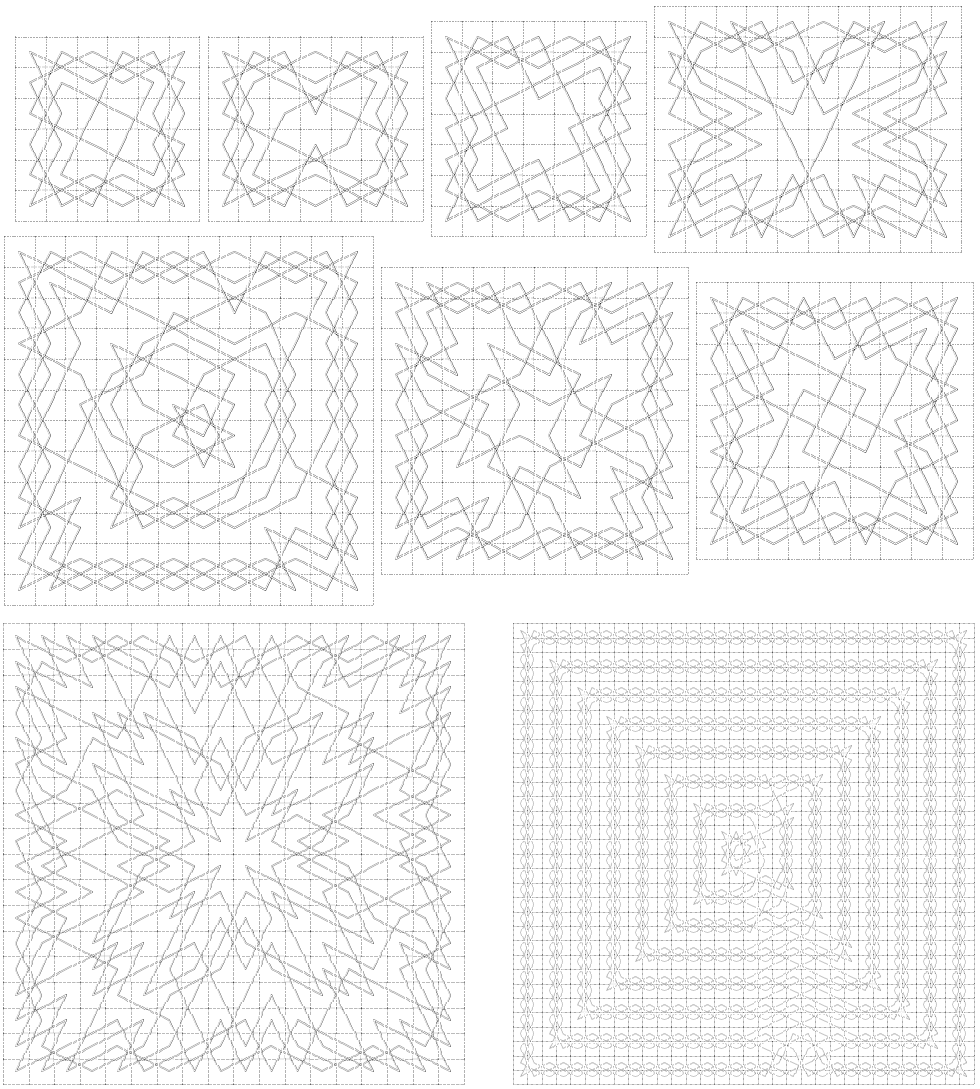


Fig. A–24. A gallery of whirling knight's tours with special qualities.

246. Let B have three vertices $\{v, v_L, v_R\}$ for each vertex v of G and H. Its bidirected edges are $v \langle\langle v_L$, $v \langle\langle v_R$, $v \rangle\rangle v_L$, $v \rangle\rangle v_R$ for all v; also $u_L \langle\rangle v_R$ when $u \to v$ in G, $u_L \rangle\langle v_R$ when $u \Rightarrow v$ in H. The Hamiltonian cycles of B fall into a pattern $\cdots u_R \langle\langle u \langle\langle u_L \langle\rangle v_R \rangle\rangle v \rangle\rangle v_L \rangle\langle w_R \langle\langle w \langle\langle w_L \cdots$ when $u \to v \Rightarrow w$.

247. The construction of answer 246 yields a bidirected graph with 192 vertices, 536 edges, and $44 = 8 \cdot 4 + 4 \cdot 3$ Hamiltonian cycles (found in just 252 kilomems!). One of the three essentially different symmetrical solutions is shown, together with one of the $4 \cdot 3$ solutions of size 8×9. (There are 14544 10×10 solutions, found in 139 Mμ; 8151216 12×12 solutions, found in 41 Gμ.)

bidirected graph
purge
topological sorting
theoretical
practical

1	64	23	22	47	46	61	60		1	72	55	54	15	14	69	68	35
24	9	8	63	62	21	20	45		56	17	16	71	70	37	36	51	34
25	2	49	48	7	6	59	44		57	2	25	24	53	52	13	50	67
50	3	10	37	36	43	58	19		18	3	58	39	38	23	12	33	66
51	26	11	4	5	42	35	18		19	40	59	26	27	22	65	32	49
12	27	38	39	16	17	34	57		4	41	20	21	8	9	64	11	48
13	52	53	30	31	40	41	56		5	60	61	44	45	28	29	10	31
28	29	14	15	54	55	32	33		42	43	6	7	62	63	46	47	30

250. Start with `ADJ[`v`][`w`]` $\leftarrow \infty$ and `DEG(`v`)` $\leftarrow 0$ for $0 \le v, w < 2n$. Then for $0 \le v < n$ and $a \leftarrow$ `ARCS(`v`)`, do the following while $a \neq \Lambda$: Set $w \leftarrow$ `TIP(`a`)` and $l \leftarrow$ `LEN(`a`)`; set $v' \leftarrow 2v + ((l \gg 1)\ \&\ 1)$, $w' \leftarrow 2w + (((-l) \gg 1)\ \&\ 1)$; if $v \neq w$ and `ADJ[`v'`][`w'`]` $= \infty$, insert(v', w') and insert(w', v'); set $a \leftarrow$ `NEXT(`a`)`. Here 'insert(v, w)' means 'set $d \leftarrow$ `DEG(`v`)`, `NBR[`v`][`d`]` $\leftarrow w$, `ADJ[`v`][`w`]` $\leftarrow d$, `DEG(`v`)` $\leftarrow d + 1$'.

251. First do step B4. Then purge(`CURU`, `CURV`) and purge(`CURV`, `CURU`) (see answer 112). Then makeinner(`CURU`) and makeinner(`CURV`); activate(`CURU`$\oplus 1$) and activate(`CURV`$\oplus 1$); makemates(`CURU`$\oplus 1$, `CURV`$\oplus 1$).

252. (a) makeinner(u); activate($u \oplus 1$); makemates($u \oplus 1, v \oplus 1$).
(b) deactivate(u); makemates($v \oplus 1$, `MATE(`u`)`).

253. We use five arrays (x, xs, y, ys, sg) of length n and two arrays (v, vs) of length $n + 1$. Start with $x_k \leftarrow -1$ for $0 \le k < n$. For $0 \le k < n$ do this: Set $i \leftarrow$ `EU[`k`]` $\gg 1$ and $j \leftarrow$ `EV[`k`]` $\gg 1$. If $x_i < 0$, set $x_i \leftarrow j$, $sg_i \leftarrow$ `EU[`k`]` $\&\ 1$, $xs_i \leftarrow$ `EV[`k`]` $\&\ 1$; otherwise set $y_i \leftarrow j$ and $ys_i \leftarrow$ `EV[`k`]` $\&\ 1$. If $x_j < 0$, set $x_j \leftarrow i$, $sg_j \leftarrow$ `EV[`k`]` $\&\ 1$, $xs_j \leftarrow$ `EU[`k`]` $\&\ 1$; otherwise set $y_j \leftarrow i$ and $ys_j \leftarrow$ `EU[`k`]` $\&\ 1$.

Then set $v_0 \leftarrow 0$ and $vs_0 \leftarrow 1$. For $k = 1, \ldots, n$, set $j \leftarrow v_{k-1}$, $v_k \leftarrow (vs_{k-1} = sg_j ?\ y_j : x_j)$, $vs_k \leftarrow (vs_{k-1} = sg_j ?\ ys_j : xs_j)$. This yields (53) with $l = n$, $u = v = 0$, $t_k = v_k$, and $\mathbb{I}_k = (vs_k = 1 ?\ \langle : \rangle)$.

254. Steps B15 and B16 backtrack to level 0, and remove the first root edge '$0^- \text{—} 1^-$' from the graph. Now 0^- and 1^- have degree 1; so they go onto the trigger stack. The subpath $0^+ \text{—} 0^- \text{—} 1^+ \text{—} 1^- \text{—} 2^+ \text{—} 2^-$ is quickly forced; and the second Hamiltonian matching is visited, namely $\{0^- \text{—} 1^+,\ 1^- \text{—} 2^+, 2^- \text{—} 0^+\}$. There are no others.

256. In answer 250 (for step B1), let $v' = 2v$ and $w' = 2w + 1$.

In place of answer 253 (for step B13), first do this for $0 \le k < n$: Set $i \leftarrow$ `EU[`k`]` $\gg 1$ and $j \leftarrow$ `EV[`k`]` $\gg 1$; if `EU[`k`]` $\&\ 1 = 1$, set $x_j \leftarrow i$, otherwise set $x_i \leftarrow j$. Then set $k \leftarrow 0$ and, for $j = 0, 1, \ldots, n$, do this: $v_j \leftarrow k$, $k \leftarrow x_k$. The desired Hamiltonian cycle is $0 = v_0 \longrightarrow v_1 \longrightarrow \cdots \longrightarrow v_n = 0$.

