9-1-1992

A boolean calculus and its application to the simplification of boolean functions.

C.T. Zahn

SCHOOL OF COMPUTER SCIENCE AND INFORMATION SYSTEMS

TECHNICAL REPORT

Number 54, September 1992

A Boolean Calculus

and

Its Application to the Simplification of Boolean Functions

C. T. Zahn

Department of Computer Science
Pace University
1 Martine Avenue
White Plains, NY 10606

≈ ≈ ≈ ≈ ≈ ≈ ≈ ≈ also ≈ ≈ ≈ ≈ ≈ ≈ ≈ ≈

Rio: A Missed Opportunity

Dietrich Fischer

Department of Computer Science
Pace University
1 Martine Avenue
White Plains, NY 10606

PACE
UNIVERSITY

We kick-off the new academic year with two distinguished papers.

Carroll's paper is his first write-up what appears to be a major discovery. I hope its appearance here generates some local excitement and brings him the critique, commentary, and informed praise so needed to nurture an intellectual creation.

Dietrich's paper should remind us that we, as individuals of learning, are responsible to work for social betterment in general. That same perspicacity enabling us to make technical and scientific contributions provides the acuity for effective participation in addressing the problems of our society and the world. (The last Technical Report from Dietrich was number 47, in December 1991, which was a reprint of his article from Parallel Computing "On Superlinear Speedups.")

≈≈≈≈≈≈≈≈≈≈

Charles T. Zahn, a Professor of Computer Science at Pace University, was the last chairperson of Pace-Westchester's CS Department. Among his publications, which number over fifteen, is C Notes, which was one of the earliest guides to programming in C.

His academic background includes eleven years at Stanford University as a research computer scientist specializing in pattern recognition, graph theory, programming methodology, and language design. While there he was also a lecturer in the Computer Science Department. In addition, Professor Zahn spent two years as a visiting scientist at the CERN Laboratory in Geneva.

His industrial background includes work for the General Electric Company, Yourdon Inc., the Mobil Corporation, and Advanced Computer Techniques as well as consulting for corporations and public agencies too numerous to name.

≈≈≈≈≈≈≈≈≈≈

Dietrich Fischer, a Professor of Computer Science in the Westchester Computer Science Department, earned his doctorate in computer science at New York University's Courant Institute.

He is an active researcher, writer, and speaker on international security and world peace. His three books are:

- Preventing War in the Nuclear Age, which was hailed as "perhaps the most important book ever written about nuclear war" when the arms race was still in high gear the Soviet Union was a feared military adversary
- Warfare and Welfare, with Nobel laureate Jan Tinbergen
- Winning Peace: Strategies and Ethics for a Nuclear-Free World, with N. Nolte and J. Oberg

A Boolean Calculus and its Application to the Simplification of Boolean Functions

by C. T. Zahn

Abstract

Quine's original hope to find a simplest formula for a boolean function without generating all rows of a truth table has been realized by an algorithm which obtains all prime implicants from an arbitrary sum-of-products formula. The algorithm is based on a calculus for representing and manipulating boolean faces in an N-cube corresponding to simple product formulas. A key concept of this theory is the joining face that spans the edges between two faces that have a Hamming distance of one.

Although the results are preliminary at this time, it appears that the information generated by the search for all prime implicants (maximal faces) is exactly what is needed to adapt the Quine/McCluskey minimal set covering algorithm without having to deal with the individual points.

The time complexity of the algorithm to find all prime implicants seems to be roughly proportional to the square of the number of prime implicants and independent of the number of variables.

An improved bottom-up version of the Quine/McCluskey method is described which avoids searches and the duplicate face discovery that occurs in standard Quine/McCluskey.

Unate functions and a separability criterion introduced by Quine are also discussed vis a vis the new approach.

Contents

9. Analysis of Unate functions.

10. Problem Separability based on FIJG connectivity or disjoint letter sets.

1. Introduction to Boolean Calculus

Every boolean function can be expressed in sum of products (SOP) form (also known as disjunctive normal form). Using the distributive law and DeMorgan to flatten out the nesting and push negations down to the letters themselves we can reduce f to

$$f = t_1 + t_2 + \ldots + t_n$$

where each $t_k = l_1 \cdot l_2 \cdot \cdot \cdot l_m$

and each l_j is a positive letter like A or a negated letter like $\bar{A}$ (We will use lower case a to represent $\bar{A}$)

Example : $a\ (B + c) + C\ \overline{(A + D)}$
$= aB + ac + Cad$

Any boolean function can be expanded from SOP form to one in which every term t_k involves <u>all</u> letters involved in f. Such terms are called <u>minterms</u>. Each term t missing a letter like X is replaced by the equivalent tX + tx and this process is repeated until all remaining terms are minterms.

Example : aB = aBC + aBc
= aBCD + aBCd + aBcD + aBcd

Problem : Interpret the above in terms of truth tables and show the relationships that obtain.

In defining the minimal cost SOP form of a function the cost of each term is the number of letters and terms appearing in it. There is not always a unique such minimal SOP for f, but there is a unique minimal cost and at least one representative SOP. The total cost of an SOP is, of course, the sum of the costs of its terms.

Example : aB + ac + Cad has cost 3 + 3 + 4 = 10

Problem : What is the cost of the minterm SOP equivalent to aB + ac + Cad ?

Starting with any boolean function f we can express f in SOP form and then minterm SOP form . We can then look for all formulas ϕ that are single terms, imply f and such that no ϕ' exists with $\phi \Rightarrow \phi' \Rightarrow f$. In other words, dropping any one letter from ϕ results in a term that does not imply f. These terms are called the prime implicants of the function and it is not hard to prove that any minimal SOP for f consists of a sum of prime implicants.

Quine (1952) and McCluskey (1956) devised a systematic method to determine a minimal SOP. The function is transformed to minterms and then the uniting theorem

$$\Phi a + \Phi A = \Phi$$

is used to combine pairs of minterms into terms with one letter missing. These are then combined to form terms missing two letters etc. until no more combinations are possible. This rather exhaustive process leads to the set of all prime implicants of the original function.

The second part of the solution involves the selection of a subset of these prime implicants that is implied by f and has minimum total cost. A two dimensional table is constructed with each row corresonding to a prime implicant and each column to a minterm.

	t_1	t_2	t_3	..	t_5		t_m
pi1			X				
pi2	X		X	..	X		X
pi3	X		X		X		
.							
pin		X					

an X means that pi "covers" t_k.

The principal techniques for selection are :

1) a column with one X means the row is <u>essential</u> since its pi uniquely covers the t (t_2 above ⇒ pin will be selected)

2) a row all of whose Xs are covered by those of a second row is <u>dominated</u> and can be eliminated from consideration since it will always be as cost effective to choose the pi corresponding to the dominant row rather than the dominated one (so long as its cost is no greater).
(pi3 is dominated by pi2 above)

3) a column which <u>dominates</u> another can be dropped from consideration since coverage of the dominated t will ensure coverage of the dominating t. ($t_3 \gg t_5$ above)

4) otherwise backtracking on the smallest column etc.

To understand these methods better and to see how Quine/McCluskey actually calculate the prime implicants we introduce the N dimensional hypercube and show how the boolean functions of N letter variables, minterms, SOPs and general terms are interpreted geometrically. The new boolean calculus we shall present relies almost entirely on this N-cubes view of a boolean function and its geometry.

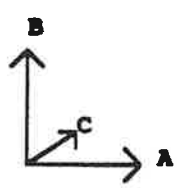
Rather than jumping immediately into general N-cubes we start with the 3-cube which represents all 8 points in 3-dimensional space with coordinates 0 or 1. If we consider 1 as true, 0 as false then these 8 triples of bits correspond to the 8 rows of a truth table for any boolean function of 3 variables. We illustrate the basic concepts with an example f = ac + BC + abC .
If we expand this to minterms we get

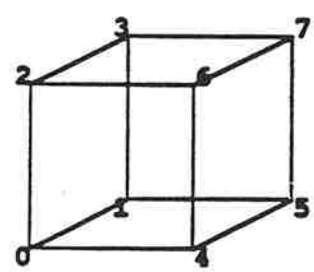
f = abc + aBc + aBC + ABC + abC

corresponding to the truth table

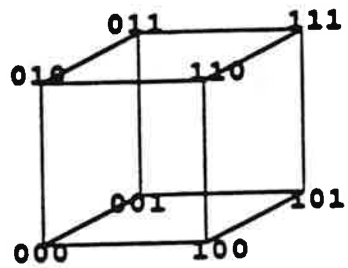
	A	B	C	f	letter minterms
0)	0	0	0	1	abc
1)	0	0	1	1	abC
2)	0	1	0	1	aBc
3)	0	1	1	1	aBC
4)	1	0	0	0	Abc
5)	1	0	1	0	AbC
6)	1	1	0	0	ABc
7)	1	1	1	1	ABC

It is standard practice to label the truth table with increasing integers starting at zero and writing the binary equivalent for the truth values as shown. Because of this correspondence any boolean function of N variables is equivalent to a subset of binary strings of length N (or alternatively a subset of integers in the range 0 .. 2^N - 1). Here is the 3-cube with its nodes labeled 0..7

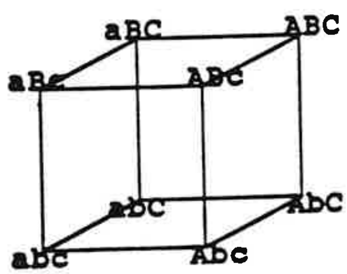




Two nodes are connected by an edge if and only if their binary representations vary in exactly one bit position. We see this most readily by labeling the nodes by the binary versions:



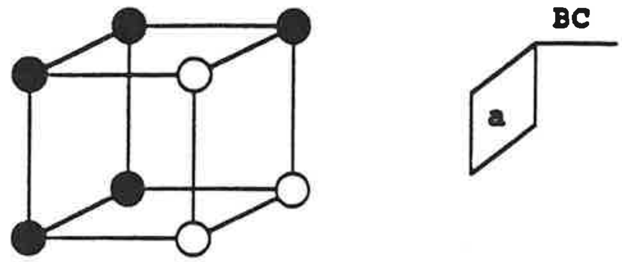
Remembering that each bit string corresponds to a minterm letter formula we can redraw the 3-cube as



Now we can represent the function:

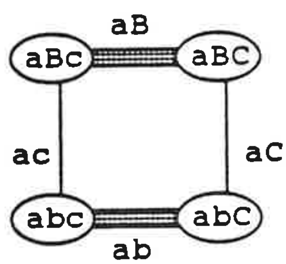
ac + BC + abC = abc + aBc + aBC + ABC + abC

by the 3-cube with solid nodes for the 5 minterms



The most important thing to notice here is that the 8 nodes correspond to 3-letter minterms, the 12 edges to 2 letter terms, the 6 faces to single letter terms and the entire 3-cube to the constantly true term 1. The 5 points of our simple function can be grouped into two terms a and BC corresponding to a face and an edge as shown.

Also notice that the face a consists of opposite parallel edges in two ways

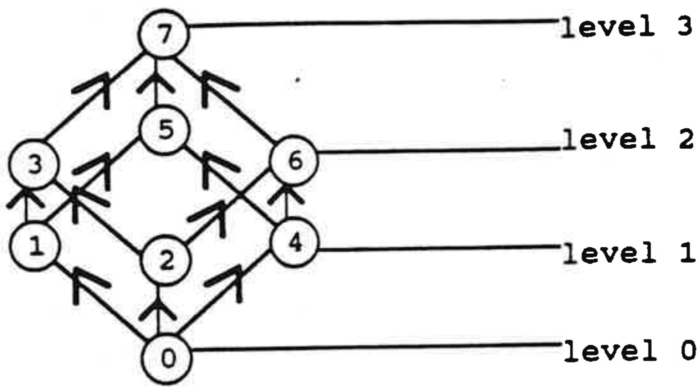


Each edge is the sum of its end nodes via the uniting theorem. For example aBC + abC = aC. The entire face is the sum of a pair of parallel edges via the same theorem. For example,

a = ac + aC = aB + ab.

