ORF 307: Lecture 1

Linear Programming Chapter 1

Introduction

Robert Vanderbei

February 4, 2014

Slides last edited on February 3, 2014
Engineering is the process of taking the discoveries from science... implementing them as practical devices, and then ...

making them better, ...

and better, ...

and better.

This is optimization.
Express an *objective function* to be *minimized* or *maximized* in terms of one independent variable.

Differentiate with respect to this variable.

Set derivative equal to zero.

Solve for the independent variable.

If in doubt as to whether it’s a max, min, or saddle point, take second derivative and look at its sign.

If the independent variable is restricted to lying in an interval of the real line, check the endpoints—the optimal solution could be there.
Snell’s Law
An Example: Fermat’s Principle

Consider a light source in one medium and a detector (eyeball) in a second medium.

Fermat’s principle says that the light reaching the detector follows the \textit{minimum} time path from the source to the detector.

Assuming that the speed of light in the two media are \( v_1 \) and \( v_2 \), the time is given by:

\[
T(x) = \frac{\sqrt{(a_1 - x)^2 + b_1^2}}{v_1} + \frac{\sqrt{(x - a_2)^2 + b_2^2}}{v_2}
\]

Derivative is:

\[
\frac{dT}{dx} = -\frac{1}{v_1} \frac{a_1 - x}{\sqrt{(a_1 - x)^2 + b_1^2}} + \frac{1}{v_2} \frac{x - a_2}{\sqrt{(x - a_2)^2 + b_2^2}} = 0.
\]

Rearranging, we get \textit{Snell’s Law}: \( n_1 \sin \theta_1 = n_2 \sin \theta_2 \), where \( n_1 = c/v_1 \) and \( n_2 = c/v_2 \).
Solving it on the Computer (using AMPL via NEOS)

# Snell’s law of refraction obtained by minimizing the time for the light to get from point a to b.
param info symbolic, := "File name: snell.mod; Author: R.J. Vanderbei"

display info;

param a{1..2};
param b{1..2};
param v1;
param v2;

var x;

minimize time: sqrt((a[1]-x)^2 + b[1]^2)/v1 + sqrt((x-a[2])^2 + b[2]^2)/v2;

data;
param a :=
    1 1
    2 -1 ;

param b :=
    1 1
    2 -1 ;

param v1 := 1;
param v2 := 0.8;

solve;
display x;
display ((a[1]-x)/sqrt((a[1]-x)^2 + b[1]^2)) / ((x-a[2])/sqrt((x-a[2])^2 + b[2]^2));
quit;
• The language is called *AMPL*, which stands for *A Mathematical Programming Language*.

• The book describing the language is called “AMPL” by Fourer, Gay, and Kernighan. Amazon.com “rents” it for $27.28. It is also available for free at [http://www.ampl.com/BOOK/download.html](http://www.ampl.com/BOOK/download.html).

There’s a link to the AMPL website on the course webpage:

http://orfe.princeton.edu/~rvdb/307/lectures.html

• There are also online tutorials:

  - [https://webspace.utexas.edu/sdb382/www/teaching/ce4920/ampl_tutorial.pdf](https://webspace.utexas.edu/sdb382/www/teaching/ce4920/ampl_tutorial.pdf)
  - [http://www2.isye.gatech.edu/~jswann/teaching/AMPLTutorial.pdf](http://www2.isye.gatech.edu/~jswann/teaching/AMPLTutorial.pdf)

  – Google: “AMPL tutorial” for several more.
NEOS Info

NEOS is the *Network Enabled Optimization Server* supported by our federal government and located at the *University of Wisconsin*. To submit an AMPL model to NEOS...

- visit [http://www.neos-server.org/neos/](http://www.neos-server.org/neos/),
- click on the [Neos](http://www.neos-server.org/neos/) icon,
- scroll down to the *Nonlinearly Constrained Optimization* list,
- click on LOQO [AMPL input],
- scroll down to *Model File:*,
- click on *Choose File*,
- select a file from your computer that contains an AMPL model,
- check box: *Put in priority queue*,
- scroll down to *e-mail address:*,
- type in your email address, and
- click *Submit to NEOS*.

Piece of cake!
Job 2630833 sent to neos-4.neos-server.org

password: H1LzdusB

-------- Begin Solver Output ---------
Executing /opt/neos/Drivers/loqo-ampl/loqo-driver.py at time: 2014-02-03 08:57:44.063865
File exists
You are using the solver loqo.
Executing AMPL.
processing data.
processing commands.
info = 'File name: snell.mod; Author: R.J. Vanderbei'

1 variable, all nonlinear
0 constraints
1 nonlinear objective; 1 nonzero.