270. (a) $v_1 \longrightarrow \cdots \longrightarrow v_n$ in G if and only if $s \longrightarrow v_1 \longrightarrow \cdots \longrightarrow v_n \longrightarrow t \longrightarrow s$ in $\widehat{G}$.

(b) Suppose $u_1 \longrightarrow \cdots \longrightarrow u_n$ and $v_1 \longrightarrow \cdots \longrightarrow v_n$ are Hamiltonian, with $u_j \neq v_j$ and j minimum. Then $u_j = v_k$ and $v_j = u_l$ for some $k > j$ and $l > j$. Consequently $u_l \longrightarrow v_{j+1} \longrightarrow \cdots \longrightarrow v_k \longrightarrow u_{j+1} \longrightarrow \cdots \longrightarrow u_l$ is a cycle.

(c) True. We can assume that the vertices of G are labeled "topologically" (see Algorithm 2.2.3T) so that $v_j \longrightarrow v_k$ implies $j < k$. Algorithm B will choose $t \longrightarrow s \longrightarrow v_1$, since s and v_1 have in-degree 1. If $v_1 \longrightarrow \cdots \longrightarrow v_k$ have been chosen, for $k \ge 1$, then $\{v_1, \ldots, v_{k-1}\}$ are inner; hence v_{k+1} has in-degree 0 or 1. No backtracking is needed.

271. Construct $O(n^p)$ subgraphs of $\widehat{G}$ by removing one arc from each cycle in all possible ways. Then G has a Hamiltonian path if and only if at least one of those subgraphs is Hamiltonian. And each subgraph can be tested in $O(m+n)$ steps by exercise 270(c).

(Of course this result is purely theoretical, by no means practical!)

272. Define $\widehat{G}$ by generalizing exercise 270: The bidirected edges involving the new vertices s and t are now $s \rangle\rangle v$, $s \rangle\langle v$, $v \rangle\rangle t$, $v \langle\rangle t$, and $t \rangle\rangle s$. Each Hamiltonian path $a \cdots z$ of G corresponds to exactly two Hamiltonian cycles of $\widehat{G}$, namely the cycles $s \rangle\!| a \cdots z |\!\langle t \rangle\rangle s$ and $s \rangle\!| z \cdots a |\!\langle t \rangle\rangle s$; we can reject the cycles for which $a > z$.

275. (This exercise was first solved in November 2025, by Peter Weigel. The main new idea is that every outer vertex of an m-config (every endpoint of a subpath) is further distinguished as either a "source" or a "sink." An online version of Weigel's program, called DYNABIDIHAM, has been updated to include the compression and compaction refinements of exercises 203 and 204. There's also DYNADIHAM, a variant for directed (not bidirected) graphs, which exploits optimizations that apply to this special case.)

276. The total number is 4 for $n = 2$; 460,170,137,868,876 for $n = 8$; and 27,822,591,187,156,792,320,392,844,273,708,099,840,713,716,687,359,860,389,736,933,210,508,256 for $n = 32$ (see OEIS A391979). Surprise: That last value is a lot *less* than (40), because the growth ratio (≈ 211.48) is much less than the ratio 526.46 for knights alone.

The calculation doesn't take long or require a lot of RAM, because the number of m-classes stays small. Indeed, when $76 \le m \le 240$, the class sizes are just (22603069, 23798237, 33108128, 34378026, 38995642, 39841197, 43707686, 42988988) respectively, for $m \bmod 8 = (0, \ldots, 7)$; only about 642 million pointers are needed for the main trie.

277. Let $ij \longrightarrow i'j'$ when $i + j$ is even and $ij \text{ — } i'j'$ is a knight move; $ij \longrightarrow i'j'$ when $i + j$ is odd and $ij \text{ — } i'j'$ is a wazir move. Every move changes the parity of $i + j$; hence the two types of move must alternate.

(DYNADIHAM will need fewer `MEM` pointers than DYNABIDIHAM.)

279. Consider the adjacency matrix with 4-bit entries $a_{vw} = 8[v \langle\langle w] + 4[v \rangle\rangle w] + 2[v \langle\rangle w] + [v \rangle\langle w]$. We can find an equivalent that maximizes the lexicographic order of entries on and above the diagonal. (See the online program CANONIZE-BIDIGRAPH.)

A similar canonical form, using the 4-bit entries $a_{vw} = 8[v \langle\rangle w] + 4[v \rangle\langle w] + 2[v \langle\langle w] + [v \rangle\rangle w]$, would have the property that $[B]$ is signed (has no directed edges) if and only if B is signed.

281. (a) $b \langle\rangle c \rangle\rangle d \rangle\langle a \langle\rangle e$; $b \rangle\rangle a \rangle\langle d \langle\langle c \langle\langle e$. (But no bidicycle contains both b and e.)

(b) Let $C = v \langle\langle w \langle \cdots \langle v$ be a bidirected cycle that contains the edge $v \langle\langle w$, and let $P = u \langle \cdots v$ be a bidirected path of cyclic edges. We will prove that w is reachable from $u\langle$. This is clear if $w \in P$. Otherwise, if v is the only vertex common to P and C, we can append C or its reverse to P. Otherwise let $P = u \langle \cdots x \cdots v$ and $C = v \langle\langle w \langle \cdots x \cdots \langle v$, where x is the first vertex of P that's in C. Either $u \langle \cdots x \cdots \langle v \langle\langle w$ or $u \langle \cdots x \cdots \rangle w$ is legitimate, using a subpath of C or its reverse to get from x to w.

A similar proof shows that w is reachable also from $u\rangle$.

(c) Proofs like (b) show that $u \Rightarrow w$ whenever we have $u \Rightarrow v$ and either $v \langle\rangle w$, $v \rangle\langle w$, or $v \rangle\rangle w$ is a cyclic edge of B.

(d) $u \Rightarrow v$ implies $u \text{ — } v$. $u \Rightarrow v$ and $v \text{ — } w$ implies $u \Rightarrow w$.

(e) By induction on l. (Thus the "strong components" of B, the equivalence classes of $\Leftrightarrow$, are the same as the "cyclic edge components." Can the cyclic edges of B all be identified in linear time?)

[The structure theory in this exercise and exercise 283 was developed by N. Kita in arXiv:1709.07414 [math.CO] (2017), 10 pages, and by S. Wiederrecht in *Foundations of Computing* **16** (Universitätsverlag der TU Berlin, 2022), xvii + 463 pages, §9.2.]

282. Let B have four vertices $\{a, u, v, w\}$ and six edges $a \rangle\langle u$, $a \rangle\rangle u$, $a \langle\langle v$, $a \langle\rangle v$, $a \rangle\langle w$, $a \rangle\rangle w$. Then $u \Leftrightarrow v$ and $v \Leftrightarrow w$. But $u \not\Rightarrow w$, because every bidirected walk from u to w contains vertex a twice.

Weigel
outer vertex
source
sink
online
DYNABIDIHAM
compression
compaction
DYNADIHAM
directed
OEIS
growth ratio
parity
adjacency matrix
lexicographic order
online
signed
strong components
linear time
historical notes
Kita
Wiederrecht
bidirected walk

switch
Kasteleyn
infinite *undirected* grid

283. (a) Let u, v, w be distinct vertices with $u \langle\sim\rangle v$ and $v \langle\sim\rangle w$. Assume that $P = u\langle\cdots\rangle w$ is a bidirected path of cyclic edges. There must be a bidirected path $Q = v\langle\cdots\langle u$, since $v \Rightarrow u$. Some vertex $x \neq u$ must be common to P and Q, because there's no bidirected path $v\langle\cdots\rangle w$. Hence $P = u\langle\cdots x\cdots\rangle w$ and $Q = v\langle\cdots x\cdots\langle u$, with no elements of P preceding x in Q. There are four cases, each of which leads to a contradiction. *Case 1:* $\langle x\langle$ in P and $\langle x\langle$ in Q. Then $v\langle\cdots\langle x\langle\cdots\langle w$. *Case 2:* $\langle x\langle$ in P and $\rangle x\rangle$ in Q. Then $v\langle\cdots x\rangle\cdots\rangle u$. *Case 3:* $\rangle x\rangle$ in P and $\langle x\langle$ in Q. Then $v\langle\cdots\langle x\langle\cdots\langle u$. *Case 4:* $\rangle x\rangle$ in P and $\rangle x\rangle$ in Q. Then $v\langle\cdots\rangle x\rangle\cdots\langle w$.

We've proved that '$\langle\sim\rangle$' is an equivalence relation. So is '$\rangle\sim\langle$', because we can switch all vertices as in exercise 279.

(b) Switch vertex w. (This doesn't change the strong components.)

285. (a) $\big(i+j$ even? $2\max(|i|,|j|)$: $1+2\max(|i|-1,|j|)\big)$.