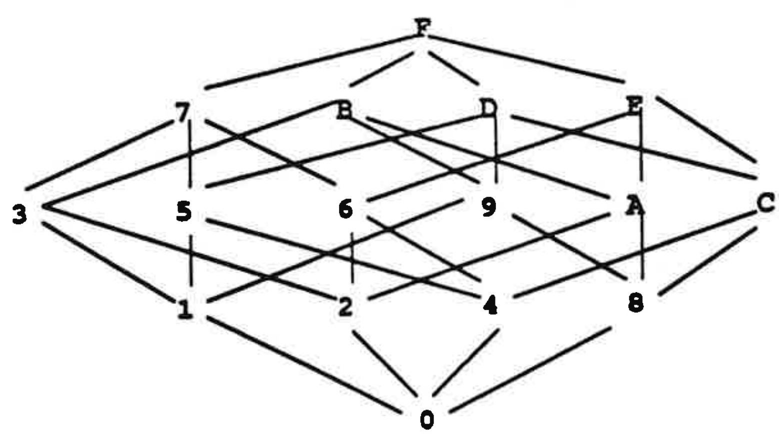
These concepts are not limited to 3-dimensional space. They apply uniformly to boolean functions of any number of variables. It is easier to "see" the geometry, however, in the easily visualized 3 space.

Another way to depict the 3-cube as a directed acyclic graph is shown here with decimal node labels



The levels correspond to the count of 1 bits in the node's binary representation and edges are directed toward the larger numbered node. It is no accident that the population of levels follows the binary coefficients 1,3,3,1.

Here is the 4-cube using hexadecimal labels (nibbles) for the 4-bit strings



We might call this an <u>ordered</u> <u>multipartite</u> <u>directed</u> <u>graph</u>, meaning that the node set can be partitioned into an ordered family of subsets $(S_0, S_2, \ldots, S_N)$ with all edges directed from S_k to S_{k+1}.

No edges enter S_0 or exit S_N. There is more structure than this implies and we will soon see it.

Quine/McCluskey exploits the levels to cut down on pairs that must be tested for being "united" into a larger face. Notice that larger faces correspond to terms with fewer letters and, hence, lower cost.

Problem : Analyze the cost trade-off involved in an application of the uniting theorem

The first step in Quine/McCluskey is to array the points as bitstrings segregated by levels as depicted below

level	points
0	000
1	001 010
2	011
3	111

Rather than checking all $\binom{5}{2}$ = 10 pairs of points for a potential edge, we may simply try pairs in adjacent levels (there are only 5). It turns out that all 5 are actually edges in this function.

The edges are commonly denoted in a ternary notation with X representing a position for a letter missing from the term.

Here are the 5 pairs and the resulting edges:

000	001	→	00X
000	010	→	0X0
001	011	→	0X1
010	011	→	01X
011	111	→	X11

All points participated in at least one edge so they are all marked as non-maximal. The edges are still arrayed in levels and the next round of 2-d face construction can use the same restriction to test the 6 pairs of edges in adjacent levels. Here we are looking for ternary strings that are identical except in one position where they differ (0 vs. 1). We obtain

00X	01X	→	0XX
0X0	0X1	→	0XX

Notice that we obtain the face a (≡ 0XX) in both ways, corresponding to the two pairs of opposite parallel edges depicted earlier.

We thus get edge BC (≡ X11) and face a (≡ 0XX) as the prime implicants (maximal faces) for f.

The covering table is

	abc	aBc	abC	aBC	ABC	Essential
a	⊗	X	X	X		√
BC				X	⊗	√

and circled ⊗s indicate unique covers establishing essential faces. Both faces are essential and cover all columns so the minimal SOP is a + BC

Problem : Find a minimal SOP for the function of 3 variables that is false for abc and ABC, true at the other 6 points.

2. Boolean calculus

We begin our presentation of the boolean calculus by defining our representation of an arbitrary face of the N-cube. Any such face F can be represented uniquely by a base point and an internal structure. We write F = b(Δ) where both b and Δ are N-bit strings and $b \wedge \Delta = \hat{0}$. The $\wedge$ operation is vector "and" bit by bit and $\hat{0}$ is the all zeroes bit string. The string b is the binary form of the closest point to $\hat{0}$ in face F and the bits of Δ represent the vector directions of the edges of face F. If p and q are neighboring points (nodes) of an N-cube then the vector direction for the edge joining p and q is just $p \oplus q$. The number of 1 bits in Δ gives the dimension of the face.

Returning to an earlier example we illustrate these definitions:

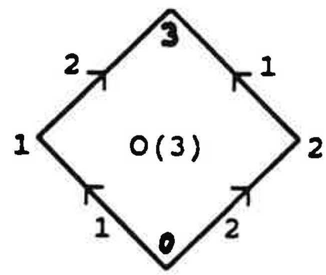
The function f = ac + BC + abC consists of three faces with the corresponding b(Δ) forms

ac : 000(010) ≡ 0(2)
BC : 011(100) ≡ 3(4)
abC : 001(000) ≡ 1(0)

Face ac consists of two points abc and aBc or in binary 000 and

010. Hence the face contains a single edge whose direction is 000 ⊕ 010 = 010. Clearly 000 is the base since it is δ. Face abC is a single point and hence has zero internal structure.

A more instructive example is the face "a" which is represented by 000(011) ≡ 0(3). It can be depicted as a subgraph of the 3-cube:



Notice that this face has 4 edges, 2 with direction 1, and 2 with direction 2. The internal structure is 3, indicating a 2 dimensional face.

The b(Δ) format separates the shape and size of a face from its base. As a result we can easily detect when two edges combine to form a face. The edges 0(2) and 1(2) in northeasterly direction above have identical internal structure and we call such faces parallel. since the base nodes 0 and 1 form an edge we may conclude that the two edges combine to form the face 0(3). The base is the smaller of the two bases (0 < 1) and the internal structure is gotten by including 0 ⊕ 1 = 1 with the common edge structure, 2.

It is easy to check whether a point p is contained in face b(Δ). The answer is yes precisely when $p \wedge \overline{\Delta} = b$. For example, with p = 7, b(Δ) = 0(3) we get

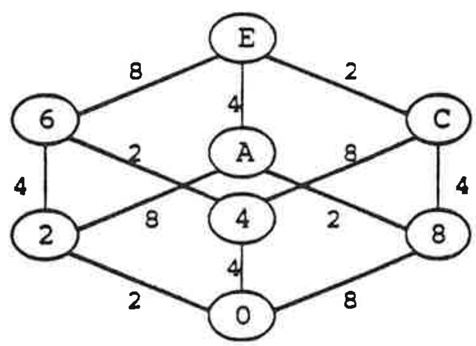
$$p \wedge \overline{\Delta} = 7 \wedge \overline{3} = 111 \wedge \overline{011} = 111 \wedge 100 = 100$$

but b = 0 = 000 so the answer is no.
If p = 1 we get

$$p \wedge \overline{\Delta} = 001 \wedge 100 = 000 = b \text{ so } \underline{\text{yes}}.$$

In the 4-cube (see Introduction) with hexadecimal labels we have, for example, the 3-cube face 0(E) consisting of points 0,2,4,8,6,A,C,E. We shall usually give points of a face from base to top through ascending levels with points in numerical order within a level. The structure vector, $\Delta = E_{16} = 1110_2$ has 1 bits at positions corresponding to $2^3 = 8$, $2^2 = 4$, and $2^1 = 2$, precisely the bits corresponding to the vector directions from any of the 8 points to its 3 neighbors in the 3-cube.

Let's redraw this 3-cube with edge vectors shown. Notice that all edges go from a level to the level immediately above and if p is below its neighbor q, then the direction vector is given by either $p \oplus q$ or alternatively $q - p$.



Notice, for example, that edge (4, C) has direction 8 since

$$4 \oplus C = 0100_2 \oplus 1100_2 = 1000_2 = 8$$

but that $C_{16} - 4_{16} = 12 - 4 = 8$ also.

Edges with the same direction vector are indeed parallel as we have been calling them.

A fairly obvious question is "What is the relationship between an arbitrary pair of faces in an N-cube ?". The structure of an N-cube has such an overwhelming symmetry that it is natural to expect that intersecting faces will form a face rather than some irregularly shaped set of points. Examples in the 3-cube do nothing to damage this hope. Indeed, it is generally true but more is true, as we shall soon see.

Any two points of an N-cube have a so-called Hamming Distance HD(p,q) which is the shortest path length connecting p to q in the graph of the N-cube. If p and q are represented by bitstrings then

$$HD(p,q) = \beta(p \oplus q)$$

where $\beta(s)$ is the bitcount of bitstring s, the total number of its 1 bits.

[This is the same Hamming Distance that plays such an important role in the error detection and error correction schemes, especially Hamming codes]

In the 4-cube HD(1,E) = 4 since

$$1 \oplus E = 0001 \oplus 1110 = 1111$$

which has a bitcount of 4. Searching for a shortest path visually is somewhat more difficult. One such path, (1, 0, 2, 6, E) might be considered as canonical since it proceeds downwards to $1 \wedge E$ and then upwards to E, smaller vectors first.

We can now define the Hamming Distance between any pair of faces as the smallest HD between two points one from each face. This is exactly analogous to the usual definition of the distance between two point sets.

In the 4-cube the two edges (7,F) and (0,8) are at distance 3 which is not so easy to determine visually, although the levels tell us it cannot be less than 2. We can exhaustively check all 4 point pairs

$$7 \oplus 0 = 0111 \oplus 0000 = 0111 \rightarrow 3$$
$$7 \oplus 8 = 0111 \oplus 1000 = 1111 \rightarrow 4$$
$$F \oplus 0 = 1111 \oplus 0000 = 1111 \rightarrow 4$$
$$F \oplus 8 = 1111 \oplus 1000 = 0111 \rightarrow 3$$

and see the minimum is 3. What do we do if we need to find the distance between two 10-cubes in 16-space? It would certainly be convenient if we could finesse the $(2^{10} \cdot 2^{10})/2 \approx .5 \cdot 10^6$ point to point comparisons. Happily we can and at a cost of about 5 bitstring operations readily available on typical computers. The one not available is bitcount which we come back to later.

Let $F_1 = b_1(\Delta_1)$ and $F_2 = b_2(\Delta_2)$ be any two faces in an N-cube. We don't exclude $F_1 = F_2$; also, either or both of F_k may be points or the entire N-cube. Two points p_1 and p_2 can be calculated, representing a closest approach between F_1 and F_2. That is, p_1 and p_2 are at the minimum distance $HD(F_1, F_2)$.

The formulas for the p_k are similar,

$$p_1 = b_1 + \Delta_1 \wedge b_2 \qquad\qquad p_2 = b_2 + \Delta_2 \wedge b_1$$

where "+" can be implemented as $\vee$, $\oplus$ or arithmetic +(!)
The Hamming Distance is then just

$$HD\ (F_1,\ F_2) = \beta(p_1 \oplus p_2)$$

A Hamming Distance of zero, of course, means that F_1 and F_2 intersect and in that case the points common to F_1 and F_2 form a face whose representation $b(\Delta)$ is given by the following simple calculation :

$$b(\Delta)_{F1\cap F2} = p_1 \ (\Delta_1 \wedge \Delta_2)$$

[Note that $HD(F_1, F_2) = \beta(p_1 \oplus p_2) = 0$ implies that $p_1 \oplus p_2 = \delta$ and therefore $p_1 = p_2$]

Thus the p_k are both equal to the common point of F_1 and F_2 closest to the origin point δ.

It is time to look at some examples. First, the edges (7,F) and (0,8) that we claimed were at a distance of 3. These faces have $b(\Delta)$ form 7(8) and 0(8) respectively so

$$p_1 = 7 = 8 \wedge 0 = 7 \qquad\qquad p_2 = 0 + 8 \wedge 7 = 0$$

Then we get $p_1 \oplus p_2 = 7 \oplus 0 = 7$ so that

$$HD\ (7(8),\ 0(8)) = \beta(p_1 \oplus p_2) = \beta(0111) = 3.$$

A second example, whose faces intersect is the 3-cube face 0(7) and the 3-cube face 4(B). These faces have the points

$$0(7) \equiv (0,\ 1,\ 2,\ 4,\ 3,\ 5,\ 6,\ 7)$$

and

$$4(B) \equiv (4,\ 5,\ 6,\ C,\ 7,\ D,\ E,\ F)$$

We can see that the faces intersect in the 4 points (4, 5, 6, 7) and can readily check that this forms the face 4(3).

