THE ART OF COMPUTER PROGRAMMING

VOLUME 4   PRE-FASCICLE 0B

A DRAFT OF SECTION 7.1.1: BOOLEAN BASICS

DONALD E. KNUTH *Stanford University*

ADDISON–WESLEY

Internet
Stanford GraphBase
MMIX

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

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

Zeroth printing (revision 14), 30 January 2008

Internet
Stanford GraphBase

PREFACE

Bitte ein Bit!
— Slogan of Bitburger Brauerei (1951)

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, and 3 were at the time of their first printings. And those carefully-checked volumes, alas, 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 so vast, I cannot hope to have surrounded it enough to corral it completely. Therefore I beg you to let me know about any deficiencies that you discover.

To put the material in context, this pre-fascicle contains Section 7.1.1 of a long, long chapter on combinatorial algorithms. Chapter 7 will eventually fill at least three volumes (namely Volumes 4A, 4B, and 4C), assuming that I'm able to remain healthy. It will begin with a short review of graph theory, with emphasis on some highlights of significant graphs in the Stanford GraphBase, from which I will be drawing many examples. Then comes Section 7.1: Zeros and Ones, beginning with the stuff you're about to read here. Section 7.1.2 will deal with efficient Boolean function evaluation; Section 7.1.3 will deal with tricks and techniques of bitwise calculation; Section 7.1.4 will discuss data structures for Boolean functions.

The next section, 7.2, is about generating all possibilities, and it begins with Section 7.2.1: Generating Basic Combinatorial Patterns. Fascicles for this section have already appeared on the Web and/or in print. Section 7.2.2 will deal with backtracking in general. And so it will go on, if all goes well; an outline of the entire Chapter 7 as currently envisaged appears on the `taocp` webpage that is cited on page ii.

The topic of Boolean functions and bit manipulation can of course be interpreted so broadly that it encompasses the entire subject of computer programming. My original title for Section 7.1 — "Bit Fiddling" — was much more modest; I decided, however, that those words were a bit too low-brow. The real goal of this fascicle is to focus on concepts that appear at the lowest levels, on which we can erect significant superstructures. And even these apparently lowly notions turn out to be surprisingly rich, with explicit ties to Sections 1.2.3, 1.2.6, 2.2.1, 2.2.3, 2.3.3, 3.3.2, 4.6, 4.6.1, 4.6.4, 5, 5.3.4, and 6.5 of the first three volumes. I strongly believe in building up a firm foundation, so I have discussed Boolean topics much more thoroughly than I will be able to do with material that is newer or less basic. After typing the manuscript I was astonished to discover that I had come up with 133 exercises, even though — believe it or not — I had to eliminate quite a lot of the interesting material that appears in my files.

Schensted
Knuth
Floyd

My notes on combinatorial algorithms have been accumulating for more than forty years, so I fear that in several respects my knowledge is woefully behind the times. Please look, for example, at the exercises that I've classed as research problems (rated with difficulty level 46 or higher), namely exercises 68, 91, 92, 119, and 123; I've also implicitly mentioned or posed additional unsolved questions in the answers to exercises 27 and 95. Are those problems still open? Please let me know 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 15, 25, 34(b,c,d), 41 (Solution 2), 47, 52, 83, 98, 106, 109(g,h), 118, 121, 122, 126(c), 127, and 128, or the remark in the answer to exercise 125. Furthermore I've credited exercises 62, 67, and 69 to unpublished work of Craige Schensted. Have any of those results appeared in print, to your knowledge?

I shall happily pay 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 reward you with immortal glory instead of mere money, 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.
05 September 2005

P.S. I would like to express my indebtedness to the late Robert W Floyd, who made dozens of valuable suggestions when I asked him to look over the first draft of this material in 1977.

Yet now and then your men of wit
Will condescend to take a bit.
— JONATHAN SWIFT, *Cadenus and Vanessa* (1713)

If the base 2 is used the resulting units may be called binary digits,
or more briefly bits, *a word suggested by J. W. Tukey.*
— CLAUDE E. SHANNON, in *Bell System Technical Journal* (1948)

bit (bit), n *. . . [A] boring tool . . .*
— *Random House Dictionary of the English Language* (1987)

SWIFT
Tukey
SHANNON
Dictionary
binary number system
bitwise
binary operations–
true
truth table

7.1. ZEROS AND ONES

Combinatorial algorithms often require special attention to efficiency, and the proper representation of data is an important way to gain the necessary speed. It is therefore wise to beef up our knowledge of elementary representation techniques before we set out to study combinatorial algorithms in detail.

Most of today's computers are based on the binary number system, instead of working directly with the decimal numbers that human beings prefer, because machines are especially good at dealing with the two-state on-off quantities that we usually denote by the digits 0 and 1. But in Chapters 1 to 6 we haven't made much use of the fact that binary computers can do several things quickly that decimal computers cannot. A binary machine can usually perform "logical" or "bitwise" operations just as easily as it can add or subtract; yet we have seldom capitalized on that capability. We've seen that binary and decimal computers are not significantly different, for many purposes, but in a sense we've been asking a binary computer to operate with one hand tied behind its back.

The amazing ability of 0s and 1s to encode information as well as to encode the logical relations between items, and even to encode algorithms for processing information, makes the study of binary digits especially rich. Indeed, we not only use bitwise operations to enhance combinatorial algorithms, we also find that the properties of binary logic lead naturally to new combinatorial problems that are of great interest in their own right.

Computer scientists have gradually become better and better at taming the wild 0s and 1s of the universe and making them do useful tricks. But as bit players on the world's stage, we'd better have a thorough understanding of the low-level properties of binary quantities before we launch into a study of higher-level concepts and techniques. Therefore we shall start by investigating basic ways to combine individual bits and sequences of bits.

7.1.1. Boolean Basics

There are 16 possible functions $f(x,y)$ that transform two given bits x and y into a third bit $z = f(x,y)$, since there are two choices for each of $f(0,0)$, $f(0,1)$, $f(1,0)$, and $f(1,1)$. Table 1 indicates the names and notations that have traditionally been associated with these functions in studies of formal logic, assuming that 1 corresponds to "true" and 0 to "false." The sequence of four values $f(0,0)f(0,1)f(1,0)f(1,1)$ is customarily called the *truth table* of the function f.

Boole
Carroll
Tweedledee
Jevons
distributive law
notation
Peirce
Funk
Wagnalls
dictionary
Knuth, John Martin, see Truth
Greek

Let us conceive, then, of an Algebra
in which the symbols x, y, z, *&c. admit indifferently of*
the values 0 *and* 1, *and of these values alone.*
— GEORGE BOOLE, *An Investigation of the Laws of Thought* (1854)

'Contrariwise,' continued Tweedledee, 'if it was so, it might be;
and if it were so, it would be;
but as it isn't, it ain't. That's logic.'
— LEWIS CARROLL, *Through the Looking Glass* (1871)

Such functions are often called "Boolean operations" in honor of George Boole, who first discovered that algebraic operations on 0s and 1s could be used to construct a calculus for logical reasoning [*The Mathematical Analysis of Logic* (Cambridge: 1847); *An Investigation of the Laws of Thought* (London: 1854)]. But Boole never actually dealt with the "logical or" operation $\vee$; he confined himself strictly to ordinary arithmetic operations on 0s and 1s. Thus he would write $x+y$ to stand for disjunction, but he took pains never to use this notation unless x and y were mutually exclusive (not both 1). If necessary, he wrote $x+(1-x)y$ to ensure that the result of a disjunction would never be equal to 2.

When rendering the + operation in English, Boole sometimes called it "and," sometimes "or." This practice may seem strange to modern mathematicians until we realize that his usage was in fact normal English; we say, for example, that "boys and girls are children," but "children are boys or girls."

Boole's calculus was extended to include the unconventional rule $x+x=x$ by W. Stanley Jevons [*Pure Logic* (London: Edward Stanford, 1864), §69], who pointed out that $(x+y)z$ was equal to $xz+yz$ using his new + operation. But Jevons did not know the other distributive law $xy+z=(x+z)(y+z)$. Presumably he missed this because of the notation he was using, since the second distributive law has no familiar counterpart in arithmetic; the more symmetrical notations $x \wedge y$, $x \vee y$ in Table 1 make it easier for us to remember both distributive laws

$$(x \vee y) \wedge z \;=\; (x \wedge z) \vee (y \wedge z); \tag{1}$$

$$(x \wedge y) \vee z \;=\; (x \vee z) \wedge (y \vee z). \tag{2}$$

The second law (2) was introduced by C. S. Peirce, who had discovered independently how to extend Boole's calculus [*Proc. Amer. Acad. Arts and Sciences* **7** (1867), 250–261]. Incidentally, when Peirce discussed these early developments several years later [*Amer. J. Math.* **3** (1880), 32], he referred to "the Boolian algebra, with Jevons's addition"; his now-unfamiliar spelling of "Boolean" was in use for many years, appearing in the Funk and Wagnalls unabridged dictionary as late as 1963.

The notion of truth-value combination is actually much older than Boolean algebra. Indeed, propositional logic had been developed by Greek philosophers already in the fourth century B.C. There was considerable debate in those days about how to assign an appropriate true-or-false value to the proposition "if x then y" when x and y are propositions; Philo of Megara, about 300 B.C., defined

Table 1

THE SIXTEEN LOGICAL OPERATIONS ON TWO VARIABLES

Truth table	Notation(s)	Operator symbol $\circ$	Name(s)
0000	0	$\perp$	Contradiction; falsehood; antilogy; constant 0
0001	$xy,\ \ x \wedge y,\ \ x \mathbin{\&} y$	$\wedge$	Conjunction; and
0010	$x \wedge \bar{y},\ \ x \not\supset y,\ \ [x > y],\ \ x \mathbin{\dot{-}} y$	$\overline{\supset}$	Nonimplication; difference; but not
0011	x	$\llcorner$	Left projection
0100	$\bar{x} \wedge y,\ \ x \not\subset y,\ \ [x < y],\ \ y \mathbin{\dot{-}} x$	$\overline{\subset}$	Converse nonimplication; not ... but
0101	y	R	Right projection
0110	$x \oplus y,\ \ x \not\equiv y,\ \ x\,\hat{}\,y$	$\oplus$	Exclusive disjunction; nonequivalence; “xor”
0111	$x \vee y,\ \ x \mid y$	$\vee$	(Inclusive) disjunction; or; and/or
1000	$\bar{x} \wedge \bar{y},\ \ \overline{x \vee y},\ \ x \mathbin{\overline{\vee}} y,\ \ x \downarrow y$	$\overline{\vee}$	Nondisjunction; joint denial; neither ... nor
1001	$x \equiv y,\ \ x \leftrightarrow y,\ \ x \Leftrightarrow y$	$\equiv$	Equivalence; if and only if; “iff”
1010	$\bar{y},\ \ \neg y,\ \ !y,\ \ {\sim}y$	$\overline{\mathsf{R}}$	Right complementation
1011	$x \vee \bar{y},\ x \subset y,\ x \Leftarrow y,\ [x \geq y],\ x^y$	$\subset$	Converse implication; if
1100	$\bar{x},\ \ \neg x,\ \ !x,\ \ {\sim}x$	$\overline{\llcorner}$	Left complementation
1101	$\bar{x} \vee y,\ x \supset y,\ x \Rightarrow y,\ [x \leq y],\ y^x$	$\supset$	Implication; only if; if ... then
1110	$\bar{x} \vee \bar{y},\ \ \overline{x \wedge y},\ \ x \mathbin{\overline{\wedge}} y,\ \ x \mid y$	$\overline{\wedge}$	Nonconjunction; not both ... and; “nand”
1111	1	$\top$	Affirmation; validity; tautology; constant 1

it by the truth table shown in Table 1, which states in particular that the implication is true when both x and y are false. Much of this early work has been lost, but there are passages in the works of Galen (2nd century A.D.) that refer to both inclusive and exclusive disjunction of propositions. [See I. M. Bocheński, *Formale Logik* (1956), English translation by Ivo Thomas (1961), for an excellent survey of the development of logic from ancient times up to the 20th century.]

A function of two variables is often written $x \circ y$ instead of $f(x, y)$, using some appropriate operator symbol $\circ$. Table 1 shows the sixteen operator symbols that we shall adopt for Boolean functions of two variables; for example, $\perp$ symbolizes the function whose truth table is 0000, $\wedge$ is the symbol for 0001, $\overline{\supset}$ is the symbol for 0010, and so on. We have $x \perp y = 0$, $x \wedge y = xy$, $x \mathbin{\overline{\supset}} y = x \mathbin{\dot{-}} y$, $x \mathbin{\llcorner} y = x$, $\ldots$, $x \mathbin{\overline{\wedge}} y = \bar{x} \vee \bar{y}$, $x \top y = 1$.

Of course the operations in Table 1 aren't all of equal importance. For example, the first and last cases are trivial, since they have a constant value independent of x and y. Four of them are functions of x alone or y alone. We write $\bar{x}$ for $1 - x$, the *complement* of x.

The four operations whose truth table contains just a single 1 are easily expressed in terms of the AND operator $\wedge$, namely $x \wedge y$, $x \wedge \bar{y}$, $\bar{x} \wedge y$, $\bar{x} \wedge \bar{y}$. Those with three 1s are easily written in terms of the OR operator $\vee$, namely $x \vee y$, $x \vee \bar{y}$, $\bar{x} \vee y$, $\bar{x} \vee \bar{y}$. The basic functions $x \wedge y$ and $x \vee y$ have proved to be more useful in practice than their complemented or half-complemented cousins, although the NOR and NAND operations $x \mathbin{\overline{\vee}} y = \bar{x} \wedge \bar{y}$ and $x \mathbin{\overline{\wedge}} y = \bar{x} \vee \bar{y}$ are also of interest because they are easily implemented in transistor circuits.

Philo of Megara
truth table
Galen
Bocheński
Thomas
binary operations, table
contradiction
falsehood
antilogy
conjunction
nonimplication
left projection
converse nonimplication
right projection
exclusive disjunction
inclusive disjunction
nondisjunction
equivalence+
right complementation
converse implication
left complementation
implication
nonconjunction
affirmation
validity
tautology
projection functions
trivial function
complement
and
or
nand+
nor+
NOR
NAND

In 1913, H. M. Sheffer showed that all 16 of the functions can be expressed in terms of just one, starting with either $\overline{\vee}$ or $\overline{\wedge}$ as the given operation (see exercise 4). Actually C. S. Peirce had made the same discovery about 1880, but his work on the subject remained unpublished until after his death [*Collected Papers of Charles Sanders Peirce* **4** (1933), §§12–20, 264]. Table 1 indicates that NAND and NOR have occasionally been written $x \mid y$ and $x \downarrow y$; sometimes they have been called "Sheffer's stroke" and the "Peirce arrow." Nowadays it is best *not* to use Sheffer's vertical line for NAND, because $x \mid y$ denotes bitwise $x \vee y$ in programming languages like C.

So far we have discussed all but two of the functions in Table 1. The remaining two are $x \equiv y$ and $x \oplus y$, "equivalence" and "exclusive-or," which are related by the identities

$$x \equiv y \;=\; \bar{x} \oplus y \;=\; x \oplus \bar{y} \;=\; 1 \oplus x \oplus y; \tag{3}$$

$$x \oplus y \;=\; \bar{x} \equiv y \;=\; x \equiv \bar{y} \;=\; 0 \equiv x \equiv y. \tag{4}$$

Both operations are associative (see exercise 6). In propositional logic, the notion of equivalence is more important than the notion of exclusive-or, which means inequivalence; but when we consider bitwise operations on full computer words, we shall see in Section 7.1.3 that the situation is reversed: Exclusive-or turns out to be more useful than equivalence, in typical programs. The chief reason why $x \oplus y$ has significant applications, even in the one-bit case, is the fact that

$$x \oplus y \;=\; (x + y) \bmod 2. \tag{5}$$

Therefore $x \oplus y$ and $x \wedge y$ denote addition and multiplication in the field of two elements (see Section 4.6), and $x \oplus y$ naturally inherits many "clean" mathematical properties.

Basic identities. Now let's take a look at interactions between the fundamental operators $\wedge$, $\vee$, $\oplus$, and $^{-}$, since the other operations are easily expressed in terms of these four. Each of $\wedge$, $\vee$, $\oplus$ is associative and commutative. Besides the distributive laws (1) and (2), we also have

$$(x \oplus y) \wedge z \;=\; (x \wedge z) \oplus (y \wedge z), \tag{6}$$

as well as the *absorption laws*

$$(x \wedge y) \vee x \;=\; (x \vee y) \wedge x \;=\; x. \tag{7}$$

One of the simplest, yet most useful, identities is

$$x \oplus x \;=\; 0, \tag{8}$$

since it implies among other things that

$$(x \oplus y) \oplus x \;=\; y, \qquad (x \oplus y) \oplus y \;=\; x, \tag{9}$$

when we use the obvious fact that $x \oplus 0 = x$. In other words, given $x \oplus y$ and either x or y, it is easy to determine the other. And let us not overlook the simple *complementation law*

$$\bar{x} \;=\; x \oplus 1. \tag{10}$$

Sheffer
Peirce
bitwise or
xor
exclusive-or
associative
field
associative
commutative
distributive laws
absorption laws
complementation law

Another important pair of identities is known as *De Morgan's laws* in honor of Augustus De Morgan, who stated that "The contrary of an aggregate is the compound of the contraries of the aggregants; the contrary of a compound is the aggregate of the contraries of the components. Thus (A, B) and AB have ab and (a, b) for contraries." [*Trans. Cambridge Philos. Soc.* **10** (1858), 208.] In more modern notation, these are the rules we have implicitly derived via truth tables in connection with the operations NAND and NOR in Table 1, namely

$$\overline{x \wedge y} \;=\; \bar{x} \vee \bar{y}; \tag{11}$$
$$\overline{x \vee y} \;=\; \bar{x} \wedge \bar{y}. \tag{12}$$

Incidentally, W. S. Jevons knew (12) but not (11); he consistently wrote $\bar{A}B + \bar{B}A + \bar{A}\bar{B}$ instead of $\bar{A} + \bar{B}$ for the complement of AB. Yet De Morgan was not the first Englishman who enunciated the laws above. Both (11) and (12) can be found in the early 14th century writings of two scholastic philosophers, William of Ockham [*Summa Logicæ* **2** (1323)] and Walter Burley [*De Puritate Artis Logicæ* (c. 1330)].

De Morgan's laws and a few other identities can be used to express $\wedge$, $\vee$, and $\oplus$ in terms of each other:

$$x \wedge y \;=\; \overline{\bar{x} \vee \bar{y}} \;=\; x \oplus y \oplus (x \vee y); \tag{13}$$
$$x \vee y \;=\; \overline{\bar{x} \wedge \bar{y}} \;=\; x \oplus y \oplus (x \wedge y); \tag{14}$$
$$x \oplus y \;=\; (x \vee y) \wedge \overline{x \wedge y} \;=\; (x \wedge \bar{y}) \vee (\bar{x} \wedge y). \tag{15}$$

According to exercise 7.1.2–77, all computations of $x_1 \oplus x_2 \oplus \cdots \oplus x_n$ that use only the operations $\wedge$, $\vee$, and $\bar{\ }$ must be at least $4(n-1)$ steps long; thus, the other three operations are not an especially good substitute for $\oplus$.

Functions of n variables. A Boolean function $f(x, y, z)$ of three Boolean variables x, y, z can be defined by its 8-bit truth table $f(0,0,0)f(0,0,1)\ldots f(1,1,1)$; and in general, every n-ary Boolean function $f(x_1, \ldots, x_n)$ corresponds to a 2^n-bit truth table that lists the successive values of $f(0,\ldots,0,0)$, $f(0,\ldots,0,1)$, $f(0,\ldots,1,0)$, $\ldots$, $f(1,\ldots,1,1)$.

We needn't devise special names and notations for all these functions, since they can all be expressed in terms of the binary functions that we've already learned. For example, as observed by I. I. Zhegalkin [*Matematicheskiĭ Sbornik* **35** (1928), 311–369], we can always write

$$f(x_1, \ldots, x_n) \;=\; g(x_1, \ldots, x_{n-1}) \;\oplus\; h(x_1, \ldots, x_{n-1}) \wedge x_n \tag{16}$$

when $n > 0$, for appropriate functions g and h, by letting

$$\begin{aligned} g(x_1, \ldots, x_{n-1}) &= f(x_1, \ldots, x_{n-1}, 0); \\ h(x_1, \ldots, x_{n-1}) &= f(x_1, \ldots, x_{n-1}, 0) \oplus f(x_1, \ldots, x_{n-1}, 1). \end{aligned} \tag{17}$$

(The operation $\wedge$ conventionally takes precedence over $\oplus$, so we need not use parentheses to enclose the subformula '$h(x_1, \ldots, x_{n-1}) \wedge x_n$' on the right-hand side of (16).) Repeating this process recursively on g and h until we're down to

De Morgan's laws
truth tables
Jevons
Ockham
Burley
n-ary Boolean function
truth table
Zhegalkin
Gégalkine (= Zhegalkin), Jean Jean (and give th
precedence

0-ary functions leaves us with an expression that involves only the operators $\oplus$, $\wedge$, and a sequence of 2^n constants. Furthermore, those constants can usually be simplified away, because we have

$$x \wedge 0 = 0 \qquad \text{and} \qquad x \wedge 1 = x \oplus 0 = x. \tag{18}$$

After applying the associative and distributive laws, we end up needing the constant 0 only if $f(x_1, \ldots, x_n)$ is identically zero, and the constant 1 only if $f(0, \ldots, 0) = 1$.

We might have, for instance,

$$\begin{aligned} f(x,y,z) &= \big((1 \oplus 0 \wedge x) \oplus (0 \oplus 1 \wedge x) \wedge y\big) \oplus \big((0 \oplus 1 \wedge x) \oplus (1 \oplus 1 \wedge x) \wedge y\big) \wedge z \\ &= (1 \oplus x \wedge y) \;\oplus\; (x \oplus y \oplus x \wedge y) \wedge z \\ &= 1 \;\oplus\; x \wedge y \;\oplus\; x \wedge z \;\oplus\; y \wedge z \;\oplus\; x \wedge y \wedge z. \end{aligned}$$

And by rule (5), we see that we're simply left with the polynomial

$$f(x,y,z) \;=\; (1 + xy + xz + yz + xyz) \bmod 2, \tag{19}$$

because $x \wedge y = xy$. Notice that this polynomial is linear (of degree ≤ 1) in each of its variables. In general, a similar calculation will show that *any* Boolean function $f(x_1, \ldots, x_n)$ has a unique representation such as this, called its *multilinear representation*, which is a sum (modulo 2) of zero or more of the 2^n possible terms 1, x_1, x_2, x_1x_2, x_3, x_1x_3, x_2x_3, $x_1x_2x_3$, $\ldots$, $x_1x_2 \ldots x_n$.

George Boole decomposed Boolean functions in a different way, which is often simpler for the kinds of functions that arise in practice. Instead of (16), he essentially wrote

$$f(x_1, \ldots, x_n) \;=\; \big(g(x_1, \ldots, x_{n-1}) \wedge \bar{x}_n\big) \;\vee\; \big(h(x_1, \ldots, x_{n-1}) \wedge x_n\big) \tag{20}$$

and called it the "law of development," where we now have simply

$$\begin{aligned} g(x_1, \ldots, x_{n-1}) &= f(x_1, \ldots, x_{n-1}, 0), \\ h(x_1, \ldots, x_{n-1}) &= f(x_1, \ldots, x_{n-1}, 1), \end{aligned} \tag{21}$$

instead of (17). Repeatedly iterating Boole's procedure, using the distributive law (1), and eliminating constants, leaves us with a formula that is a disjunction of zero or more *minterms*, where each minterm is a conjunction such as $x_1 \wedge \bar{x}_2 \wedge \bar{x}_3 \wedge x_4 \wedge x_5$ in which every variable or its complement is present. Notice that a minterm is a Boolean function that is true at exactly one point.

For example, let's consider the more-or-less random function $f(w,x,y,z)$ whose truth table is

$$1100\,1001\,0000\,1111. \tag{22}$$

When this function is expanded by repeatedly applying Boole's law (20), we get a disjunction of eight minterms, one for each of the 1s in the truth table:

$$\begin{aligned} f(w,x,y,z) \;=\; & (\bar{w} \wedge \bar{x} \wedge \bar{y} \wedge \bar{z}) \vee (\bar{w} \wedge \bar{x} \wedge \bar{y} \wedge z) \vee (\bar{w} \wedge x \wedge \bar{y} \wedge \bar{z}) \vee (\bar{w} \wedge x \wedge y \wedge z) \\ & \vee (w \wedge x \wedge \bar{y} \wedge \bar{z}) \vee (w \wedge x \wedge \bar{y} \wedge z) \vee (w \wedge x \wedge y \wedge \bar{z}) \vee (w \wedge x \wedge y \wedge z). \end{aligned} \tag{23}$$

linear
multilinear representation
Boole
development
minterms
truth table
pi, as "random" example

In general, a disjunction of minterms is called a *full disjunctive normal form.* Every Boolean function can be expressed in this way, and the result is unique — except, of course, for the order of the minterms. *Nitpick:* A special case arises when $f(x_1,\ldots,x_n)$ is identically zero. We consider '0' to be an empty disjunction, with no terms, and we also consider '1' to be an empty conjunction, for the same reasons as we defined $\sum_{k=1}^{0} a_k = 0$ and $\prod_{k=1}^{0} a_k = 1$ in Section 1.2.3.

C. S. Peirce observed, in *Amer. J. Math.* **3** (1880), 37–39, that every Boolean function also has a *full conjunctive normal form*, which is a conjunction of "minclauses" like $\bar{x}_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4 \vee x_5$. A minclause is 0 at only one point; so each clause in such a conjunction accounts for a place where the truth table has a 0. For example, the full conjunctive normal form of our function in (22) and (23) is

$$\begin{aligned} f(w,x,y,z) \;=\; &(w\vee x\vee\bar{y}\vee z)\wedge(w\vee x\vee\bar{y}\vee\bar{z})\wedge(w\vee\bar{x}\vee y\vee\bar{z})\wedge(w\vee\bar{x}\vee\bar{y}\vee z)\\ &\wedge(\bar{w}\vee x\vee y\vee z)\wedge(\bar{w}\vee x\vee y\vee\bar{z})\wedge(\bar{w}\vee x\vee\bar{y}\vee z)\wedge(\bar{w}\vee x\vee\bar{y}\vee\bar{z}). \qquad (24)\end{aligned}$$

Not surprisingly, however, we often want to work with disjunctions and conjunctions that *don't* necessarily involve full minterms or minclauses. Therefore, following nomenclature introduced by Paul Bernays in his *Habilitationsschrift* (1918), we speak in general of a *disjunctive normal form* or "DNF" as *any* disjunction of conjunctions,

$$\bigvee_{j=1}^{m}\bigwedge_{k=1}^{s_j} u_{jk} \;=\; (u_{11}\wedge\cdots\wedge u_{1s_1})\vee\cdots\vee(u_{m1}\wedge\cdots\wedge u_{ms_m}), \qquad (25)$$

where each u_{jk} is a *literal*, namely a variable x_i or its complement. Similarly, any conjunction of disjunctions of literals,

$$\bigwedge_{j=1}^{m}\bigvee_{k=1}^{s_j} u_{jk} \;=\; (u_{11}\vee\cdots\vee u_{1s_1})\wedge\cdots\wedge(u_{m1}\vee\cdots\vee u_{ms_m}), \qquad (26)$$

is called a *conjunctive normal form*, or "CNF" for short.

A great many electrical circuits embedded inside today's computer chips are composed of "programmable logic arrays" (PLAs), which are ORs of ANDs of possibly complemented input signals. In other words, a PLA basically computes one or more disjunctive normal forms. Such building blocks are fast, versatile, and relatively inexpensive; and indeed, DNFs have played a prominent role in electrical engineering ever since the 1950s, when switching circuits were implemented with comparatively old-fashioned devices like relays or vacuum tubes. Therefore people have long been interested in finding the simplest DNFs for classes of Boolean functions, and we can expect that an understanding of disjunctive normal forms will continue to be important as technology continues to evolve.

The terms of a DNF are often called *implicants*, because the truth of any term in a disjunction implies the truth of the whole formula. In a formula like

$$f(x,y,z) \;=\; (x\wedge\bar{y}\wedge z)\vee(y\wedge z)\vee(\bar{x}\wedge y\wedge\bar{z}),$$

for example, we know that f is true when $x\wedge\bar{y}\wedge z$ is true, namely when $(x,y,z) = (1,0,1)$. But notice that in this example the shorter term $x\wedge z$ also turns out to be an implicant of f, even though not written explicitly, because the additional term $y \wedge z$ makes the function true whenever $x = z = 1$, regardless of the value of y. Similarly, $\bar{x} \wedge y$ is an implicant of this particular function. So we might as well work with the simpler formula

$$f(x, y, z) \;=\; (x \wedge z) \vee (y \wedge z) \vee (\bar{x} \wedge y). \qquad (27)$$

At this point no more deletions are possible within the implicants, because neither x nor y nor z nor $\bar{x}$ is a strong enough condition to imply the truth of f.

An implicant that can't be factored further by removing any of its literals without making it too weak is called a *prime implicant*, following the terminology of W. V. Quine in *AMM* **59** (1952), 521–531.

These basic concepts can perhaps be understood most easily if we simplify the notation and adopt a more geometric viewpoint. We can write simply '$f(x)$' instead of $f(x_1, \ldots, x_n)$, and regard x as a vector, or as a binary string $x_1 \ldots x_n$ of length n. For example, the strings $wxyz$ where the function of (22) is true are

$$\{0000, 0001, 0100, 0111, 1100, 1101, 1110, 1111\}, \qquad (28)$$

and we can think of them as eight points in the 4-dimensional hypercube $2 \times 2 \times 2 \times 2$. The eight points in (28) correspond to the minterm implicants that are explicitly present in the full disjunctive normal form (23); but none of those implicants is actually prime. For example, the first two points of (28) make the subcube $000*$, and the last four points constitute the subcube $11{*}{*}$, if we use asterisks to denote "wild cards" as we did when discussing database queries in Section 6.5; therefore $\bar{w} \wedge \bar{x} \wedge \bar{y}$ is an implicant of f, and so is $w \wedge x$. Similarly, we can see that the subcube $0{*}00$ accounts for two of the eight points in (28), making $\bar{w} \wedge \bar{y} \wedge \bar{z}$ an implicant.

In general, each prime implicant corresponds in this way to a *maximal* subcube that stays within the set of points that make f true. (The subcube is maximal in the sense that it isn't contained in any larger subcube with the same property; we can't replace any of its explicit bits by an asterisk. A maximal subcube has a maximal number of asterisks, hence a minimal number of constrained coordinates, hence a minimal number of variables in the corresponding implicant.) The maximal subcubes of the eight points in (28) are

$$000*,\ 0{*}00,\ {*}100,\ {*}111,\ 11{*}{*}; \qquad (29)$$

so the prime implicants of the function $f(w, x, y, z)$ in (23) are

$$(\bar{w} \wedge \bar{x} \wedge \bar{y}) \vee (\bar{w} \wedge \bar{y} \wedge \bar{z}) \vee (x \wedge \bar{y} \wedge \bar{z}) \vee (x \wedge y \wedge z) \vee (w \wedge x). \qquad (30)$$

The *disjunctive prime form* of a Boolean function is the disjunction of all its prime implicants. Exercise 30 contains an algorithm to find all the prime implicants of a given function, based on a list of the points where the function is true.

We can define a *prime clause* in an exactly similar way: It is a disjunctive clause that is implied by f, having no subclause with the same property. And the *conjunctive prime form* of f is the conjunction of all its prime clauses. (An example appears in exercise 19.)

prime implicant
Quine
vector
binary string
hypercube
subcube
maximal subcube
asterisks
disjunctive prime form
prime clause
clause: a disjunction of literals
conjunctive prime form

In many simple cases, the disjunctive prime form is the shortest possible disjunctive normal form that a function can have. But we can often do better, because we might be able to cover all the necessary points with only a few of the maximal subcubes. For example, the prime implicant $(y \wedge z)$ is unnecessary in (27). And in expression (30) we don't need both $(\bar{w} \wedge \bar{y} \wedge \bar{z})$ and $(x \wedge \bar{y} \wedge \bar{z})$; either one is sufficient, in the presence of the other terms.

Unfortunately, we will see in Section 7.9 that the task of finding a best disjunctive normal form is NP-complete, thus quite difficult in general. But many useful shortcuts have been developed for sufficiently small problems, and they are well explained in the book *Introduction to the Theory of Switching Circuits* by E. J. McCluskey (New York: McGraw–Hill, 1965). For later developments, see Petr Fišer and Jan Hlavička, *Computing and Informatics* **22** (2003), 19–51.

There's an important special case for which the shortest DNF is, however, easily characterized. A Boolean function is said to be *monotone* or *positive* if its value does not change from 1 to 0 when any of its variables changes from 0 to 1. In other words, f is monotone if and only if $f(x) \le f(y)$ whenever $x \subseteq y$, where the bit string $x = x_1 \ldots x_n$ is regarded as contained in or equal to the bit string $y = y_1 \ldots y_n$ if and only if $x_j \le y_j$ for all j. An equivalent condition (see exercise 21) is that the function f either is constant or can be expressed entirely in terms of $\wedge$ and $\vee$, without complementation.

Theorem Q. *The shortest disjunctive normal form of a monotone Boolean function is its disjunctive prime form.*

Proof. [W. V. Quine, *Boletín de la Sociedad Matemática Mexicana* **10** (1953), 64–70.] Let $f(x_1, \ldots, x_n)$ be monotone, and let $u_1 \wedge \cdots \wedge u_s$ be one of its prime implicants. We cannot have, say, $u_1 = \bar{x}_i$, because in that case the shorter term $u_2 \wedge \cdots \wedge u_s$ would also be an implicant, by monotonicity. Therefore no prime implicant has a complemented literal.

Now if we set $u_1 \leftarrow \cdots \leftarrow u_s \leftarrow 1$ and all other variables to 0, the value of f will be 1, but all of f's other prime implicants will vanish. Thus $u_1 \wedge \cdots \wedge u_s$ must be in every shortest DNF, because every implicant of a shortest DNF is clearly prime. ▌

Corollary Q. *A disjunctive normal form is the disjunctive prime form of a monotone Boolean function if and only if it has no complemented literals and none of its implicants is contained in another.* ▌

Satisfiability. A Boolean function is said to be *satisfiable* if it is not identically zero — that is, if it has at least one implicant. The most famous unsolved problem in all of computer science is to find an efficient way to decide whether a given Boolean function is satisfiable or unsatisfiable. More precisely, we ask: Is there an algorithm that inputs a Boolean formula of length N and tests it for satisfiability, always giving the correct answer after performing at most $N^{O(1)}$ steps?

When you hear about this problem for the first time, you might be tempted to ask a question of your own in return: "What? Are you serious that computer scientists still haven't figured out how to do such a simple thing?"

McCluskey
Fišer
Hlavička
monotone
positive
notations: $\subseteq$
Quine
satisfiability–satisfiable
P=NP?

random Boolean functions
3CNF
3SAT
associative block design
Rivest

Well, if you think satisfiability testing is trivial, please tell us your method. We agree that the problem isn't always difficult; if, for example, the given formula involves only 30 Boolean variables, a brute-force trial of 2^{30} cases — that's about a billion — will indeed settle the matter. But an enormous number of practical problems that still await solution can be formulated as Boolean functions with, say, 100 variables, because mathematical logic is a very powerful way to express concepts. And the solutions to those problems correspond to the vectors $x = x_1 \ldots x_{100}$ for which $f(x) = 1$. So a truly efficient solution to the satisfiability problem would be a wonderful achievement.

There is at least one sense in which satisfiability testing is a no-brainer: If the function $f(x_1, \ldots, x_n)$ has been chosen at random, so that all 2^n truth tables are equally likely, then f is almost surely satisfiable, and we can find an x with $f(x) = 1$ after making fewer than 2 trials (on the average). It's like flipping a coin until it comes up heads; we rarely need to wait long. But the catch, of course, is that practical problems do not have random truth tables.

Okay, let's grant that satisfiability testing does seem to be tough, in general. In fact, satisfiability turns out to be difficult even when we try to simplify it by requiring that the Boolean function be presented as a "formula in 3CNF" — namely as a conjunctive normal form that has only *three* literals in each clause:

$$f(x_1, \ldots, x_n) = (t_1 \vee u_1 \vee v_1) \wedge (t_2 \vee u_2 \vee v_2) \wedge \cdots \wedge (t_m \vee u_m \vee v_m). \tag{31}$$

Here each t_j, u_j, and v_j is x_k or $\bar{x}_k$ for some k. The problem of deciding satisfiability for formulas in 3CNF is called "3SAT," and exercise 39 explains why it is not really easier than satisfiability in general.

We will be seeing many examples of hard-to-crack 3SAT problems, especially in Section 7.9, where satisfiability testing will be discussed in great detail. The situation is a little peculiar, however, because a formula needs to be fairly long before we need to think twice about its satisfiability. For example, the shortest unsatisfiable formula in 3CNF is $(x \vee x \vee x) \wedge (\bar{x} \vee \bar{x} \vee \bar{x})$; but it is obviously no challenge to the intellect. We don't get into rough waters unless the three literals t_j, u_j, v_j of a clause correspond to three different variables. And in that case, each clause rules out exactly 1/8 of the possibilities, because seven different settings of (t_j, u_j, v_j) will make it true. Consequently every such 3CNF with at most seven clauses is automatically satisfiable, and a random setting of its variables will succeed with probability $\geq 1 - 7/8 = 1/8$.

The shortest interesting formula in 3CNF therefore has at least eight clauses. And in fact, an interesting 8-clause formula does exist, based on the associative block design by R. L. Rivest that we considered in 6.5–(13):

$$\begin{aligned}&(x_2 \vee x_3 \vee \bar{x}_4) \wedge (x_1 \vee x_3 \vee x_4) \wedge (\bar{x}_1 \vee x_2 \vee x_4) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee x_3)\\ &\quad \wedge (\bar{x}_2 \vee \bar{x}_3 \vee x_4) \wedge (\bar{x}_1 \vee \bar{x}_3 \vee \bar{x}_4) \wedge (x_1 \vee \bar{x}_2 \vee \bar{x}_4) \wedge (x_1 \vee x_2 \vee \bar{x}_3).\end{aligned} \tag{32}$$

Any seven of these eight clauses are satisfiable, in exactly two ways, and they force the values of three variables; for example, the first seven imply that we have $x_1x_2x_3 = 001$. But the complete set of eight cannot be satisfied simultaneously.

Simple special cases. Two important classes of Boolean formulas have been identified for which the satisfiability problem does turn out to be pretty easy. These special cases arise when the conjunctive normal form being tested consists entirely of "Horn clauses" or entirely of "Krom clauses." A *Horn clause* is an OR of literals in which all or nearly all of the literals are complemented — at most one of its literals is a pure, unbarred variable. A *Krom clause* is an OR of exactly two literals. Thus, for example,

$$\bar{x} \vee \bar{y}, \qquad w \vee \bar{y} \vee \bar{z}, \qquad \bar{u} \vee \bar{v} \vee \bar{w} \vee \bar{x} \vee \bar{y} \vee z, \quad \text{and} \quad x$$

are examples of Horn clauses; and

$$x \vee x, \qquad \bar{x} \vee \bar{x}, \qquad \bar{x} \vee \bar{y}, \qquad x \vee \bar{y}, \qquad \bar{x} \vee y, \quad \text{and} \quad x \vee y$$

are examples of Krom clauses, only the last of which is not also a Horn clause. (The first example qualifies because $x \vee x = x$.) Notice that a Horn clause is allowed to contain any number of literals, but when we restrict ourselves to Krom clauses we are essentially considering the 2SAT problem. In both cases we will see that satisfiability can be decided in linear time — that is, by carrying out only $O(N)$ simple steps, when given a formula of length N.

Let's consider Horn clauses first. Why are they so easy to handle? The main reason is that a clause like $\bar{u} \vee \bar{v} \vee \bar{w} \vee \bar{x} \vee \bar{y} \vee z$ can be recast in the form $\neg(u \wedge v \wedge w \wedge x \wedge y) \vee z$, which is the same as

$$u \wedge v \wedge w \wedge x \wedge y \;\Rightarrow\; z.$$

In other words, if u, v, w, x, and y are all true, then z must also be true. For this reason, parameterized Horn clauses were chosen to be the basic underlying mechanism of the programming language called Prolog. Furthermore there is an easy way to characterize exactly which Boolean functions can be represented entirely with Horn clauses:

Theorem H. *The Boolean function $f(x_1,\ldots,x_n)$ is expressible as a conjunction of Horn clauses if and only if*

$$f(x_1,\ldots,x_n) = f(y_1,\ldots,y_n) = 1 \quad \text{implies} \quad f(x_1 \wedge y_1,\ldots,x_n \wedge y_n) = 1 \qquad (33)$$

for all Boolean values x_j and y_j.

Proof. [Alfred Horn, *J. Symbolic Logic* **16** (1951), 14–21, Lemma 7.] If we have $x_0 \vee \bar{x}_1 \vee \cdots \vee \bar{x}_k = 1$ and $y_0 \vee \bar{y}_1 \vee \cdots \vee \bar{y}_k = 1$, then

$$\begin{aligned}(x_0 \wedge y_0) &\vee \overline{x_1 \wedge y_1} \vee \cdots \vee \overline{x_k \wedge y_k}\\ &= (x_0 \vee \bar{x}_1 \vee \bar{y}_1 \vee \cdots \vee \bar{x}_k \vee \bar{y}_k) \wedge (y_0 \vee \bar{x}_1 \vee \bar{y}_1 \vee \cdots \vee \bar{x}_k \vee \bar{y}_k)\\ &\ge (x_0 \vee \bar{x}_1 \vee \cdots \vee \bar{x}_k) \wedge (y_0 \vee \bar{y}_1 \vee \cdots \vee \bar{y}_k) \;=\; 1;\end{aligned}$$

and a similar (but simpler) calculation applies when the unbarred literals x_0 and y_0 are not present. Therefore every conjunction of Horn clauses satisfies (33).

Conversely, condition (33) implies that every prime clause of f is a Horn clause (see exercise 44). ∎

Horn function
definite
core
compiler technology
grammar
programming language
syntax

Let's say that a *Horn function* is a Boolean function that satisfies condition (33), and let's also call it *definite* if it satisfies the further condition $f(1,\ldots,1)=1$. It's easy to see that a conjunction of Horn clauses is definite if and only if each clause has *exactly* one unbarred literal, because only an entirely negative clause like $\bar{x}\vee\bar{y}$ will fail if all variables are true. Definite Horn functions are slightly simpler to work with than Horn functions in general, because they are obviously always satisfiable. Thus, by Theorem H, they have a unique least vector x such that $f(x)=1$, namely the bitwise AND of all vectors that satisfy all clauses. The *core* of a definite Horn function is the set of all variables x_j that are true in this minimum vector x. Notice that the variables in the core must be true whenever f is true, so we can essentially factor them out.