LOQO 7.01: optimal solution (7 iterations, 13 evaluations)
primal objective 3.162737989
dual objective 3.162737989
x = -0.214687

First Problem of First Assignment

Make a full three-dimensional version of the Snell’s Law problem. Solve it on NEOS.
Freshman Calculus

• One variable
• Nonlinear objective function
• Sometimes variable constrained to an interval

ORF 307

• Thousands of variables
• Linear objective function
• Linear (equality and inequality) constraints
Diet Problem
The McDonald’s Diet Problem

*In words:*

Minimize:
the cost (or calories) of eating at McDonalds.

Subject to:
the total amounts of foods and nutrients fall between certain minimum and maximum values.
An AMPL Model

set NUTR;
set FOOD;

param cost {FOOD};
param f_min {FOOD} default 0;
param f_max {FOOD} default Infinity;

param n_min {NUTR} default 0;
param n_max {NUTR} default Infinity;

param amt {NUTR,FOOD};

# --------------------------------------------------------
var Buy {j in FOOD} integer, >= f_min[j], <= f_max[j];
# --------------------------------------------------------
minimize Total_Cost: sum {j in FOOD} cost[j] * Buy[j];

minimize Nutr_Amt {i in NUTR}: sum {j in FOOD} amt[i,j] * Buy[j];

# --------------------------------------------------------
subject to Diet {i in NUTR}:
    n_min[i] <= sum {j in FOOD} amt[i,j] * Buy[j] <= n_max[i];
The Data

param: FOOD: cost f_min f_max :=
"Quarter Pounder w/ Cheese" 1.84 . .
"McLean Deluxe w/ Cheese" 2.19 . .
"Big Mac" 1.84 . .
"Filet-O-Fish" 1.44 . .
"McGrilled Chicken" 2.29 . .
"Fries, small" .77 . .
"Sausage McMuffin" 1.29 . .
"1% Lowfat Milk" .60 . .
"Orange Juice" .72 . . ;

param: NUTR: n_min n_max :=
Cal  2000   .
Carbo 350  375
Protein 55  .
VitA 100  .
VitC 100  .
Calc 100  .
Iron 100  . ;

param amt (tr): Cal  Carbo  Protein  VitA  VitC  Calc  Iron :=
"Quarter Pounder w/ Cheese" 510  34  28  15  6  30  20
"McLean Deluxe w/ Cheese" 370  35  24  15  10  20  20
"Big Mac" 500  42  25  6  2  25  20
"Filet-O-Fish" 370  38  14  2  0  15  10
"McGrilled Chicken" 400  42  31  8  15  15  8
"Fries, small" 220  26  3  0  15  0  2
"Sausage McMuffin" 345  27  15  4  0  20  15
"1% Lowfat Milk" 110  12  9  10  4  30  0
"Orange Juice" 80  20  1  2  120  2  2 ;

Complete table of nutritional data can be found at:

teal: ampl

ampl: model diet1.mod;
ampl: data diet1.dat;
ampl: solve;
MINOS 5.4: ignoring integrality of 9 variables
MINOS 5.4: optimal solution found.
8 iterations, objective 14.8557377

ampl: display Buy;
Buy [*] :=
'Quarter Pounder w/ Cheese'  4.38525
  'Fries, small'       6.14754
  '1% Lowfat Milk'    3.42213

ampl: display n_min, Diet.body, n_max;
:  n_min  Diet.body  n_max :=
Cal   2000  3965.37 Infinity
Carbo  350   350      375
Protein  55  172.029 Infinity
VitA   100   100      Infinity
VitC   100  132.213 Infinity
Calc    100  234.221 Infinity
Iron    100   100      Infinity

Cheap, but 4000 calories!
Second Run

Put an upper limit on Calories.

ampl: let n_max["Cal"] := 2500;

ampl: solve;
MINOS 5.4: ignoring integrality of 9 variables
MINOS 5.4: optimal solution found.
9 iterations, objective 16.67097416

ampl: display Buy;
Buy [*] :=
'Quarter Pounder w/ Cheese' 0.231942
'McLean Deluxe w/ Cheese' 3.85465
'1% Lowfat Milk' 2.0433
'Orange Juice' 9.13408
;

ampl: display n_min,Diet.body,n_max;
: n_min Diet.body n_max :=
Cal   2000   2500   2500
Carbo 350    350    375
Protein 55    126.53  Infinity
VitA  100    100    100
VitC  100    1144.2  Infinity
Calc  100    163.618 Infinity
Iron  100    100    100
;