(b) $\big(i=j=0$? 0: $i+j$ even? $f(|i+j|,|j-i-1|)$: $f(|j-i-1|+1,|i+j|)\big)$, where $f(d,p) = d + 4[p>d]\lceil(p-d)/4\rceil$. (The vertices at fixed distance k belong to diagonals that form an interesting pattern.) Here are the distances near 00:

(a)

$\cdots$	9	6	5	6	5	6	5	6	9 $\cdots$
$\cdots$	8	7	4	3	4	3	4	7	8 $\cdots$
$\cdots$	9	6	5	2	1	2	5	6	9 $\cdots$
$\cdots$	8	7	4	3	0	3	4	7	8 $\cdots$
$\cdots$	9	6	5	2	1	2	5	6	9 $\cdots$
$\cdots$	8	7	4	3	4	3	4	7	8 $\cdots$
$\cdots$	9	6	5	6	5	6	5	6	9 $\cdots$

(b)

$\cdots$	7	6	5	4	3	6	5	8	7 $\cdots$
$\cdots$	6	7	4	5	2	3	4	5	6 $\cdots$
$\cdots$	5	4	3	2	1	4	3	6	5 $\cdots$
$\cdots$	8	5	6	3	0	1	2	3	4 $\cdots$
$\cdots$	7	6	5	4	3	2	5	4	7 $\cdots$
$\cdots$	10	7	8	5	6	3	4	5	6 $\cdots$
$\cdots$	9	8	7	6	5	4	7	6	9 $\cdots$

(c) Let $[ij]$ denote the always-turn vertex $(i{+}j)(j{-}i)$. Then Manhattan vertices $(2i)(2j)$, $(2i)(2j{+}1)$, $(2i{+}1)(2j)$, $(2i{+}1)(2j{+}1)$ correspond respectively to the always-turn arcs $[ij]{+}0\bar1 \longrightarrow [ij]$, $[ij] \longrightarrow [ij]{+}10$, $[ij]{+}1\bar1 \longrightarrow [ij]{+}0\bar1$, $[ij]{+}10 \longrightarrow [ij]{+}1\bar1$. [This correspondence was noted by P. W. Kasteleyn in *Physica* **29** (1963), 1333.]

(d) The (infinite) Hamiltonian path can't have a beginning or ending vertex. For if it began, say, at 00, it would have to contain both $0\bar1 \longrightarrow 1\bar1$ and $10 \longrightarrow 1\bar1$.

One solution is the doubly infinite spiral $\cdots \longrightarrow 0\bar2 \longrightarrow \bar1\bar2 \longrightarrow \bar2\bar2 \longrightarrow \bar2\bar1 \longrightarrow \bar20 \longrightarrow \bar21 \longrightarrow \bar22 \longrightarrow \bar23 \longrightarrow \bar13 \longrightarrow 03 \longrightarrow 13 \longrightarrow 12 \longrightarrow 11 \longrightarrow 10 \longrightarrow 00 \longrightarrow 01 \longrightarrow 02 \longrightarrow \bar12 \longrightarrow \bar11 \longrightarrow \bar10 \longrightarrow \bar1\bar1 \longrightarrow 0\bar1 \longrightarrow 1\bar1 \longrightarrow 2\bar1 \longrightarrow 20 \longrightarrow \cdots$. (Every solution corresponds to a spanning tree of the infinite *undirected* grid that doesn't disconnect the plane; see exercise 286.)

286. (a) The arcs of $Q(m,n)$ surround $(m{-}1)(n{-}1)$ square "cells," of which $(\frac{m}{2})(\frac{n}{2})$ are surrounded by a clockwise $\vec{C_4}$; they're indicated by large shaded dots in the diagram. Diagonally between them are $(\frac{m}{2}{-}1)(\frac{n}{2}{-}1)$ cells that are surrounded by a *counter*clockwise $\vec{C_4}$. The remaining cells are always bounded by a pair of oppositely directed arcs, either horizontal or vertical, in any Hamiltonian cycle; the diagram uses shaded lines to illustrate this property. It's not difficult to see that the shaded lines must connect the shaded dots, without forming a cycle; thus they form a spanning tree.

(b) We must prove first that every Hamiltonian cycle of $\overline{Q}(m,n)$ unwraps to a polyomino, namely that the unwrapped path always returns to its starting point. Assuming that the unwrapped path starts at 00, it will end at $(am)(bn)$ for some integers a and b. Indeed, a is the number of arcs from $(m{-}1)j$ to $0j$ in the cycle, for some odd j, minus the number of arcs that go from $0j$ to $(m{-}1)j$ for some even j. And $a = 0$, because the cycle contains $0j \longrightarrow (m{-}1)j$ if and only if it doesn't contain $0j \longrightarrow 0(j{+}1)$ if and only if it contains $(m{-}1)(j{+}1) \longrightarrow 0(j{+}1)$. Similarly, $b = 0$.

Kasteleyn
determinants
unit square
OEIS
Janson

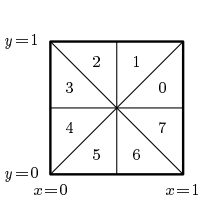
The arcs of $\overline{Q}(m,n)$ surround mn square cells, with clockwise and counterclockwise $C_4^{\rightarrow}$s equally prominent. When a Hamiltonian cycle is shifted by 11, we get another one with the opposite direction. Hence we argue as in (a), but multiply by 2.

[P. W. Kasteleyn, in *Physica* **29** (1963), 1329–1337, enumerated the Hamiltonian cycles of $\overline{Q}(m,n)$ via determinants; and he also stated the result of (b) without proof.]

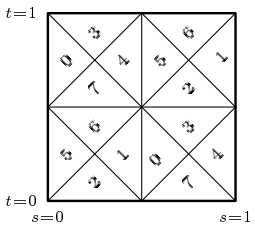
287. (a) Indeed, the double integral doesn't change when we integrate over any "reasonable" region where $(x \bmod 1, y \bmod 1)$ includes each point of $(0\,..\,1, 0\,..\,1)$ exactly once.

(b) True. Substituting $u = mx$ and $v = ny$, we have $\int_0^1\int_0^1 f(mx,ny)\,dy\,dx = \frac{1}{mn}\int_0^m\big(\int_0^n f(u,v)\,dv\big)\,du = \frac{mn}{mn}\int_0^1\big(\int_0^1 f(u,v)\,dv\big)\,du$, for any nonzero integers m and n.

(c) We can divide the unit square on (x,y) into eight regions, and assemble two copies of each region into the unit square on (s,t):



$x = s+t$
$y = -s+t$



$s = (x-y)/2$
$t = (x+y)/2$

(d) By exercise 7.2.1.6–106, $c(C_m \mathbin{\square} C_n) = mnP(m,n)$. (See OEIS A212796.)

(e) (Suggested by S. Janson.) We have $\frac{1}{mn}\ln P(m,n) = \ln 4 + S(m,n) + O(1/m) + O(1/n)$, where $S(m,n) = \frac{1}{mn}\sum_{j=1}^{m-1}\sum_{k=1}^{n-1}\ln(\sin^2\frac{j\pi}{m}+\sin^2\frac{k\pi}{n})$. The summand for j is the same as for $m-j$; so we'll restrict $1 \le j \le m' = \lfloor\frac{m}{2}\rfloor - 1$ and double the result. Similarly for k. Letting $S'(m,n) = \frac{1}{mn}\sum_{j=1}^{m'}\sum_{k=1}^{n'}\ln(\sin^2\frac{j\pi}{m}+\sin^2\frac{k\pi}{n})$, we have $S(m,n) = 4S'(m,n)+O(1/m)+O(1/n)$, because we've left out $O(m+n)$ terms in the "middle," each of which is $O(1/mn)$. Now $p(x,y) \le p(x',y')$ when $0 \le x \le x' \le \frac{1}{2}$ and $0 \le y \le y' \le \frac{1}{2}$. Hence $\int_0^{m'/m}\int_0^{n'/n} p(x,y)\,dy\,dx \le S'(m,n) \le \int_{1/m}^{1/2}\int_{1/n}^{1/2} p(x,y)\,dy\,dx$. Both of those bounds are $I'+O(1/m)+O(1/n)$, where $I' = \int_0^{1/2}\int_0^{1/2} p(x,y)\,dy\,dx = I/4$. Indeed, on the left we extend from m'/m to $\frac{1}{2}$ by integrating $p(x,y) = O(1)$ over a strip of area $O(1/m)$. The same idea extends from n'/n to $\frac{1}{2}$. On the right, we note that $\ln\sin^2\pi y \le p(x,y) \le \ln 2$; hence $\int_0^1 |p(x,y)|\,dy \le \ln 2 + \int_0^1 |\ln\sin^2\pi y|\,dy = O(1)$, for all x. Therefore $\int_0^{1/m}\int_0^{1/2} p(x,y)\,dy\,dx = O(1/m)$; $\int_0^{1/n}\int_0^{1/2} p(y,x)\,dx\,dy = O(1/n)$.

Jacquet
trigonometric identities+
tangent half-angle transformation
integration by parts
contour integration
Gamma function
Adamchik
Fisher
Catalan
historical notes
complete graph

(f) Among several ways to evaluate this remarkable double integral, one of them exploits the transformation of part (c), as suggested by Philippe Jacquet: Since $\cos 2\theta = 1-2\sin^2\theta$ we can rewrite the integrand as $p(x,y) = \ln(1-\frac12(\cos 2\pi x+\cos 2\pi y))$. Since $\cos\theta+\cos\phi = 2\cos(\frac{\theta+\phi}{2})\cos(\frac{\theta-\phi}{2})$, $p(x,y)$ is also $\ln(1-\cos\pi(x+y)\cos\pi(x-y))$. So $I=\int_0^1\int_0^1 \ln(1-\cos 2\pi s\,\cos 2\pi t)\,dt\,ds$, by (c); and $I=\int_0^1\int_0^1 \ln(1+\cos 2\pi s\,\cos 2\pi t)\,dt\,ds$, because we can replace t by $t+\frac12$. Now we change variables yet again, setting $u=\tan\pi s$ and $v=\tan\pi t$. This well-known "tangent half-angle transformation" satisfies the nice laws $\sin 2\pi s = 2u/(1+u^2)$, $\cos 2\pi s = (1-u^2)/(1+u^2)$, and $du = \pi(1+u^2)\,ds$; hence

$$\begin{aligned}
I &= \frac{1}{\pi^2}\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \ln\biggl(1+\frac{(1-u^2)(1-v^2)}{(1+u^2)(1+v^2)}\biggr)\,\frac{dv}{1+v^2}\,\frac{du}{1+u^2}\\
&= \frac{4}{\pi^2}\int_0^{\infty}\int_0^{\infty} \ln\biggl(\frac{2+2u^2v^2}{(1+u^2)(1+v^2)}\biggr)\,\frac{dv}{1+v^2}\,\frac{du}{1+u^2}\\
&= \frac{4}{\pi^2}\int_0^{\infty}\biggl(\arctan v\,\ln\frac{2+2u^2v^2}{(1+u^2)(1+v^2)}\biggr|_0^{\infty} - \int_0^{\infty}\frac{2\arctan v}{v}\biggl(\frac{dv}{1+v^2}-\frac{dv}{1+u^2v^2}\biggr)\biggr)\frac{du}{1+u^2}\\
&= \frac{4}{\pi^2}\int_0^{\infty}\biggl(\frac{\pi}{2}\ln\frac{2u^2}{1+u^2}-\biggl(\pi\ln 2-\pi\ln\frac{u+1}{u}\biggr)\biggr)\frac{du}{1+u^2} = \frac{4}{\pi}\,\biggl(-\frac{\pi}{4}\ln 2 - J\biggr),
\end{aligned}$$

where $J=\int_0^\infty \ln\frac{2u}{u+1}\frac{du}{1+u^2}$. We shall prove that $J=\frac14\pi\ln 2-G$, by breaking it into four integrals: $J=(\int_0^1\ln 2+\int_0^1\ln u-\int_0^1\ln(1+u)+\int_1^\infty\ln\frac{2u}{1+u})\frac{du}{1+u^2} = J_1+J_2-J_3+J_4$.

