from *SIAM News*, Volume 36, Number 4, May 2003

Some People Just Know How to Optimize

By Barry A. Cipra

"Some people just know how to fly," Northwest Airlines proclaimed in TV ads in the 1990s. But according to a week-long workshop on travel and transportation held in November at the Institute for Mathematics and Its Applications at the University of Minnesota, if an airline—or anyone in the business of moving people or products from place to place—wants to make money, it also needs to know how to optimize.

The workshop, organized by Cynthia Barnhart of MIT, Ranga Anbil of OHM Technologies, in Cincinnati, Ellis Johnson of Georgia Tech, and William Pulleyblank of IBM Research, included talks on aircraft and aircrew scheduling, railroad reliability, truck routing, mass transit systems, and even the future of online grocery shopping. Speakers described new opportunities for optimization theory. The payoff, they say, could be enormous: By accurately modeling their operations, transportation-based businesses could save billions of dollars every year—in effect creating billions of dollars of instant profit.

Jet Sets

He saw my complications
And he mirrored me back simplified
—*Joni Mitchell,* "Refuge of the Roads"

Take aircrew scheduling—an aspect of flying passengers rarely think about. David Ryan of the University of Auckland described an optimization approach he and colleagues have taken to the problem of who's going to fly which planes when and where. Their solutions, undertaken for Air New Zealand, save the airline an estimated $8 million per year.

Aircrew scheduling is done in two steps. Starting with a (supposedly inviolate) schedule of flights, an airline first incorporates them into "tours of duty." Each tour of duty (ToD) is a sequence of flight segments and rest periods, typically starting and stopping at the same airport (since most pilots and flight attendants prefer to go home when their shifts are over). Some amount of rest is required by law or contract, and some is unavoidable (you have to allow time for passengers to get on and off, or what's the point of flying?), but some can be eliminated by smart planning. This is the tour-of-duty problem: minimizing crew-related costs while covering all flight segments.

The second step is known as rostering: deciding, for all employees, which tours of duty they will fly during, say, a two-week or month-long period, which days they'll have off, which days they'll go in for training, and so forth. Each assignment, known as a line of work (LoW), must satisfy the airline's legal and contractual obligations for total flight hours and time off. As one might expect, some LoWs are more desirable than others, because of the way they space days off or because some tours of duty are more fun to fly than others. Ryan's group has worked with Air New Zealand's unions to develop measures of crew satisfaction that quantify the desirability of lines of work. The rostering problem, then, is to maximize crew satisfaction while covering all tours of duty.

Mathematically, both problems can be described as set-partitioning problems. The generic set-partitioning problem is to find a 0–1 vector x that minimizes the inner product with a "cost" vector c (whose entries are non-negative), subject to a constraint of the form $Ax = u$, where A is a 0–1 matrix and u is the vector of all ones. In the tour-of-duty problem, the rows of A correspond to flight segments and the columns to the various ToDs: The ijth entry is 1 if and only if flight segment i is part of tour j. The 0–1 vector x corresponds to a set of ToDs. The equation $Ax = u$ simply stipulates that each segment is accounted for, and the inner product with the cost vec-tor is simply the total cost of the solution.

The rostering problem is actually a *generalized* set-partitioning problem, because in this case the constraint vector u can have (integer) entries greater than 1, reflecting the fact that most flights have several flight attendants. The columns of the 0–1 matrix A contain the possible lines of work for each employee. The first batch of rows is, in effect, a stretched-out identity matrix; its purpose is to ensure that each employee is assigned one and only one LoW. The rest of the rows correspond to the tours of duty, with a 1 in the ijth entry if and only if the ith ToD is part of the jth LoW.

Each problem winds up with thousands of rows and, potentially, many millions of columns. This would seem to be bad news, because the set-partitioning problem is NP-complete—meaning that researchers are unlikely to come up with an algorithm that's guaranteed to give an exact answer every time in a timely fashion. But the matrices in the ToD and LoW problems have special properties that make them similar to matrices for which linear integer programming works well.

