The Allocation of Resources by Linear Programming

Abstract, crystal-like structures in many geometrical dimensions can help to solve problems in planning and management. A new algorithm has set upper limits on the complexity of such problems

by Robert G. Bland

Consider the situation of a small brewery whose ale and beer are always in demand but whose production is limited by certain raw materials that are in short supply. Suppose the scarce ingredients are corn, hops and barley malt. The recipe for a barrel of ale calls for the ingredients in proportions different from those in the recipe for a barrel of beer. For instance, ale requires more malt per barrel than beer does. Furthermore, the brewer sells ale at a profit of $13 per barrel and beer at a profit of $23 per barrel. Subject to these conditions, how can the brewer maximize his profit?

It may seem that the brewer's best plan would be to devote all his resources to the production of beer, his more profitable product. This choice may not be well advised, however, because making beer may consume some of the available resources much faster than making ale does. If five pounds of corn are required for brewing a barrel of ale and 15 pounds are needed for brewing a barrel of beer, it may be possible to make three times as much ale as beer. Moreover, in brewing only beer the brewer may find that all his corn is used up long before his supplies of hops and malt are exhausted. It may turn out that by producing some beer and some ale he can take better advantage of his resources and thereby increase his profit. Determining such an optimum production program is not a trivial problem. It is the kind of problem that can be solved by the technique of linear programming.

Linear programming is a mathematical field of study concerned with the explicit formulation and analysis of such questions. It is a part of the broader field of inquiry called operations research, in which various methods of mathematical modeling and quantitative analysis are applied to large organizations and undertakings. Linear programming was developed shortly after World War II in response to logistical problems that arose during the war and immediately after it. One of the earliest publications on the uses of linear programming discussed a model of the 1948 Berlin airlift.

Although the computer is an indispensable tool for solving problems in linear programming, the term "programming" is employed in the sense of planning, not computer programming. "Linear" refers to a mathematical property of certain problems that simplifies their analysis. In the brewery problem the amount of any one resource needed to make either ale or beer is assumed to be proportional to the amount of the beverage produced. Doubling the amount of beer doubles the amount of each ingredient required for the brewing of beer, and it also doubles the profit attributable to the sale of beer. If the amount of corn consumed in making beer is plotted as a function of the amount of beer produced, the graph is a straight line. In order to apply the techniques of linear programming one must also assume that products and resources are divisible, or at least approximately so. For example, half a barrel of beer can be produced, and it has half the value of a full barrel.

Problems in linear programming are generally concerned with the allocation of scarce resources among a number of products or activities, under the proportionality and divisibility conditions I have described. The scarce resources may be raw materials, partly finished products, labor, investment capital or processing time on large, expensive machines. An optimum allocation may be one that maximizes some measure of benefit or utility, such as profit, or minimizes some measure of cost. In this era of declining productivity and dwindling resources a technique that can aid in allocating resources with the greatest possible efficiency may be well worth examining.

The properties of linear-programming problems derive from elementary principles of algebra and geometry. Solving such problems efficiently depends on algorithms, or step-by-step procedures, that cleverly exploit these algebraic and geometrical principles. The algorithms are also simple in conception, although the details of their operation can be quite intricate. It is the efficiency and versatility of a single algorithm called the simplex method that is largely responsible for the economic importance of linear programming.

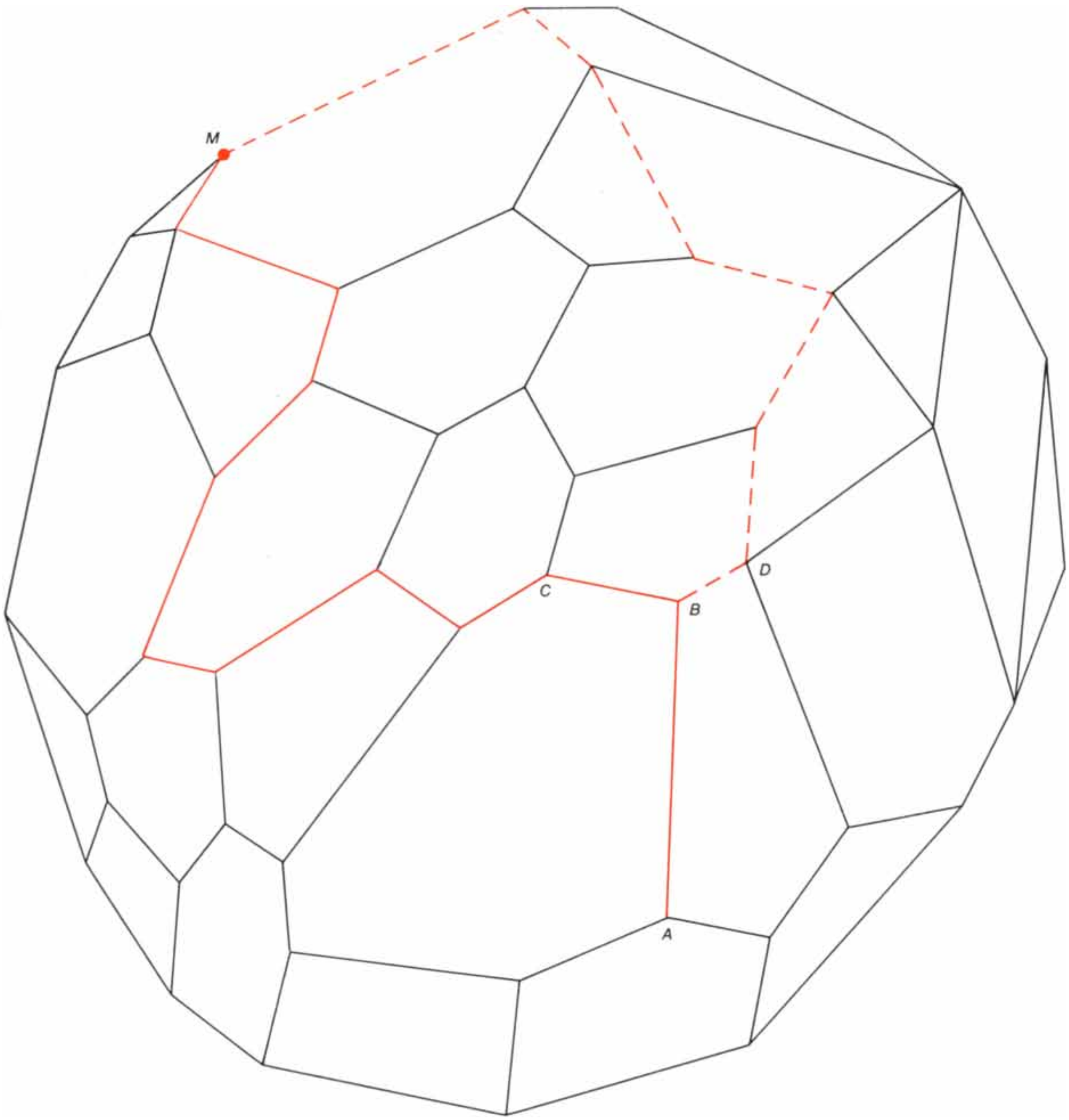
The simplex method was introduced in 1947 by George B. Dantzig, who is now at Stanford University. It has substantial value because it is fast, it is rich in applications and it can answer important questions about the sensitivity of solutions to variations in the input data. Such questions might take the following form: How should the production plan of the brewer change if the availability of hops or the profitability of beer is altered? How much should he be willing to pay for additional supplies of the scarce resources? What price should he ask from another entrepreneur who wants to buy some of his supplies? The simplex method can help in determining whether to buy a machine or sell one, whether to borrow money or lend it and whether to pay overtime wages or discontinue overtime labor. With the simplex method one can also impose additional constraints and solve the problem again to examine their effect. For example, the method can quickly tell an entrepreneur the cost of providing an unprofitable service in order to maintain the goodwill of a customer.

The simplex method has proved extremely efficient in solving complex linear-programming problems with thousands of constraints. For theoreticians, however, its speed in solving such problems has been somewhat puzzling. There is no definitive explanation of why the method does so well. Indeed, there are problems devised by mathematicians for which the simplex method is intolerably slow For reasons that are not entirely clear, such problems do not seem to arise in practice.

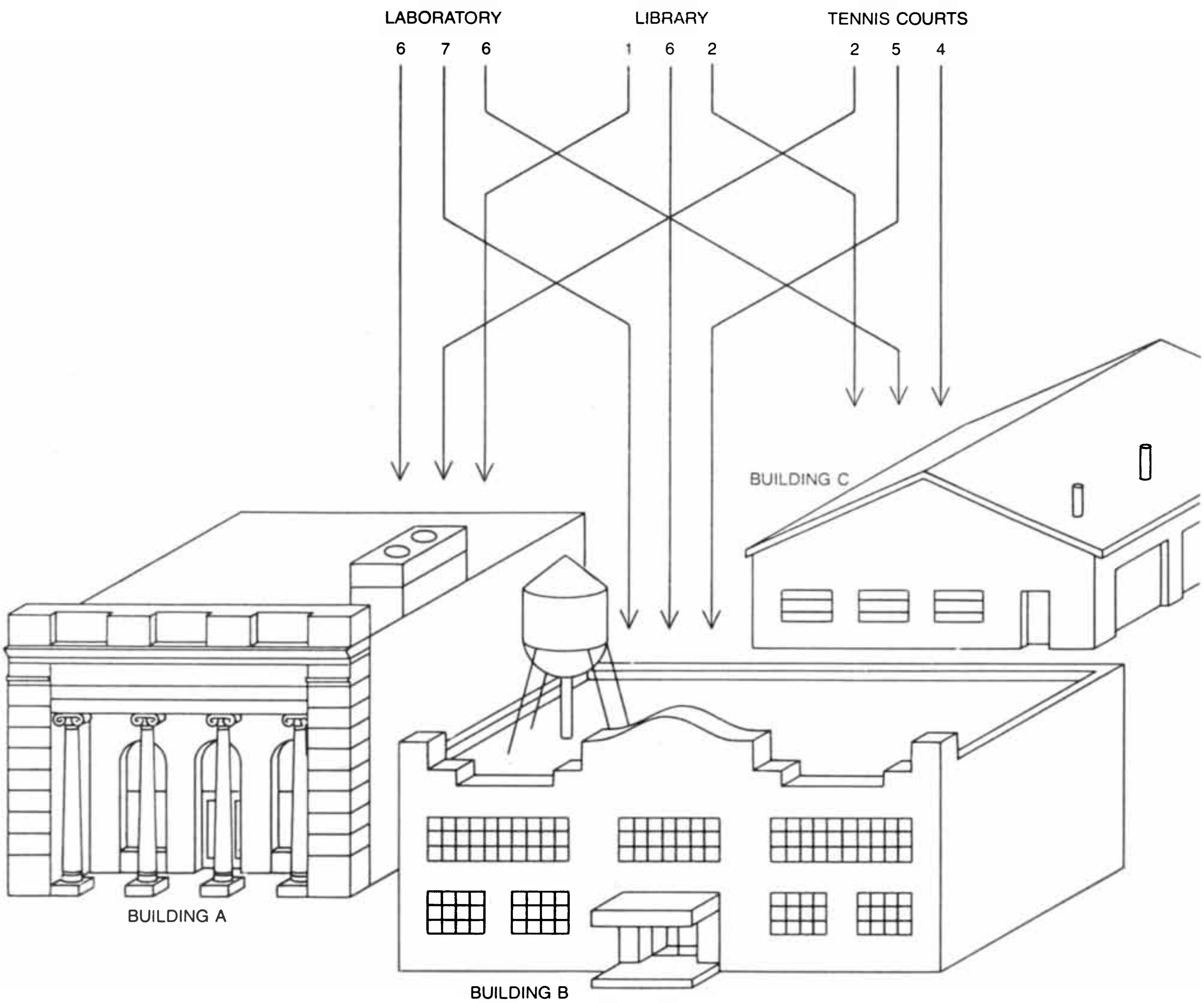
Mathematicians in the U.S.S.R. have recently developed a new algorithm for linear programming that in a certain sense avoids some of the theoretical difficulties that have been attributed to the simplex method. The development was reported in front-page articles in newspapers throughout the world, which suggests the economic significance of linear programming today. Unfortunately the new algorithm, which is called the ellipsoid method, has so far shown little prospect of outperforming the simplex method in practice. For now there is a curious divergence of practical and theoretical measures of computational performance.