Since $\frac{d}{du}\arctan u=\frac{1}{1+u^2}$ and $\arctan 1=\frac{\pi}{4}$, we have $J_1=\frac14\pi\ln 2$. Integrating J_2 by parts gives $J_2+\int_0^1\frac{\arctan u}{u}\,du = \arctan u\,\ln u\,\big|_0^1 = 0$, since $\arctan\frac1N\,\ln\frac1N \approx -\frac{\ln N}{N}$ as $N\to\infty$. Also $\frac{\arctan u}{u}=1-\frac13u^2+\frac15u^4-\frac17u^6+\cdots$. Hence $-J_2=1-\frac19+\frac{1}{25}-\frac{1}{49}+\cdots=G$. The substitution $u=\frac{1-v}{1+v}$ now tells us that $J_3=\int_0^1\ln\frac{2}{1+v}\frac{dv}{1+v^2}$; and we needn't evaluate this integral, because the substitution $u=1/v$ tells us that $J_4=J_3$(!).

Another approach, based on binomialcoefficientology, is also instructive. Beginning as before with $p(x,y)=\ln(1-\frac12(\cos 2\pi x+\cos 2\pi y))$, we now expand the logarithm via Eq. 1.2.9–(17): $I=-\sum_{n=1}^\infty 2^{-n}I_n/n$, where $I_n=\int_0^1\int_0^1(\cos 2\pi x+\cos 2\pi y)^n\,dy\,dx = \sum_k\binom nk A_kA_{n-k}$ and $A_k=\int_0^1\cos^k 2\pi x\,dx$. Since $\cos\theta=(e^{i\theta}+e^{-i\theta})/2$, we have $A_k = 2^{-k}\int_0^1(e^{2\pi ix}+e^{-2\pi ix})^k dx = 2^{-k}\sum_j\binom kj\int_0^1 e^{2\pi i(k-2j)x}dx = 2^{-k}\binom{k}{k/2}[k\text{ even}]$. Hence $I_n=0$ when n is odd; and $I_{2m}=\sum_j\binom{2m}{2j}\binom{2j}{j}\binom{2m-2j}{m-j}/4^m = \sum_j\binom{2m}{m}\binom mj\binom{m}{m-j}/4^m = \binom{2m}{m}^2/4^m$. We're left with $I=-\sum_{m=1}^\infty\frac{1}{2m}\bigl(\binom{2m}{m}4^{-m}\bigr)^2$. And this sum can be evaluated by contour integration of the function $\Gamma(s)\Gamma(\frac12-s)/(s\Gamma(1-s)\Gamma(\frac12+s))$, as shown by V. S. Adamchik, *Journal for Analysis and its Applications* **21** (2002), 817–826.

For a third approach, which in fact is an elegant solution of a considerably more general problem, see M. E. Fisher, *Physical Review* **124** (1961), 1664–1672.

Catalan had introduced his constant (and evaluated it to 10 significant figures) in *Mém. Acad. royale Belgique* **33** (Brussels, 1865), 50 pages.

290. (a) No. For example, if G is the complete graph on $\{a,b,c,d\}$ we might have $a\longrightarrow c$, $b\longrightarrow c$, $c\longrightarrow d$, $d\longrightarrow a$.

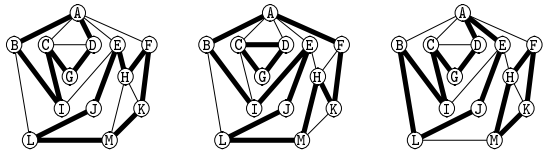
(b) Yes. Starting at any vertex v_0, there's an oriented walk $v_0\longrightarrow v_1\longrightarrow\cdots\longrightarrow v_l$ for arbitrarily large l. By finiteness, there's a least l such that $v_l=v_k$ for some $k<l$. We must have $k=0$; otherwise v_l would have more than one predecessor. Hence the selected arcs form a cycle cover.

author
asymmetry clauses
at-least-one clauses
at-most-one clauses
reverse a linked list

291. The author prefers (65) because of its simplicity and the fact that *langford*(n) was a winner in §7.2.2.2. Of course its $\Omega(d^2)$ clauses are problematic when d is large; but Hamiltonian cycles are easy to find when vertices have high degree.

Suppose G is a Hamiltonian graph with vertices of high degree. Assign the weight $\deg(u)+\deg(v)$ to every edge u — v; and obtain a subgraph G' by repeatedly removing an edge of maximum weight, until all vertices have degree ≤ 10 (say). Then G' is almost certainly Hamiltonian, and we'll need to investigate further only when it's not.

292. Ignoring orientation, the graph of Fig. 127(a) has exactly seven cycle covers, namely



(f) (g) (h)

besides those already shown. Merging (ACGD) and (BLJEI) to form C_1'' = (ADGCIBLJE) gives (h); and C_1'' can be merged with (FKMH), yielding (ADGCIBLJEHMKF). But the third SAT round might have found (h) instead of (e).

294. vu. (And $l\oplus1$ represents uv; $l\oplus2$ represents $\overline{vu}$.)

296. (Step C1 will have initialized the solver, telling it that the variables are numbered from 2 to $N-1$. In this answer, '$l_1,\ldots,l_k$' denotes a clause to be sent to the solver, containing the literals numbered l_1, ..., l_k.) Send '$4j+1, 4j+3$' for $1\le j<N/2$ (the asymmetry clauses). For $0\le v<n$, let v_1, ..., v_d be the vertices such that $v\longrightarrow v_j$. Send '2ADJ[v][v_1],...,2ADJ[v][v_d]' and '2ADJ[v_1][v],...,2ADJ[v_d][v]' (the at-least-one clauses). Also, for $1\le i<j\le d$, send '2ADJ[v][v_i]+1, 2ADJ[v][v_j]+1' and '2ADJ[v_i][v]+1, 2ADJ[v_j][v]+1' (the at-most-one clauses).

297. Set $j\leftarrow0$, and do the following while $j<t$: "Set $c\leftarrow$ CYC[j], $k\leftarrow0$, and $v\leftarrow$ HEAD[c]. For each u with $v\longrightarrow u$ and CID[u] $\ne c$, set $k\leftarrow k+1$ and $l_k\leftarrow$ 2ADJ[v][u]. Then set $v\leftarrow$ SUCC[v], and repeat the loop on u if $v\ne$ HEAD[j]. Finally send '$l_1,\ldots,l_k$' and '$l_1\oplus2,\ldots,l_k\oplus2$' to the solver. If $t>2$, set $j\leftarrow j+1$; otherwise set $j\leftarrow2$." (We've used the fact that $v\in$ CYC[0] $\iff v\notin$ CYC[1] when $t=2$.)

298. During this routine we've fully merged CYC[i] for $0\le i<j$.

C6.1. [Begin loop on j.] Set $j\leftarrow0$.

C6.2. [Choose c.] Set $c\leftarrow$ CYC[j]. (We'll try to absorb other cycles into c.)

C6.3. [Begin loop on v.] Set $v\leftarrow$ HEAD[c] and $w\leftarrow$ SUCC[v].

C6.4. [Begin loop on v'.] Set v' to the first vertex such that $v\longrightarrow v'$.

C6.5. [Is v' in c?] Set $c'\leftarrow$ CID[v'], and go to C6.11 if $c'=c$.

C6.6. [Check PRED[v'].] Set $w'\leftarrow$ PRED[v']. Go to C6.9 if ADJ[w'][w] $\ne0$.

C6.7. [Check SUCC[v'].] Set $w'\leftarrow$ SUCC[v']. Go to C6.11 if ADJ[w'][w] $=0$.

C6.8. [Reverse subpath.] Set $u\leftarrow w'$, $u'\leftarrow$ SUCC[u]. While $u\ne v'$, set $u''\leftarrow$ SUCC[u'], SUCC[u'] $\leftarrow u$, PRED[u] $\leftarrow u'$, $u\leftarrow u'$, $u'\leftarrow u''$.

C6.9. [Merge.] Set SUCC[v] $\leftarrow v'$, SUCC[w'] $\leftarrow w$, PRED[v'] $\leftarrow v$, PRED[w] $\leftarrow w'$, $u\leftarrow v'$. Repeatedly set CID[u] $\leftarrow c$ and $u\leftarrow$ SUCC[u] until $u=w$.