Calories are down, cost is up, diet looks better.
Try for a 2000 Calorie diet.

ampl: let n_max["Cal"] := 2000;

ampl: solve;
MINOS 5.4: ignoring integrality of 9 variables
MINOS 5.4: infeasible problem.
2 iterations
Objective = Total_Cost

No can do!
How about ignoring cost and minimizing calories:

```ampl
ampl: let n_max["Cal"] := Infinity;

ampl: objective Nutr_Amt["Cal"];

ampl: solve;
MINOS 5.4: ignoring integrality of 9 variables
MINOS 5.4: optimal solution found.
3 iterations, objective 2466.981132

ampl: display Total_Cost;
Total_Cost = 16.7453

ampl: display Buy;
Buy [*] :=
   'McLean Deluxe w/ Cheese'  4.08805
   '1% Lowfat Milk'           2.04403
   'Orange Juice'             9.1195
;
```

Minimum number of Calories is 2467 at a cost of $16.75.
Let’s Add Some Variety

ampl: let {j in FOOD} f_max[j] := 2;

ampl: solve;
MINOS 5.4: ignoring integrality of 9 variables
MINOS 5.4: optimal solution found.
6 iterations, objective 16.76576923

ampl: display Buy;
Buy [*] :=
'Quarter Pounder w/ Cheese' 2
'McLean Deluxe w/ Cheese' 2
'Big Mac' 2
'Fries, small' 1.42308
'Sausage McMuffin' 1
'1% Lowfat Milk' 2
'Orange Juice' 2

ampl: display n_min,Diet.body,n_max;
: n_min Diet.body n_max :=
Cal 2000 3798.08 Infinity
Carbo 350 350 375
Protein 55 193.269 Infinity
VitA 100 100 Infinity
VitC 100 305.346 Infinity
Calc 100 234 Infinity
Iron 100 141.846 Infinity

More interesting, cost is up a bit, Calories went down (compared to original).
Keep the Variety but Minimize Calories.

ampl: objective Nutr_Amt["Cal”];
ampl: solve;
MINOS 5.4: ignoring integrality of 9 vars
MINOS 5.4: optimal solution found.
6 iterations, objective 3488.286853

ampl: display Total_Cost;
Total_Cost = 17.2484

ampl: display Buy;
Buy [*] :=
'Quarter Pounder w/ Cheese’ 1.95219
'McLean Deluxe w/ Cheese’ 2
Filet-O-Fish 0.358566
'McGrilled Chicken’ 2
'Fries, small’ 2
'1% Lowfat Milk’ 2
'Orange Juice’ 2 ;

ampl: display n_min,Diet.body,n_max;
: n_min Diet.body n_max :=
Cal 2000 3488.29 Infinity
Carbo 350 350 375
Protein 55 195.681 Infinity
VitA 100 100 Infinity
VitC 100 339.713 Infinity
Calc 100 197.944 Infinity
Iron 100 106.629 Infinity ;

Almost 3500 Calories. Wow!
Whole Number Solution

ampl: option solver cplex;

ampl: solve;
CPLEX 2.1: optimal integer solution; objective 15.05
129 simplex iterations
112 branch-and-bound nodes
Objective = Total_Cost

ampl: display Buy;

Buy [*] := ’Quarter Pounder w/ Cheese’ 4
        Filet-O-Fish 1
        ’Fries, small’ 5
        ’1% Lowfat Milk’ 4
;

ampl: display n_min,Diet.body,n_max;

:                n_min Diet.body               n_max       :=
Cal  2000  3950    Infinity
Carbo 350   352     375
Protein 55    177    Infinity
VitA  100    102    Infinity
VitC  100    115    Infinity
Calc 100    255    Infinity
Iron 100    100    Infinity
;