Definite Horn functions arise in many ways, for example in the analysis of games (see exercises 51 and 52). Another nice example comes from compiler technology. Consider the following typical (but simplified) grammar for algebraic expressions in a programming language:

$$\begin{aligned}
&\langle\,\text{expression}\,\rangle \to \langle\,\text{term}\,\rangle \mid \langle\,\text{expression}\,\rangle\,\texttt{+}\,\langle\,\text{term}\,\rangle \mid \langle\,\text{expression}\,\rangle\,\texttt{-}\,\langle\,\text{term}\,\rangle\\
&\langle\,\text{term}\,\rangle \to \langle\,\text{factor}\,\rangle \mid \texttt{-}\langle\,\text{factor}\,\rangle \mid \langle\,\text{term}\,\rangle\,\texttt{*}\,\langle\,\text{factor}\,\rangle \mid \langle\,\text{term}\,\rangle\,\texttt{/}\,\langle\,\text{factor}\,\rangle\\
&\langle\,\text{factor}\,\rangle \to \langle\,\text{variable}\,\rangle \mid \langle\,\text{constant}\,\rangle \mid \texttt{(}\langle\,\text{expression}\,\rangle\texttt{)}\\
&\langle\,\text{variable}\,\rangle \to \langle\,\text{letter}\,\rangle \mid \langle\,\text{variable}\,\rangle\langle\,\text{letter}\,\rangle \mid \langle\,\text{variable}\,\rangle\langle\,\text{digit}\,\rangle\\
&\langle\,\text{letter}\,\rangle \to \texttt{a} \mid \texttt{b} \mid \texttt{c}\\
&\langle\,\text{constant}\,\rangle \to \langle\,\text{digit}\,\rangle \mid \langle\,\text{constant}\,\rangle\langle\,\text{digit}\,\rangle\\
&\langle\,\text{digit}\,\rangle \to \texttt{0} \mid \texttt{1}
\end{aligned}\tag{34}$$

For example, the string `a/(-b0-10)+cc*cc` meets the syntax for $\langle\,\text{expression}\,\rangle$ and uses each of the grammatical rules at least once.

Suppose we want to know what pairs of characters can appear next to each other in such expressions. Definite Horn clauses provide the answer, because we can set the problem up as follows: Let the quantities `Xx`, `xX`, and `xy` denote Boolean "propositions," where `X` is one of the symbols $\{\texttt{E},\texttt{T},\texttt{F},\texttt{V},\texttt{L},\texttt{C},\texttt{D}\}$ standing respectively for $\langle\,\text{expression}\,\rangle$, $\langle\,\text{term}\,\rangle$, $\ldots$, $\langle\,\text{digit}\,\rangle$, and where `x` and `y` are symbols in the set $\{\texttt{+},\texttt{-},\texttt{*},\texttt{/},\texttt{(},\texttt{)},\texttt{a},\texttt{b},\texttt{c},\texttt{0},\texttt{1}\}$. The proposition `Xx` means, "`X` can end with `x`"; similarly, `xX` means, "`X` can start with `x`"; and `xy` means, "The character `x` can be followed immediately by `y` in an expression." (There are $7\times 11+11\times 7+11\times 11=275$ propositions altogether.) Then we can write

$$\begin{array}{lllrr}
\texttt{xT}\Rightarrow\texttt{xE} & \Rightarrow\texttt{-T} & \texttt{xC}\Rightarrow\texttt{xF} & \texttt{Vx}\wedge\texttt{yL}\Rightarrow\texttt{xy} & \Rightarrow\texttt{Lc}\\
\texttt{Tx}\Rightarrow\texttt{Ex} & \texttt{xF}\Rightarrow\texttt{-x} & \texttt{Cx}\Rightarrow\texttt{Fx} & \texttt{Vx}\wedge\texttt{yD}\Rightarrow\texttt{xy} & \texttt{xD}\Rightarrow\texttt{xC}\\
\texttt{Ex}\Rightarrow\texttt{x+} & \texttt{Tx}\Rightarrow\texttt{x*} & \Rightarrow\texttt{(F} & \texttt{Dx}\Rightarrow\texttt{Vx} & \texttt{Dx}\Rightarrow\texttt{Cx}\\
\texttt{xT}\Rightarrow\texttt{+x} & \texttt{xF}\Rightarrow\texttt{*x} & \texttt{xE}\Rightarrow\texttt{(x} & \Rightarrow\texttt{aL} & \texttt{Cx}\wedge\texttt{yD}\Rightarrow\texttt{xy}\\
\texttt{Ex}\Rightarrow\texttt{x-} & \texttt{Tx}\Rightarrow\texttt{x/} & \texttt{Ex}\Rightarrow\texttt{x)} & \Rightarrow\texttt{La} & \Rightarrow\texttt{0D}\\
\texttt{xT}\Rightarrow\texttt{-x} & \texttt{xF}\Rightarrow\texttt{/x} & \Rightarrow\texttt{F)} & \Rightarrow\texttt{bL} & \Rightarrow\texttt{D0}\\
\texttt{xF}\Rightarrow\texttt{xT} & \texttt{xV}\Rightarrow\texttt{xF} & \texttt{xL}\Rightarrow\texttt{xV} & \Rightarrow\texttt{Lb} & \Rightarrow\texttt{1D}\\
\texttt{Fx}\Rightarrow\texttt{Tx} & \texttt{Vx}\Rightarrow\texttt{Fx} & \texttt{Lx}\Rightarrow\texttt{Vx} & \Rightarrow\texttt{cL} & \Rightarrow\texttt{D1}
\end{array}\tag{35}$$

where `x` and `y` run through the eleven terminal symbols $\{\texttt{+},\ldots,\texttt{1}\}$. This schematic specification gives us a total of $24\times 11+3\times 11\times 11+13\times 1=640$ definite

theorem proving
hypothesis
stack

Horn clauses, which we could write out formally as

$$\bigl(\overline{\texttt{+T}} \vee \texttt{+E}\bigr) \wedge \bigl(\overline{\texttt{-T}} \vee \texttt{-E}\bigr) \wedge \cdots \wedge \bigl(\overline{\texttt{V+}} \vee \overline{\texttt{0L}} \vee \texttt{+0}\bigr) \wedge \cdots \wedge \bigl(\texttt{D1}\bigr)$$

if we prefer the cryptic notation of Boolean algebra to the $\Rightarrow$ convention of (35).

Why did we do this? Because *the core of all these clauses is the set of all propositions that are true in this particular grammar.* For example, one can verify that `-E` is true, hence the symbols `(-` can occur next to each other within an expression; but the symbol pairs `++` and `*-` cannot (see exercise 46).

Furthermore, we can find the core of any given set of definite Horn clauses without great difficulty. We just start out with the propositions that appear alone, on the right-hand side of $\Rightarrow$ when the left-hand side is empty; thirteen clauses of that kind appear in (35). And once we assert the truth of those propositions, we might find one or more clauses whose left-hand sides are now known to be true. Hence their right-hand sides also belong to the core, and we can keep going in the same way. The whole procedure is pretty much like letting water run downhill until it has found its proper level. In fact, when we choose appropriate data structures, this downhill process goes quite fast, requiring only $O(N+n)$ steps, when N denotes the total length of the clauses and n is the number of propositional variables. (We assume here that all clauses have been expanded out, not abbreviated in terms of parameters like `x` and `y` above. More sophisticated techniques of theorem proving are available to deal with parameterized clauses, but they are beyond the scope of our present discussion.)

Algorithm C (*Core computation for definite Horn clauses*). Given a set P of propositional variables and a set C of clauses, each having the form

$$u_1 \wedge \cdots \wedge u_k \;\Rightarrow\; v \qquad \text{where } k \geq 0 \text{ and } \{u_1, \ldots, u_k, v\} \subseteq P, \tag{36}$$

this algorithm finds the set $Q \subseteq P$ of all propositional variables that are necessarily true whenever all of the clauses are true.

We use the following data structures for clauses c, propositions p, and hypotheses h, where a "hypothesis" is the appearance of a proposition on the left-hand side of a clause:

`CONCLUSION`(c) is the proposition on the right of clause c;

`COUNT`(c) is the number of hypotheses of c not yet asserted;

`TRUTH`(p) is 1 if p is known to be true, otherwise 0;

`LAST`(p) is the last hypothesis in which p appears;

`CLAUSE`(h) is the clause for which h appears on the left;

`PREV`(h) is the previous hypothesis containing the proposition of h.

We also maintain a stack $S_0, S_1, \ldots, S_{s-1}$ of all propositions that are known to be true but not yet asserted.

C1. [Initialize.] Set `LAST`$(p) \leftarrow \Lambda$ and `TRUTH`$(p) \leftarrow 0$ for each proposition p. Also set $s \leftarrow 0$, so that the stack is empty. Then for each clause c, having the form (36), set `CONCLUSION`$(c) \leftarrow v$ and `COUNT`$(c) \leftarrow k$. If $k = 0$ and

$\mathtt{TRUTH}(v) = 0$, set $\mathtt{TRUTH}(v) \leftarrow 1$, $S_s \leftarrow v$, and $s \leftarrow s+1$. Otherwise, for $1 \le j \le k$, create a hypothesis record h and set $\mathtt{CLAUSE}(h) \leftarrow c$, $\mathtt{PREV}(h) \leftarrow \mathtt{LAST}(u_j)$, $\mathtt{LAST}(u_j) \leftarrow h$.

C2. [Prepare to assert p.] Terminate the algorithm if $s = 0$; the desired core now consists of all propositions whose TRUTH has been set to 1. Otherwise set $s \leftarrow s-1$, $p \leftarrow S_s$, and $h \leftarrow \mathtt{LAST}(p)$.

C3. [Done with hypotheses?] If $h = \Lambda$, return to C2.

C4. [Validate h.] Set $c \leftarrow \mathtt{CLAUSE}(h)$ and $\mathtt{COUNT}(c) \leftarrow \mathtt{COUNT}(c) - 1$. If the new value of $\mathtt{COUNT}(c)$ is still nonzero, go to step C6.

C5. [Deduce $\mathtt{CONCLUSION}(c)$.] Set $p \leftarrow \mathtt{CONCLUSION}(c)$. If $\mathtt{TRUTH}(p) = 0$, set $\mathtt{TRUTH}(p) \leftarrow 1$, $S_s \leftarrow p$, $s \leftarrow s+1$.

C6. [Loop on h.] Set $h \leftarrow \mathtt{PREV}(h)$ and return to C3. ▮

Notice how smoothly the data structures work together, avoiding any need to search for a place to make progress in the calculation. Algorithm C is similar in many respects to Algorithm 2.2.3T (topological sorting), which was the first example of multilinked data structures that we discussed long ago in Chapter 2; in fact, we can regard Algorithm 2.2.3T as the special case of Algorithm C in which every proposition appears on the right-hand side of exactly one clause. (See exercise 47.)

Exercise 48 shows that a slight modification of Algorithm C solves the satisfiability problem for Horn clauses in general. Further discussion can be found in a paper by W. F. Dowling and J. H. Gallier, *J. Logic Programming* **1** (1984), 267–284.

We turn now to Krom functions and the 2SAT problem. Again there's a linear-time algorithm; but again, we can probably appreciate it best if we look first at a simplified-but-practical application. Let's suppose that seven comedians have each agreed to do one-night standup gigs at two of five hotels during a three-day festival, but each of them is available for only two of those days because of other commitments:

Tomlin should do Aladdin and Caesars on days 1 and 2;
Unwin should do Bellagio and Excalibur on days 1 and 2;
Vegas should do Desert and Excalibur on days 2 and 3;
Williams should do Aladdin and Desert on days 1 and 3; (37)
Xie should do Caesars and Excalibur on days 1 and 3;
Yankovic should do Bellagio and Desert on days 2 and 3;
Zany should do Bellagio and Caesars on days 1 and 2.

Is it possible to schedule them all without conflict?

To solve this problem, we can introduce seven Boolean variables $\{t, u, v, w, x, y, z\}$, where t (for example) means that Tomlin does Aladdin on day 1 and Caesars on day 2 while $\bar{t}$ means that the days booked for those hotels occur in the opposite order. Then we can set up constraints to ensure that no two comedians

topological sorting
multilinked data structures
satisfiability problem for Horn clauses
Dowling
Gallier
Krom functions
comedy festival
hotels and comedians
Las Vegas hotels
schedule
Tomlin, Mary Jean (= Lily)
Williams, Robin McLaurim

contrapositive

are booked in the same hotel on the same day:

$$\begin{array}{llllllll}
\neg(t \wedge w) & \text{[A1]} & \neg(y \wedge \bar{z}) & \text{[B2]} & \neg(t \wedge z) & \text{[C2]} & \neg(w \wedge y) & \text{[D3]} \\
\neg(u \wedge z) & \text{[B1]} & \neg(\bar{t} \wedge x) & \text{[C1]} & \neg(v \wedge \bar{y}) & \text{[D2]} & \neg(\bar{u} \wedge \bar{x}) & \text{[E1]} \\
\neg(\bar{u} \wedge y) & \text{[B2]} & \neg(\bar{t} \wedge \bar{z}) & \text{[C1]} & \neg(\bar{v} \wedge w) & \text{[D3]} & \neg(u \wedge \bar{v}) & \text{[E2]} \\
\neg(\bar{u} \wedge \bar{z}) & \text{[B2]} & \neg(x \wedge \bar{z}) & \text{[C1]} & \neg(\bar{v} \wedge y) & \text{[D3]} & \neg(v \wedge x) & \text{[E3]}
\end{array} \qquad (38)$$

Each of these constraints is, of course, a Krom clause; we must satisfy

$$\begin{aligned}
&(\bar{t}\vee\bar{w}) \wedge (\bar{u}\vee\bar{z}) \wedge (u\vee\bar{y}) \wedge (u\vee z) \wedge (\bar{y}\vee z) \wedge (t\vee\bar{x}) \wedge (t\vee z) \wedge (\bar{x}\vee z) \\
&\quad \wedge (\bar{t}\vee\bar{z}) \wedge (\bar{v}\vee y) \wedge (v\vee\bar{w}) \wedge (v\vee\bar{y}) \wedge (\bar{w}\vee\bar{y}) \wedge (u\vee x) \wedge (\bar{u}\vee v) \wedge (\bar{v}\vee\bar{x}). \qquad (39)
\end{aligned}$$

Furthermore, Krom clauses (like Horn clauses) can be written as implications:

$$\begin{aligned}
&t \Rightarrow \bar{w}, \quad u \Rightarrow \bar{z}, \quad \bar{u} \Rightarrow \bar{y}, \quad \bar{u} \Rightarrow z, \quad y \Rightarrow z, \quad \bar{t} \Rightarrow \bar{x}, \quad \bar{t} \Rightarrow z, \quad x \Rightarrow z, \\
&t \Rightarrow \bar{z}, \quad v \Rightarrow y, \quad \bar{v} \Rightarrow \bar{w}, \quad \bar{v} \Rightarrow \bar{y}, \quad w \Rightarrow \bar{y}, \quad \bar{u} \Rightarrow x, \quad u \Rightarrow v, \quad v \Rightarrow \bar{x}. \qquad (40)
\end{aligned}$$

And every such implication also has an alternative, “contrapositive” form:

$$\begin{aligned}
&w \Rightarrow \bar{t}, \quad z \Rightarrow \bar{u}, \quad y \Rightarrow u, \quad \bar{z} \Rightarrow u, \quad \bar{z} \Rightarrow \bar{y}, \quad x \Rightarrow t, \quad \bar{z} \Rightarrow t, \quad \bar{z} \Rightarrow \bar{x}, \\
&z \Rightarrow \bar{t}, \quad \bar{y} \Rightarrow \bar{v}, \quad w \Rightarrow v, \quad y \Rightarrow v, \quad y \Rightarrow \bar{w}, \quad \bar{x} \Rightarrow u, \quad \bar{v} \Rightarrow \bar{u}, \quad x \Rightarrow \bar{v}. \qquad (41)
\end{aligned}$$

But oops — alas — there is a vicious cycle,

$$u \underset{\text{[B1]}}{\Rightarrow} \bar{z} \underset{\text{[B2]}}{\Rightarrow} \bar{y} \underset{\text{[D2]}}{\Rightarrow} \bar{v} \underset{\text{[E2]}}{\Rightarrow} \bar{u} \underset{\text{[B2]}}{\Rightarrow} z \underset{\text{[C2]}}{\Rightarrow} \bar{t} \underset{\text{[C1]}}{\Rightarrow} \bar{x} \underset{\text{[E1]}}{\Rightarrow} u \qquad (42)$$

This cycle tells that u and $\bar{u}$ must both have the same value; so there is no way to accommodate all of the conditions in (37). The festival organizers will have to renegotiate their agreement with at least one of the six comedians $\{t, u, v, x, y, z\}$, if a viable schedule is to be achieved. (See exercise 53.)

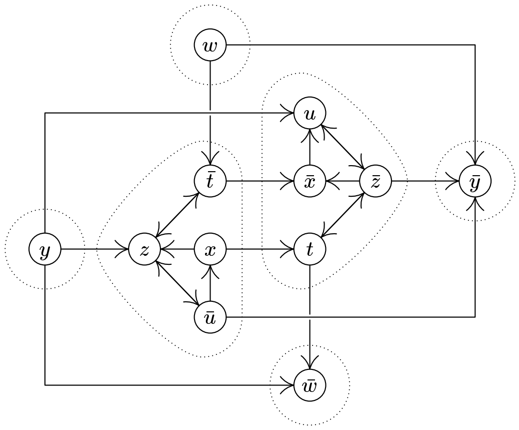


Fig. 6. The digraph corresponding to all implications of (40) and (41) that do not involve either v or $\bar{v}$. Assigning appropriate values to the literals in each strong component will solve a binary scheduling problem that is an instance of 2SAT.

The organizers might, for instance, try to leave v out of the picture temporarily. Then five of the sixteen constraints in (38) would go away and only 22 of the implications from (40) and (41) would remain, leaving the directed graph illustrated in Fig. 6. This digraph does contain cycles, like $z \Rightarrow \bar{u} \Rightarrow x \Rightarrow z$ and $t \Rightarrow \bar{z} \Rightarrow t$; but no cycle contains both a variable and its complement. Indeed, we can see from Fig. 6 that the values $tuwxyz = 110000$ do satisfy every clause of (39) that doesn't involve v or $\bar{v}$. These values give us a schedule that satisfies six of the seven original stipulations in (37), starting with (Tomlin, Unwin, Zany, Williams, Xie) at the (Aladdin, Bellagio, Caesars, Desert, Excalibur) on day 1.

In general, given any 2SAT problem with m Krom clauses that involve n Boolean variables, we can form a directed graph in the same way. There are $2n$ vertices $\{x_1, \bar{x}_1, \ldots, x_n, \bar{x}_n\}$, one for each possible literal; and there are $2m$ arcs of the form $\bar{u} \to v$ and $\bar{v} \to u$, two for each clause $u \vee v$. Two literals u and v belong to the same *strong component* of this digraph if and only if there are oriented paths from u to v and from v to u. For example, the six strong components of the digraph in Fig. 6 are indicated by dotted contours. All literals in a strong component must have the same Boolean value, in any solution to the corresponding 2SAT problem.

Theorem K. *A conjunctive normal form with two literals per clause is satisfiable if and only if no strong component of the associated digraph contains both a variable and its complement.*

Proof. [Melven Krom, *Zeitschrift für mathematische Logik und Grundlagen der Mathematik* **13** (1967), 15–20, Corollary 2.2.] If there are paths from x to $\bar{x}$ and from $\bar{x}$ to x, the formula is certainly unsatisfiable.

Conversely, assume that no such paths exist. Any digraph has at least one strong component S that is a "source," having no incoming arcs from vertices in any other strong component. Moreover, our digraph always has an attractive antisymmetry, illustrated in Fig. 6: We have $u \to v$ if and only if $\bar{v} \to \bar{u}$. Therefore the complements of the literals in S form another strong component $\overline{S} \neq S$ that is a "sink," having no *outgoing* arcs to other strong components. Hence we can assign the value 0 to all literals in S and 1 to all literals in $\overline{S}$, then remove them from the digraph and proceed in the same way until all literals have received a value. The resulting values satisfy $u \leq v$ whenever $u \to v$ in the digraph; hence they satisfy $\bar{u} \vee v$ whenever $\bar{u} \vee v$ is a clause of the formula. ▌

Theorem K leads immediately to an efficient solution of the 2SAT problem, thanks to an algorithm by R. E. Tarjan that finds strong components in linear time. [See *SICOMP* **1** (1972), 146–160; D. E. Knuth, *The Stanford GraphBase* (1993), 512–519.] We shall study Tarjan's algorithm in detail in Section 7.4.1. Exercise 54 shows that the condition of Theorem K is readily checked whenever the algorithm detects a new strong component. Furthermore, the algorithm detects "sinks" first; thus, as a simple byproduct of Tarjan's procedure, we can assign values that establish satisfiability by choosing the value 1 for each literal in a strong component that occurs before its complement.

Medians. We've been focusing on Boolean binary operations like $x \vee y$ or $x \oplus y$. But there's also a significant *ternary* operation $\langle xyz \rangle$, called the *median* of x, y, and z:

$$\langle xyz \rangle \;=\; (x \wedge y) \vee (y \wedge z) \vee (x \wedge z) = (x \vee y) \wedge (y \vee z) \wedge (x \vee z). \qquad (43)$$

Tomlin
Williams
strong component
Krom
source
antisymmetry
sink
Tarjan
Knuth
GraphBase
medians–
Pun resisted (re "co-medians")
notation $< xyz >$+

In fact, $\langle xyz\rangle$ is probably the most important ternary operation in the entire universe, because it has amazing properties that are continually being discovered and rediscovered.

In the first place, we can see easily that this formula for $\langle xyz\rangle$ describes the *majority* value of any three Boolean quantities x, y, and z: $\langle 000\rangle = \langle 001\rangle = 0$ and $\langle 011\rangle = \langle 111\rangle = 1$. We call $\langle xyz\rangle$ the "median" instead of the "majority" because, if x, y, and z are arbitrary *real* numbers, and if the operations $\wedge$ and $\vee$ denote min and max, then

$$\langle xyz\rangle \;=\; y \qquad \text{when } x \le y \le z. \tag{44}$$

Secondly, the basic binary operations $\wedge$ and $\vee$ are special cases of medians:

$$x \wedge y \;=\; \langle x0y\rangle; \qquad\qquad x \vee y \;=\; \langle x1y\rangle. \tag{45}$$

Thus *any* monotone Boolean function can be expressed entirely in terms of the ternary median operator and the constants 0 and 1. In fact, if we lived in a median-only world, we could let $\wedge$ stand for falsehood and $\vee$ for truth; then $x \wedge y = \langle x{\wedge}y\rangle$ and $x \vee y = \langle x{\vee}y\rangle$ would be perfectly natural expressions, and we could even use Polish notation like $\langle {\wedge}xy\rangle$ and $\langle {\vee}xy\rangle$ if that was our preference! The same idea applies to extended real numbers under the min-max interpretation of $\wedge$ and $\vee$, if we take medians with respect to the constants $\wedge = -\infty$ and $\vee = +\infty$.

A Boolean function $f(x_1, x_2, \ldots, x_n)$ is called *self-dual* when it satisfies

$$\overline{f(x_1, x_2, \ldots, x_n)} \;=\; f(\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n). \tag{46}$$

We've noted that a Boolean function is monotone if and only if it can be expressed in terms of $\wedge$ and $\vee$; by De Morgan's laws (11) and (12), a monotone formula is self-dual if and only if the symbols $\wedge$ and $\vee$ can be interchanged without changing the formula's value. Thus the median operation defined in (43) is both monotone and self-dual. In fact, it is the simplest nontrivial function of that kind, since none of the binary operations in Table 1 are both monotone and self-dual except the projections $\llcorner$ and R.

Furthermore, *any* expression that has been formed entirely with the median operator, without using constants, is both monotone and self-dual. For example, the function $\langle w\langle xyz\rangle\langle w\langle uvw\rangle x\rangle\rangle$ is self-dual because

$$\begin{aligned}\overline{\langle w\langle xyz\rangle\langle w\langle uvw\rangle x\rangle\rangle} \;&=\; \langle \bar{w}\,\overline{\langle xyz\rangle}\,\overline{\langle w\langle uvw\rangle x\rangle}\rangle\\ &=\; \langle \bar{w}\langle \bar{x}\bar{y}\bar{z}\rangle\langle \bar{w}\,\overline{\langle uvw\rangle}\,\bar{x}\rangle\rangle \;=\; \langle \bar{w}\langle \bar{x}\bar{y}\bar{z}\rangle\langle \bar{w}\langle \bar{u}\bar{v}\bar{w}\rangle\bar{x}\rangle\rangle.\end{aligned}$$

Emil Post, while working on his Ph.D. thesis (Columbia University, 1920), proved that the converse statement is also true:

Theorem P. *Every monotone, self-dual Boolean function $f(x_1, \ldots, x_n)$ can be expressed entirely in terms of the median operation $\langle xyz\rangle$.*

ternary operation
majority
min+
max+
monotone
and
or
falsehood
truth
Polish notation
Extended real numbers: Real numbers together v
self-dual+
De Morgan's laws
trivial function
projections
Post

prime form
prime implicant
trivial function
median of five
disjunctive prime form
Miiller
Winder
median algebras–

Proof. [*Annals of Mathematics Studies* **5** (1941), 74–75.] Observe first that

$$\begin{aligned}&\langle x_1 y\langle x_2 y \ldots y\langle x_{s-1} y x_s\rangle \ldots\rangle\rangle \\ &\quad = \big((x_1 \vee x_2 \vee \cdots \vee x_{s-1} \vee x_s) \wedge y\big) \vee (x_1 \wedge x_2 \wedge \cdots \wedge x_{s-1} \wedge x_s); \qquad (47)\end{aligned}$$

this formula for repeated medianing is easily proved by induction on s.

Now suppose $f(x_1,\ldots,x_n)$ is monotone, self-dual, and has the disjunctive prime form

$$f(x_1,\ldots,x_n) \;=\; t_1 \vee \cdots \vee t_m, \qquad t_j = x_{j1} \wedge \cdots \wedge x_{js_j},$$

where no prime implicant t_j is contained in another (Corollary Q). Any two prime implicants must have at least one variable in common. For if we had, say, $t_1 = x \wedge y$ and $t_2 = u \wedge v \wedge w$, the value of f would be 1 when $x = y = 1$ and $u = v = w = 0$, as well as when $x = y = 0$ and $u = v = w = 1$, contradicting self-duality. Therefore if any t_j consists of a single variable x, it must be the only prime implicant — in which case f is the trivial function $f(x_1,\ldots,x_n) = x = \langle xxx\rangle$.

Define the functions g_0, g_1, …, g_m by composing medians as follows:

$$\begin{aligned}&g_0(x_1,\ldots,x_n) = x_1;\\ &g_j(x_1,\ldots,x_n) = h(x_{j1},\ldots,x_{js_j}; g_{j-1}(x_1,\ldots,x_n)), \text{ for } 1 \le j \le m;\end{aligned} \qquad (48)$$

here $h(x_1,\ldots,x_s;y)$ denotes the function on the top line of (47). By induction on j, we can prove from (47) and (48) that $g_j(x_1,\ldots,x_n) = 1$ whenever we have $t_1 \vee \cdots \vee t_j = 1$, because $(x_{j1} \vee \cdots \vee x_{js_j}) \wedge t_k = t_k$ when $k < j$.

Finally, $f(x_1,\ldots,x_n)$ must equal $g_m(x_1,\ldots,x_n)$, because both functions are monotone and self-dual, and we have shown that $f(x_1,\ldots,x_n) \le g_m(x_1,\ldots,x_n)$ for all combinations of 0s and 1s. This inequality suffices to prove equality, because a self-dual function equals 1 in exactly half of the 2^n possible cases. ▌

One consequence of Theorem P is that we can express the median of five elements via medians of three, because the median of any odd number of Boolean variables is obviously a monotone and self-dual Boolean function. Let's write $\langle x_1 \ldots x_{2k-1}\rangle$ for such a median. Then the disjunctive prime form of $\langle vwxyz\rangle$ is

$$\begin{aligned}&(v\wedge w\wedge x) \vee (v\wedge w\wedge y) \vee (v\wedge w\wedge z) \vee (v\wedge x\wedge y) \vee (v\wedge x\wedge z)\\ &\qquad \vee (v\wedge y\wedge z) \vee (w\wedge x\wedge y) \vee (w\wedge x\wedge z) \vee (w\wedge y\wedge z) \vee (x\wedge y\wedge z);\end{aligned}$$

so the construction in the proof of Theorem P expresses $\langle vwxyz\rangle$ as a huge formula $g_{10}(v,w,x,y,z)$ involving 2,046 median-of-3 operations. Of course this expression isn't the shortest possible one; we actually have

$$\langle vwxyz\rangle \;=\; \langle v\langle xyz\rangle\langle wx\langle wyz\rangle\rangle\rangle. \qquad (49)$$

[See H. S. Miiller and R. O. Winder, *IRE Transactions* **EC-11** (1962), 89–90.]

***Median algebras and median graphs.** We noted earlier that the ternary operation $\langle xyz\rangle$ is useful when x, y, and z belong to any ordered set like the real numbers, when $\wedge$ and $\vee$ are regarded as the operators min and max. In fact, the operation $\langle xyz\rangle$ also plays a useful role in far more general circumstances.

A *median algebra* is any set M on which a ternary operation $\langle xyz\rangle$ is defined that takes elements of M into elements of M and obeys the following three axioms:

median algebras–
majority law
commutative law
associative law
distributive law
ideal
interval
$[u\,..\,v]$ (closed interval between u and v)
notation [u..v]
between

$$\langle xxy\rangle = x \quad \text{(majority law)}; \tag{50}$$

$$\langle xyz\rangle = \langle xzy\rangle = \langle yxz\rangle = \langle yzx\rangle = \langle zxy\rangle = \langle zyx\rangle \quad \text{(commutative law)}; \tag{51}$$

$$\langle xw\langle ywz\rangle\rangle = \langle\langle xwy\rangle wz\rangle \quad \text{(associative law)}. \tag{52}$$

In the Boolean case, for example, the associative law (52) holds for $w=0$ and $w=1$ because $\wedge$ and $\vee$ are associative. Exercises 75 and 76 prove that these three axioms imply also a *distributive law* for medians, which has both a short form

$$\big\langle\langle xyz\rangle uv\big\rangle \;=\; \big\langle x\langle yuv\rangle\langle zuv\rangle\big\rangle \tag{53}$$

and a more symmetrical long form

$$\big\langle\langle xyz\rangle uv\big\rangle \;=\; \big\langle\langle xuv\rangle\langle yuv\rangle\langle zuv\rangle\big\rangle. \tag{54}$$

No simple proof of this fact is known, but we can at least verify the special case of (53) and (54) when $y=u$ and $z=v$: We have

$$\big\langle\langle xyz\rangle yz\big\rangle \;=\; \langle xyz\rangle \tag{55}$$

because both sides equal $\langle xy\langle zyz\rangle\rangle$. In fact, the associative law (52) is just the special case $y=u$ of (53). And with (55) and (52) we can also verify the case $x=u$: $\big\langle\langle uyz\rangle uv\big\rangle = \big\langle vu\langle yuz\rangle\big\rangle = \big\langle\langle vuy\rangle uz\big\rangle = \big\langle\langle yuv\rangle uz\big\rangle = \big\langle\langle\langle yuv\rangle uv\rangle uz\big\rangle = \big\langle\langle yuv\rangle u\langle vuz\rangle\big\rangle = \big\langle u\langle yuv\rangle\langle zuv\rangle\big\rangle$.

An *ideal* in a median algebra M is a set $C \subseteq M$ for which we have

$$\langle xyz\rangle \;\in\; C \qquad \text{whenever } x\in C,\ y\in C, \text{ and } z\in M. \tag{56}$$

If u and v are any elements of M, the *interval* $[u\,..\,v]$ is defined as follows:

$$[u\,..\,v] \;=\; \big\{\langle xuv\rangle \;\big|\; x\in M\big\}. \tag{57}$$

We say that "x is between u and v" if and only if $x\in[u\,..\,v]$. According to these definitions, u and v themselves always belong to the interval $[u\,..\,v]$.

Lemma M. *Every interval* $[u\,..\,v]$ *is an ideal, and* $x\in[u\,..\,v] \iff x=\langle uxv\rangle$.

Proof. Let $\langle xuv\rangle$ and $\langle yuv\rangle$ be arbitrary elements of $[u\,..\,v]$. Then

$$\big\langle\langle xuv\rangle\langle yuv\rangle z\big\rangle \;=\; \big\langle\langle xyz\rangle uv\big\rangle \;\in\; [u\,..\,v]$$

for all $z\in M$, by (51) and (53), so $[u\,..\,v]$ is an ideal. Furthermore every element $\langle xuv\rangle\in[u\,..\,v]$ satisfies $\langle xuv\rangle = \big\langle u\langle xuv\rangle v\big\rangle$ by (51) and (55). ▮

Our intervals $[u\,..\,v]$ have nice properties, because of the median laws:

$$v\in[u\,..\,u] \implies u=v; \tag{58}$$

$$x\in[u\,..\,v] \text{ and } y\in[u\,..\,x] \implies y\in[u\,..\,v]; \tag{59}$$

$$x\in[u\,..\,v] \text{ and } y\in[u\,..\,z] \text{ and } y\in[v\,..\,z] \implies y\in[x\,..\,z]. \tag{60}$$

Equivalently, $[u\,..\,u]=\{u\}$; if $x\in[u\,..\,v]$ then $[u\,..\,x]\subseteq[u\,..\,v]$; and $x\in[u\,..\,v]$ also implies that $[u\,..\,z]\cap[v\,..\,z]\subseteq[x\,..\,z]$ for all z. (See exercise 72.)

shortest path
cube

Now let's define a graph on the vertex set M, with the following edges:

$$u \mathbin{—} v \quad\iff\quad u \neq v \text{ and } \langle xuv\rangle \in \{u, v\} \text{ for all } x \in M. \qquad (61)$$

In other words, u and v are adjacent if and only if the interval $[u\,..\,v]$ consists of just the two points u and v.

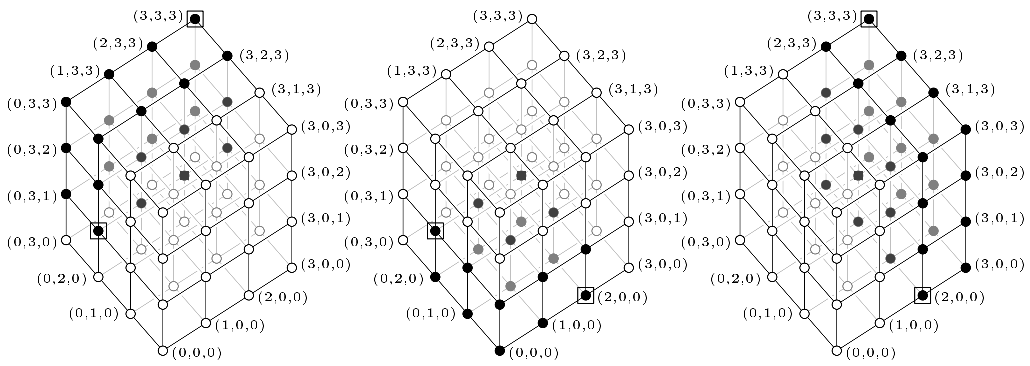
Theorem G. *If M is any finite median algebra, the graph defined by (61) is connected. Moreover, vertex x belongs to the interval $[u\,..\,v]$ if and only if x lies on a shortest path from u to v.*

Proof. If M isn't connected, choose u and v so that there is no path from u to v and the interval $[u\,..\,v]$ has as few elements as possible. Let $x \in [u\,..\,v]$ be distinct from u and v. Then $\langle xuv\rangle = x \neq v$, so $v \notin [u\,..\,x]$; similarly $u \notin [x\,..\,v]$. But $[u\,..\,x]$ and $[x\,..\,v]$ are contained in $[u\,..\,v]$, by (59). So they are smaller intervals, and there must be a path from u to x and from x to v. Contradiction.

The other half of the theorem is proved in exercise 73. ▌

Our definition of intervals implies that $\langle xyz\rangle \in [x\,..\,y] \cap [x\,..\,z] \cap [y\,..\,z]$, because $\langle xyz\rangle = \langle\langle xyz\rangle xy\rangle = \langle\langle xyz\rangle xz\rangle = \langle\langle xyz\rangle yz\rangle$ by (55). Conversely, if $w \in [x\,..\,y] \cap [x\,..\,z] \cap [y\,..\,z]$, exercise 74 proves that $w = \langle xyz\rangle$. In other words, *the intersection $[x\,..\,y] \cap [x\,..\,z] \cap [y\,..\,z]$ always contains exactly one point*, whenever x, y, and z are points of M.

Figure 7 illustrates this principle in a $4 \times 4 \times 4$ cube, where each point x has coordinates (x_1, x_2, x_3) with $0 \le x_1, x_2, x_3 < 4$. The vertices of this cube form a median algebra because $\langle xyz\rangle = \bigl(\langle x_1y_1z_1\rangle, \langle x_2y_2z_2\rangle, \langle x_3y_3z_3\rangle\bigr)$; furthermore, the edges of the graph in Fig. 7 are those defined in (61), running between vertices whose coordinates agree except that one coordinate changes by ± 1. Three typical intervals $[x\,..\,y]$, $[x\,..\,z]$, and $[y\,..\,z]$ are shown; the only point common to all three intervals is the vertex $\langle xyz\rangle = (2, 2, 1)$.



(a) The interval $[x\,..\,y]$. (b) The interval $[x\,..\,z]$. (c) The interval $[y\,..\,z]$.

Fig. 7. Intervals between the vertices $x = (0, 2, 1)$, $y = (3, 3, 3)$, and $z = (2, 0, 0)$ in a $4 \times 4 \times 4$ cube.

median graph–
interval
median graph
free tree
hyperrectangle
Cartesian products
median labels–
homomorphism
long distributive law
projects
labels

So far we've started with a median algebra and used it to define a graph with certain properties. But we can also start with a graph that has those properties and use it to define a median algebra. If u and v are vertices of *any* graph, let us define the interval $[u\,.\,.\,v]$ to be the set of all points on shortest paths between u and v. A finite graph is said to be a *median graph* if exactly one vertex lies in the intersection $[x\,.\,.\,y]\cap[x\,.\,.\,z]\cap[y\,.\,.\,z]$ of the three intervals that tie any three given vertices x, y, and z together; and we denote that vertex by $\langle xyz\rangle$. Exercise 75 proves that the resulting ternary operation satisfies the median axioms.

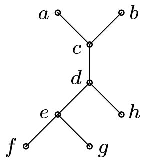
Many important graphs turn out to be median graphs according to this definition. For example, any free tree is easily seen to be a median graph; and a graph like the $n_1\times n_2\times\cdots\times n_m$ hyperrectangle provides another simple example. Cartesian products of arbitrary median graphs also satisfy the required condition.

***Median labels.** If u and v are any elements of a median algebra, the mapping $f(x)$ that takes $x\mapsto\langle xuv\rangle$ is a *homomorphism*; that is, it satisfies

$$f\bigl(\langle xyz\rangle\bigr) \;=\; \bigl\langle f(x)f(y)f(z)\bigr\rangle, \tag{62}$$

because of the long distributive law (54). This function $\langle xuv\rangle$ "projects" any given point x into the interval $[u\,.\,.\,v]$, by (57). And it is particularly interesting in the case when u — v is an edge of the corresponding graph, because $f(x)$ is then two-valued, essentially a Boolean mapping.

For example, consider the typical free tree shown below, with eight vertices and seven edges. We can project each vertex x onto each of the edge intervals $[u\,.\,.\,v]$ by deciding whether x is closer to u or to v:



$$\begin{array}{cccccccccc} & & ac & bc & cd & de & ef & eg & dh & \\ a & \mapsto & a & c & c & d & e & e & d & 0000000\\ b & \mapsto & c & b & c & d & e & e & d & 1100000\\ c & \mapsto & c & c & c & d & e & e & d & 1000000\\ d & \mapsto & c & c & d & d & e & e & d & 1010000\\ e & \mapsto & c & c & d & e & e & e & d & 1011000\\ f & \mapsto & c & c & d & e & f & e & d & 1011100\\ g & \mapsto & c & c & d & e & e & g & d & 1011010\\ h & \mapsto & c & c & d & d & e & e & h & 1010001 \end{array} \tag{63}$$

On the right we've reduced the projections to 0s and 1s, arbitrarily deciding that $a\mapsto 0000000$. The resulting bit strings are called *labels* of the vertices, and we write, for example, $l(b)=1100000$. Since each projection is a homomorphism, we can calculate the median of any three points by simply taking Boolean medians in each component of their labels. For example, to compute $\langle bgh\rangle$ we find the bitwise median of $l(b)=1100000$, $l(g)=1011010$, and $l(h)=1010001$, namely $1010000=l(d)$.

When we project onto all the edges of a median graph, we might find that two columns of the binary labels are identical. This situation cannot occur with a free tree, but let's consider what would happen if the edge g — h were added to the tree in (63): The resulting graph would still be a median graph, but the columns for eg and dh would become identical (except with $e \leftrightarrow d$ and $g \leftrightarrow h$). Furthermore, the new column for gh would turn out to be equivalent to the column for de. Redundant components should be omitted from the labels in such cases; therefore the vertices of the augmented graph would have six-bit labels, like $l(g) = 101101$ and $l(h) = 101001$, instead of seven-bit labels.

0–1 principle
zero-one principle
word problem
associative law
median of five
Post
convex
ideal
convex hull

The elements of any median algebra can always be represented by labels in this way. Therefore *any identity that holds in the Boolean case will be true in all median algebras.* This "zero-one principle" makes it possible to test whether any two given expressions built from the ternary operation $\langle xyz\rangle$ can be shown to be equal as a consequence of axioms (50), (51), and (52) — although we do have to check $2^{n-1}-1$ cases when we test n-variable expressions by this method.

For example, the associative law $\langle xw\langle ywz\rangle\rangle = \langle\langle xwy\rangle wz\rangle$ suggests that there should be a symmetrical interpretation of both sides that does not involve nested brackets. And indeed, there is such a formula:

$$\langle xw\langle ywz\rangle\rangle \;=\; \langle\langle xwy\rangle wz\rangle \;=\; \langle xwywz\rangle, \tag{64}$$

where $\langle xwywz\rangle$ denotes the median of the five-element multiset $\{x,w,y,w,z\} = \{w,w,x,y,z\}$. We can prove this formula by using the zero-one principle, noting also that median is the same thing as majority in the Boolean case. In a similar way we can prove (49), and we can show that the function used by Post in (47) can be simplified to

$$\langle x_1y\langle x_2y\ldots y\langle x_{s-1}yx_s\rangle\ldots\rangle\rangle \;=\; \langle x_1yx_2y\ldots yx_{s-1}yx_s\rangle; \tag{65}$$

it's a median of $2s-1$ quantities, where nearly half of them are equal to y.

A set C of vertices in a graph is called *convex* if $[u\,..\,v] \subseteq C$ whenever $u \in C$ and $v \in C$. In other words, whenever the endpoints of a shortest path belong to C, all vertices of that path must also be present in C. (A convex set is therefore identical to what we called an "ideal," a few pages ago; now our language has become geometric instead of algebraic.) The *convex hull* of $\{v_1,\ldots,v_m\}$ is defined to be the smallest convex set that contains each of the vertices v_1, ..., v_m. Our theoretical results above have shown that every interval $[u\,..\,v]$ is convex; hence $[u\,..\,v]$ is the convex hull of the two-point set $\{u,v\}$. But in fact much more is true:

Theorem C. *The convex hull of* $\{v_1,v_2,\ldots,v_m\}$ *in a median graph is the set of all points*

$$C \;=\; \bigl\{\langle v_1xv_2x\ldots xv_m\rangle \bigm| x \in M\bigr\}. \tag{66}$$

Furthermore, x is in C if and only if $x = \langle v_1xv_2x\ldots xv_m\rangle$.

