# Branch-and-Cut Approach to a Variant of the Traveling Salesman Problem

Manfred Padberg*
*New York University, New York, New York*
and
Giovanni Rinaldi†
*Istituto di Analisi dei Sistemi ed Informatica del CNR, Rome, Italy*

**A modified traveling salesman problem is formulated as a zero-one linear program. The associated (relaxed) linear programming problem can be solved in polynominal time despite an exponentiation of the proposed constraint system in terms of the underlying parameter $n$ of the number of cities considered, when a nonlinear constraint of the problem is either ignored or approximated by linearization. A software system, AIAA/SOLVER, has been implemented to solve the problem to optimality under an apparently weak assumption about its stochastic cost structure using branch-and-cut. In this paper we give a brief outline of the problem formulation and solution.**

## Introduction

THE AIAA problem is a variant of the well-known traveling salesman problem. The description of this problem is given in Ref. 1. In this paper we formulate the AIAA problem as a zero-one linear program. In the following sections we discuss the formulation and describe a software system that solves the problem to optimality. Details can be found in Ref. 2.

## Formulation of the AIAA Problem as a 0-1 Linear Program

The formulation of the problem involves essentially three different parts: the modeling of flow conservation and connectivity, the modeling of the budget constraint, and the modeling of the local conditions. To model these we need the following decision variables. Let

$$x_{ij}^k = \begin{cases} 1 & \text{if the arc from } i \text{ to } j \text{ is chosen for the } k\text{th time} \\ 0 & \text{if not,} \end{cases}$$

where $k = 1,2,...,m$ and $m$ is some common upper bound on the number of multiple trips, e.g., $m = n - 1$, i.e., we permit multiple trips between any two cities. Such "multiple trip" variables can be carried "implicitly" in a computer program and are generated as needed by a column-generation scheme. To model the decision to visit a city or not, we let

$$z_i = \begin{cases} 1 & \text{if city } i \text{ is visited,} \\ 0 & \text{if not} \end{cases}$$

Furthermore, letting $c_{ij}$ be the coach fare from city $i$ to city $j$, $f_j$ be the fixed cost of visiting city $j$, and $h$ be the home city, denote

$$c_{ij}^+ = \begin{cases} f_i + c_{ij} & \text{for all } j \neq i,h \text{ and all } i \\ c_{ij} & \text{for all } i \neq h \text{ and } j = h \end{cases}$$

as the deterministic or minimum cost incurred by traveling coach from city $i$ to city $j$. With these notations and conventions, we can now formulate the first two parts of the problem completely.

To express the local conditions exactly, we introduce four logical variables, $y_1$, $y_2$, $y_3$, and $y_4$. We let $y_1 = 1$ if we decide to collect the valuation in city $p$ (LAX), $y_1 = 0$ otherwise. Variables $y_2$, $y_3$, and $y_4$ are all connected to the existence of certain directed paths. Let $y_2 = 1$ if we decide to make a directed path from $p$ (LAX) to $q$ (BOS) that avoids $h$ (home, e.g., DTT), $y_2 = 0$ otherwise. Likewise, let $y_3 = 1$ if we decide to make a directed path from $q$ to $h$ that avoids $p$, $y_3 = 0$ otherwise, and $y_4 = 1$ if we decide to make a directed path from $h$ to $p$ that avoids $q$, $y_4 = 0$ otherwise. In Fig. 1 we display all minimal configurations possible if we decide to collect the valuation in city $p$. (The "arcs" in Fig. 1 indicate the existence of *directed paths*, not necessarily direct links.) Every possible feasible tour must contain at least one of these configurations as a partial subgraph if we decide to collect the valuation in city $p$.