Now let's do it the calculus way !

$$p_1 = 0 + 7 \wedge 4 = 7 \wedge 4 = 0111 \wedge 0100 = 0100 = 4$$
$$p_2 = 4 + B \wedge 0 = 4$$

So $HD(0(7),\ 4(B)) = \beta(4 \oplus 4) = 0$ and the intersection face is

$$b(\Delta)_{0(7) \cap 4(B)} = 4(7 \wedge B) = 4(0111 \wedge 1011) = 4(3)$$

as we suspected from doing it the long way.

It is always a good practice to check formulas in simple or degenerate cases so let us calculate the distance from point 1 to point E. They have face representations 1(0) and E(0) so we get

$$p_1 = 1 + 0 \wedge E = 1 \qquad p_2 = E + 0 \wedge 1 = E$$

and so the Hamming Distance for point faces is just the Hamming Distance for the points

$$\beta(p_1 \oplus p_2) = \beta(1 \oplus E) = 4$$

Problem : Show that the point in face test is a special case of the face comparison calculation.

Problem : Try calculating the intersection between two 3-cubes in 4-space, both based at the origin and spanning levels 0..3. Can you prove that all choices lead to a quadrilateral face (i.e. 2-dimensional) ?

We know that $F_1 \cap F_2 \neq \emptyset$ has four possible subcases:

A) $F_1 = F_2$

B) $F_1 \subset F_2$

C) $F_2 \subset F_1$

D) $F_1 - F_2 \neq \emptyset$ and $F_2 - F_1 \neq \emptyset$

Can we determine which of these cases holds with simple calculations as before ? Once we know that $F_1 \cap F_2 \neq \emptyset$, case D is assured by the negation of the first 3. Otherwise, using the obvious test for equality ($b_1 = b_2$ and $\Delta_1 = \Delta_2$) and the following test for $\subseteq$, we can resolve the first 3 cases.

$$F_1 \subseteq F_2 \quad \underline{\text{iff}} \quad \Delta_1 \subseteq \Delta_2$$

Where $\subseteq$ for bitstrings means that the set of positions of 1 bits in Δ_1 is a subset of the set of positions of 1 bits in Δ_2. This is equivalent to the following calculation

$$\Delta_1 \wedge \overline{\Delta}_2 = \hat{0}$$

or equivalently

$$\Delta_1 \wedge \Delta_2 = \Delta_1 \quad , \text{ or } \Delta_1 \vee \Delta_2 = \Delta_2.$$

Let's try the example $F_1 = 2(5)$ and $F_2 = 0(7)$.

$p_1 = 2 + 5 \wedge 0 = 2 \qquad p_2 = 0 + 7 \wedge 2 = 2$
$p_1 \oplus p_2 = 0$ so $HD = 0$ and
$b(\Delta)_{F1 \cap F2} = 2(5 \wedge 7) = 2(5)$

We can clearly detect $F_1 \subseteq F_2$ by $F_1 \cap F_2 = F_1$ and similarly for F_2.

We could have tested $\Delta_1 \wedge \overline{\Delta}_2 = 5 \wedge \overline{7} = 5 \wedge 8 = \hat{0}$
since $0101 \wedge 1000 = 0000$.

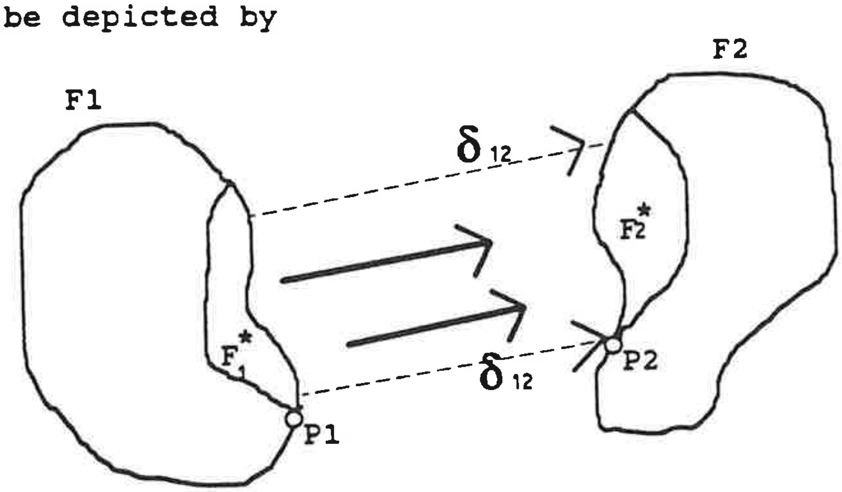
We also get all 4 subcases of $F_1 \cap F_2 \neq \emptyset$ immediately by the following decision tree (truth table) :

$F_1 \cap F_2 = F_1$?	$F_1 \cap F_2 = F_2$?	Case
false	false	D
false	true	C
true	false	B
true	true	A

Intersections and face containments are important since they correspond to logical operations, "and" and implication, respectively. However, we have found that the key concept missing from earlier work on boolean formulas and their manipulation is what we call the <u>join</u> between two faces at Hamming Distance 1. So here goes !

If $HD(F_1 , F_2) = 1$ then $\beta(p_1 \oplus p_2) = 1$ so there is at least one edge joining a point of F_1 and a point of F_2, namely (p_1 , p_2). There may be more single edges between the two faces but if there are more then they all go in the same direction and they all have the same direction vector $(p_1 \oplus p_2) = \delta_{12}$. Furthermore, the points of F_1 that are end points of such bridging edges form a subface of F_1 (call it F_1^*) and similarly for F_2 whose subface we call F_2^* . Somewhat surprisingly, F_1^* and F_2^* turn out to be parallel faces of the same size, shape, orientation and p_1, p_2 are their respective bases. Since these bases are neighbors, the points of F_1^* and F_2^* combine to form a single face F_1 <u>join</u> F_2 (denoted $F_1 \mid F_2$) whose base is the smaller of p_1 , p_2 and whose internal structure consists of the common structure of F_1 and F_2 along with the bridging vector $\delta_{12} = p_1 \oplus p_2$.

This can be depicted by



$$F_1 \mid F_2 = \min(p_1, p_2)\ (\Delta_1 \wedge \Delta_2 \vee \delta_{12})$$

In case F_1^* is all of F_1 and F_2^* is all of F2 our join subsumes both original faces

$$F_1 \subset F_1 \mid F_2 \quad \text{and} \quad F_2 \subset F_1 \mid F_2$$

and this symmetric join is the basis for the Quine/McCluskey generation of all maximal faces (aka prime implicants). The theoretical basis for this is the well-known Uniting Theorem of Boolean Algebra.

We shall see later that the more general join of arbitrary faces at distance 1 is related to the so-called Consensus Theorem of Boolean Algebra:

$$X \cdot Y + \overline{X} \cdot Z = X \cdot Y + \overline{X} \cdot Z + Y \cdot Z$$

in the cases where X is a single letter and Y and Z are not contradictory (i.e. do not differ vis a vis any letter). In any case $Y \cdot Z = 0$ implies that we learn absolutely nothing from the consensus theorem. The formulas $X \cdot Y$ and $\overline{X} \cdot Z$ represent two faces whose join is the extra term $Y \cdot Z$.

Let's investigate some examples of joins. First we try $F_1 = 0(3)$, $F_2 = A(5)$ in 4-space.

$$p_1 = 0 + 3 \wedge A = 2 \qquad p_2 = A + 5 \wedge 0 = A$$

$$p_1 \oplus p_2 = 2 \oplus A = 8 \quad \text{so that}$$

$$HD = 1$$

$$F_1 \mid F_2 = \min(2, A)\ (3 \wedge 5 \vee 8)$$

$$= 2(9)$$

$$= (2, 3, A, B)$$

Another example is $F_1 = 0(C)$, $F_2 = 2(5)$.

$$p_1 = 0 + C \wedge 2 = 0 \qquad p_2 = 2 + 5 \wedge 0 = 2$$

$\delta_{12} = p_1 \oplus p_2 = 2$ so $HD = 1$

and

$$F_1 \mid F_2 = \min(0,2)\ (C \wedge 5 + 2)$$
$$= 0(6)$$

As a check we shall do it the long way

$$F_1 = 0(C) = (0,\ 4,\ 8,\ C)$$
$$F_2 = 2(5) = (2,\ 3,\ 6,\ 7)$$

The 16 pairs compare as follows

p	q	HD	p ⊕ q	Bridge?
0	2	1	2	✓
	3	2	3	
	6	2	6	
	7	3	7	
4	2	2	6	
	3	3	7	
	6	1	2	✓
	7	2	3	
8	2	2	A	
	3	3	B	
	6	3	E	
	7	4	F	
C	2	3	E	
	3	4	F	
	6	2	A	
	7	3	B	

So the two bridging edges form face (0, 2, 4, 6) = 0(6).

Quine (1952) originally hoped to find the minimal sum of products formula for a boolean function given as a sum of products without exploding the formula into the totality of its minterms (i.e.

single points or rows of the truth table). To this end he defined two ways to incrementally shorten an SOP formula without changing the actual function. One flavor of shortening amounted to recognizing one of the terms as a face that could be united with a parallel face in the function and replacing the joinable face by the resulting join that is one letter shorter. This was a perfectly reasonable idea. Unfortunately, the second flavor of shortening involved dropping an entire term that is superfluous in the sense that after eliminating the term the reduced formula represents the same function as before. This term corresponds to a face covered by all the other faces. This was a bad idea for the reason that it leaves one with a formula covering all points but not all edges, admittedly a subtlety since ultimately we need only cover points.

Quine discovered, much to his dismay, a very small example in which an irreducible formula was not minimal. Indeed, the example shows that irreducible formulas may be missing some key prime implicants (maximal faces).

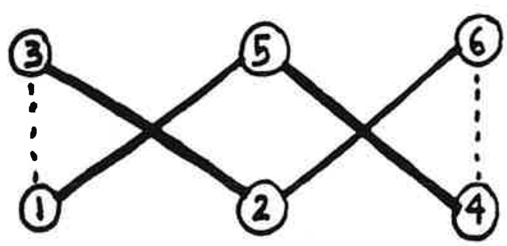
Quine's example in 3-d is

Ab + aB + Bc + bC

which corresponds to 4 edges in face notation:

4(1), 2(1), 2(4), 1(4)

and can be depicted as



There are two minimal SOP, each consisting of an alternating triple around the 6-cycle DAG of the function.

What is wrong here is that there are 6 maximal faces, the above 4 edges and the other 2 that are not in Quine's original formula

1(2) and 4(2)

Not only does Quine's reducibility operation fail to add in the two new terms, he actually could produce the 4 terms from all 6 by dropping the two.

What I noticed early on from this simple example is that all the missing maximal faces are faces that bridge between existing faces. I therefore struggled to understand those joining structures and in time came to the calculus above.

I had been developing, with considerable success, an improved version of the bottom-up Quine/McCluskey procedure for finding all maximal faces, but the idea of a top-down procedure that avoids the combinatorial explosion into minterms was, of course, to be greatly prefered. It is easy to write down formulas in 20 variables that involve hundreds of thousands of points but only 10 or 50 faces. There are several orders of magnitude difference between operating on points versus faces.

The key ingredient along with our face calculus is a small theorem that gives us a sufficient condition for a set of faces to include each and every maximal face of its function without any duplications or other non-maximal faces.

Theorem: A reduced* family of faces $\mathbb{F}$ representing a function f consists of the set of all maximal faces of f if every join between two faces of $\mathbb{F}$ is contained in some face of $\mathbb{F}$. (* no pair related by $\subseteq$.)

This leads to an efficient algorithm taking us from any set of faces covering all points of f to the set of maximal faces. The algorithm can be applied to the raw points but in that case I suspect our improved Quine/McCluskey may be better. There are some interesting data structure and heuristic issues involved in this algorithm but I consider its development to have been simple professional technique given the face calculus and the joins theorem.

Now to get back to the theory and show off a bit.
First try a little exercise:

<u>Problem</u>: See if you can work out the 2 maximal faces for the earlier problem, ac + BC + abC, using the join concept and the face calculus. Calculate the face to face intersections also for all pairs of maximal faces.

The set of all maximal faces for a function is a unique signature or canonical form. Many SOP formulas can represent the same boolean function (identical set of minterms or points) but they all have exactly the same maximal faces. Quine (1952) was aware of the potential usefulness of this canonical form. The nice thing is that the set of maximal faces may be of reasonable size when the point set is ridiculously large.

Note this minimal SOP is <u>not</u> a canonical form.

