Examination and testing of the computer software TSMP-1, ver.1.0

International Intellectual Group, Inc. (I.I.G., Inc.)

1999 (updated by 2016)

Background

History

The traveling salesman problem, or TSP for short, is easy to state: given a finite number of "cities" along with the cost of travel between each pair of them, find the cheapest way of visiting all the cities and returning to your starting point. (The travel costs are symmetric in the sense that traveling from city X to city Y costs just as much as traveling from Y to X; the "way of visiting all the cities" is simply the order in which the cities are visited.) To put it differently, the data consist of integer weights assigned to the edges of a finite complete graph; the objective is to find a hamiltonian cycle (that is, a cycle passing through all the vertices) of the minimum total weight. In this context, hamiltonian cycles are commonly called *tours.*

The origins of the TSP are obscure. In the 1920's, the mathematician and economist Karl Menger publicized it among his colleagues in Vienna. In the 1930's, the problem reappeared in the mathematical circles of Princeton. In the 1940's, it was studied by statisticians (Mahalanobis (1940), Jessen (1942), Gosh (1948), Marks (1948)) in connection with an agricultural application and the mathematician Merill Flood popularized it among his colleagues at the RAND Corporation. Eventually, the TSP gained notoriety as the prototype of a hard problem in combinatorial optimization: examining the tours one by one is out of the question because of their large number, and no other idea was on the horizon for a long time.

A breakthrough came when George Dantzig, Ray Fulkerson, and Selmer Johnson (1954) published a description of a method for solving the TSP and illustrated the power of this method by solving an instance with 49 cities, an impressive size at that time. They created this instance by picking one city from each of the 48 states in the U.S.A. (Alaska and Hawaii became states only in 1959) and adding Washington, D.C.; the costs of travel between these cities were defined by road distances. Rather than solving this problem, they solved the 42-city problem obtained by removing Baltimore, Wilmington, Philadelphia, Newark, New York, Hartford, and Providence. As it turned out, an optimal tour through the 42 cities used the edge joining Washington, D.C. to Boston; since the shortest route between these two cities passes through the seven removed cities, this solution of the 42-city problem yields a solution of the 49-city problem.

Proctor and Gamble ran a contest in 1962. The contest required solving a TSP on a specified 33 cities. There was a tie between many people who found the optimum. An early TSP researcher, Gerald Thompson, was one of the winners.

Groetschel (1977) found the optimal tour of 120 cities from what was then West Germany.

Padberg and Rinaldi (1987) found the optimal tour of 532 AT&T switch locations in the USA.

Groetschel and Holland (1988) found the optimal tour of 666 interesting places in the world.

Applegate, Bixby, Chvatal, and Cook (1998) found the optimal tour of the 13,509 cities in the USA with populations > 500.

See also in **Annotated Bibliography*.***

Annotated Bibliography

We present below a bibliography of work on the solution of the traveling salesman problem (TSP). We have attempted to give a complete collection of the papers that deal directly with solution methods, but we have also included a number of historically important papers on problem formulations, heuristic methods, and structural results. For a broader collection of papers we refer the reader to the excellent book *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*, editited by E. L Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan, and D. B. Shmoys and to the recent bibliography "The traveling salesman problem" by M. Junger, G. Reinelt, and G. Rinaldi (in *Annotated Bibliographies in Combinatorial Optimization*, edited by M. Dell'Amico, F. Maffioli, and S. Martello).

1932

K. Menger, "Das botenproblem", in *Ergebnisse eines Mathematischen Kolloquiums* **2** (K. Menger, editor), Teubner, Leipzig, pages 11-12.

> This volume contains a statement of a problem posed by Karl Menger on February 5, 1930, at a mathematical colloquium in Vienna. Several historical studies of the TSP (Bock (1963), Hoffman and Wolfe (1985)) point to this statement as a precursor of the TSP. Bock (1963) translates Menger's statement as: "We designate as the Messenger Problem (since this problem is encountered by every postal messenger, as well as by many travelers) the task of finding, for a finite number of points whose pairwise distances are known, the shortest path connecting the points. This problem is naturally always solvable by making a finite number of trials. Rules are not known which would reduce the number of trials below the number of permutations of the given points. The rule, that one should first go from the starting point to the point nearest this, etc., does not in general result in the shortest path."