Summary

<table>
<thead>
<tr>
<th>varied?</th>
<th>integer?</th>
<th>calories</th>
<th>cost</th>
<th>varied?</th>
<th>integer?</th>
<th>calories</th>
<th>cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>no</td>
<td>no</td>
<td>3965</td>
<td>14.86</td>
<td>no</td>
<td>yes</td>
<td>2467</td>
<td>16.75</td>
</tr>
<tr>
<td>no</td>
<td>yes</td>
<td>3950</td>
<td>15.05</td>
<td>no</td>
<td>yes</td>
<td>2500</td>
<td>17.16</td>
</tr>
<tr>
<td>yes</td>
<td>no</td>
<td>3798</td>
<td>16.77</td>
<td>yes</td>
<td>yes</td>
<td>3488</td>
<td>17.25</td>
</tr>
<tr>
<td>yes</td>
<td>yes</td>
<td>3630</td>
<td>17.04</td>
<td>yes</td>
<td>yes</td>
<td>3530</td>
<td>17.49</td>
</tr>
</tbody>
</table>

Moral: Often there is no one clear choice of a correct model. The objective might be debatable as might the constraints.
Linear Programming
**Standard Form.**

maximize \[ c_1 x_1 + c_2 x_2 + \cdots + c_n x_n \]

subject to \[ a_{11} x_1 + a_{12} x_2 + \cdots + a_{1n} x_n \leq b_1 \]
\[ a_{21} x_1 + a_{22} x_2 + \cdots + a_{2n} x_n \leq b_2 \]
\[ \vdots \]
\[ a_{m1} x_1 + a_{m2} x_2 + \cdots + a_{mn} x_n \leq b_m \]
\[ x_1, x_2, \ldots, x_n \geq 0. \]

Why it’s hard:

- Lots of variables (\( n \) of ’em).
- Lots of “boundaries” to check (the inequalities).

Why it’s not impossible:

- All expressions are linear.
Climate Change
Average Daily Temperatures at McGuire AFB

Date

Avg. Temp. (F)


0 10 20 30 40 50 60 70 80 90 100

Avg. Temp. (F)

0 10 20 30 40 50 60 70 80 90 100

Date

Running Average

Regressions: Least Abs. Dev. (solid) and Least Squares (dashed)
Regression Models

Let $T_d$ denote the average temperature in degrees Fahrenheit on year $d \in D$ where $D$ is the set of years from 1955 to 2010.

$$T_d = x_0 + x_1 d \quad \text{linear trend}$$

$$+ \varepsilon_d. \quad \text{error term}$$

The parameters $x_0$ and $x_1$ are unknown regression coefficients.

Either

$$\min \sum_{d \in D} |\varepsilon_d| \quad \text{Least Absolute Deviations (LAD)}$$

or

$$\min \sum_{d \in D} \varepsilon_d^2 \quad \text{Least Squares}$$
Search For Earth-Like Planets
One Petal Down, Fifteen To Go
What You Should Expect To Get From This Course

Learn how to formulate optimization problems.

Learn to distinguish easy problems from hard ones from impossible ones.

Learn some of the theory of Linear Programming (Duality Theory!).

Learn how to express optimization problems in AMPL and solve them on NEOS.
History of Optimization

- Origins date back to Newton, Liebnitz, Lagrange, etc.

- The subfield of *Linear Programming* was created by George Dantzig, John von Neumann (Princeton), and Leonid Kantorovich in the 1940’s.

- In 1947, Dantzig invented the *Simplex Method*.

- In 1979, L. Khachain found a new *efficient* algorithm for linear programming. It was terribly slow.

- In 1984, Narendra Karmarkar discovered yet another new *efficient* algorithm for linear programming. It proved to be a strong competitor of the simplex method.
Breakthrough in Problem Solving

By JAMES GLEICK

A 28-year-old mathematician at A.T.
Bell Laboratories has made a startling theoretical breakthrough in the solving of systems of equations that often grow too vast and complex for the most powerful computers.

The discovery, which is to be formally published next month, is already circulating rapidly through the mathematical world. It has also set off a deluge of inquiries from brokerage houses, oil companies and airlines, industries with millions of dollars at stake in problems known as linear programming.

Faster Solutions Seen

These problems are fiendishly complicated systems, often with thousands of variables. They arise in a variety of commercial and government applications, ranging from allocating time on a communications satellite to routing millions of telephone calls over long distances, or whenever a limited, expensive resource must be spread most efficiently among competing users. And investment companies use them in creating portfolios with the best mix of stocks and bonds.

The Bell Labs mathematician, Dr. Narendra Karmarkar, has devised a radically new procedure that may speed the routine handling of such problems by businesses and Government agencies and also make it possible to tackle problems that are now far out of reach.

"This is a path-breaking result," said Dr. Ronald L. Graham, director of mathematical sciences for Bell Labs in Murray Hill, N.J.