Equality of two SOP boolean formulas can now be determined by a systematic procedure - find the $MF(f_1)$ and $MF(f_2)$ and test for identity. This could be shortened by a sorting order for maximal faces or a hash code for quickly rejecting certain function pairs as unequal.

We can also test $f \rightarrow g$ represented by $\mathbb{F}$, $\mathbb{G}$ by calculating $\mathbb{G}^* = MF(\mathbb{G})$ and determining if every F in $\mathbb{F}$ is contained in some G in $\mathbb{G}^*$.

Another useful manipulation of boolean formulas is that of factoring, one of the two ways to apply the distributive law, $X \cdot (Y + Z) = X \cdot Y + X \cdot Z$.

Factoring in the context of face calculus has a somewhat different appearance. Given two faces X and Y, how large a common factor Z exists such that $Z \cdot (S + T) = X + Y$ and, of course, what exactly are Z, S, T?

An example might be X = AbC, Y = Abd with the solution Z = Ab, S = C, T =d.

If we denote the X and Y faces as $b_x(\Delta_x)$, $b_y(\Delta_y)$ then faces Z, S and T can be calculated in $b(\Delta)$ form by :

$$b_Z^{MAX} = \overline{\Delta_X \vee \Delta_Y \vee (b_X \oplus b_Y)}$$

$$b_Z = b_Z^{MAX} \wedge b_X = b_Z^{MAX} \wedge b_Y$$
$$\Delta_Z = \Delta_X \vee \Delta_Y$$

$$b_S = b_X \wedge \overline{b_Z^{MAX}}$$
$$\Delta_S = \Delta_X \vee b_Z^{MAX}$$

$$b_T = b_Y \vee \overline{b_Z^{MAX}}$$
$$\Delta_T = \Delta_Y \vee b_Z^{MAX}$$

Note that b_Z^{MAX} gives the bits corresponding to letters that are the same in X and Y.

Applying these formulas to the example AbC + Abd we obtain using

$$b_X \oplus b_Y = 1010 \oplus 1000 = 0010$$
$$\Delta_X \vee \Delta_Y = 0001 \vee 0010 = 0011$$

$$b_Z^{MAX} = \overline{0011 \vee 0010} = \overline{0011} = 1100 \quad \text{(A and B)}$$

$$\left.\begin{aligned} b_Z &= 1100 \wedge 1010 = 1000 \\ \Delta_Z &= 0011 \end{aligned}\right\} \quad 8(3) \equiv \text{Ab}$$

$$\left.\begin{aligned} b_S &= 1010 \wedge 0011 = 0010 \\ \Delta_S &= 0001 \vee 1100 = 1101 \end{aligned}\right\} \quad 2(D) \equiv \text{C}$$

$$\left.\begin{aligned} b_T &= 1000 \wedge 0011 = 0000 \\ \Delta_T &= 0010 \vee 1100 = 1110 \end{aligned}\right\} \quad 0(E) \equiv \text{d}$$

<u>Problem</u> : Work out the factoring of Ad + BcD and decide if you like it. Just for fun try Ab + Ab.

<u>Problem</u> : Try to <u>prove</u> that the calculations do the correct thing when X = Y.

It turns out that we can interpret Hamming Distance, intersection faces and joins in terms of letter formulas (terms).

The Hamming Distance between two products of letters is simply a count of the number of letters that appear in contradictory form in both formulas. For example, AbCd and abcd differ in the letters A and C so they are at Hamming Distance 2. In fact, with point faces this is quite obvious.

Two formulas are non-contradictory iff they differ in no letter. For example, AbC and Bd are contradictory, differing as they do in letter B. On the other hand, Abd and bCd are non-contradictory and their conjunction is AbCd gotten by concatenating the formulas and suppressing any duplicates.

their conjunction is AbCd gotten by concatenating the formulas and suppressing any duplicates.

In case two formulas are at distance 1 the consensus or joining face is gotten by concatenating the two formulas while suppressing any duplicates and eliminating all reference to the single differing letter. For example, HD(AbC, Bd) = 1 and

$$AbC \mid Bd = ACd$$

You can check all this using the calculus for faces.

It is natural to ask what logical negation looks like in our face calculus and it turns out there are two answers. First we point out that the negative of any formula f can be expressed as

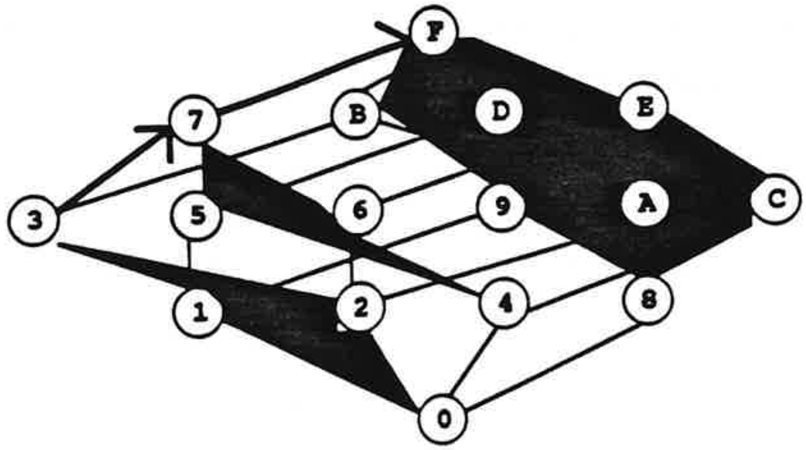
$$\overline{f} = \hat{1} \wedge \overline{f} = \hat{1} - f$$

So we shall develop a calculation (actually 2) for the difference of two faces assuming one is contained in the other.

Let F and G be faces of an N-cube with $G \subset F$. Let dim (F) = n and dim(G) = k. Then the points of F - G can be expressed as a sum of a sequence of increasingly larger faces forming a chain at Hamming Distance 1 from one another. The smallest face has dimension k and the largest has dimension n-1 so there are in all (n-k) faces. They partition F-G. In general, this partition is not unique.

We present an example in 4-space which represents the negation of ab, that is $\overline{ab}$.

In face notation we desire 0(F) - 0(3)



This picture shows that

$$0(F) - 0(3) = 4(3) + 8(7)$$

Starting from 0(3) we select any direction vector not in 0(3). We have chosen 4. The face 4(3) parallel to 0(3) and joinable to it is our first face. We form the symmetric join $0(3)|4(3) = 0(7)$ and repeat. This time there is but one choice and we add face 8(7) parallel to 0(7) joined via direction 8. This time the join is $\hat{1}$ so we stop.

DeMorgan's Law tells us that $\overline{ab} = A + B$ whereas the previous construction gave us

$$\overline{ab} = aB + A$$

The construction that represents DeMorgan's Law is the one that calculates $\hat{1} - 0(3)$ as the sum of two faces of dimension 3 based at the neighbors of 0 that are not in 0(3). These are the same two direction vectors we used in the earlier chain.

Hence

$$0(F) - 0(3) = 4(B) + 8(7)$$

This represents the maximal faces of the difference whereas the earlier form represents the most economical non-overlapping form. Both probably have their uses.

I just noticed that there is an easy way to partition F - G into faces exactly parallel to G and the same size, at least when G = $0(\Delta_G)$.

First calculate $(\Delta_F - \Delta_G) = \Delta^*$, the vector directions of the larger F not in the smaller G.

Then if $\beta(\Delta^*) = k$ we get exactly 2^k faces based at nodes of $b_G \wedge \overline{\Delta^*}$ (Δ^*) with internal structure Δ_G. One of these is Δ_G itself and the remaining $2^k - 1$ partition F - G. The base nodes for the 2^k faces are exactly the nodes of the unique face of dimension k, containing b_G and having internal structure Δ^*.

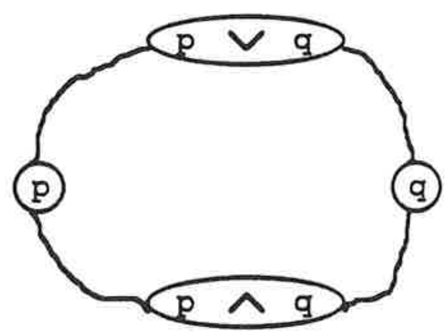
Note that $\beta(\Delta^*) = k = \dim(F) - \dim(G)$.

About the smallest face spanning two points

If p and q are points of the N-cube, then there is a smallest face containing both and this face $\phi(p,q)$ has internal structure $(p \oplus q)$ and base $p \wedge \overline{p \oplus q} = p - (p \oplus q)$

But $\overline{p \oplus q} = \overline{p\bar{q} + \bar{p}q} = (\bar{p} + q) \cdot (p + \bar{q})$

so $p \wedge \overline{p \oplus q} = p \cdot (\bar{p} + q) \cdot (p + \bar{q})$
$= p\overline{pq} + pqp$
$= p \wedge q$



[Note: there is a partition of $\hat{1}$ - 0(3) into 3 faces parallel to 0(3) -- 4(3), 8(3), C(3) and, in general, F - G can be partitioned into $(2^{n-k} - 1)$ faces parallel to G]

$$\phi(p,q) = p \wedge q \ (p \oplus q) \text{ with top} = p \vee q$$

Presumably we could prove $(p \vee q) \oplus (p \wedge q) = p \oplus q$.
We can use the previous formulas to obtain the maximal faces of a difference $(F - G)$ where $G \subseteq F$.

Take any point in G (say its base b_G) and compute its antipode in F(call it t) by:

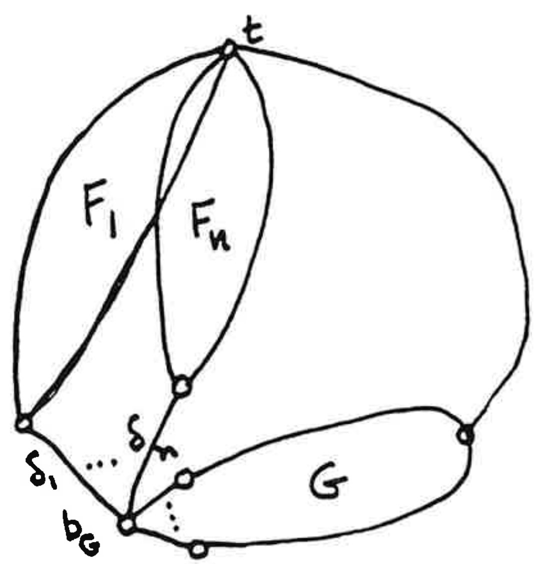
$$t = b_G \oplus \Delta_F$$

which inverts those and only those bits of b_G that are internal structure vectors of F.

For each direction vector δ_k in $\Delta_F - \Delta_G$ we construct the spanning face

$$F_k = \phi(b_G \oplus \delta_k, t)$$

as depicted below (n = dim(F) - dim(G))



We finish this section with two useful formulas for the manipulation of boolean faces.

It is useful in some circumstances to transform the coordinate system of the N-cube to one with a different origin. Because of the geometrical symmetry this does not change a function nor does it change any of its faces or maximal faces. It does change the base (delta) representation as follows. Any face $b(\Delta)$ will have the representation

$$(b \oplus \omega) \wedge \overline{\Delta} \; (\Delta)$$

in the N-cube with origin transformed to point ω. This shows that the internal structure of a face is invariant to coordinate transformation which is a pleasant if not surprising result.

The antipode p'_F of any point p in face F with respect to face F can be calculated by

$$p'_F = p \oplus \Delta_F$$

and, in particular, the antipode of p in the entire N-cube is

$$p^* = p \oplus \hat{1} = p \quad [\text{since N-cube} \equiv \hat{0}(\hat{1})]$$

which we knew already.

3. Improved Quine/McCluskey solution to the Maximal Faces determination

The proposed method builds from points to the set of all maximal faces in bottom-up fashion as does the standard Quine/McCluskey method. It begins exactly as the earlier method by joining points to obtain all edges (1-faces) of the given function exploiting bitcounts to control the search somewhat. It should be noted that edges can be constructed from their end points in exactly one way so there are no duplicates in this round of joining.

It is at this point that our method deviates from Quine/McCluskey. In subsequent joining stages we shall generate each face of the function exactly once, whereas the earlier method generated each k-face exactly k times and incurred some overhead to check the duplication. Furthermore, our method involves minimal searching after the first round collects all the edges.