Ryan is currently keen on modifying the aircrew scheduling problem to make its solutions robust, so that an airline's whole schedule doesn't unravel just because one airport is fogged in. Indeed, something like this happened last summer to the British budget airline EasyJet: Ripple effects from minor disruptions early in the day left passengers stranded at airports throughout Europe, as flight after flight was cancelled because the available crews had run out of legal flying hours. The snafu cost EasyJet many millions of dollars.

The problem, as Ryan puts it, is that minimizing costs tends to maximize the potential for (financial) disaster. Given the choice between a schedule with little leeway and one that accommodates the occasional delay, a cost-conscious computer will blithely opt

for the former if it saves even a single shilling. But measuring—or even defining—robustness is tricky; balancing it with cost is trickier; and selling the concept to the bean counters who run things is trickier yet. Nonetheless, Ryan and colleagues have roughed out a workable solution.

Their basic idea is to assess the (non-)robustness of each tour of duty, in effect estimating the likely cost entailed if something goes wrong. They then treat the cost objective as an elastic constraint added to the set-partitioning problem of minimizing the (non-)robustness product rx (subject to $Ax = u$). Elasticity makes it possible to find solutions to the integer programming problem and to see (or not see) large gains in robustness coming from relatively small increases in cost—which is what it will take to convince airlines to adopt the new criteria.

The Price Is Right

Crew scheduling is only one of many airline optimization problems. Fleet assignment—figuring out which type of plane to use on which route—is another, and pricing, or "yield management"—directing the seemingly lunatic way fares bounce around, like unbuckled passengers in heavy turbulence—yet another. Barry Smith of Sabre, Inc., which owns the online booking site Travelocity.com, described the method behind the pricing madness, and how it could be made even more methodical. Indeed, Smith says, when it comes to wringing profit out of pricing, "airlines are getting about half of what they could be getting."

Offhand, the pricing problem doesn't seem overly large. A typical (large) airline has only a few thousand flights each day—far short of the number of different items in a well-stocked WalMart. What complicates things is that these flights, for the most part, fly every day, and that most airlines sell tickets up to a year in advance. Each day's flight on each route is, in effect, a separate product. Moreover, airlines typically offer different kinds of tickets, from the nonrefundable discount ticket to the top-dollar exchangeable ticket. And on top of this is the origin/destination factor: A substantial fraction of airlines' business consists not of passengers simply flying from A to B, but from A to C *through* B (not to mention that that most passengers also want to fly home later on). When the counting's all done, an airline is left with the task of setting prices for several hundred million different products.

As time goes by and tickets sell (or fail to sell), fares go up or down. On a typical day, a typical airline may change a hundred thousand fares, often several times. Setting fares involves a delicate probabilistic analysis that is usually crudely done. The goal of yield management, American Airlines once said in its annual report, is "selling the right seats to the right customers at the right prices." (Or, as Smith puts it, "If there are people on the plane who are smiling too much, you know the price is too low.") As models have improved (partly because of the increase in sheer computer power), the benefits of yield management have grown, from roughly 2% of revenue in the 1960s to 6% in the 1990s. These numbers might seem small, but for a large airline they translate into hundreds of millions of dollars annually.

Smith traced a rough history of yield management. In the 1960s, overbooking was essentially the only factor models could handle; par for the course was three hundred thousand variables, and solutions of the optimization problem increased revenue by about 2%. Taking fare classes into account in the 1970s took the number of variables to about three million, and increased revenue by about 4%. Origin/destination analysis in the 1980s increased revenue by about 5%, at the cost of juggling 30 million variables. In the 1990s, a new approach known as "bid pricing"—in essence making fares bid for flight legs—took the number of variables back to about a million, while still increasing revenue by about 6%.