"Science has its moments of great progress, and this may well be one of them." Because problems in linear programming can have billions or more possible answers, even high-speed computers cannot check every one. So computers must use a special procedure, an algorithm, to examine as few answers as possible before finding the best one — typically the one that minimizes cost or maximizes efficiency.

A procedure devised in 1947, the simplex method, is now used for such problems,

Continued on Page A19, Column 1

THE NEW YORK TIMES, November 19, 1984

Folding the Perfect Corner

A young Bell scientist makes a major math breakthrough

Every day 1,200 American Airlines jets crisscross the U.S., Mexico, Canada and the Caribbean, stopping in 110 cities and bearing over 80,000 passengers. More than 4,000 pilots, copilots, flight personnel, maintenance workers and baggage carriers are shuffled among the flights; a total of 3.6 million gal. of high-octane fuel is burned. Nuts, bolts, altimeters, landing gears and the like must be checked at each destination. And while performing these scheduling gymnastics, the company must keep a close eye on costs, projected revenue and profits.

Like American Airlines, thousands of companies must routinely untangle the myriad variables that complicate the efficient distribution of their resources. Solving such monstrous problems requires the use of an abstruse branch of mathematics known as linear programming. It is the kind of math that has frustrated theoreticians for years, and even the fastest and most powerful computers have had great difficulty juggling the bits and pieces of data. Now Narendra Karmarkar, a 28-year-old Indian-born mathematician at Bell Laboratories in Murray Hill, N.J., has cracked the puzzle of linear programming by devising a new algorithm, a step-by-step mathematical formula. He has translated the procedure into a program that should allow computers to track a greater combination of tasks than ever before and in a fraction of the time.

"Breakthrough is one of the most abused words in science," says Ronald Graham, director of mathematical sciences at Bell Labs. "But this is one situation where it is truly appropriate."

Unlike most advances in theoretical mathematics, Karmarkar's work will have an immediate and major impact on the real world.

"Before the Kannarkar method, linear equations could be solved only in a cumbersome fashion, ironically known as the simplex method, devised by Mathematician George Dantzig in 1947. Problems are conceived of as giant geodesic domes with thousands of sides. Each corner of a facet on the dome..."
Karmarkar Algorithm Proves Its Worth

Less than two years after discovery of a mathematical procedure that Bell Labs said could solve a broad range of complex business problems 50 to 100 times faster than current methods, AT&T is filing for patents covering its use. The Karmarkar algorithm, which drew headlines when discovered by researcher Narendra Karmarkar, will be applied first to AT&T's long-distance network. Thus far, Bell Labs has verified the procedure's capabilities in developing plans for new fiber-optic transmission and satellite capacity linking 20 countries bordering the Pacific Ocean. That jointly owned network will be built during the next 10 years. Planning requires a tremendous number of "what if" scenarios involving 45,000 variables describing transmission capacity, location and construction schedules, all juggled amid political considerations of each connected country.

The Karmarkar algorithm was able to solve the Pacific Basin problem in four minutes, against 80 minutes by the method previously used, says Neil Dinn, head of Bell Labs' international transmission planning department. The speedier solutions will enable international committees to agree on network designs at one meeting instead of many meetings stretched out over months.

AT&T now is using the Karmarkar procedure to plan construction for its domestic network, a problem involving 800,000 variables. In addition, the procedure may be written into software controlling routing of domestic phone calls, boosting the capacity of AT&T's current network.

THE STARTLING DISCOVERY
BELL LABS KEPT IN THE SHADOWS

Now its breakthrough mathematical formula could save business millions

I t happened all too often in science. An obscure researcher announces a stunning breakthrough and achieves instant fame. But when other scientists try to repeat his results, they fail. Fame quickly turns to notoriety, and eventually the episode is all but forgotten. That seemed to be the case with Narendra Karmarkar, a young scientist at AT&T Bell Laboratories. In late 1984 the 28-year-old researcher cracked one of the thorniest aspects of computer-aided problem-solving. If so, his feat would have meant an instant windfall for many big companies. It could have also pointed to better software for small companies that use computers to help manage their business.

Karmarkar said he had discovered a quick way to solve problems so hideously complicated that they often defy even the most powerful supercomputers. Such problems involve a broad range of business activities, from assessing risk factors in stock portfolios to drawing up production schedules in factories. Just about any company that distributes products through more than a handful of warehouses bumps into such problems when calculating the cheapest routes for getting goods to customers. Even when the problems aren't terribly complex, solving them can chew up so much computer time that the answer is useless before it's found.

linear programming (LP) has evolved, and most scientists thought that was as far as it could go. Sure enough, when other researchers independently tried to test Karmarkar's process, their results were disappointing. At scientific conferences skeptics attacked the algorithm's validity as well as Karmarkar's veracity.