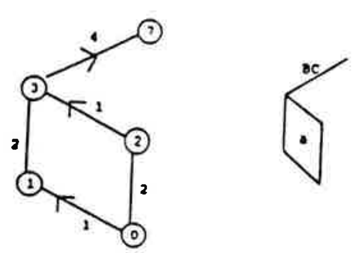
We exploit the DAG(directed acyclic graph) of a boolean function f by using the local information about each point that may be inferred from the points of the function that share an edge. The essence of the approach can be appreciated by looking at the simple 5-point function we used earlier

$$f = abc + aBc + aBC + ABC + abC = a + BC$$

The five points correspond to the numbers

$$f = \Sigma\ (0,\ 1,\ 2,\ 3,\ 7)$$

The DAG of f is a subgraph of the DAG of the 3-cube and is shown below with the vector directions of each edge:



Notice the following important facts that emerge from considerations of purely local graph structure at each node:

(1) Node 7 has degree 1 and therefore cannot participate in any face of dimension > 1.

(2) Node 1 has degree 2 and is connected to its 2 neighbors by direction vectors 1 and 2 which limits node 1 to at most a 2-face and in particulor to a 2-face with internal structure 1 + 2 = 3.

(3) Node 2 has degree 2 and belongs to maximal 2-face 0(3) which means that node 2 is uniquely covered by 0(3) among maximal faces since node 2 would require at least one more edge to belong to a second maximal face.

(4) The four points (0, 1, 2, 3) of face 0(3) all share the property that the set of adjacent edge directions contains both 1 and 2, the constituents of the internal structure of 0(3). Indeed this is a necessary condition.

(5) If we attribute a Thru structure to each point which is the sum of adjacent edge directions then we have

Point	Thru	UpTo	UpFrom
0	3	0	3
1	3	1	2
2	3	2	1
3	7	3	4
7	4	4	0

where we have separated out the incoming UpTo edge structure from the outgoing UpFrom edge structure. This tells us that point 0 is the base of two edges 0(1), 0(2) and that point 3 is the top of two edges 2(1) and 1(2) as well as the base of edge 3(4).

(6) If we define the Thru, UpTo and UpFrom structure of any face of the DAG in the appropriate way then we can immediately tell if the face is maximal or not and in the latter case how it can be extended. When nodes 1 and 3 form edge 1(2) we get

Thru = UpTo + Delta + UpFrom

or

3 = 1 + 2 + 0

the value of Thru showing that edge 1(2) can be at most in a 2-face with structure 3, the value of UpTo indicating the actual existence of a face extending 1(2) downwards via direction 1. UpTo records edge directions incoming to all (both) points of edge 1(2). Upfrom is defined similarly.

(7) The 5 edges with their relevant structure are

Edge	Delta	UpTo	UpFrom	Disposition
0(1)	1	0	2	Lower canonical
0(2)	2	0	1	Non-canonical
1(2)	2	1	0	Non-canonical
2(1)	1	2	0	Upper-canonical
3(4)	4	0	0	Maximal

This local edge information allows us to recognize each face in exactly one way. The non-canonical join 0(3) = 0(2) | 1(2) has UpFrom, UpTo respectively for 0(2), 1(2) both < Delta. On the other hand, the canonical join 0(3) = 0(1) | 2(1) has values of UpFrom, UpTo > Delta. We define the unique canonical join for any face to be created via the largest direction vector connecting parallel faces of smaller dimension whose internal parallel faces of smaller dimension whose internal structure consists of the remaining direction vectors. For example,

$$2(D) = 2(5) \mid A(5) \quad \text{via } 8$$

since $D \equiv 1101_2$ and 1000_2 is the largest vector and $0101_2 \equiv 5$ is the remainder.

The algorithm progresses by keeping all faces that are lower canonical joins in an active queue and holding all upper canonical joins in small lists associated with their base nodes. This should reduce the searching considerably. It does require an array of pointers of size 2^N or some form of hash coding to avoid the large array.

By keeping track of the minimum node degree of each face we can determine the essential and the inessential faces by testing the face dimension against the minimum degree. Equality means essentiality, inequality the opposite.

Formulas for calculating the structure of joins are:

$$\text{Thru}(F \mid G) = \text{Thru}(F) \wedge \text{Thru}(G)$$
$$\text{Delta}(F \mid G) = \text{Delta}(F) + (\text{Base}(F) \oplus \text{Base}(G))$$
$$\text{Base}(F \mid G) = \min(\text{Base}(F), \text{Base}(G))$$
$$\text{MinDegree}(F \mid G) = \min(\text{MinDegree}(F), \text{MinDegree}(G))$$

and the derived structures:

$$\text{UpFrom}(F) = (\text{Thru}(F) - \text{Delta}(F)) - \text{Base}(F)$$
$$\text{UpTo}(F) = \text{Thru}(F) \wedge \text{Base}(F)$$

When a face has Thru = Delta or equivalently, UpTo = UpFrom = 0 we have a Maximal Face and otherwise a non-maximal Face, so our stopping condition is quite direct.

4. A Top Down Algorithm to determine the Maximal Faces of a Boolean Function given in SOP form

This algorithm starts from a family of boolean faces represented as pairs b(d), each face being a product term of the desired function. It terminates with a family of faces, no pair satisfying $F \subseteq G$, and each face a maximal face of the function. In addition, a face intersection/join graph FIJG is calculated which has edges between each pair of maximal faces that intersect or are at Hamming Distance 1 apart.

The original faces are placed into a family called Try and moved one by one from there to an initially empty family called Good or else discarded. An invariant relationship governs this principal iteration until Try is finally empty. This relationship assures that the formula represented by all terms from Good and Try is always the function f given by the original Try family. Furthermore, the faces of Good contain no duplicates and no containments ($F_1 \subseteq F_2$). Finally, the property that makes everything work:

> Every face H that is a join between two faces of Good is subsumed by some face of either Good or Try.

When the algorithm terminates Good is a family of terms that represents the original function f and subsumes all its pairwise joins. A theorem stated earlier implies such a family is indeed precisely the set of Maximal Faces of f.

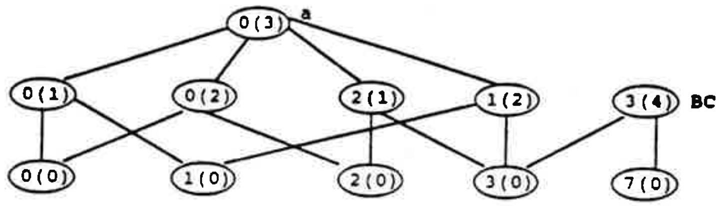
Each principal iteration step disposes of one face in Try by comparing it to all faces in Good, discarding it immediately as redundant if it is subsumed in a face of Good. If not discarded it enters Good causing deletion of some faces originally in Good. Its intersections and joins with faces of Good are recorded in the FIJG and the new join faces are added to Try if they are not subsumed in Good or Try.

The termination of this process is governed by a function which measures the size of Try plus the number of faces of f not yet subsumed in Good or Try. Each step starts by decreasing Try by one. Each new join face that is added to Try is one not subsumed by Good or Try since we check for these possibilities. Each addition to Try increases the size of Try and decreases the number of non-subsumed faces by at least one.
[Note: our last adjustment to this algorithm was the check for joins being subsumed in Try, added to make the algorithm conform to the termination proof.]

To restrict the set of faces in Good that must be compared to the candidate face E from Try we keep references to both parents F, G of each join H = F | G. If E = F_1 | F_2 is a join then the only faces of Good that might intersect or join with E are faces that are adjacent to F_1 or F_2 in the FIJG. This graph of intersections and joins was originally designed to make the maximal faces algorithm more efficient, as also the recording of parents for each join. Subsequently, we have found this information to be crucial for efficient solution of the minimum cover problem after the maximal faces have been found.

The termination argument actually gives a rather precise bound on the number of steps required by the primary iteration. The bound is the original number of faces plus the number of faces of the function not subsumed by the original faces. This tells us with some precision that starting from raw points will take quite a while but the method may converge rapidly if the original faces are "near maximal".

Returning to our previous little example we can depict the faces of f = a + BC as a partially ordered set based on set inclusion:



There are 11 faces in f including 5 points and the two maximal faces. These maximal faces are easily recognized as relative maximal in the poset.

If we start the maximal faces algorithm from the 5 points then the size of Try is 5 while the deficit of non-subsumed faces is 11 - 5 = 6 for a total of 11. On the other hand starting from the formula

$$f = abC + BC + ac = 1(0) + 3(4) + 0(2)$$

we have |Try| = 3 and a deficit of 4 since the only non-subsumed faces are 0(3), 0(1), 2(1), 1(2). Hence there will be no more than 7 steps.

Some remarks about the Hasse Diagram of the poset of faces of f is in order. The function f involves 3 letters so can be represented by a set of 5 points in the 3-cube which consists of $8 = 2^3$ points. The DAG of the 3-cube is isomorphic to the Hasse Diagram of the lattice of all 8 subsets of some 3 element set. The faces in the 3-cube are specially shaped subsets of the 8 points of the 3-cube and hence represent a portion of the lattice of all $2^8 = 256$ subsets of points of the 3-cube.

The Hasse Diagram shown above represents what is called an <u>order ideal</u> in lattice theory and its relative maximal elements are called a <u>chain</u> . In our context the maximal faces of f are a chain that generates the order ideal of all faces of f.

There is an upper bound to the size of Good when creating the maximal faces of a function. Since we allow no containments between faces of Good we can claim that |Good| is never larger than the largest chain in the poset of faces of the function. Unfortunately, we do not yet know any good way to use this in general. Actually we might be able to organize the ordering of candidates in such a way that |Good| would be bounded by the largest chain lying above the order ideal generated by the original faces.

In the small example f = a + BC, if we started from the faces abC + BC + ac (1(0), 3(4), 0(2)), then the largest chain that contains these three faces is the chain of all edges (0(1), 0(2), 2(1), 1(2), 3(4)). By lucky choice we can generate 1(2) along with 3(4), 0(2), giving us only 3 of these edges which is enough to get the final 2 maximal faces without ever generating the non-canonical joining pair 0(1) and 2(1).

The problem with using the above observation is that some functions seem wider at the bottom, others at the top and some in the middle. For example, the function A in 16 dimensions has 2^{15} = 32768 points (all faces of a single given size always constitutes a chain) but only 1 maximal face. On the other hand $S^9_{3..6}$ has 420 points and 1680 maximal faces of dimension 3. The monotonic function $S^9_{3..9}$ has 466 points, 84 maximal faces but still contains a chain larger than 1680 since the maximal faces of $S^9_{3..6}$ are faces of $S^9_{3..9}$.

Clearly the largest chain is at least as large as the total point count and as large as the number of maximal faces. It seems likely that the overwhelming majority of functions will be widest at the bottom or point level, the symmetric functions being anomalous. If we also expect most functions to be presented via a set of faces high up in the poset of all faces and if the functions tend to narrow toward the top then we have reason to expect our algorithm to be quite efficient.

According to the earlier termination argument the number of steps is bounded by the original size of Try plus the deficit. The work required per step is roughly bounded by the size of Good so the total effort seems bounded by

$$(|Try| + Deficit) \times (|MaxChain|)$$

For functions that narrow towards the top or at least do not get wider than the maximal faces it looks like the effort will be $O(NFM^2)$ if the original faces (i.e. Try) are close to the top.

This discussion suggests that we can make the method more efficient if we find ways to eat up the deficit more quickly (less steps) and minimize the size of the chain in Good (fewer comparisons per step).

5. Approaches to the Minimum Cover of a Boolean Function by its Maximal Faces without Points

The standard Quine/McCluskey procedure leading to a minimal cover via maximal faces uses a two-dimensional tableau with a row for each maximal face and a column for each point. We want to avoid the combinatorial explosion that this implies while still using the covering and face discarding techniques used in the older method .

The concept of essential face will be generalized to that of a necessary set -- a subset of maximal faces at least one of which must appear in any minimal cover. An essential face represents a singleton necessary set. Essential faces can be immediately incorporated as chosen just as is done by Quine/McCluskey. Backtracking uses a smallest necessary set.

The role of points will be played by the maximal faces in our "pointless" scheme. We will have to cover all these maximal faces using a subset of the maximal faces. For coverage information we use joins, both those implied by parent references and others that can be discovered fairly easily from the face intersection/join graph. If H = F | G we know that H is entirely covered by the union of F and G.