C6.10. [Delete c'.] Set $t\leftarrow t-1$; go to C7 if $t=1$. Otherwise set $k\leftarrow$ CLOC[c']. If $k>j$, set CYC[k] $\leftarrow$ CYC[t] and CLOC[CYC[k]] $\leftarrow k$. Otherwise set $j\leftarrow j-1$, and while $k<t$ set CYC[k] $\leftarrow$ CYC[$k+1$], CLOC[CYC[k]] $\leftarrow k$, $k\leftarrow k+1$.

C6.11. [Advance v'.] Set v' to the next vertex such that $v \longrightarrow v'$ and go to C6.5, unless v' was the last neighbor of v.

C6.12. [Advance v.] If $w \neq$ `HEAD[`c`]`, set $v \leftarrow w$ and $w \leftarrow$ `SUCC[`w`]`. Return to C6.4.

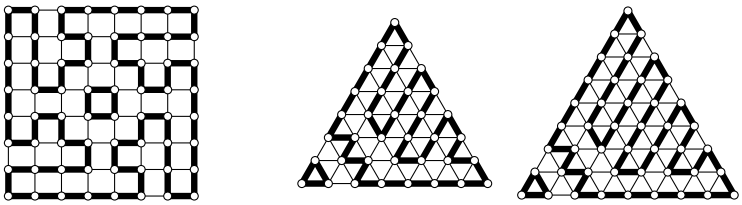
C6.13. [Advance j.] Set $j \leftarrow j{+}1$, and return to C6.2 if $j < t$. ▮

parity
Rivest
associative block design
close
pigeonhole problem
author

300. (a) Let $u \equiv v$ mean that u and v belong to the same part. We have $u \mathrel{—} v$ if and only if $u \neq v$ and $u \not\equiv v$. Suppose $u \mathrel{—} v$ is an edge of one cycle and $x \mathrel{—} y \mathrel{—} z$ is a path in another. Assume without loss of generality that $u \mathrel{—} y$ (otherwise $v \mathrel{—} y$). If $v \not— x$ and $v \not— z$ then $x \equiv v \equiv z$, in which case $u \mathrel{—} x$ and $v \mathrel{—} y$.

(b) False in $P_8 \mathbin{\square} P_8$ (see below!). (But apparently true when $\min(m, n) < 8$.)

(c) The construction shown here generalizes in a hopefully straightforward way.



(d) Represent each vertex by an n-bit string, and let α be the first $n{-}4$ bits. We get cycles of length 8 when the edges for even-parity α are $\alpha{*}001$, $\alpha0{*}00$, $\alpha10{*}0$, $\alpha110{*}$, $\alpha{*}110$, $\alpha1{*}11$, $\alpha01{*}1$, $\alpha001{*}$, $\alpha{*}100$, $\alpha0{*}01$, $\alpha11{*}1$, $\alpha100{*}$, $\alpha{*}011$, $\alpha1{*}10$, $\alpha00{*}0$, $\alpha011{*}$; for odd-parity α they are $\alpha{*}010$, $\alpha0{*}11$, $\alpha10{*}1$, $\alpha111{*}$, $\alpha{*}101$, $\alpha1{*}00$, $\alpha01{*}0$, $\alpha000{*}$, $\alpha{*}000$, $\alpha1{*}01$, $\alpha00{*}1$, $\alpha010{*}$, $\alpha{*}111$, $\alpha0{*}10$, $\alpha11{*}0$, $\alpha101{*}$. (The first eight edges here are Rivest's associative block design 6.5–(13); the other groups of eight are the same but $\oplus$-ed with 0101, 0011, and 1001.) Notice that no two of these edges are "close"; that is, their endpoints don't make a 4-cycle.

302. (a) There are $(t{+}1)m$ "local" vertices l_{ij} for $0 \le i \le t$ and $1 \le j \le m$, and t "global" vertices g_k for $1 \le k \le t$. (i) All global vertices belong to the same cycle. (For if $g_k \longrightarrow v$ and $g_{k'} \longrightarrow v'$ then $g_k \longrightarrow v'$ and $g_{k'} \longrightarrow v$.) (ii) All other cycles are m-cycles on the vertices $l_{i*} = \{l_{i1}, \ldots, l_{im}\}$ for some i. (We can't have $l_{ij} \longrightarrow l_{i'j'}$ when $i \neq i'$; a cycle containing l_{ij} and another cycle containing $l_{ij'}$ can be merged.) (iii) The global cycle cannot contain consecutive global vertices (since it can't merge with a local cycle).

For example, here are two of $N_{5,3}$'s unmergeable cycle covers:

$$(g_1 l_{11} l_{12} l_{13} g_2 l_{21} l_{22} l_{23} g_3 l_{31} l_{32} l_{33} g_4 l_{41} l_{42} l_{43} g_5 l_{51} l_{52} l_{53})(l_{01} l_{02} l_{03});$$
$$(g_2 l_{01} g_4 l_{31} g_1 l_{03} g_5 l_{33} l_{32} g_3 l_{02})(l_{11} l_{12} l_{13})(l_{21} l_{23} l_{22})(l_{41} l_{43} l_{42})(l_{51} l_{52} l_{53}).$$

(b) Every such clause arises from the covers in (a). For example, the first cover above generates the clauses L_0 and R_0, where

$$L_i = \bigvee_{j=1}^{m} \bigvee_{k=1}^{t} l_{ij} g_k; \qquad R_i = \bigvee_{j=1}^{m} \bigvee_{k=1}^{t} g_k l_{ij}.$$

The second cover generates $L_1 \wedge R_1 \wedge L_2 \wedge R_2 \wedge L_4 \wedge R_4 \wedge L_5 \wedge R_5$, and two larger clauses $(L_1 \vee L_2 \vee L_4 \vee L_5) \wedge (R_1 \vee R_2 \vee R_4 \vee R_5)$ (which are subsumed by the others).

(c) If $m < 3$, there are no cycle covers; the problem is UNSAT on round 1. Otherwise cycle covers exist until all clauses $L_0, \ldots, L_t$ (hence also $R_0, \ldots, R_t$) have been generated. A single round generates at least one L_j and at most $t{+}1{-}\lceil t/m \rceil$ of them. So we might have UNSAT as early as round 3, or as late as round $t{+}2$. (The final unsatisfiable clauses represent a pigeonhole problem, so the running time grows exponentially with t. In the author's experiments, 12 Tμ were needed to refute $N_{7,7}$.)

cycle cover
Merging
disjoint oriented cycles

303. There's some good news for fans of both algorithms in the raw data:

graph	A	B	C	D	E	F	G	H	P	Q	R	S	T	U
C mems	80K	19K	8M	52M	33K	5M	61K	153K	30M	20K	64K	27K	28K	133K
C rounds	2	1	72	5	2	31	1	1	1	1	2	1	1	1
H mems	185M	7K	42K	65G	8K	542M	10M	62K	203G	5K	11K	5K	8K	94G

Algorithm H is the winner for graphs B, C, E, H, Q, R, S, T; but Algorithm C is actually quite fast too in those cases, except for graph C. Algorithm C is a clear winner in the other graphs. (Graph P doesn't even have a cycle cover!) Merging often succeeds.

340. (a) There's one solution for every way to cover the vertices of G by disjoint oriented cycles of length ≥ 4. A cycle $u_0 \longrightarrow v_0 \longrightarrow u_1 \longrightarrow v_1 \longrightarrow u_2 \longrightarrow \cdots \longrightarrow v_{k-1} \longrightarrow u_0$ corresponds to choosing the options '$u_0^-\ v_0\ u_1^+$', '$u_1^-\ v_1\ u_2^+$', ..., '$u_{k-1}^-\ v_{k-1}\ u_0^+$'.

(b) From the 332 options, Algorithm 7.2.2.1X needs about 180 Mμ to find 185868 solutions, of which 2·9862 are the closed knight's tours (without removing symmetries).

341. Set up an exact cover problem as in exercise 340, where $n = 32$ and the vertices of the "first part" are the cells ij with $1 \le i, j \le 8$ and $i+j$ odd. Also add primary items $ij^\times$ for $i+j$ odd and $i > 2$. Each option now contains at least *six* items, not three: '$u_1^-\ v_1\ w_1^+\ u_2^-\ v_2\ w_2^+$' where u_1 — v_1 — w_1 and u_2 — v_2 — w_2, the six vertices are distinct, the i-coordinate of u_1 is less than the i-coordinate of u_2, and the j coordinates of (u_1, u_2), (v_1, v_2), (w_1, w_2) are equal. (The u's and w's belong to the "first part." This option represents a pair of moves with matching columns.) Furthermore, append $ij^\times$ to each option for which $\{w_1, w_2\} = \{mj, ij\}$ or for which $\{u_1, u_2\} = \{mj, kj\}$ and $k \ne i$, where $m \in \{1, 2\}$. This trick forces the pairs of paths to "hook up" properly. For example, two of the options are '$12^-\ 24\ 16^+\ 52^-\ 44\ 56^+\ 32^\times\ 72^\times\ 56^\times$' and '$41^-\ 22\ 43^+\ 61^-\ 42\ 23^+\ 43^\times$'. Exploit symmetry by removing options with $v_1 = 11$ and $w_1 = 32$.

That makes a total of 1998 options, and Algorithm 7.2.2.1X finds 383080 solutions in 14 Gμ. Each solution chooses 16 options, and a good one yields a cycle $(v_0 v_1 \ldots v_{63})$ in which the chosen options involve v_k^-, v_{k+1}, v_{k+2}^+, v_{k+32}^-, v_{k+33}, v_{k+34}^+ for $k = 0, 2, \ldots, 30$. Most solutions define short cyclic paths; but 5264 of them yield correct tours, such as the one shown.