1940

P. C. Mahalanobis, "A sample survey of the acreage under jute in Bengal", *Sankhyu* **4**, 511-530.

> The well-known statistician Mahalanobis discusses some aspects of TSP solutions through randomly chosen locations in the Euclidean plane. This work was in connection with a survey of farm lands in Bengal that took place in 1938, where one of the major costs in carrying out the survey was the transportation of men and equipment from one survey point to the next. This work appears to independent of the Merill Flood led efforts on TSP applications that took place several years earlier.

1942

R. J. Jessen, "Statistical investigation of a sample survey for obtaining farm facts", *Research Bulletin* #304, Iowa State College of Agriculture.

> The title suggests that this report deals with a problem similar to the application studied by Mahalanobis. If this is the case, there may be a case for referring to the TSP as the "traveling farmer's problem." We were not able to obtain a copy of this paper. We would certainly appreciate receiving any information on how we could locate this historically interesting paper.

1949

J.B. Robinson, "On the Hamiltonian game (a traveling-salesman problem)", *RAND Research Memorandum* RM-303.

This paper is the earliest reference we could find that uses the term "traveling salesman problem" in the context of mathematical optimization. The introduction to the paper, however, makes it clear that the TSP was already a well-known problem at that time (at least at the RAND Corporation). The paper contains an algorithm for solving a variation of the assignment problem (given a complete directed graph, with weights on the arcs, find a minimum-weight set of disjoint directed circuits that cover the nodes of the graph) is described. The author writes that she was led to the solution of the problem in an unsuccessful attempt to solve the TSP. In the introduction to the paper, the TSP is described as follows: "One formulation is to find the shortest route for a salesman starting from Washington, visiting all the state capitals and then returning to Washington." It is interesting to note that this is the instance that was solved several years later by Dantzig, Fulkerson, and Johnson.

1954

G. Dantzig, R. Fulkerson, and S. Johnson, "Solution of a large-scale traveling-salesman problem", *Operations Research* **2**, 393-410.

This is the granddaddy of TSP papers. It reports on the solution of a 49-city TSP via linear-programming methods. Many of the ideas used to solve integer programming problems can be traced back to this paper.

I. Heller, "The travelling salesman's problem: part 1 -- basic facts", *Research Report*, George Washington University Logistics Reserch Project.

An 88-page research report containing many basic results on the asymmetric TSP polytope. In the author's words, it is "an assembly of facts and tools, hence, a working paper in the literal sense of the word." (We thank Hernan Abeledo for kindly locating this manuscript for us at George Washington University.)

1955

I. Heller, "On the travelling salesman's problem", *Proceedings of the Second Symposium in Linear Programming (Volume 1)*, Washington, D.C., January 27-29.

This manuscript discusses linear systems for the TSP polytope, and some neighbor relations for the asymmetric TSP polytope.

H.W. Kuhn, "On certain convex polyhedra", Abstract 799t, *Bulletin of the American Mathematical Society* **61,** 557-558.

The author announces a complete description of the 5-city asymmetric TSP polytope. (Several corrections are discussed in Kuhn [1991].)

G. Morton and A.H. Land, "A contribution to the `travelling-salesman' problem", *Journal of the Royal Statistical Society, Series B* **17,** 185-194.

The main portion of the paper discusses a linear programming approach to the TSP, but it also contains several interesting side discussions. In one of these discussions, the authors present their version of 3-opt: "We have developed a method for investigating the change in cost of all three-in, three-out changes which are possible without destroying the circuit." In another discussion, they introduce the capacitated vehicle routing problem. The authors also make several interesting historical remarks. They write that the TSP "may be familiar to statisticians as the `Mean Minimum Distance' problem", and they give several references dating back to 1942. They also write that they began their own study of the TSP independently from the work that was going on in the United States: "we used to call it the laundry van problem, where the conditions were a daily service by a one-van laundry."