Even when an efficient algorithm is employed, the setup costs of solving a large linear-programming problem can be considerable. Expressing a practical set of circumstances in terms of linear programming is not a trivial enterprise, and neither is the collection and organization of the data describing the circumstances. Moreover, solving the problem is feasible only with the aid of a high-speed computer.

Nevertheless, the economic benefits



SIMPLEX METHOD of solving problems in linear programming finds an optimum allocation of resources by moving from one vertex to another along the edges of a polytope, a three-dimensional solid whose faces are polygons. Each point in the region where the polytope is constructed corresponds to a particular program, or plan, for allocating labor, capital or other resources. Associated with each such program is a net benefit or a net cost. The object of linear programming is to find the program with the maximum benefit or the minimum cost. The polytope defines a region of feasibility: all programs of allocation represented by a point within the polytope or on its surface are feasible, whereas those represented by a point outside the polytope are infeasible because of a scarcity of resources. When the relation between resources and benefits or costs is linear, the maximum and minimum values must lie at one of the vertexes of the polytope. The simplex algorithm examines the vertexes selectively, tracing a path (*ABC...M*) along the edges of the polytope. At each step along the path the measure of benefits or costs is improved until at the point *M* the maximum or minimum is reached. Often there are many paths from *A,* the starting vertex, to *M.* The polytope need not be three-dimensional and it commonly has thousands of dimensions.



	LABORATORY	LIBRARY	TENNIS COURTS
BUILDING A	**6**	1	2
BUILDING B	7	**6**	5
BUILDING C	6	2	**4**

6 + 6 + 4 = 16

	LABORATORY	LIBRARY	TENNIS COURTS
BUILDING A	6	**1**	2
BUILDING B	**7**	6	5
BUILDING C	6	2	**4**

1 + 7 + 4 = 12

	LABORATORY	LIBRARY	TENNIS COURTS
BUILDING A	6	1	**2**
BUILDING B	**7**	6	5
BUILDING C	6	**2**	4

2 + 7 + 2 = 11

	LABORATORY	LIBRARY	TENNIS COURTS
BUILDING A	**6**	1	2
BUILDING B	7	6	**5**
BUILDING C	6	**2**	4

6 + 5 + 2 = 13

	LABORATORY	LIBRARY	TENNIS COURTS
BUILDING A	6	**1**	2
BUILDING B	7	6	**5**
BUILDING C	**6**	2	4

1 + 5 + 6 = 12

	LABORATORY	LIBRARY	TENNIS COURTS
BUILDING A	6	1	**2**
BUILDING B	7	**6**	5
BUILDING C	**6**	2	4

2 + 6 + 6 = 14

ASSIGNMENT PROBLEM seeks to minimize the cost of matching buildings available for renovation with functions that must each be confined to one building. With three buildings and three functions there are 3^2, or nine, renovation costs that must be considered. The costs (which are given here in millions of dollars) are conveniently represented in a matrix of numbers; a feasible assignment picks one number from each row and from each column of the matrix. There are $3 \times 2 \times 1$, or six, ways of accomplishing this. In general, for assignments of size n-by-n, the number of assignments is n factorial (written $n!$), which is equal to n multiplied by all positive integers less than n. Because the value of $n!$ increases rapidly, determining the minimum cost by enumeration of all possible assignments is impractical.

of linear programming are often substantial. In the mid-1950's, when linear-programming methods were developed to guide the blending of gasoline, the Exxon Corporation began saving from 2 to 3 percent of the cost of its blending operations. The application soon spread within the petroleum industry to the control of additional refinery operations, including catalytic cracking, distillation and polymerization. At about the same time other industries, notably paper products, food distribution, agriculture, steel and metalworking, began to adopt linear programming. Charles Boudrye of Linear Programming, Inc., of Silver Spring, Md., estimates that a paper manufacturer increased its profits by $15 million in a single year by employing linear programming to determine its assortment of products.

Today "packages" of computer programs based on the simplex algorithm are offered commercially by some 10 companies. Roughly 1,000 customers make use of the packages under licenses from the developers. Each customer pays a sizable monthly fee, and so it is likely that each makes use of the method regularly. Many more organizations have access to the packages through consultants. In addition special-purpose programs for solving problems of flows in networks have been developed, and they may be in even wider service than the general-purpose linear-programming algorithms.

The broad applicability of linear programming can be illustrated even within a single organization. Exxon currently applies linear programming to the scheduling of drilling operations, to the allocation of crude oil among refineries, to the setting of refinery operating conditions, to the distribution of products and to the planning of business strategy. David Smith of the Communication and Computer Sciences Department of Exxon reports that linear programming and its extensions account for from 5 to 10 percent of the company's total computing load. This share has kept pace for the past 20 years with rapidly expanding general applications of information processing.

Although the simplex method is a powerful tool, it is founded on elementary ideas. In order to understand some of these ideas it is useful to examine a specially structured allocation problem called the assignment problem. Consider the situation of a university planning committee with three buildings available for renovation and three functions the buildings are to serve. Suppose the functions are those of a laboratory, a library and indoor tennis courts, so that there can be only one function to a building. The tables in the illustration on the opposite page indicate the cost of renovation for each of the nine possible matches of a building with a function. How can the committee minimize the renovation cost?

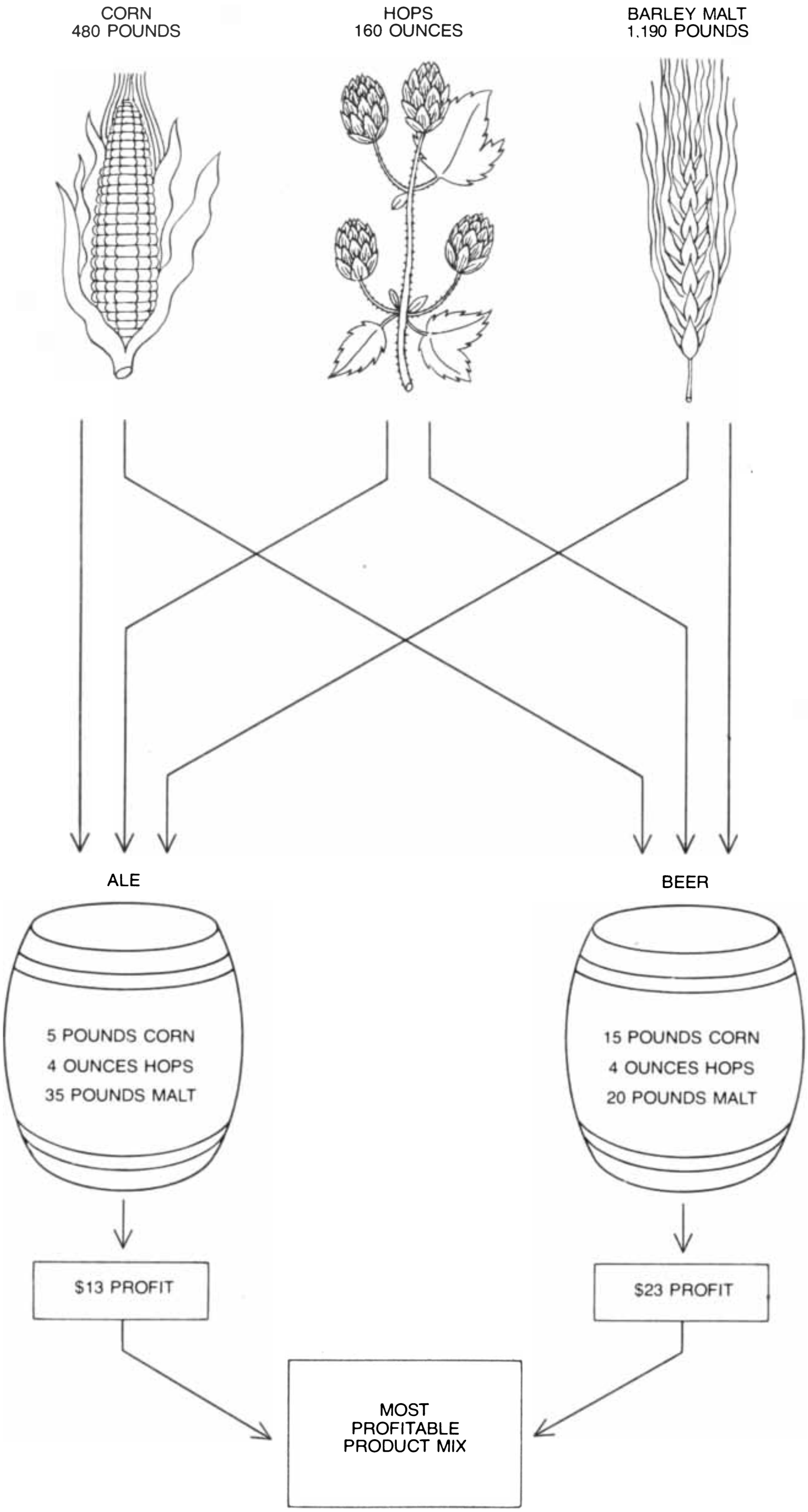
The problem can be solved by marking three of the squares in the 3-by-3 table. Precisely one square must be marked in each row and one must be marked in each column, so that each building has a function and all the functions are accommodated. The solution being sought is the one in which the sum of the costs in the marked squares is as small as possible. Finding an optimum solution is not difficult because there are only a few possible ways of marking the squares. Once one of the three squares in the first row is marked, only two squares in the second row remain available for marking. In the third row the choice is forced, because only one square appears in a heretofore unmarked column. Hence there are $3 \times 2 \times 1$, or six, ways of making the assignment. It is an easy matter to enumerate all six possibilities,