1	32	57	30	3	12	55	16
58	29	2	11	56	15	52	13
33	64	31	4	35	54	17	50
28	59	34	41	10	51	14	53
7	40	63	36	5	22	49	18
60	27	6	9	42	19	46	21
39	8	25	62	37	44	23	48
26	61	38	43	24	47	20	45

343. Notice that we must have $a_{23} = 2$, $a_{28} = 17$, $a_{47} = 18$, $a_{67} = 50$, $a_{76} = 48$, $a_{88} = 49$. To find such a tour, we can begin by finding a knight's path of length 14 from step 2 to step 16 that doesn't interfere with 180° rotation, nor does it involve any of the "reserved" cells. All paths of length 14 are efficiently found by pasting together compatible paths of length 7. Useful paths also have the property that each vertex in the set U of cells available for steps $(18\,..\,30)$ and $(50\,..\,64)$ has degree ≥ 2 in the graph restricted to $U \cup I$, where $I = \{47, 52, 67, 32\}$ is the set of endpoints for future paths. The endpoints must also have degree ≥ 1 in that graph. A similar method finds 14-step paths for steps 18 through 32 and 50 through 64. One of the 46,596 solutions is shown.

1	30	9	6	3	16	61	24
10	7	2	15	62	25	4	17
31	64	29	8	5	38	23	60
28	11	14	63	26	59	18	37
13	32	27	58	51	36	39	22
54	57	12	45	42	21	50	19
33	46	55	52	35	48	43	40
56	53	34	47	44	41	20	49

344. Adapting the method in the previous exercise to paths of other lengths, we find that there are respectively (2, 47, 3217, 280244, 1205980, 259230, 41366) feasible solutions for the first (4, 9, 16, 25, 36, 49, 64) steps. The first full solution is shown. [*Brentano's Chess Monthly* **1**, 1 (May 1881), 36; **1**, 5 (September 1881), 248–249. See George P. Jelliss, *Mathematical Spectrum* **25** (1992), 16–20, for information about many similar "figured tours."]

1	4	9	16	25	36	49	64
8	15	24	3	10	17	26	37
5	2	7	14	35	50	63	48
56	13	34	23	18	11	38	27
33	6	55	12	51	40	47	62
54	57	22	19	46	43	28	39
21	32	59	52	41	30	61	44
58	53	20	31	60	45	42	29

Jelliss
figured tours
Hamiltonian *paths*
Bradley
Teguia
Godbole
pancyclic
gallery of meander friezes
2-cycle
multigraph
Historical notes

350. Let X_n be that number, and let u, v, w be the "middle" vertices on the boundary. A Hamiltonian cycle on $S_{n+1}^{(3)}$ has the form u —$\cdots$— v —$\cdots$— w —$\cdots$— u, where the portions from u to v, v to w, and w to u are Hamiltonian *paths* from corner to corner of $S_n^{(3)}$. Consequently $X_{n+1} = Y_n^3$, where Y_n is the number of such paths.

Write uv for the corner between u and v. A Hamiltonian path from uw to vw has the form uw —$\cdots$— u —$\cdots$— v —$\cdots$— vw; and there are two cases, depending on whether w appears before u or after v. Thus $Y_{n+1} = Z_nY_nY_n + Y_nY_nZ_n$, where there are Z_n Hamiltonian paths from corner to corner in a graph that's like $S_n^{(3)}$ but with the third corner removed. Similarly, $Z_{n+1} = Z_nZ_nY_n + Y_nZ_nZ_n + [n=1]$.

We have $(X_1, Y_1, Z_1) = (1, 1, 1)$ and $(X_2, Y_2, Z_2) = (1, 2, 3)$. Hence, for $n \ge 3$, the formulas $X_n = 2^{3^n-2}3^{3^1+3^2+\cdots+3^{n-3}}$, $Y_n = 3X_n$, $Z_n = \frac{3}{2}Y_n$ hold by induction.

We can in fact write $X_n = 8\cdot 12^{(3^n-2-3)/2}$. [This problem was first solved by R. M. Bradley, *J. de Physique* **47** (1986), 9–14. See also A. M. Teguia and A. P. Godbole, *Australasian J. Combinatorics* **35** (2006), 181–192, who showed among other things that $S_n^{(3)}$ is *pancyclic*: It has cycles of every length, from 3 to $(3^n+3)/2$.]

360. (a) True. The infinite rightward path that's traced by repetitions of the patterns will cross a vertical line once more when traveling to the right than when traveling to the left; it will cross a horizontal line equally often when going up as when going down.

(b) The total number of edges is $mn = v_1+\cdots+v_{m-1}+h_1+\cdots+h_n$. But the left side of this equation is even, while the right side is odd by (a).

(c) Frieze 5×4a in Fig. A–30 reduces to example (iv).

(d) The case $m = 1$ is trivial. When $m = 3$, the only possibilities are (i) and its cyclic shifts and/or left-right reversal; we consider them all to be equivalent (isomorphic), although the mirror reflection looks different. When $m = 5$ and $m = 7$, there are 3+10 essentially distinct friezes, shown (except for (vi)) in Fig. A–30.

(e) Figure A–30 shows the 1+1+2+3 solutions for $n = 5, 6, 7, 8$.

(We need a special convention when $n = 2$, because the "2-cycle" C_2 is a multigraph with two edges 0 —— 1. We consider '⊓⊔⊓⊔' to be a 2×2 meander frieze. The 3×2a frieze reduces to it; the 2×4a frieze is a multiple of it.)

(f) The $4n$ automorphisms are generated by σ, τ, υ, where $ij\sigma = i((j+1) \bmod n)$, $ij\tau = i((-j) \bmod n)$, $ij\upsilon = (m-1-i)j$. Notice that $\sigma^n = \tau^2 = \upsilon^2 = (\tau\upsilon)^2 = 1$.

(g, h) The equivalence class sizes (with symmetric counts in parentheses) are:

	$n=3$	$n=4$	$n=5$	$n=6$	$n=7$
$m=3$	1 (1)	1 (1)	1 (1)	1 (1)	2 (2)
$m=4$	0 (0)	3 (2)	0 (0)	16 (8)	0 (0)
$m=5$	3 (2)	12 (7)	43 (12)	154 (35)	534 (53)
$m=6$	0 (0)	30 (5)	0 (0)	1152 (63)	0 (0)
$m=7$	10 (4)	117 (27)	1216 (75)	12326 (383)	97969 (873)

(i) See (iii) and Fig. A–30. (Friezes 5×4d and 5×5a have only twofold symmetry.)

Jones
Chinese fretwork
fretwork
Coldstream
Dipylon Master

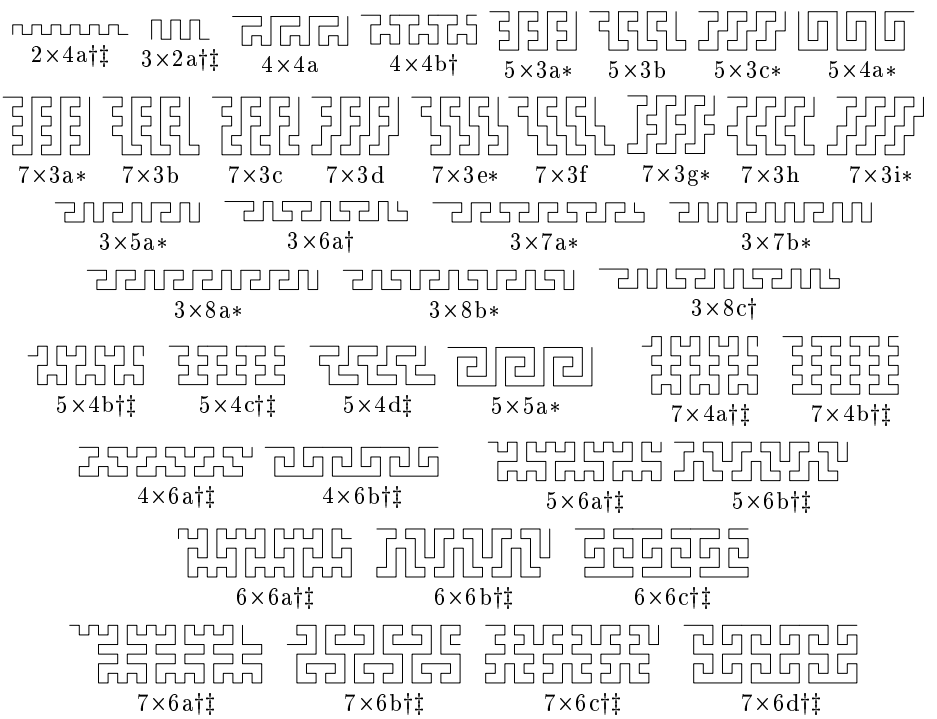
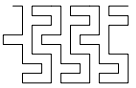


Fig. A–30. A gallery of meander friezes.
$*$ = 180° symmetry; † = left-right symmetry; ‡ = top-bottom symmetry.

Historical notes: It's interesting to find instances of meander friezes in ancient artifacts from many cultures. For example, Chapter 4 of O. Jones's *The Grammar of Ornament* (1856) included (i) as a first example of a Greek design, and mentioned (ii) as a related pattern found in Chinese fretwork. J. N. Coldstream's book *Greek Geometric Pottery* (1968), which contains detailed information about the most important early discoveries of Greek vases from the Geometric Period, illustrates more than 40 specimens with the 3×3 frieze (i), and six with the 5×4 frieze (v). His Plate 7 shows three ancient vases with a symmetrical 7×4 frieze, which is related to 7×3e in Fig. A–30 as (v) is related to 5×3b. (And 5×3b, upside down, is in his Plate 13b.)

The most elaborate meander frieze found so far in ancient sites is the magnificent 9×4 example shown here, due to the Dipylon Master and now in the collection of the National Archæological Museum in Athens. [See *Corpus Vasorum Antiquorum, Greece*, Fascicule 8 (Athens, 2002), Plates 102–105.]