R.Z. Norman, "On the convex polyhedra of the symmetric traveling salesman problem", Abstract 804t, *Bulletin of the American Mathematical Society* **61,** 559-559.

The abstract announces complete descriptions of n-city TSP polytopes, for $n <= 7$. (The system for $n = 7$ was later shown to be insufficient by Boyd and Cunningham [1991], Fleischmann [1988], and Naddef and Rinaldi [1992].)

J.T. Robacker, "Some experiments on the traveling-salesman problem", *RAND Research Memorandum* RM-1521.

The author carries out computational tests of the Dantzig-Fulkerson-Johnson method by hand on 10 instances, each having 9 cities. His report that "the average time to work one of the problems was about 3 hours" is cited as a benchmark for the next few years of computational work on the TSP. He also describes the cheapest-insertion heuristic, giving credit for it to A. W. Boldyreff.

1956

M.M. Flood, "The traveling-salesman problem", *Operations Research* **4,** 61-75.

Some heuristic methods for obtaining good tours are discussed, including the nearest-neighbor algorithm and 2-opt. It is very interesting to read the manuscript with knowledge of the results that have come afterwards. For example, the author writes: "It seems very likely that a quite different approach from any yet used may be required for successful treatment of the problem. In fact, there may be no general method for treating the problem and impossibility results would also be valuable."

J.B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem", *Proceedings of the American Mathematical Society* **2,** 48-50.

This is the famous Kruskal's Algorithm for computing minimum-length spanning trees. The author writes that the computation of minimum spanning trees is interesting because that problem "on the surface is closely related to the well-known traveling salesman problem."

1957

L.L. Barachet, "Graphic solution of the traveling-salesman problem", *Operations Research* **5,** 841-845.

The author describes an enumeration scheme for computing near-optimal tours and makes several observations on the structure of optimal tours for geometric problems. A tour for a 10-city instance is solved by hand to demonstrate the practicality of the method for small instances.

1958

F. Bock, "An algorithm for solving `traveling-salesman' and related network optimization problems", *Research Report*, Armour Research Foundation. (Presented at the Operations Research Society of America Fourteenth National Meeting, St. Louis, October 24, 1958.)

A 3-opt algorithm is described, together with an enumeration scheme for computing an optimal tour. The author tests his algorithm on the examples of Robacker and Barachet, as well as on a new 10-city instance. The tests were carried out on an IBM 650 computer. The author notes that the method had an "impractically slow rate of convergence on Dantizig's 42-city problem." The 42-city instance is derived from the

49-city Dantzig-Fulkerson-Johnson example via a simple reduction. (We thank Jan Karel Lenstra for kindly providing us with a copy of this manuscript.)

G.A. Croes, "A method for solving traveling-salesman problems", *Operations Research* **6,** 791-812.

A variant of 3-opt is proposed, together with an enumeration scheme for computing an optimal tour. The authored solved the Dantzig-Fulkerson-Johnson 42-city example in 70 hours by hand. He also solved several of the Robacker examples in an average time of 25 minutes per example.

W.L. Eastman, "Linear programming with pattern constraints", Ph.D. Dissertation, Harvard.

A branch-and-bound algorithm using the assignment problem to obtain lower bounds is described. The branching step chooses a subtour from the solution to the assignment problem, and considers, in turn, setting $x_e = 0$ for each edge e in the subtour. The algorithm is tested on examples having up to 10 cities.

M.J. Rossman and R.J. Twery, "A solution to the travelling salesman problem", *Operations Research* **6**, page 687, Abstract E3.1.3.

The authors announce an implicit enumeration algorithm for the TSP. The algorithm was used to solve a 13-city instance by hand.

1959

G.B. Dantzig, D.R. Fulkerson, and S.M. Johnson, "On a linear-programming, combinatorial approach to the traveling-salesman problem", *Operations Research* **7,** 59-66.