But this story may end with a different twist. Other scientists weren't able to duplicate Karmarkar's work. It turns out, because his employer wanted it that way. Vital details about how best to translate the algorithm, whose mathematical notations run on for about 20 printed pages, into digital computer code were withheld to give Bell Labs a head start at developing commercial products.

Following the breakup of American Telephone Telegraph Co. in January, 1984, Bell Labs was no longer prevented from exploiting its research for profit. While the underlying concept could not be patented or copyrighted because it is pure knowledge, any computer programs that AT&T developed to implement the procedure can be protected.

Now, AT&T may soon be selling the first product based on Karmarkar's work—the U.S. Air Force. It includes a multiprocessor computer from Alliant Computer Systems Corp. and a software version of Karmarkar's algorithm that has been optimized for high-speed parallel processing. The system would be installed at St. Louis' Scott Air Force Base, headquarters of the Military Airlift Command (MAC). Neither party will comment on the deal's cost or where the negotiations stand, but the Air Force's interest is easy to fathom.

JUGGLING ACT. On a typical day, thousands of planes ferry cargo and passengers among air fields scattered around the world. To keep those jets flying, MAC
AT&T Markets Problem Solver, Based On Math Whiz's Find, for $8.9 Million

By ROGER LOWENSTEIN

NEW YORK—American Telephone & Telegraph Co. has called its math whiz, Narendra Karmarkar, a latter-day Isaac Newton. Now, it will see if he can make the firm some money.

Four years after AT&T announced an "astonishing" discovery by the Indian-born Mr. Karmarkar, it is marketing an $8.9 million problem solver based on his invention.

Dubbed Korbx, the computer-based system is designed to solve major operational problems of both business and government. AT&T predicts "substantial" sales for the product, but outsiders say the price is high and point out that its commercial viability is unproven.

"At $9 million a system, you're going to have a small number of users," says Thomas Magnanti, an operations-research specialist at Massachusetts Institute of Technology. "But for very large-scale problems, it might make the difference."

Korbx uses a unique algorithm, or step-by-step procedure, invented by Mr. Karmarkar, a 32-year old, an AT&T Bell Laboratories mathematician.

"It's designed to solve extremely difficult or previously unsolvable resource-allocation problems—which can involve hundreds of thousands of variables—such as personnel planning, vendor selection, and equipment scheduling," says Aristides Fronistas, president of an AT&T division created to market Korbx.

Potential customers might include an airline trying to determine how to route many planes between numerous cities and an oil company figuring how to feed different grades of crude oil into various refineries and have the best blend of refined products emerge.

AT&T says that fewer than 10 companies, which it won't name, are already using Korbx. It adds that, because of the price, it is targeting only very large companies—mostly in the Fortune 100.

Korbx "won't have a significant bottom-line impact initially," for AT&T, though it might in the long term, says Charles Nichols, an analyst with Bear, Stearns & Co. "They will have to expose it to users and demonstrate it uses.

AMR Corp.'s American Airlines says it's considering buying AT&T's system. Like other airlines, the Fort Worth, Texas, carrier has the complex task of scheduling pilots, crews and flight attendants on thousands of flights every month.

Thomas M. Cook, head of operations research at American, says, "Every airline has programs that do this. The question is: Can AT&T do it better and faster? The jury is still out."

The U.S. Air Force says it is considering using the system at the Scott Air Force Base in Illinois.

One reason for the uncertainty is that AT&T has, for reasons of commercial secrecy, deliberately kept the specifics of Mr. Karmarkar's algorithm under wraps.

"I don't know the details of their system," says Eugene Bryan, president of Decision Dynamics Inc., a Portland, Ore., consulting firm that specializes in linear programming, a mathematical technique that employs a series of equations using many variables to find the most efficient way of allocating resources.

Mr. Bryan says, though, that if the Karmarkar system works, it would be extremely useful. "For every dollar you spend on optimization," he says, "you usually get them back many-fold."

AT&T has used the system in-house to help design equipment and routes on its Pacific Basin system, which involves 22 countries. It's also being used to plan AT&T's evolving domestic network, a problem involving some 800,000 variables.