Proof. Clearly $v_j \in C$ for $1 \le j \le m$. Every point of C must belong to the convex hull, because the point $x' = \langle v_2x\ldots xv_m\rangle$ is in the hull (by induction on m), and because $\langle v_1x\ldots xv_m\rangle \in [v_1\,..\,x']$. The zero-one principle proves that

$$\bigl\langle x\langle v_1yv_2y\ldots yv_m\rangle\langle v_1zv_2z\ldots zv_m\rangle\bigr\rangle = \bigl\langle v_1\langle xyz\rangle v_2\langle xyz\rangle\ldots\langle xyz\rangle v_m\bigr\rangle; \tag{67}$$

hence C is convex. Setting $x = y$ in this formula proves that $\langle v_1xv_2x\ldots xv_m\rangle$ is the closest point of C to x, and that $\langle v_1xv_2x\ldots xv_m\rangle \in [x\,..\,z]$ for all $z \in C$. ▮

4-cycle
Jha
Slutzki
Hagauer
Imrich
Klavžar
early neighbor
late neighbor
union-find algorithm

Corollary C. *Let the label of v_j be $v_{j1} \ldots v_{jt}$ for $1 \le j \le m$. Then the convex hull of $\{v_1, \ldots, v_m\}$ is the set of all $x \in M$ whose label $x_1 \ldots x_t$ satisfies $x_j = c_j$ whenever $v_{1j} = v_{2j} = \cdots = v_{mj} = c_j$.* ▌

For example, the convex hull of $\{c, g, h\}$ in (63) consists of all elements whose label matches the pattern 10∗∗0∗∗, namely $\{c, d, e, g, h\}$.

When a median graph contains a 4-cycle $u — x — v — y — u$, the edges $u — x$ and $v — y$ are equivalent, in the sense that projection onto $[u \mathinner{..} x]$ and projection onto $[v \mathinner{..} y]$ both yield the same label coordinates. The reason is that, for any z with $\langle zux \rangle = u$, we have

$$\begin{aligned} y = \langle uvy \rangle &= \big\langle \langle zux \rangle vy \big\rangle \\ &= \big\langle \langle zvy \rangle \langle uvy \rangle \langle xvy \rangle \big\rangle \\ &= \big\langle \langle zvy \rangle yv \big\rangle, \end{aligned}$$

hence $\langle zvy \rangle = y$; similarly $\langle zux \rangle = x$ implies $\langle zvy \rangle = v$. The edges $x — v$ and $y — u$ are equivalent for the same reasons. Exercise 77 shows, among other things, that two edges yield equivalent projections if and only if they can be proved equivalent by a chain of equivalences obtained from 4-cycles in this way. Therefore the number of bits in each vertex label is the number of equivalence classes of edges induced by the 4-cycles; and it follows that the reduced labels for vertices are uniquely determined, once we specify a vertex whose label is $00 \ldots 0$.

A nice way to find the vertex labels of any median graph was discovered by P. K. Jha and G. Slutzki [*Ars Combin.* **34** (1992), 75–92] and improved by J. Hagauer, W. Imrich, and S. Klavžar [*Theor. Comp. Sci.* **215** (1999), 123–136]:

Algorithm H (*Median labels*). Given a median graph G and a source vertex a, this algorithm determines the equivalence classes defined by the 4-cycles of G, and computes the labels $l(v) = v_1 \ldots v_t$ of each vertex, where t is the number of classes and $l(a) = 0 \ldots 0$.

H1. [Initialize.] Preprocess G by visiting all vertices in order of their distance from a. For each edge $u — v$, we say that u is an *early neighbor* of v if a is closer to u than to v, otherwise u is a *late neighbor*; in other words, the early neighbors of v will already have been visited when v is encountered, but the late neighbors will still be awaiting their turn. Rearrange all adjacency lists so that early neighbors are listed first. Place each edge initially in its own equivalence class; a "union-find algorithm" like Algorithm 2.3.3E will be used to merge classes when the algorithm learns that they're equivalent.

H2. [Call the subroutine.] Set $j \leftarrow 0$ and invoke Subroutine I with parameter a. (Subroutine I appears below. The global variable j will be used to create a master list of edges $r_j — s_j$ for $1 \le j < n$, where n is the total number of vertices; there will be one entry with $s_j = v$, for each vertex $v \ne a$.)

H3. [Assign the labels.] Number the equivalence classes from 1 to t. Then set $l(a)$ to the t-bit string $0 \ldots 0$. For $j = 1, 2, \ldots, n-1$ (in this order), set $l(s_j)$ to $l(r_j)$ with bit k changed from 0 to 1, where k is the equivalence class of edge $r_j — s_j$. ▌

recursive subroutine
breadth-first search
doubly linked
free median algebra+
monotone self-dual Boolean functions

Subroutine I (*Process descendants of r*). This recursive subroutine, with parameter r and global variable j, does the main work of Algorithm H on the graph of all vertices currently reachable from vertex r. In the course of processing, all such vertices will be recorded on the master list, except r itself, and all edges between them will be removed from the current graph. Each vertex has four fields called its `LINK`, `MARK`, `RANK`, and `MATE`, initially null.

I1. [Loop over s.] Choose a vertex s with $r \longrightarrow s$. If there is no such vertex, return from the subroutine.

I2. [Record the edge.] Set $j \leftarrow j+1$, $r_j \leftarrow r$, and $s_j \leftarrow s$.

I3. [Begin breadth-first search.] (Now we want to find and delete all edges of the current graph that are equivalent to $r \longrightarrow s$.) Set $\texttt{MARK}(s) \leftarrow s$, $\texttt{RANK}(s) \leftarrow 1$, $\texttt{LINK}(s) \leftarrow \Lambda$, and $v \leftarrow q \leftarrow s$.

I4. [Find the mate of v.] Find the early neighbor u of v for which $\texttt{MARK}(u) \neq s$. (There will be exactly one such vertex u. Recall that early neighbors have been placed first, in step H1.) Set $\texttt{MATE}(v) \leftarrow u$.

I5. [Delete $u \longrightarrow v$.] Make the edges $u \longrightarrow v$ and $r \longrightarrow s$ equivalent by merging their equivalence classes. Remove u and v from each other's adjacency lists.

I6. [Classify the neighbors of v.] For each early neighbor u of v, do step I7; for each late neighbor u of v, do step I8. Then go to step I9.

I7. [Note a possible equivalence.] If $\texttt{MARK}(u) = s$ and $\texttt{RANK}(u) = 1$, make the edge $u \longrightarrow v$ equivalent to the edge $\texttt{MATE}(u) \longrightarrow \texttt{MATE}(v)$. Return to I6.

I8. [Rank u.] If $\texttt{MARK}(u) = s$ and $\texttt{RANK}(u) = 1$, return to I6. Otherwise set $\texttt{MARK}(u) \leftarrow s$ and $\texttt{RANK}(u) \leftarrow 2$. Set w to the first neighbor of u (it will be early). If $w = v$, reset w to u's second early neighbor; but return to I6 if u has only one early neighbor. If $\texttt{MARK}(w) \neq s$ or $\texttt{RANK}(w) \neq 2$, set $\texttt{RANK}(u) \leftarrow 1$, $\texttt{LINK}(u) \leftarrow \Lambda$, $\texttt{LINK}(q) \leftarrow u$, and $q \leftarrow u$. Return to I6.

I9. [Continue breadth-first search.] Set $v \leftarrow \texttt{LINK}(v)$. Return to I4 if $v \neq \Lambda$.

I10. [Process subgraph s.] Call Subroutine I recursively with parameter s. Then return to I1. ▌

This algorithm and subroutine have been described in terms of relatively high-level data structures; further details are left to the reader. For example, adjacency lists should be doubly linked, so that edges can readily be deleted in step I5. Any convenient method for merging equivalence classes can be used in that step.

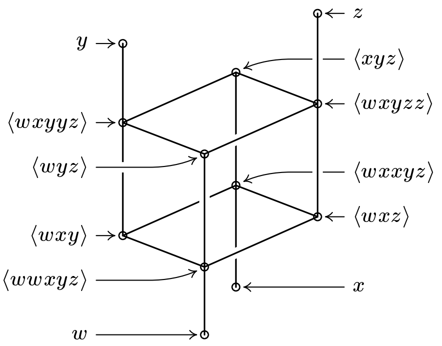
Exercise 77 explains the theory that makes this algorithm work, and exercise 78 proves that each vertex is encountered at most $\lg n$ times in step I4. Furthermore, exercise 79 shows that a median graph has at most $O(n \log n)$ edges. Therefore the total running time of Algorithm H is $O(n(\log n)^2)$, except perhaps for the bit-setting in step H3.

The reader may wish to play through Algorithm H by hand on the median graph in Table 2, whose vertices represent the twelve monotone self-dual Boolean functions of four variables $\{w, x, y, z\}$. All such functions that actually involve all four variables can be expressed as a median of five things, like (64). With

truth tables
median of five
majority
prime implicants
disjunctive prime form

Table 2

LABELS FOR THE FREE MEDIAN ALGEBRA ON FOUR GENERATORS



j	r_j	s_j	$l(s_j)$
		w	0000000
1	w	$\langle wwxyz\rangle$	0000001
2	$\langle wwxyz\rangle$	$\langle wyz\rangle$	0010001
3	$\langle wyz\rangle$	$\langle wxyzz\rangle$	0010101
4	$\langle wxyzz\rangle$	$\langle xyz\rangle$	0010111
5	$\langle wxyzz\rangle$	z	1010101
6	$\langle wyz\rangle$	$\langle wxyyz\rangle$	0010011
7	$\langle wxyyz\rangle$	y	0110011
8	$\langle wwxyz\rangle$	$\langle wxz\rangle$	0000101
9	$\langle wxz\rangle$	$\langle wxxyz\rangle$	0000111
10	$\langle wxxyz\rangle$	x	0001111
11	$\langle wwxyz\rangle$	$\langle wxy\rangle$	0000011

starting vertex $a = w$, the algorithm computes the master list of edges r_j — s_j and the binary labels shown in the table. (The actual order of processing depends on the order in which vertices appear in adjacency lists. But the final labels will be the same under any ordering, except for permutations of the columns.)

Notice that the number of 1-bits in each label $l(v)$ is the distance of v from the starting vertex a. In fact, the uniqueness of labels tells us that *the distance between any two vertices is the number of bit positions in which their labels differ*, because we could have started at any particular vertex.

The special median graph in Table 2 could actually have been handled in a completely different way, without using Algorithm H at all, because the labels in this case are essentially the same as the *truth tables* of the corresponding functions. Here's why: We can say that the simple functions w, x, y, z have the respective truth tables $t(w) = 0000000011111111$, $t(x) = 0000111100001111$, $t(y) = 0011001100110011$, $t(z) = 0101010101010101$. Then the truth table of $\langle wwxyz\rangle$ is the bitwise majority function $\langle t(w)t(w)t(x)t(y)t(z)\rangle$, namely the string 0000000101111111; and a similar computation gives the truth tables of all the other vertices.

The last half of any self-dual function's truth table is the same as the first half, but complemented and reversed, so we can eliminate it. Furthermore the leftmost bit in each of our truth tables is always zero. We are left with the seven-bit labels shown in Table 2; and the uniqueness property guarantees that Algorithm H will produce the same result, except for possible permutation of columns, when it is presented with this particular graph.

This reasoning tells us that the edges of the graph in Table 2 correspond to pairs of functions whose truth tables are almost the same. We move between neighboring vertices by switching only two complementary bits of their truth tables. In fact, the degree of each vertex turns out to be exactly the number of prime implicants in the disjunctive prime form of the monotone self-dual function represented by that vertex (see exercises 70 and 84).

***Median sets.** A *median set* is a collection X of binary vectors with the property that $\langle xyz\rangle \in X$ whenever $x \in X$, $y \in X$, and $z \in X$, where the medians are computed componentwise as we've done with median labels. Thomas Schaefer noticed in 1978 that median sets provide us with an attractive counterpoint to the characterization of Horn functions in Theorem H:

Theorem S. *The Boolean function $f(x_1,\ldots,x_n)$ is expressible as a conjunction of Krom clauses if and only if*

$$f(x_1,\ldots,x_n) = f(y_1,\ldots,y_n) = f(z_1,\ldots,z_n) = 1 \quad \text{implies} \quad f(\langle x_1y_1z_1\rangle,\ldots,\langle x_ny_nz_n\rangle) = 1 \tag{68}$$

for all Boolean values x_j, y_j, and z_j.

Proof. [*STOC* **10** (1978), 216–226, Lemma 3.1B.] If we have $x_1 \vee x_2 = y_1 \vee y_2 = z_1 \vee z_2 = 1$, say, with $x_1 \le y_1 \le z_1$, then $\langle x_1y_1z_1\rangle \vee \langle x_2y_2z_2\rangle = y_1 \vee \langle x_2y_2z_2\rangle = 1$, since $y_1 = 0$ implies that $x_2 = y_2 = 1$. Thus (68) is necessary.

Conversely, if (68) holds, let $u_1 \vee \cdots \vee u_k$ be a prime clause of f, where each u_j is a literal. Then, for $1 \le j \le k$, the clause $u_1 \vee \cdots \vee u_{j-1} \vee u_{j+1} \vee \cdots \vee u_k$ is not a clause of f; so there's a vector $x^{(j)}$ with $f(x^{(j)}) = 1$ but with $u_i^{(j)} = 0$ for all $i \ne j$. If $k \ge 3$, the median $\langle x^{(1)}x^{(2)}x^{(3)}\rangle$ has $u_i = 0$ for $1 \le i \le k$; but that's impossible, because $u_1 \vee \cdots \vee u_k$ was supposedly a clause. Hence $k \le 2$. ∎

Thus median sets are the same as "2SAT instances," the sets of points that satisfy some formula f in 2CNF.

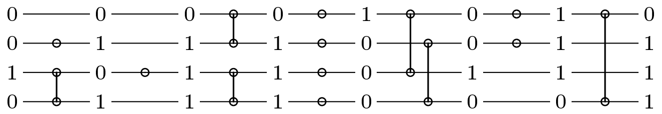
A median set is said to be *reduced* if its vectors $x = x_1 \ldots x_t$ contain no redundant components. In other words, for each coordinate position k, a reduced median set has at least two vectors $x^{(k)}$ and $y^{(k)}$ with the property that $x_k^{(k)} = 0$ and $y_k^{(k)} = 1$ but $x_i^{(k)} = y_i^{(k)}$ for all $i \ne k$. We've seen that the labels of a median graph satisfy this condition; in fact, if coordinate k corresponds to the edge $u \mathrel{-\!\!-} v$ in the graph, we can let $x^{(k)}$ and $y^{(k)}$ be the labels of u and v. Conversely, any reduced median set X defines a median graph, with one vertex for each element of X and with adjacency defined by all-but-one equality of coordinates. The median labels of these vertices must be identical to the original vectors in X, because we know that median labels are essentially unique.

Median labels and reduced median sets can also be characterized in yet another instructive way, which harks back to the networks of *comparator modules* that we studied in Section 5.3.4. We noted in that section that such networks are useful for "oblivious sorting" of numbers, and we noted in Theorem 5.3.4Z that a network of comparators will sort all $n!$ possible input permutations if and only if it correctly sorts all 2^n combinations of 0s and 1s. When a comparator module is attached to two horizontal lines, with inputs x and y entering from the left, it outputs the same two values on the right, but with $\min(x,y) = x \wedge y$ on the upper line and $\max(x,y) = x \vee y$ on the lower line. Let's now extend the concept slightly by also allowing *inverter modules*, which change 0 to 1 and vice versa. Here, for example, is a comparator-inverter network (or CI-net, for

Schaefer
Krom clauses
2SAT
2CNF
redundant
reduced median set
comparator modules
oblivious sorting
inverter modules
comparator-inverter network
CI-net

graph homomorphism
Feder
Wiedemann
topologically sorted

short), which transforms the binary value 0010 into 0111:



(69)

(A single dot denotes an inverter.) Indeed, this network transforms

$$\begin{array}{llll} 0000 \mapsto 0110; & 0100 \mapsto 0111; & 1000 \mapsto 0111; & 1100 \mapsto 0110; \\ 0001 \mapsto 0111; & 0101 \mapsto 1111; & 1001 \mapsto 0101; & 1101 \mapsto 0111; \\ 0010 \mapsto 0111; & 0110 \mapsto 1111; & 1010 \mapsto 0101; & 1110 \mapsto 0111; \\ 0011 \mapsto 0110; & 0111 \mapsto 0111; & 1011 \mapsto 0111; & 1111 \mapsto 0110. \end{array} \tag{70}$$

Suppose a CI-net transforms the bit string $x = x_1 \ldots x_t$ into the bit string $x'_1 \ldots x'_t = f(x)$. This function f, which maps the t-cube into itself, is in fact a *graph homomorphism*. In other words, we have $f(x) \mathbin{—} f(y)$ whenever $x \mathbin{—} y$ in the t-cube: Changing one bit of x always causes exactly one bit of $f(x)$ to change, because every module in the network has this behavior. Moreover, CI-nets have a remarkable connection with median labels:

Theorem F. *Every set X of t-bit median labels can be represented by a comparator-inverter network that computes a Boolean function $f(x)$ with the property that $f(x) \in X$ for all bit vectors $x_1 \ldots x_t$, and $f(x) = x$ for all $x \in X$.*

Proof. [Tomás Feder, *Memoirs Amer. Math. Soc.* **555** (1995), 1–223, Lemma 3.37; see also the Ph. D. thesis of D. H. Wiedemann (University of Waterloo, 1986).] Consider columns i and j of the median labels, where $1 \le i < j \le t$. Any such pair of columns contains at least three of the four possibilities $\{00, 01, 10, 11\}$, if we look through the entire set of labels, because median labels have no redundant columns. Let us write $\bar{\jmath} \to i$, $j \to i$, $i \to j$, or $i \to \bar{\jmath}$ if the value 00, 01, 10, or 11 (respectively) is missing from those two columns; we can also note the equivalent relations $\bar{\imath} \to j$, $\bar{\imath} \to \bar{\jmath}$, $\bar{\jmath} \to \bar{\imath}$, or $j \to \bar{\imath}$, respectively, which involve $\bar{\imath}$ instead of i. For example, the labels in Table 2 give us the relations

$$\begin{array}{ll} 1 \to \bar{2}, 3, \bar{4}, 5, \bar{6}, 7 & 2, \bar{3}, 4, \bar{5}, 6, \bar{7} \to \bar{1}; \\ 2 \to 3, \bar{4}, \bar{5}, 6, 7 & \bar{3}, 4, 5, \bar{6}, \bar{7} \to \bar{2}; \\ 3 \to \bar{4}, 7 & 4, \bar{7} \to \bar{3}; \\ 4 \to 5, 6, 7 & \bar{5}, \bar{6}, \bar{7} \to \bar{4}; \\ 5 \to 7 & \bar{7} \to \bar{5}; \\ 6 \to 7 & \bar{7} \to \bar{6}. \end{array} \tag{71}$$

(There is no relation between 3 and 5 because all four possibilities occur in those columns. But we have $3 \to \bar{4}$ because 11 doesn't appear in columns 3 and 4. The vertices whose label has a 1 in column 3 are those closer to $\langle wyz \rangle$ than to $\langle wwxyz \rangle$ in Table 2; they form a convex set in which column 4 of the labels is always 0, because they are also closer to $\langle wxxyz \rangle$ than to x.)

These relations between the literals $\{1, \bar{1}, 2, \bar{2}, \ldots, t, \bar{t}\}$ contain no cycles, so they can always be topologically sorted into an anti-symmetrical sequence

bitwise
homomorphism
retraction
retract
Hell
–median graphs
–median labels

$u_1\,u_2\,\ldots\,u_{2t}$ in which u_j is the complement of u_{2t+1-j}. For example,

$$1\ \bar{7}\ 4\ 2\ \bar{3}\ \bar{5}\ \bar{6}\ 6\ 5\ 3\ \bar{2}\ \bar{4}\ 7\ \bar{1} \tag{72}$$

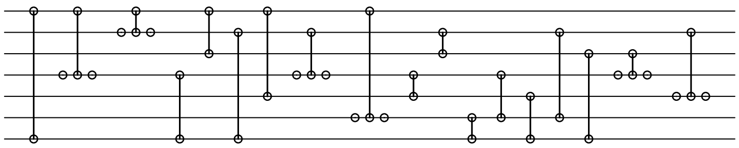
is one such way to sort the relations in (71) topologically.

Now we proceed to construct the network, by starting with t empty lines and successively examining elements u_k and u_{k+d} in the topological sequence, for $d = 2t-2,\ 2t-3,\ \ldots,\ 1$ (in this order), and for $k = 1,\ 2,\ \ldots,\ t - \lceil d/2 \rceil$. If $u_k \le u_{k+d}$ is a relation between columns i and j, where $i < j$, we append new modules to lines i and j of the network as follows:

If $i \to j$ If $i \to \bar{\jmath}$ If $\bar{\imath} \to j$ If $\bar{\imath} \to \bar{\jmath}$

(73)

For example, from (71) and (72) we first enforce $1 \to 7$, then $1 \to \bar{4}$, then $1 \to \bar{2}$, then $\bar{7} \to \bar{4}$ (that is, $4 \to 7$), etc., obtaining the following network:



(74)

(Go figure. No modules are contributed when, say, u_k is $\bar{7}$ and u_{k+d} is 3, because the relation $\bar{3} \to 7$ does not appear in (71).)

Exercise 89 proves that each new cluster of modules (73) preserves all of the previous inequalities and enforces a new one. Therefore, if x is any input vector, $f(x)$ satisfies all of the inequalities; so $f(x) \in X$ by Theorem S. Conversely, if $x \in X$, every cluster of modules in the network leaves x unchanged. ▮

Corollary F. *Suppose the median labels in Theorem F are closed under the operations of bitwise AND and OR, so that $x \,\&\, y \in X$ and $x \mid y \in X$ whenever $x \in X$ and $y \in X$. Then there is a permutation of coordinates under which the labels are representable by a network of comparator modules only.*

Proof. The bitwise AND of all labels is $0\ldots0$, and the bitwise OR is $1\ldots1$. So the only possible relations between columns are $i \to j$ and $j \to i$. By topologically sorting and renaming the columns, we can ensure that only $i \to j$ occurs when $i < j$; and in this case the construction in the proof never uses an inverter. ▮

In general, if G is any graph, a homomorphism f that maps the vertices of G onto a subset X of those vertices is called a *retraction* if it satisfies $f(x) = x$ for all $x \in X$; and we call X a *retract* of G when such an f exists. The importance of this concept in the theory of graphs was first pointed out by Pavol Hell [see *Lecture Notes in Math.* **406** (1974), 291–301]. One consequence, for example, is that the distance between vertices in X — the number of edges on a shortest path — remains the same even if we restrict consideration to paths that lie entirely in X. (See exercise 93.)

Theorem F demonstrates that every t-dimensional set of median labels is a retract of the t-dimensional hypercube. Conversely, exercise 94 shows that hypercube retracts are always median graphs.

Threshold functions. A particularly appealing and important class of Boolean functions $f(x_1, x_2, \ldots, x_n)$ arises when f can be defined by the formula

$$f(x_1, x_2, \ldots, x_n) \;=\; [w_1x_1 + w_2x_2 + \cdots + w_nx_n \geq t], \tag{75}$$

where the constants w_1, w_2, $\ldots$, w_n are integer "weights" and t is an integer "threshold" value. For example, threshold functions are important even when all the weights are unity: We have

$$x_1 \wedge x_2 \wedge \cdots \wedge x_n = [x_1 + x_2 + \cdots + x_n \geq n]; \tag{76}$$

$$x_1 \vee x_2 \vee \cdots \vee x_n = [x_1 + x_2 + \cdots + x_n \geq 1]; \tag{77}$$

and $$\langle x_1x_2 \ldots x_{2t-1}\rangle = [x_1 + x_2 + \cdots + x_{2t-1} \geq t], \tag{78}$$

where $\langle x_1x_2 \ldots x_{2t-1}\rangle$ stands for the median (or majority) value of a multiset that consists of any odd number of Boolean values $\{x_1, x_2, \ldots, x_{2t-1}\}$. In particular, the basic mappings $x \wedge y$, $x \vee y$, and $\langle xyz\rangle$ are all threshold functions, and so is

$$\bar{x} \;=\; [-x \geq 0]. \tag{79}$$

With more general weights we get many other functions of interest, such as

$$[2^{n-1}x_1 + 2^{n-2}x_2 + \cdots + x_n \geq (t_1t_2 \ldots t_n)_2], \tag{80}$$

which is true if and only if the binary string $x_1x_2 \ldots x_n$ is lexicographically greater than or equal to a given binary string $t_1t_2 \ldots t_n$. Given a set of n objects having sizes w_1, w_2, $\ldots$, w_n, a subset of those objects will fit into a knapsack of size $t-1$ if and only if $f(x_1, x_2, \ldots, x_n) = 0$, where $x_j = 1$ represents the presence of object j in the subset. Simple models of neurons, originally proposed by W. McCullough and W. Pitts in *Bull. Math. Biophysics* **5** (1943), 115–133, have led to thousands of research papers about "neural networks" built from threshold functions.

We can get rid of any negative weight w_j by setting $x_j \leftarrow \bar{x}_j$, $w_j \leftarrow -w_j$, and $t \leftarrow t + |w_j|$. Thus a general threshold function can be reduced to a positive threshold function in which all weights are nonnegative. Furthermore, any positive threshold function (75) can be expressed as a special case of the median/majority-of-odd function, because we have

$$\langle 0^a 1^b x_1^{w_1} x_2^{w_2} \ldots x_n^{w_n}\rangle = [b + w_1x_1 + w_2x_2 + \cdots + w_nx_n \geq b + t], \tag{81}$$

where x^m stands for m copies of x, and where a and b are defined by the rules

$$a = \max(0, 2t-1-w), \quad b = \max(0, w+1-2t), \quad w = w_1+w_2+\cdots+w_n. \tag{82}$$

For example, when all weights are 1, we have

$$\langle 0^{n-1}x_1 \ldots x_n\rangle = x_1 \wedge \cdots \wedge x_n \quad\text{and}\quad \langle 1^{n-1}x_1 \ldots x_n\rangle = x_1 \vee \cdots \vee x_n; \tag{83}$$

we've already seen these formulas in (45) when $n = 2$. In general, either a or b is zero, and the left-hand side of (81) specifies a median of $2T-1$ elements, where

$$T \;=\; b + t \;=\; \max(t, w_1 + w_2 + \cdots + w_n + 1 - t). \tag{84}$$

separable, see threshold
threshold functions++
majority
median of odd
lexicographically
knapsack
McCullough
Pitts
neural networks
positive threshold function
majority of odd, see median of odd

There would be no point in letting both a and b be greater than zero, because the majority function clearly satisfies the cancellation law

$$\langle 01x_1x_2\ldots x_{2t-1}\rangle \;=\; \langle x_1x_2\ldots x_{2t-1}\rangle. \qquad (85)$$

One important consequence of (81) is that every positive threshold function comes from the pure majority function

$$g(x_0,x_1,x_2,\ldots,x_n) \;=\; \langle x_0^{a+b}x_1^{w_1}x_2^{w_2}\ldots x_n^{w_n}\rangle \qquad (86)$$

by setting $x_0=0$ or 1. In other words, we know all threshold functions of n variables if and only if we know all of the distinct median-of-odd functions of $n+1$ or fewer variables (containing no constants). Every pure majority function is monotone and self-dual; thus we've seen the pure majority functions of four variables $\{w,x,y,z\}$ in column s_j of Table 2 on page 71, namely $\langle w\rangle$, $\langle wwxyz\rangle$, $\langle wyz\rangle$, $\langle wxyzz\rangle$, $\langle xyz\rangle$, $\langle z\rangle$, $\langle wxyyz\rangle$, $\langle y\rangle$, $\langle wxz\rangle$, $\langle wxxyz\rangle$, $\langle x\rangle$, $\langle wxy\rangle$. By setting $w=0$ or 1, we obtain all the positive threshold functions $f(x,y,z)$ of three variables:

$$\begin{aligned}&\langle 0\rangle,\langle 1\rangle,\langle 00xyz\rangle,\langle 11xyz\rangle,\langle 0yz\rangle,\langle 1yz\rangle,\langle 0xyzz\rangle,\langle 1xyzz\rangle,\langle xyz\rangle,\langle z\rangle,\\ &\quad\langle 0xyyz\rangle,\langle 1xyyz\rangle,\langle y\rangle,\langle 0xz\rangle,\langle 1xz\rangle,\langle 0xxyz\rangle,\langle 1xxyz\rangle,\langle x\rangle,\langle 0xy\rangle,\langle 1xy\rangle. \qquad (87)\end{aligned}$$

All 150 positive threshold functions of four variables can be obtained in a similar fashion from the self-dual majority functions in the answer to exercise 84.

There are infinitely many sequences of weights $(w_1,w_2,\ldots,w_n)$, but only finitely many threshold functions for any given value of n. So it is clear that many different weight sequences are equivalent. For example, consider the pure majority function

$$\langle x_1^2x_2^3x_3^5x_4^7x_5^{11}x_6^{13}\rangle,$$

in which prime numbers have been used as weights. A brute-force examination of 2^6 cases shows that

$$\langle x_1^2x_2^3x_3^5x_4^7x_5^{11}x_6^{13}\rangle = \langle x_1x_2^2x_3^2x_4^3x_5^4x_6^5\rangle; \qquad (88)$$

thus we can express the same function with substantially smaller weights. Similarly, the threshold function

$$[(x_1x_2\ldots x_{20})_2 \ge (01100100100001111110)_2] = \langle 1^{225028}x_1^{524288}x_2^{262144}\ldots x_{20}\rangle,$$

a special case of (80), turns out to be simply

$$\langle 1^{323}x_1^{764}x_2^{323}x_3^{323}x_4^{118}x_5^{118}x_6^{87}x_7^{31}x_8^{31}x_9^{25}x_{10}^{6}x_{11}^{6}x_{12}^{6}x_{13}^{6}x_{14}x_{15}x_{16}x_{17}x_{18}x_{19}\rangle. \qquad (89)$$

Exercise 103 explains how to find a minimum set of weights without resorting to a huge brute-force search, using linear programming.

A nice indexing scheme by which a unique identifier can be assigned to any threshold function was discovered by C. K. Chow [*FOCS* **2** (1961), 34–38]. Given any Boolean function $f(x_1,\ldots,x_n)$, let $N(f)$ be the number of vectors $x=(x_1,\ldots,x_n)$ for which $f(x)=1$, and let $\Sigma(f)$ be the sum of all those vectors. For example, if $f(x_1,x_2)=x_1\vee x_2$, we have $N(f)=3$ and $\Sigma(f)=(0,1)+(1,0)+(1,1)=(2,2)$.

cancellation law
pure majority function
positive threshold functions
pi
Chow

Theorem T. *Let $f(x_1,\ldots,x_n)$ and $g(x_1,\ldots,x_n)$ be Boolean functions with $N(f)=N(g)$ and $\Sigma(f)=\Sigma(g)$, where f is a threshold function. Then $f=g$.*

Proof. Suppose there are exactly k vectors $x^{(1)}, \ldots, x^{(k)}$ such that $f(x^{(j)})=1$ and $g(x^{(j)})=0$. Since $N(f)=N(g)$, there must be exactly k vectors $y^{(1)}, \ldots, y^{(k)}$ such that $f(y^{(j)})=0$ and $g(y^{(j)})=1$. And since $\Sigma(f)=\Sigma(g)$, we must also have $x^{(1)}+\cdots+x^{(k)}=y^{(1)}+\cdots+y^{(k)}$.

Now suppose f is the threshold function (75); then we have $w\cdot x^{(j)}\geq t$ and $w\cdot y^{(j)}<t$ for $1\leq j\leq k$. But if $f\neq g$ we have $k>0$, and $w\cdot(x^{(1)}+\cdots+x^{(k)})\geq kt>w\cdot(y^{(1)}+\cdots+y^{(k)})$, a contradiction. ▮

Threshold functions have many curious properties, some of which are explored in the exercises below. Their classical theory is well summarized in Saburo Muroga's book *Threshold Logic and its Applications* (Wiley, 1971).

Symmetric Boolean functions. A function $f(x_1,\ldots,x_n)$ is called *symmetric* if $f(x_1,\ldots,x_n)$ is equal to $f(x_{p(1)},\ldots,x_{p(n)})$ for all permutations $p(1)\ldots p(n)$ of $\{1,\ldots,n\}$. When all the x_j are 0 or 1, this condition means that f depends only on the number of 1s that are present in the arguments, namely the "sideways sum" $\nu x=\nu(x_1,\ldots,x_n)=x_1+\cdots+x_n$. The notation $S_{k_1,k_2,\ldots,k_r}(x_1,\ldots,x_n)$ is commonly used to stand for the Boolean function that is true if and only if νx is either k_1 or k_2 or $\cdots$ or k_r. For example, $S_{1,3,5}(v,w,x,y,z)=v\oplus w\oplus x\oplus y\oplus z$; $S_{3,4,5}(v,w,x,y,z)=\langle vwxyz\rangle$; $S_{4,5}(v,w,x,y,z)=\langle 00vwxyz\rangle$.

Many applications of symmetry involve the basic functions $S_k(x_1,\ldots,x_n)$ that are true only when $\nu x=k$. For example, $S_3(x_1,x_2,x_3,x_4,x_5,x_6)$ is true if and only if exactly half of the arguments $\{x_1,\ldots,x_6\}$ are true and the other half are false. In such cases we obviously have

$$S_k(x_1,\ldots,x_n) \;=\; S_{\geq k}(x_1,\ldots,x_n)\wedge\overline{S_{\geq k+1}(x_1,\ldots,x_n)}, \tag{90}$$

where $S_{\geq k}(x_1,\ldots,x_n)$ is an abbreviation for $S_{k,k+1,\ldots,n}(x_1,\ldots,x_n)$. The functions $S_{\geq k}(x_1,\ldots,x_n)$ are, of course, the threshold functions $[x_1+\cdots+x_n\geq k]$ that we have already studied.

More complicated cases can be treated as threshold functions of threshold functions. For example, we have

$$\begin{aligned} S_{2,3,6,8,9}(x_1,\ldots,x_{12}) \;&=\; \bigl[\nu x\geq 2+4[\nu x\geq 4]+2[\nu x\geq 7]+5[\nu x\geq 10]\bigr] \\ &=\; \bigl\langle 00x_1\ldots x_{12}\langle 0^5\bar x_1\ldots\bar x_{12}\rangle^4\langle 1\bar x_1\ldots\bar x_{12}\rangle^2\langle 1^7\bar x_1\ldots\bar x_{12}\rangle^5\bigr\rangle, \end{aligned} \tag{91}$$

because the number of 1s in the outermost majority-of-25 turns out to be respectively $(11,12,13,14,11,12,13,12,13,14,10,11,12)$ when $x_1+\cdots+x_{12}=(0,1,\ldots,12)$. A similar two-level scheme works in general [R. C. Minnick, *IRE Trans.* **EC-10** (1961), 6–16]; and with three or more levels of logic we can reduce the number of thresholding operations even further. (See exercise 113.)

A variety of ingenious tricks have been discovered for evaluating symmetric Boolean functions. For example, S. Muroga attributes the following remarkable sequence of formulas to F. Sasaki:

$$\begin{aligned} &x_0\oplus x_1\oplus\cdots\oplus x_{2m}=\langle\bar x_0 s_1 s_2\ldots s_{2m}\rangle, \\ &\qquad\text{where}\quad s_j=\langle x_0x_jx_{j+1}\ldots x_{j+m-1}\bar x_{j+m}\bar x_{j+m+1}\ldots\bar x_{j+2m-1}\rangle, \end{aligned} \tag{92}$$

Muroga
symmetric Boolean functions–
sideways sum
notation
xor
threshold functions of threshold functions
Minnick
Muroga
Sasaki
parity function

if $m > 0$ and if we consider x_{2m+k} to be the same as x_k for $k \geq 1$. In particular, when $m = 1$ and $m = 2$ we have the identities

$$x_0 \oplus x_1 \oplus x_2 = \langle \bar{x}_0 \langle x_0 x_1 \bar{x}_2 \rangle \langle x_0 x_2 \bar{x}_1 \rangle \rangle; \tag{93}$$

$$x_0 \oplus \cdots \oplus x_4 = \langle \bar{x}_0 \langle x_0 x_1 x_2 \bar{x}_3 \bar{x}_4 \rangle \langle x_0 x_2 x_3 \bar{x}_4 \bar{x}_1 \rangle \langle x_0 x_3 x_4 \bar{x}_1 \bar{x}_2 \rangle \langle x_0 x_4 x_1 \bar{x}_2 \bar{x}_3 \rangle \rangle. \tag{94}$$

The right-hand sides are fully symmetric, but not obviously so! (See exercise 115.)

Canalizing functions
forcing
Kauffman
equivalence under permutations+
equivalence under permutations and complement
ternary Boolean operations, table

Canalizing functions. A Boolean function $f(x_1, \ldots, x_n)$ is said to be *canalizing* or "forcing" if we might be able to deduce its value by examining at most one of its variables. More precisely, f is canalizing if $n = 0$ or if there's a subscript j for which $f(x)$ either has a constant value when we set $x_j = 0$ or a constant value when we set $x_j = 1$. For example, $f(x, y, z) = (x \oplus z) \vee \bar{y}$ is canalizing because it always equals 1 when $y = 0$. (When $y = 1$ we don't know the value of f without examining also x and z; but half a loaf is better than none.) Such functions, introduced by Stuart Kauffman [*Lectures on Mathematics in the Life Sciences* **3** (1972), 63–116; *J. Theoretical Biology* **44** (1974), 167–190], have proved to be important in many applications, especially in chemistry and biology. Some of their properties are examined in exercises 125–129.

Quantitative considerations. We've been studying many different kinds of Boolean functions, so it's natural to ask: How many n-variable functions of each type actually exist? Tables 3, 4, and 5 provide the answers, at least for small values of n.

All functions are counted in Table 3. There are 2^{2^n} possibilities for each n, since there are 2^{2^n} possible truth tables. Some of these functions are self-dual, some are monotone; some are both monotone and self-dual, as in Theorem P. Some are Horn functions as in Theorem H; some are Krom functions as in Theorem S; and so on.

But in Table 4, two functions are considered identical if they differ only because the names of variables have changed. Thus only 12 different cases arise when $n = 2$, because (for example) $x \vee \bar{y}$ and $\bar{x} \vee y$ are essentially the same.

Table 5 goes a step further: It allows us to complement individual variables, and even to complement the entire function, without essentially changing it. From this perspective the 256 Boolean functions of (x, y, z) fall into only 14 different equivalence classes:

Representative	Class size	Representative	Class size
0	2	$x \wedge (y \oplus z)$	24
x	6	$x \oplus (y \wedge z)$	24
$x \wedge y$	24	$(x \wedge y) \vee (\bar{x} \wedge z)$	24
$x \oplus y$	6	$(x \vee y) \wedge (x \oplus z)$	48
$x \wedge y \wedge z$	16	$(x \oplus y) \vee (x \oplus z)$	8
$x \oplus y \oplus z$	2	$\langle xyz \rangle$	8
$x \wedge (y \vee z)$	48	$S_1(x, y, z)$	16

(95)

We shall study ways to count and to list inequivalent combinatorial objects in Section 7.2.3.

table of Boolean function counts
enumeration of Boolean functions
self-dual
monotone
monotone self-dual
Horn functions
Krom functions
threshold functions
self-dual threshold functions
symmetric functions
canalizing functions
Carroll
Tweedledee
false
true

Table 3

BOOLEAN FUNCTIONS OF n VARIABLES

	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$
arbitrary	2	4	16	256	65,536	4,294,967,296	18,446,744,073,709,551,616
self-dual	0	2	4	16	256	65,536	4,294,967,296
monotone	2	3	6	20	168	7,581	7,828,354
both	0	1	2	4	12	81	2,646
Horn	2	4	14	122	4,960	2,771,104	151,947,502,948
Krom	2	4	16	166	4,170	224,716	24,445,368
threshold	2	4	14	104	1,882	94,572	15,028,134
symmetric	2	4	8	16	32	64	128
canalizing	2	4	14	120	3,514	1,292,276	103,071,426,294

Table 4

BOOLEAN FUNCTIONS DISTINCT UNDER PERMUTATION OF VARIABLES

	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$
arbitrary	2	4	12	80	3,984	37,333,248	25,626,412,338,274,304
self-dual	0	2	2	8	32	1,088	6,385,408
monotone	2	3	5	10	30	210	16,353
both	0	1	1	2	3	7	30
Horn	2	4	10	38	368	29,328	216,591,692
Krom	2	4	12	48	308	3,028	49,490
threshold	2	4	10	34	178	1,720	590,440
canalizing	2	4	10	38	294	15,774	149,325,022

Table 5

BOOLEAN FUNCTIONS DISTINCT UNDER COMPLEMENTATION/PERMUTATION

	$n=0$	$n=1$	$n=2$	$n=3$	$n=4$	$n=5$	$n=6$
arbitrary	1	2	4	14	222	616,126	200,253,952,527,184
self-dual	0	1	1	3	7	83	109,950
threshold	1	2	3	6	15	63	567
both	0	1	1	2	3	7	21
canalizing	1	2	3	6	22	402	1,228,158

EXERCISES

1. [*15*] (Lewis Carroll.) Make sense of Tweedledee's comment, quoted near the beginning of this section. [*Hint:* See Table 1.]

2. [*17*] Logicians on the remote planet Pincus use the symbol 1 to represent "false" and 0 to represent "true." Thus, for example, they have a binary operation called "or" whose properties

$$1 \text{ or } 1 = 1, \qquad 1 \text{ or } 0 = 0, \qquad 0 \text{ or } 1 = 0, \qquad 0 \text{ or } 0 = 0$$

we associate with $\wedge$. What operations would we associate with the 16 logical operators that Pincusians respectively call "falsehood," "and," . . . , "nand," "validity" (see Table 1)?

▶ **3.** [*13*] Suppose logical values were respectively -1 for falsehood and $+1$ for truth, instead of 0 and 1. What operations $\circ$ in Table 1 would then correspond to (a) $\max(x, y)$? (b) $\min(x, y)$? (c) $-x$? (d) $x \cdot y$?

4. [*24*] (H. M. Sheffer.) The purpose of this exercise is to show that all of the operations in Table 1 can be expressed in terms of NAND. (a) For each of the 16 operators $\circ$ in that table, find a formula equivalent to $x \circ y$ that uses only $\overline{\wedge}$ as an operator. Your formula should be as short as possible. For example, the answer for operation $\llcorner$ is simply "x", but the answer for $\overline{\llcorner}$ is "$x \,\overline{\wedge}\, x$". Do not use the constants 0 or 1 in your formulas. (b) Similarly, find 16 short formulas when constants *are* allowed. For example, $x \,\overline{\llcorner}\, y$ can now be expressed also as "$x \,\overline{\wedge}\, 1$".