The AIAA design challenge problem can now be formulated as a linear program in zero-one variables as follows, where $v_i$ is the valuation received at city $i$:

$$\max \sum_{i \neq p,h} v_i z_i + v_p y_1 - \epsilon \sum_{i \neq j} (c_{ij}^+ + \pi\gamma c_{ij}) \sum_k x_{ij}^k + v_h$$

subject to

$$\sum_{j \neq i} \sum_k x_{ji}^k - \sum_{j \neq i} \sum_k x_{ij}^k = 0 \quad \text{for all } i \tag{1}$$

$$-\sum_{i \in U} \sum_{j \in V - U} x_{ij}^1 + z_k \leq 0 \quad \text{for all } \begin{cases} k \in U, \\ \phi \neq U \subseteq V - h \end{cases} \tag{2}$$

$$x_{ji}^1 - z_i \leq 0 \quad \text{for all } i \neq j \tag{3}$$

$$x_{ij}^{k+1} - x_{ij}^k \leq 0 \text{ for all } i \neq j \text{ and all } k \tag{4}$$

$$\sum_{i \neq j} \sum_k (c_{ij}^+ + \pi\gamma c_{ij}) x_{ij}^k + \sigma(x) F^{-1}(1 - \alpha) \leq B \tag{5}$$

$$-\sum_{i \in U} \sum_{j \in V - U} x_{ij}^1 + y_1 \leq 0 \quad \text{for all } \begin{cases} U \subseteq V, \\ U \ni p, U \not\ni q \end{cases} \tag{6}$$

$$-\sum_{i \in U} \sum_{j \in V - h - U} x_{ij}^1 + y_2 \leq 0 \quad \text{for all } \begin{cases} U \subseteq V - h, \\ U \ni p, U \not\ni q \end{cases} \tag{7}$$

$$-\sum_{i \in U} \sum_{j \in V - h - U} \sum_k x_{ij}^k + 2y_1 - y_3 - y_4 \leq 0 \quad \text{for all } \begin{cases} U \subseteq V - h, \\ U \ni q, U \not\ni p \end{cases} \tag{8}$$

Received July 23, 1987; presented as Paper 87-2331 at the AIAA 1987 Guidance, Navigation, and Control Conference, Monterey, CA, Aug. 17-19, 1987; revision received Jan. 15, 1988. 

*Professor.

†Visiting Associate Professor.

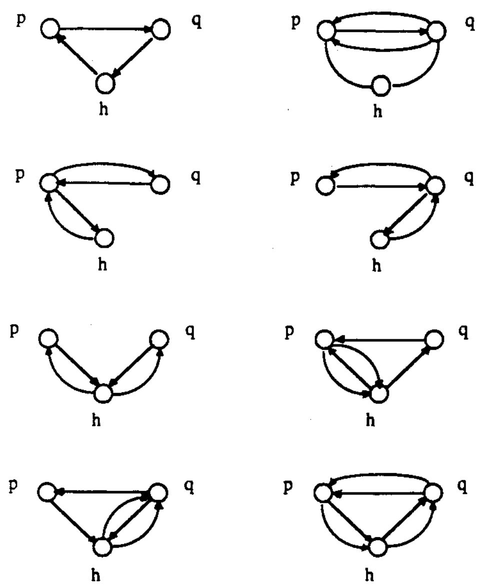


**Fig. 1 Minimal path configurations for the local constraint if $y_1 = 1$.**

$$-\sum_{i\in U}\sum_{j\in V-p-U} x^1_{ij} + y_3 \le 0 \quad \text{for all } \begin{cases} U \subseteq V-p, \\ U \ni q, U \not\ni h \end{cases} \tag{9}$$

$$-\sum_{i\in U}\sum_{j\in V-p-U}\sum_{k} x^k_{ij} + 2y_1 - y_2 - y_4 \le 0 \quad \text{for all } \begin{cases} U \subseteq V-p, \\ U \ni h, U \not\ni q \end{cases} \tag{10}$$

$$-\sum_{i\in U}\sum_{j\in V-q-U} x^1_{ij} + y_4 \le 0 \quad \text{for all } \begin{cases} U \subseteq V-q, \\ U \ni h, U \not\ni p \end{cases} \tag{11}$$