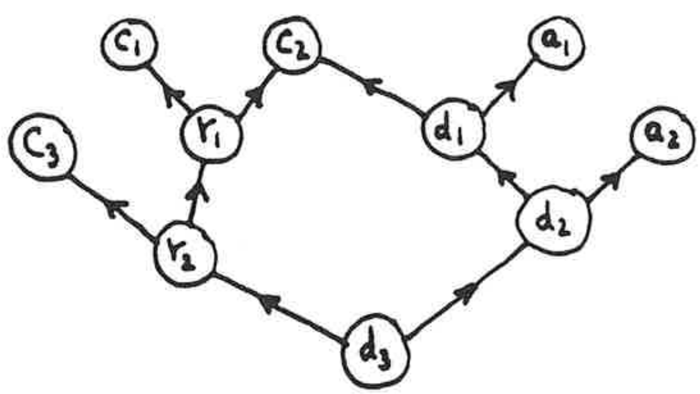
We introduce the concepts of dominance and redundancy, the first being the row dominance of Quine/McCluskey and the second indicating that a face is entirely covered by faces already chosen. In terms of joins we have

> H is redundant if H = F | G and F and G are each chosen or redundant

and

> H is dominated by G if H = F | G and F is chosen, redundant or dominated

There is a problem with dominance not encountered in the Quine/McCluskey method and that is the possibility of cycles in the redundancy dominance graph. The following graph depicts the interrelationship of several join based redundancies and dominances. Nodes are labelled c for chosen, r for redundant, d for dominated and a for still active.



This graph proves that r1, r2, d1, d2, d3 are covered by c1, c2, c3, a1, a2. It does not, however, assure minimality.

We must not allow any directed cycles in this graph but we have several tricks for assuring this without enormous effort!

There are indeed general methods apart from joins to determine that a face is covered by the union of two or more others, but we have found considerable success using joins alone.

Returning to the matter of essential faces and necessary sets we have the following techniques:

(A) A maximal F whose intersections with other faces $G_1, \ldots, G_k$ satisfies

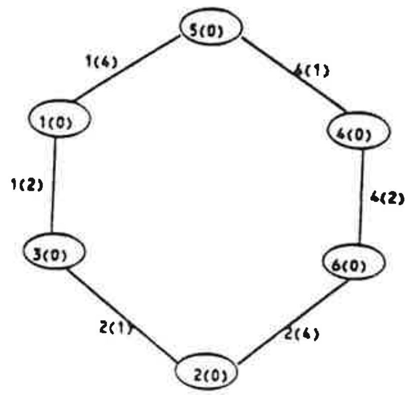
$$\sum_{i=1}^{k} |G_i \cap F| < |F|$$

is essential, since these intersections cover less points than are in face F .

(B) If face $G_0 = b(d)$ is such that b is contained in maximal faces $G_1, \ldots, G_k$ but no others then $(G_0, G_1, \ldots, G_k)$ is a necessary set. This can be repeated with top points (or any others, for that matter) to obtain more such sets.

When we are absolutely forced to use a more detailed analysis we can determine if F is entirely covered by $(G_1, \ldots, G_k)$ by calculating the maximal faces of the function whose faces are $\{G_i \cap F\}^k{}_1$ and if the answer is F itself we have coverage, otherwise no. This can be used to determine essentiality with certainty, whereas our other methods provide only sufficient conditions.

We can illustrate most of these ideas by solving the Quine 6-cycle problem defined by points (1, 2, 4, 3, 5, 6) in the 3-cube. This has 6 maximal faces and 6 points arranged in a cyclic fashion as shown below:



The face intersection graph has the above edges as its <u>nodes</u> and the points common to two edges are the intersecting faces. We see that each maximal face has 2 intersecting faces consisting of one point each so the overlap count is 1 + 1 = 2 just matching the size of the edge. We therefore have no essential faces.

Condition (B) can be used on the bases and tops to generate 6 necessary sets each of size 2. We show two such sets along with the relevant points covered:

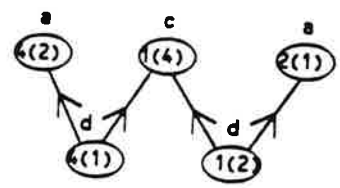
1(4) or 4(1) to cover 5(0)
4(1) or 4(2) to cover 4(0)

In addition we have the following join information since each edge on the 6-cycle is the join of its two neighboring edges:

1(4) = 1(2) | 4(1)
4(1) = 1(4) | 4(2)
4(2) = 4(1) | 2(4)
2(4) = 4(2) | 2(1)
2(1) = 2(4) | 1(2)
1(2) = 2(1) | 1(4)

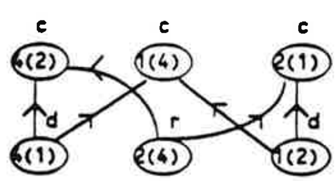
There are no essential faces so the best we can do is to try each of the faces from some necessary set. We choose face 1(4) from the first such set covering point 5(0). We must later try solutions involving 4(1).

With 1(4) chosen we can mark 4(1) dominated by 4(2) using the second join above and also can mark 1(2) dominated by 2(1) using the last join. This can be depicted by the dominance graph below:



There is no more we can do here without creating a cycle in the graph. The third join, for example, suggests that 4(2) is dominated by 2(4), but this would produce the cycle 4(1), 4(2), 4(1) in the dominance graph.

Although we can not discard any more faces at this time we can take advantage of the newly discovered dominated faces by recomputing overlap counts some of which will have decreased due to the disappearance of the discarded faces 4(1) and 1(2). In particular, faces 4(2) and 2(1) both now have a single overlapping face and an overlap count one less than their size and therefore are essential. The remaining face 2(4) = 4(2) | 2(1) is covered by the two new chosen faces and so is redundant giving us the dominance graph



All three discarded faces could now be marked redundant since there are no more active ones. We have thus shown that there is only one minimum cover candidate that contains the face 1(4).

In exploring the other alternatives we may assume that 1(4) is not used and 4(1) is . With 1(4) gone both its neighbors are essential so 4(1) and 1(2) are chosen, 2(1) and 4(2) are dominated via their joins and 2(4) is essential, having no overlap by chosen or active faces. We get a similar graph to that above. This shows that there are exactly two solutions. Note that we must generate them both and measure their costs before we can conclude they both are optimal.

Are there situations that would allow us to discard an active face that is a join of two dominated faces and therefore covered by the chosen faces and two other active faces?

The answer seems to be yes -- if the residual points of face F not yet covered by chosen faces are all covered by $G \cup H$, the union of two other active faces, and if the cost of the pair of faces (G, H) is no more than the cost of F, then we may discard F since any cover containing F can be modified by the elimination of F and the addition of both G and H at no increase in cost.

The above idea can be extended to a k-face covering in an obvious way. The two-face cover condition is already difficult to achieve since the two covering faces must be very large compared to the one covered. For example, in N-space we require

$$\text{cost }(G) + \text{cost }(H) \leq \text{cost }(F) \qquad \text{or}$$

$$N\text{-dim }(G) + N\text{-dim }(H) \leq N\text{-dim }(F)$$

which resolves to

$$\dim (G) + \dim (H) \geq \dim (F) + N$$

or equivalently

★ $$(\dim (G) + \dim (H)) - \dim (F) \geq N$$

6. Experimental Results and Timings for the Top Down Maximal Faces Algorithm

The Top Down Maximal Faces Algorithm has bean programmed in Turbo Pascal and run on an IBM PC driven by a 5 mega Hertz Intel 8088 engine. It is reasonable to expect two orders of magnitude improvement from tuning the program and moving to a heftier 25 mega Hertz PS/2 or similar PC. As can be seen from the timings below we have been able to calculate hundreds of maximal faces in a few minutes. Textbook problems with 20 or so faces have typically taken only a few seconds.

The following summary of 12 random test cases shows that the times vary widely, correlating most closely with the number of face comparisons. We have not given much effort to an analysis of timing. Suffice it to say that the time seems almost certainly to depend on the number of steps times the average number of Good faces near the candidate face. The number of steps is very much a function of the deficit, that is, the number of non-subsumed faces at the outset. The other factor is influenced by the final count of maximal faces.

We should point out that we have not investigated how best to order the candidates from the Try set, nor have we improved the searches for containment in ways that would give a factor of four improvement.

Random Test Cases

	NOrig	NMF	#Steps	#Compares	Time* (secs)	Face Size %
1)	10	23	23	230	1	100
2)	20	90	100	5300	9	100
3)	25	173	220	19000	35	100
4)	25	80	101	3700	7	100
5)	30	206	244	23400	42	100
6)	30	172	200	11000	20	100
7)	30	290	409	83000	164	100
8)	50	63	65	1500	3	50
9)	100	139	161	6700	9	50
10)	100	149	168	7300	11	50
11)	100	419	679	58000	160	75
12)	100	487	757	65000	186	75

Legend : NOrig is the original number of faces.
NMF is the number of final maximal faces.
Face Size % is the upper bound of the dimension of each random face as a percentage of 16.

----

* 5 MHz Intel 8088 in old (circa '83) IBM PC

In addition to these randomly generated functions we have also tested the method against one of the so-called symmetric functions $S^6_{2..4}$. which is all points of the 6-cube with bitcounts between 2 and 4 inclusive. This function has 50 points in 3 layers of size (15, 20, 15). Each of the 15 points of layer 2 is the base for 6 different squares and these are the 90 maximal faces. This function was submitted to the algorithm in three different forms:

a) all 50 points

b) 15 faces of dimension 2

c) 15 2-d faces and the middle layer of 20 points

The 90 2-d faces were calculated from raw points in 62 seconds while the other two cases took 18 seconds. This should <u>not</u> be taken as evidence that the top down method will work nearly as well on points as on larger faces. The small size of faces in this problem is misleading in this regard.

We also did two versions of a test we call McCluskey Cyclic taken from his 1986 text on Digital Design principles. It consists of 22 points and 17 maximal faces in the 6-cube. The result emerged after 4 seconds from the 22 points, after only 1 second from the 17 maximal faces.

Another artificially constructed function of 40 faces and 8 maximal faces in 16 dimensions took only 4 seconds to solve.

It should be pointed out that our methods depend on the number of dimensions in a very modest way, and even that dependence is influenced by the hardware register width. On old PCs it will take twice as long to perform 32-bit vector operations as to perform 16-bit operations. In any case the dependency is strictly linear in the worst case. The only possible trouble spot is the bitcount calculation which depends at most linearly on dimension. Bitcount is intrinsically $\log_2 N$, less than linear.

7. Early Results from a Meager Covering Algorithm

We have implemented and tested a simple covering algorithm using ideas from section 5. A rough sketch of the method is as follows:

```
repeat
    Find new essential faces using overlap counts
    Find newly redundant or dominated faces using the
     parental join information
until No faces were discarded in the last step
```

This algorithm is meager in the sense that overlap counts are a sufficient condition for essentiality but by no means necessary so we miss recognizing some essential faces. Furthermore, this algorithm does not generate any other necessary sets and does no backtracking; it just gives up. Finally, it uses only the join information left over from the parent references in the maximal faces algorithm, whereas maximal faces may be joins in several different ways providing more chances for dominance and redundancy.

In spite of its meagerness this method has been rather successful on a number of random functions containing some essential faces. The following table summarizes several test cases:

Minimum Cover Results

NMF	Chosen	Discards	Active
23	10	13	0
63	50	13	0
80	21	56	3
90	20	70	0
139	97	37	5
149	100	49	0
173	25	138	10
206	30	172	4
290	10	80	200
419	66	247	106
487	94	388	5

In 11 random cases only 2 failed to be solved nearly completely, and in one of these two we resolved 75% of the faces. These results give us every reason to hope that the full method will be successful without enormous backtracking. The strategy will be to get as much mileage out of the meager methods as is possible and only when necessary turn to more detailed analysis and exhaustive backtracking.

8. Top Down Solution of the McCluskey Cyclic Problem

The McCluskey Cyclic function consists of 22 points that form 17 maximal faces as listed below with letter designations for convenience of reference (base and delta are given in octal):

Face	b(Δ)	Face	b(Δ)	Face	b(Δ)
X	1(14)	Q	23(4)	I	11(24)
Y	4(21)	E	5(22)	K	24(11)
Z	10(21)	D	5(12)	R	30(3)
W	22(11)	F	5(30)	M	30(5)
O	16(1)	G	11(6)	N	30(6)
P	16(20)	H	11(22)		

The face intersections are depicted by the following matrix in which each non-blank entry indicates a face intersection of the dimension given. In this problem, all faces are squares (dimension 2) except for the edges O, P and Q. Therefore all face intersections are points or edges.