	SUBTRACT 3	SUBTRACT 4	SUBTRACT 8	SUBTRACT 2	SUBTRACT 9	SUBTRACT 7	SUBTRACT 0	SUBTRACT 5	SUBTRACT 1	SUBTRACT 6
SUBTRACT 2	6	6	12	4	14	11	3	9	4	10
SUBTRACT 5	8	12	17	8	16	16	6	13	7	13
SUBTRACT 6	11	13	18	9	15	16	8	14	8	14
SUBTRACT 4	8	10	15	10	16	13	7	9	7	11
SUBTRACT 8	13	15	16	11	19	19	10	16	12	18
SUBTRACT 5	9	12	17	8	16	15	7	13	7	11
SUBTRACT 3	8	8	13	8	13	12	6	8	4	11
SUBTRACT 1	6	8	11	3	14	10	2	8	3	9
SUBTRACT 0	5	5	12	2	12	7	2	9	6	12
SUBTRACT 7	15	14	17	13	17	18	7	18	10	16

↓

1	0	2	0	3	2	1	2	1	2
0	3	4	1	2	4	1	3	1	2
2	3	4	1	0	3	2	3	1	2
1	2	3	4	3	2	3	0	2	1
2	3	0	1	2	4	2	3	3	4
1	3	4	1	2	3	2	3	1	0
2	1	2	3	1	2	3	0	0	2
2	3	2	0	4	2	1	2	1	2
2	1	4	0	3	0	2	4	5	6
5	3	2	4	1	4	0	6	2	3

CONVINCING A SKEPTIC that an assignment is optimum (*colored areas in upper illustration*) does not entail evaluating the cost of all the feasible assignments. Here the cost matrix is a 10-by-10 one, and so there are 10!, or 3,628,800, possible assignments. If a number is subtracted from each entry in a row or column, the number is also subtracted from the total cost of any possible assignment and the relative costs remain unchanged. The reason is that every assignment picks exactly one number from the transformed row or column of the matrix. The set of numbers to be subtracted can be chosen so that the original matrix is transformed into a matrix that has no negative entries and has at least one entry with a cost of zero in each row and column (*lower illustration*). Because no assignment can have a total cost less than zero the optimum assignment for the transformed matrix must be one that has a total cost of zero. It follows that the entries in the corresponding positions in the original matrix are also optimum. Efficient algorithms have been devised for generating the set of numbers to be subtracted.



BREWER'S DILEMMA illustrates an application of linear programming to problems of optimizing the apportionment of resources among various products. The brewer's production of beer and ale is limited by the scarcity of three essential ingredients: corn, hops and malt. Feasible production levels are determined not only by the total amount of each ingredient on hand but also by the proportions of the ingredients required to make the two products. The objective function, or the quantity to be optimized, is the brewer's profit. In linear programming all the resources, all the products and all the benefits are assumed to be divisible quantities: the brewer can use half a pound of corn, sell a fourth of a barrel of ale and realize proportional profits.

evaluate the total cost of each one and select the least expensive assignment.

This enumerative approach solves the problem of a 3-by-3 matrix, but it becomes impractical for larger problems. Suppose there are four buildings and four functions; the number of possible assignments is then $4 \times 3 \times 2 \times 1$, or 24. In the general statement of the problem there are n buildings and n functions, and the number of assignments is n factorial (written $n!$), which signifies n multiplied by all the integers from 1 to $n - 1$. For $n = 10$ there are $10!$, or more than 3.6 million, distinct assignments.

The rapid growth of $n!$ dispels any enthusiasm one might have for the enumerative method. Suppose one had to solve a 35-by-35 assignment problem by enumeration and one had a computer that could sort through the possible assignments, evaluate the cost of each one and compare it with the lowest-cost assignment encountered up to that point at a rate of a billion assignments per second. (A computer capable of this speed would be much faster than any available now.) Even if the task of enumerating the $35!$ assignments were to be shared by a billion such computers, only an insignificant fraction of the required computations would be completed after a billion years.

The 35-by-35 assignment problem is not large. If the problem were one of assigning personnel to jobs in order to minimize the total cost of job training, n might well be equal to 35. There are numerous other assignment problems in which n is equal to 1,000 or more. Clearly such problems require a procedure cleverer than enumeration.

The burden of enumeration might be greatly reduced if one could avoid examining assignments that are costlier than the ones already checked. This effect would be achieved if there were a stopping rule or optimality criterion that would allow an optimum assignment to be recognized quickly once it was encountered. Any algorithm that incorporates such a criterion offers important collateral benefits. The benefits are summarized by what Jack Edmonds of the University of Waterloo in Ontario calls "the principle of the absolute supervisor," or what might also be called "the problem of the skeptical boss."

Suppose after tedious enumeration you have solved the 10-by-10 assignment problem indicated in the upper illustration on the preceding page by examining all 3,628,800 possible assignments. The optimum assignment, you maintain, is the one shaded in color in the illustration. You present the solution to your boss, who looks you in the eye, puffs on his cigar and demands, "How do I know there is no less costly solution?" You might swallow hard at


STOP
STOP
INTRODUCING THE SUBARU HILL-HOLDER.™*
WHAT YOU ALWAYS NEEDED IN THE CLUTCH.
You know that sinking feeling that can come over you when you stop on a hill with a standard shift automobile?
Well, now, in a Subaru, you don't have to hold your breath. With an exclusive Subaru feature called Hill-Holder, your car won't roll backwards.
Just depress the clutch and brake, and the Hill-Holder automatically engages.
When your foot is removed from the brake, the car will not roll backward.
In fact, it will remain motionless until the clutch is released and the accelerator depressed to proceed up the hill.
No drifting back. No white knuckles. No sweat. You no longer go downhill on your way uphill.
SUBARU®
INEXPENSIVE. AND BUILT TO STAY THAT WAY.
*AVAILABLE ON CERTAIN MODELS.
© SUBARU OF AMERICA, INC. 1981

this question, because it seems the only way to demonstrate the merits of your solution is to repeat the examination of all 3,628,800 possibilities under the scrutiny of your boss.

A stopping rule offers a concise proof of optimality. Suppose you go to your boss not only with the optimum assignment but also with a set of numbers to be subtracted from the entries in each row and column. Setting forth how these numbers are obtained would require a detailed discussion of the assignment problem; it will suffice to point out that the set of numbers can be specified by an efficient computer algorithm. The utility of the numbers, once they have been found, is readily appreciated. Note that subtracting the same number from every entry in a given row or column is equivalent to subtracting this amount from the total cost of every possible assignment. The reason is that every feasible assignment must choose one and only one entry from each row and each column. For example, if 5 is subtracted from every entry in the sixth row, every possible assignment will include exactly one entry that is 5 less than the corresponding assignment made with the original array of costs. The relative costs of all the assignments will therefore remain unchanged. Such subtraction can be done repeatedly, provided it is always applied uniformly to every entry in a given row or column.

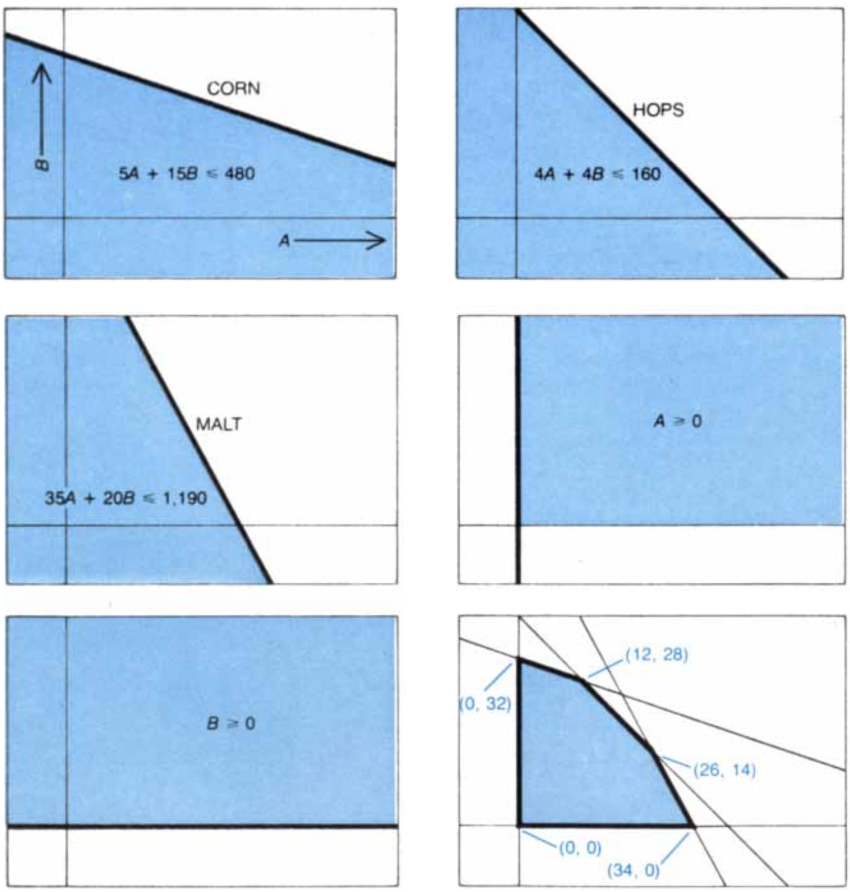
By means of repeated subtraction you can transform the original cost matrix into the matrix shown in the lower illustration on page 129. The latter array of costs has a remarkable property. You can now point out to your boss that the costs corresponding to the squares selected by your assignment are all zero and that no entries in the matrix are less than zero. Since the sum of the costs marked by your assignment is zero and there are no negative costs, no other possible assignment can have a lower cost. In short, you have shown your boss, with a few hundred calculations rather than tens of millions, that no assignment can be less expensive than the one you have chosen.

Although I have not demonstrated how to solve an assignment problem or how to find the row and column numbers to subtract, the assignment problem does illustrate the necessity of avoiding enumeration and the possibility of doing so by means of a stopping rule that recognizes an optimum solution. These are design characteristics of algorithms that can be applied not merely to the assignment problem but to linear-programming problems in general.