$$-\sum_{i\in U}\sum_{j\in V-q-U}\sum_{k} x^k_{ij} + 2y_1 - y_2 - y_3 \le 0 \quad \text{for all } \begin{cases} U \subseteq V-q, \\ U \ni p, U \not\ni h \end{cases} \tag{12}$$

$$-\sum_{j\neq h}\sum_{k} x^k_{jh} + z_h + y_1 - y_2 \le 0 \tag{13}$$

$$-\sum_{j\neq p}\sum_{k} x^k_{jp} + z_p + y_1 - y_3 \le 0 \tag{14}$$

$$-\sum_{j\neq q}\sum_{k} x^k_{jq} + z_q + y_1 - y_4 \le 0 \tag{15}$$

$$y_2 \le y_1, \quad y_3 \le y_1, \quad y_4 \le y_1, \quad y_1 \le z_p, \quad y_1 \le z_q$$

$$z_i \le z_h \qquad \text{for all } i \neq h \tag{16}$$

$$x^k_{ij} \in \{0,1\}, \qquad z_i \in \{0,1\}, \qquad y_r \in \{0,1\} \quad \text{for all } i,j,k,r \tag{17}$$

In this formulation, $\pi$ is the probability of having to fly first-class, $\gamma$ is the surcharge factor for first class expressed as a decimal fraction, and $\varepsilon$ is a small positive scalar to accommodate the secondary objective of achieving minimal expected cost, the primary objective being to collect a maximum total value. The city $p$ is LAX, the city $q$ is BOS, and $h$ denotes the home city.

The budget constraint (5) is obtained by formulating it as a "chance constraint" and then converting it to its "deterministic equivalent."[3] The term $\sigma(x)$ is the standard deviation

$$\sigma(x) = \gamma\sqrt{\pi(1-\pi)}\sqrt{\sum_{i\neq j}\sum_{k}(c_{ij}x^k_{ij})^2}$$

of the excess fare for first class, and the term $F^{-1}(1-\alpha)$ is the fractile of a *normalized* random variable. For large enough $n$, this term can be approximated with sufficient accuracy by a normal (or Gaussian) random variable having zero mean and a standard deviation of one. For a given $\alpha$, the fractile $F^{-1}(1-\alpha)$ for such a distribution becomes a number that can be read from a table.

The remainder of the constraints express the local condition. It is assumed here that $p$, $q$, and $h$ are three distinct cities. A priori, there may be any number of such local conditions, and the formulation generalizes canonically to several such conditions by introducing an additional four logical variables (and the corresponding constraint set) for every one of them. The entire formulation of the problem is described in greater detail in Ref. 2.

## Discussion of the Problem Formulation

The number of variables required by the model is modest: we need $n(n-1)$ variables to represent the arcs and $n$ variables for the cities. Furthermore, for each local condition, an additional four variables suffice. If no multiple trips are considered, then $n(n-1) + n + 4$ zero-one variables model the problem with one local condition. For every additional local constraint, exactly four additional logical variables are required. In the case that the triangle inequality holds for the distance table, i.e., if

$$c_{ij} + c_{jk} \ge c_{ik} \text{ for all } i < j < k$$

multiple trips need not be considered. To permit the use of an arc from any $i$ to any $j$ more than once—as might be desirable if the triangle inequality does not hold—we introduce any number of multiple arc variables as required "on the fly." These variables can be carried implicitly and are introduced or forgotten as required. The economy in the number of variables is achieved by the observation that the local conditions can be stated in terms of the existence of certain directed paths. These, in turn, we know are the "blockers" of certain cuts, hence the formulation.

On first sight, the number of constraints might seem too large to be solvable. However, from related work on the symmetric traveling salesman problem,[4-6] we know that it is *not the number* of constraints that matters, but rather whether or not we can efficiently *identify* a violated constraint. Indeed, despite the exponentiation of the constraint set in terms of the number of cities $n$, the associated linear programming problem—obtained by ignoring the integrality requirement—can be solved in polynomial time, since all of its constraints can be "identified" efficiently. Of course, this does not mean that all basic feasible solutions to this linear program are zero-one valued. Rather, we will need additional inequalities to ensure integrality of the solution. The polynomial solvability of the associated linear program – when the nonlinear budget constraint is either ignored or linearized – follows, e.g., by a theorem of Ref. 7, using the ellipsoid method as the linear programming algorithm; see Ref. 2 for details.