A step-by-step application of the Dantzig-Fulkerson-Johnson algorithm is given for Barachet's 10-city example. They write: "judging from the number of queries we have received from readers, this method was not elaborated sufficiently to make the proposal clear."

W. Riley, III, "A new approach to the traveling-salesman problem", *Operations Research (Supplement 1)* **7**, Abstract B4.

The author announces an algorithm for the TSP. He writes: "The algorithm employs the concept of basic minimal change that can be made to a given route. Successive examination of particular subsets of these minimal changes are found to form the optimum (minimum or maximum) route quickly." The details of the algorithm are not given in the abstract.

1960

R. Bellman, "Combinatorial processes and dynamic programming", in: *Combinatorial Analysis* (R. Bellman and M. Hall, Jr., eds.), American Mathematical Society, pp. 217-249.

Bellman uses the TSP as an example of a combinatorial problem that can be solved via dynamic programming. He writes: "It follows that with current machines it would be possible to solve problems of this type in a direct fashion for $N <= 17$."

M.F. Dacey, "Selection of an initial solution for the traveling-salesman problem", *Operations Research* **8,** 133-134.

The author announces a new heuristic algorithm for the TSP. The details of the algorithm were given in a technical report from the Department of Geography, University of Washington. He reports that on the 10 Robacker instances, the solutions found by the heuristic were on average 4.8 percent longer than the optimal solutions.

F. Lambert, "The traveling-salesman problem", *Cahiers du Centre de Recherche Opérationelle* **2,** 180-191.

A 5-city example of the TSP is solved using Gomory cutting planes.

C.E. Miller, A.W. Tucker, and R.A. Zemlin, "Integer programming formulation of traveling salesman problems", *Journal of the Association for Computing Machinery* **7,** 326-329.

The authors present an integer programming formulation of the TSP and report their computational experience on solving several small problems using Gomory's cutting-plane algorithm. The largest instance solved was the 10-city example of Barachet.

W. Riley, III, "Micro-analysis applied to the travelling salesman problem", *Operations Research (Supplement 1)* **8**, Abstract D4.

This is a continuation of the author's earlier work. The details of his method are not given.

1961

E.L. Arnoff and S.S. Sengupta, "Mathematical programming", in: *Progress in Operations Research* (R.L. Ackoff, editor), John Wiley and Sons, New York, pp. 105-210.

A good survey of the computational work on the TSP that was carried out in the 1950's.

H. Müuller - Merbach, "Die ermittlung des kürzesten rundreiseweges mittels linearer programmierung", *Ablauf und Planungsforschung* **2,** 70-83.

An algorithm for the asymmetric TSP is proposed. It is illustrated on a 7-city example.

B. Thüring, "Zum problem der exakten ermittlung des kürzesten rundreiseweges", *Elektronische Datenverarbeitung* **3,** 147-156.

A TSP algorithm is described and illustrated on a 10-city example.

1962

R. Bellman, "Dynamic programming treatment of the travelling salesman problem", *Journal of the Association for Computing Machinery* **9,** 61-63.

Some additional remarks on dynamic programming algorithms for the TSP are given.

R.H. Gonzales, "Solution to the traveling salesman problem by dynamic programming on the hypercube", *Technical Report* Number 18, Operations Research Center, Massachusetts Institute of Technology.

The author solved instances with up to 10 cities using dynamic programming. The tests were carried out on an IBM 1620 computer.

M. Held and R.M. Karp, "A dynamic programming approach to sequencing problems", *Journal of the Society of Industrial and Applied Mathematics* **10,** 196-210.

Dynamic programming algorithm are described for solving small instances and for finding approximate solutions to larger instances. The exact algorithm was used to solve 13-city instances on an IBM 7090 computer. The approximation algorithm (also programmed for the IBM 7090) found the optimal solution to the 42-city Dantzig-Fulkerson-Johnson example on two out of five trials, and was also tested on a new 48-city instance.