All these techniques, however, tend to treat ticketing requests as random, uncorrelated events. Upcoming in the current decade, Smith says, is a whole new paradigm dubbed "customer relations management," which does—or should do—a better job of forecasting actual demand. Airlines and consulting firms like Sabre (which spun off from American Airlines—Sabre is an acronym for semi-automated business research environment) are also looking into ways to integrate yield management and fleet assignment. Currently, the two problems are solved separately. Planes are assigned on the basis of anticipated demand (and other factors); seats are sold on the basis of the planes' capacities; and planes are frequently reassigned, sometimes at the last minute. If the two problems can be brought within the purview of a single model, airlines might see a significant jump in revenue. It's possible their customers might benefit as well.

Flight from Reality

I dreamed of 747's
Over geometric farms

—*Joni Mitchell,* "Amelia"

Airfares are one of two airline operations that affect passengers directly. The other is flight delays. Cynthia Barnhart's group at MIT has looked at the latter problem, seeking to quantify trends in delays and, more important, their effects on passengers.

Barnhart draws a crucial distinction between flight delays and what she calls "disrupted passengers." By FAA definition, a flight is delayed if it's 15 minutes late. Airlines keep closer track than that (each additional minute burns a lot of additional fuel, for one thing) but use the FAA mark for reporting their "on-time" performance. According to Barnhart, this is not the best way to measure how well the airlines are doing at getting people to their destinations.

Certainly no one relishes spending an extra quarter hour cooped up in a tin can. But for most people, even an extra half hour is no more than an aggravating inconvenience, provided that's the extent of the delay. The people who are really affected by delays are those who miss connecting flights. These are Barnhart's disrupted passengers.

Actually, Barnhart defines a disrupted passenger as anyone who winds up flying an itinerary other than the one originally booked. This includes people whose flights are simply cancelled. "Disrupted passengers are a big problem in the system," Barnhart says. For someone who misses a connection, a flight delay of a few minutes can turn into multiple hours, often with an overnight stay in a hotel. During the busy summer months in 2000, one major U.S. airline disrupted roughly 4% of all its passengers. That's a

significant number of angry people.

Delays and disruption are, of course, related. But the correlation is weak, because many delays don't lead to disruption. For one thing, not all passengers on any given flight will be heading for another gate. In addition, delays often affect the entire system. If your connecting departure is delayed, then it doesn't matter if your incoming flight arrives late.

In fact, that's part of the strategy Barnhart's group has taken to address the disrupted-passenger problem. If the goal is to minimize the total number of disrupted passengers, part of the solution is to postpone certain departures—and even strategically cancel some flights (freeing up planes for other, more "important" flights—that is, intentionally disrupting some passengers in order to undisrupt others). The trick is to do this in a manner that reduces overall passenger delay.

It can be done—at least in simulations. In one study, the MIT optimizers took an example with 67,500 passengers, of whom 21,500 had connecting flights. In the real-world base case, there were 1896 disrupted passengers, of whom 900 wound up overnighting. Their total delay came to 762,152 minutes—an average of just under seven hours. The total for everybody else was 986,609 minutes—i.e., an average delay of about 15 minutes per person. The overall average passenger delay was 25.9 minutes.

Optimization (embedded in an airline operations simulation) decreased the overall average delay to 22.6 minutes. (The optimal solution actually increased the average for delayed *plane* minutes. That's why Barnhart considers on-time performance an inadequate metric.) For connecting passengers, the average delay dropped dramatically, from 34.1 to 18.5 minutes, while for local passengers it increased, but only slightly, from 22.4 to 24.4 minutes. The number of disrupted passengers dropped to 1131, with only 491 overnights.

Barnhart's group is also looking at optimal routing as a way to minimize the number of disrupted passengers. The basic idea is to reduce the propagation of delays by redesigning the routes—which amounts to looking at the aircraft rotation problem from a passenger point of view. The models they are developing show a lot of promise for making flight delays less of a hassle, Barnhart says. And "there's still a lot more potential."

Ground Delays

The station master's shuffling cards
Boxcars are banging in the yards
—Joni Mitchell, "Just Like This Train"

Airlines aren't the only aspiring optimizers in the transportation industry. Trucks, buses, and trains are also fodder for algorithmic improvements. Bruce Patty of Exostrategy Partners in Sausalito, California, for example, outlined the use of optimization to improve reliability in freight train service.