5. [*24*] Consider exercise 4 with $\overline{\subset}$ as the basic operation instead of $\overline{\wedge}$.

6. [*21*] (E. Schröder.) (a) Which of the 16 operations in Table 1 are associative — in other words, which of them satisfy $x \circ (y \circ z) = (x \circ y) \circ z$? (b) Which of them satisfy the identity $(x \circ y) \circ (y \circ z) = x \circ z$?

7. [*20*] Which operations in Table 1 have the property that $x \circ y = z$ if and only if $y \circ z = x$?

8. [*24*] Which of the 16^2 pairs of operations $(\circ, \square)$ satisfy the left-distributive law $x \circ (y \,\square\, z) = (x \circ y) \,\square\, (x \circ z)$?

9. [*16*] True or false? (a) $(x \oplus y) \vee z = (x \vee z) \oplus (y \vee z)$; (b) $(w \oplus x \oplus y) \vee z = (w \vee z) \oplus (x \vee z) \oplus (y \vee z)$; (c) $(x \oplus y) \vee (y \oplus z) = (x \oplus z) \vee (y \oplus z)$.

10. [*17*] What is the multilinear representation of the "random" function (22)?

11. [*M25*] Is there an intuitive way to understand exactly when the multilinear representation of $f(x_1, \ldots, x_n)$ contains, say, the term $x_2x_3x_6x_8$? (See (19).)

▶ **12.** [*M23*] The *integer multilinear representation* of a Boolean function extends representations like (19) to a polynomial $f(x_1, \ldots, x_n)$ with integer coefficients, in such a way that $f(x_1, \ldots, x_n)$ has the correct value (0 or 1) for all 2^n possible 0–1 vectors $(x_1, \ldots, x_n)$, *without* taking a remainder mod 2. For example, the integer multilinear representation corresponding to (19) is $1 - xy - xz - yz + 3xyz$.

a) What is the integer multilinear representation of the "random" function (22)?
b) How large can the coefficients of such a representation $f(x_1, \ldots, x_n)$ be?
c) Show that, in every integer multilinear representation, $0 \le f(x_1, \ldots, x_n) \le 1$ whenever $x_1, \ldots, x_n$ are real numbers with $0 \le x_1, \ldots, x_n \le 1$.
d) Similarly, if $f(x_1, \ldots, x_n) \le g(x_1, \ldots, x_n)$ whenever $\{x_1, \ldots, x_n\} \subseteq \{0, 1\}$, then $f(x_1, \ldots, x_n) \le g(x_1, \ldots, x_n)$ whenever $\{x_1, \ldots, x_n\} \subseteq [0\,.\,.\,1]$.
e) If f is monotone and $0 \le x_j \le y_j \le 1$ for $1 \le j \le n$, prove that $f(x) \le f(y)$.

▶ **13.** [*20*] Consider a system that consists of n units, each of which may be "working" or "failing." If x_j represents the condition "unit j is working," then a Boolean function like $x_1 \wedge (\bar{x}_2 \vee \bar{x}_3)$ represents the statement "unit 1 is working, but either unit 2 or unit 3 is failing"; and $S_3(x_1, \ldots, x_n)$ means "exactly three units are working."

Suppose each unit j is in working order with probability p_j, independent of the other units. Show that the Boolean function $f(x_1, \ldots, x_n)$ is true with probability $F(p_1, \ldots, p_n)$, where F is a polynomial in the variables $p_1, \ldots, p_n$.

14. [*20*] The probability function $F(p_1, \ldots, p_n)$ in exercise 13 is often called the *availability* of the system. Find the self-dual function $f(x_1, x_2, x_3)$ of maximum availability when the probabilities (p_1, p_2, p_3) are (a) $(.9, .8, .7)$; (b) $(.8, .6, .4)$; (c) $(.8, .6, .1)$.

Sheffer
NAND
Schröder
associative
distributive
multilinear representation
pi, as "random" example+
multilinear representation
integer multilinear representation
working
failing
reliability
availability
self-dual function

▶ **15.** [*M20*] If $f(x_1,\ldots,x_n)$ is any Boolean function, show that there is a polynomial $F(x)$ with the property that $F(x)$ is an integer when x is an integer, and $f(x_1,\ldots,x_n) = F\bigl((x_n\ldots x_1)_2\bigr) \bmod 2$. *Hint:* Consider $\binom{x}{k} \bmod 2$.

16. [*13*] Can we replace each $\vee$ by $\oplus$ in a full disjunctive normal form?

17. [*10*] By De Morgan's laws, a general disjunctive normal form such as (25) is not only an OR of ANDs, it is a NAND of NANDs:

$$\overline{\overline{(u_{11}\wedge\cdots\wedge u_{1s_1})}\;\wedge\cdots\wedge\;\overline{(u_{m1}\wedge\cdots\wedge u_{ms_m})}}.$$

Both levels of logic can therefore be considered to be identical.

A student named J. H. Quick rewrote this expression in the form

$$(u_{11}\,\overline{\wedge}\cdots\overline{\wedge}\,u_{1s_1})\;\overline{\wedge}\cdots\overline{\wedge}\;(u_{m1}\,\overline{\wedge}\cdots\overline{\wedge}\,u_{ms_m}).$$

Was that a good idea?

▶ **18.** [*20*] Let $u_1\wedge\cdots\wedge u_s$ be an implicant in a disjunctive normal form for a Boolean function f, and let $v_1\vee\cdots\vee v_t$ be a clause in a conjunctive normal form for the same function. Prove that $u_i = v_j$ for some i and j.

19. [*20*] What is the conjunctive prime form of the "random" function in (22)?

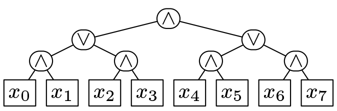
20. [*M21*] True or false: Every prime implicant of $f\wedge g$ can be written $f'\wedge g'$, where f' is a prime implicant of f and g' is a prime implicant of g.

21. [*M20*] Prove that a nonconstant Boolean function is monotone if and only if it can be expressed entirely in terms of the operations $\wedge$ and $\vee$.

22. [*20*] Suppose $f(x_1,\ldots,x_n) = g(x_1,\ldots,x_{n-1}) \oplus h(x_1,\ldots,x_{n-1})\wedge x_n$ as in (16). What conditions on the functions g and h are necessary and sufficient for f to be monotone?

23. [*15*] What is the conjunctive prime form of $(v\wedge w\wedge x)\vee(v\wedge x\wedge z)\vee(x\wedge y\wedge z)$?

24. [*M20*] Consider the complete binary tree with 2^k leaves, illustrated here for $k=3$. Operate alternately with $\wedge$ or $\vee$ on each level, using $\wedge$ at the root, obtaining for example $((x_0\wedge x_1)\vee(x_2\wedge x_3))\wedge((x_4\wedge x_5)\vee(x_6\wedge x_7))$. How many prime implicants does the resulting function contain?



25. [*M21*] How many prime implicants does $(x_1\vee x_2)\wedge(x_2\vee x_3)\wedge\cdots\wedge(x_{n-1}\vee x_n)$ have?

26. [*M23*] Let $\mathcal{F}$ and $\mathcal{G}$ be the families of index sets for the prime clauses and the prime implicants of a monotone CNF and a monotone DNF:

$$f(x) = \bigwedge_{I\in\mathcal{F}}\bigvee_{i\in I} x_i; \qquad g(x) = \bigvee_{J\in\mathcal{G}}\bigwedge_{j\in J} x_j.$$

Exhibit an x such that $f(x)\ne g(x)$ if any of the following conditions hold:

a) There is an $I\in\mathcal{F}$ and a $J\in\mathcal{G}$ with $I\cap J=\emptyset$.
b) $\bigcup_{I\in\mathcal{F}} I \ne \bigcup_{J\in\mathcal{G}} J$.
c) There's an $I\in\mathcal{F}$ with $|I|>|\mathcal{G}|$, or a $J\in\mathcal{G}$ with $|J|>|\mathcal{F}|$.
d) $\sum_{I\in\mathcal{F}} 2^{n-|I|} + \sum_{J\in\mathcal{G}} 2^{n-|J|} < 2^n$, where $n = \bigl|\bigcup_{I\in\mathcal{F}} I\bigr|$.

27. [*M31*] Continuing the previous exercise, consider the following algorithm $\mathrm{X}(\mathcal{F},\mathcal{G})$, which either returns a vector x with $f(x)\ne g(x)$, or returns Λ if $f=g$:

polynomial
full disjunctive normal form
De Morgan
Quick
implicant
disjunctive normal form
clause
conjunctive normal form
conjunctive prime form
monotone
complete binary tree
Krom function

Quine
shortest DNF
bitwise operations
subcube
asterisks
buddies
tag
maximal subcube
prime implicant

X1. [Check necessary conditions.] Return an appropriate value x if condition (a), (b), (c), or (d) in exercise 26 applies.

X2. [Done?] If $|\mathcal{F}||\mathcal{G}| \le 1$, return Λ.

X3. [Recurse.] Compute the following reduced families, for a "best" index k:

$$\mathcal{F}_1 = \{I \mid I \in \mathcal{F},\ k \notin I\}, \qquad \mathcal{F}_0 = \mathcal{F}_1 \cup \{I \mid k \notin I,\ I \cup \{k\} \in \mathcal{F}\};$$
$$\mathcal{G}_0 = \{J \mid J \in \mathcal{G},\ k \notin J\}, \qquad \mathcal{G}_1 = \mathcal{G}_0 \cup \{J \mid k \notin J,\ J \cup \{k\} \in \mathcal{G}\}.$$

Delete any member of $\mathcal{F}_0$ or $\mathcal{G}_1$ that contains another member of the same family. The index k should be chosen so that the ratio $\rho = \min(|\mathcal{F}_1|/|\mathcal{F}|, |\mathcal{G}_0|/|\mathcal{G}|)$ is as small as possible. If $\mathrm{X}(\mathcal{F}_0, \mathcal{G}_0)$ returns a vector x, return the same vector extended with $x_k = 0$. Otherwise if $\mathrm{X}(\mathcal{F}_1, \mathcal{G}_1)$ returns a vector x, return the same vector extended with $x_k = 1$. Otherwise return Λ. ▌

If $N = |\mathcal{F}| + |\mathcal{G}|$, prove that step X1 is executed at most $N^{O(\log N)^2}$ times. *Hint:* Show that we always have $\rho \le 1 - 1/\lg N$ in step X3.

28. [*21*] (W. V. Quine, 1952.) If $f(x_1, \ldots, x_n)$ is a Boolean function with prime implicants $p_1, \ldots, p_q$, let $g(y_1, \ldots, y_q) = \bigwedge_{f(x)=1} \bigvee\{y_j \mid p_j(x) = 1\}$. For example, the "random" function (22) is true at the eight points (28), and it has five prime implicants given by (29) and (30); so $g(y_1, \ldots, y_5)$ is

$$(y_1 \vee y_2) \wedge (y_1) \wedge (y_2 \vee y_3) \wedge (y_4) \wedge (y_3 \vee y_5) \wedge (y_5) \wedge (y_5) \wedge (y_4 \vee y_5)$$
$$= (y_1 \wedge y_2 \wedge y_4 \wedge y_5) \vee (y_1 \wedge y_3 \wedge y_4 \wedge y_5)$$

in this case. Prove that every shortest DNF expression for f corresponds to a prime implicant of the monotone function g.

29. [*22*] (The next several exercises are devoted to algorithms that deal with the implicants of Boolean functions by representing points of the n-cube as n-bit numbers $(b_{n-1} \ldots b_1 b_0)_2$, rather than as bit strings $x_1 \ldots x_n$.) Given a bit position j, and given n-bit values $v_0 < v_1 < \cdots < v_{m-1}$, explain how to find all pairs (k, k') such that $0 \le k < k' < m$ and $v_{k'} = v_k \oplus 2^j$, in increasing order of k. The running time of your procedure should be $O(m)$, if bitwise operations on n-bit words take constant time.

▸ **30.** [*27*] The text points out that an implicant of a Boolean function can be regarded as a subcube such as 01∗0∗, contained in the set V of all points for which the function is true. Every subcube can be represented as a pair of binary numbers $a = (a_{n-1} \ldots a_0)_2$ and $b = (b_{n-1} \ldots b_0)_2$, where a records the positions of the asterisks and b records the bits in non-∗ positions. For example, the numbers $a = (00101)_2$ and $b = (01000)_2$ represent the subcube $c = 01{*}0{*}$. We always have $a \,\&\, b = 0$.

The "j-buddy" of a subcube is defined whenever $a_j = 0$, by changing b to $b \oplus 2^j$. For example, 01∗0∗ has three buddies, namely its 4-buddy 11∗0∗, its 3-buddy 00∗0∗, and its 1-buddy 01∗1∗. Every subcube $c \subseteq V$ can be assigned a tag value $(t_{n-1} \ldots t_0)_2$, where $t_j = 1$ if and only if the j-buddy of c is defined and contained in V. With this definition, c represents a maximal subcube (hence a prime implicant) if and only if its tag is zero.

Use these concepts to design an algorithm that finds all maximal subcubes (a, b) of a given set V, where V is represented by the n-bit numbers $v_0 < v_1 < \cdots < v_{m-1}$.

▸ **31.** [*28*] The algorithm in exercise 30 requires a complete list of all points where a Boolean function is true, and that list may be quite long. Therefore we may prefer to work directly with subcubes, never going down to the level of explicit n-tuples unless necessary. The key to such higher-level methods is the notion of *consensus* between subcubes c and c', denoted by $c \sqcup c'$ and defined to be the largest subcube c'' such that

$$c'' \subseteq c \cup c', \qquad c'' \not\subseteq c, \quad \text{and} \quad c'' \not\subseteq c'.$$

Such a c'' does not always exist. For example, if $c = 000*$ and $c' = *111$, every subcube contained in $c \cup c'$ is contained either in c or in c'.

a) Prove that the consensus, when it exists, can be computed componentwise using the following formulas in each coordinate position:

$$x \sqcup x = x \sqcup * = * \sqcup x = x \quad \text{and} \quad x \sqcup \bar{x} = * \sqcup * = *, \qquad \text{for } x = 0 \text{ and } x = 1.$$

Furthermore, $c \sqcup c'$ exists if and only if the rule $x \sqcup \bar{x} = *$ has been used in exactly one component.

b) A subcube with k asterisks is called a k-cube. Show that, if c is a k-cube and c' is a k'-cube, and if the consensus $c'' = c \sqcup c'$ exists, then c'' is a k''-cube where $1 \le k'' \le \min(k, k') + 1$.

c) If C and C' are families of subcubes, let

$$C \sqcup C' \;=\; \{\, c \sqcup c' \mid c \in C,\ c' \in C', \text{ and } c \sqcup c' \text{ exists}\}.$$

Explain why the following algorithm works.

Algorithm E (*Find maximal subcubes*). Given a family C of subcubes of the n-cube, this algorithm outputs the maximal subcubes of $V = \bigcup_{c \in C} c$, without actually computing the set V itself.

E1. [Initialize.] Set $j \leftarrow 0$. Delete any subcube c of C that is contained in another.

E2. [Done?] (At this point, every j-cube $\subseteq V$ is contained in some element of C, and C contains no k-cubes with $k < j$.) If C is empty, the algorithm terminates.

E3. [Take consensuses.] Set $C' \leftarrow C \sqcup C$, and remove all subcubes from C' that are k-cubes for $k \le j$. While performing this computation, also output any j-cube $c \in C$ for which $c \sqcup C$ does not produce a $(j+1)$-cube of C'.

E4. [Advance.] Set $C \leftarrow C \cup C'$, but delete all j-cubes from this union. Then delete any subcube $c \in C$ that is contained in another. Set $j \leftarrow j+1$ and go to E2. ▌

(See exercise 7.1.3–142 for an efficient way to perform these computations.)

▸ **32.** [*M29*] Let $c_1, \ldots, c_m$ be subcubes of the n-cube.

a) Prove that $c_1 \cup \cdots \cup c_m$ contains at most one maximal subcube c that is not contained in $c_1 \cup \cdots \cup c_{j-1} \cup c_{j+1} \cup \cdots \cup c_m$ for any $j \in \{1, \ldots, m\}$. (If c exists, we call it the *generalized consensus* of $c_1, \ldots, c_m$, because $c = c_1 \sqcup c_2$ in the notation of exercise 31 when $m = 2$.)

b) Find a set of m subcubes for which each of the $2^m - 1$ nonempty subsets of $\{c_1, \ldots, c_m\}$ has a generalized consensus.

c) Prove that a DNF with m implicants has at most $2^m - 1$ prime implicants.

d) Find a DNF that has m implicants and $2^m - 1$ prime implicants.

33. [*M21*] Let $f(x_1, \ldots, x_n)$ be one of the $\binom{2^n}{m}$ Boolean functions that are true at exactly m points. If f is chosen at random, what is the probability that $x_1 \wedge \cdots \wedge x_k$ is (a) an implicant of f? (b) a prime implicant of f? [Give the answer to part (b) as a sum; but evaluate it in closed form when $k = n$.]

consensus
generalized consensus
random

orthogonal
ODNF, see Orthogonal DNF
reliability polynomial
monotone
asterisk codes
shadow
consecutive 1s
shelling

▸ **34.** [*HM37*] Continuing exercise 33, let $c(m,n)$ be the average total number of implicants, and let $p(m,n)$ be the average total number of prime implicants.

a) If $0 \le m \le 2^n/n$, show that $m \le c(m,n) \le \frac{3}{2}m + O(m/n)$ and $p(m,n) \ge me^{-1} + O(m/n)$; hence $p(m,n) = \Theta(c(m,n))$ in this range.

b) Now let $2^n/n \le m \le (1-\epsilon)2^n$, where ϵ is a fixed positive constant. Define the numbers t and α_{mn} by the relations

$$n^{-4/3} \le \left(\frac{m}{2^n}\right)^{2^t} = \alpha_{mn} < n^{-2/3}, \qquad \text{integer } t.$$

Express the asymptotic values of $c(m,n)$ and $p(m,n)$ in terms of n, t, and α_{mn}. [*Hint:* Show that almost all of the implicants have exactly $n-t$ or $n-t-1$ literals.]

c) Estimate $c(m,n)/p(m,n)$ when $m = 2^{n-1}$ and $n = \lfloor(\ln t - \ln\ln t)2^{2^t}\rfloor$, integer t.

d) Prove that $c(m,n)/p(m,n) = O(\log\log n/\log\log\log n)$ when $m \le (1-\epsilon)2^n$.

▸ **35.** [*M25*] A DNF is called *orthogonal* if its implicants correspond to disjoint subcubes. Orthogonal disjunctive normal forms are particularly useful when the reliability polynomial of exercise 13 is being calculated or estimated.

The full DNF of every function is obviously orthogonal, because its subcubes are single points. But we can often find an orthogonal DNF that has significantly fewer implicants, especially when the function is monotone. For example, the function $(x_1\wedge x_2) \vee (x_2\wedge x_3) \vee (x_3\wedge x_4)$ is true at eight points, and it has the orthogonal DNF

$$(x_1\wedge x_2) \vee (\bar x_1\wedge x_2\wedge x_3) \vee (\bar x_2\wedge x_3\wedge x_4).$$

In other words, the overlapping subcubes $11{*}{*}$, ${*}11{*}$, ${*}{*}11$ can be replaced by the disjoint subcubes $11{*}{*}$, $011{*}$, ${*}011$. Using the binary notation for subcubes in exercise 30, these subcubes have asterisk codes 0011, 0001, 1000 and bit codes 1100, 0110, 0011.

Every monotone function can be defined by a list of bit codes $B_1, \ldots, B_p$, when the asterisk codes are respectively $\bar B_1, \ldots, \bar B_p$. Given such a list, let the "shadow" S_k of B_k be the bitwise OR of $B_j \mathbin{\&} \bar B_k$, for all $1 \le j < k$ such that $\nu(B_j \mathbin{\&} \bar B_k) = 1$:

$$S_k = \beta_{1k} \mid \cdots \mid \beta_{(k-1)k}, \quad \beta_{jk} = \bigl((B_j\mathbin{\&}\bar B_k) \oplus ((B_j\mathbin{\&}\bar B_k)-1)\bigr) \mathbin{\dot-} ((B_j\mathbin{\&}\bar B_k)-1).$$

For example, when the bit codes are $(B_1, B_2, B_3) = (1100, 0110, 0011)$, we get the shadow codes $(S_1, S_2, S_3) = (0000, 1000, 0100)$.

a) Show that the asterisk codes $A'_j = \bar B_j - S_j$ and bit codes B_j define subcubes that cover the same points as the subcubes with asterisk codes $A_j = \bar B_j$.

b) A list of bit codes $B_1, \ldots, B_p$ is called a *shelling* if $B_j \mathbin{\&} S_k$ is nonzero for all $1 \le j < k \le p$. For example, $(1100, 0110, 0011)$ is a shelling; but if we arrange those bit codes in the order $(1100, 0011, 0110)$ the shelling condition fails when $j = 1$ and $k = 2$, although we do have $S_3 = 1001$. Prove that the subcubes in part (a) are disjoint if and only if the list of bit codes is a shelling.

c) According to Theorem Q, every prime implicant must appear among the B's when we represent a monotone Boolean function in this way. But sometimes we need to add additional implicants if we want the subcubes to be disjoint. For example, there is no shelling for the bit codes 1100 and 0011. Show that we can, however, obtain a shelling for this function $(x_1\wedge x_2)\vee(x_3\wedge x_4)$ by adding one more bit code. What is the resulting orthogonal DNF?

d) Permute the bit codes $\{11000, 01100, 00110, 00011, 11010\}$ to obtain a shelling.

e) Add two bit codes to the set $\{110000, 011000, 001100, 000110, 000011\}$ in order to make a shellable list.

36. [*M21*] Continuing exercise 35, let f be any monotone function, not identically 1. Show that the set of bit vectors

$$B = \{\, x \mid f(x) = 1 \text{ and } f(x') = 0 \,\}, \qquad x' = x \,\&\, (x-1),$$

is always shellable when listed in decreasing lexicographic order. (The vector x' is obtained from x by zeroing out the rightmost 1.) For example, this method produces an orthogonal DNF for $(x_1 \wedge x_2) \vee (x_3 \wedge x_4)$ from the list $(1100, 1011, 0111, 0011)$.

▶ **37.** [*M31*] Find a shellable DNF for $(x_1 \wedge x_2) \vee (x_3 \wedge x_4) \vee \cdots \vee (x_{2n-1} \wedge x_{2n})$ that has $2^n - 1$ implicants, and prove that no orthogonal DNF for this function has fewer.

38. [*05*] Is it hard to test the satisfiability of functions in *disjunctive* normal form?

▶ **39.** [*25*] Let $f(x_1, \ldots, x_n)$ be a Boolean formula represented as an extended binary tree with N internal nodes and $N+1$ leaves. Each leaf is labeled with a variable x_k, and each internal node is labeled with one of the sixteen binary operators in Table 1; applying the operators from bottom to top yields $f(x_1, \ldots, x_n)$ as the value of the root.

Explain how to construct a formula $F(x_1, \ldots, x_n, y_1, \ldots, y_N)$ in 3CNF, having exactly $4N+1$ clauses, such that $f(x_1, \ldots, x_n) = \exists y_1 \ldots \exists y_N F(x_1, \ldots, x_n, y_1, \ldots, y_N)$. (Thus f is satisfiable if and only if F is satisfiable.)

40. [*23*] Given an undirected graph G, construct the following clauses on the Boolean variables $\{p_{uv} \mid u \neq v\} \cup \{q_{uvw} \mid u \neq v,\ u \neq w,\ v \neq w,\ u \not- w\}$, where u, v, and w denote vertices of G:

$$A = \bigwedge \{\, (p_{uv} \vee p_{vu}) \wedge (\bar p_{uv} \vee \bar p_{vu}) \mid u \neq v \,\};$$
$$B = \bigwedge \{\, (\bar p_{uv} \vee \bar p_{vw} \vee p_{uw}) \mid u \neq v,\ u \neq w,\ v \neq w \,\};$$
$$C = \bigwedge \{\, (\bar q_{uvw} \vee p_{uv}) \wedge (\bar q_{uvw} \vee p_{vw}) \wedge (q_{uvw} \vee \bar p_{uv} \vee \bar p_{vw}) \mid u \neq v,\ u \neq w,\ v \neq w,\ u \not- w \,\};$$
$$D = \bigwedge \{ (\textstyle\bigvee_{v \notin \{u,w\}} (q_{uvw} \vee q_{wvu})) \mid u \neq w,\ u \not- w \,\}.$$

Prove that the formula $A \wedge B \wedge C \wedge D$ is satisfiable if and only if G has a Hamiltonian path. *Hint:* Think of p_{uv} as the statement '$u < v$'.

41. [*20*] (*The pigeonhole principle.*) The island of San Serriffe contains m pigeons and n holes. Find a conjunctive normal form that is satisfiable if and only if each pigeon can be the sole occupant of at least one hole.

42. [*20*] Find a short, unsatisfiable CNF that is not totally trivial, although it consists entirely of Horn clauses that are also Krom clauses.

43. [*20*] Is there an efficient way to decide satisfiability of a conjunctive normal form that consists entirely of Horn clauses and/or Krom clauses (possibly mixed)?

44. [*M23*] Complete the proof of Theorem H by studying the implications of (33).

45. [*M20*] (a) Show that exactly half of the Horn functions of n variables are definite. (b) Also show that there are more Horn functions of n variables than monotone functions of n variables (unless $n = 0$).

46. [*20*] Which of the 11×11 character pairs `xy` can occur next to each other in the context-free grammar (34)?

47. [*20*] Given a sequence of relations $j \prec k$ with $1 \leq j, k \leq n$ as in Algorithm 2.2.3T (topological sorting), consider the clauses

$$x_{j_1} \wedge \cdots \wedge x_{j_t} \Rightarrow x_k \qquad \text{for } 1 \leq k \leq n,$$

where $\{j_1, \ldots, j_t\}$ is the set of elements such that $j_i \prec k$. Compare the behavior of Algorithm C on these clauses to the behavior of Algorithm 2.2.3T.

lexicographic order
satisfiability
disjunctive
binary tree
Hamiltonian path
pigeonhole principle
Horn clauses
Krom clauses
definite
monotone functions
context-free grammar
topological sorting

▶ **48.** [*21*] What's a good way to test a set of Horn clauses for satisfiability?

49. [*22*] Show that, if $f(x_1,\ldots,x_n)$ and $g(x_1,\ldots,x_n)$ are both defined by Horn clauses in CNF, there is an easy way to test if $f(x_1,\ldots,x_n) \le g(x_1,\ldots,x_n)$ for all $x_1, \ldots, x_n$.

50. [*HM42*] There are $(n+2)2^{n-1}$ possible Horn clauses on n variables. Select $c\cdot 2^n$ of them at random, with repetition permitted, where $c > 0$; and let $P_n(c)$ be the probability that all of the selected clauses are simultaneously satisfiable. Prove that

$$\lim_{n\to\infty} P_n(c) \;=\; 1-(1-e^{-c})(1-e^{-2c})(1-e^{-4c})(1-e^{-8c})\ldots.$$

▶ **51.** [*22*] A great many two-player games can be defined by specifying a directed graph in which each vertex represents a game position. There are two players, Alice and Bob, who construct an oriented path by starting at a particular vertex and taking turns to extend the path, one arc at a time. Before the game starts, each vertex has either been marked A (meaning that Alice wins), or marked B (meaning that Bob wins), or marked C (meaning that the cat wins), or left unmarked.

When the path reaches a vertex v marked A or B, that player wins. The game stops without a winner if v has been visited before, with the same player to move. If v is marked C, the currently active player has the option of accepting a draw; otherwise he or she must choose an outgoing arc to extend the path, and the other player becomes active. (If v is an unmarked vertex with out-degree zero, the active player loses.)

Associating four propositional variables $A^+(v)$, $A^-(v)$, $B^+(v)$, and $B^-(v)$ with every vertex v of the graph, explain how to construct a set of definite Horn clauses such that $A^+(v)$ is in the core if and only if Alice can force a win when the path starts at v and she moves first; $A^-(v)$ is in the core if and only if Bob can force her to lose in that game; $B^+(v)$ and $B^-(v)$ are similar to $A^+(v)$ and $A^-(v)$, but with roles reversed.

52. [*25*] (*Boolean games.*) Any Boolean function $f(x_1,\ldots,x_n)$ leads to a game called "two steps forward or one step back," in the following way: There are two players, 0 and 1, who repeatedly assign values to the variables x_j; player y tries to make $f(x_1,\ldots,x_n)$ equal to y. Initially all variables are unassigned, and the position marker m is zero. Players take turns, and the currently active player either sets $m \leftarrow m+2$ (if $m+2 \le n$) or $m \leftarrow m-1$ (if $m-1\ge 1$), then sets

$$\begin{cases} x_m \leftarrow 0 \text{ or } 1, & \text{if } x_m \text{ was not previously assigned;}\\ x_m \leftarrow \bar{x}_m, & \text{if } x_m \text{ was previously assigned.}\end{cases}$$

The game is over as soon as a value has been assigned to all variables; then $f(x_1,\ldots,x_n)$ is the winner. A draw is declared if the same state (including the value of m) is reached twice. Notice that at most four moves are possible at any time.

Study examples of this game when $2 \le n \le 9$, in the following four cases:

a) $f(x_1,\ldots,x_n) = [x_1\ldots x_n < x_n \ldots x_1]$ (in lexicographic order);
b) $f(x_1,\ldots,x_n) = x_1 \oplus \cdots \oplus x_n$;
c) $f(x_1,\ldots,x_n) = [x_1\ldots x_n$ contains no two consecutive 1s];
d) $f(x_1,\ldots,x_n) = [(x_1\ldots x_n)_2$ is prime].

53. [*23*] Show that the impossible comedy festival of (37) *can* be scheduled if a change is made to the requirements of only (a) Tomlin; (b) Unwin; (c) Vegas; (d) Xie; (e) Yankovic; (f) Zany.

54. [*20*] Let $S = \{u_1, u_2, \ldots, u_k\}$ be the set of literals in some strong component of a digraph that corresponds to a 2CNF formula as in Fig. 6. Show that S contains both a variable and its complement if and only if $u_j = \bar{u}_1$ for some j with $2 \le j \le k$.

games
cat's game
Horn clauses
core
Boolean games
Lenin
Ulyanov
lexicographic order
consecutive 1s
comedy festival
Tomlin

▸ **55.** [*30*] Call $f(x_1,\ldots,x_n)$ a *renamed Horn function* if there are Boolean constants $y_1, \ldots, y_n$ such that $f(x_1 \oplus y_1,\ldots,x_n \oplus y_n)$ is a Horn function.

a) Given $f(x_1,\ldots,x_n)$ in CNF, explain how to construct $g(y_1,\ldots,y_n)$ in 2CNF so that the clauses of $f(x_1 \oplus y_1,\ldots,x_n \oplus y_n)$ are Horn clauses if and only if $g(y_1,\ldots,y_n)=1$.

b) Design an algorithm that decides in $O(m)$ steps whether or not all clauses of a given CNF of length m can be converted into Horn clauses by complementing some subset of the variables.

▸ **56.** [*20*] The satisfiability problem for a Boolean function $f(x_1,x_2,\ldots,x_n)$ can be stated formally as the question of whether or not the quantified formula

$$\exists x_1\,\exists x_2 \ldots \exists x_n\, f(x_1,x_2,\ldots,x_n)$$

is true; here '$\exists x_j\,\alpha$' means, "there exists a Boolean value x_j such that α holds."

A much more general evaluation problem arises when we replace one or more of the existential quantifiers $\exists x_j$ by the universal quantifier $\forall x_j$, where '$\forall x_j\,\alpha$' means, "for all Boolean values x_j, α holds."

Which of the eight quantified formulas $\exists x\,\exists y\,\exists z\, f(x,y,z)$, $\exists x\,\exists y\,\forall z\, f(x,y,z)$, $\ldots$, $\forall x\,\forall y\,\forall z\, f(x,y,z)$ are true when $f(x,y,z) = (x\vee y) \wedge (\bar{x}\vee z) \wedge (y\vee \bar{z})$?

▸ **57.** [*30*] (B. Aspvall, M. F. Plass, and R. E. Tarjan.) Continuing exercise 56, design an algorithm that decides in linear time whether or not a given fully quantified formula $f(x_1,\ldots,x_n)$ is true, when f is any formula in 2CNF (any conjunction of Krom clauses).

▸ **58.** [*37*] Continuing exercise 57, design an efficient algorithm that decides whether or not a given fully quantified conjunction of *Horn* clauses is true.

▸ **59.** [*M20*] (D. Pehoushek and R. Fraer, 1997.) If the truth table for $f(x_1,x_2,\ldots,x_n)$ has a 1 in exactly k places, show that exactly k of the fully quantified formulas $\mathbb{Q}x_1\,\mathbb{Q}x_2 \ldots \mathbb{Q}x_n\, f(x_1,x_2,\ldots,x_n)$ are true, when each $\mathbb{Q}$ is either $\exists$ or $\forall$.

60. [*12*] Which of the following expressions yield the median $\langle xyz\rangle$, as defined in (43)? (a) $(x\wedge y) \oplus (y\wedge z) \oplus (x\wedge z)$. (b) $(x\vee y) \oplus (y\vee z) \oplus (x\vee z)$. (c) $(x\oplus y) \wedge (y\oplus z) \wedge (x\oplus z)$. (d) $(x\equiv y) \oplus (y\equiv z) \oplus (x\equiv z)$. (e) $(x\barwedge y) \wedge (y\barwedge z) \wedge (x\barwedge z)$. (f) $(x\barwedge y) \vee (y\barwedge z) \vee (x\barwedge z)$.

61. [*13*] True or false: If $\circ$ is any one of the Boolean binary operations in Table 1, we have the distributive law $w \circ \langle xyz\rangle = \langle (w\circ x)(w\circ y)(w\circ z)\rangle$.

62. [*25*] (C. Schensted.) If $f(x_1,\ldots,x_n)$ is a monotone Boolean function and $n \geq 3$, prove the median expansion formula

$$f(x_1,\ldots,x_n) = \langle f(x_1,x_1,x_3,x_4,\ldots,x_n)\,f(x_1,x_2,x_2,x_4,\ldots,x_n)\,f(x_3,x_2,x_3,x_4,\ldots,x_n)\rangle.$$

63. [*20*] Equation (49) shows how to compute the median of five elements via medians of three. Conversely, can we compute $\langle xyz\rangle$ with a subroutine for medians of five?

64. [*23*] (S. B. Akers, Jr.) (a) Prove that a Boolean function $f(x_1,\ldots,x_n)$ is monotone and self-dual if and only if it satisfies the following condition:

For all $x = x_1 \ldots x_n$ and $y = y_1 \ldots y_n$ there exists k such that $f(x) = x_k$ and $f(y) = y_k$.

(b) Suppose f is undefined for certain values, but the stated condition holds whenever both $f(x)$ and $f(y)$ are defined. Show that there is a monotone self-dual Boolean function g for which $g(x) = f(x)$ whenever $f(x)$ is defined.

▸ **65.** [*M21*] Any subset X of $\{1,2,\ldots,n\}$ corresponds to a binary vector $x = x_1x_2\ldots x_n$ via the rule $x_j = [j \in X]$. And any family $\mathcal{F}$ of such subsets corresponds to a Boolean function $f(x) = f(x_1,x_2,\ldots,x_n)$ of n variables, via the rule $f(x) = [X \in \mathcal{F}]$. Therefore

renamed Horn function
quantified formula
existential quantifiers
universal quantifier
Aspvall
Plass
Tarjan
Krom clauses
Horn
Pehoushek
Fraer
distributive law
Schensted
monotone Boolean function
median expansion formula
medians of five
Akers
families of subsets

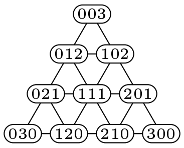
every statement about families of subsets corresponds to a statement about Boolean functions, and vice versa.

A family $\mathcal{F}$ is called *intersecting* if $X \cap Y \neq \emptyset$ whenever $X, Y \in \mathcal{F}$. An intersecting family that loses this property whenever we try to add another subset is said to be *maximal*. Prove that $\mathcal{F}$ is a maximal intersecting family if and only if the corresponding Boolean function f is monotone and self-dual.

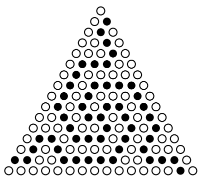
▶ **66.** [*M25*] A *coterie* of $\{1, \ldots, n\}$ is a family $\mathcal{C}$ of subsets called *quorums*, which have the following properties whenever $Q \in \mathcal{C}$ and $Q' \in \mathcal{C}$: (i) $Q \cap Q' \neq \emptyset$; (ii) $Q \subseteq Q'$ implies $Q = Q'$. Coterie $\mathcal{C}$ *dominates* coterie $\mathcal{C}'$ if $\mathcal{C} \neq \mathcal{C}'$ and if, for every $Q' \in \mathcal{C}'$, there is a $Q \in \mathcal{C}$ with $Q \subseteq Q'$. For example, the coterie $\{\{1,2\},\{2,3\}\}$ is dominated by $\{\{1,2\},\{1,3\},\{2,3\}\}$ and also by $\{\{2\}\}$. [Coteries were introduced in classic papers by L. Lamport, *CACM* **21** (1978), 558–565; H. Garcia-Molina and D. Barbara, *JACM* **32** (1985), 841–860. They have numerous applications to distributed system protocols, including mutual exclusion, data replication, and name servers. In these applications $\mathcal{C}$ is preferred to any coterie that it dominates.]

Prove that $\mathcal{C}$ is a nondominated coterie if and only if its quorums are the index sets of variables in the prime implicants of a monotone self-dual Boolean function $f(x_1, \ldots, x_n)$. (Thus Table 2 illustrates the nondominated coteries on $\{1,2,3,4\}$.)

▶ **67.** [*M30*] (J. W. Milnor and C. Schensted.) A triangular grid of order n, illustrated here for $n = 3$, contains $(n+2)(n+1)/2$ points with nonnegative "barycentric coordinates" xyz, where $x+y+z = n$. Two points are adjacent if they differ by ± 1 in exactly two coordinate positions. A point is said to lie on the x side if its x coordinate is zero, on the y side if its y coordinate is zero, or on the z side if its z coordinate is zero; thus each side contains $n+1$ points. If $n > 0$, a point lies on two different sides if and only if it occupies one of the three corner positions.



A "Y" is a connected set of points with at least one point on each side. Suppose each vertex of a triangular grid is covered with a white stone or a black stone. For example, the 52 black stones in



contain a (somewhat distorted) Y; but if any of them is changed from black to white, there is a white Y instead. A moment's thought makes it intuitively clear that, in any placement, the black stones contain a Y if and only if the white stones do not.

We can represent the color of each stone by a Boolean variable, with 0 for white and 1 for black. Let $Y(t) = 1$ if and only if there's a black Y, where t is a triangular grid comprising all the Boolean variables. This function Y is clearly monotone; and the intuitive claim made in the preceding paragraph is equivalent to saying that Y is also self-dual. The purpose of this exercise is to prove the claim rigorously, using median algebra.

Given $a, b, c \geq 0$, let t_{abc} be the triangular subgrid containing all points whose coordinates xyz satisfy $x \geq a$, $y \geq b$, $z \geq c$. For example, t_{001} denotes all points except those on the z side (the bottom row). Notice that, if $a + b + c = n$, t_{abc} is the single point with coordinates abc; and in general, t_{abc} is a triangular grid of order $n-a-b-c$.

intersecting
intersecting families of sets
maximal intersecting families
monotone self-dual functions
coterie
quorums
dominates
Lamport
Garcia-Molina
Barbara
distributed system
mutual exclusion
data replication
name servers
Milnor
Schensted
triangular grid
barycentric coordinates
Y
stone representation

a) Prove that, if $n > 0$, $Y(t) = \langle Y(t_{100})Y(t_{010})Y(t_{001})\rangle$.

b) If $n > 0$, let t^* be the triangular grid of order $n-1$ defined by the rule

$$t^*_{xyz} = \langle t_{(x+1)yz}t_{x(y+1)z}t_{xy(z+1)}\rangle, \qquad \text{for } x+y+z = n-1.$$

Prove that $Y(t) = Y(t^*)$. [In other words, t^* condenses each small triangle of stones by taking the median of their colors. Repeating this process defines a *pyramid* of stones, with the top stone black if and only if there is a black Y at the bottom. It's fun to apply this condensation principle to the twisted Y above.]

68. [*46*] The just-barely-Y configuration shown in the previous exercise has 52 black stones. What is the largest number of black stones possible in such a configuration? (That is, how many variables can there be in a prime implicant of the function $Y(t)$?)

▶ **69.** [*M26*] (C. Schensted.) Exercise 67 expresses the Y function in terms of medians. Conversely, let $f(x_1,\ldots,x_n)$ be any monotone self-dual Boolean function with $m+1$ prime implicants p_0, p_1, …, p_m. Prove that $f(x_1,\ldots,x_n) = Y(T)$, where T is any triangular grid of order $m-1$ in which T_{abc} is a variable common to p_a and p_{a+b+1}, for $a+b+c = m-1$. For example, when $f(w,x,y,z) = \langle xwywz\rangle$ we have $m = 3$ and

$$f(w,x,y,z) = (w\wedge x)\vee(w\wedge y)\vee(w\wedge z)\vee(x\wedge y\wedge z) = Y\Bigl(\begin{smallmatrix} & w & \\ w & & w \\ x \; & y & \; z\end{smallmatrix}\Bigr).$$

▶ **70.** [*M20*] (A. Meyerowitz, 1989.) Given any monotone self-dual Boolean function $f(x) = f(x_1,\ldots,x_n)$, choose any prime implicant $x_{j_1}\wedge\cdots\wedge x_{j_s}$ and let

$$g(x) = \bigl(f(x)\wedge[x\neq t]\bigr)\vee[x=\bar t],$$

where $t = t_1\ldots t_n$ is the bit vector that has 1s in positions $\{j_1,\ldots,j_s\}$. Prove that $g(x)$ is also monotone and self-dual. (Notice that $g(x)$ is equal to $f(x)$ except at the two points t and $\bar t$.)

▶ **71.** [*M21*] Given the axioms (50), (51), and (52) of a median algebra, prove that the long distributive law (54) is a consequence of the shorter law (53).

72. [*M22*] Derive (58), (59), and (60) from the median laws (50)–(53).

73. [*M32*] (S. P. Avann.) Given a median algebra M, whose intervals are defined by (57) and whose corresponding median graph is defined by (61), let $d(u,v)$ denote the distance from u to v. Also let '$[uxv]$' stand for the statement "x lies on a shortest path from u to v."

a) Prove that $[uxv]$ holds if and only if $d(u,v) = d(u,x)+d(x,v)$.

b) Suppose $x\in[u\,..\,v]$ and $u\in[x\,..\,y]$, where $x\neq u$ and y — v is an edge of the graph. Show that x — u is also an edge.

c) If $x\in[u\,..\,v]$, prove $[uxv]$, by induction on $d(u,v)$.

d) Conversely, prove that $[uxv]$ implies $x\in[u\,..\,v]$.

74. [*M21*] In a median algebra, show that $w = \langle xyz\rangle$ whenever we have $w\in[x\,..\,y]$, $w\in[x\,..\,z]$, and $w\in[y\,..\,z]$ according to definition (57).