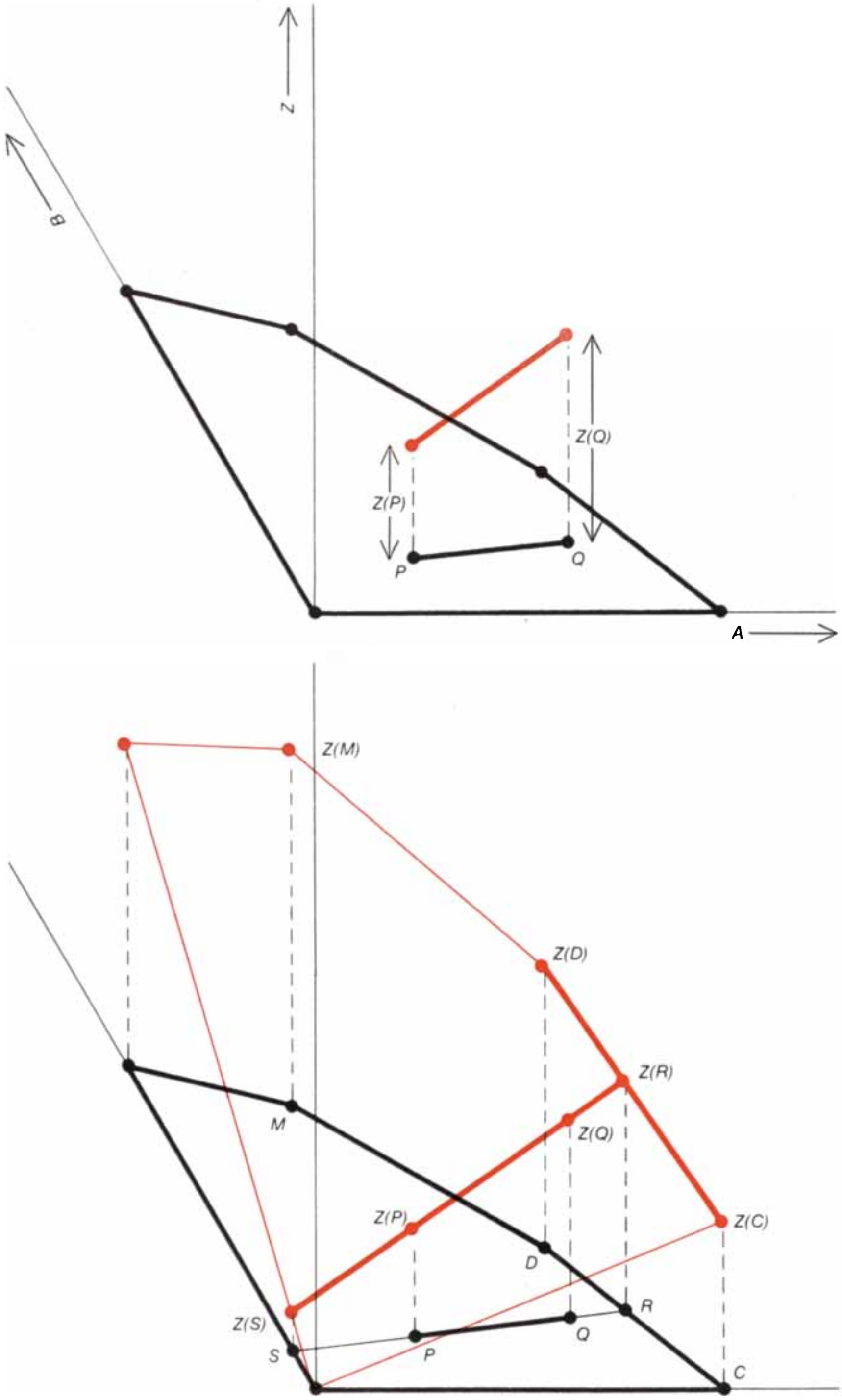
Consider again the situation of a brewer whose two products, ale and beer, are made from different proportions of corn, hops and malt. Suppose 480 pounds of corn, 160 ounces of hops and 1,190 pounds of malt are immediately available and the output is limited by the scarcity of these raw materials. Other resources, such as water, yeast, labor and energy, may be consumed in the manufacturing process, but they are considered to be plentiful. Although they may influence the brewer's willingness to produce beer and ale because of their costs, they do not directly limit the ability to produce. Assume that the brewing of each barrel of ale consumes five pounds of corn, four ounces of hops and 35 pounds of malt, whereas each barrel of beer requires 15 pounds of corn, four ounces of hops and 20 pounds of malt. Assume further that all the ale and beer that can be produced can be sold at current prices, which yield a profit of \$13 per barrel of ale and \$23 per barrel of beer.

The scarcity of corn, hops and malt limits the feasible levels of production. For example, although there are enough hops and malt to brew more than 32 barrels of beer, production of this much beer would exhaust the supply of corn, allowing no greater output of beer and no output of ale at all. Another feasible production program calls for no beer and 34 barrels of ale, depleting all 1,190 pounds of malt. The first alternative seems preferable to the second. The profit realized by the first program is $32 \times \$23$, or \$736, whereas the second program yields only $34 \times \$13$, or \$442.

There are other production programs that are better than either of these. Six barrels of ale and 30 barrels of beer use all 480 pounds of corn, 154 of the 160 ounces of hops and 810 of the 1,190 pounds of malt, yielding a profit of $(6 \times \$13) + (30 \times \$23)$, or \$768. Many programs earn even greater profit. In this case it is not merely impractical to



FEASIBILITY REGION in the brewer's problem is made up of the intersection of five half planes. Any point (A,B) in the plane corresponds to a production program that calls for making A barrels of ale and B barrels of beer. The first three half planes graphically represent all the production programs that are achievable, given the available quantities of each ingredient. For example, the amount of corn required is $5A + 15B$ (the weight in pounds of the corn needed to make a barrel of ale times A plus the weight of the corn needed to make a barrel of beer times B). This quantity must not exceed the 480 pounds of corn available to the brewer. Hence any point in the half plane to the lower left of the line $5A + 15B = 480$ represents a feasible production program that requires no more than the 480 pounds of corn available. The half planes associated with hops and malt can be constructed in a similar way. The remaining two half planes express the fact that only those programs having nonnegative production are feasible.



VALUE OF THE OBJECTIVE FUNCTION for any point *P* in the feasible region can be graphed as the distance *Z*(*P*) above or below *P* measured on the *z* axis. (Distances along the *z* axis have not been drawn to the same scale as distances in the plane of the feasible region.) One can think of the distance *Z*(*P*) as a point in three-dimensional space. If the function is linear, the graph of the values the function takes along any straight line in the feasible region is also a straight line (*upper graph*). For any point *P* in the interior of the region, there is some line through *P* that intersects the boundary of the region at two points, say the points *R* and *S* (*lower graph*). If the line segment in space connecting *Z*(*R*) and *Z*(*S*) is not parallel to the plane of the feasible region, the objective function must assume its maximum value along the line segment at one of its endpoints, say *Z*(*S*), which corresponds to the point *S* on the edge of the feasible region. The graph of the objective function along an edge is also a straight line and it too assumes its maximum at one of its endpoints, say *Z*(*D*), which corresponds to a vertex of the feasible region. Thus there is always an edge point that dominates a given interior point, and there is always a vertex that dominates any point along an edge. To find the maximum value *Z*(*M*) of the objective function one need examine only the vertexes. When the feasible region is two-dimensional, the objective function forms a plane whose maximum height is attained at a vertex.

enumerate all the possibilities, as it was in the assignment problem; here it is not even possible. There are infinitely many production programs that meet the conditions of the brewer's problem. Each such program is called a feasible solution. Fortunately there is a small set of feasible solutions called extremal solutions to which attention can be confined.

The importance of the extremal solutions becomes apparent when the set of all feasible solutions is represented graphically as a set of points in a plane; the set of points constitutes the feasible region. Let A designate the number of barrels of ale brewed according to any possible production strategy and let B designate the number of barrels of beer. A and B are known in linear programming as decision variables. They can be associated with the coordinate axes of the plane. Any point on the plane can be specified by a pair of coordinates (A,B), which also correspond to a particular set of production levels.

Because negative levels of production are not possible, the feasible region can immediately be confined to the upper right-hand quadrant of the plane, where A and B are both nonnegative. How does the scarcity of malt affect production? Since 35 pounds of malt are needed for each barrel of ale and 20 pounds are needed for each barrel of beer, the total amount of malt needed to make A barrels of ale and B barrels of beer is $35A + 20B$. If all 1,190 pounds of malt are used, $35A + 20B = 1{,}190$. The set of points (A,B) that satisfy this equation form a straight line. All points (A,B) corresponding to production plans that call for more than 1,190 pounds of malt lie on one side of the line and the points that require less malt lie on the other side. Only the latter set of points and the points actually on the line are feasible because of the limited supply of malt.

In a similar way the scarcity of hops confines the feasible region to one side of the line $4A + 4B = 160$ and the scarcity of corn confines the region to one side of the line $5A + 15B = 480$. The points that satisfy all these requirements make up the feasible region [*see illustration on page 132*]. Note that the feasible region is convex: any line segment that connects two points in the region (including the points on the perimeter lines) lies entirely within the region.

Since the brewer's profit is \$13 per barrel of ale and \$23 per barrel of beer, his problem is to maximize his total profit: $13A + 23B$. In order to do so he must find a point (A,B) in the convex feasible region where $13A + 23B$ has its maximum value. In linear programming such a measure of benefits to be maximized (or sometimes of costs to be minimized) is called an objective function.

The objective function can be incorporated into the graph of the feasible

ENTRY® computer services news

MCDONNELL DOUGLAS AUTOMATION COMPANY

MCAUTO UNIGRAPHICS–the CAD/CAM workhorse for Division Engineering at 3M.

Supporting a variety of engineering needs in a multi-billion dollar company can tax even the most professional organizations, the most advanced techniques and equipment.

But for Miland Meek, Manager of Computer Aided Design Service for Division Engineering at 3M, the work has been simplified with UNIGRAPHICS,™ the minicomputer-based graphics system from MCAUTO.

"We can use UNIGRAPHICS on a variety of capital equipment design projects—from a piece of equipment a block long and three stories high, to a very small die assembly," says Mr. Meek. "We need that versatility, and we get it with UNIGRAPHICS."

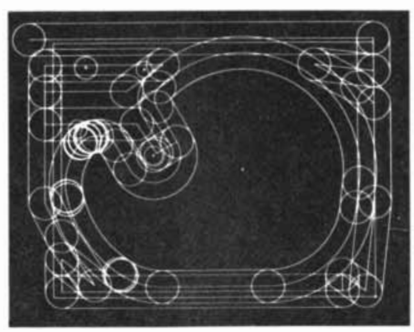
Besides capital equipment design, 3M uses UNIGRAPHICS to produce engineering drawings and to perform graphics machining for numerical control equipment. "It's a truly integrated tool here at 3M. We're now using most of the basic modules and we're exploring the use of both the Finite Element and Schematics modules," says Mr. Meek.

He also reports that UNIGRAPHICS has proven easy to learn and use. With 15 hours of instruction and 80 hours of practice, a trainee is well prepared to use it.

"Productivity is the goal. We've trained 85 people on the system and we're pleased to see that a trained user can be away from the system for some months, come back, and with the system's tutorial features be productive very quickly. Also, with remote stations—some are five miles from the host CPU—engineers turn travel time into productive time," says Mr. Meek.

With a single data base and full Automatic Programmed Tool capabilities, MCAUTO UNIGRAPHICS is the only totally-integrated CAD/CAM turnkey system available. It's the direct link between design and manufacturing that delivers the productivity and competitive edge you need. For more facts, call MCAUTO at (314) 232-8021, or mail the coupon.

"UNIGRAPHICS gives us the versatility and the productivity we need to keep our competitive edge," says Miland Meek, manager of CAD Service for Division Engineering at 3M.



The ubiquitous 3M tape dispenser, as seen on a UNIGRAPHICS terminal.

W. R. Vickroy, MCAUTO, Dept. KI-500
Box 516, St. Louis, MO 63166

☐ Please send me information on MCAUTO UNIGRAPHICS.