The only "tricky" part of the formulation rests with the probabilistic aspect of the budget constraint. Probabilistic constraints—especially when few discrete random variables are involved—are notoriously difficult to model exactly. Whether or not our (standard) approximation of the term $F^{-1}(1-\alpha)$ in the deterministic equivalent works remains to be seen in large-scale practice. Of course, if the budget constraint has to hold with certainty or if it suffices to have the expected travel cost meet the budget, then the standard deviation disappears and the proposed formulation is exact. In any case, the budget constraint is asymptotically equivalent to a constraint that defines a convex region in the space considered here because

$$\sigma[\lambda x^1 + (1-\lambda)x^2] \le \lambda\sigma(x^1) + (1-\lambda)\sigma(x^2)$$

for all $0 \le \lambda \le 1$. While the resulting constraint is nonlinear as a result of its convexity, we can use standard methods to deal with it in a linear programming context. This means we can replace it by the totality of its tangential hyperplanes, i.e., by "enveloping" the convex region and invoking at every step of the algorithm only a finite subset of the tangential hyperplanes. As we are interested in finding a zero-one extreme point, this proceeding is theoretically and computationally feasible. See, e.g., Ref. 8 for a related recent study in this vein.

To the best of our knowledge, the *polyhedral structure* of the problem formulated here has not been studied, while the polyhedral structure of the related asymmetric standard traveling salesman problem has been studied in some detail.[4] For the resolution of large-scale problems of this kind, it would be necessary to study the polyhedral structure in depth. Once enough polyhedral information is accumulated and used in the process of calculation, we are confident that problems involving 1000 cities and more can be solved using the proposed formulation. Our estimate is based on our numerical experience with symmetric traveling salesman problems,[6] where we report the optimization of symmetric traveling salesman problems with up to 2392 cities, i.e., 2,859,636 zero-one variables, on the Cyber 205 Supercomputer of the National Bureau of Standards in Gaithersburg, MD.

### Implementation of the AIAA/SOLVER System

To implement a software system for the AIAA problem, we followed the same basic methodology that we have developed for the symmetric traveling salesman problem[6] and that we are currently developing for general zero-one linear programs.[9] Besides a "top" procedure, the software system consists essentially of four components: 1) a heuristic procedure, 2) a linear program solver, 3) a constraint or cut generator, and 4) a branch and cut procedure.

The four components of the AIAA/SOLVER system are described in detail in Ref. 2. Here we will outline the changes that we made to the formulation and state the assumption about the stochastic cost structure under which we find optimal solutions to the problem. Besides the assumption about the stochastic cost structure, we made an additional assumption in the current version of our software system AIAA/SOLVER. We have *not* yet implemented the possibility of using the arcs between two cities more than once. In terms of our formulation of the problem, this means we assume that $m = 1$. It is conceptually not difficult to remove this assumption; however, time constraints have prohibited us from implementing these ideas because of the necessary additional work and code implementation.

The AIAA problem formulation poses the problem as one that has two rather than a single objective function: maximum total value collected at minimum actual (or expected) total cost. As we are permitting a possible *budget excess*, i.e., an excess of the *actual* total cost over the fixed budget with some positive probability, it makes more sense to us to look for the smallest *actual* cost rather than the minimum *expected* cost. So we chose to look for a tour (a feasible solution to the problem as previously formulated) that permits one to collect the maximum total value at minimum actual cost. However, if one is inclined toward the minimum expected cost tour, in our framework, all that is required, essentially, is to set up the coefficients of a single array differently from the way we are setting it up in our current program. Of course, these minor "conceptual" changes require additional work and code development.