▶ **75.** [*M36*] (M. Sholander, 1954.) Suppose M is a set of points with a betweenness relation "x lies between u and v," symbolized by $[uxv]$, which satisfies the following three axioms:

i) If $[uvu]$ then $u = v$.

ii) If $[uxv]$ and $[xyu]$ then $[vyu]$.

iii) Given x, y, and z, exactly one point $w = \langle xyz\rangle$ satisfies $[xwy]$, $[xwz]$, and $[ywz]$.

The object of this exercise is to prove that M is a median algebra.

a) Prove the majority law $\langle xxy\rangle = x$, Eq. (50).

pyramid
condensation principle
prime implicant
Schensted
monotone self-dual Boolean function
Meyerowitz
median algebra
Avann
Sholander
betweenness
majority law

b) Prove the commutative law $\langle xyz\rangle = \langle xzy\rangle = \cdots = \langle zyx\rangle$, Eq. (51).
c) Prove that $[uxv]$ if and only if $x = \langle uxv\rangle$.
d) If $[uxy]$ and $[uyv]$, prove that $[xyv]$.
e) If $[uxv]$ and $[uyz]$ and $[vyz]$, prove that $[xyz]$. *Hint:* Construct the points $w = \langle yuv\rangle$, $p = \langle wux\rangle$, $q = \langle wvx\rangle$, $r = \langle pxz\rangle$, $s = \langle qxz\rangle$, and $t = \langle rsz\rangle$.
f) Finally, deduce the short distributive law, Eq. (53): $\langle\langle xyz\rangle uv\rangle = \langle x\langle yuv\rangle\langle zuv\rangle\rangle$.

76. [*M33*] Derive the betweenness axioms (i), (ii), and (iii) of exercise 75, starting from the three median axioms (50), (51), and (52), letting $[uxv]$ be an abbreviation for "$x = \langle uxv\rangle$." Do not use the distributive law (53). *Hint:* See exercise 74.

77. [*M28*] Let G be a median graph containing the edge r — s. For each edge u — v, call u an *early neighbor* of v if and only if r is closer to u than to v. Partition the vertices into "left" and "right" parts, where left vertices are closer to r than to s and right vertices are closer to s than to r. Each right vertex v has a *rank*, which is the shortest distance from v to a left vertex. Similarly, each left vertex u has rank $1 - d$, where d is the shortest distance from u to a right vertex. Thus u has rank zero if it is adjacent to a right vertex, otherwise its rank is negative. Vertex r clearly has rank 0, and s has rank 1.
a) Show that every vertex of rank 1 is adjacent to exactly one vertex of rank 0.
b) Show that the set of all right vertices is convex.
c) Show that the set of all vertices with rank 1 is convex.
d) Prove that steps I3–I9 of Subroutine I correctly mark all vertices of ranks 1 and 2.
e) Prove that Algorithm H is correct.

▶ **78.** [*M26*] If the vertex v is examined k times in step I4 during the execution of Algorithm H, prove that the graph has at least 2^k vertices. *Hint:* There are k ways to start a shortest path from v to a; thus at least k 1s appear in $l(v)$.

▶ **79.** [*M27*] (R. L. Graham.) A *subgraph of a hypercube* is a graph whose vertices v can be labeled with bit strings $l(v)$ in such a way that u — v if and only if $l(u)$ and $l(v)$ differ in exactly one bit position. (Each label has the same length.)
a) One way to define an n-vertex subgraph of a hypercube is to let $l(v)$ be the binary representation of v, for $0 \le v < n$. Show that this subgraph has exactly $f(n) = \sum_{k=0}^{n-1} \nu(k)$ edges, where $\nu(k)$ is the sideways addition function.
b) Prove that $f(n) \le n\lceil \lg n\rceil/2$.
c) Prove that no n-vertex subgraph of a hypercube has more than $f(n)$ edges.

80. [*27*] A *partial cube* is an "isometric" subgraph of a hypercube, namely a subgraph in which the distances between vertices are the same as they are in the full graph. The vertices of a partial cube can therefore be labeled in such a way that the distance from u to v is the "Hamming distance" between $l(u)$ and $l(v)$, namely $\nu(l(u) \oplus l(v))$. Algorithm H shows that every median graph is a partial cube.
a) Find a subgraph of the 4-cube that isn't a partial cube.
b) Give an example of a partial cube that isn't a median graph.

81. [*16*] Is every median graph bipartite?

82. [*25*] (*Incremental changes in service.*) Given a sequence of vertices $(v_0, v_1, \ldots, v_t)$ in a graph G, consider the problem of finding another sequence $(u_0, u_1, \ldots, u_t)$ for which $u_0 = v_0$ and the sum

$$\bigl(d(u_0,u_1) + d(u_1,u_2) + \cdots + d(u_{t-1},u_t)\bigr) \;+\; \bigl(d(u_1,v_1) + d(u_2,v_2) + \cdots + d(u_t,v_t)\bigr)$$

commutative law
distributive law
early neighbor
convex
Graham
subgraph of a hypercube
sideways addition function
partial cube
Isometric: Distance-preserving
Hamming distance
Incremental changes

is minimized, where $d(u,v)$ denotes the distance from u to v. (Each v_k can be regarded as a request for a resource needed at that vertex; a server moves to u_k as those requests are handled in sequence.) Prove that if G is a median graph, we get an optimum solution by choosing $u_k = \langle u_{k-1}v_kv_{k+1}\rangle$ for $0 < k < t$, and $u_t = v_t$.

▸ **83.** [*28*] Generalizing exercise 82, find an efficient way to minimize

$$\big(d(u_0,u_1)+d(u_1,u_2)+\cdots+d(u_{t-1},u_t)\big) \;+\; \rho\big(d(u_1,v_1)+d(u_2,v_2)+\cdots+d(u_t,v_t)\big)$$

in a median graph, given any positive ratio ρ.

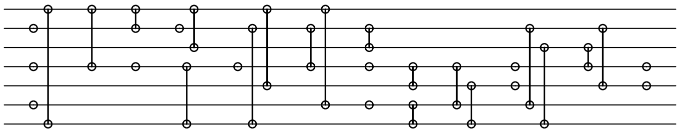
84. [*30*] Write a program to find all monotone self-dual Boolean functions of five variables. What are the edges of the corresponding median graph? (Table 2 illustrates the four-variable case.)

▸ **85.** [*M22*] Theorem S tells us that every formula in 2CNF corresponds to a median set; therefore every antisymmetric digraph such as Fig. 6 also corresponds to a median set. Precisely which of those digraphs correspond to *reduced* median sets?

86. [*15*] If v, w, x, y, and z belong to a median set X, does their five-element median $\langle vwxyz\rangle$, computed componentwise, always belong to X?

87. [*24*] What CI-net does the proof of Theorem F construct for the free tree (63)?

88. [*M21*] We can use parallel computation to condense the network (74) into



by letting each module act at the earliest possible time. Prove that, although the network constructed in the proof of Theorem F may contain $\Omega(t^2)$ modules, it always requires at most $O(t\log t)$ levels of delay.

89. [*24*] When the construction (73) appends a new cluster of modules to enforce the condition $u \to v$, for some literals u and v, prove that it preserves all previously enforced conditions $u' \to v'$.

▸ **90.** [*21*] Construct a CI-net with input bits $x_1 \ldots x_t$ and output bits $y_1 \ldots y_t$, where $y_1 = \cdots = y_{t-1} = 0$ and $y_t = x_1 \oplus \cdots \oplus x_t$. Try for only $O(\log t)$ levels of delay.

91. [*46*] Can a retraction mapping for the labels of every median graph of dimension t be computed by a CI-net that has only $O(\log t)$ levels of delay? [This question is motivated by the existence of asymptotically optimum networks for the analogous problem of sorting; see M. Ajtai, J. Komlós, and E. Szemerédi, *Combinatorica* **3** (1983), 1–19.]

92. [*46*] Can a CI-net sort n Boolean inputs with fewer modules than a "pure" sorting network that has no inverters?

93. [*M20*] Prove that every retract X of a graph G is an isometric subgraph of G. (In other words, distances in X are the same as in G; see exercise 80.)

94. [*M21*] Prove that every retract X of a hypercube is a set of median labels, if we suppress coordinates that are constant for all $x \in X$.

95. [*M25*] True or false: The set of all outputs produced by a comparator-inverter network, when the inputs range over all possible bit strings, is always a median set.

96. [*HM25*] Instead of insisting that the constants $w_1, w_2, \ldots, w_n$, and t in (75) must be integers, we could allow them to be arbitrary real numbers. Would that increase the number of threshold functions?

server locations
free median algebra
median set
antisymmetric digraph
reduced
five-element median
CI-net
parallel computation
retraction mapping
Ajtai
Komlós
Szemerédi
sorting networks
sorting network
isometric subgraph
retract
real numbers

97. [*10*] What median/majority functions arise in (8₁) when $n = 2$, $w_1 = w_2 = 1$, and $t = -1$, 0, 1, 2, 3, or 4?

98. [*M23*] Prove that any self-dual threshold function can be expressed in the form

$$f(x_1, x_2, \ldots, x_n) = [v_1 y_1 + \cdots + v_n y_n > 0],$$

where each y_j is either x_j or $\bar{x}_j$. For example, $2x_1 + 3x_2 + 5x_3 + 7x_4 + 11x_5 + 13x_6 \ge 21$ if and only if $2x_1 + 3x_2 + 5x_3 - 7\bar{x}_4 + 11x_5 - 13\bar{x}_6 > 0$.

▶ **99.** [*20*] (J. E. Mezei, 1961.) Prove that

$$\langle\langle x_1 \ldots x_{2s-1}\rangle\, y_1 \ldots y_{2t-2}\rangle = \langle x_1 \ldots x_{2s-1}\, y_1^s \ldots y_{2t-2}^s\rangle.$$

100. [*20*] True or false: If $f(x_1, \ldots, x_n)$ is a threshold function, so are the functions $f(x_1, \ldots, x_n) \wedge x_{n+1}$ and $f(x_1, \ldots, x_n) \vee x_{n+1}$.

101. [*M23*] The *Fibonacci threshold function* $F_n(x_1, \ldots, x_n)$ is defined by the formula $\langle x_1^{F_1} x_2^{F_2} \ldots x_{n-1}^{F_{n-1}} x_n^{F_{n-2}}\rangle$ when $n \ge 3$; for example, $F_7(x_1, \ldots, x_7) = \langle x_1 x_2 x_3^2 x_4^3 x_5^5 x_6^8 x_7^5\rangle$.

a) What are the prime implicants of $F_n(x_1, \ldots, x_n)$?
b) Find an orthogonal DNF for $F_n(x_1, \ldots, x_n)$ (see exercise 35).
c) Express $F_n(x_1, \ldots, x_n)$ in terms of the Y function (see exercises 67 and 69).

102. [*M21*] The *self-dualization* of a Boolean function is defined by the formulas

$$\begin{aligned}\hat{f}(x_0, x_1, \ldots, x_n) &= (x_0 \wedge f(x_1, \ldots, x_n)) \vee (\bar{x}_0 \wedge \overline{f(\bar{x}_1, \ldots, \bar{x}_n)})\\ &= (\bar{x}_0 \vee f(x_1, \ldots, x_n)) \wedge (x_0 \vee \overline{f(\bar{x}_1, \ldots, \bar{x}_n)}).\end{aligned}$$

a) If $f(x_1, \ldots, x_n)$ is any Boolean function, prove that $\hat{f}$ is self-dual.
b) Prove that $\hat{f}$ is a threshold function if and only if f is a threshold function.

103. [*HM25*] Explain how to use linear programming to test whether or not a monotone, self-dual Boolean function is a threshold function, given a list of its prime implicants. Also, if it is a threshold function, explain how to minimize the size of its representation as a majority function $\langle x_1^{w_1} \ldots x_n^{w_n}\rangle$.

104. [*25*] Apply the method of exercise 103 to find the shortest representations of the following threshold functions as majority functions: (a) $\langle x_1^2 x_2^3 x_3^5 x_4^7 x_5^{11} x_6^{13} x_7^{17} x_8^{19}\rangle$; (b) $[(x_1 x_2 x_3 x_4)_2 \ge t]$, for $0 \le t \le 16$ (17 cases); (c) $\langle x_1^{29} x_2^{25} x_3^{19} x_4^{15} x_5^{12} x_6^8 x_7^8 x_8^3 x_9^3 x_{10}\rangle$.

105. [*M25*] Show that the Fibonacci threshold function in exercise 101 has no shorter representation as a majority function than the one used to define it.

▶ **106.** [*M25*] The median-of-three operation $\langle x\bar{y}\bar{z}\rangle$ is true if and only if $x \ge y + z$.

a) Generalizing, show that we can test the condition $(x_1 x_2 \ldots x_n)_2 \ge (y_1 y_2 \ldots y_n)_2 + z$ by performing a median of $2^{n+1} - 1$ Boolean variables.
b) Prove that no median of fewer than $2^{n+1} - 1$ will suffice for this problem.

107. [*17*] Calculate $N(f)$ and $\Sigma(f)$ for the 16 functions in Table 1. (See Theorem T.)

108. [*M21*] Let $g(x_0, x_1, \ldots, x_n)$ be a self-dual function; thus $N(g) = 2^n$ in the notation of Theorem T. Express $N(f)$ and $\Sigma(f)$ in terms of $\Sigma(g)$, when $f(x_1, \ldots, x_n)$ is (a) $g(0, x_1, \ldots, x_n)$; (b) $g(1, x_1, \ldots, x_n)$.

109. [*M25*] The binary string $\alpha = a_1 \ldots a_n$ is said to *majorize* the binary string $\beta = b_1 \ldots b_n$, written $\alpha \succeq \beta$ or $\beta \preceq \alpha$, if $a_1 + \cdots + a_k \ge b_1 + \cdots + b_k$ for all $k \ge 0$.

a) Let $\bar{\alpha} = \bar{a}_1 \ldots \bar{a}_n$. Show that $\alpha \succeq \beta$ if and only if $\bar{\beta} \succeq \bar{\alpha}$.

self-dual threshold function
Mezei
threshold functions of threshold functions
Fibonacci threshold function
prime implicants
orthogonal DNF
self-dualization
linear programming
Fibonacci threshold function
comparison of binary numbers
self-dual
majorization++
dominance order, see majorization
lattice+

greatest lower bound
least upper bound
distributive laws
covers
regular
optimum coterie
availability
binary majorization lattice

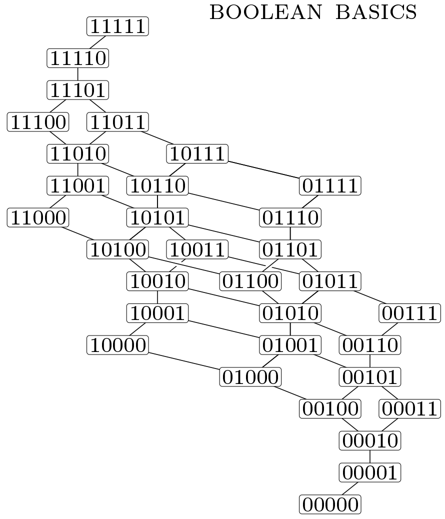


Fig. 8. The binary majorization lattice for strings of length 5. (See exercise 109.)

b) Show that any two binary strings of length n have a greatest lower bound $\alpha \wedge \beta$, which has the property that $\alpha \succeq \gamma$ and $\beta \succeq \gamma$ if and only if $\alpha \wedge \beta \succeq \gamma$. Explain how to compute $\alpha \wedge \beta$, given α and β.
c) Similarly, explain how to compute a least upper bound $\alpha \vee \beta$, with the property that $\gamma \succeq \alpha$ and $\gamma \succeq \beta$ if and only if $\gamma \succeq \alpha \vee \beta$.
d) True or false: $\alpha \wedge (\beta \vee \gamma) = (\alpha \wedge \beta) \vee (\alpha \wedge \gamma)$; $\alpha \vee (\beta \wedge \gamma) = (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$.
e) Say that α *covers* β if $\alpha \succeq \beta$ and $\alpha \neq \beta$, and if $\alpha \succeq \gamma \succeq \beta$ implies that we have either $\gamma = \alpha$ or $\gamma = \beta$. For example, Fig. 8 illustrates the covering relations between binary strings of length 5. Find a simple way to describe the strings that are covered by a given binary string.
f) Show that every path $\alpha = \alpha_0, \alpha_1, \ldots, \alpha_r = 0\ldots0$ from a given string α to $0\ldots0$, where α_{j-1} covers α_j for $1 \le j \le r$, has the same length $r = r(\alpha)$.
g) Let $m(\alpha)$ be the number of binary strings β with $\beta \succeq \alpha$. Prove that $m(1\alpha) = m(\alpha)$ and $m(0\alpha) = m(\alpha) + m(\alpha')$, where α' is α with its leftmost 1 (if any) changed to 0.
h) How many strings α of length n satisfy $\bar{\alpha} \succeq \alpha$?

110. [*M23*] A Boolean function is called *regular* if $x \preceq y$ implies that $f(x) \le f(y)$ for all vectors x and y, where $\preceq$ is the majorization relation in exercise 109. Prove or disprove the following statements:
a) Every regular function is monotone.
b) If f is a threshold function (75) for which $w_1 \ge w_2 \ge \cdots \ge w_n$, f is regular.
c) If f is as in (b) and $\Sigma(f) = (s_1, \ldots, s_n)$, then $s_1 \ge s_2 \ge \cdots \ge s_n$.
d) If f is a threshold function (75) with $s_1 \ge s_2 \ge \cdots \ge s_n$, then $w_1 \ge w_2 \ge \cdots \ge w_n$.

111. [*M36*] An *optimum coterie* for a system with working probabilities $(p_1, \ldots, p_n)$ is a coterie that corresponds to a monotone self-dual function with maximum availability, among all monotone self-dual functions with n variables. (See exercises 14 and 66.)
a) Prove that if $1 \ge p_1 \ge \cdots \ge p_n \ge \frac{1}{2}$, at least one self-dual function with maximum availability is a regular function. Describe such a function.
b) Furthermore, it suffices to test the optimality of a regular self-dual function f at points y of the binary majorization lattice for which $f(y) = 1$ but $f(x) = 0$ for all x covered by y.
c) What coterie is optimum when some probabilities are $< \frac{1}{2}$?

▶ **112.** [*M37*] (J. Håstad.) If $f(x_1, x_2, \ldots, x_m)$ is a Boolean function, let $M(f)$ be its representation as a multilinear polynomial with integer coefficients (see exercise 12). Arrange the terms in this polynomial by using Chase's sequence $\alpha_0 = 00\ldots0$, $\alpha_1 = 10\ldots0$, $\ldots$, $\alpha_{2^m-1} = 11\ldots1$ to order the exponents; Chase's sequence, obtained by concatenating the sequences A_{n0}, $A_{(n-1)1}$, $\ldots$, A_{0n} of 7.2.1.3–(35), has the nice property that α_j is identical to α_{j+1} except for a slight change, either $0 \to 1$ or $01 \to 10$ or $001 \to 100$ or $10 \to 01$ or $100 \to 001$. For example, Chase's sequence is

$$0000, 1000, 0010, 0001, 0100, 1100, 1010, 1001, 0011, 0101, 0110, 1110, 1101, 1011, 0111, 1111$$

when $m = 4$, corresponding to the respective terms 1, x_1, x_3, x_4, x_2, x_1x_2, $\ldots$, $x_2x_3x_4$, $x_1x_2x_3x_4$; so the relevant representation of, say, $((x_1 \oplus \bar{x}_2) \wedge x_3) \vee (x_1 \wedge \bar{x}_3 \wedge x_4)$ is

$$x_3 - x_1x_3 + x_1x_4 - x_2x_3 + 2x_1x_2x_3 - x_1x_3x_4$$

when the terms have been arranged in this order. Now let

$$F(f) \;=\; [\text{the most significant coefficient of } M(f) \text{ is positive}].$$

For example, the most significant (final) nonzero term of $((x_1 \oplus \bar{x}_2) \wedge x_3) \vee (x_1 \wedge \bar{x}_3 \wedge x_4)$ is $-x_1x_3x_4$ in Chase's ordering, so $F(f) = 0$ in this case.

a) Determine $F(f)$ for each of the 16 functions in Table 1.

b) Show that $F(f)$ is a threshold function of the $n = 2^m$ entries $\{f_{0\ldots00}, f_{0\ldots01}, \ldots, f_{1\ldots11}\}$ of the truth table for f. Write this function out explicitly when $m = 2$.

c) Prove that, when m is large, all the weights in any threshold representation of F must be huge: Their absolute values must all exceed

$$\frac{3^{\binom{m}{3}}\,7^{\binom{m}{4}}\,15^{\binom{m}{5}}\ldots(2^{m-1}-1)^{\binom{m}{m}}}{n}(1-O(n^{-1})) \;=\; 2^{mn/2-n-2(3/2)^m/\ln 2+O((5/4)^m)}.$$

Hint: Consider discrete Fourier transforms of the truth table entries.

113. [*24*] Show that the following three threshold operations suffice to evaluate the function $S_{2,3,6,8,9}(x_1, \ldots, x_{12})$ in (91):

$$g_1(x_1, \ldots, x_{12}) = [\nu x \geq 6] = \langle 1x_1 \ldots x_{12}\rangle;$$
$$g_2(x_1, \ldots, x_{12}) = [\nu x - 6g_1 \geq 2] = \langle 1^3 x_1 \ldots x_{12}\bar{g}_1^6\rangle;$$
$$g_3(x_1, \ldots, x_{12}) = [-2\nu x + 13g_1 + 7g_2 \geq 1] = \langle 0^5 \bar{x}_1^2 \ldots \bar{x}_{12}^2 g_1^{13} g_2^7\rangle.$$

Also find a four-threshold scheme that evaluates $S_{1,3,5,8}(x_1, \ldots, x_{12})$.

114. [*20*] (D. A. Huffman.) What is the function $S_{3,6}(x, x, x, x, y, y, z)$?

115. [*M22*] Explain why (92) correctly computes the parity function $x_0 \oplus x_1 \oplus \cdots \oplus x_{2m}$.

▶ **116.** [*HM28*] (B. Dunham and R. Fridshal, 1957.) By considering symmetric functions, one can prove that Boolean functions of n variables might have many prime implicants.

a) Suppose $0 \leq j \leq k \leq n$. For which symmetric functions $f(x_1, \ldots, x_n)$ is the term $x_1 \wedge \cdots \wedge x_j \wedge \bar{x}_{j+1} \wedge \cdots \wedge \bar{x}_k$ a prime implicant?

b) How many prime implicants does the function $S_{3,4,5,6}(x_1, \ldots, x_9)$ have?

c) Let $\hat{b}(n)$ be the maximum number of prime implicants, over all symmetric Boolean functions of n variables. Find a recurrence formula for $\hat{b}(n)$, and compute $\hat{b}(9)$.

d) Prove that $\hat{b}(n) = \Theta(3^n/n)$.

e) Show that, furthermore, there are symmetric functions $f(x_1, \ldots, x_n)$ for which both f and $\bar{f}$ have $\Theta(2^{3n/2}/n)$ prime implicants.

117. [*M26*] A disjunctive normal form is called *irredundant* if none of its implicants implies another. Let $b^*(n)$ be the maximum number of implicants in an irredundant

Håstad
Chase's sequence
integer multilinear rep
truth table
discrete Fourier transforms
Huffman
parity function
Dunham
Fridshal
symmetric functions
prime implicant
irredundant

DNF, over all Boolean functions of n variables. Find a simple formula for $b^*(n)$, and determine its asymptotic value.

118. [*29*] How many Boolean functions $f(x_1, x_2, x_3, x_4)$ have exactly m prime implicants, for $m = 0, 1, \ldots$?

119. [*M48*] Continuing the previous exercises, let $b(n)$ be the maximum number of prime implicants in a Boolean function of n variables. Clearly $\hat{b}(n) \le b(n) < b^*(n)$; what is the asymptotic value of $b(n)$?

120. [*23*] What is the shortest DNF for the symmetric functions (a) $x_1 \oplus x_2 \oplus \cdots \oplus x_n$? (b) $S_{0,1,3,4,6,7}(x_1, \ldots, x_7)$? (c) Prove that every Boolean function of n variables can be expressed as a DNF with at most 2^{n-1} prime implicants.

▸ **121.** [*M23*] The function $\langle 1(x_1 \oplus x_2) y_1 y_2 y_3 \rangle$ is partially symmetric, since it is symmetric in $\{x_1, x_2\}$ and in $\{y_1, y_2, y_3\}$, but not in all five variables $\{x_1, x_2, y_1, y_2, y_3\}$.

a) Exactly how many Boolean functions $f(x_1, \ldots, x_m, y_1, \ldots, y_n)$ are symmetric in $\{x_1, \ldots, x_m\}$ and $\{y_1, \ldots, y_n\}$?
b) How many of those functions are monotone?
c) How many of those functions are self-dual?
d) How many of those functions are monotone and self-dual?

122. [*M25*] Continuing exercises 110 and 121, find all Boolean functions $f(x_1, x_2, x_3, y_1, y_2, y_3, y_4, y_5, y_6)$ that are simultaneously symmetric in $\{x_1, x_2, x_3\}$, symmetric in $\{y_1, y_2, \ldots, y_6\}$, self-dual, and regular. Which of them are threshold functions?

123. [*46*] Determine the exact number of self-dual Boolean functions of ten variables that are threshold functions.

124. [*20*] Find a Boolean function of four variables that is equivalent to 767 other functions, under the ground rules of Table 5.

125. [*18*] Which of the function classes in (95) are canalizing?

126. [*23*] (a) Show that a Boolean function is canalizing if and only if its sets of prime implicants and prime clauses have a certain simple property. (b) Show that a Boolean function is canalizing if and only if its Chow parameters $N(f)$ and $\Sigma(f)$ have a certain simple property (see Theorem T). (c) Define the Boolean vectors

$$\vee(f) = \bigvee\{x \mid f(x) = 1\} \qquad \text{and} \qquad \wedge(f) = \bigwedge\{x \mid f(x) = 1\};$$

by analogy with the integer vector $\Sigma(f)$. Show that it's possible to decide whether or not f is canalizing, given only the four vectors $\vee(f)$, $\vee(\bar{f})$, $\wedge(f)$, and $\wedge(\bar{f})$.

127. [*M25*] Which canalizing functions are (a) self-dual? (b) definite Horn functions?

▸ **128.** [*20*] Find a noncanalizing $f(x_1, \ldots, x_n)$ that is true at exactly two points.

129. [*M25*] How many different canalizing functions of n variables exist?

130. [*M21*] According to Table 3, there are 168 monotone Boolean functions of four variables. But some of them, like $x \wedge y$, depend on only three variables or fewer.

a) How many 4-variable monotone Boolean functions actually involve each variable?
b) How many of those functions are distinct under permutation, as in Table 4?

131. [*HM42*] Table 3 makes it clear that there are many more Horn functions than Krom functions. What is the asymptotic number, as $n \to \infty$?

▸ **132.** [*HM30*] The Boolean function $g(x) = g(x_1, \ldots, x_n)$ is called *affine* if it can be written in the form $y_0 \oplus (x_1 \wedge y_1) \oplus \cdots \oplus (x_n \wedge y_n) = (y_0 + x \cdot y) \bmod 2$ for some Boolean constants $y_0, y_1, \ldots, y_n$.

a) Given any Boolean function $f(x)$, show that some affine function agrees with $f(x)$ at $2^{n-1}+2^{n/2-1}$ or more points x. *Hint:* Let $s(y)=\sum_x(-1)^{f(x)+x\cdot y}$, and prove that $\sum_y s(y)s(y\oplus z)=2^{2n}[z=0\ldots0]$ for all vectors z.

b) The Boolean function $f(x)$ is called *bent* if no affine function agrees with it at more than $2^{n-1}+2^{n/2-1}$ points. Prove that

$$(x_1\wedge x_2)\oplus(x_3\wedge x_4)\oplus\cdots\oplus(x_{n-1}\wedge x_n)\oplus h(x_2,x_4,\ldots,x_n)$$

is a bent function, when n is even and $h(y_1,y_2,\ldots,y_{n/2})$ is arbitrary.

c) Prove that $f(x)$ is a bent function if and only if

$$\sum_x\bigl(f(x)\oplus f(x\oplus y)\bigr)=2^{n-1}\qquad\text{for all } y\neq 0\ldots0.$$

d) If a bent function $f(x_1,\ldots,x_n)$ is represented by a multilinear polynomial mod 2 as in (19), show that it never contains the term $x_1\ldots x_r$ when $r>n/2>1$.

▸ **133.** [*20*] (Mark A. Smith, 1990.) Suppose we flip n independent coins to get n random bits, where the kth coin produces bit 1 with probability p_k. Find a way to choose $(p_1,\ldots,p_n)$ so that $f(x_1,\ldots,x_n)=1$ with probability $(t_0t_1\ldots t_{2^n-1})_2/(2^{2^n}-1)$, where $t_0t_1\ldots t_{2^n-1}$ is the truth table of the Boolean function f. (Thus, n suitable random coins can generate a probability with 2^n-bit precision.)

bent
multilinear polynomial
Smith
coins
truth table
random number generation

SECTION 7.1.1

1. (Solution by C. Sartena.) He was describing the implication $x \Rightarrow y$, with "it" standing respectively for y, x, x, y, y, x. (Other solutions are possible.)

2. The Earth operation corresponding to the Pincusian $x \circ y$ is $\overline{\bar{x} \circ \bar{y}}$; its truth table is therefore the reverse of the complement of the truth table for $\circ$. Hence the respective answers are $\top$, $\vee$, $\subset$, L, $\supset$, R, $\equiv$, $\wedge$, $\barwedge$, $\oplus$, $\bar{\mathsf{R}}$, $\bar\supset$, $\bar{\mathsf{L}}$, $\bar\subset$, $\bar\vee$, $\perp$. (Any identity involving the 16 operations of Table 1 implies a corresponding dual identity obtained by substituting the Pincusian equivalents. For example, each of De Morgan's laws (11) and (12) is the dual of the other, as are the identities (3), (4) relating $\equiv$ and $\oplus$. In this sense $\equiv$ can be considered to be just as useful as its dual, $\oplus$.)

3. (a) $\vee$; (b) $\wedge$; (c) $\bar{\mathsf{L}}$; (d) $\equiv$. [Many formulas actually work out better if we use -1 for truth and $+1$ for falsehood, even though this convention seems a bit immoral; then $x \cdot y$ corresponds to $\oplus$. Notice that $\langle xyz \rangle = \mathrm{sign}(x + y + z)$, with either convention.]

4. [*Trans. Amer. Math. Soc.* **14** (1913), 481–488.] (a) Start with the truth tables for L and R; then compute truth table $\alpha \barwedge \beta$ bitwise from each known pair of truth tables α and β, generating the results in order of the length of each formula and writing down a shortest formula that leads to each new 4-bit table:

$\perp$: $(x \barwedge (x \barwedge x)) \barwedge (x \barwedge (x \barwedge x))$	$\bar\vee$: $(x \barwedge (x \barwedge x)) \barwedge ((y \barwedge y) \barwedge (x \barwedge x))$
$\wedge$: $(x \barwedge y) \barwedge (x \barwedge y)$	$\equiv$: $(x \barwedge y) \barwedge ((y \barwedge y) \barwedge (x \barwedge x))$
$\bar\supset$: $(x \barwedge (x \barwedge y)) \barwedge (x \barwedge (x \barwedge y))$	$\bar{\mathsf{R}}$: $y \barwedge y$
L: x	$\subset$: $y \barwedge (x \barwedge x)$
$\bar\subset$: $(y \barwedge (x \barwedge x)) \barwedge (y \barwedge (x \barwedge x))$	$\bar{\mathsf{L}}$: $x \barwedge x$
R: y	$\supset$: $x \barwedge (x \barwedge y)$
$\oplus$: $(y \barwedge (x \barwedge x)) \barwedge (x \barwedge (x \barwedge y))$	$\barwedge$: $x \barwedge y$
$\vee$: $(y \barwedge y) \barwedge (x \barwedge x)$	$\top$: $x \barwedge (x \barwedge x)$

(b) In this case we start with four tables $\perp$, $\top$, L, R, and we prefer formulas with fewer occurrences of variables whenever there's a choice between formulas of a given length:

$\perp$: 0	$\bar\vee$: $1 \barwedge ((y \barwedge 1) \barwedge (x \barwedge 1))$
$\wedge$: $(x \barwedge y) \barwedge 1$	$\equiv$: $(x \barwedge y) \barwedge ((y \barwedge 1) \barwedge (x \barwedge 1))$
$\bar\supset$: $((y \barwedge 1) \barwedge x) \barwedge 1$	$\bar{\mathsf{R}}$: $y \barwedge 1$
L: x	$\subset$: $y \barwedge (x \barwedge 1)$
$\bar\subset$: $(y \barwedge (x \barwedge 1)) \barwedge 1$	$\bar{\mathsf{L}}$: $x \barwedge 1$
R: y	$\supset$: $(y \barwedge 1) \barwedge x$
$\oplus$: $(y \barwedge (x \barwedge 1)) \barwedge ((y \barwedge 1) \barwedge x)$	$\barwedge$: $x \barwedge y$
$\vee$: $(y \barwedge 1) \barwedge (x \barwedge 1)$	$\top$: 1

5. (a) $\perp$: $x \mathbin{\bar\subset} x$; $\wedge$: $(x \mathbin{\bar\subset} y) \mathbin{\bar\subset} y$; $\bar\supset$: $y \mathbin{\bar\subset} x$; L: x; $\bar\subset$: $x \mathbin{\bar\subset} y$; R: y; the other 10 cannot be expressed. (b) With constants, however, all 16 are possible:

$\perp$: 0	$\bar\vee$: $y \mathbin{\bar\subset} (x \mathbin{\bar\subset} 1)$
$\wedge$: $(y \mathbin{\bar\subset} 1) \mathbin{\bar\subset} x$	$\equiv$: $(y \mathbin{\bar\subset} x) \mathbin{\bar\subset} ((x \mathbin{\bar\subset} y) \mathbin{\bar\subset} 1)$
$\bar\supset$: $y \mathbin{\bar\subset} x$	$\bar{\mathsf{R}}$: $y \mathbin{\bar\subset} 1$
L: x	$\subset$: $(x \mathbin{\bar\subset} y) \mathbin{\bar\subset} 1$
$\bar\subset$: $x \mathbin{\bar\subset} y$	$\bar{\mathsf{L}}$: $x \mathbin{\bar\subset} 1$
R: y	$\supset$: $(y \mathbin{\bar\subset} x) \mathbin{\bar\subset} 1$
$\oplus$: $((y \mathbin{\bar\subset} x) \mathbin{\bar\subset} ((x \mathbin{\bar\subset} y) \mathbin{\bar\subset} 1)) \mathbin{\bar\subset} 1$	$\barwedge$: $((y \mathbin{\bar\subset} 1) \mathbin{\bar\subset} x) \mathbin{\bar\subset} 1$
$\vee$: $(y \mathbin{\bar\subset} (x \mathbin{\bar\subset} 1)) \mathbin{\bar\subset} 1$	$\top$: 1

[B. A. Bernstein, *University of California Publications in Mathematics* **1** (1914), 87–96.]

Sartena
complement
truth table
dual identity
De Morgan's laws
equivalence
dynamic programming
bitwise
Bernstein

6. (a) $\bot$, $\wedge$, $\llcorner$, R, $\oplus$, $\vee$, $\equiv$, $\top$. (b) $\bot$, $\llcorner$, R, $\oplus$, $\equiv$, $\top$. [Notice that all of these operators are associative. In fact, the stated identity implies the associative law in general: First we have (i) $(x \circ y) \circ ((z \circ y) \circ w) = ((x \circ z) \circ (z \circ y)) \circ ((z \circ y) \circ w) = (x \circ z) \circ w$, and similarly (ii) $(x \circ (y \circ z)) \circ (y \circ w) = x \circ (z \circ w)$. Furthermore (iii) $(x \circ y) \circ (z \circ w) = (x \circ y) \circ ((z \circ y) \circ (y \circ w)) = (x \circ z) \circ (y \circ w)$ by (i). Thus $(x \circ z) \circ w = (x \circ z) \circ ((z \circ z) \circ w) = (x \circ (z \circ z)) \circ (z \circ w) = x \circ (z \circ w)$ by (i), (iii), (ii). The free system generated by $\{x_1, \ldots, x_n\}$ has exactly $n + 2^n n^2$ distinct elements, namely $\{x_j \mid 1 \le j \le n\}$ and $\{x_i \circ x_{j_1} \circ \cdots \circ x_{j_r} \circ x_k \mid r \ge 0 \text{ and } 1 \le i, k \le n \text{ and } 1 \le j_1 < \cdots < j_r \le n\}$.]

7. Equivalently, we want the identity $y \circ (x \circ y) = x$, which holds only for $\oplus$ and $\equiv$. [Jevons noticed this property of $\oplus$ in *Pure Logic* §151, but he did not pursue the matter. We will investigate general systems of this nature, called "gropes," in Section 7.2.3.]

8. $(\{\bot, \wedge, \bar{\subset}\}, S_0)$, $(\{\top, \vee, \supset\}, S_1)$, $(\{\llcorner, \bar{\llcorner}\}, S_0 \cap S_1)$, $(\{\oplus, \equiv, \bar{\mathsf{R}}\}, S_2)$, $(\{\bar{\supset}, \bar{\vee}\}, S_0 \cap S_2)$, $(\{\subset, \bar{\wedge}\}, S_1 \cap S_2)$, and (R, any), where $S_0 = \{\square \mid 0 \mathbin{\square} 0 = 0\}$, $S_1 = \{\square \mid 1 \mathbin{\square} 1 = 1\}$, and $S_2 = \{\square \mid \bar{x} \mathbin{\square} \bar{y} = \overline{x \mathbin{\square} y}\} = \{\llcorner, \mathsf{R}, \bar{\llcorner}, \bar{\mathsf{R}}\}$. Thus 92 of the 256 pairs are left-distributive. [This problem and those of exercise 6 were first treated by E. Schröder in §55 of his posthumously published *Vorlesungen über die Algebra der Logik* **2**, 2 (1905). He expressed the answer by saying in essence that the respective truth tables $(pqrs, wxyz)$ of $(\circ, \square)$ must satisfy the relation $((pq \vee rs) \wedge \bar{z}) \vee ((\bar{p}\bar{q} \vee \bar{r}\bar{s}) \wedge w) \vee ((p\bar{q} \vee r\bar{s}) \wedge ((w \equiv z) \vee (x \equiv y))) = 0$.]

9. (a) False; $(x \oplus y) \vee z = (x \vee z) \oplus (y \vee z) \oplus z$. (b) True, because the identity obviously holds when $z = 0$ and when $z = 1$. (c) True; it's also $(x \oplus y) \vee (x \oplus z) = 1 - [x = y = z]$.

10. The first stage of decomposition (16) yields the functions with truth tables $g = 10100011$ and $h = 10100011 \oplus 10010011 = 00110000$; and the process continues in a similar way, yielding $1 + y + xz + w + wy + wx + wxz$ (modulo 2).

11. The stated term is present if and only if $f(x_1, \ldots, x_n)$ is true an odd number of times when $x_1 = x_4 = x_5 = x_7 = x_9 = x_{10} = \cdots = 0$. (There are 2^k such cases when we set all but k variables to zero.) In other words the multilinear representation can be expressed in a suggestive notation like

$$f(x, y, z) = (f_{000} + f_{00*}z + f_{0*0}y + f_{0**}yz + f_{*00}x + f_{*0*}xz + f_{**0}xy + f_{***}xyz) \bmod 2$$

illustrated here for $n = 3$, where $f_{**0} = f(1,1,0) \oplus f(1,0,0) \oplus f(0,1,0) \oplus f(0,0,0)$, etc.

12. (a) Substitute $1 - w$ for $\bar{w}$, etc., in (23), getting $1 - y - xz + 2xyz - w + wy + wx + wxz - 2wxyz$. [Some authors have called this the "Zhegalkin polynomial"; but I. I. Zhegalkin himself always worked modulo 2. Other names in the literature are "availability polynomial," "reliability polynomial," "characteristic polynomial."]

(b) The corresponding coefficients for an arbitrary n-ary function can be as large as 2^{n-1} in absolute value (and this, by induction, is the maximum). For example, the integer multilinear representation of $x_1 \oplus \cdots \oplus x_n$ over the integers turns out to be $e_1 - 2e_2 + 4e_3 - \cdots + (-2)^{n-1}e_n$, where e_k is the kth elementary symmetric function of $\{x_1, \ldots, x_n\}$. The formula in the previous answer becomes

$$f(x, y, z) \;=\; f_{000} + f_{00*}z + f_{0*0}y + f_{0**}yz + f_{*00}x + f_{*0*}xz + f_{**0}xy + f_{***}xyz$$

over the integers, where we now have $f_{**0} = f(1,1,0) - f(1,0,0) - f(0,1,0) + f(0,0,0)$, etc. This expansion is a disguised form of the Hadamard transform, Eq. 4.6.4–(38).

(c,d) The polynomial is the sum of its minterms like $x_1(1 - x_2)(1 - x_3)x_4$. Each minterm is nonnegative for $0 \le x_1, \ldots, x_n \le 1$, and the sum of all minterms is 1.

(e) $\partial f / \partial x_j = h(x) - g(x)$, where $h(x) \ge g(x)$ by (d). (See exercise 21.)

13. In fact, F is precisely the integer multilinear representation (see exercise 12).

universal algebra
free system
Jevons
gropes
Schröder
notation
Zhegalkin
availability polynomial
reliability polynomial
characteristic polynomial
elementary symmetric function
Hadamard transform
discrete Fourier transform
minterms

14. Let $r_j = p_j/(1-p_j)$. We want $f(0,0,0)=0$ and $f(1,1,1)=1 \Leftrightarrow r_1r_2r_3 > 1$, $f(0,0,1)=0$ and $f(1,1,0)=1 \Leftrightarrow r_1r_2 > r_3$, $f(0,1,0)=0$ and $f(1,0,1)=1 \Leftrightarrow r_1r_3 > r_2$, $f(0,1,1)=0$ and $f(1,0,0)=1 \Leftrightarrow r_1 > r_2r_3$. So we get (a) $\langle x_1x_2x_3\rangle$; (b) x_1; (c) $\bar{x}_3$.

15. Exercise 1.2.6–10 tells us that $\binom{x}{k} \bmod 2 = [x \mathbin{\&} k = k]$. Hence, for example, $\binom{x}{11} \equiv x_4 \wedge x_2 \wedge x_1$ (modulo 2) when $x = (x_n \ldots x_1)_2$; and we can obtain every term in a multilinear representation like (19) in this way. Moreover, we needn't work mod 2, because the interpolating polynomial $\binom{x}{11}\binom{15-x}{4}$ represents $x_4 \wedge \bar{x}_3 \wedge x_2 \wedge x_1$ exactly.

16. Yes, or even by $+$, because different minterms can't be simultaneously true. (But we can't do that in ordinary disjunctive normal forms like (25). See exercise 35.)

17. The binary operation $\barwedge$ is not associative, so an expression like $x \barwedge y \barwedge z$ must be interpreted as a *ternary* operation. Quick's notation is fine if one understands NAND to be an n-ary operation, being careful to note that the NAND of a *single* variable x is $\bar{x}$.