361. (i) 00—01—02—03—13—23—22—12—11—21—20—10—00.
(ii) 00—01—11—10—20—21—22—23—13—12—02—03—00.
(iii) 00—01—02—03—13—12—11—21—22—23—20—10—00.
(iv) 00—01—11—21—22—12—02—03—13—23—20—10—00.
(v) 00—01—11—21—22—02—12—13—03—23—20—10—00.
(vi) 00—01—11—12—13—23—03—02—22—21—20—10—00.
(vii) 00—03—13—10—11—21—01—02—12—22—23—20—00.

Here (i) is in $P_3 \square P_4$; (ii) and (iii) are the meander friezes in exercise 360; (iv) is a multiple of the meander 3×2a; (vi) and (vii) are disjoint(!). These cycles have respectively 2, 4, 2, 8, 4, 2, 2 automorphisms; hence their equivalence classes contribute respectively 24, 12, 24, 6, 12, 24, 24 cycles to the total of 126 found by Algorithm H.

369. (a) For example, (000 012 110 102 121 113 011 023 103 001 022 003 123 111 013 021 002 010 112 100 020 122 101 120) is findable by hand with Warnsdorf's rule. (And there are 4·24 solutions altogether.)

(b) No. If so, 32 steps would take us to a cell of the opposite parity.

(c) Cells that are 32 steps apart are 110-complements. [This remarkable cycle was constructed by A. Vandermonde, when he introduced the concept of 3D knight's tours. See *Mémoires Acad. Roy. Sciences* (Paris, 1771), 566–574.]

(d) S is 201 — 000 — 012 and 031 — 133 — 213 — 023 — 121 together with all eight complements of those six edges.

(e) 136656. (The total number of 4×4×4 knight's cycles is evidently vast. Is there any feasible way to compute it exactly?)

370. (a) Bipartiteness is obvious. Regularity is readily verified (but *not* obvious).

(b) The 48 symmetries of the cube all apply, leaving at most four equivalence classes of vertices, represented by vertices $\{000, 001, 011, 111\}$. And there are no further automorphisms, because no automorphism takes any of those vertices into another: The number of vertices at distance 2 from (000, 001, 011, 111) is respectively (16, 21, 22, 22); and 111 is at distance 2 from a corner vertex, but 011 isn't.

(c) (Solution by E. Weisstein.) Hamiltonian cycles are so abundant that we can simply choose one at random, and remove it to obtain a 4-regular graph; then partition those residual edges into two cycles. The following solution needed only a few trials:

(000 003 222 030 211 333 303 111 323 201 020 023 231 313 121 302 002 210 022 203 200 122
310 232 202 320 112 230 312 100 103 133 011 223 101 131 213 032 220 012 130 322 110 113
332 120 301 001 031 331 123 311 233 021 321 102 132 010 013 221 033 212 330 300)
(000 030 033 333 112 300 121 203 321 133 312 120 002 223 220 001 222 010 202 021 200 012
231 201 123 302 332 210 213 331 110 032 211 003 303 221 103 322 022 100 130 311 132 320
101 023 323 131 013 313 310 102 020 232 011 230 233 111 330 122 301 113 031 212)
(000 122 303 300 222 100 321 113 231 010 310 131 312 012 233 203 011 311 103 021 213 001
123 120 202 023 211 133 130 212 020 320 323 102 220 302 110 232 013 201 022 230 200 322
101 313 132 210 031 223 301 331 112 030 330 333 121 003 033 111 332 032 002 221)

(With so many cycles, could this graph perhaps even be *perfectly* Hamiltonian, in the sense of exercise 24? No! A. Kotzig [*Časopis pro Pěst. Mat.* **83** (1958), 348–354] proved that every n-vertex perfectly Hamiltonian regular bipartite graph has $n \bmod 4 = 2$.)

[See the "Sticky Chain" puzzle in S. Grabarchuk, *Age of Puzzles: Puzzle Galleries* (2019), 170, which asks for the longest path or cycle that doesn't touch or cross itself.]

371. (a) When $2n+1 = p$ is prime, it's easy: Use the cycles $0 — k — 2k \bmod p — \cdots — 2nk \bmod p — 0$ for $1 \le k \le n$. For arbitrary n an elegant construction by F. Walecki [presented by É. Lucas in his *Récréations Mathématiques* **2** (Paris, 1883), 162] is also easy once discovered: $0 — a_k — a_{k+1} — a_{k-1} — a_{k+2} — \cdots — a_{k-n+1} — a_{k+n} — 0$, for $0 \le k < n$, where $a_k = 1+(k \bmod 2n)$.

[We saw K_{11} decomposed into eleven 5-cycles in exercise 7.2.1.3–106; now we've decomposed it into five 11-cycles, in two different ways. Considerably more is actually true: It's always possible to decompose K_{2n+1} into t cycles of respective lengths l_1, ..., l_t, whenever $3 \le l_j \le 2n+1$ and $l_1+\cdots+l_t = \binom{2n+1}{2}$! See D. Bryant, D. Horsley, and W. Pettersson, *Proc. London Math. Soc.* (3) **108** (2014), 1153–1192.]

meander friezes
Warnsdorf's rule
parity
Vandermonde
Historical remarks
symmetries of the cube
hyperoctahedral symmetries
$\mathcal{B}_3$
Weisstein
P1F
perfectly
Kotzig
Sticky Chain
Grabarchuk
Walecki
Lucas
ultimate generalization
Bryant
Horsley
Pettersson

(b) The 4-regular graph $C_m \mathbin{\square} C_n$ has horizontal edges h_{ij} $(ij \mathrel{—} i(j{+}1 \bmod n))$ and vertical edges v_{ij} $(ij \mathrel{—} (i{+}1 \bmod m)j)$, for $0 \le i < m$ and $0 \le j < n$.

When m and n are both even, let one cycle consist of the following mn edges: h_{0j}, j odd; h_{1j}, j even; h_{i0}, $i > 1$; v_{i0}, i odd; v_{i1}, i even; v_{ij}, $i \ne 0$ and $j > 1$. The other cycle consists of the other mn edges.

When m and n are both odd, let one cycle be: h_{01}, h_{10}, h_{11}, h_{21}, v_{02}; h_{2j}, j even > 0; h_{3j}, j odd; h_{i1}, $i > 3$; v_{i0}, $i \ne 1$; v_{i1}, i even > 0; v_{i2}, i odd > 1; v_{ij}, $i \ne 2$ and $j > 2$. (When $m = n = 3$ the cycles are 3×3 *meanders*, at right angles to each other!)

When m is odd and n is even, the first cycle is: h_{01}, h_{11}, h_{22}, v_{00}, v_{03}; h_{0j}, $j > 2$; h_{1j}, j even > 0; h_{2j}, j odd > 1; h_{ij}, $i > 2$, $j > 1$; v_{i0}, i odd; v_{i1}, $i > 0$; v_{i2}, i even > 0; v_{1j}, $j > 3$. (The decomposition for $(m, n) = (3, 4)$ is essentially unique.)

[The case $m = n$ was proved by B. R. Myers, *Networks* **2** (1972), 1–9, Lemma 1. After A. Kotzig had established the general result in unpublished notes (1973), M. F. Foregger supplied a proof in *Discrete Math.* **24** (1978), 251–260.]

(c) Let the edges of G be those of $\Gamma \cup C_m$, where Γ is an m-cycle $0 \mathrel{—} v_1 \mathrel{—} \cdots \mathrel{—} v_{m-1} \mathrel{—} 0$ with $a = v_1 < v_{m-1} = b$, having no common edges with C_m. (Hence $2 \le a < b \le m{-}2$ and $m \ge 5$.) Then $G \mathbin{\square} C_n$ contains the following Hamiltonian cycle H when n is even: $00 \mathrel{—} b0 \mathrel{—} \cdots \mathrel{—} a0 \mathrel{—} a1 \mathrel{—} \cdots \mathrel{—} b1 \mathrel{—} b2 \mathrel{—} \cdots \mathrel{—} a2 \mathrel{—} \cdots \mathrel{—} a(n{-}2) \mathrel{—} a(n{-}1) \mathrel{—} \cdots \mathrel{—} b(n{-}1) \mathrel{—} 0(n{-}1) \mathrel{—} \cdots \mathrel{—} 01 \mathrel{—} 00$. A similar cycle H, ending with $b(n{-}2) \mathrel{—} b(n{-}1) \mathrel{—} \cdots \mathrel{—} a(n{-}1) \mathrel{—} 0(n{-}1) \mathrel{—} \cdots \mathrel{—} 01 \mathrel{—} 00$, works when n is odd. The 4-regular residual graph $G \mathbin{\square} C_m \setminus H$ can be called $R_{m,n,a,b}$, because its edges depend only on m, n, a, and b—not on the other elements $\{v_2, \ldots, v_{m-2}\}$ of the cycle Γ. In *Discrete Mathematics* **38** (1982), 7–16, J. Aubert and B. Schneider proved that every such graph $R_{m,n,a,b}$ can be decomposed into two Hamiltonian cycles, via a recursive construction that extends one of 16 base cases, two rows or two columns at a time.

(d) When $r = 2$, the cases $r' = 2$ and $r' = 4$ are (b) and (c).

When $r > 2$, notice that we actually have an improved law $(G \cup G') \mathbin{\square} (H \cup H') = (G \mathbin{\square} H) \cup (G' \mathbin{\square} H')$, because $G' \mathbin{\square} H \subseteq (G \mathbin{\square} H) \cup (G' \mathbin{\square} H')$. (If $uv \mathrel{—} u'v'$ then $u = u'$ or $v = v'$.) Also $G \mathbin{\square} H$ and $G' \mathbin{\square} H'$ have no common edges when $G \cap G' = H \cap H' = \emptyset$.