1963

F. Bock, "Mathematical programming solution of traveling salesman examples", in: *Recent Advances in Mathematical Programming* (R.L. graves and P. Wolfe, eds.), McGraw-Hill, New York.

> The author gives a sketch of a linear-programming based algorithm for the TSP. The paper contains the now well-known 1930 quote from Karl Menger describing the Hamiltonian path problem.

J.D.C. Little, K.G. Murty, D.W. Sweeney, and C. Karel, "An algorithm for the traveling salesman problem", *Operations Research* **11,** 972-989.

> The authors coin the term branch-and-bound: "The subsets of tours are conveniently represented as the nodes of a tree and the process of partitioning as a branching of the tree. Hence we have called the method `branch-and-bound'." They use a combinatorial bounding method in their algorithm, and they branch on edges being either in or not in the tour. Their algorithm was implemented on an IBM 7090 computer. Some interesting computational tests are given, including the solution of a 25-city problem that was contained in the Held and Karp test set. Their most cited success is the solution of a set of 30-city asymmetric problems having random edge lengths. They also report that 10-city instances can be solved in under 1 hour by hand.

V.I. Mudrov, "A method of solution of the traveling salesman problem by means of integer linear programming (the problem of finding the Hamiltonian paths of shortest length in a complete graph)" (in Russian), *Zhurnal Vychislennoi Fiziki (USSR)* **3,** 1137-1139. (Abstract in: *International Abstracts in Operations Research* **5** (1965), Abstract Number 3330.)

> This is a review of an paper written in Russian. The paper contains an integer programming formulation of the TSP. The author proposes to use this formulation together with Gomory's cutting-plane algorithm to solve TSP instances.

1964

R.L. Karg and G.L. Thompson, "A heuristic approach to solving travelling salesman problems", *Management Science* **10,** 225-248.

> The cheapest-insertion heuristic is described, together with a proposal for using it to improve subsections of tours. The authors employ their heuristic methods on several large instances, including a 57-city instance with data taken from a road atlas.

W. Suurballe, "Network algorithm for the traveling salesman problem", *Operations Research (Supplement 1)* **12**, Abstract 3Tc.1.

> The author announces a enumeration algorithm for computing all optimal tours to TSP instances.

1965

S. Lin, "Computer solutions of the traveling salesman problem", *The Bell System Technical Journal,* 2245-2269.

> This is an important paper in heuristic methods for the TSP. The author defines *k*-optimal tours, and gives an efficient way to implement 3-opt (extending the work of Croes). Computational results are given for instances with up to 105 cities. The paper also includes a discussion about the probability of obtaining optimal tours via repeated trials of the heuristic.

S. Reiter and G. Sherman, "Discrete optimizing", *SIAM Journal on Applied Mathematics* **13,** 864-889.

A local-search algorithm for the TSP is proposed and tested on problems from the literature. The largest instance considered is the 57-city example of Karg and Thompson. In each case, the algorithm was able to produce a tour at least as good as the tours that were reported in earlier papers.

N.P. Salz, "The NORMA: a possible basis for a theory for the traveling-salesman problem", *Operations Research (Supplement 1)* **13**, Abstract C4h.

A framework for studying the TSP is proposed.

1966

R.E. Gomory, "The traveling salesman problem", in: *Proceedings of the IBM Scientific Computing Symposium on Combinatorial Problems*, IBM, White Plains, pp. 93-121.

Gomory's paper includes very nice descriptions of the methods of Dantzig, Fulkerson, and Johnson [1954], Held and Karp [1962], and Little, Murty, Sweeney, and Karel [1963]. The paper also includes the text of a beautiful discussion between Gomory, J. Edmonds, and H.W. Kuhn, concerning the use of linear programming methods to solve combinatorial problems.

E.L. Lawler and D.E. Wood, "Branch-and-bound methods: a survey", *Operations Research* **14,** 699-719.

The paper includes descriptions of the branch-and-bound algorithms of Eastman [1958] and Little, Murty, Sweeney, and Karel [1963]. The authors suggest the use of minimum spanning trees as a lower bound in a branch-and-bound algorithm for the TSP.