Face Intersection Overlap for McCluskey Cyclic

	X	Y	Z	W	O	P	Q	E	D	F	G	H	I	K	R	M	N
X																	
Y	0																
Z	0																
W																	
O																	
P					0												
Q				0													
E	0	1					0										
D	1	0			0			1									
F	1	1						1	1								
G	1		0		0				1	0							
H	0		1	0							1						
I	1		1						0	1	1	1					
K		1						0		1			0				
R			1	1								1	0				
M			1							0		0	1	1	1		
N			0	0		0								0	1	1	

There are 6 maximal faces that are not joins -- X, Y, Z, W, E and N. The remaining 11 faces are joins in one or more ways as shown in the table below:

D = E | G

H = G | R

P = O | N

Q = W | E

G = D | H

K = Y | M

O = G | P , D | P

I = G | M , Z | F , X | M

R = Z | W , M | W , H | N

M = K | Z , K | R , N | I

F = X | K , I | Y , E | I , K | D

Necessary sets generated from bases and tops are:

Base	Covering Faces
1	X
4	Y
10	Z
22	W
16	O P
23	Q W
5	E D F X Y
11	G H I X Z
24	K Y
30	R M N Z

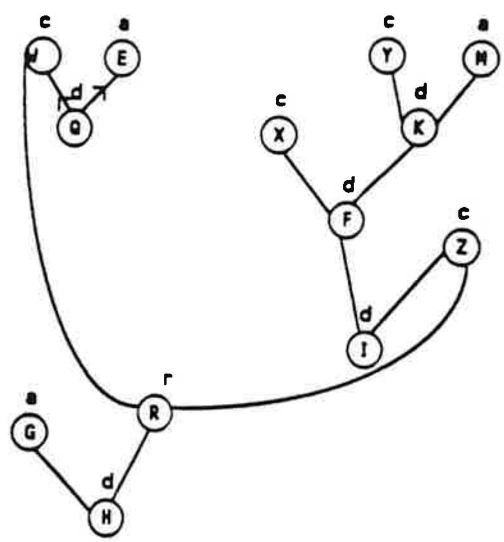
Top	Covering Faces
15	X I F D G
17	O D G
27	Q E
33	W H R
35	F I K M
36	P N
25	Y E K F
31	Z M I R H

Simple overlap counts do not reveal any essential faces but there are 4 necessary sets of size one so we start off quickly with chosen faces X, Y, Z and W. Face R, being a join with chosen parents, is discarded as redundant.

We can then discard several faces via dominance:

H is dominated by G via H = G | R with R redundant;
Q is dominated by E via Q = W | E with W chosen;
F is dominated by K via F = X | K with X chosen;
K is dominated by M via K = Y | M with Y chosen;
I is dominated by F via I = Z | F with Z chosen.

The dominance graph of these discards is:



in which we have studiously avoided cycles. Indeed, there are more joins tempting one to discard more faces, but they all get disqualified because they introduce cycles into the dominance graph. M alone has three joins that would normally be candidates for dominance since Z is chosen, R is redundant, and I is dominated but all three introduce a cycle.

The remaining 7 active faces with their face intersection overlaps are now:

Face	Size	Overlaps	Count
O	1	P(0) , D(0) , G(0)	3
P	1	O(0) , N(0)	2
E	2	X(0) , Y(1) , D(1)	5
D	2	X(1) , Y(0) , O(0) , E(1) , G(1)	8
G	2	X(1) , Z(0) , O(0) , D(1)	6
M	2	Z(1) , N(1)	4
N	2	Z(0) , W(0) , P(0) , M(1)	5

Unfortunately, this reveals no new essential face although P and M are exactly equal. The remaining necessary sets are:

Point	Faces Covering
16	O , P
17	O , D , G
27	E
35	M
36	P , N

We have eliminated each necessary set containing a chosen face and dropped all discarded faces.

The result is to give us two more essential faces, E and M, which are now chosen. Now D is dominated by G via newly chosen E, which does not introduce a cycle into the dominance graph.

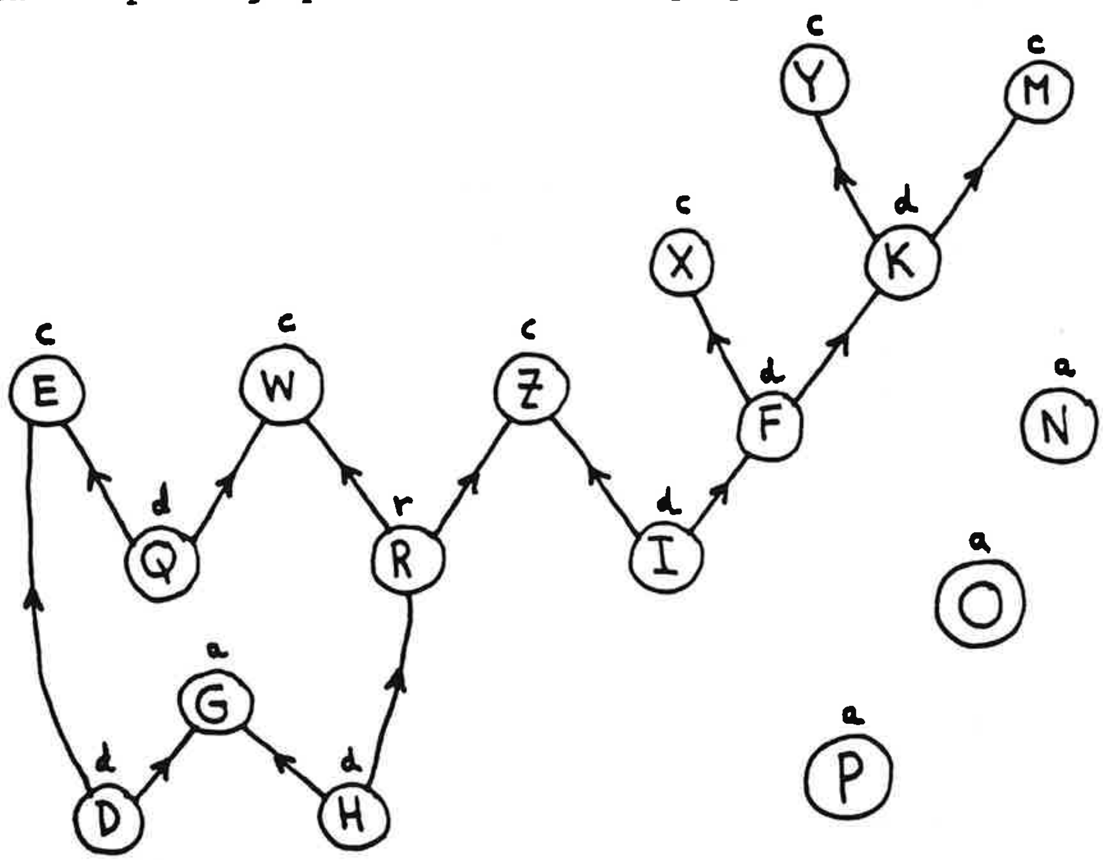
With D discarded, we have necessary sets

$[\ P\ O\ ]_{16}$ $[\ G\ O\ ]_{17}$ $[\ P\ N\ ]_{36}$.

At this point, we have the following partitioning of our 17 maximal faces:

Chosen: X, Y, Z, W, M, E
Redundant: R
Dominated: F, K, Q, H, D, I
Active: N, O, P, G

The complete graph of dominances employed is:



At this point we are down to 4 active faces and 3 necessary sets of 2 faces each. The final stage requires two-way backtracking and some detailed overlap analysis to arrive at the final solution
Chosen: X, Y, Z, W, M, E, P, G

9. Analysis of Unate Functions

A <u>monotonic</u> boolean function is one whose points form a <u>dual order ideal</u> in the lattice of the N-cube. This means that any point in the function drags along all points above it on all paths to the $\hat{1}$ point. Alternatively, a monotonic function is one whose maximal faces all contain the $\hat{1}$ point. This means simply that each face is of the form $b(\bar{b})$. As a result all maximal faces are essential via their base points so recognizing monotonicity helps to bypass the minimum cover problem.

It is not difficult to see that in a monotonic function there cannot be two different maximal faces whth the same base - this proves each such base is uniquely covered and hence the unique face is essential.

By the symmetry of the N-cube there must be other functions sharing the geometric constrainsts of monotonicity without actually being monotonic. Indeed, the so-called <u>unate</u> functions capture this idea. A unate function is one all of whose maximal faces have a common point, p. A transformation of the N-cube coordinates that maps p to the $\hat{1}$ point makes the transformed function monotonic. Since this coordinate transformation does not change any relevant geometric properties (e.g. faces, maximal faces etc.) we may conclude that unate functions also are minimum covered by all their maximal faces.

It is clear from the definition of unate that such a function has an FIG of maximal faces that forms a complete graph. It may be that this is also a sufficient condition. We are not sure.

In any case, we can iteratively calculate the intersection of all maximal faces of a function by performing one face comparison per maximal face until we get an empty intersection or find the maximal faces have one or more common points.

We need not perform this test if the FIG is not complete.

It may be a lot to expect functions to be unate but after a function is separated into independently simplifiable parts (see the next section) some of the smaller parts may be unate.

10. Function Separability Criteria

We know of two ways to break a function into parts whose minimal coverage can be handled independently. The first decomposition is into the connected components of the face intersection graph, each of whose pieces can be solved independently. This can come in handy if a function is not unate but one or more of its connected components is unate and therefore has a trivial to calculate mininmum cover (all the maximal faces of the component).

In a bottom up algorithm the connectivity of the DAG formed by edges of the function can be used to partition the problem early in the determination of maximal faces.

The second decomposition was recognized by Quine in 1952 and was partly attributed to Goodman. The idea is that sets of product terms that share letters amongst themselves but not with any other terms can be simplified independently. In our face calculus we can define perpendicular faces by

$$F \perp G \ \underline{\text{iff}} \ \overline{\Delta}_F \wedge \overline{\Delta}_G = 0.$$

If we now form a face graph according to the non-$\perp$ relation (the complement of the $\perp$ graph), then its connected components can be solved independently.

An example of this latter separability is the function f = a + BC whose two faces are perpendicular. Indeed the edge BC is geometrically perpendicular to the face a so our terminology is appropriate. The calculation is

$$a \equiv 0(3) \quad ; \quad BC \equiv 3(4)$$

so

$$\bar{\Delta}_a \wedge \bar{\Delta}_{BC} = \bar{3} \wedge \bar{4} = 4 \wedge 3 = 0$$

making $a \perp BC$.

Hence a and BC are in different components.

≈ ≈ ≈ ≈ ≈ ≈ ≈ ≈ ≈ ≈

Acknowledgements:

The desktop text entry (what we formerly called typing) of this difficult document was competently and zealously attacked by Gang Chen, a graduate student in computer science. Gang's efforts were supported by Andrew Tannen, another graduate student in computer science upon whom we have grown to rely for great and loyal service. Mark Svechin, a CS major with the aficionado's dexterity and the yeoman's dedication, produced the diagrams.

I wish to thank each of them for their fine service.

C. T. Z.

≈ ≈ ≈ ≈ ≈ ≈ ≈ ≈ ≈ ≈

Rio: A Missed Opportunity
by Dietrich Fischer

The United Nations Conference on Environment and Development in Rio de Janeiro in June 1992 (the Earth Summit), which I attended as representative of the organization "Economists Against the Arms Race," was the largest UN conference ever held. It was reportedly attended by 114 heads of state, about 10,000 government delegates from 178 countries, 8,000 representatives of 6,000 Non-Governmental Organizations and 7,000 journalists from all over the world.

One of the most controversial issues at the official conference was the question of how to finance sustainable development. The Group of 77, which now has over 100 developing countries as members, pressed hard for firm commitments from the developed countries to meet the agreed target of 0.7% of their GNP for development assistance by the year 2000. They were supported by France, the Netherlands and the Scandinavian countries, which have already exceeded that target, but were opposed by several of the developed countries, including the United States, which lags far behind. I participated with a group of NGO delegates from both developing and developed countries who discussed possible amendments to the draft text of the finance committee in the hope to achieve consensus. We proposed that the countries that have already achieved or exceeded the 0.7% target be commended and called upon to continue to provide leadership by increasing their efforts further. We also proposed that the countries that have not yet achieved that target commit themselves to reducing the gap by 20% by 1995, as a first step toward closing it entirely. Finally, we proposed that reductions in military spending be used to finance sustainable development. Standing outside the meeting of the finance committee, which met behind closed doors, we handed leaflets with these proposed amendments to all delegates as they entered and discussed them with as many as were willing to talk with us. The first of the three proposals was included in the final text agreed on by governments, but the others were not.