18. If not, we could set $u_1 \leftarrow \cdots \leftarrow u_s \leftarrow 1$ and $v_1 \leftarrow \cdots \leftarrow v_t \leftarrow 0$, making f both true and false. (And if we consider applying the distributive law (2) repeatedly to a DNF until it becomes a CNF, we find that the converse is also true: The disjunction $v_1 \vee \cdots \vee v_t$ is implied by f if and only if it has a literal in common with every implicant of f, if and only if it has a literal in common with every prime implicant of f, if and only if it has a literal in common with every implicant of some DNF for f.)

19. The maximal subcubes contained in 0010, 0011, 0101, 0110, 1000, 1001, 1010, and 1011 are 0∗10, 0101, ∗01∗, and 10∗∗; so the answer is $(w\vee\bar{y}\vee z)\wedge(w\vee\bar{x}\vee y\vee\bar{z})\wedge(x\vee\bar{y})\wedge(\bar{w}\vee x)$. (This CNF is also shortest.)

20. True. The corresponding maximal subcube is contained in some maximal subcubes f' and g', and their intersection can't be larger. (This observation is due to Samson and Mills, whose paper is cited in answer 31 below.)

21. By Boole's law (20), we see that an n-ary function f is monotone if and only if its $(n-1)$-ary projections g and h are monotone and satisfy $g \le h$. Therefore

$$f = (g \wedge \bar{x}_n) \vee (h \wedge x_n) \;=\; (g \wedge \bar{x}_n) \vee (g \wedge x_n) \vee (h \wedge x_n) \;=\; g \vee (h \wedge x_n),$$

so we can do without complementation. The constants 0 and 1 disappear unless the function is identically constant. Conversely, any expression built up from $\wedge$ and $\vee$ is obviously monotone.

Note on terminology: Strictly speaking, we should say "monotone nondecreasing" instead of simply "monotone," if we want to preserve the language of classical mathematics, because a decreasing function of a real variable is also said to be monotonic. (See, for example, the "run test" in Section 3.3.2G.) But "nondecreasing" is quite a mouthful; so researchers who work extensively on Boolean functions have almost unanimously opted to assume that "monotone" automatically implies nondecreasing, in a Boolean context. Similarly, the mathematical term "positive function" normally refers to a function whose value exceeds zero; but authors who write about "positive Boolean functions" are referring to the functions that we are calling monotone. Since a monotone function is order-preserving, some authors have adopted the term *isotone*; but that word has already been coopted by physicists, chemists, and musicologists.

A Boolean function like $\bar{x} \vee y$, which becomes monotone if some subset of its variables is complemented, is called *unate*. Theorem Q obviously applies to unate functions.

22. Both g and $g \oplus h$ must be monotone, and $g(x) \wedge h(x) = 0$.

23. $x \wedge (v\vee y) \wedge (v\vee z) \wedge (w\vee z)$. (Corollary Q applies also to *conjunctive* prime forms of monotone functions. Therefore, to solve any problem of this kind, we need only

multilinear representation
interpolating polynomial
associative
ternary
distributive law
shortest
Samson
Mills
terminology
monotone
positive function
isotone
unate

apply the distributive law (2) until no $\wedge$ occurs within a $\vee$, then remove any clause that contains all the variables of another.)

24. By induction on k, the similar tree with $\vee$ at the root gives a function with $2^{2^{\lceil k/2\rceil}-1}$ prime implicants of length $2^{\lfloor k/2\rfloor}$, while the tree with $\wedge$ gives $4^{2^{\lfloor k/2\rfloor}-1}$ of length $2^{\lceil k/2\rceil}$. When $k=6$, for example, the $4^7=2^{14}$ prime implicants in the $\wedge$ case have the form

$$x_{(0t_0 0t_{00} 0t_{000})_2} \wedge x_{(0t_0 0t_{00} 1t_{001})_2} \wedge x_{(0t_0 1t_{01} 0t_{010})_2} \wedge x_{(0t_0 1t_{01} 1t_{011})_2}$$
$$\wedge\, x_{(1t_1 0t_{10} 0t_{100})_2} \wedge x_{(1t_1 0t_{10} 1t_{101})_2} \wedge x_{(1t_1 1t_{11} 0t_{110})_2} \wedge x_{(1t_1 1t_{11} 1t_{111})_2},$$

with the t's either 0 or 1. [For further information about such Boolean functions, see D. E. Knuth and R. W. Moore, *Artificial Intelligence* **6** (1975), 293–326; V. Gurvich and L. Khachiyan, *Discrete Mathematics* **169** (1997), 245–248.]

25. Let a_n be the answer. Then $a_2=a_3=2$, $a_4=3$, and $a_n=a_{n-2}+a_{n-3}$ for $n>4$, because the prime implicants when $n>4$ are either $p_{n-2}\wedge x_{n-1}$ or $p_{n-3}\wedge x_{n-2}\wedge x_n$ for some prime implicant p_k in the k-variable case. (These prime implicants correspond to minimal vertex covers of the path graph P_n. They are *shellable*, in the sense of exercise 35, when listed in lexicographic order. We have $a_n=(7P_n+10P_{n+1}+P_{n+2})/23$ when P_n is the Perrin number of exercise 7.1.4–15.)

26. (a) Let $x_j=[j\in J]$. Then $f(x)=0$ and $g(x)=1$. (This fact was exercise 18.)

(b) Suppose, for example, that $k\in J\in\mathcal{G}$ and $k\notin\bigcup_{I\in\mathcal{F}} I$, and assume that test (a) has been passed. Let $x_j=[j\in J \text{ and } j\neq k]$. Then $f(x)=1$; and $g(x)=0$, because every $J'\in\mathcal{G}$ with $J'\neq J$ contains an element $\notin J$.

(c) Again assume that condition (a) has been ruled out. If, say, $|J|>|\mathcal{F}|$, let $x_j=[j$ is the smallest element of $I\cap J$, for some $I\in\mathcal{F}]$. Then $f(x)=1$, $g(x)=0$.

(d) Now we assume that $\bigcup_{I\in\mathcal{F}} I=\bigcup_{J\in\mathcal{G}} J$. Each $I\in\mathcal{F}$ stands for $2^{n-|I|}$ vectors where $f(x)=0$; similarly, each $J\in\mathcal{G}$ stands for $2^{n-|J|}$ vectors where $g(x)=1$. If the sum s is less than 2^n, we can compute $s=s_0+s_1$, where s_0 counts the contributions to s when $x_n=0$. If $s_0<2^{n-1}$, set $x_n\leftarrow 0$; otherwise $s_1<2^{n-1}$, so we set $x_n\leftarrow 1$. Then we set $n\leftarrow n-1$; eventually all x_j are known, and $f(x)=1$, $g(x)=0$.

27. Let $m=\min(\{|I|\mid I\in\mathcal{F}\}\cup\{|J|\mid J\in\mathcal{G}\})$ be the length of the shortest prime clause or implicant. Then $N\cdot 2^{n-m}\geq\sum_{I\in\mathcal{F}}2^{n-|I|}+\sum_{J\in\mathcal{G}}2^{n-|J|}\geq 2^n$; so we have $m\leq\lg N$. If, say, $|I|=m$, some index k must appear in at least $1/m$ of the members $J\in\mathcal{G}$, because each J intersects I. This observation proves the hint.

Now let $A(0)=A(1)=1$ and $A(v)=1+A(v-1)+A(\lfloor\rho v\rfloor)$ for $v>1$. Then $A(|\mathcal{F}||\mathcal{G}|)$ is an upper bound on the number of recursive calls (the number of times X1 is performed). Letting $B(v)=A(v)+1$, we have $B(v)=B(v-1)+B(\lfloor\rho v\rfloor)$ for $v>1$, hence $B(v)\leq B(v-k)+kB(\lfloor\rho v\rfloor)$ for $v>k$. Taking $k=v-\lfloor\rho v\rfloor$ shows that $B(v)\leq((1-\rho)v+2)B(\lfloor\rho v\rfloor)$; hence $B(v)=O(((1-\rho)v+2)^t)$ when $\rho^t v\leq 1$, namely when $t\geq\ln v/\ln(1/\rho)=\Theta((\log v)(\log N))$. Consequently $A(|\mathcal{F}||\mathcal{G}|)\leq A(N^2/4)=N^{O(\log N)^2}$.

In practice the algorithm will run much faster than the pessimistic bounds just derived. Since the prime clauses of a function are the prime implicants of its dual, this problem is essentially the same as verifying that one given DNF is the dual of another. Moreover, if we start with $f(x)=0$ and repeatedly find minimal x's where $f(x)=g(\bar{x})=0$, we can "grow" f until we've obtained the dual of g.

The ideas presented here are due to M. L. Fredman and L. Khachiyan, *J. Algorithms* **21** (1996), 618–628, who also presented refinements that reduce the running time to $N^{O(\log N/\log\log N)}$. No polynomial-time algorithm is known; yet the problem is unlikely to be NP-complete, because we can solve it in less-than-exponential time.

distributive law
Knuth
Moore
Gurvich
Khachiyan
recurrences
minimal vertex covers
path graph
shellable
Perrin number
recurrences
dual
Fredman
Khachiyan
NP-complete
exponential time

irredundant
Cutler
Muroga
stack
Morreale

28. This result is obvious once understood, but the notations and terminology can make it confusing; so let's consider a concrete example: If, say, $y_1 = y_4 = y_6 = 1$ and the other y_k are zero, the function g is true if and only if the prime implicants p_1, p_4, and p_6 cover all the places where f is true. Thus we see that there is a one-to-one correspondence between every implicant of g and every DNF for f that contains only prime implicants p_j. In this correspondence, the prime implicants of g correspond to the "irredundant" DNFs in which no p_j can be left out.

Numerous refinements of this principle have been discussed by R. B. Cutler and S. Muroga, *IEEE Transactions* **C-36** (1987), 277–292.

29. **B1.** [Initialize.] Set $k \leftarrow k' \leftarrow 0$. (Similar methods are discussed in exercise 5–19.)

B2. [Find a zero.] Increase k zero or more times, until either $k = m$ (terminate) or $v_k \,\&\, 2^j = 0$.

B3. [Make $k' > k$.] If $k' \le k$, set $k' \leftarrow k + 1$.

B4. [Advance k'.] Increase k' zero or more times, until either $k' = m$ (terminate) or $v_{k'} \ge v_k + 2^j$.

B5. [Skip past a big mismatch.] If $v_k \oplus v_{k'} \ge 2^{j+1}$, set $k \leftarrow k'$ and return to B2.

B6. [Record a match.] If $v_{k'} = v_k + 2^j$, output (k, k').

B7. [Advance k.] Set $k \leftarrow k + 1$ and return to B2. ▌

(Steps B3 and B5 are optional, but recommended.)

30. The following algorithm keeps variable-length, sorted lists in a stack S whose size will never exceed $2m + n$. When the topmost entry of the stack is $S_t = s$, the topmost list is the ordered set $S_s < S_{s+1} < \cdots < S_{t-1}$. Tag bits are maintained in another stack T, having the same size as S (after the initialization step).

P1. [Initialize.] Set $T_k \leftarrow 0$ for $0 \le k < m$. Then for $0 \le j < n$, apply the j-buddy scan algorithm of exercise 29, and set $T_k \leftarrow T_k + 2^j$, $T_{k'} \leftarrow T_{k'} + 2^j$ for all pairs (k, k') found. Then set $s \leftarrow t \leftarrow 0$ and repeat the following operations until $s = m$: If $T_s = 0$, output the subcube $(0, v_s)$ and set $s \leftarrow s+1$; otherwise set $S_t \leftarrow v_s$, $T_t \leftarrow T_s$, $t \leftarrow t + 1$, $s \leftarrow s + 1$. Finally set $A \leftarrow 0$ and $S_t \leftarrow 0$.

P2. [Advance A.] (At this point stack S contains $\nu(A) + 1$ lists of subcubes. Namely, if $A = 2^{e_1} + \cdots + 2^{e_r}$ with $e_1 > \cdots > e_r \ge 0$, the stack contains the b-values of all subcubes $(a, b) \subseteq V$ whose a-values are respectively 0, 2^{e_1}, $2^{e_1} + 2^{e_2}$, …, A, except that subcubes whose tags are zero do not appear. All of these lists are nonempty, except possibly the last. We will now increase A to the next relevant value.) Set $j \leftarrow 0$. If $S_t = t$ (that is, if the topmost list is empty), increase j zero or more times until $j \ge n$ or $A \,\&\, 2^j \ne 0$. Then while $j < n$ and $A \,\&\, 2^j \ne 0$, set $t \leftarrow S_t - 1$, $A \leftarrow A - 2^j$, and $j \leftarrow j + 1$. Terminate the algorithm if $j \ge n$; otherwise set $A \leftarrow A + 2^j$.

P3. [Generate list A.] Set $r \leftarrow t$, $s \leftarrow S_t$, and apply the j-buddy scan algorithm of exercise 29 to the $r - s$ numbers $S_s < \cdots < S_{r-1}$. For all pairs (k, k') found, set $x \leftarrow (T_k \,\&\, T_{k'}) - 2^j$; and if $x = 0$, output the subcube (A, S_k), otherwise set $t \leftarrow t + 1$, $S_t \leftarrow S_k$, $T_t \leftarrow x$. Finally set $t \leftarrow t + 1$, $S_t \leftarrow r + 1$, and go back to step P2. ▌

This algorithm is based in part on ideas of Eugenio Morreale [*IEEE Trans.* **EC-16** (1967), 611–620; *Proc. ACM Nat. Conf.* **23** (1968), 355–365]. The running time is essentially proportional to mn (for step P1) plus the total number of subcubes contained in V. If $m \le 2^n(1 - \epsilon)$, and if V is chosen at random with size m, exercise 34 shows that the average total number of subcubes is at most $O(\log\log n/\log\log\log n)$ times the average number of maximal subcubes; hence the average running time in most cases will be nearly proportional to the average amount of output produced. On the other hand, exercises 32 and 116 show that the amount of output might be huge.

31. (a) Let $c = c_{n-1} \ldots c_0$, $c' = c'_{n-1} \ldots c'_0$, $c'' = c''_{n-1} \ldots c''_0$. There must be some j with $c_j \neq *$ and $c_j \neq c''_j$; otherwise $c'' \subseteq c$. Similarly there must be some k with $c'_k \neq *$ and $c'_k \neq c''_k$. If $j \neq k$, there would be a point $x_{n-1} \ldots x_0 \in c''$ that is in neither c nor c', because we could let $x_j = \bar{c}_j$ and $x_k = \bar{c}'_k$. Hence $j = k$, and the value of j is uniquely determined. Furthermore it's easy to see that $c'_j = \bar{c}_j$. And if $i \neq j$, we have either $c_i = *$ or $c_i = c''_i$, and either $c'_i = *$ or $c'_i = c''_i$.

(b) This statement is an obvious consequence of (a).

(c) First we prove that the parenthesized remark in step E2 is true whenever that step is encountered. It's clearly true when $j = 0$. Otherwise, let $c \subseteq V$ be a j-cube, and suppose $c = c_0 \cup c_1$ where c_0 and c_1 are $(j-1)$-cubes. On the preceding execution of step E2 we had $c_0 \subseteq c'_0 \in C$ and $c_1 \subseteq c'_1 \in C$ for some c'_0 and c'_1; hence either $c \subseteq c'_0 \sqcup c'_1$ or $c \subseteq c'_0$ or $c \subseteq c'_1$. In each case, c is now contained in some element of C.

Secondly, we prove that the outputs in step E3 are precisely the maximal j-cubes contained in V: Let $c \subseteq V$ be any k-cube. If c is maximal, then c will be in C when we reach step E3 with $j = k$, and it will be output. If c isn't maximal, it has a buddy $c' \subseteq V$, which is a k-cube contained in some subcube $c'' \in C$ when we reach E3. Since $c \not\subseteq c''$, the consensus $c \sqcup c''$ will be a $(j+1)$-cube of C', and c will not be output.

References: The notion of consensus was first defined by Archie Blake in his Ph.D. dissertation at the University of Chicago (1937); see *J. Symbolic Logic* **3** (1938), 93, 112–113. It was independently rediscovered by Edward W. Samson and Burton E. Mills [Air Force Cambridge Research Center Tech. Report 54-21 (Cambridge, Mass.: April 1954), 54 pp.] and by W. V. Quine [*AMM* **62** (1955), 627–631]. The operation is also sometimes called the *resolvent*, since J. A. Robinson used it in a more general form (but with respect to clauses rather than implicants) as the basis of his "resolution principle" for theorem proving [*JACM* **12** (1965), 23–41]. Algorithm E is due to Ann C. Ewing, J. Paul Roth, and Eric G. Wagner, *AIEE Transactions*, Part 1, **80** (1961), 450–458.

32. (a) Change the definition of $\sqcup$ in exercise 31 to the following associative and commutative operation on the four symbols $A = \{0, 1, *, \bullet\}$, for all $a \in A$ and $x \in \{0, 1\}$:

$$* \sqcup a = a \sqcup * = a, \qquad \bullet \sqcup a = a \sqcup \bullet = x \sqcup \bar{x} = \bullet, \qquad \text{and} \qquad x \sqcup x = x.$$

Also let $h(0) = 0$, $h(1) = 1$, $h(*) = *$, and $h(\bullet) = *$. Then $c = h(c_1 \sqcup \cdots \sqcup c_m)$, computed componentwise, is the generalized consensus if and only if this subcube is contained in $c_1 \cup \cdots \cup c_m$. [See P. Tison, *IEEE Transactions* **EC-16** (1967), 446–456.]

(b) For example, let $c_j = *^{j-1}1*^{m-j}1^{j-1}0*^{m-j}$. [The final component is superfluous. All solutions have been characterized by R. H. Sloan, B. Szörényi, and G. Turán, in *Electronic Colloquium on Computational Complexity* (2005), Report 23.]

(c) By (a), every prime implicant corresponds uniquely to the subset of implicants that it "meets." [A. K. Chandra and G. Markowsky, *Discrete Math.* **24** (1978), 7–11.]

(d) For example, $(y_1 \wedge \bar{x}_1) \vee (y_2 \wedge x_1 \wedge \bar{x}_2) \vee \cdots \vee (y_m \wedge x_1 \wedge \cdots \wedge x_{m-1} \wedge \bar{x}_m)$ as in (b). [J.-M. Laborde, *Discrete Math.* **32** (1980), 209–212.]

33. (a) $\binom{2^n - 2^{n-k}}{m - 2^{n-k}} / \binom{2^n}{m}$. (b) We must exclude the cases when $x_1 \wedge \cdots \wedge x_{j-1} \wedge \bar{x}_j \wedge x_{j+1} \wedge \cdots \wedge x_k$ is also an implicant. By the inclusion-exclusion principle, the answer is

$$\sum_l \binom{k}{l} (-1)^l \binom{2^n - (l+1)2^{n-k}}{m - (l+1)2^{n-k}} \Big/ \binom{2^n}{m};$$

it simplifies to $\binom{2^n - n - 1}{m-1} / \binom{2^n}{m}$ when $k = n$.

Blake
Samson
Mills
Quine
resolvent
Robinson
resolution principle
Ewing
Roth
Wagner
associative
commutative
Tison
Sloan
Szörényi
Turán
Chandra
Markowsky
Laborde
inclusion-exclusion

Mileto
Putzolu
irredundant
random Boolean functions
Weber

34. (a) We have $c(m,n)=\sum c_j(m,n)$, where $c_j(m,n)=2^{n-j}\binom{n}{j}\binom{2^n-2^j}{m-2^j}/\binom{2^n}{m}$ is the average number of implicants with $n-j$ literals (the average number of subcubes of dimension j in the terminology of exercise 30). Clearly $c_0(m,n)=m$, and

$$c_1(m,n)=\frac{nm(m-1)}{2(2^n-1)}\le\frac{mn}{2}\Big(\frac{m}{2^n}\Big)\le\frac{1}{2}m;$$

similarly $c_j(m,n)\le m/(2^j j!\,n^{2^j-1-j})$. Also $p(m,n)=\sum_j p_j(m,n)$, where we have

$$p_0(m,n)=2^n\binom{2^n-n-1}{m-1}\Big/\binom{2^n}{m}=m\frac{(2^n-n-1)^{\underline{m-1}}}{(2^n-1)^{\underline{m-1}}}\ge m\frac{(2^n-n-m)^{m-1}}{(2^n-m)^{m-1}}$$
$$\ge\ m\Big(1-\frac{n}{2^n-m}\Big)^m\ \ge\ m\Big(1-\frac{n}{2^n-2^n/n}\Big)^{2^n/n}\ =\ m\exp\Big(\frac{2^n}{n}\ln\Big(1-\frac{n^2}{2^n(n-1)}\Big)\Big).$$

(b) Notice that $t=\lfloor\lg\lg n-\lg\lg(2^n/m)+\lg(4/3)\rfloor\le\lg\lg n+O(1)$ is quite small. We will repeatedly use the fact that $\binom{2^n-j\cdot2^t}{m-j\cdot2^t}/\binom{2^n}{m}<\alpha_{mn}^j$, and indeed that

$$\binom{2^n-j\cdot2^t}{m-j\cdot2^t}\Big/\binom{2^n}{m}\ =\ \alpha_{mn}^j(1+O(j^2 2^{2t}/m))$$

is an extremely good approximation when j isn't too large. To establish the hint, note that $\sum_{j<t}c_j(m,n)/c_t(m,n)\ =\ O(tc_{t-1}(m,n)/c_t(m,n))\ =\ O(t^2/(n\sqrt{\alpha_{mn}}))\ =\ O((\log\log n)^2/n^{1/3})$; and $c_{t+j}(m,n)/c_t(m,n)=O((n/(2t))^j\alpha_{mn}^{2^j-1})$. Consequently we have $c(m,n)/c_t(m,n)\approx1+\frac{1}{2}\big(\frac{n-t}{t+1}\big)\alpha_{mn}$, where the second term dominates when α_{mn} is in the upper part of its range. Furthermore

$$\sum_l\binom{n-t}{l}(-1)^l\alpha_{mn}^l\Big(1+O\Big(\frac{l^2 2^{2t}}{m}\Big)\Big)\ =\ (1-\alpha_{mn})^{n-t}+O(n^2\alpha_{mn}(1+\alpha_{mn})^n 2^{2t}/m)$$

has an exponentially small error term, because $(1+\alpha_{mn})^n=O(e^{n^{1/3}})\ll m$. Therefore $p(m,n)/c_t(m,n)$ is asymptotically $e^{-n\alpha_{mn}}+\frac{1}{2}\big(\frac{n-t}{t+1}\big)\alpha_{mn}e^{-n\alpha_{mn}^2}$.

(c) Here $\alpha_{mn}=2^{-2^t}\approx n^{-1}\ln(t/\ln t)$; so $c(m,n)/c_t(m,n)=1+O(t^{-1}\log t)$, $p(m,n)/c_t(m,n)=t^{-1}\ln t+\frac{1}{2}t^{-1}\ln t+O(t^{-1}\log\log t)$. We conclude that, in this case,

$$\frac{c(m,n)}{p(m,n)}\ =\ \frac{2}{3}\frac{\lg\lg n}{\lg\lg\lg n}\Big(1+O\Big(\frac{\log\log\log\log n}{\log\log\log n}\Big)\Big).$$

(d) If $n\alpha_{mn}\le\ln t-\ln\ln t$, we have $p(m,n)/c(m,n)\ge p_t(m,n)/c(m,n)\ge t^{-1}\ln t+O(t^{-1}\log t)^2$. On the other hand if $n\alpha_{mn}\ge\ln t-\ln\ln t$, we have $p(m,n)/c(m,n)\ge p_{t+1}(m,n)/c(m,n)\ge\frac{1}{2}t^{-1}\ln t+O(t^{-1}\log\log t)$.

[The means $c(m,n)$ and $p(m,n)$, and the variance of $c(m,n)$, were first studied by F. Mileto and G. Putzolu, *IEEE Trans.* **EC-13** (1964), 87–92; *JACM* **12** (1965), 364–375. Detailed asymptotic information about implicants, prime implicants, and irredundant DNFs of random Boolean functions, when each value $f(x_1,\ldots,x_n)$ is independently equal to 1 with probability $p(n)$, has been derived by Karl Weber, *Elektronische Informationsverarbeitung und Kybernetik* **19** (1983), 365–374, 449–458, 529–534.]

35. (a) By rearranging coordinates we can assume that the pth subcube is $0^k1^u{*}^v$, so that $B_p=0^k1^u0^v$ and $S_p=1^k0^{u+v}$. Then all points of ${*}^k1^u{*}^v$ are still covered, by induction on p, because all points of ${*}^{j-1}1{*}^{k-j}1^u{*}^v$ have been covered for $1\le j\le k$.

(b) The jth and kth subcubes differ in every coordinate position where $B_j\ \&\ S_k$ is nonzero. On the other hand if $B_j\ \&\ S_k$ is zero, the point $\bar S_k$ of subcube k lies in a previous subcube, by (a), because we have $\bar S_k\supseteq B_j$.

(c) From the list 1100, $1\underline{0}11$, $\underline{0}011$ (with the bits of each S_k underlined) we obtain the orthogonal DNF $(x_1 \wedge x_2) \vee (x_1 \wedge \bar{x}_2 \wedge x_3 \wedge x_4) \vee (\bar{x}_1 \wedge x_3 \wedge x_4)$.

(d) There are eight solutions; for example, $(01100, 0\underline{0}110, 00\underline{0}11, 11\underline{0}1\underline{0}, 11\underline{00}0)$.

(e) $(001100, 011\underline{0}00, 00\underline{0}110, 11\underline{00}10, 11\underline{0}0\underline{0}0, \underline{0}1\underline{00}11, 0\underline{0}0\underline{0}11)$ is a symmetrical solution. And there are many more possibilities; for example, 42 permutations of the bit codes $\{110000, 011000, 001100, 000110, 000011, 110010, 011010\}$ are shellings.

[The concept of a shelling for monotone Boolean functions was introduced by Michael O. Ball and J. Scott Provan, *Operations Research* **36** (1988), 703–715, who discussed many significant applications.]

36. If $j < k$ we have $B_j = \alpha 1 \beta$ and $B_k = \alpha 0 \gamma$ for some strings α, β, γ. Form the sequence $x_0 = \alpha 1 \gamma$, $x_1 = x_0'$, …, $x_l = x_{l-1}'$, where $x_l = \alpha 0 0^{|\gamma|}$. We have $f(x_0) = 1$ since $x_0 \supseteq B_k$, but $f(x_l) = 0$ since $x_l \subseteq B_j'$. So the string x_i, where $f(x_i) = 1$ and $f(x_{i+1}) = \cdots = f(x_l) = 0$, is in B. It precedes B_k and proves that $B_j \mathbin{\&} S_k \supseteq 0^{|\alpha|} 1 0^{|\beta|}$.

[This construction and parts of exercise 35 are due to E. Boros, Y. Crama, O. Ekin, P. L. Hammer, T. Ibaraki, and A. Kogan, *SIAM J. Discrete Math.* **13** (2000), 212–226.]

37. The shelling order $(000011,\ 0011\underline{0}1,\ 00110\underline{0},\ 11\underline{0}1\underline{0}1,\ 11\underline{0}10\underline{0},\ 110\underline{00}1,\ 110\underline{0}0\underline{0})$ generalizes to all n. There also are interesting solutions not based on shelling, like the cyclically symmetrical ($110{*}{*}{*}$, $1110{*}{*}$, ${*}{*}110{*}$, ${*}{*}1110$, $0{*}{*}{*}11$, $10{*}{*}11$, 111111).

For the lower bound, assign the weight $w_x = -\prod_{j=1}^{n}(x_{2j-1} + x_{2j} - 3x_{2j-1}x_{2j})$ to each point x, and notice that the sum of w_x over all x in any subcube is 0 or ± 1. (It suffices to verify this curious fact for each of the nine possible subcubes when $n = 1$.) Now choose a set of disjoint subcubes that partition the set $F = \{x \mid f(x) = 1\}$; we have

$$\sum_{C \text{ chosen}} 1 \;\geq\; \sum_{C \text{ chosen}} \sum_{x \in C} w_x \;=\; \sum_{x \in F} w_x \sum_{C \text{ chosen}} [x \in C] \;=\; \sum_{x \in F} w_x.$$

There are $\binom{n}{k} 2^{n-k}$ vectors x with exactly k pairs $x_{2j-1}x_{2j} = 1$ and nonzero weight. Their weight is $(-1)^{k-1}$, and they lie in F except when $k = 0$. Hence $\sum_{x \in F} w_x = \sum_{k>0} \binom{n}{k} 2^{n-k} (-1)^{k-1} = 2^n - (2-1)^n$. [See M. O. Ball and G. L. Nemhauser, *Mathematics of Operations Research* **4** (1979), 132–143.]

38. Certainly not; a DNF is satisfiable if and only if it has at least one implicant. The hard problem for a DNF is to decide whether or not it is a *tautology* (always true).

39. Associate variables y_1, …, y_N with each internal node in preorder, so that every tree node corresponds to exactly one variable of F. For each internal node y, with children (l, r) and labeled with the binary operator $\circ$, construct four 3CNF clauses $c_{00} \wedge c_{01} \wedge c_{10} \wedge c_{11}$, where

$$c_{pq} = (y^{\overline{p \circ q} N} \vee l^{pN} \vee r^{qN})$$

and N denotes complementation (so that $x^{0N} = x$ and $x^{1N} = \bar{x}$). These clauses state in effect that $y = l \circ r$; for example, if $\circ$ is $\wedge$, the four clauses are $(\bar{y} \vee l \vee r) \wedge (\bar{y} \vee l \vee \bar{r}) \wedge (\bar{y} \vee \bar{l} \vee r) \wedge (y \vee \bar{l} \vee \bar{r})$. Finally, add one more clause, $(y_1 \vee y_1 \vee y_1)$, to force $F = 1$.

Every higher number can be formed by mere complications of threes.
. . . Take the quadruple fact that A sells B to C for the price D.
This is a compound of two facts:
first, that A makes with C a certain transaction, which we may name E;
and second, that this transaction E is a sale of B for the price D.
— CHARLES S. PEIRCE, *A Guess at the Riddle* (1887)

shelling
Ball
Provan
Boros
Crama
Ekin
Hammer
Ibaraki
Kogan
Ball
Nemhauser
tautology
PEIRCE

linear ordering
Cook
Reckhow
Haken
3SAT
queue
stack
core
Quick

40. Following the hint, A says '$u < v \oplus v < u$' and B says '$u < v \wedge v < w \Rightarrow u < w$'. So $A \wedge B$ says that there's a linear ordering of the vertices, $u_1 < u_2 < \cdots < u_n$. (There are $n!$ ways to satisfy $A \wedge B$.) Now C says that q_{uvw} is equivalent to $u < v < w$; so D says that u and w are not consecutive in the ordering, when $u \not- w$. Thus $A \wedge B \wedge C \wedge D$ is satisfiable if and only if there is a linear ordering in which all nonadjacent vertices are nonconsecutive (that is, in which all consecutive vertices are adjacent).

41. Solution 0: '$[m \le n]$' is such a formula, but it is not in the spirit of this exercise.

Solution 1: Let x_{jk} mean that pigeon j occupies hole k. Then the clauses are $(x_{j1} \vee \cdots \vee x_{jn})$ for $1 \le j \le m$ and $(\bar{x}_{ik} \vee \bar{x}_{jk})$ for $1 \le i < j \le m$ and $1 \le k \le n$. [See S. A. Cook and R. A. Reckhow, *J. Symbolic Logic* **44** (1979), 36–50; A. Haken, *Theoretical Comp. Sci.* **39** (1985), 297–308.]

Solution 2: Assume that $n = 2^t$ and let pigeon j occupy hole $(x_{j1} \ldots x_{jt})_2$. The clauses $\big((x_{i1} \oplus x_{j1}) \vee \cdots \vee (x_{it} \oplus x_{jt})\big)$ for $1 \le i < j \le m$ can be put into the CNF form $(y_{ij1} \vee \cdots \vee y_{ijt})$ as in exercise 39, by introducing auxiliary clauses $(\bar{y}_{ijk} \vee x_{ik} \vee x_{jk}) \wedge (y_{ijk} \vee x_{ik} \vee \bar{x}_{jk}) \wedge (y_{ijk} \vee \bar{x}_{ik} \vee x_{jk}) \wedge (\bar{y}_{ijk} \vee \bar{x}_{ik} \vee \bar{x}_{jk})$. The total size of this CNF is $\Theta(m^2 \log n)$, compared to $\Theta(m^2 n)$ in Solution 1. If n is not a power of 2, $O(m \log n)$ additional clauses of size $O(\log n)$ will rule out inappropriate values.

42. $(\bar{x} \vee y) \wedge (\bar{z} \vee x) \wedge (\bar{y} \vee \bar{z}) \wedge (z \vee z)$.

43. Probably not, because every 3SAT problem can be converted to this form. For example, the clause $(x_1 \vee x_2 \vee \bar{x}_3)$ can be replaced by $(x_1 \vee \bar{y} \vee \bar{x}_3) \wedge (\bar{y} \vee \bar{x}_2) \wedge (y \vee x_2)$, where y is a new variable (essentially equivalent to $\bar{x}_2$).

44. Suppose $f(x) = f(y) = 1$ implies $f(x \& y) = 1$ and also that, say, $c = x_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4$ is a prime clause of f. Then $c' = \bar{x}_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4$ is *not* a clause; otherwise $c \wedge c' = x_2 \vee \bar{x}_3 \vee \bar{x}_4$ would also be a clause, contradicting primality. So there's a vector y with $f(y) = 1$ and $y_1 = 1$, $y_2 = 0$, $y_3 = y_4 = 1$. Similarly, there's a z with $f(z) = 1$ and $z_1 = 0$, $z_2 = 1$, $z_3 = z_4 = 1$. But then $f(y \,\&\, z) = 1$, and c isn't a clause. The same argument works for a clause c that has a different number of literals, as long as at least two of the literals aren't complemented.

45. (a) A Horn function $f(x_1, \ldots, x_n)$ is indefinite if and only if it is unequal to the definite Horn function $g(x_1, \ldots, x_n) = f(x_1, \ldots, x_n) \vee (x_1 \wedge \cdots \wedge x_n)$. So $f \leftrightarrow g$ is a one-to-one correspondence between indefinite and definite Horn functions. (b) If f is monotone, its complement $\bar{f}$ is either identically 1 or an indefinite Horn function.

46. Algorithm C puts 88 pairs `xy` in the core: When `x` = `a`, `b`, `c`, `0`, or `1`, the following character `y` can be anything but `(`. When `x` = `(`, `*`, `/`, `+`, `-`, we can have `y` = `(`, `a`, `b`, `c`, `0`, `1`; also `y` = `-` when `x` = `(`, `+`, or `-`. Finally, the legitimate pairs beginning with `x` = `)` are `)+`, `)-`, `)*`, `)/`, `))`.

47. The order in which Algorithm C brings vertices into the core is a topological sort, since all predecessors of k are asserted before the algorithm sets `TRUTH`$(x_k) \leftarrow 1$. But Algorithm 2.2.3T uses a queue instead of a stack, so the ordering it actually produces is usually different from that of Algorithm C.

48. Let $\perp$ be a new variable, and change every indefinite Horn clause to a definite one by ORing in this new variable. (For example, '$\bar{w} \vee \bar{y}$' becomes '$\bar{w} \vee \bar{y} \vee \perp$', namely '$w \wedge y \Rightarrow \perp$'; definite Horn clauses stay unchanged.) Then apply Algorithm C. The original clauses are unsatisfiable if and only if $\perp$ is in the core of the new clauses. The algorithm can therefore be terminated as soon as it is about to set `TRUTH`$(\perp) \leftarrow 1$.

(J. H. Quick thought of another solution: We could apply Algorithm C to the function g constructed in the answer to exercise 45(a), because f is unsatisfiable if and only if *every* variable x_j is in the core of g. However, indefinite clauses of f such as $\bar{w}\vee\bar{y}$ become many different clauses $(\bar{w}\vee\bar{y}\vee z)\wedge(\bar{w}\vee\bar{y}\vee x)\wedge(\bar{w}\vee\bar{y}\vee v)\wedge(\bar{w}\vee\bar{y}\vee u)\wedge\cdots$ of g, one for each variable not in the original clause. So Quick's suggestion, which might sound elegant at first blush, could increase the number of clauses by a factor of $\Omega(n)$.)

Kleine Büning
Lettmann
renaming
Istrate
chess
palindrome
Williams

49. We have $f\le g$ if and only if $f\wedge\bar{g}$ is unsatisfiable, if and only if $f\wedge\bar{c}$ is unsatisfiable for every clause c of g. But $\bar{c}$ is an AND of literals, so we can apply exercise 48. [See H. Kleine Büning and T. Lettmann, *Aussagenlogik: Deduktion und Algorithmen* (1994), §5.6, for further results including an efficient way to test if g is a "renaming" of f, namely to determine whether or not there exist constants $(y_1,\ldots,y_n)$ such that $f(x_1,\ldots,x_n)=g(x_1\oplus y_1,\ldots,x_n\oplus y_n)$.]

50. See Gabriel Istrate, *Random Structures & Algorithms* **20** (2002), 483–506.

51. If vertex v is marked A, introduce the clauses $\Rightarrow A^+(v)$ and $\Rightarrow B^-(v)$; if it is marked B, introduce $\Rightarrow A^-(v)$ and $\Rightarrow B^+(v)$. Otherwise let v have k outgoing arcs $v\to u_1,\ \ldots,\ v\to u_k$. Introduce the clauses $A^-(u_j)\Rightarrow B^+(v)$ and $B^-(u_j)\Rightarrow A^+(v)$ for $1\le j\le k$. Also, if v is not marked C, introduce the clauses $A^+(u_1)\wedge\cdots\wedge A^+(u_k)\Rightarrow B^-(v)$ and $B^+(u_1)\wedge\cdots\wedge B^+(u_k)\Rightarrow A^-(v)$. All forcing strategies are consequences of these clauses. Exercise 2.2.3–28 and its answer provide further information.

Notice that, in principle, Algorithm C can therefore be used to decide whether or not the game of chess is a forced victory for the white pieces — except for the annoying detail that the corresponding digraph is larger than the physical universe.

52. With best play, the results (see exercise 51) are:

n	(a)	(b)	(c)	(d)
2	0 wins	second player wins	1 wins	second player wins
3	0 wins	first player wins	first player wins	first player wins
4	first player wins	first player wins	first player wins	first player wins
5	second player wins	draw	draw	1 loses if first
6	second player wins	second player wins	1 loses if first	1 loses if first
7	1 loses if first	second player wins	1 loses if first	1 loses if first
8	draw	draw	draw	1 loses if first
9	draw	draw	draw	1 loses if first

(Here "1 loses if first" means that the game is a draw if player 0 plays first, otherwise 0 can win.) *Comments:* In (a), player 1 has a slight disadvantage, because $f(x)=0$ when $x_1\ldots x_n$ is a palindrome. This small difference affects the result even when $n=7$. Although player 1 would seem to be better off playing 0s in the left half of the board, it turns out that his/her first move when $n=4$ must be to $*1{*}{*}$; the alternative, $*0{*}{*}$, draws. Game (b) is essentially a race to see who can eliminate the last $*$. In game (c), a random choice of $x_1\ldots x_n$ makes $f(x)=1$ with probability $F_{n+2}/2^n=\Theta((\phi/2)^n)$; in game (d), this probability approaches zero more slowly, as $\Theta(1/\log n)$. Still, player 1 does better in (c) than in (d) when $n=2$, 5, 8, and 9; no worse in the other cases.

53. (a) She should switch either day 1 or day 2 to day 3. (b, f) Several possibilities; for example, change day 2 to day 3. (c) This case is illustrated in Fig. 6; change either Desert or Excalibur to Aladdin. (d) Change either Caesars or Excalibur to Aladdin. (e) Change either Bellagio or Desert to Aladdin. Of course Williams, who doesn't appear in the cycle (42), bears no responsibility whatever for the conflicts.

54. If x and $\bar{x}$ are both in S, then $u\in S\iff\bar{u}\in S$, because the existence of paths from x to $\bar{x}$ and $\bar{x}$ to x and x to u and u to x implies the existence of paths from $\bar{u}$ to $\bar{x}$ and $\bar{x}$ to $\bar{u}$, hence from u to $\bar{u}$ and $\bar{u}$ to u.

Lewis
Aspvall
Kleine Büning
Lettmann
2CNF
Krom
first-order predicate calculus
unsolvable

55. (a) Necessary and sufficient conditions for successfully renaming a clause such as $x_1 \vee \bar{x}_2 \vee x_3 \vee \bar{x}_4$ are $(y_1 \vee \bar{y}_2) \wedge (y_1 \vee y_3) \wedge (y_1 \vee \bar{y}_4) \wedge (\bar{y}_2 \vee y_3) \wedge (\bar{y}_2 \vee \bar{y}_4) \wedge (y_3 \vee \bar{y}_4)$. A similar set of $\binom{k}{2}$ clauses of length 2 in the variables $\{y_1, \ldots, y_n\}$ corresponds to any clause of length k in $\{x_1, \ldots, x_n\}$. [H. R. Lewis, *JACM* **25** (1978), 134–135.]

(b) A given clause of length $k > 3$ in $\{x_1, \ldots, x_n\}$ can be converted into $3(k-2)$ clauses of length 2, instead of the $\binom{k}{2}$ clauses above, by introducing $k-3$ new variables $\{t_2, \ldots, t_{k-2}\}$, illustrated here for the clause $x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_5$:

$$(y_1 \vee y_2) \wedge (y_1 \vee t_2) \wedge (y_2 \vee t_2) \wedge (\bar{t}_2 \vee y_3) \wedge (\bar{t}_2 \vee t_3) \wedge (y_3 \vee t_3) \wedge (\bar{t}_3 \vee y_4) \wedge (\bar{t}_3 \vee y_5) \wedge (y_4 \vee y_5).$$

In general, the clauses from $x_1 \vee \cdots \vee x_k$ are $(\bar{t}_{j-1} \vee y_j) \wedge (\bar{t}_{j-1} \vee t_j) \wedge (y_j \vee t_j)$ for $1 < j < k$, but with t_1 replaced by $\bar{y}_1$ and t_{k-1} replaced by y_k; change y_j to $\bar{y}_j$ if $\bar{x}_j$ appears instead of x_j. Do this for each given clause, using different auxiliary variables t_j for different clauses; the result is a formula in 2CNF that has length $< 3m$ and is satisfiable if and only if Horn renaming is possible. Now apply Theorem K.

[See B. Aspvall, *J. Algorithms* **1** (1980), 97–103. One consequence, noted by H. Kleine Büning and T. Lettmann in *Aussagenlogik: Deduktion und Algorithmen* (1994), Theorem 5.24, is that any satisfiable formula in 2CNF can be renamed to Horn clauses. Notice that two CNFs for the same function may give different outcomes; for example, $(x \vee y \vee z) \wedge (\bar{x} \vee \bar{y} \vee \bar{z}) \wedge (\bar{x} \vee z) \wedge (\bar{y} \vee z)$ is actually a Horn function, but the clauses in this representation cannot be converted to Horn form by complementation.]