☐ Have your representative phone me for an appointment.

Name ____________ Title ____________

Company Name ____________ Phone ____________

Address ____________

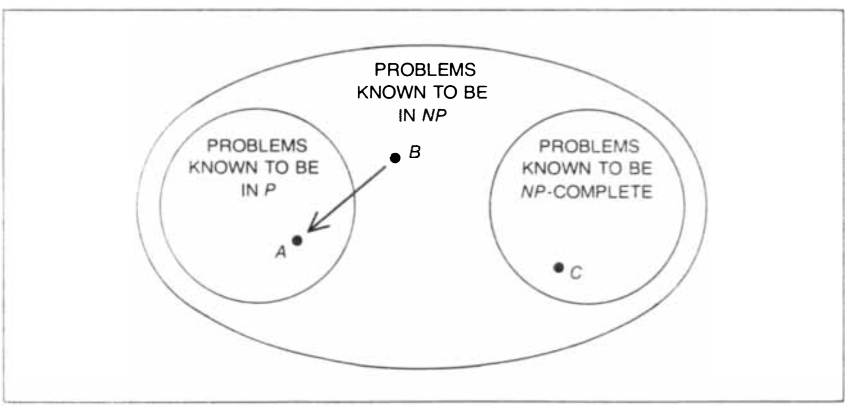
City ____________ State ____________ Zip ____________

MCAUTO
MCDONNELL DOUGLAS
Setting the standard in data processing

MCAUTO and ENTRY are registered trademarks of McDonnell Douglas Automation Company region by adding a third dimension. For any point (A,B) representing a production plan the expected profit is given by the height of the function $13A + 23B$ above the plane at that point. Clearly the task confronting the brewer is to find a point in the feasible region where the objective function is at its greatest height. If the function had to be evaluated at all the points, the task would be impossible, but two distinctive properties of the problem act to narrow the search. The properties are the convexity of the feasible region and the linearity of the objective function.

Since the region is convex, any point in the interior of the region can be included in a line segment whose endpoints lie on the boundary of the region. (Indeed, an infinite number of such line segments can be drawn through any given point; which line segment is chosen is immaterial.) In the space above each such line segment it is possible to construct a graph of the objective function giving the profit for each point on the line segment. Since the objective function is linear, the graph is a straight line [*see illustration on page 134*]. The straight-line graph may be parallel to the plane, in which case all the brewing strategies along the line segment have the same profit. If the graph of the objective function over the line segment is not parallel to the plane, it must assume its maximum value at one of the two endpoints, which lie on the boundary of the feasible region. Hence the maximum of the objective function over such a line segment must always be attained at one of the points where the line segment intersects the boundary. Because the same analysis can be applied to any line segment in the feasible region, it follows that the overall maximum of the objective function is invariably found somewhere on the boundary of the region. The brewer in search of maximum profit can ignore the entire interior of the feasible region and consider only those brewing strategies that correspond to points on the boundary.

The analysis can be taken a step further by the same argument. If the feasible region is a polygon, every point on the boundary lies on a line segment whose endpoints are two of the vertexes of the polygon. A graph of the objective function for such a boundary segment can be constructed in the same way it is for a line segment that crosses the interior. Again the maximum must be found at one of the endpoints (or at both endpoints if the objective function is constant and parallel to the plane). Thus a maximum value of the objective function throughout the feasible region can be found among the vertexes. The brewer needs only to check, at most, the value of the function at all the vertexes of the feasible region and select the one yielding the best profit. He can then be certain that no other brewing strategy would bring a higher profit. In the example considered here there are five vertexes. The one at point (12,28), which represents the production of 12 barrels of ale and 28 barrels of beer, yields a profit of $(12 \times \$13) + (28 \times \$23)$, or \$800. That is the maximum profit the brewer can realize.



COMPUTATIONAL-COMPLEXITY THEORY has recently been able to place linear programming in the set *P* of polynomially bounded classes of problems. This assignment, represented by point *A*, was made possible because of a proof by the Russian mathematician L. G. Khachian that a newly discovered algorithm for linear programming, called the ellipsoid method, is polynomially bounded. A polynomial function of a number *n* is a finite sum of powers of *n*, each power being multiplied by a constant. A class of problems is polynomially bounded if the number of elementary arithmetical operations needed to solve any problem in the class is bounded by a polynomial function of some measure *s* of the size of the problem. Until Khachian's result it was not known whether such an algorithm existed for linear programming (although there was no proof that it did not exist). Linear programming was previously known to be in the larger set *NP* of nondeterministic polynomial functions (point *B*). Roughly speaking, *NP* is the set of problem classes for which the feasibility of a proposed solution can quickly be checked. A second subset of *NP* is called the set of *NP*-complete problems; proving that any *NP*-complete class of problems is polynomially bounded would show that all classes in *NP* are also in *P*. The *NP*-complete problems are, in a sense, the hardest problems in *NP* (point *C*).

The inclusion of an additional constraint (such as a shortage of yeast) would not significantly alter the geometrical interpretation of the brewer's problem. The polygon might then have six sides instead of five. The introduction of a third product, on the other hand, would have a more profound effect: the geometrical model would then be three-dimensional. Inequalities in three variables correspond to half spaces defined by planes in three-dimensional space instead of half planes defined by lines in two-dimensional space. The feasible region is no longer a polygon but might instead look like a cut gemstone, a three-dimensional polytope whose faces are all polygons. As the number n of decision variables increases further, the geometrical interpretation remains valid but it becomes more difficult to visualize the n-dimensional polytope formed by the intersections of $(n - 1)$-dimensional hyperplanes. Vertexes retain their special status, however, and their positions can be determined by algebraic methods that replace geometrical intuition.

It may appear that by confining the evaluation of the objective function to the vertexes of the feasible region, the solution of linear-programming problems by enumeration becomes practical. As in the assignment problem, however, the number of possibilities to be enumerated grows explosively. A problem with 35 decision variables and 35 constraints would be impossible.

Dantzig's simplex method examines vertexes, but it does so selectively. In the brewery example the method might begin with the vertex at the origin, (0,0). Here nothing is produced and the profit is zero. Following either of the incident edges away from the origin leads to points with larger objective-function values. The simplex method selects such an edge, say the B axis, and follows it to its other end, the vertex (0,32). The program here calls for making 32 barrels of beer but no ale, for a profit of \$736. From this vertex an incident edge leads to still greater objective-function values. The simplex algorithm therefore proceeds to the vertex (12,28) at the other end of the edge. Here the profit from brewing 12 barrels of ale and 28 barrels of beer is \$800. At (12,28) all incident edges lead in unfavorable directions; the algorithm therefore halts with a declaration that the vertex (12,28) is optimal.

In general the simplex method moves along edges of a polytope from vertex to adjacent vertex, always improving the value of the objective function. The procedure can begin at any vertex, and it halts when no adjacent vertex has a better objective-function value than the current one. This stopping rule is val-

NOW WE KNOW WHAT IT'S LIKE TO BE ALONE.

It wasn't always this way.

It used to be hard to decide among all the 35mm SLR's.

Then we created the new Minolta XG-M. A camera so extraordinary it stands alone in its class. With an unrivaled combination of creative features.

First, it's automatic. So it's easy to get sharp, clear pictures. You just point, focus, and shoot. It even has electronic features that keep you from making mistakes.

As your skills advance, you'll appreciate advanced features like metered manual and exposure override for full creative control.

To further separate ourselves from the competition, we built in the option of professional motor-drive. Something normally found only on more expensive cameras.

It lets you shoot a blazing 3.5 frames per second. So you can catch a baseball as it comes off the bat. Or halt a horse leaping a hurdle.

But to fully grasp the XG-M's advanced design, you have to hold it.

The body feels rugged yet light. With a built-in textured grip that's sculpted to fit snugly in your hand. And oversized controls that make it easy to adjust to changing conditions.

As your creative potential develops, you'll have access to over 45 interchangeable, computer-designed Minolta lenses. As well as the Minolta system of SLR accessories.

All in all, the XG-M is a remarkable achievement. But then, we have over 50 years of remarkable achievements to draw on.

The new Minolta XG-M. Now we know how it feels to be alone. But we're not complaining.

WAIT 'TIL YOU SEE HOW GOOD YOU CAN BE.

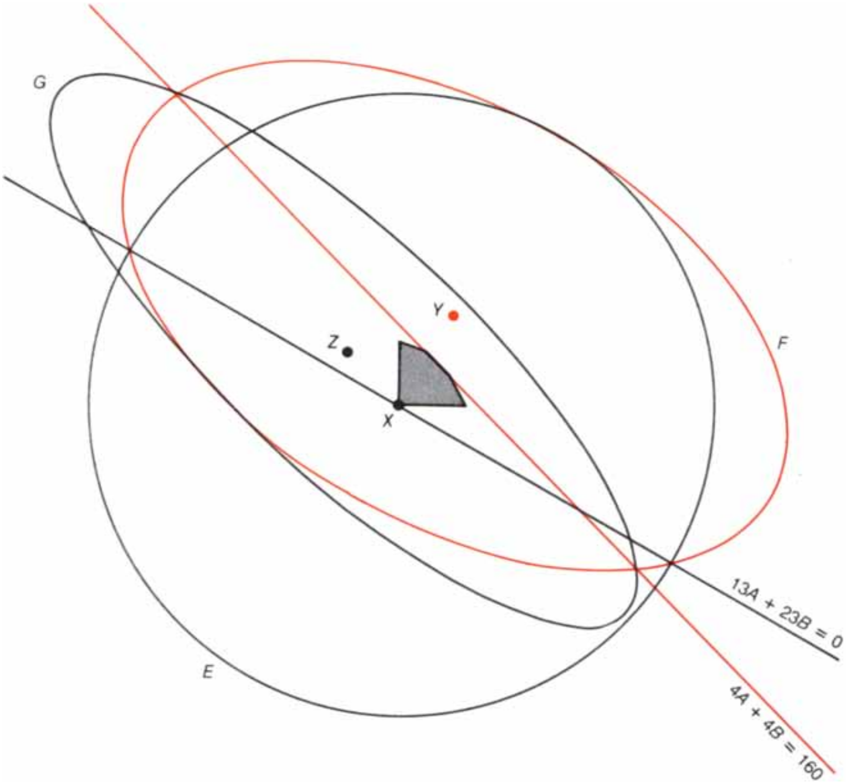
MINOLTA

For more information, write Minolta Corporation, 101 Williams Drive, Ramsey, N.J. 07446. Or see your Minolta dealer. In Canada: Minolta, Ontario, L4W 1A4.