Consider now a diagram where one plots the total collected value $v$ vs the actual cost $c$ of tours. Clearly—just like, e.g., in portfolio theory,[10]—one is interested in "efficient" $(v,c)$ combinations, i.e., combinations that for a given value $v$ give the minimum attainable $c$ and that for a given cost $c$ give the maximum attainable $v$.

Ignoring the probability constraint for a moment, one can thus approach the problem by looking, given a value $v_0$, for a minimum actual cost tour, e.g., a tour that yields a cost of $c_0$. If $c_0 > B$, then any value greater than or equal to $v$ is under "reasonable" assumptions not attainable; thus, $v_0$ is too big and must be decreased. If, on the other hand, $c_0 \leq B$, then one has a possibly attainable total value $v_0$, and one must check the probability constraint. If it is satisfied, a total value bigger than $v_0$ is possibly attainable, and $v_0$ can be increased. Otherwise, one concludes as in the case of $c_0 > B$.

To make the preceding exact, we need the following assumption about the underlying stochastic cost structure: Let $x^1$ with *actual cost* $C(x^1)$ and $x^2$ with $C(x^2)$ be any two tours. We assume

$$C(x^1) \leq C(x^2) \Rightarrow Pr[\text{budget excess}(x^1)] \leq Pr[\text{budget excess}(x^2)] \tag{18}$$

If the actual or minimum (deterministic) cost of a tour $x^1$ is smaller than the actual cost of a tour $x^2$, then the probability of exceeding the budget under $x^1$ is also smaller than the probability of exceeding the budget under $x^2$. Essentially, this amounts to barring ruinous competition among airlines on certain flight legs, i.e., excluding maxsavers, supersavers, "red-eye" flights, etc., on, say, trips from Chicago to Los Angeles. In other words, we exclude airfares that distort a "natural" cost structure and that a businessman who considers taking a first-class seat, if necessary, in order to get on a particular flight from $A$ to $B$ would normally not consider. Under these circumstances, we think that our assumption (18) is justified. If this assumption is satisfied, then the set of efficient $(v,c)$ combinations becomes a convex curve, i.e., an "efficient frontier" in the sense of portfolio theory, and we can "trace" the efficient frontier by the following procedure.

Denote $(P_v)$ with the following problem:

$$\min z(v) = \sum_{i \neq j} \sum_{k} c_{ij}^{+} x_{ij}^{k}$$

subject to (1),...,(4)

$$\sum_{i \neq p,h} v_i z_i + v_p y_1 \geq v - v_h \tag{5'}$$

and (6),...,(17)

Procedure AIAA/DRIVER:

Step 1: Find an initial *feasible* tour $x$ yielding a total value $v$. In particular, the local condition and the chance constraint are satisfied for $x$, and $x$ is the "incumbent."

Step 2: Solve $(P_v)$ and denote $x^v$ the corresponding tour with minimum actual cost $C(x^v)$ and total value $V(x^v) \geq v$.

Step 3: Calculate the probability of exceeding the budget if $x^v$ is used. If the chance constraint is violated, stop: the current incumbent is optimal.

Else: Call $x^v$ the incumbent, replace $v$ by $\max\{V(x^v),v\} + 1$ in $(P_v)$, and go to step 2.

We assume positive *integer* data for the values $v_i$, which explains the increase of $\max\{V(x^v),v\}$ by $+1$ in step 3. We note that the tour consisting of "staying home" is always feasible, and thus, step 1 is always well defined. If Eq. (18) is satisfied by the stochastic cost structure, then the described procedure finds an optimal solution to our problem because the optimal objective function value $z(v)$ of the parametric problem $(P_v)$ is *monotonically increasing* in the "cut-off" value $v$. However, rather than incrementing by $+1$ in step 3, we pick a next "possible" value for the total collected value from a table that is calculated once using *dynamic programming*[11] as the set of distinct values that the function

$$\sum_{i \neq p,h} v_i z_i + v_p y_1$$

can assume for $z_i \in \{0,1\}$ and $y_1 = 0$ or 1. This table is also used in the heuristic procedure and is calculated in pseudopolynomial time using a simple recursion formula.