56. Here $f(x, y, z)$ corresponds to the digraph shown below (analogous to Fig. 6), and it can also be simplified to $y \wedge (\bar{x} \vee z)$. Each vertex is a strong component. So the formula is true with respect to the quantifiers $\exists\exists\exists$, $\exists\exists\forall$, $\forall\exists\exists$; false in the other cases $\forall\exists\forall$, (any)$\forall$(any). In general the eight possibilities can be arranged at the corners of a cube, with each change from $\exists$ to $\forall$ making the formula more likely to be false.

$$\begin{tikzcd} x \arrow[r] & z \arrow[r] & y \\ \bar{y} \arrow[u] \arrow[r] & \bar{z} \arrow[r] & \bar{x} \arrow[u] \end{tikzcd}$$

57. Forming the digraph as in Theorem K, we can prove that the quantified formula holds if and only if (i) no strong component contains both x and $\bar{x}$; (ii) there is no path from one universal variable x to another universal variable y or to its complement $\bar{y}$; (iii) no strong component containing a universal variable x also contains an existential variable v or its complement $\bar{v}$, when '$\exists v$' appears to the left of '$\forall x$'. These three conditions are clearly necessary, and they are readily tested as the strong components are being found.

To show that they are sufficient, notice first that if S is a strong component with only existential literals, condition (i) allows us to set them all equal as in Theorem K. Otherwise S has exactly one universal literal, $u_j = x_j$ or $u_j = \bar{x}_j$; all other literals in S are existential and declared to the right of x_j, so we can equate them to u_j. And all paths into S in such a case come from purely existential strong components, whose value can be set to 0 because the complements of such strong components cannot also lead into S; for if v and $\bar{v}$ imply u_j, then $\bar{u}_j$ implies $\bar{v}$ and v.

[*Information Proc. Letters* **8** (1979), 121–123. By contrast, M. Krom had proved in *J. Symbolic Logic* **35** (1970), 210–216, that an analogous problem in first-order predicate calculus (where parameterized predicates take the place of simple Boolean variables, and quantification is over the parameters) is actually unsolvable in general.]

58. We can assume that each clause is definite, by introducing '$\perp$' as in exercise 48 and placing '$\forall\perp$' at the left. Call the universal variables x_0, x_1, $\ldots$, x_m (where x_0 is $\perp$) and call the existential variables y_1, $\ldots$, y_n. Let '$u \prec v$' mean that variable u appears to the left of variable v in the list of quantifiers. Remove $\bar{x}_j$ from any clause whose unbarred literal is y_k when $y_k \prec x_j$. Then, for $0 \le j \le m$, let C_j be the core of the Horn clauses when the additional clauses $(x_0) \wedge \cdots \wedge (x_{j-1}) \wedge (x_{j+1}) \wedge \cdots \wedge (x_m) \wedge \bigwedge\{(y_k) \mid y_k \prec x_j$ and $y_k \in C_0\}$ are appended. (In other words, C_j tells us what can be deduced when all the x's except x_j are assumed to be true.) We claim that the given formula is true if and only if $x_j \notin C_j$, for $0 \le j \le m$.

To prove this claim, note first that the formula is certainly false if $x_j \in C_j$ for some j. (When $y_k \in C_0$ and $y_k \prec x_j$ and $x_i = 1$ for $i \ne j$ we must set $y_k \leftarrow 1$.) Otherwise we can choose each y_k to make the formula true, as follows: If $y_k \notin C_0$, set $y_k \leftarrow 0$; otherwise set $y_k \leftarrow \bigwedge\{x_j \mid y_k \notin C_j\}$. Notice that y_k depends on x_j only when $x_j \prec y_k$. Each clause c with unbarred literal x_j is now true: For if $x_j = 0$, some $\bar{y}_k$ appears in c for which $y_k \notin C_j$, because $x_j \notin C_j$; hence $y_k = 0$. And each clause c with unbarred literal y_k is also true: If $y_k = 0$, we either have $y_k \notin C_0$, in which case some $\bar{y}_l$ in c is $\notin C_0$, hence $y_l = 0$; or $y_k \in C_0 \setminus C_j$ for some j, in which case some $x_j = 0$ and either $\bar{x}_j$ appears in c or some $\bar{y}_l$ appears in c where $y_l \notin C_j$, making $y_l = 0$.

[This solution is due to T. Dahlheimer. See M. Karpinski, H. Kleine Büning, and P. H. Schmitt, *Lecture Notes in Comp. Sci.* **329** (1988), 129–137; H. Kleine Büning, K. Subramani, and X. Zhao, *Lecture Notes in Comp. Sci.* **2919** (2004), 93–104.]

59. By induction on n: Suppose $f(0, x_2, \ldots, x_n)$ leads to the quantified results $y_1, \ldots, y_{2^{n-1}}$, while $f(1, x_2, \ldots, x_n)$ leads similarly to $z_1, \ldots, z_{2^{n-1}}$. Then $\exists x_1 f(x_1, x_2, \ldots, x_n)$ leads to $y_1 \vee z_1, \ldots, y_{2^{n-1}} \vee z_{2^{n-1}}$, and $\forall x_1 f(x_1, x_2, \ldots, x_n)$ leads to $y_1 \wedge z_1, \ldots, y_{2^{n-1}} \wedge z_{2^{n-1}}$. Now use the fact that $(y \vee z) + (y \wedge z) = y + z$. [See *Proc. Mini-Workshop on Quantified Boolean Formulas* **2** (QBF-02) (Cincinnati: May 2002), 1–16.]

60. Both (a) and (b). But (c) is always 0; (d) is always 1; (e) is $\overline{\langle xyz \rangle}$; (f) is $\bar{x} \vee \bar{y} \vee \bar{z}$.

61. True — indeed obviously so, when $w = 0$, and when $w = 1$.

62. Since $\{x_1, x_2, x_3\} \subseteq \{0, 1\}$, we can assume by symmetry that x_1 equals x_2. Then either $f(x_1, x_1, x_3, x_4, \ldots, x_n) = f(x_1, x_1, x_1, x_4, \ldots, x_n)$ or $f(x_1, x_1, x_3, x_4, \ldots, x_n) = f(x_3, x_1, x_3, x_4, \ldots, x_n)$, assuming only that f is monotone in its first three variables.

63. $\langle xyz \rangle = \langle xxyyz \rangle$. *Note:* Emil Post proved, in fact, that a single subroutine for *any* nontrivial monotone self-dual function will suffice to compute them all. (By induction on n, at least one appropriate way to call such an n-ary subroutine will yield $\langle xyz \rangle$.)

64. [*FOCS* **3** (1962), 149–157.] (a) If f is monotone and self-dual, Theorem P says that $f(x) = x_k$ or $f(x) = \langle f_1(x) f_2(x) f_3(x) \rangle$. The condition therefore holds either immediately or by induction. Conversely, if the condition holds it implies that f is monotone (when x and y differ in just one bit) and self-dual (when they differ in all bits).

(b) We merely need to show that it is possible to define f at one new point without introducing a conflict. Let x be the lexicographically smallest point where $f(x)$ is undefined. If $f(\bar{x})$ is defined, set $f(x) = \overline{f(\bar{x})}$. Otherwise if $f(x') = 1$ for some $x' \subseteq x$, set $f(x) = 1$; otherwise set $f(x) = 0$. Then the condition still holds.

65. If $\mathcal{F}$ is maximal intersecting, we have (i) $X \in \mathcal{F} \implies \overline{X} \notin \mathcal{F}$, where $\overline{X}$ is the complementary set $\{1, 2, \ldots, n\} \setminus X$; (ii) $X \in \mathcal{F}$ and $X \subseteq Y \implies Y \in \mathcal{F}$, because $\mathcal{F} \cup \{Y\}$ is intersecting; and (iii) $X \notin \mathcal{F} \implies \overline{X} \in \mathcal{F}$, because $\mathcal{F} \cup \{X\}$ must contain an element $Y \subseteq \overline{X}$. Conversely, one can prove without difficulty that any family $\mathcal{F}$ satisfying (i) and (ii) is intersecting, and maximal if it also satisfies (iii).

Punch line: All three statements are simple, in the language of Boolean functions: (i) $f(x) = 1 \implies f(\bar{x}) = 0$; (ii) $x \subseteq y \implies f(x) \le f(y)$; (iii) $f(x) = 0 \implies f(\bar{x}) = 1$.

66. [T. Ibaraki and T. Kameda, *IEEE Transactions on Parallel and Distributed Systems* **4** (1993), 779–794.] Every family with the property that $Q \subseteq Q'$ implies $Q = Q'$

Schensted
Gardner
Milnor
ZDDs
Bioch
Ibaraki

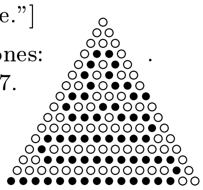
clearly corresponds to the prime implicants of a monotone Boolean function f. The further condition that $Q \cap Q' \neq \emptyset$ corresponds to the further relation $f(\bar{x}) \leq \overline{f(x)}$, because $f(\bar{x}) = f(x) = 1$ holds if and only if x and $\bar{x}$ both make prime implicants true.

If coteries $\mathcal{C}$ and $\mathcal{C}'$ correspond in this way to functions f and f', then $\mathcal{C}$ dominates $\mathcal{C}'$ if and only if $f \neq f'$ and $f'(x) \leq f(x)$ for all x. Then f' is not self-dual, because there is an x with $f'(\bar{x}) = 0$, $f(\bar{x}) = 1$; and we have $f(x) = 0$, hence $f'(x) = 0$.

Conversely, if f' is not self-dual, there's a y with $f'(y) = f'(\bar{y}) = 0$. If $y = 0\ldots0$, coterie $\mathcal{C}'$ is empty, and dominated by every other coterie. Otherwise define $f(x) = f'(x) \vee [x \supseteq y]$. Then f is monotone, and $f(\bar{x}) \leq \overline{f(x)}$ for all x; so it corresponds to a coterie that dominates $\mathcal{C}'$.

67. (a) By induction, if $Y(t_{100}) = Y(t_{010}) = 1$ we have a Y of 1s, because the Ys in t_{100} and t_{010} either intersect or are adjacent. Similarly, if $Y(t_{100}) = Y(t_{010}) = 0$ we do not have a Y of 1s, because there is a Y of 0s.

(b) This formula follows from (a) and the fact that $(t_{abc})_{def} = t_{(a+d)(b+e)(c+f)} = (t_{def})_{abc}$. [Schensted stated the results of this exercise, and those of exercises 62 and 69, in a 28-page letter sent to Martin Gardner on 21 January 1979. Milnor had written to Gardner on 26 March 1957 about a corresponding game called "Triangle."]

68. When $n = 15$, the author's best attempt so far has 59 black stones:  . (The answers for $1 \leq n \leq 10$ are respectively 2, 3, 4, 6, 8, 11, 14, 18, 23, 27. The prime implicants for these functions can be represented by fairly small ZDDs; see Section 7.1.4.)

69. The proof of Theorem P shows that we need only prove $Y(T) \leq f(x)$. A Y in T means that we've got at least one variable in each p_j. Therefore $f(\bar{x}_1, \ldots, \bar{x}_n) = 0$, and $f(x_1, \ldots, x_n) = 1$.

70. Self-duality of g is obvious for arbitrary t when f is self-dual: $\overline{g(\bar{x})} = \bigl(\overline{f(\bar{x})} \vee [\bar{x} = t]\bigr) \wedge [\bar{x} \neq \bar{t}] = \bigl(f(x) \vee [x = \bar{t}]\bigr) \wedge [x \neq t] = \bigl(f(x) \wedge [x \neq t]\bigr) \vee \bigl([x = \bar{t}] \wedge [x \neq t]\bigr) = g(x)$.

Let $x = x_1 \ldots x_{j-1} 0 x_{j+1} \ldots x_n$ and $y = x_1 \ldots x_{j-1} 1 x_{j+1} \ldots x_n$; for monotonicity we must prove that $g(x) \leq g(y)$. If $x = t$ or $y = t$, we have $g(x) = 0$; if $x = \bar{t}$ or $y = \bar{t}$, we have $g(y) = 1$; otherwise $g(x) = f(x) \leq f(y) = g(y)$. [*European J. Combinatorics* **16** (1995), 491–501; discovered independently by J. C. Bioch and T. Ibaraki, *IEEE Transactions on Parallel and Distributed Systems* **6** (1995), 905–914.]

71. $\langle\langle xyz\rangle uv\rangle = \langle\langle\langle xyz\rangle uv\rangle uv\rangle = \langle\langle\langle yuv\rangle x\langle zuv\rangle\rangle uv\rangle = \langle\langle yuv\rangle\langle xuv\rangle\langle\langle zuv\rangle uv\rangle\rangle = \langle\langle xuv\rangle\langle yuv\rangle\langle zuv\rangle\rangle$.

72. For (58), $v = \langle uvu\rangle = u$. For (59), $\langle uyv\rangle = \langle vu\langle xuy\rangle\rangle = \langle\langle vux\rangle uy\rangle = \langle xuy\rangle = y$. And for (60), $\langle xyz\rangle = \langle\langle xuv\rangle yz\rangle = \langle x\langle uyz\rangle\langle vyz\rangle\rangle = \langle xyy\rangle = y$.

73. (a) If $d(u,v) = d(u,x) + d(x,v)$, we obviously obtain a shortest path of the form $u \mathrel{-\!\!-} \cdots \mathrel{-\!\!-} x \mathrel{-\!\!-} \cdots \mathrel{-\!\!-} v$. Conversely, if $[uxv]$, let $u \mathrel{-\!\!-} \cdots \mathrel{-\!\!-} x \mathrel{-\!\!-} \cdots \mathrel{-\!\!-} v$ be a shortest path, with l steps to x followed by m steps to v. Then $d(u,v) = l + m \geq d(u,x) + d(x,v) \geq d(u,v)$.

(b) For all z, $\langle zxu\rangle = \langle z\langle vux\rangle\langle yux\rangle\rangle = \langle\langle zvy\rangle ux\rangle \in \bigl\{\langle yux\rangle, \langle vux\rangle\bigr\} = \{u, x\}$.

(c) We can assume that $d(x,u) \geq d(x,v) > 0$. Let $u \mathrel{-\!\!-} \cdots \mathrel{-\!\!-} y \mathrel{-\!\!-} v$ be a shortest path, and let $w = \langle xuy\rangle$. Then $\langle vxw\rangle = \langle v\langle vux\rangle\langle wux\rangle\rangle = \langle\langle vvw\rangle ux\rangle = \langle vux\rangle = x$, so $x \in [w \mathinner{.\,.} v]$. We have $[uwy]$, because $d(u,y) < d(u,v)$ and $w \in [u \mathinner{.\,.} y]$. If $w \neq u$ we have $d(w,v) < d(u,v)$; hence $[wxv]$, hence $[uxv]$. If $w = u$ we have $x \mathrel{-\!\!-} u$ by (b). But $d(x,u) \geq d(x,v)$; therefore $x \mathrel{-\!\!-} v$, and $[uxv]$.

(d) Let $y = \langle uxv\rangle$. Since $y \in [u\,..\,x]$, we have $d(u,x) = d(u,y) + d(y,x)$ by (a) and (c). Similarly, $d(u,v) = d(u,y) + d(y,v)$ and $d(x,v) = d(x,y) + d(y,v)$. But these three equations, together with $d(u,v) = d(u,x) + d(x,v)$, yield $d(x,y) = 0$. [*Proc. Amer. Math. Soc.* **12** (1961), 407–414.]

74. $w = \langle yxw\rangle = \langle yx\langle zxw\rangle\rangle = \langle yx\langle zx\langle yzw\rangle\rangle\rangle = \langle\langle yxz\rangle x\langle yzw\rangle\rangle = \langle x\langle xyz\rangle\langle wyz\rangle\rangle = \langle\langle xxw\rangle yz\rangle = \langle xyz\rangle$ by (55), (55), (55), (52), (51), (53), and (50).

75. (a) If $w = \langle xxy\rangle$ we have $[xwx]$ by (iii), hence $w = x$ by (i).

(b) Axiom (iii) and part (a) tell us that $[xxy]$ is always true. So we can set $x = y$ in (ii) to conclude that $[uxv] \iff [vxu]$. The definition of $\langle xyz\rangle$ in (iii) is therefore perfectly symmetrical between x, y, and z.

(c) By the definition of $\langle uxv\rangle$ in (iii), we have $x = \langle uxv\rangle$ if and only if $[uxx]$, $[uxv]$, and $[xxv]$. But we know that $[uxx]$ and $[xxv]$ are always true.

(d) In this step and subsequent steps, we will construct one or more auxiliary points of M and then use Algorithm C to derive every consequence of the betweenness relations that are known. (The axioms have the convenient form of Horn clauses.) For example, here we define $z = \langle xyv\rangle$, so that we know $[uxy]$, $[uyv]$, $[xzy]$, $[xzv]$, and $[yzv]$. From these hypotheses we deduce $[uzy]$ and $[uzv]$. So $z = \langle uyv\rangle = y$.

(e) The hinted construction implies, among other things, $[utv]$, $[utz]$, $[vtz]$, $[uwv]$, $[uwz]$, $[vwz]$; hence $t = w$. (A computer program is helpful here.) Adding the hypotheses $[rws]$, $[rwz]$, $[swz]$ now yields $[xyz]$ as desired; it also turns out that $r = p$ and $s = q$.

(f) Let $r = \langle yuv\rangle$, $s = \langle zuv\rangle$, $t = \langle xyz\rangle$, $p = \langle xrs\rangle$, $q = \langle tuv\rangle$; then $[pqp]$ flows out. [*Proc. Amer. Math. Soc.* **5** (1954), 801–807. For early studies of betweenness axioms, see E. V. Huntington and J. R. Kline, *Trans. Amer. Math. Soc.* **18** (1917), 301–325.]

76. Axiom (i) obviously holds, and axiom (ii) follows from commutativity and (52). The answer to exercise 74 derives (iii) from the identity $\langle xyz\rangle = \langle x\langle xyz\rangle\langle wyz\rangle\rangle$; so we need only verify that formula: $\langle x\langle xyz\rangle\langle wyz\rangle\rangle = \langle\langle yxz\rangle x\langle wyz\rangle\rangle = \langle\langle\langle yxz\rangle xz\rangle x\langle wyz\rangle\rangle = \langle\langle yxz\rangle x\langle zx\langle wyz\rangle\rangle\rangle = \langle x\langle xyz\rangle\langle z\langle xyz\rangle w\rangle\rangle = \langle\langle x\langle xyz\rangle z\rangle\langle xyz\rangle w\rangle = \langle\langle xyz\rangle\langle xyz\rangle w\rangle$.

Notes: The original treatment of median algebra by Birkhoff and Kiss in *Bull. Amer. Math. Soc.* **53** (1947), 749–752, assumed (50), (51), and the short distributive law (53). The fact that associativity (52) actually implies distributivity was not realized until many years later; M. Kolibiar and T. Marcisová, *Matematický Časopis* **24** (1974), 179–185, proved it via Sholander's axioms as in this exercise. A mechanical derivation of (53) from (50)–(52) was found in 2005 by R. Veroff and W. McCune, using an extension of the Otter theorem prover.

77. (a) In coordinate $r \longrightarrow s$ of the labels, suppose $l(r)$ has a 0 and $l(s)$ has a 1; then the left vertices have 0 in that coordinate. If $u \longrightarrow v \longrightarrow u'$, where u and u' are on the left but v is on the right, $\langle uu'v\rangle$ lies on the left. But $[u\,..\,v] \cap [u'\,..\,v] = \{v\}$, unless $u = u'$.

(b) This statement is obvious, by Corollary C.

(c) Suppose $u \longrightarrow v$ and $u' \longrightarrow v'$, where u and u' are on the left, v and v' are on the right. Let $v = v_0 \longrightarrow \cdots \longrightarrow v_k = v'$ be a shortest path, and let $u_0 = u$, $u_k = u'$. All vertices v_j lie on the right, by (b). The left vertex $u_1 = \langle u_0v_1u_k\rangle$ must be a common neighbor of u_0 and v_1, since the distance $d(u_0,v_1) = 2$. (We cannot have $u_1 = u_0$, because that would imply the existence of a shortest path from v to v' going through the left vertex u.) Therefore v_1 has rank 1; and so do v_2, …, v_{k-1}, by the same argument. [L. Nebeský, *Commentationes Mathematicæ Universitatis Carolinæ* **12** (1971), 317–325; M. Mulder, *Discrete Math.* **24** (1978), 197–204.]

(d) These steps visit all vertices v of rank 1 in order of their distance $d(v,s)$ from s. If such a v has a late neighbor u not yet seen, the rank of u must be 1 or 2. If the

Horn clauses
Huntington
Kline
Birkhoff
Kiss
distributive law
Kolibiar
Marcisová
Sholander
Veroff
McCune
Otter
automated deduction
theorem proving
Nebeský
Mulder

rank is 1, u will have at least two early neighbors, namely v and the future `MATE(u)`. Step I8 bases its decision on an arbitrary early neighbor w of u such that $w \neq v$. Since $d(w,v) = 2$, the vertex $x = \langle svw \rangle$ has rank 1 by (c). If w has rank 0, then $x = v$; so u has rank 1. Otherwise $d(x,s) < d(v,s)$, and the rank of w was correctly determined when x was visited. If w has rank 1, u lies on a shortest path from v to w; if w has rank 2, w lies on a shortest path from u to s. In both cases u and w have the same rank, by (c).

(e) The algorithm removes all edges equivalent to r — s, by (a) and (d). Their removal clearly disconnects the graph; the two pieces that remain are convex by (b), so they are connected and in fact they are median graphs. Step I7 records all of the relevant relations between the two pieces, because all 4-cycles that disappear are examined there. By induction on the number of vertices, each piece is properly labeled.

78. Every time v appears in step I4, it loses one of its neighbors u_j. Each of these edges v — u_j corresponds to a different coordinate of the labels, so we can assume that $l(v)$ has the form $\alpha 1^k$ for some binary string α. The labels for $u_1, u_2, \ldots, u_k$ are then $\alpha 01^{k-1}$, $\alpha 101^{k-2}$, $\ldots$, $\alpha 1^{k-1}0$. By taking componentwise medians, we can now prove that all 2^k labels of the form $\alpha\beta$ occur for vertices in the graph, since $\langle(\alpha\beta)(\alpha\beta')(0\ldots0)\rangle$ is the bit string $\alpha(\beta \mathbin{\&} \beta')$.

79. (a) If $l(v) = k$, exactly $\nu(k)$ smaller vertices are neighbors of v.

(b) At most $\lfloor n/2 \rfloor$ 1s appear in bit position j, for $0 \le j < \lceil \lg n \rceil$.

(c) Suppose exactly k vertices have labels beginning with 0. At most $\min(k, n-k)$ edges correspond to that bit position, and at most $f(k) + f(n-k)$ other edges are present. But

$$f(n) = \max_{0 \le k \le n} \bigl(\min(k, n-k) + f(k) + f(n-k)\bigr),$$

because the function $g(m,n) = f(m+n) - m - f(m) - f(n)$ satisfies the recurrence

$$g(2m+a, 2n+b) = ab + g(m+a, n) + g(m, n+b) \qquad \text{for } 0 \le a, b \le 1.$$

It follows by induction that $g(m,m) = g(m,m+1) = 0$, and that $g(m,n) \ge 0$ when $m \le n$. [*Annals of the New York Academy of Sciences* **175** (1970), 170–186; D. E. Knuth, *Proc. IFIP Congress 1971* (1972), 24.]

80. (a) (Solution by W. Imrich.) The graph with vertex labels 0000, 0001, 0010, 0011, 0100, 0110, 0111, 1100, 1101, 1110, 1111 cannot be labeled in any essentially different way; but the distance from 0001 to 1101 is 4, not 2.

(b) The cycle C_{2m} is a partial cube, because its vertices can be labeled $l(k) = 1^k 0^{m-k}$, $l(m+k) = 0^k 1^{m-k}$ for $0 \le k < m$. But the bitwise median of $l(0)$, $l(m-1)$, and $l(m+1)$ is $01^{m-2}0$; and indeed those vertices don't have a median, when $m > 2$.

81. Yes. A median graph is a subgraph of a hypercube, which is bipartite.

82. The general case reduces to the simple case where G has only two vertices $\{0,1\}$, because we can operate componentwise on the median labels, and because $d(u,v)$ is the Hamming distance between $l(u)$ and $l(v)$.

In the simple case, the stated rule sets $u_k \leftarrow v_k$ except when $u_{k-1} = v_{k-1} = v_{k+1} \neq v_k$, and it is readily proved optimum. (Other optimum possibilities do exist, however; for example, if $v_0v_1v_2v_3 = 0110$, we could set $u_0u_1u_2u_3 = 0000$.)

[This problem was motivated by the study of self-organizing data structures. F. R. K. Chung, R. L. Graham, and M. E. Saks, in *Discrete Algorithms and Complexity* (Academic Press, 1987), 351–387, have proved that median graphs are the *only* graphs for which u_k can always be chosen optimally as a function of $(v_0, v_1, \ldots, v_{k+1})$, regardless of the subsequent values $(v_{k+2}, \ldots, v_t)$. They have also characterized all cases for which a given finite amount of lookahead will suffice, in *Combinatorica* **9** (1989), 111–131.]

binary recurrence
Knuth
Imrich
median labels
Hamming distance
self-organizing data structures
Chung
Graham
Saks

83. Again it suffices to consider the simple two-vertex case. An optimum solution in that case can never have $u_{k-1} \neq u_k$ except when $u_{k-1} = v_{k-1} \neq v_k = u_k$. Therefore we must only decide which runs of consecutive 0s or 1s in $v_0v_1 \ldots v_t$ should induce matching runs in $u_0u_1 \ldots u_t$.

Suppose $v_{k-1}v_k \ldots v_{k+r} = 01^r0$ and $u_{k-1} = 0$. The cost of $u_{k-1}u_k \ldots u_{k+r} = 00\ldots0$ is $r\rho$, compared to a cost of 2 if $u_{k-1}u_k \ldots u_{k+r} = 01^r0$; so we must choose the former if $\rho < 2/r$ and the latter if $\rho > 2/r$. This policy can be expressed in terms of medians, if we set $u_k = \langle u_{k-1} \ldots u_{k-1}v_k \ldots v_{k+r}\rangle$ when $2/(r+1) \leq \rho < 2/r$ for $r \geq 0$, where there are r occurrences of u_{k-1}. (This median-of-$(2r{+}1)$ gives correct results in a general median graph, since it will produce valid median labels; see exercise 86.)

But we must change this rule when we get near the end, because it eventually asks for the values of v_j with $j > t$. The endgame is tricky; for if $v_{k-1}v_k \ldots v_t = 01^r$ and $u_{k-1} = 0$, with $r = t+1-k$, the choice $u_{k-1}u_k \ldots u_t = 00\ldots0$ is now preferable only if $\rho < 1/r$. For example, if $2/4 < \rho < 2/3$ and $v_0v_1 = 01$ and $t = 1$, we want to choose $u_0u_1 = 00$; but if $v_0v_1v_2 = 011$ and $t = 2$ we want $u_0u_1u_2 = 011$. One solution is to set $v_{t+j} \leftarrow v_t$ for $1 \leq j \leq \lceil r/2 \rceil$ and $v_{t+j} \leftarrow v_0$ for $\lceil r/2 \rceil < j \leq r$.

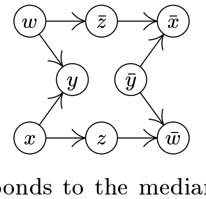
84. There are 81 such functions, each of which can be represented as the median of an odd number of elements. Seven types of vertices occur:

Type	Typical vertex	Cases	Adjacent to	Degree
1	$\langle z\rangle$	5	$\langle vwxyzzz\rangle$	1
2	$\langle vwxyzzz\rangle$	5	$\langle z\rangle, \langle wxyzz\rangle$	5
3	$\langle wxyzz\rangle$	20	$\langle vwxyzzz\rangle, \langle vwxxyyzzz\rangle$	4
4	$\langle vwxxyyzzz\rangle$	30	$\langle xyz\rangle, \langle wxyzz\rangle, \langle vwxyyzz\rangle$	5
5	$\langle vwxyyzz\rangle$	10	$\langle vwxxyyzzz\rangle, \langle vwxyz\rangle$	7
6	$\langle vwxyz\rangle$	1	$\langle vwxyyzz\rangle$	10
7	$\langle xyz\rangle$	10	$\langle vwxxyyzzz\rangle$	3

[Von Neumann and Morgenstern enumerated these seven types in their book *Theory of Games and Economic Behavior* (1944), §52.5, in connection with the study of an equivalent problem about systems of winning coalitions that they called *simple games*. The graph for six-variable functions, which has 2646 vertices of 30 types, appears in the paper by Meyerowitz cited in exercise 70. Only 21 of those types can be represented as a simple median-of-odd; a vertex like $\langle\langle abd\rangle\langle ace\rangle\langle bcf\rangle\rangle$, for example, has no such representation. Let the corresponding graph for n variables have M_n vertices; P. Erdős and N. Hindman, in *Discrete Math.* **48** (1984), 61–65, showed that $\lg M_n$ is asymptotic to $\binom{n-1}{\lfloor n/2\rfloor}$. D. Kleitman, in *J. Combin. Theory* **1** (1966), 153–155, showed that the vertices for distinct projection functions like x and y are always furthest apart in this graph.]

85. Every strong component must consist of a single vertex; otherwise two coordinates would always be equal, or always complementary. Thus the digraph must be acyclic.

Furthermore, there must be no path from a vertex to its complement; otherwise a coordinate would be constant.



When these two conditions are satisfied, we can prove that no vertex x is redundant, by assigning the value 0 to all vertices that precede x or $\bar x$, assigning 1 to all vertices that follow, and giving appropriate values to all other vertices.

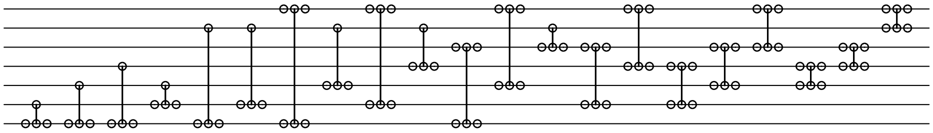
(Consequently we obtain a completely different way to represent a median graph. For example, the digraph shown corresponds to the median graph whose labels are $\{0000, 0001, 0010, 0011, 0111, 1010\}$.)

runs
Von Neumann
Morgenstern
coalitions
simple games
Meyerowitz
pure majority
Erdős
Hindman
Kleitman
projection functions
diameter of free median graph
acyclic
median graph

Mayr
Subramanian
distributive lattice
isometric subgraph
Bandelt
Feder

86. Yes. By Theorem P, *any* monotone self-dual function maps elements of X into X.

87. Here the topological ordering $7\,6\,5\,4\,3\,2\,1\,\bar{1}\,\bar{2}\,\bar{3}\,\bar{4}\,\bar{5}\,\bar{6}\,\bar{7}$ can replace (72); we get

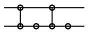


(Consecutive inverters on the same line can, of course, be canceled out.)

88. A given value of d contributes at most $6\lceil t/d\rceil$ units of delay (for $2\lceil t/d\rceil$ clusters).

89. Suppose first that the new condition is $i \to j$ while the old was $i' \to j'$, where $i < j$ and $i' < j'$ and there are no complemented literals. The new module changes $x_1 \ldots x_t$ to $y_1 \ldots y_t$, where $y_i = x_i \wedge x_j$, $y_j = x_i \vee x_j$, and $y_k = x_k$ otherwise. We certainly have $y_{i'} \le y_{j'}$ when $\{i', j'\} \cap \{i, j\} = \emptyset$. And there is no problem if $i = i'$, since $y_{i'} = y_i \le x_i = x_{i'} \le x_{j'} = y_{j'}$. But the case $i = j'$ is trickier: Here the relations $i' \to i$ and $i \to j$ imply also $i' \to j$; and this relation has been enforced by *previous* modules, because modules have been appended in order of decreasing distance d in the topological ordering $u_1 \ldots u_{2t}$. Therefore $y_{i'} = x_{i'} \le x_j$ and $y_{i'} \le x_{j'} = x_i$, hence $y_{i'} \le x_i \wedge x_j = y_i = y_{j'}$. A similar proof works when $j = i'$ or $j = j'$.

Finally, with complemented literals, the construction cleverly reduces the general case to the uncomplemented case by inverting and un-inverting the bits.

90. When $t = 2$,  does the job. The general case follows recursively from this building block by reducing t to $\lceil t/2\rceil$.

[The study of CI-nets, and other networks of greater generality, was initiated by E. W. Mayr and A. Subramanian, *J. Computer and System Sci.* **44** (1992), 302–323.]

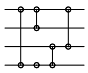
91. The answer does not yet seem to be known even in the special case when the median graph is a free tree (with $t+1$ vertices), or in the monotone case when it is a distributive lattice as in Corollary F. In the latter case, inverters may be unnecessary.

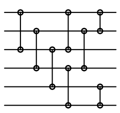
93. Let $d_X(u,v)$ be the number of edges on a shortest path between u and v, when the path lies entirely within X. Clearly $d_X(u,v) \ge d_G(u,v)$. And if $u = u_0 — u_1 — \cdots — u_k = v$ is a shortest path in G, the path $u = f(u_0) — f(u_1) — \cdots — f(u_k) = v$ lies in X when f is a retraction from G to X; hence $d_X(u,v) \le d_G(u,v)$.

94. If f is a retraction of the t-cube onto X, two different coordinate positions cannot always be equal or always complementary for all $x \in X$, unless they are constant. For if, say, all elements of X have the forms $00{*}\ldots{*}$ or $11{*}\ldots{*}$, there would be no path between vertices of those two types, contradicting the fact that X is an isometric subgraph (hence connected).

Given $x, y, z \in X$, let $w = \langle xyz\rangle$ be their median in the t-cube. Then $f(w) \in [x\,..\,y] \cap [x\,..\,z] \cap [y\,..\,z]$, because (for example) $f(w)$ lies on a shortest path from x to y in X. So $f(w) = w$, and we have proved that $w \in X$. [This result and its considerably more subtle converse are due to H. J. Bandelt, *J. Graph Theory* **8** (1984), 501–510.]

95. False (although the author was hoping otherwise); the network at the right takes $0001 \mapsto 0000$, $0010 \mapsto 0011$, $1101 \mapsto 0110$, but nothing $\mapsto 0010$.





(The set of all possible outputs appears to have no easy characterization, even when no inverters are used. For example, the pure-comparator network at the left, constructed by Tomás Feder, takes $000000 \mapsto 000000$, $010101 \mapsto 010101$, and $101010 \mapsto 011001$, but nothing $\mapsto 010001$. See also exercises 5.3.4–50, 5.3.4–52.)

96. No. If f is a threshold function based on real parameters $w=(w_1,\ldots,w_n)$ and t, let $\max\{w\cdot x \mid f(x)=0\}=t-\epsilon$. Then $\epsilon>0$, and f is defined by the 2^n inequalities $w\cdot x-t\ge 0$ when $f(x)=1$, $t-w\cdot x-\epsilon\ge 0$ when $f(x)=0$. If A is any $M\times N$ matrix of integers for which the system of linear inequalities $Av\ge(0,\ldots,0)^T$ has a real-valued solution $v=(v_1,\ldots,v_N)^T$ with $v_N>0$, there also is such a solution in integers. (Proof by induction on N.) So we can assume that $w_1,\ \ldots,\ w_n,\ t$, and ϵ are integers.

[A closer analysis using Hadamard's inequality (see Eq. 4.6.1–(25)) proves in fact that integer weights of magnitude at most $(n+1)^{(n+1)/2}/2^n$ will suffice; see S. Muroga, I. Toda, and S. Takasu, *J. Franklin Inst.* **271** (1961), 376–418, Theorem 16. Furthermore, exercise 112 shows that weights nearly that large are sometimes needed.]

97. $\langle 11111x_1x_2\rangle$, $\langle 111x_1x_2\rangle$, $\langle 1x_1x_2\rangle$, $\langle 0x_1x_2\rangle$, $\langle 000x_1x_2\rangle$, $\langle 00000x_1x_2\rangle$.

98. We may assume that $f(x_1,\ldots,x_n)=\langle y_1^{w_1}\ldots y_n^{w_n}\rangle$, with positive integer weights w_j and with $w_1+\cdots+w_n$ odd. Let δ be the minimum positive value of the 2^n sums $\pm w_1\pm\cdots\pm w_n$, with n independently varying signs. Renumber all subscripts so that $w_1+\cdots+w_k-w_{k+1}-\cdots-w_n=\delta$. Then $w_1y_1+\cdots+w_ny_n>\frac12(w_1+\cdots+w_n)\iff w_1(y_1-\frac12)+\cdots+w_n(y_n-\frac12)>0\iff w_1(y_1-\frac12)+\cdots+w_n(y_n-\frac12)>-\delta/2\iff w_1y_1+\cdots+w_ny_n>\frac12(w_1+\cdots+w_n-(w_1+\cdots+w_k-w_{k+1}-\cdots-w_n))=w_{k+1}+\cdots+w_n\iff w_1y_1+\cdots+w_ky_k-w_{k+1}\bar y_{k+1}-\cdots-w_n\bar y_n>0$.

99. We have $[x_1+\cdots+x_{2s-1}+s(y_1+\cdots+y_{2t-2})\ge st]=[\lfloor(x_1+\cdots+x_{2s-1})/s\rfloor+y_1+\cdots+y_{2t-2}\ge t]$; and $\lfloor(x_1+\cdots+x_{2s-1})/s\rfloor=[x_1+\cdots+x_{2s-1}\ge s]$.

(For example, $\langle\langle xyz\rangle uv\rangle=\langle xyzu^2v^2\rangle$, a quantity that we also know is equal to $\langle x\langle yuv\rangle\langle zuv\rangle\rangle$ and $\langle\langle xuv\rangle\langle yuv\rangle\langle zuv\rangle\rangle$ by Eqs. (53) and (54). *Reference:* C. C. Elgot, *FOCS* **2** (1961), 238.)

100. True, because of the preceding exercise and (45).

101. (a) When $n=7$ they are $x_7\wedge x_6$, $x_6\wedge x_5$, $x_7\wedge x_5\wedge x_4$, $x_6\wedge x_4\wedge x_3$, $x_7\wedge x_5\wedge x_3\wedge x_2$, $x_6\wedge x_4\wedge x_2\wedge x_1$, $x_7\wedge x_5\wedge x_3\wedge x_1$; and in general there are n prime implicants, forming a similar pattern. (We have either $x_n=x_{n-1}$ or $x_n=\bar x_{n-1}$. In the first case, $x_n\wedge x_{n-1}$ is obviously a prime implicant. In the second case, $F_n(x_1,\ldots,x_{n-1},\bar x_{n-1})=F_{n-1}(x_1,\ldots,x_{n-1})$; so we use the prime implicants of the latter, and insert x_n when x_{n-1} does not appear.)

(b) The shelling pattern (0000011, $000011\underline{0}$, $00011\underline{0}1$, $0011\underline{0}1\underline{0}$, $011\underline{0}1\underline{0}1$, $11\underline{0}1\underline{0}1\underline{0}$, $1\underline{0}1\underline{0}1\underline{0}1$) for $n=7$ works for all n.

(c) Two of several possibilities for $n=7$ illustrate the general case:

$$F_7(x_1,\ldots,x_7)\;=\;Y\begin{pmatrix} & & & & & x_6 & & & & & \\ & & & & x_7 & & x_5 & & & & \\ & & & x_6 & & x_6 & & x_4 & & & \\ & & x_7 & & x_5 & & x_7 & & x_3 & & \\ & x_6 & & x_6 & & x_4 & & x_6 & & x_2 & \\ x_7 & & x_5 & & x_7 & & x_3 & & x_7 & & x_1\end{pmatrix}\;=\;Y\begin{pmatrix} & & & & & x_6 & & & & & \\ & & & & x_7 & & x_5 & & & & \\ & & & x_6 & & x_6 & & x_4 & & & \\ & & x_7 & & x_5 & & x_5 & & x_3 & & \\ & x_6 & & x_6 & & x_4 & & x_4 & & x_2 & \\ x_7 & & x_5 & & x_5 & & x_3 & & x_3 & & x_1\end{pmatrix}.$$

[The Fibonacci threshold functions were introduced by S. Muroga, who also discovered the optimality result in exercise 105; see *IEEE Transactions* **EC-14** (1965), 136–148.]

102. (a) By (1) and (2), $\hat f(\bar x_0,\bar x_1,\ldots,\bar x_n)$ is the complement of $\hat f(x_0,x_1,\ldots,x_n)$.

(b) If f is given by (75), $\hat f$ is $[(w+1-2t)x_0+w_1x_1+\cdots+w_nx_n\ge w+1-t]$, where $w=w_1+\cdots+w_n$. Conversely, if $\hat f$ is a threshold function, so is $f(x_1,\ldots,x_n)=\hat f(1,x_1,\ldots,x_n)$. [E. Goto and H. Takahasi, *Proc. IFIP Congress* (1962), 747–752.]

linear inequalities
Hadamard's inequality
Muroga
Toda
Takasu
Elgot
shelling
Muroga
Goto
Takahasi

103. [See R. C. Minnick, *IRE Transactions* **EC-10** (1961), 6–16.] We want to minimize $w_1 + \cdots + w_n$ subject to the constraints $w_j \geq 0$ for $1 \leq j \leq n$ and $(2e_1 - 1)w_1 + \cdots + (2e_n - 1)w_n \geq 1$ for each prime implicant $x_1^{e_1} \wedge \cdots \wedge x_n^{e_n}$. For example, if $n = 6$, the prime implicant $x_2 \wedge x_5 \wedge x_6$ would lead to the constraint $-w_1 + w_2 - w_3 - w_4 + w_5 + w_6 \geq 1$. If the minimum is $+\infty$, the given function is not a threshold function. (The answer to exercise 84 gives one of the simplest examples of such a case.) Otherwise, if the solution $(w_1, \ldots, w_n)$ involves only integers, it minimizes the desired size. When noninteger solutions arise, additional constraints must be added until the best solution is found, as in part (c) of the following exercise.

Minnick
prime implicants of a majority function
dual
Muroga
Tsuboi
Baugh

104. First we need an algorithm to generate the prime implicants $x_1^{e_1} \wedge \cdots \wedge x_n^{e_n}$ of a given majority function $\langle x_1^{w_1} \ldots x_n^{w_n} \rangle$, when $w_1 \geq \cdots \geq w_n$ and $w_1 + \cdots + w_n$ is odd:

K1. [Initialize.] Set $t \leftarrow 0$. Then for $j = n, n-1, \ldots, 1$ (in this order), set $a_j \leftarrow t$, $t \leftarrow t + w_j$, $e_j \leftarrow 0$. Finally set $t \leftarrow (t+1)/2$, $s_1 \leftarrow 0$, and $l \leftarrow 0$.

K2. [Enter level l.] Set $l \leftarrow l + 1$, $e_l \leftarrow 1$, $s_{l+1} \leftarrow s_l + w_l$.

K3. [Below threshold?] If $s_{l+1} < t$, return to K2.

K4. [Visit a prime implicant.] Visit the exponents $(e_1, \ldots, e_n)$.

K5. [Downsize.] Set $e_l \leftarrow 0$. Then if $s_l + a_l \geq t$, set $s_{l+1} \leftarrow s_l$ and go to K2.