I found some of the personal conversations I held even more valuable than large meetings. One of the people I met was Airat Gumarov, a young Russian geologist who is training with an international, interdisciplinary team of scientists for a trip from 1993 to 1999 through 89 countries on every continent to study the conditions of the environment and to publicize their findings. Their project, which has the support of the Russian military, also envisages the creation of technology centers in about 20 countries, where new environmentally sound technologies are to be developed and made available for free to everyone. They hope that the world media will help widely to disseminate technologies that can help solve problems of sustainable development.

The Earth Summit could hardly have taken place during the Cold War. Now that most people and governments begin to realize that the greatest threat they face is not military aggression from an enemy, but underdevelopment and the destruction of the environment, the whole earth could jointly focus on these common problems. Although the concrete agreements reached fell short of expectations, the earth charter approved at the conference sets new standards, to which future generations must hold their governments accountable, in the same way as the Universal Declaration of Human Rights of 1948 set a new standard, even though the struggle to implement it still continues.

The new political climate has offered a great opportunity to reach fundamental new agreements that can shape the course of the world for decades to come, as the founding of the UN in San Francisco and the establishment of a world financial order at Bretton Woods did after World War II. But as the Soviet Union, by refusing to participate in the Bretton Woods agreements, prevented the creation of a truly global economy, the Bush administration played a similarly negative role in Rio. Its refusal to participate may be even more devastating. While international trade is possible within a limited area, the global environment can be saved from destruction only if all major industrial countries participate. As long as even one single nation continues to pollute the atmosphere or oceans, people all over the world are in danger. Since the United States, with 5 percent of the earth's population, emits about 30 percent of the world's carbon dioxide and is the major user of nonrenewable resources, its participation in global agreements on environment and development is indispensable. Hopefully, a future administration will soon join the rest of the world in these efforts.

Of the two major agreements to be reached, on climate and biodiversity, the United States refused to sign one and seriously weakened the other. On preventing the greenhouse effect, the grand bargain foreseen was that the developed countries would agree to limit their emissions of carbon dioxide and that the developing countries would in return protect their forests. When the Bush administration insisted that all references to any targets or timetables for limitations on carbon dioxide emissions be deleted from the climate convention as a condition for signing it, and the other countries reluctantly caved in, it became difficult to demand that the developing countries adhere to their side of the bargain. President Bush's last minute initiative for a treaty on the protection of forests, in itself a good idea, was perceived as a ploy to divert attention from the United States' refusal to limit its carbon dioxide emissions. Malaysia argued that telling it not to cut down its tropical forests was interference with its national sovereignty. When the Bush administration announced a week before the Earth Summit that half of the remaining old growth forests in the United States would be opened for logging, this was not helpful either.

The other agreement that had the support of every country except the United States was an innovative approach to protect species from extinction by making their preservation economically rewarding. The biodiversity treaty foresees that if pharmaceutical companies develop new drugs on the basis of substances extracted from rare plants or animals, they compensate the country of origin with a share of the profits derived from that drug. The amount of compensation was not specified, to be left for future negotiations, only the general principle was enunciated. The Bush administration argued that this would undermine the incentive for biotechnology firms to do research, because they would be deprived of a portion of their profits. But it ignores that developing countries, in which most of the world's remaining plant and animal species are found, also need some incentive to protect them from extinction. About 97 percent of all species have never been tested for their potential therapeutic value, and tens of thousands of species are lost every year, a rate far more rapid than during great periods of extinction in past geological ages. The recent discovery that an extract from a rare plant found only on Madagascar can cure a certain type of leukemia should make it clear what a tremendous resource we are carelessly squandering.

The developing countries were asking that environmentally sound production technologies be made available to them at concessional rates. If developing countries begin to use obsolete, highly polluting production methods, this also hurts people in the developed countries, since we are all affected by global warming, destruction of the ozone layer, acid rain and other pollutants that cross borders. But some developed countries, including the United States, seemed more concerned about intellectual property rights. The question of technology transfer is currently one of the major issues dividing developed and developing countries, but need not be. Knowledge is perhaps the most underutilized resource for sustainable development. Unlike physical resources, which must be given up by someone to be given to someone else, knowledge, once discovered, can be copied an unlimited number of times at very low costs. If the least polluting, least energy-, resource- and labor-intensive technology known anywhere on earth was available everywhere, all of us could live much better, in a healthier environment.

Clearly, those who discover new knowledge must be rewarded, otherwise the stream of innovations could dry up. But if there is a larger pie, it must logically be possible to divide it in such a way that everyone is better off, not only some at the expense of others. The patent system does reward innovation, but it excludes many potential users from access to valuable information. A more effective method to disseminate useful invention may be "compulsory licensing", as it is currently applied, for example, in the music industry. A composer has no right to prevent anyone else from playing or recording his or her compositions, but is entitled to a share of the profits from record sales. This has the effect that composers have no incentive to keep their melodies secret, but to spread them as widely as possible, to maximize their revenue. If the same system were applied to technological innovations, the whole world might benefit.

A useful method to finance sustainable development would be a pollution tax, especially on carbon dioxide, as the members of the European Community have advocated. Contrary to a widespread belief, such taxes would not increase overall tax levels, but could help reduce them, because they would also help reduce government expenses for cleaning up the environment. It is easy to see this if we imagine what would happen if gasoline were available for free. We would not pay less for gasoline, but a great deal more, because many people would start wasting it, and the tax payers would have to foot the bill anyway at the end of the year. This is the way we now typically treat clean air and clean water, and so it should not surprise us that these resources are being wasted. The current tax systems, which penalize hard work and creative ideas for new products that can meet human needs, ought to be replaced with tax systems that instead penalize harmful activities, such as pollution, the depletion of nonrenewable resources, military spending, and wasteful consumption.

The Bush administration's adamant refusal to agree to a carbon dioxide tax, which could offset other taxes, is short-sighted. Its insistence that we find out the precise consequences of global warming before taking action to prevent it is comparable to driving a car with closed eyes, waiting until we hit an obstacle before trying another direction, instead of anticipating and avoiding dangers. It is even worse, because global warming is irreversible. Once coastal areas become inundated, we cannot suddenly freeze huge quantities of ocean water and deposit them back on Antarctica. The entire world is held hostage to a myopic policy by one government, as if we were sitting on a bus being driven blindly toward an abyss from which there is no return.

Largely at the insistence of President Bush, reductions in military spending as a source of funding for sustainable development were left off the conference agenda. Yet with the end of the cold war, military spending is a tremendous untapped resource. A mere 12 percent of world military spending would be sufficient to fully fund the estimated $125 billion per year needed to implement Agenda 21, a comprehensive strategy for sustainable development. The $3.7 billion in new financing offered by developed countries at the conference represent about one day's military spending. Another example may illustrate the enormous waste of military spending. UNICEF has estimated that it would cost $1.50 on average to inoculate one child against six major diseases from which about 3 million children needlessly die each year. This death toll corresponds to the number of victims of about 30 Hiroshima bombs being dropped each year on the world's children, but is hardly ever mentioned. To inoculate all 120 million children born each year in rich and poor countries alike would cost about $180 million, or less than 10 percent of the $2.1 billion price for a single Stealth bomber.

Making some of these funds available for sustainable development is not simply a sacrifice by the developed countries to benefit the developing countries. It can also provide many new jobs in developed countries for workers who are displaced by reductions in military spending after the Cold War. There is no need to continue to build bombs and missiles just to keep people employed. Similar technical skills can be applied to the development of less polluting technologies. The Brundtland Commission has pointed out that those firms that have taken the lead in developing environmentally sound new technologies have been able to expand their markets, whereas those that have resisted popular demand for cleaner technologies have tended to decline.

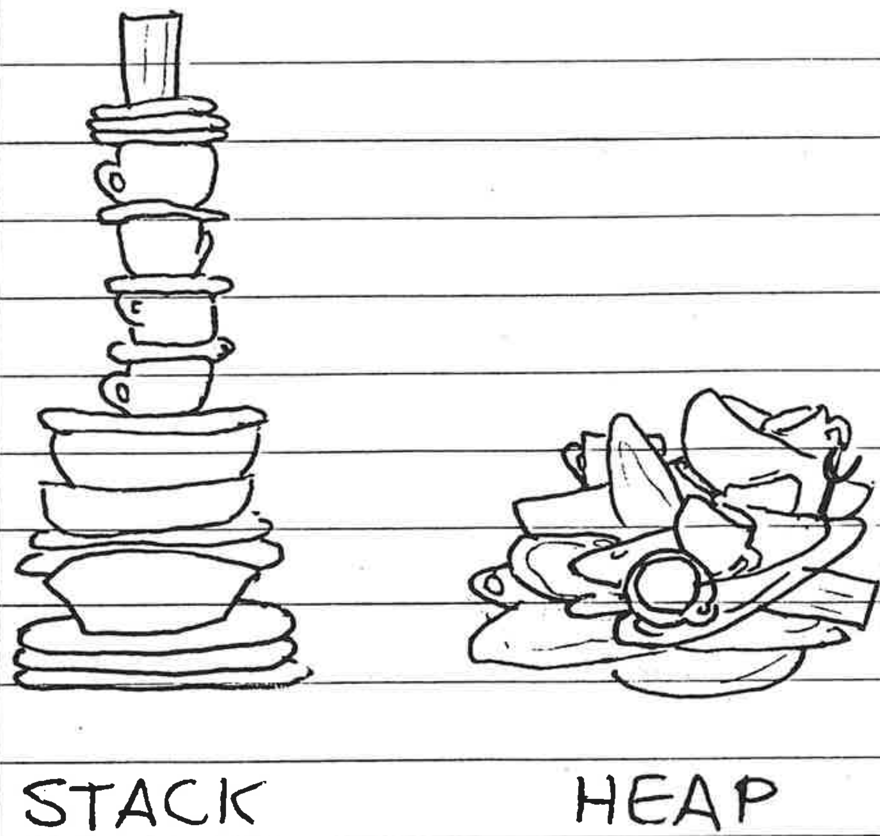
In addition to converting military into civilian industries, it would also make good sense to broaden the mission of defense departments to include environmental security. People whose livelihood has depended on military spending must be offered a new role, otherwise they will naturally oppose change. If security is redefined to offer protection also from the threats of pollution and resource shortages, satellites can be used to survey the global environment and provide advance warning of droughts, plant diseases and other causes of crop failures, troops can be redeployed to plant trees and clean up toxic wastes, and helicopters, tents and food rations can be used to save people in case of natural or industrial disasters.

It appears that the main reason why President Bush was so opposed to any meaningful global agreement was a reluctance to give up some national sovereignty, the fear that a UN bureaucrat could tell the United States what to do. But that fear is mistaken. Joining global agreements for mutual benefit does not reduce a nation's sovereignty, it extends it into new areas. No country today, for example, has sovereign control over the ozone layer. Only by agreeing to emission quotas on ozone-depleting gases and enforcement measures can we protect ourselves from carcinogenic ultraviolet radiation. By creating new global institutions in areas where they are necessary we lose nothing. On the contrary, we gain added control over our destiny, which we did not possess before and could never achieve at the national level alone.

It is interesting to note that the first advanced civilizations emerged some six thousand years ago when people faced problems that they could not solve alone. The recurrent floods and droughts in the Nile and Euphrates valleys required the cooperation of thousands of individuals to build dams for the control of those rivers. This led to the formation of organized states, the development of written language, the codification of laws, and a flourishing of science and the arts. Today we face some global problems that even a superpower cannot solve alone. Hopefully this will lead to international cooperation to address these problems before it is too late. We should heed Benjamin Franklin's admonition, "We must all hang together, or we shall all hang separately."

Dietrich Fischer, a Professor of Computer Science at Pace University, is a consultant to the United Nations Institute for Disarmament Research and author of **Nonmilitary Aspects of Security: A Systems Approach**. A shorter version of this report will appear in the Newsletter of Economists Against the Arms Race.

September 8 - Class #1 on Dynamic Memory Allocation



-Mollie