"Railroads are way more complex than airlines," Patty points out. In particular, airlines are not constrained by tracks. Moving a freight car to an adjacent parallel track—a distance that might be as little as ten feet—can be a time-consuming task. Railroad switchyards, with their acres of parallel tracks, pose a combinatorial challenge: Given an incoming, N-car train whose cars have destinations d_1, d_2, ..., d_N, how do you "best" recombine it with other trains in the yard? Northern Pacific's acronym, NP, seems appropriate.

As usual, "best" is a slippery word. The switchyard probably wants to keep its own labor costs down. The freight haulers are similarly cost conscious, but they also have paying customers to keep happy. It's good to make the trains run on time.

Reliability is especially important in the new paradigm of just-in-time manufacturing. For traditional freight items, such as coal, grain, and bulk chemicals, being a few days late is no big deal; on reaching their destinations, those materials tend to get dumped into never-depleted stockpiles. But for finished or semi-finished products like auto parts, running late can idle an entire plant. Railroads don't need to worry about minutes the way the airlines do, but they do have to watch the hour hand.

There is a lot of room for improvement. The average freight car, Patty points out, spends only about 14% of its time on a moving train. And the dense tangle of tracks, especially in the eastern half of the U.S., is an opportunity for "dynamic" scheduling: picking routes in response to changing traffic patterns, rather than according to fixed rules. The railroads can also do a better job of information gathering; right now, Patty says, they typically dispatch cars for loading without bothering to ask their customers where they want the stuff sent.

Scheduling buses for urban mass transit systems is another challenge, as Guy Desaulniers of l'Ecole Polytechnique in Montréal explained. Whereas planes and trains run on well-defined graphs (airports and train tracks being the vertices and edges of their respective graphs), bus lines can be laid out in myriad changeable ways, with a variable number of buses. In New York City, for example, nearly 5000 buses are assigned to 300 different routes, with each bus making 10 to 20 complete trips per day. (Paris is not far behind, with 4000 buses on 250 routes. Desaulniers' Montréal has 1500 buses on 200 lines.)

The scheduler's job is to lay out an efficient plan that satisfies the demands of a commuting population. Inefficiencies include "deadhead" miles racked up by empty buses heading to and from their terminals (New York has 18 depots, Paris 23) or from one route to another, and shift changes, when drivers become paid passengers or pedestrians. It may make sense, for example, to switch drivers in the middle of a route, so that one can have lunch and then take over another route.

Once routes have been determined (most bus routes are stable from year to year—passengers would protest frequent changes), the daily assignment of buses to routes and drivers to buses is similar to the airlines' fleet-assignment and rostering problems. Tackled individually, the problem of assigning buses to routes has been solved to near-optimality for large to very large systems, Desaulniers reports, and there are good solutions for medium to large cases of the problem of assigning drivers to buses. The simultaneous problem has near-optimal solutions for small to medium cases, but so far the algorithms apply only to buses coming from a single depot. Work is under way on the combined problem with multiple depots, as well as on the potential benefits of tinkering with timetables.

Delivering the Goods

I can't decide
I don't know
Which way to go?

—Joni Mitchell, "Fiction"

When it comes to online grocery shopping, timing is everything. Webvan, the billion-dollar venture that went belly up in 2001, is a case in point. Martin Savelsbergh of Georgia Tech described how Webvan's scheduling and pricing of home deliveries helped bring about its demise—and how good algorithms could have made a difference.

Offhand, online grocery shopping seems simple: The customer places an order, and the company puts the order in a bag, the bag in a truck, and the truck on the road. But it gets complicated: To make money, the company needs lots of customers placing lots of orders. Even that would be a simple traveling salesman–type problem, but for one extra wrinkle: Online grocers need to be able to guarantee delivery in specific time windows.