For future reference, we would like to point out that Eq. (18) could be replaced by the following seemingly weaker assumption:

$$EC(x^1) \leq EC(x^2) \Rightarrow Pr[\text{budget excess}(x^1)] \leq Pr[\text{budget excess}(x^2)] \tag{19}$$

where $E$ denotes the expected value operator. As in the case of the point that we made concerning actual cost vs expected cost minimization, such a change is conceptually simple and possible within the given framework. However, we have implemented in AIAA/SOLVER the former assumption (18).

The first time that step 2 is executed, we solve a linear program (LP) using a proper subset of the XMP software package for linear programming[12] with a few minor changes. It has $2n + 7$ equations and inequalities and $n^2 + 4$ structural variables. Since all of our variables are bounded and the upper bound for slack/surplus variables are readily computed, we can use the dual simplex method throughout; i.e., only the XMP routine XDUAL and those called by it are used. In addition, the solution found by the heuristic is used to supply the linear program with an advanced starting basis. This is done in order to reduce the computation time for the initial linear program. This program is solved to optimality and the optimal solution "fed" into the cut generator.

The cut generator is a collection of subroutines that 1) set up the necessary data structures, 2) carry out the minimum weighted cut and maximum flow calculations necessary to identify violated constraints, and 3) enumerate and check those constraints of which there are only polynomially many. At the end of the call to these subroutines, we have one of two outcomes: Either we have not identified any violated constraint, or one or several violated constraints were found and added to the constraint set of the linear program. In the latter case, we have a new linear program with a larger number of constraints that we reoptimize using the dual simplex method and *repeat* by feeding the new LP solution into the cut generator. (In the iterative step we "purge" all cut constraints whose associated slacks have become positive, provided that the objective function increased strictly.) In the former case, we may have one of two possibilities: either the LP solution is zero-one valued in the structural variables, in which case we have solved the current problem ($P_v$) and a new "macroiteration in the procedure AIAA/DRIVER is initiated, or the LP solution is fractional, in which case we have no choice but to "branch" on some fractional-valued structural variable.

The branch-and-cut (B&C) procedure consists of several subroutines that 1) maintain and update the data structures necessary to represent a search tree, 2) choose a "next" variable to branch on, and 3) fathom branches of the search tree. The basic methodology for this technique has been described by us elsewhere.[6] As we are working within the overall framework of the described AIAA/DRIVER procedure, we had to implement branch-and-cut somewhat differently from the scheme that we developed for the symmetric traveling salesman problem. In short, for every new value $v$ in step 2 of the AIAA/DRIVER, a new search tree has to be started but, of course, such a tree may consist of a single node only. For the necessary details of this procedure, we refer the reader to Ref. 2.

## Numerical Experiments

To illustrate the software system AIAA/SOLVER and its output, we include three sample runs with different home cities and various levels of $\alpha$ using the sample data of the AIAA problem. In all three runs, the local condition concerns Los Angeles (city $p$) and Boston (city $q$). The problems are executed on an IBM Personal Computer AT. The software system is designed to display the "current best" solution as soon as it is found by the program. The sequence of the cities to be visited is displayed starting and ending with the home city. Then the total collected value, the deterministic and the expected total cost of the tour, and the probability of exceeding the budget are displayed. While the program is still working, either at finding a better tour or at proving optimality of the currently displayed tour, it gives the message "Still working...." The optimal tour is displayed in the same fashion, with some additional summary statistics. These display the initial valuation found by the heuristic and the heuristic error, if any. The number of LP calls states the total number of all linear programs that had to be optimized and/or reoptimized until termination. The total number of B&C nodes is the grand total of all nodes for all search trees that may have been developed by the program, while the number of macroiterations counts the number of times the procedure AIAA/DRIVER had to reset the cutoff value until termination. Finally, we print the total CPU time that was required to execute the entire system AIAA/SOLVER.

In the first run, the home city is Detroit (DTT) and $\alpha = 0.05$. In the second run, the home city is Atlanta (ATL) and $\alpha = 0.10$. The last run has Seattle (SEA) as home town and $\alpha = 0.25$.