K6. [Backtrack.] Set $l \leftarrow l - 1$. Terminate if $l = 0$; otherwise go to K5 if $e_l = 1$; otherwise repeat this step. ▌

(a) $\langle x_1 x_2^2 x_3^3 x_4^5 x_5^6 x_6^8 x_7^{10} x_8^{12} \rangle$ (21 prime implicants).

(b) The optimum weights for $\langle x_0^{16-2t} x_1^8 x_2^4 x_3^2 x_4 \rangle$ are $w_0 w_1 w_2 w_3 w_4 =$ 10000, 31111, 21110, 32211, 11100, 23211, 12110, 13111, 01000, for $0 \leq t \leq 8$; the other cases are dual.

(c) Here the optimum weights $(w_1, \ldots, w_{10})$ are $(29, 25, 19, 15, 12, 8, 8, 3, 3, 0)/2$; so we learn that x_{10} is irrelevant, and we must deal with fractional weights. Constraining $w_8 \geq 2$ gives integer weights $(15, 13, 10, 8, 6, 4, 4, 2, 1, 0)$, which must be optimum because their sum exceeds the previous sum by 2. (Only two of the 175,428 self-dual threshold functions on nine variables have nonintegral weights minimizing $w_1 + \cdots + w_n$; the other one is $\langle x_1^{17} x_2^{15} x_3^{11} x_4^9 x_5^7 x_6^5 x_7^4 x_8^2 x_9 \rangle$. The largest w_1 in a minimum representation occurs in $\langle x_1^{42} x_2^{22} x_3^{18} x_4^{15} x_5^{13} x_6^{10} x_7^8 x_8^4 x_9^3 \rangle$; the largest $w_1 + \cdots + w_9$ occurs uniquely in $\langle x_1^{34} x_2^{32} x_3^{28} x_4^{27} x_5^{24} x_6^{20} x_7^{18} x_8^{15} x_9^{11} \rangle$, which is also an example of the largest w_9. See S. Muroga, T. Tsuboi, and C. R. Baugh, *IEEE Transactions* **C-19** (1970), 818–825.)

105. When $n = 7$, the inequalities generated in exercise 103 are $w_7 + w_6 - w_5 - w_4 - w_3 - w_2 - w_1 \geq 1$, $-w_7 + w_6 + w_5 - w_4 - w_3 - w_2 - w_1 \geq 1$, $w_7 - w_6 + w_5 + w_4 - w_3 - w_2 - w_1 \geq 1$, $-w_7 + w_6 - w_5 + w_4 + w_3 - w_2 - w_1 \geq 1$, $w_7 - w_6 + w_5 - w_4 + w_3 + w_2 - w_1 \geq 1$, $-w_7 + w_6 - w_5 + w_4 - w_3 + w_2 + w_1 \geq 1$, $w_7 - w_6 + w_5 - w_4 + w_3 - w_2 + w_1 \geq 1$. Multiply them respectively by 1, 1, 2, 3, 5, 8, 5 to get $w_1 + \cdots + w_7 \geq 1 + 1 + 2 + 3 + 5 + 8 + 5$. The same idea works for all $n \geq 3$.

106. (a) $\langle x_1^{2^{n-1}} x_2^{2^{n-2}} \ldots x_n \; \bar{y}_1^{2^{n-1}} \bar{y}_2^{2^{n-2}} \ldots \bar{y}_n \bar{z} \rangle$. (By exercise 99, we could also perform n medians-of-three: $\langle \langle \ldots \langle x_n \bar{y}_n \bar{z} \rangle \ldots x_2 \bar{y}_2 \rangle x_1 \bar{y}_1 \rangle$.)

(b) If $\langle x_1^{u_1} x_2^{u_2} \ldots x_n^{u_n} \bar{y}_1^{v_1} \bar{y}_2^{v_2} \ldots \bar{y}_n^{v_n} \bar{z}^w \rangle$ solves the problem, $2^{n+1} - 1$ basic inequalities need to hold; for example, when $n = 2$ they are $u_1 + u_2 - v_1 + v_2 - w \geq 1$, $u_1 + u_2 - v_1 - v_2 + w \geq 1$, $u_1 - u_2 + v_1 - v_2 - w \geq 1$, $u_1 - u_2 - v_1 + v_2 + w \geq 1$, $-u_1 + u_2 + v_1 + v_2 - w \geq 1$, $-u_1 + u_2 + v_1 - v_2 + w \geq 1$, $-u_1 - u_2 + v_1 + v_2 + w \geq 1$. Add them all up to get $u_1 + u_2 + \cdots + u_n + v_1 + v_2 + \cdots + v_n + w \geq 2^{n+1} - 1$.

107.

f	$N(f)$	$\Sigma(f)$	f	$N(f)$	$\Sigma(f)$	f	$N(f)$	$\Sigma(f)$	f	$N(f)$	$\Sigma(f)$
$\bot$	0	$(0,0)$	$\overline{\subset}$	1	$(0,1)$	$\overline{\vee}$	1	$(0,0)$	$\overline{\mathrm{L}}$	2	$(0,1)$
$\wedge$	1	$(1,1)$	R	2	$(1,2)$	$\equiv$	2	$(1,1)$	$\supset$	3	$(1,2)$
$\overline{\supset}$	1	$(1,0)$	$\oplus$	2	$(1,1)$	$\overline{\mathrm{R}}$	2	$(1,0)$	$\overline{\wedge}$	3	$(1,1)$
L	2	$(2,1)$	$\vee$	3	$(2,2)$	$\subset$	3	$(2,1)$	$\top$	4	$(2,2)$

Notice that $\oplus$ and $\equiv$ have the same parameters $N(f)$ and $\Sigma(f)$; they are the only Boolean binary operations that aren't threshold functions.

108. If $\Sigma(g) = (s_0, s_1, \ldots, s_n)$, the value of g is 1 in s_0 cases when $x_0 = 1$ and in $2^n - s_0$ cases when $x_0 = 0$. We also have $\Sigma(f_0) + \Sigma(f_1) = (s_1, \ldots, s_n)$, and

$$\begin{aligned}\Sigma(f_0) &= \sum_{x_1=0}^{1} \cdots \sum_{x_n=0}^{1} (\bar{x}_1, \ldots, \bar{x}_n)\,g(0, \bar{x}_1, \ldots, \bar{x}_n)\\ &= \sum_{x_1=0}^{1} \cdots \sum_{x_n=0}^{1} \big((1, \ldots, 1) - (x_1, \ldots, x_n)\big)\big(1 - g(1, x_1, \ldots, x_n)\big)\\ &= (2^{n-1} - s_0, \ldots, 2^{n-1} - s_0) + \Sigma(f_1).\end{aligned}$$

So the answers, for $n > 0$, are (a) $N(f_0) = 2^n - s_0$, $\Sigma(f_0) = \frac{1}{2}(s_1 - s_0 + 2^{n-1}, \ldots, s_n - s_0 + 2^{n-1})$; (b) $N(f_1) = s_0$, $\Sigma(f_1) = \frac{1}{2}(s_1 + s_0 - 2^{n-1}, \ldots, s_n + s_0 - 2^{n-1})$. [Equivalent results were presented by E. Goto in lectures at MIT in 1963.]

109. (a) $a_1 + \cdots + a_k \ge b_1 + \cdots + b_k$ if and only if $k - a_1 - \cdots - a_k \le k - b_1 - \cdots - b_k$.

(b) Let $\alpha^+ = (a_1, a_1{+}a_2, \ldots, a_1{+}\cdots{+}a_n)$. Then the vector $(c_1, \ldots, c_n)$ obtained by componentwise minimization of α^+ and β^+ is $(\alpha \wedge \beta)^+$. (Clearly $c_j = c_{j-1} + a_j$ or b_j.)

(c) Proceed as in (b) but with componentwise *maximization*; or take $\overline{\bar{\alpha} \wedge \bar{\beta}}$.

(d) True, because max and min satisfy these distributive laws. (In fact, we obtain a distributive *mixed-radix majorization lattice* in a similar way from the set of all n-tuples $a_1 \ldots a_n$ such that $0 \le a_j < m_j$ for $1 \le j \le n$.)

(e) $\alpha 1$ covers $\alpha 0$ and $\alpha 10\beta$ covers $\alpha 01\beta$. [This characterization is due to R. O. Winder, *IEEE Trans.* **EC-14** (1965), 315–325, but he didn't prove the lattice property. The lattice is often called $M(n)$; see B. Lindström, *Nordisk Mat. Tidskrift* **17** (1969), 61–70; R. P. Stanley, *SIAM J. Algebraic and Discrete Methods* **1** (1980), 177–179.]

(f) Because of (e) we have $r(\alpha) = na_1 + (n-1)a_2 + \cdots + a_n$.

(g) The point is that $0\beta \succeq 0\alpha$ if and only if $\beta \succeq \alpha$ and that $1\beta \succeq 0\alpha$ if and only if $1\beta \succeq 10\ldots0 \vee 0\alpha = 1\alpha'$.

(h) That is, how many $a_1 \ldots a_n$ have the property that $a_1 \ldots a_k$ contains no more 1s than 0s? The answer is $\binom{n}{\lfloor n/2 \rfloor}$; see, for example, exercise 2.2.1–4 or 7.2.1.6–42(a).

110. (a) If $x \subseteq y$ then $x \preceq y$, hence $f(x) \le f(y)$; QED.

(b) No; a threshold function need not be monotone (see (79)). But we *can* show that f is regular if we also require $w_n \ge 0$: For if $f(x) = 1$ and y covers x we then have $w \cdot y \ge w \cdot x$.

(c) Whenever $f(x) = 1$ and $x_j < x_{j+1}$, we have $f(y) = 1$ when y covers x with $x_j \leftrightarrow x_{j+1}$; hence $s_j \ge s_{j+1}$. (This argument holds even when $w_n < 0$.)

(d) No; consider, for example, $\langle x_1 x_2^2 x_3^2 \rangle$, which equals $\langle x_1 x_2 x_3 \rangle$. Counterexamples can arise even when the weights minimize $w_1 + \cdots + w_n$, because the solution to the linear program in exercise 103 is not always unique. One such case, found by Muroga, Tsuboi, and Baugh, is $\langle x_1^{17} x_2^9 x_3^8 x_4^6 x_5^7 x_6^5 x_7^3 x_8^2 x_9^2 \rangle$, a function that is actually symmetric in x_4 and x_5. But if $s_j > s_{j+1}$ we must have $w_j > w_{j+1}$, because of (c).

111. (a) Find an optimum self-dual function f pointwise as in exercise 14; in case of ties, set $f(x_1,\ldots,x_n) = x_1$. Then $f(x_1,\ldots,x_n) = [\,r_1^{x_1}\ldots r_n^{x_n} \ge \sqrt{r_1\ldots r_n}\,]$, except that '$\ge$' becomes '$>$' when $x_1 = 0$. This function is regular when $r_1 \ge \cdots \ge r_n \ge 1$.

(b) Let g be the regular, self-dual function constructed in (a). If f is a given regular, self-dual function, we want to verify that $f(x) \le g(x)$ for all vectors x; this will imply that $f = g$, because both functions are self-dual.

Suppose $f(x) = 1$, and let $y \preceq x$ be minimal such that $f(y) = 1$. If we have verified that $g(y) = 1$, then indeed $g(x) = 1$, as desired. [See K. Makino and T. Kameda, *SIAM Journal on Discrete Mathematics* **14** (2001), 381–407.]

For example, there are only seven self-dual regular Boolean functions when $n = 5$, generated by the following minimal elements in Fig. 8: 10000; 01111, 10001; 01110, 10010; 01101, 10011, 10100; 01100; 01011, 11000; 00111. So an optimum coterie can be found by examining only a few function values.

(c) Suppose $1 > p_1 \ge \cdots \ge p_r \ge \frac{1}{2} > p_{r+1} \ge \cdots \ge p_n > 0$. Let $f_k(x_1,\ldots,x_n)$ be the kth monotone, self-dual function and $F_k(x_1,\ldots,x_n)$ its integer multilinear representation. We want to find the optimum availability $G(p_1,\ldots,p_n) = \max_k F_k(p_1,\ldots,p_n)$. If $p_1 \le p'_1$, $\ldots$, $p_n \le p'_n$, we have $F_k(p_1,\ldots,p_n) \le F_k(p'_1,\ldots,p'_n)$ by exercise 12(e); hence $G(p_1,\ldots,p_n) \le G(p'_1,\ldots,p'_n)$.

Therefore if $0 < r < n$ we have $G(p_1,\ldots,p_n) \le G(p_1,\ldots,p_r,\frac{1}{2},\ldots,\frac{1}{2})$. And the latter is $F(p_1,\ldots,p_r,\frac{1}{2},\ldots,\frac{1}{2})$, where f is the function derived in part (a). This function does not depend on $(x_{r+1},\ldots,x_n)$, so it gives the optimum.

If $r = 0$ the problem seems to be deeper. We have $G(p_1,\ldots,p_n) \le G(p_1,\ldots,p_1)$; so we can conclude that the optimum coterie is $f(x_1,\ldots,x_n) = x_1$ if we can show that $F_k(p,\ldots,p) \le p$ for all k whenever $p < \frac{1}{2}$. In general $F_k(p\ldots,p) = \sum_m c_m p^m (1-p)^{n-m}$, where c_m is the number of vectors x such that $f_k(x) = 1$ and $\nu x = m$. Exercise 70 tells us that $c_m + c_{n-m} = \binom{n}{m}$, for all k. And the Erdős–Ko–Rado theorem (exercise 7.2.1.3–111) tells us that $c_m \le \binom{n-1}{m-1}$ for any intersecting family of m-sets when $m \le n/2$. The result follows.

[See Y. Amir and A. Wool, *Information Processing Letters* **65** (1998), 223–228.]

112. (a) The leading terms are respectively 0, $+xy$, $-xy$, $+x$, $-xy$, $+y$, $-2xy$, $-xy$, $+xy$, $+2xy$, $-y$, $+xy$, $-x$, $+xy$, $-xy$, 1; so $F(f) = 1$ when f is $\wedge$, $\llcorner$, R, $\bar{\vee}$, $\equiv$, $\subset$, $\supset$, $\top$.

(b) The coefficient corresponding to exponents 01101, say, is $f_{0{*}{*}0{*}}$ in the notation of answer 12; it is a linear combination of truth table entries, always lying in the range $\lceil -2^{k-1}\rceil \le f_{0{*}{*}0{*}} \le \lceil 2^{k-1}\rceil$ when there are k asterisks. Thus the leading coefficient is positive if and only if the mixed-radix number

$$\begin{bmatrix} f_{{*}{*}\ldots{*}}, & f_{0{*}\ldots{*}}, & \ldots, & f_{{*}0\ldots0}, & f_{00\ldots0} \\ 2^m{+}1, & 2^{m-1}{+}1, & \ldots, & 2^1{+}1, & 2^0{+}1 \end{bmatrix}$$

is positive, where the f's are arranged in reverse order of Chase's sequence and the radix 2^k+1 corresponds to an f with k asterisks. For example, when $m = 2$ we have $F(f) = 1$ if and only if the sum $18f_{{*}{*}} + 6f_{0{*}} + 2f_{{*}0} + f_{00} = 18(f_{11} - f_{01} - f_{10} + f_{00}) + 6(f_{01} - f_{00}) + 2(f_{10} - f_{00}) + f_{00} = 18f_{11} - 12f_{01} - 16f_{10} + 11f_{00}$ is positive; so the threshold function can be written $\langle f_{11}^{18}\bar{f}_{01}^{12}\bar{f}_{10}^{16}f_{00}^{11}\rangle$.

(In this particular case the much simpler expression $\langle f_{11}f_{11}\bar{f}_{01}\bar{f}_{10}f_{00}\rangle$ is actually valid. But part (c) will show that when m is large we can't do a great deal better.)

(c) Suppose $F(f) = \bigl[\sum_\alpha v_\alpha(f_\alpha - \frac{1}{2}) > 0\bigr]$, where the sum is over all 2^m binary strings α of length m and where each v_α is an integer weight. Define

$$w_\alpha = \sum_\beta (-1)^{\nu(\alpha \dot{-} \beta)} v_\beta \qquad \text{and} \qquad F_\alpha = \sum_\beta (-1)^{\nu(\alpha \dot{-} \beta)} f_\beta - 2^{m-1}[\alpha = 00\ldots0];$$

Makino
Kameda
integer multilinear representation
Erdős
Ko
Rado
intersecting family
Amir
Wool
mixed-radix number

Alon
Vũ
Fischler
Tannenbaum
Kautz

thus, for example, $w_{01} = -v_{00} + v_{01} - v_{10} + v_{11}$ and $F_{11} = f_{00} - f_{01} - f_{10} + f_{11}$. One can show that $F_{1^k0^l} = 2^l f_{*^k0^l}$, if $F_\alpha = 0$ whenever $\nu(\alpha) > k > 0$; therefore the signs of the transformed truth coefficients F_α determine the sign of the leading coefficient in the multilinear representation. Furthermore, we now have $F(f) = [\sum_\alpha w_\alpha F_\alpha > 0]$.

The general idea of the proof is to choose test functions f from which we can derive properties of the transformed weights w_α. For example, if $f(x_1, \ldots, x_m) = x_1 \oplus \cdots \oplus x_k$, we find $F_\alpha = 0$ for all α except that $F_{1^k0^{m-k}} = (-1)^{k-1}2^{m-1}$. The multilinear representation of $x_1 \oplus \cdots \oplus x_k$ has leading term $\lceil(-2)^{k-1}\rceil x_1 \ldots x_k$; hence we can conclude that $w_{1^k0^{m-k}} > 0$, and in a similar way that $w_\alpha > 0$ for all α. In general if m changes to $m+1$ but f does not depend on x_{m+1}, we have $F_{\alpha 0} = 2F_\alpha$ and $F_{\alpha 1} = 0$.

The test function $x_2 \oplus \cdots \oplus x_m \oplus x_1\bar{x}_2 \ldots \bar{x}_m$ proves that

$$w_{1^m} > (2^{m-1}{-}1)w_{01^{m-1}} + \sum_{k=1}^{m-1} w_{1^k01^{m-1-k}} + \text{smaller terms},$$

where the smaller terms involve only w_α with $\nu(\alpha) \le m-2$. In particular, $w_{11} > w_{01} + w_{10} + w_{00}$. The test function $x_1 \oplus \cdots \oplus x_{m-1} \oplus \bar{x}_1 \ldots \bar{x}_{m-2}(x_{m-1} \oplus \bar{x}_m)$ proves

$$w_{1^{m-2}01} > (2^{m-2}{-}1)w_{1^{m-2}10} + \sum_{k=0}^{m-3} (w_{1^k01^{m-3-k}10} + w_{1^k01^{m-3-k}01}) + \text{smaller terms},$$

where the smaller terms this time have $\nu(\alpha) \le m-3$. In particular, $w_{101} > w_{110} + w_{010} + w_{001}$. By permuting subscripts, we obtain similar inequalities leading to

$$w_{\alpha_j} \;>\; (2^{\nu(\alpha_j)-1} - 1)w_{\alpha_{j-1}} \qquad \text{for } 0 < j < 2^m,$$

because the w's begin to grow rapidly. But we have $v_\alpha = \sum_\beta (-1)^{\nu(\beta \dot- \alpha)} w_\beta/n$; hence $|v_\alpha| = w_{11\ldots1}/n + O(w_{11\ldots1}/n^2)$. [*SIAM J. Discrete Math.* **7** (1994), 484–492. Important generalizations of this result have been obtained by N. Alon and V. H. Vũ, *J. Combinatorial Theory* **A79** (1997), 133–160.]

113. The stated g_3 is $S_{2,3,6,8,9}$ because the stated g_2 is $S_{2,3,4,5,8,9,10,11,12}$.

For the more difficult function $S_{1,3,5,8}$, let $g_1 = [\nu x \ge 6]$; $g_2 = [\nu x \ge 3]$; $g_3 = [\nu x - 5g_1 - 2g_2 \ge 2] = S_{2,4,5,9,10,11,12}$; $g_4 = [2\nu x - 15g_1 - 9g_3 \ge 1] = S_{1,3,5,8}$. [See M. A. Fischler and M. Tannenbaum, *IEEE Transactions* **C-17** (1968), 273–279.]

114. $[4x + 2y + z \in \{3,6\}] = (\bar{x}\wedge y\wedge z) \vee (x\wedge y\wedge \bar{z})$. In the same way, *any* Boolean function of n variables is a special case of a symmetric function of $2^n - 1$ variables. [See W. H. Kautz, *IRE Transactions* **EC-10** (1961), 378.]

115. Both sides are self-dual, so we may assume that $x_0 = 0$. Then

$$s_j = [x_j + \cdots + x_{j+m-1} > x_{j+m} + \cdots + x_{j+2m-1}].$$

If $x_1 + \cdots + x_{2m}$ is odd, we have $s_j = \bar{s}_{j+m}$; hence $s_1 + \cdots + s_{2m} = m$ and the result is 1. But if $x_1 + \cdots + x_{2m}$ is even, the difference $x_j + \cdots + x_{j+m-1} - x_{j+m} - \cdots - x_{j+2m-1}$ will be zero for at least one $j \le m$; that makes $s_j = s_{j+m} = 0$, so we will have $s_1 + \cdots + s_{2m} < m$.

116. (a) It's an implicant if and only if $f(x) = 1$ whenever $j \le \nu x \le n - k + j$. It's a prime implicant if and only if we also have $f(x) = 0$ when $\nu x = j-1$ or $\nu x = n-k+j+1$.

(b) Consider the string $v = v_0v_1 \ldots v_n$ such that $f(x) = v_{\nu x}$. By part (a), there are $\binom{a+b+c}{a,b,c}$ prime implicants when $v = 0^a1^{b+1}0^c$. In the stated case, $a = b = c = 3$, so there are 1680 prime implicants.

(c) For a general symmetric function, we add together the prime implicants for each run of 1s in v. Clearly there are more for $v = 0^{a+1}1^{b+1}0^{c-1}$ than for $v = 0^a1^{b+1}0^c$ when $a < c-1$; so v contains no two consecutive 0s when the maximum is reached.

Let $\hat{b}(m,n)$ be the maximum number of prime implicants possible when $v_m = 1$ and $v_j = 0$ for $m < j \le n$. Then when $m \le \frac{1}{2}n$ we have

$$\hat{b}(m,n) = \max_{0\le k\le m}\left(\binom{n}{k,\, m-k,\, n-m} + \hat{b}(k-2,n)\right) = \binom{n}{\lceil m/2\rceil,\, \lfloor m/2\rfloor,\, n-m} + \hat{b}(\lceil m/2\rceil - 2, n),$$

with $\hat{b}(-2,n) = \hat{b}(-1,n) = 0$. And the overall maximum is

$$\hat{b}(n) = \binom{n}{n_0, n_1, n_2} + \hat{b}(n_1-2,n) + \hat{b}(n_2-2,n), \qquad n_j = \left\lfloor\frac{n+j}{3}\right\rfloor.$$

In particular we have $\hat{b}(9) = 1698$, with the maximum occurring for $v = 1101111011$.

(d) By Stirling's approximation, $\hat{b}(n) = 3^{n+3/2}/(2\pi n) + O(3^n/n^2)$.

(e) In this case the appropriate recurrence for $m < \lceil n/2\rceil$ is

$$\tilde{b}(m,n) = \max_{0\le k\le m}\left(\binom{n}{k,\, m-k,\, n-m} + \binom{n}{k-1,\, 0,\, n-k+1} + \tilde{b}(k-2,n)\right) = \binom{n}{\lceil m/2\rceil,\, \lfloor m/2\rfloor,\, n-m} + \binom{n}{\lceil m/2\rceil - 1} + \tilde{b}(\lceil m/2\rceil - 2, n)$$

and $\tilde{b}(n) = \tilde{b}(\lceil n/2\rceil - 1, n)$ maximizes $\min(\text{prime implicants}(f), \text{prime implicants}(\bar{f}))$. We have $(\tilde{b}(1), \tilde{b}(2), \ldots) = (1,1,4,5,21,31,113,177,766,1271,4687,7999,34412,\ldots)$; for example, $\tilde{b}(9) = 766$ corresponds to $S_{0,2,3,4,8}(x_1,\ldots,x_9)$. Asymptotically, $\tilde{b}(n) = 2^{(3n+3+(n \bmod 2))/2}/(2\pi n) + O(2^{3n/2}/n^2)$.

References: Summaries, Summer Inst. for Symbolic Logic (Dept. of Math., Cornell Univ., 1957), 211–212; B. Dunham and R. Fridshal, *J. Symbolic Logic* **24** (1959), 17–19; A. P. Vikulin, *Problemy Kibernetiki* **29** (1974), 151–166, which reports on work done in 1960; Y. Igarashi, *Transactions of the IEICE of Japan* **E62** (1979), 389–394.

117. The maximum number of subcubes of the n-cube, with none contained in another, is obtained when we choose all subcubes of dimension $\lfloor n/3\rfloor$. (It is also obtained by choosing all subcubes of dimension $\lfloor (n+1)/3\rfloor$; for example, when $n = 2$ we can choose either $\{0{*}, 1{*}, {*}0, {*}1\}$ or $\{00, 01, 10, 11\}$.) Hence $b^*(n) = \binom{n}{\lfloor n/3\rfloor}2^{n-\lfloor n/3\rfloor} = 3^{n+1}/\sqrt{4\pi n} + O(3^n/n^{3/2})$. [See the paper of Vikulin in the previous answer, pages 164–166; A. K. Chandra and G. Markowsky, *Discrete Math.* **24** (1978), 7–11; N. Metropolis and G. C. Rota, *SIAM J. Applied Math.* **35** (1978), 689–694.]

118. Consider two functions equivalent if we can obtain one from the other by complementing and/or permuting variables, but not complementing the function value itself. Such functions clearly have the same number of prime implicants; this equivalence relation is studied further in answer 125 below. A computer program based on exercise 30 produces the following results:

m	Classes	Functions	m	Classes	Functions	m	Classes	Functions
0	1	1	5	87	17472	10	7	632
1	5	81	6	70	12696	11	1	96
2	18	1324	7	43	7408	12	2	24
3	46	6608	8	24	3346	13	1	16
4	87	14536	9	10	1296	14	0	0

Dunham Fridshal Vikulin Igarashi subcubes Vikulin Chandra Markowsky Metropolis Rota

Gadzhiev
full disjunctive form
Christmas tree pattern
Yablonsky
Nigmatullin
Glagolev
Korshunov
Pippenger
zigzag path

And here are the corresponding statistics for functions of five variables:

m	Classes	Functions	m	Classes	Functions	m	Classes	Functions
0	1	1	11	186447	666555696	22	338	608240
1	6	243	12	165460	590192224	23	130	197440
2	37	14516	13	129381	459299440	24	71	75720
3	244	318520	14	91026	319496560	25	37	28800
4	1527	3319580	15	57612	199792832	26	15	10560
5	6997	19627904	16	33590	113183894	27	6	2880
6	23434	73795768	17	17948	58653984	28	4	1040
7	57048	190814016	18	8880	27429320	29	2	640
8	105207	362973410	19	3986	11597760	30	2	48
9	152763	538238660	20	1795	4548568	31	2	64
10	183441	652555480	21	720	1633472	32	1	16

119. Several authors have conjectured that $b(n) = \hat{b}(n)$; M. M. Gadzhiev has proved that equality holds for $n \le 6$ [*Diskretnyĭ Analiz* **18** (1971), 3–24].

120. (a) Every prime implicant is a minterm, since no adjacent points of the n-cube have the same parity. So the full disjunctive form is the only decent DNF in this case.

(b) Now all prime implicants consist of two adjacent points. We must include the 14 subcubes $0^j{*}0^{6-j}$ and $1^j{*}1^{6-j}$ for $0 \le j \le 6$, in order to cover the points with $\nu x = 1$ and $\nu x = 6$. The other $\binom{7}{3} + \binom{7}{4} = 70$ points can be covered by 35 well-chosen prime implicants (see, for example, exercise 6.5–1, or the "Christmas tree pattern" in Section 7.2.1.6). Thus the shortest DNF has length 49. [An ingeniously plausible but fallacious argument that 70 prime implicants are necessary was presented by S. B. Yablonsky in *Problemy Kibernetiki* **7** (1962), 229–230.]

(c) For each of 2^{n-1} choices of $(x_1, \ldots, x_{n-1})$ we need at most one implicant to account for the behavior of the function with respect to x_n.

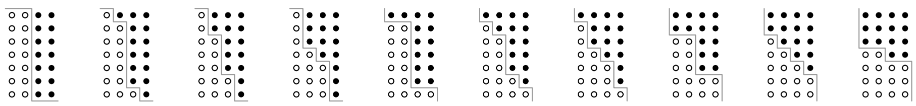
[Asymptotically, almost all Boolean functions of n variables have a shortest DNF with $\Theta(2^n/(\log n \log\log n))$ prime implicants. See R. G. Nigmatullin, *Diskretnyĭ Analiz* **10** (1967), 69–89; V. V. Glagolev, *Problemy Kibernetiki* **19** (1967), 75–94; A. D. Korshunov, *Metody Diskretnogo Analiza* **37** (1981), 9–41; N. Pippenger, *Random Structures & Algorithms* **22** (2003), 161–186.]

121. (a) Let $x = x_1 \ldots x_m$ and $y = y_1 \ldots y_n$. Since f is a function of $(\nu x, \nu y)$, there are altogether $2^{(m+1)(n+1)}$ possibilities.

(b) In this case $\nu x \le \nu x'$ and $\nu y \le \nu y'$ implies $f(x,y) \le f(x',y')$. Every such function corresponds to a zigzag path from $a_0 = (-\frac{1}{2}, n+\frac{1}{2})$ to $a_{m+n+2} = (m+\frac{1}{2}, -\frac{1}{2})$, with $a_j = a_{j-1} + (1,0)$ or $a_j = a_{j-1} - (0,1)$ for $1 \le j \le m+n+2$; we have $f(x,y) = 1$ if and only if the point $(\nu x, \nu y)$ lies above the path. So the number of possibilities is the number of such paths, namely $\binom{m+n+2}{m+1}$.

(c) Complementing x and y changes νx to $m - \nu x$ and νy to $n - \nu y$. So there are no such functions when m and n are both even; otherwise there are $2^{(m+1)(n+1)/2}$.

(d) The path in (b) must now satisfy $a_j + a_{m+n+2-j} = (m,n)$ for $0 \le j \le m+n+2$. Hence there are $\binom{\lceil m/2\rceil + \lceil n/2\rceil}{\lceil m/2\rceil}[m \text{ odd or } n \text{ odd}]$ such functions. For example, the ten cases when $m = 3$ and $n = 6$ are



122. A function of this kind is regular with the x's to the left of the y's if and only if the zigzag path does not contain two points (x, y) and $(x + 2, y)$ with $0 < y < n$; it is regular with the y's left of the x's if and only if the zigzag path does not contain both $(x, y + 2)$ and (x, y) with $0 < x < m$. It is a threshold function if and only if there is a straight line through the point $(m/2, n/2)$ with the property that (s, t) is above the line if and only if (s, t) is above the path, for $0 \le s \le m$ and $0 \le t \le n$. So cases 5 and 8, illustrated in the previous answer, fail to be regular; cases 1, 2, 3, 7, 9, and 10 are threshold functions. The regular non-threshold functions that remain can also be expressed as follows: $((x_1 \vee x_2 \vee x_3) \wedge \langle x_1x_2x_3y_1y_2y_3y_4y_5y_6\rangle) \vee (x_1 \wedge x_2 \wedge x_3)$ (case 4); $\langle 00x_1x_2x_3y_1y_2y_3y_4y_5y_6\rangle \vee (\langle x_1x_2x_3\rangle \wedge \langle 11x_1x_2x_3y_1y_2y_3y_4y_5y_6\rangle)$ (case 6).

123. Self-dual *regular* functions are relatively easy to list, for small n, but the numbers grow rapidly: When $n = 9$ there are 319,124 of them, found by Muroga, Tsuboi, and Baugh in 1967, and when $n = 10$ there are 1,214,554,343 (see exercise 7.1.4–00). The corresponding numbers for $n \le 6$ appear in Table 5, because all such functions are threshold functions when $n < 9$; there are 135 when $n = 7$, and 2470 when $n = 8$.

The threshold condition can be tested quickly for any such function by improving on the method of exercise 103, because constraints are needed only for the *minimal* vectors x (with respect to majorization) such that $f(x) = 1$.

The number θ_n of n-variable threshold functions is known to satisfy $\lg \theta_n = n^2 - O(n^2/\log n)$; see Yu. A. Zuev, *Matematicheskie Voprosy Kibernetiki* **5** (1994), 5–61.

124. The 222 equivalence classes listed in Table 5 include 24 classes of size $2^{n+1}n! = 768$; so there are $24 \times 768 = 18432$ answers to this problem. One of them is the function $(w \wedge (x \vee (y \wedge z))) \oplus z$.

125. 0; x; $x \wedge y$; $x \wedge y \wedge z$; $x \wedge (y \vee z)$; $x \wedge (y \oplus z)$. (These functions are $x \wedge f(y, z)$, where f runs through the equivalence classes of two-variable functions under permutation and/or complementation of variables but *not* of the function values. In general, let $f \simeq g$ mean that f is equivalent to g in that weaker sense, but write $f \cong g$ if they are equivalent in the sense of Table 5. Then $x \wedge f \cong x \wedge g$ if and only if $f \simeq g$, assuming that f and g are independent of the variable x. For it's easy to see that $(x \wedge f) \simeq (\bar{x} \vee \bar{g})$ is impossible. And if $(x \wedge f) \simeq (x \wedge g)$, we can prove that $f \simeq g$ by showing that, if σ is a signed permutation of $\{x_0, \ldots, x_n\}$ and if $x = x_1 \ldots x_n$, then the identity $x_0 \wedge f(x) = (x_0\sigma) \wedge g(x\sigma)$ implies $f(x) = g(x\sigma\tau)$, where τ interchanges $x_0 \leftrightarrow x_0\sigma$. Consequently the bottom line of Table 5 enumerates equivalence classes under $\simeq$, but with n increased by 1; there are, for example, 402 such classes of 4-variable functions.)

126. (a) The function is canalizing if and only if it has a prime implicant with at most one literal, or a prime clause with at most one literal.

(b) The function is canalizing if and only if at least one of the components of $\Sigma(f)$ is equal to 0, 2^{n-1}, $N(f)$, or $N(f) - 2^{n-1}$. [See I. Shmulevich, H. Lähdesmäki, and K. Egiazarian, *IEEE Signal Processing Letters* **11** (2004), 289–292, Proposition 6.]

(c) If, say, $\vee(f) = y_1 \ldots y_n$ with $y_j = 0$, then $f(x) = 0$ whenever $x_j = 1$. Therefore f is canalizing if and only if we don't have $\vee(f) = \vee(\bar{f}) = 1 \ldots 1$ and $\wedge(f) = \wedge(\bar{f}) = 0 \ldots 0$. With this test one can prove that many functions are noncanalizing when their value is known at only a few points.

127. (a) Since a self-dual function $f(x_1, \ldots, x_n)$ is true at exactly 2^{n-1} points, it is canalizing with respect to the variable x_j if and only if $f(x_1, \ldots, x_n) = x_j$.

(b) A definite Horn function is clearly canalizing if (i) it contains any clause with a single literal, or (ii) some literal occurs in every clause. Otherwise it is not canalizing. For we have $f(0, \ldots, 0) = f(1, \ldots, 1) = 1$, because (i) is false; and if x_j is any variable,

regular
Muroga
Tsuboi
Baugh
Zuev
equivalence classes
signed permutation
Shmulevich
Lähdesmäki
Egiazarian

there is a clause C_0 not containing $\bar{x}_j$ and a clause C_1 not containing x_j, because (ii) is false. By choosing appropriate values of the other variables, we can make $C_0 \wedge C_1$ false when $x_j = 0$ and also when $x_j = 1$.

128. For example, $(x_1 \wedge \cdots \wedge x_n) \vee (\bar{x}_1 \wedge \cdots \wedge \bar{x}_n)$.

129. $\sum_{k=1}^{n}(-1)^{k+1}\binom{n}{k}2^{2^{n-k}+k+1} - 2(n-1) - 4(n \bmod 2) = n2^{2^{n-1}+2} + O(n^2 2^{2^{n-2}})$. [See W. Just, I. Shmulevich, and J. Konvalina, *Physica* **D197** (2004), 211–221.]

130. (a) If there are a_n functions of n or fewer variables, but b_n functions of exactly n variables, we have $a_n = \sum_k \binom{n}{k} b_k$. Therefore $b_n = \sum_k (-1)^{n-k}\binom{n}{k} a_k$. (This rule, noted by C. E. Shannon in *Trans. Amer. Inst. Electrical Engineers* **57** (1938), 713–723, §4, applies to all rows of Table 3, *except* for the case of symmetric functions.) In particular, the answer sought here is $168 - 4 \cdot 20 + 6 \cdot 6 - 4 \cdot 3 + 2 = 114$.

(b) If there are a'_n essentially distinct functions of n or fewer variables, and b'_n of exactly n variables, we have $a'_n = \sum_{k=0}^{n} b'_k$. Hence $b'_n = a'_n - a'_{n-1}$, and the answer in this case is $30 - 10 = 20$.

131. Let there be $h(n)$ Horn functions and $k(n)$ Krom functions. Clearly $\lg h(n) \geq \binom{n}{\lfloor n/2 \rfloor}$ and $\lg k(n) \geq \binom{n}{2}$. V. B. Alekseyev [*Diskretnaîâ Matematika* **1** (1989), 129–136] has proved that $\lg h(n) = \binom{n}{\lfloor n/2 \rfloor}(1 + O(n^{-1/4}\log n))$. B. Bollobás, G. Brightwell, and I. Leader [*Israel J. Math.* **133** (2003), 45–60] have proved that $\lg k(n) \sim \frac{1}{2}n^2$.

132. (a) The hint is true because $\sum_y s(y)s(y \oplus z) = \sum_{w,x,y}(-1)^{f(w)+w\cdot y+f(x)+x\cdot(y+z)} = 2^n \sum_{w,x}(-1)^{f(w)+f(x)+x\cdot z}[x = w]$. Now suppose that $f(x) = g(x)$ for $2^{n-1} + k$ values of x; then $f(x) = g(x) \oplus 1$ for $2^{n-1} - k$ values of x. But if $|k| < 2^{n/2-1}$ for all affine g, we would have $|s(y)| < 2^{n/2}$ for all y, contradicting the hint when $z = 0$.

(b) Given $y = y_1 \ldots y_n$, there are exactly $2^{n/2}\bigl((y_1y_2 + y_3y_4 + \cdots + y_{n-1}y_n + 1 + h(y_1, y_3, \ldots, y_{n-1})) \bmod 2\bigr)$ solutions to $f(x) = x \cdot y \bmod 2$ when $x_{2k} = y_{2k-1}$ for $1 \leq k \leq n/2$, and there are $2^{n/2-1}$ solutions for each of the other $2^{n/2} - 1$ values of $(x_2, x_4, \ldots, x_n)$. So there are $2^{n-1} \pm 2^{n/2-1}$ solutions altogether. (This argument proves, in fact, that $\bigl(g(x_1, x_3, \ldots, x_{2n-1}) \cdot (x_2, x_4, \ldots, x_{2n}) + h(x_2, x_4, \ldots, x_{2n})\bigr) \bmod 2$ is bent whenever $g(x_1, x_3, \ldots, x_{2n-1})$ is a permutation of all $2^{n/2}$-bit vectors.)

(c) If $f(x)$ is bent, the argument in part (a) proves that $s(y) = 2^{n/2}(-1)^{g(y)}$ for some Boolean function $g(y)$. This function is also bent, because $\sum_y(-1)^{g(y)+w\cdot y} = 2^{-n/2}\sum_{x,y}(-1)^{f(x)+x\cdot y+w\cdot y} = 2^{n/2}\sum_x(-1)^{f(x)}[x = w] = 2^{n/2}(-1)^{f(w)}$ for all w. Consequently the hint tells us that we have $\sum_x(-1)^{f(x)+f(x\oplus z)} = 0$ for all nonzero z.

Conversely, assume that $f(x)$ satisfies the stated condition, and that $f(x) = 1$ for exactly k values of x. If y is any Boolean vector, the function $g(x) = (f(x) + x \cdot y) \bmod 2$ also satisfies the condition; therefore we need only prove that $k = 2^{n-1} \pm 2^{n/2-1}$.

Let A be the $2^n \times 2^n$ matrix with $f(x \oplus y)$ in row x and column y, and let J be the $2^n \times 2^n$ matrix of all 1s. Then we have $J^2 = 2^n J$ and $JA = AJ = kJ$, hence

$$2^n I \;=\; (J - 2A)^2 \;=\; 2^n J - 4kJ + 4A^2;$$

and it follows, when $y \neq 0 \ldots 0$, that the number of x such that $f(x) = 1$ and $f(x \oplus y) = 1$ is $k - 2^{n-2}$. In other words, there are exactly $k - 2^{n-2}$ ordered pairs of vectors (w, x) such that $f(w) = f(x) = 1$ and $w \oplus x = y$. Summing over all y gives $(2^n - 1)(k - 2^{n-2}) = k(k-1)$, and the solutions to this quadratic equation are $k = 2^{n-1} \pm 2^{n/2-1}$.

(d) By exercise 11, the term $x_1 \ldots x_r$ is present if and only if the equation $f(x_1, \ldots, x_r, 0, \ldots, 0) = 1$ has an odd number of solutions, and an equivalent condition

Just
Shmulevich
Konvalina
Shannon
Alekseyev
Bollobás
Brightwell
Leader
0–1 matrices
symmetric block design
Hadamard matrix

is $\left(\sum_{x_1,\ldots,x_r}(-1)^{f(x_1,\ldots,x_r,0,\ldots,0)}\right) \bmod 4 = 2$. We've seen in part (c) that this sum is

$$2^{-n}\sum_{x_1,\ldots,x_r,y} s(y)(-1)^{x_1y_1+\cdots+x_ry_r} \;=\; 2^{r-n}\sum_{y_{r+1},\ldots,y_n} s(0,\ldots,0,y_{r+1},\ldots,y_n).$$

If $r=n$, the latter sum is $\pm 2^{n/2}$; otherwise it contains an even number of summands, each of which is $\pm 2^{r-n/2}$. So the result is a multiple of 4.

Rothaus
Dillon
Patterson
Wiedemann

[Bent functions were introduced by O. S. Rothaus in 1966; his privately circulated paper was eventually published in *J. Combinatorial Theory* **A20** (1976), 300–305. J. F. Dillon, *Congressus Numerantium* **14** (1975), 237–249, discovered additional families of bent functions, and many other examples have subsequently been found when $n \ge 8$ and n is even. Bent functions don't exist when n is odd, but a function like $g(x_1,\ldots,x_{n-1}) \oplus x_n \wedge h(x_1,\ldots,x_{n-1})$ has distance $2^{n-1}-2^{(n-3)/2}$ from all affine functions when g and h are bent. A better construction for the case $n=15$ was found by N. J. Patterson and D. H. Wiedemann, *IEEE Transactions* **IT-29** (1983), 354–356, **IT-36** (1990), 443, achieving distance $2^{14}-108$.]

133. Let $p_k = 1/(2^{2^{n-k}}+1)$, so that $\bar p_k = 2^{2^{n-k}}/(2^{2^{n-k}}+1)$. [Ph.D. thesis (MIT, 1994).]

INDEX AND GLOSSARY

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.