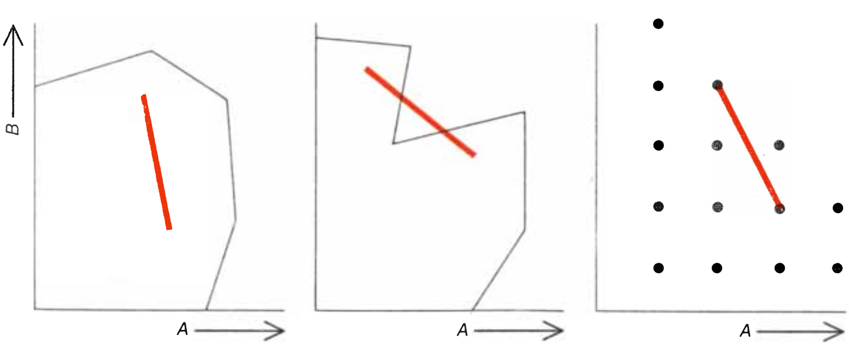
Product appearance and/or specifications are subject to change without notice.

id only because the feasible region is convex: convexity guarantees that any locally maximum vertex is a globally maximum vertex. One need look only in the vicinity of each point to tell whether improvement is possible.

How does one tell that an edge leading away from a given vertex will improve the value of the objective function? The key is the concept of a "marginal value" attributed to each resource at each vertex of the feasible polytope. For many-dimensional problems in linear programming the algebraic method that is commonly employed to choose a path from vertex to vertex can be understood when one understands the significance of the marginal values.



ELLIPSOID METHOD, like the simplex method, can be interpreted geometrically. For the brewer's problem it begins with a large circle *E* centered at the origin *X* and enclosing the feasible region. The size of the circle is determined by the problem data. The method proceeds by constructing a sequence of smaller ellipses, each of which contains the optimum solution. Because the center of the circle is feasible the next ellipse in the construction must contain all the points in the circle whose objective function is at least as great as it was at *X*. The smallest ellipse that accomplishes this is *F*. The center of *F*, at point *Y*, is not feasible. Hence the algorithm requires the next ellipse *G*, centered at *Z*, to contain every point in the ellipse *F* that lies on the feasible side of the constraint $4A + 4B = 160$, which was violated by the point *Y*. The process continues by cutting the current ellipse with a contour of the objective function if the center of the current ellipse is feasible. If the center is not feasible, the successor ellipse circumscribes the part of the current ellipse that satisfies a constraint violated at its center. The areas of the ellipses shrink rapidly enough to ensure that their centers converge to an optimum solution.



CONVEXITY PROPERTY of linear programming states that for any two points in the feasible region a line segment connecting the points must lie entirely within the region. Only the feasible region at the left is convex. The region in the diagram at the right is a set of discrete points. Convexity ensures that any local maximum of a linear objective function is a global maximum.

At the maximum vertex in the brewery problem there are 210 pounds of excess malt that are not utilized in the production program represented by that vertex. The addition or subtraction of a pound of malt from the initial supply of 1,190 pounds would not change the obtainable profit. On the other hand, another ounce of hops would make possible an increase of \$2 in total profit. This increase is the marginal value of hops at the maximum vertex. It can be interpreted as the effect of "pushing" the constraint line for hops farther from the origin to reflect the extra available ounce of hops [*see illustration on opposite page*]. Similarly, the marginal value of corn at this vertex is \$1.

Marginal prices have a natural economic interpretation. If the brewer could buy an additional ounce of hops, he could increase his profit by \$2. If hops were available for less than \$2 per ounce, it would be worthwhile to buy hops. On the other hand, if a price higher than \$2 per ounce were offered for hops, it would be worthwhile for the brewer to divert hops from the production of beer and ale and sell them on the market. This does not mean that the buying or selling of hops should continue indefinitely at \$2 per ounce, but for increases of about 19 ounces or decreases of up to 32 ounces, \$2 remains the break-even price in this example.

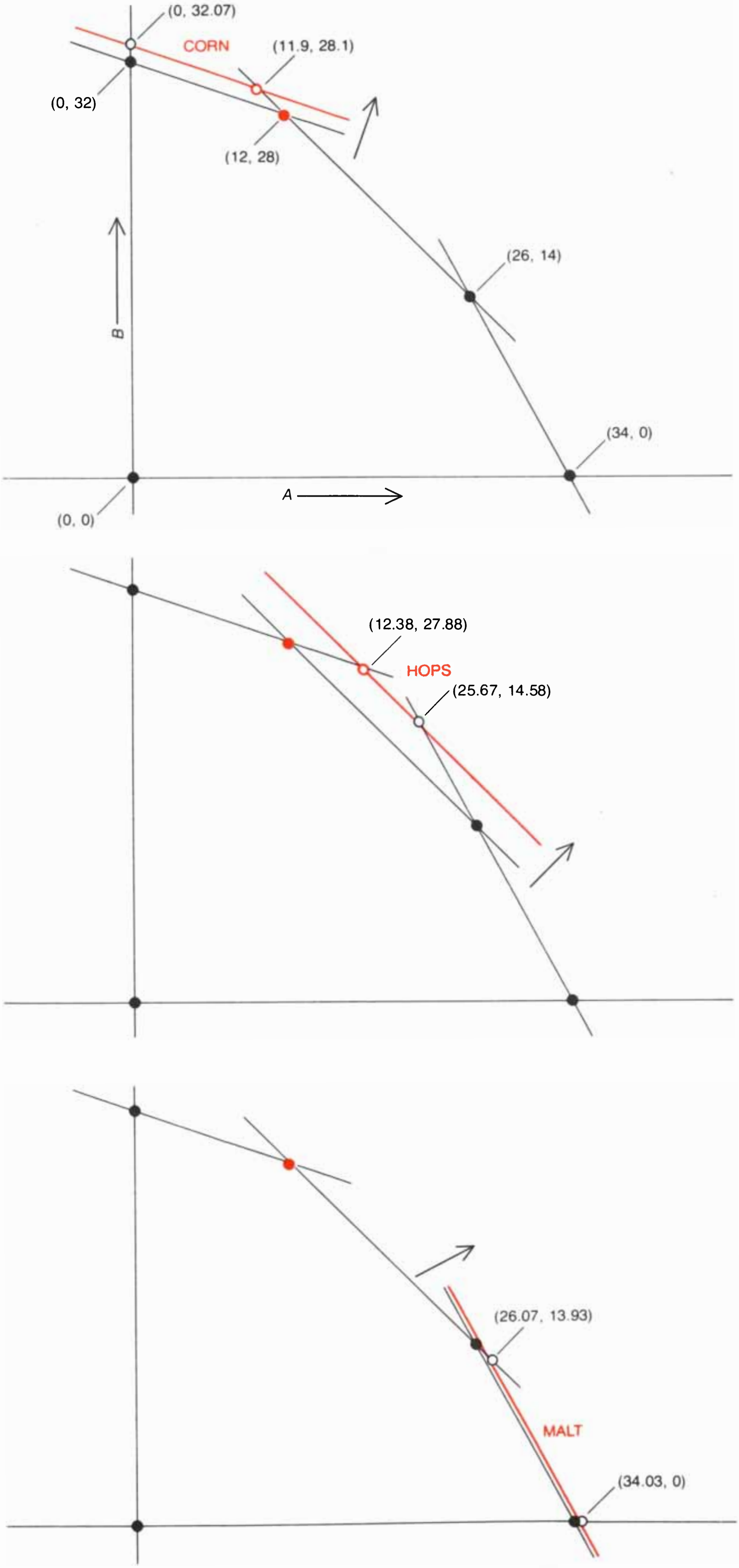
Marginal values are sometimes called shadow prices or imputed prices. They indicate the relative amount that each scarce resource contributes to the profitability of each item in production. For example, a barrel of ale requires five pounds of corn with an imputed price of \$1 per pound, four ounces of hops with an imputed price of \$2 per ounce and 35 pounds of malt with an imputed price of zero. The total imputed price of ale is equal to the profit of \$13 per barrel.

Suppose a new product, light beer, is proposed. Making a barrel of light beer requires two pounds of corn, five ounces of hops and 24 pounds of malt. How much profit per barrel must be obtained from light beer to justify diverting resources from the production of beer and ale? The imputed prices of the resources answer the question. The total imputed price of the ingredients in light beer is $(2 \times \$1) + (5 \times \$2) + (24 \times \$0)$, which is equal to \$12. This measures the profit that would be lost by diverting resources from the brewing of ale and beer to the production of one barrel of light beer. Thus for the brewing of light beer to be worthwhile it must yield a profit of at least \$12 a barrel.

The role of marginal prices in judging

the wisdom of introducing a new product can help to explain how the simplex method determines which edges of a polytope lead from a given vertex to a vertex with an improved value of the objective function. Suppose the vertex currently under examination is the one at (0,32) in the brewery problem. First one can determine the marginal price of each resource in this production program. Malt and hops are in excess supply at this point and so their marginal values are zero. Corn, however, is in short supply and so its marginal value is positive. Since only beer is being made at (0,32), the marginal value of the ingredients in a barrel of beer should be equal to the profit associated with beer: $23. Since only corn has nonzero marginal value and since 15 pounds of corn are needed for each barrel of beer, the marginal value of corn for this production program is $23 divided by 15 pounds, or about $1.53 per pound.

What is the marginal value of the ingredients in ale measured at the (0,32) vertex? If this value is less than the profit that can be realized from selling a barrel of ale, the diversion of some resources from beer production to ale would increase the total profit of the brewer. Making a barrel of ale requires five pounds of corn. Therefore the marginal value of the ingredients in a barrel of ale is ($23/15) × 5, or about $7.67. As in the example of light beer, the marginal value represents the loss in profit that would result from diverting resources from beermaking to the brewing of ale. Because this value is less than $13 (the profit expected from selling each barrel of ale) it would be profitable to brew more ale than the production program at the point (0,32) specifies. Indeed, by keeping the total consumption of corn constant and diverting corn from beer to ale, the brewer can increase his profit by $13 − $7.67, or $5.33, for each barrel of ale. Since the amount of corn is held constant, producing more ale corresponds to moving along the edge of the feasible region that represents the con-



MARGINAL INCREASES in the availability of scarce resources alter the potential profits of a hypothetical brewer in a predictable way. If one additional pound of corn were available, the maximum profit would increase by $1, a change in the objective function that is reflected in the movement of the maximum vertex in the feasible region from the colored dot to the colored circle (*upper graph*). If there were an extra ounce of hops, the maximum profit would increase by $2 (*middle graph*). A change in the available malt would not alter the attainable profit: malt is already a surplus resource (*lower graph*). The changes in profit associated with changes in the availability of each resource are known as shadow prices or imputed prices. They are used to direct the algorithm from vertex to vertex. The marginal changes are exaggerated for clarity.

THE LEADING EDGE

#1 in a series of reports on new technology from Xerox

About a year ago, Xerox introduced the Ethernet network—a pioneering new development that makes it possible to link different office machines into a single network that's reliable, flexible and easily expandable.

The following are some notes explaining the technological underpinnings of this development. They are contributed by Xerox research scientist David Boggs.

The Ethernet system was designed to meet several rather ambitious objectives.

First, it had to allow many users within a given organization to access the same data. Next, it had to allow the organization the economies that come from resource sharing; that is, if several people could share the same information processing equipment, it would cut down on the amount and expense of hardware needed. In addition, the resulting network had to be flexible; users had to be able to change components easily so the network could grow smoothly as new capability was needed. Finally, it had to have maximum reliability—a system based on the notion of shared information would look pretty silly if users couldn't get at the information because the network was broken.