### Run 1

The optimal tour is given by the following sequence of cities:

DTT-CHI-LAX-PHX-DEN-DFW-MSY-ATL-BOS-DTT

The total collected value is 88, the minimum total cost of the tour is \$2500, the expected total cost of the tour is \$2704, and the probability of exceeding the budget is 0.004791.

*Statistics*: The initial tour value is 80, the heuristic error is 10.000%, the number of LP calls is 7, the number of B&C nodes is 0, and the number of macroiterations is 3. The total time for this run is 119 s.

### Run 2

The optimal tour is given by the following sequence of cities:

ATL-MSY-DFW-LAX-MSP-CHI-DTT-BOS-ATL

The total collected value is 82, the minimum total cost of the tour is \$2500, the expected total cost of the tour is \$2716, and the probability of exceeding the budget is 0.016650.

*Statistics*: The initial tour value is 80, the heuristic error is 5.500%, the number of LP calls is 20, the number of B&C nodes is 9, and the number of macroiterations is 3. The total time for this run is 294 s.

### Run 3

The optimal tour is given by the following sequence of cities:

SEA-LAX-PHX-DFW-BOS-DTT-CHI-MSP-SEA

The total collected value is 82, the minimum total cost of the tour is \$2660, the expected total cost of the tour is \$2895, and the probability of exceeding the budget is 0.221146.

*Statistics*: The initial tour value is 74, the heuristic error is 10.811%, the number of LP calls is 25, the number of B&C nodes is 12, and the number of macroiterations is 5. The total time for this run is 477 s.

## Acknowledgments

Partial financial support was provided by NSF grant DMS8508955, ONR Grant R&T4116663, and a New York University Research Challenge Fund grant.

## References

[1]Deutsch, O., "The A.I. Design Challenge—Background, Analysis, and Relative Performance of Algorithms," *Proceedings of the AIAA Guidance, Navigation and Control Conference*, AIAA, New York, 1987, pp. 465–476.

[2]Padberg, M. and Rinaldi, G., "A Branch-and-Cut Approach to a Traveling Salesman Problem with Side Constraints," TR 204, IASI-CNR, 1987.

[3]Charnes, A. and Cooper, W., "Chance-Constrained Programming," *Management Science*, Vol. 6, 1959, pp. 73–79.

[4]Grötschel, M. and Padberg, M., "Polyhedral Theory," *The Traveling Salesman Problem*, edited by E. Lawler et al., Wiley, Chichester, England, UK, 1985, pp. 251–305.

[5]Padberg, M. and Grötschel, M., "Polyhedral Computations," *The Traveling Salesman Problem*, edited by E. Lawler et al., Wiley, Chichester, England, UK, 1985, pp. 307–360.

[6]Padberg, M. and Rinaldi, G., "Optimization of a 532-City Symmetric Traveling Salesman Problem by Branch-and-Cut," *Operations Research Letters*, Vol. 6, 1987, pp. 1–7.

[7]Padberg, M. and Rao, M. R., "The Russian Method for Linear Inequalities III: Bounded Integer Programming," New York Univ., TR 81-39, 1981.

[8]Duran, M. and Grossmann, I., "An Outer-Approximation Algorithm for a Class of Mixed-Integer Nonlinear Programs," *Mathematical Programming*, Vol. 36, 1986, pp. 307–339.

[9]Hoffman, K., Padberg, M., and Rinaldi, G., "A Branch-and-Cut Optimizer for Sparse Zero-One Linear Programs," New York Univ. preprint, (in preparation).

[10]Markowitz, H., *Portfolio Selection: Efficient Diversification of Investments*, Wiley, New York, 1959.

[11]Bellman, R., *Dynamic Programming*, Princeton Univ. Press, Princeton, NJ, 1957.

[12]Marsten, R., "The Design of the XMP Linear Programming Library," *ACM Transaction of Mathematical Software*, Vol. 7, 1981, pp. 481–497.