Consumers may be accustomed to waiting all day for the cable guy or the appliance repairman, but that's partly because they have no alternative. Grocery shopping is different. People can easily jump in the car or have their spouses stop at the store on the way home from work. The people who are most attracted to online shopping are precisely those with the least amount of time to waste. Mail-order packages can sit on the stoop all day, but groceries for the most part have to be received in person—the milk has to get into the fridge. Online grocers need to figure out how many customers they can satisfy—and how they're going to pay for it when they're peddling products that traditionally have low profit margins.

Hoping to establish a loyal customer base, Web-van initially offered free delivery. That didn't work very well. They eventually started charging for orders under $50 (and later raised the threshold to $100), which didn't help. Part of the problem, Savelsbergh says, was the simplistic rules Webvan (and most others) used for scheduling deliveries: They decided in advance how many orders they could deliver in each half-hour time window, and then let customers pick windows on a first-come, first-served basis. This certainly guaranteed on-time delivery, but it also guaranteed suboptimal solutions cost-wise.

Savelsbergh and Ann Campbell of the University of Iowa have formally defined what they call the "home delivery problem" and proposed algorithms for solving it. In the bare-bones problem, customers arrive at random and place orders, including specified time windows. The program's job is to decide whether or not to accept each order as it comes in. (In practice, the program would work out which windows are profitable and present them as options to the customer, perhaps with pricing incentives.)

The problem is thoroughly NP-complete. Even determining feasibility for a new order (i.e., can it be fit in with all the previously accepted orders?) is hard. Savelsbergh and Campbell use an insertion heuristic to get around the difficulty: The program maintains a manageable list of possible routes for the orders already accepted, and looks for places to insert the new order between two others. If it accepts the order, it updates the list.

Savelsbergh and Campbell's algorithms also anticipate likely orders that haven't arrived yet. (If you have a regular customer who always waits till Friday to place a huge order for Saturday delivery, you probably don't want to fill up all those Saturday slots with rinkydink orders that arrive on Wednesday and Thursday.) Ideally, an algorithm would consider all possible combinations of subsequent orders, together with their probabilities, and decide whether to accept the current offer so as to maximize expected profit, but a calculation like that is hopelessly intractable. An alternative, practical approach is to multiply each potential order by its probability of occurring, sort these "discounted" orders by size, and run them through the insertion heuristic with and without the current order. (Even this is tricky, because the insertion costs ultimately depend on which orders are actually placed. Various methods are available for estimating them, without producing a combinatorial explosion.)

Algorithm	Stops	Revenue	Cost	Profit	%Increase
BEST	21.2	848.00	431.92	416.08	86.5
SLOT	19.4	792.00	586.86	223.14	N/A
DYN	23.1	924.00	588.64	335.36	50.3
DSR	18.1	724.00	380.77	343.23	53.8
PATH	16.2	648.00	323.20	324.80	45.6
RAD	19.3	772.00	395.65	376.35	68.7
DIFF	14.0	560.00	269.04	290.96	30.4

Table 1. *Home Shopping. A comparison of algorithms in an average of 10 simulations with 100 customers on a 60 x 60 grid (see text for details). SLOT is a simplistic rule that limits (to 2) the number of stops permitted in each time window. DYN is a dynamic routing algorithm that accepts incoming requests whenever feasible. The other algorithms anticipate future requests, estimating their insertion costs in a variety of ways. The column labeled "% increase" compares profit for each algorithm with that for SLOT.*

To test their algorithms, Savelsbergh and Campbell have run simulations in settings where best-possible (in hindsight) solutions can be computed. They have found that their heuristics regain much of the profit lost with the traditional, simplistic approach. In one set of experiments with 100 customers on a 30 $\times$ 30 grid (with one-minute travel time between adjacent grid points), each with a 24% chance of placing a $40 order, the simplistic rule captured only about $510 of the potential (highest possible) $680 in profit. The more sophisticated algorithms netted between $660 and $670. On a sparser, 60 $\times$ 60 grid, the new algorithms don't get nearly that close to the optimum (see Table 1), but their effect on profits is even more dramatic: Good computation can increase profits by anywhere from 30% to nearly 70%. That's not bad for a few seconds' work.

Barry A. Cipra is a mathematician and writer based in Northfield, Minnesota.