Let the decompositions of G and G' be $H_1 \cup \cdots \cup H_{r/2}$ and $H'_1 \cup \cdots \cup H'_{r'/2}$. Then

$$G \mathbin{\square} G' = \bigcup_{i=1}^{k} (H_i \mathbin{\square} H'_i) \cup \bigcup_{i=1}^{(r'-r)/2} (H_{i+k} \mathbin{\square} (H'_{k+2i-1} \cup H'_{k+2i})), \quad \text{where } k = r - r'/2,$$

is a decomposition into $2k{+}3(r'{-}r)/2 = (r{+}r')/2$ disjoint Hamiltonian cycles.

(e) When $t > 2$ it's $G \mathbin{\square} G'$, where $G = C_{m_1} \mathbin{\square} \cdots \mathbin{\square} C_{m_{\lfloor t/2 \rfloor}}$ and $G' = C_{m_{\lfloor t/2 \rfloor + 1}} \mathbin{\square} \cdots \mathbin{\square} C_{m_t}$ are $2\lfloor t/2 \rfloor$-regular and $2\lceil t/2 \rceil$-regular, both decomposable by induction.

372. (a) In one cycle let $ij \longrightarrow ((i{+}1) \bmod n)j$ when $i{+}j \ne n{-}1$, otherwise $ij \longrightarrow i((j{+}1) \bmod n)$. In the other cycle swap the conditions. (These cycles correspond to two versions of the "modular Gray n-ary code" for pairs, discussed in Section 7.2.1.1. See I. Darijani, B. Miraftab, and D. W. Morris, *Ars Mathematica Contemporanea* **25** (2025), #P2.10:1–32 for generalizations.)

(b) $u_0 \longrightarrow v_0 \longrightarrow u_1 \longrightarrow v_1 \longrightarrow u_2 \longrightarrow v_2 \longrightarrow u_0$; $u_0 \longrightarrow v_1 \longrightarrow u_1 \longrightarrow v_2 \longrightarrow u_2 \longrightarrow v_0 \longrightarrow u_0$; $u_0 \longrightarrow v_2 \longrightarrow u_1 \longrightarrow v_0 \longrightarrow u_2 \longrightarrow v_1 \longrightarrow u_0$.

(c) An exact cover problem whose 11502 options correspond to the directed Hamiltonian cycles of $\vec{C_3} \mathbin{\square} \vec{C_3} \mathbin{\square} \vec{C_3}$ reveals that there are exactly 4554 solutions.

Many of those solutions can actually be generalized, giving decompositions of $\vec{C_m} \mathbin{\square} \vec{C_m} \mathbin{\square} \vec{C_m}$ into three oriented m^3-cycles, for all odd values of $m \ge 3$. For example,

4-regular graph
meanders
Myers
Kotzig
Foregger
historical notes
Aubert
Schneider
modular Gray n-ary code
Darijani
Miraftab
Morris
exact cover problem

let three subgraphs S_0, S_1, S_2 be defined as follows: Given a vertex ijk, with $0 \le i,j,k < m$, set $s \leftarrow (i+j+k) \bmod m$. If $s = 0$, set $d \leftarrow (j = m-1?\ 012 : 210)$; if $s = m-1$, set $d \leftarrow (i = 0?\ 210 : 120)$; otherwise set $d \leftarrow (i = m-1?\ 201 : 102)$. For $0 \le c < 3$, put the arc $ijk \longrightarrow i^+jk$ into S_c, if $d_c = 0$. (Here $d = d_0d_1d_2$, and i^+ denotes $(i+1) \bmod m$.) Put the arc $ijk \longrightarrow ij^+k$ into S_c if $d_c = 1$. Put the arc $ijk \longrightarrow ijk^+$ into S_c if $d_c = 2$. Each subgraph S_c will be a Hamiltonian cycle(!). [This amazing construction was discovered by "Claude Opus 4.6," after coaching by Filip Stappers; and further coaching by Keston Aquino-Michaels led to a construction for all *even* values of $m \ge 4$. See `https://cs.stanford.edu/~knuth/papers/claude-cycles.pdf`.]

Claude
Stappers
Aquino-Michaels

(d) Indeed, J. Aubert and B. Schneider [*J. Combinatorial Theory* **B32** (1982), 347–349] have proved that the symmetric digraph obtained from any cubic n-vertex graph is Hamilton decomposable only if $n \bmod 4 = 2$ (as it was in (b)).

Aubert
Schneider
cubic

(e) R. Stong [*Discrete Mathematics* **306** (2006), 2186–2204] has found these five:

Stong

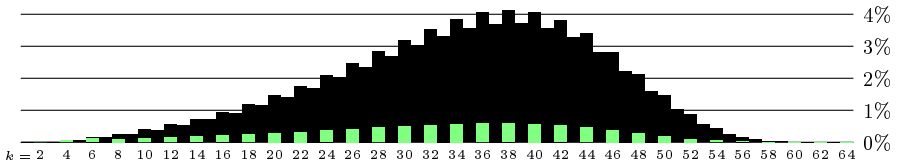
```
(00 08 0a 02 06 07 03 13 12 16 14 15 17 1f 1e 1a 1b 19 11 10 18 1c 1d 0d 0c 0e 0f 0b 09 01 05 04)
(00 01 03 0b 0f 0d 09 19 1d 1f 1b 13 11 15 05 07 17 16 12 1a 1e 1c 14 04 06 02 0a 0e 0c 08 18 10)
(00 10 12 02 03 01 11 13 17 07 05 0d 0f 1f 1d 15 14 16 06 04 0c 1c 1e 0e 0a 0b 1b 1a 18 19 09 08)
(00 04 05 15 11 01 09 0b 0a 1a 12 10 14 1c 18 08 0c 0d 1d 19 1b 1f 0f 07 06 0e 1e 16 17 13 03 02)
(00 02 12 13 1b 0b 03 07 0f 0e 06 16 1e 1f 17 15 1d 1c 0c 04 14 10 11 19 18 1a 0a 08 09 0d 05 01)
```

He proved in fact that the directed n-cube $Q_n^{\leftrightarrow}$ is Hamilton decomposable for all $n \ge 4$.

directed n-cube
n-cube

375. (a) One billion trials yield the following interesting probability distribution:

probability distribution



(b) The ten essentially different final squares (00, 01, 02, 03; 11, 12, 13; 22, 23; 33) have probabilities $\approx$ (.0845, .0369, .0118, .0216; .0123, .0005, .0030; .0002, .0009; .0035).

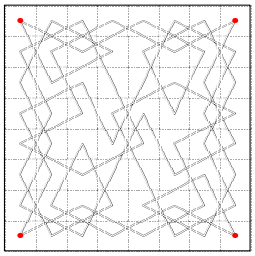
(c) When k is even, the probability of a k-cycle has been "hollowed out" in the histogram above. The exact totals, in the billion trials used for this illustration, were 595532 when $k = 4$ (595532 cycles) and 157 when $k = 64$ (17 cycles).

[Wolf carried out 1000 experiments by hand, each taking less than 15 minutes on average, and described them in admirable detail in a posthumous contribution to the *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich* **39** (1894), 147–164. Not surprisingly, however, his tables contained dozens of typographic and/or parity errors. Notice that the probability of ending with $k = 4$ is exactly 1/1680. Is it feasible to calculate the exact probability distribution for all k, with something like Algorithm E?]

historical note
parity

999. ...

ANSWERS TO PUZZLES IN THE ANSWERS



(see answer 158)

n_{00} R_{21} q_{11} K_{20} r_{31} Q_{41} r_{42} R_{40} k_{44} B_{34} q_{12} B_{02} r_{35} Q_{25} b_{45} R_{23} n_{43} K_{24} n_{33} B_{14} b_{32} N_{10} k_{22} N_{13} q_{01} K_{04} k_{05} N_{15} b_{03} Q_{30} n_{00} (see answer 220)

B_{00} N_{33} K_{12} N_{23} R_{02} N_{32} K_{13} N_{22} B_{03} N_{30} K_{11} N_{20} Q_{01} N_{31} K_{10} N_{21} B_{00} (see answer 221)

All art — symphonies, architecture, novels — it's all puzzles.
— STEPHEN SONDHEIM, *The New Yorker* (8 March 1993)

INDEX AND GLOSSARY

Florus, Lucius

Read the table that follows, my honest reader,
and it will soon guide you to hold the entire work in your mind.
— *Lucii Flori Bellorum Romanorum libri quattuor* (Vienna, 1511)

When an index entry refers to a page containing a relevant exercise, see also the *answer* to that exercise for further information. An answer page is not indexed here unless it refers to a topic not included in the statement of the exercise.

$\bar{1}$, $\bar{2}$, ...: Another way to write -1, -2, ...

2-factor, *see* Cycle cover.

Articulation point: A vertex whose removal increases the number of components of a graph.

Barry, David McAlister (= Dave), iii.

Biconnected graph: A graph without articulation points.

Cycle cover: A covering of the vertices by disjoint cycles (a 2-regular spanning subgraph, if undirected).

Cyclically k-connected: Must remove at least k edges to obtain two cyclic (nontree) components.

FGbook: *Selected Papers on Fun & Games*, a book by D. E. Knuth.

Hamiltonian graph: A graph with a spanning cycle, 2.

LIPIcs: *Leibniz International Proceedings in Informatics* (2008–).

Oniţiu, Valeriu (= Valerian).

simplex graphs, 127.

Triangular grid of order n: The SGB graph *simplex*$(n, -2, 0, 0, 0, 0, 0)$.

Nothing else is indexed yet (sorry).

Preliminary notes for indexing appear in the upper right corner of most pages.

If I've mentioned somebody's name and forgotten to make such an index note, it's an error (worth \$2.56).