Collision Detection

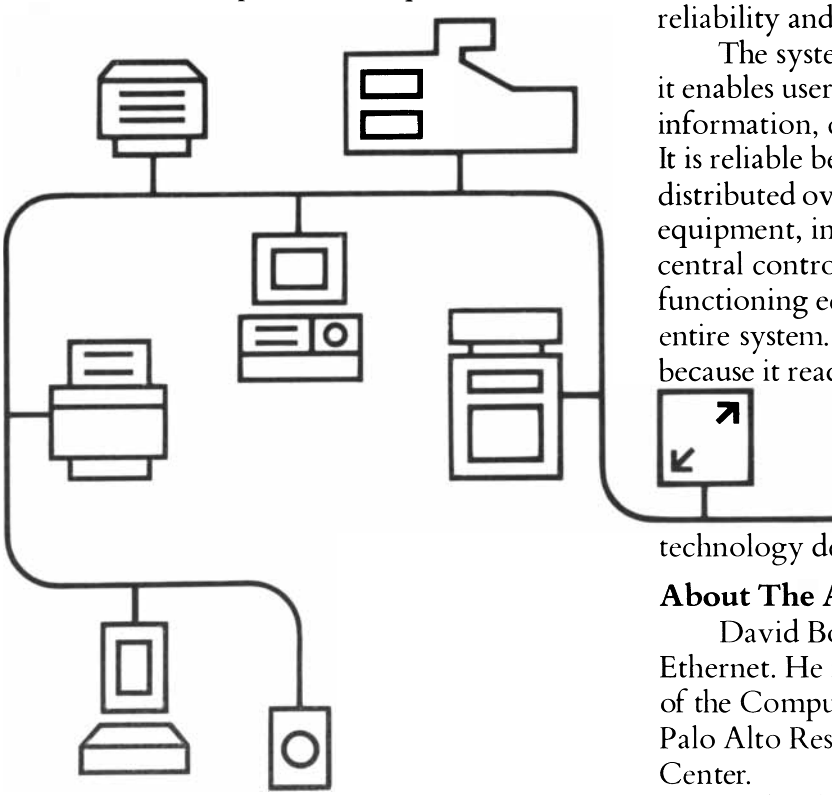
The Ethernet network uses a coaxial cable to connect various pieces of information equipment. Information travels over the cable in packets which are sent from one machine to another.

A key problem in any system of this type is how to control access to the cable: what are the rules determining when a piece of equipment can talk? Ethernet's method resembles the unwritten rules used by people at a party to decide who gets to tell the next story.

While someone is speaking, everyone else waits. When the current speaker stops, those who want to say something pause, and then launch into their speeches. If they *collide* with each other (hear someone else talking, too), they all stop and wait to start up again. Eventually one pauses the shortest time and starts talking so soon that everyone else hears him and waits.

When a piece of equipment wants to use the Ethernet cable, it listens first to hear if any other station is talking. When it hears silence on the cable, the station starts talking, but it also listens. If it hears other stations sending too, it stops, as do the other stations. Then it waits a random amount of time, on the order of microseconds, and tries again. The more times a station collides, the longer, on the average, it waits before trying again.

In the technical literature, this technique is called carrier-sense multiple-access with collision detection. It is a modification of a method developed by researchers at the University of Hawaii and further refined by my colleague Dr. Robert Metcalfe. As long as the interval during which stations elbow each other for control of the cable is short relative to the interval during which the winner uses the cable, it is very efficient. Just as important, it requires no central



control—there is no distinguished station to break or become overloaded.

The System

With the foregoing problems solved, Ethernet was ready for introduction. It consists of a few relatively simple components:

Ether. This is the cable referred to earlier. Since it consists of just copper and plastic, its reliability is high and its cost is low.

Transceivers. These are small boxes that insert and extract bits of information as they pass by on the cable.

Controllers. These are large scale integrated circuit chips which enable all sorts of equipment, from communicating typewriters to mainframe computers, regardless of the manufacturer, to connect to the Ethernet.

The resulting system is not only fast (transmitting millions of bits of information per second), it's essentially modular in design. It's largely because of this modularity that Ethernet succeeds in meeting its objectives of economy, reliability and expandability.

The system is economical simply because it enables users to share both equipment and information, cutting down on hardware costs. It is reliable because control of the system is distributed over many pieces of communicating equipment, instead of being vested in a single central controller where a single piece of malfunctioning equipment can immobilize an entire system. And Ethernet is expandable because it readily accepts new pieces of information processing equipment. This enables an organization to plug in new machines gradually, as its needs dictate, or as technology develops new and better ones.

About The Author

David Boggs is one of the inventors of Ethernet. He is a member of the research staff of the Computer Science Laboratory at Xerox's Palo Alto Research Center.

He holds a Bachelor's degree in Electrical Engineering from Princeton University and a Master's degree from Stanford University, where he is currently pursuing a Ph.D.

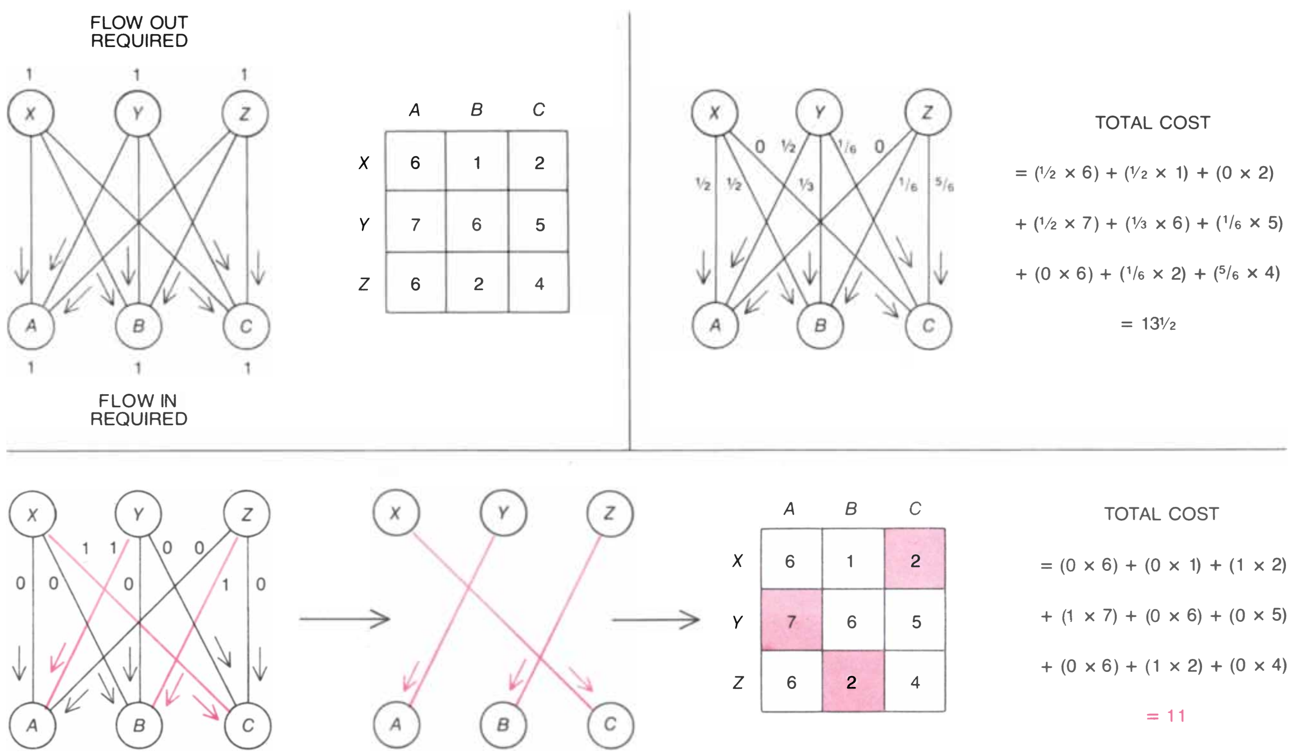
straint on the supply of corn. This is precisely how the simplex method recognizes that the objective function can be improved along this edge. In this case the edge connects the vertex (0,32) with the vertex (12,28) of the polygon.

Even in small problems of linear programming the feasible region can have an enormous number of vertexes, and there is the possibility that the simplex method will require an enormous number of iterations, or vertex-to-vertex moves, in order to solve the problem. Usually, however, the selective search encounters only a vanishingly small fraction of all the vertexes. People who regularly solve problems having from 2,000 to 3,000 resource constraints and from 10,000 to 15,000 decision variables find that such problems generally take only from a few minutes to a few hours to run on a large computer. One would like to be able to make a more precise statement concerning the efficiency of the simplex method.

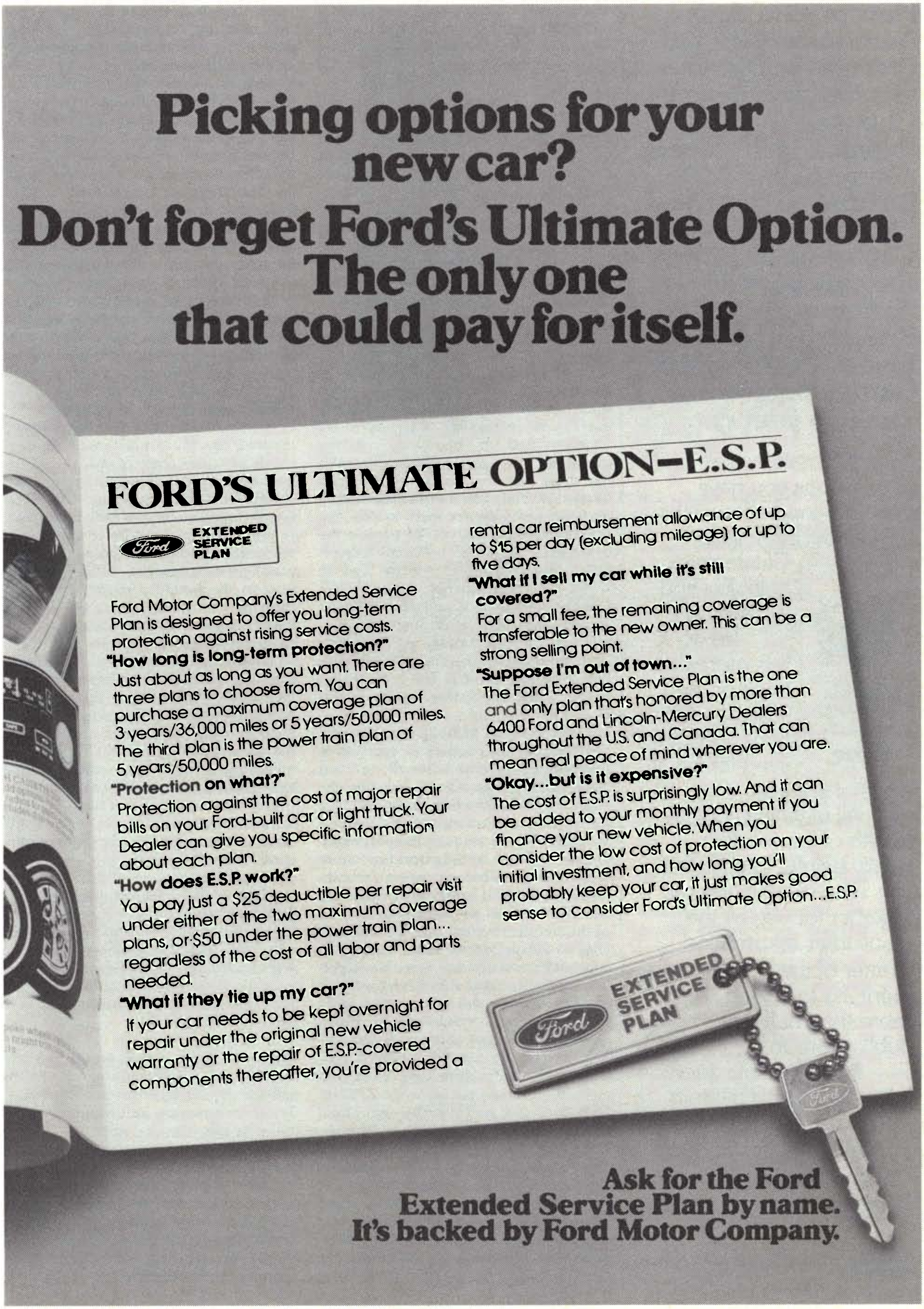
One way of convincing a skeptic that an algorithm is efficient is to provide some kind of guarantee. The guarantee takes the form of a function G and a statement that on a problem of size s the running time of the algorithm will not be greater than $G(s)$. The size s of a problem is a measure of the amount of information (often expressed as a number of binary digits) needed to specify the input data. This is called a worst-case analysis, since the value of the guarantee is determined by the most troublesome problems of size s the algorithm might be asked to solve. The running time actually required to solve a problem of size s depends on the characteristics of the particular computer. To make the measure $G(s)$ machine-independent the function is usually expressed not in minutes or hours but as the maximum number of elementary arithmetical operations (such as addition, multiplication and comparison) that must be done by the algorithm.

If $G(s)$ is appropriately small for the values of s one expects to encounter, one would be inclined to acknowledge that the proposed algorithm is efficient. For example, Harold W. Kuhn of Princeton University has devised an algorithm for the assignment problem that requires at most cn^3 elementary operations to solve any n-by-n problem, where c is a small constant. When n is equal to 2, 3, 4 or 5, the algorithm can actually take longer than enumeration (although with a computer neither method would take longer than the blink of an eye). Once n increases beyond this range, $n!$, which measures the size of the enumeration, quickly surpasses cn^3. In fact, $n!$ grows so explosively that, as I have shown, no existing computer could carry out the computation. On the other hand, n^3 grows more slowly as n increases. For n equal to 35 Kuhn's algorithm can still solve the problem in the blink of an eye. Furthermore, as n increases from 35 to 36, the computational burden on Kuhn's algorithm increases by a factor of about $(36/35)^3$, or approximately 1.09. The work required for enumeration increases by a factor of 36.

Computer scientists have a particular interest in algorithms that are said to be polynomially bounded, where the function G is a polynomial function of s. A polynomial function of s is a sum of various terms; in each term s is raised to some constant power and multiplied by a constant coefficient. If s is sufficiently large, any algorithm whose worst-case guarantee increases as $s!$ or as any exponential function of s, say 2^s, will run more slowly than any polynomi-



NETWORK-FLOW PROBLEMS are a special class of problems called integer-programming problems that yield to linear-programming techniques. In an integer-programming problem the decision variables are not divisible: they must assume integer values only. In general such problems are in the set *NP*-complete and so are regarded as hard. The cost of each link of the network (*diagram at upper left*) is given by the matrix of numbers. Note that the matrix is identical with a possible matrix for a 3-by-3 assignment problem. A solution of the assignment problem must be integer-valued, however, and it appears that the constraints on the network flow are not sufficient to guarantee this result. Many fractional flows meet the constraints (*diagram at upper right*). Nevertheless, extremal solutions of network-flow problems found by linear-programming methods are always integer-valued, provided that the flows both into and away from each node are integer-valued. Hence the assignment problem can be interpreted as a special case of the network-flow problem (*lower diagram*).


Picking options for your new car?
Don't forget Ford's Ultimate Option.
The only one that could pay for itself.
FORD'S ULTIMATE OPTION–E.S.P.
Ford EXTENDED SERVICE PLAN
Ford Motor Company's Extended Service Plan is designed to offer you long-term protection against rising service costs.
"How long is long-term protection?"
Just about as long as you want. There are three plans to choose from. You can purchase a maximum coverage plan of 3 years/36,000 miles or 5 years/50,000 miles. The third plan is the power train plan of 5 years/50,000 miles.
"Protection on what?"
Protection against the cost of major repair bills on your Ford-built car or light truck. Your Dealer can give you specific information about each plan.
"How does E.S.P. work?"
You pay just a $25 deductible per repair visit under either of the two maximum coverage plans, or $50 under the power train plan... regardless of the cost of all labor and parts needed.
"What if they tie up my car?"
If your car needs to be kept overnight for repair under the original new vehicle warranty or the repair of E.S.P.-covered components thereafter, you're provided a rental car reimbursement allowance of up to $15 per day (excluding mileage) for up to five days.
"What if I sell my car while it's still covered?"
For a small fee, the remaining coverage is transferable to the new owner. This can be a strong selling point.
"Suppose I'm out of town..."
The Ford Extended Service Plan is the one and only plan that's honored by more than 6400 Ford and Lincoln-Mercury Dealers throughout the U.S. and Canada. That can mean real peace of mind wherever you are.
"Okay...but is it expensive?"
The cost of E.S.P. is surprisingly low. And it can be added to your monthly payment if you finance your new vehicle. When you consider the low cost of protection on your initial investment, and how long you'll probably keep your car, it just makes good sense to consider Ford's Ultimate Option...E.S.P.
Ford EXTENDED SERVICE PLAN
Ask for the Ford Extended Service Plan by name. It's backed by Ford Motor Company.

ally bounded algorithm. Consequently computer scientists consider polynomial boundedness a theoretical criterion of algorithmic efficiency.

Because the simplex method performs well in practice one might guess it is a polynomially bounded algorithm. Indeed, practitioners have found that a good rule of thumb is that the number of simplex iterations increases as a linear function of the number of constraints. In standard implementations, however, the simplex method is not polynomially bounded. Victor La Rue Klee, Jr., of the University of Washington and George J. Minty, Jr., of Indiana University have constructed an infinite family of linear-programming problems in which the simplex method is essentially as bad as enumeration.

That an algorithm so successful in practice could be regarded as other than efficient in the theoretical sense may seem perplexing. Keep in mind, however, that the theoretical criterion examines worst-case results. Examples like those first constructed by Klee and Minty that cause the simplex method to perform badly do not seem to occur in practice, at least not yet. To explain the divergence between practical experience with the simplex method and its worst-case performance, investigators have attempted to demonstrate that in some probabilistic sense the bad examples really are "pathological." The demonstration faces considerable difficulties, although Dantzig has recently reported some interesting results in this direction.

There is some flexibility in how the simplex method chooses at each iteration among several adjacent vertexes when each vertex would improve the value of the objective function. The Klee-Minty construction shows that one popular strategy leads to nonpolynomial results. Other investigators have since constructed similar examples for other strategies. It is not known, however, whether there is some implementation of the simplex method that is polynomially bounded. Indeed, until recently it was not known whether there is any polynomially bounded algorithm for linear programming; it was an unsolved problem of great theoretical interest. Finally L. G. Khachian of the Academy of Sciences of the U.S.S.R. showed how an algorithm developed by three other Russian mathematicians, N. Z. Shor, D. B. Iudin and A. S. Nemirovskii, could be implemented in polynomial time.

Because the work of Shor, Iudin and Nemirovskii does not assume linearity the new algorithm differs noticeably from the simplex method. Their algorithm is the one called the ellipsoid method. It begins with a very large ellipsoid centered at the origin. In general the initial ellipsoid is a sphere with a radius large enough to ensure that it contains the optimum solution. The algorithm proceeds to construct progressively smaller ellipsoids, each of which is also known to contain the optimum solution. The volume of the successive ellipsoids shrinks quickly enough to guarantee that their centers converge on the optimum solution. One need not draw the ellipsoids in order to employ the algorithm; there are simple recursive formulas that describe them. When one accounts for the work needed to construct each ellipsoid from its predecessor, the bound on the total number of operations is on the order of $(m + n)n^6l$, where m is the number of constraints, n is the number of decision variables and l is the logarithm to the base 2 of the largest coefficient in the problem data.

This expression is a polynomial bound, but, like any such bound, it can still grow rather large for large values of m, n and l. Could it be that the ellipsoid method, like the simplex method, does much better in practice than its worst-case bound would predict? Preliminary results are not encouraging. Computational testing so far has shown extremely slow convergence for the ellipsoid method. Remarkably simple families of problems can cause the method to run as long as the bound predicts. This is in contrast with the complicated form of the problems that have been constructed to defeat the simplex method.

Another major difficulty of the ellipsoid method concerns memory requirements. Very large linear-programming problems tend in practice to be sparse: the proportion of nonzero coefficients in the data is usually small. The simplex method can be implemented so that sparseness is maintained as the data are manipulated during the execution. This is a crucial property, since one need only store in the computer the nonzero entries. There is no apparent way to exploit sparseness in the ellipsoid method.

Although the prospects for a practical implementation of the ellipsoid method seem dim at the moment, one must see what develops. In the short time since Khachian's note considerable work has been done, with some substantial improvements. The difficulties noted here do appear formidable.

Linear programming is a practical tool of great theoretical interest. To the mathematician it constitutes a theory of linear inequalities, an elegant counterpoint to the classical theory of linear equations. To the government and corporate planner it is a valuable aid in decision making and long-range planning. Perhaps the most important consequence of Khachian's theoretical breakthrough will be increased interest in both practical and theoretical aspects of linear programming.

FROM HERE TO SERENDIPITY.

In the land of Serendip, so an ancient fairy tale says, people who travelled unknown roads made wonderful discoveries when they least expected to.

To some, the fable of Serendip is just another bedtime story, but at the IBM research laboratories it's a story that hits close to home.

Recently, for instance, IBM scientists were experimenting with X-rays to explore new ways to make microscopic computer circuits.

A colleague wondered if this new technique might also help medical researchers look at living cells.

One thing led to another.

Working with biologists from the National Institutes of Health, the IBM team tested their idea by photographing an ordinary human blood platelet at a resolution of 1/5,000,000th of-an-inch.

What they saw stopped them cold.

For they found themselves staring at internal cell structures no one had ever seen before. Structures that may tell us more about how cells work.

The scientists from the N.I.H. are very interested.

And so are we.

True, it's not the kind of discovery you might expect from IBM. But that's just the point.

Because when you give good minds the freedom to follow their best hunches, you learn to expect the unexpected.

IBM®