G.T. Martin, "Solving the traveling salesman problem by integer linear programming", *Operations Research (Supplement 1)* **14**, Abstract WA7.10.

The author outlines a cutting-plane algorithm for the TSP. The algorithm adds both TSP-specific inequalities (their form is not mentioned in the text, but from the context it would seem that they are subtour inequalities) as well as cutting planes from Gomory's method of integer forms. The Dantzig-Fulkerson-Johnson problem was solved using 9 TSP cuts and 10 Gomory cuts. The author writes: "Advanced programming may well reduce both final model size and number of iterations significantly, but several present-day computers solve such problems in five to ten minutes."

G.T. Martin, "Solving the traveling salesman problem by integer linear programming", C-E-I-R, New York, 1966.

This is a full version of the work announced in the above abstract. We thank John Tomlin for kindly providing us with a copy of this paper.

H. Mueller-Merbach, "Drei neue methoden zur lösung des travelling salesman problems, teil 1", *Ablauf und Planungsforschung* **7,** 32-46.

The author describes some enumerative methods for the TSP.

H. Mueller-Merbach, "Drei neue methoden zur lösung des travelling salesman problems, teil 2", *Ablauf und Planungsforschung* **7,** 78-91.

Continuation of the above paper.

S.M. Roberts and B. Flores, "An engineering approach to the traveling salesman problem", *Management Science* **13,** 269-288.

The authors present an enumerative heuristic for the TSP. Using this algorithm they obtained a tour for Karg and Thompson's 57-city example, having cost equal to the best tour found by Karg and Thompson.

N.P. Salz, "The NORMA: a theory for the traveling salesman problem", *Operations Research (Supplement 2)* **14**, Abstract MP3.11.

Some additional results on the author's framework for studying the TSP.

D. Shapiro, "Algorithms for the solution of the optimal cost traveling salesman problem", Sc.D. Thesis, Washington University, St. Louis.

The authors describes an algorithm similar to Eastman's branch-and-bound algorithm, but with some additional ideas to improve the performance on symmetric instances.

1967

B. Fleischmann, "Computational experience with the algorithm of Balas", *Operations Research* **15,** 153-154.

This paper includes computational results for 6-city and 7-city instances of the TSP.

1968

M. Bellmore and G.L. Nemhauser, "The traveling salesman problem: a survey", *Operations Research* **16,** 538-558.

The authors present an extensive survey of algorithms for the TSP. They write: "If the authors were faced with the problem of finding a solution to a particular traveling salesman problem we would use dynamic programming for problems with 13 cities or less, Shapiro's branch-and-bound algorithm for larger problems (up to about 70-100 cities for asymmetric problems and up to about 40 cities for symmetric problems) and Shen Lin's `3-opt' algorithm for problems that cannot be handled by Shapiro's algorithm."

P. Pfluger, "Diskussion der modellwahl am beispiel des traveling-salesman problems", in: *Einführung in die Methode Branch and Bound* (M. Bechmann and H.P. Künzi, eds.), *Lecture Notes in Operations Research and Mathematical Economics* **4**, Springer, Berlin, pp. 88-106.

The author gives a short description of the TSP algorithms of Bellman [1962], Held and Karp [1962], Eastman [1958], and Little, Murty, Sweeney, and Karel [1963].

1969

A.M. Issac and E. Turban, "Some comments on the traveling salesman problem", *Operations Research* **17,** 543-546.

This paper contains a short survey of some TSP heuristic methods that were not covered in Bellmore and Nemhauser [1968].

T.C. Raymond, "Heuristic algorithm for the traveling-salesman problem", *IBM Journal of Research and Development* **13,** 400-407.

This author develops some extensions to Karg and Thompson's heuristic for the TSP. Computational results are reported for instances having up to 57 cities.

1970

M. Held and R.M. Karp, "The traveling-salesman problem and minimum spanning trees", *Operations Research* **18,** 1138-1162.

This paper introduces the 1-tree relaxation of the TSP and the idea of using node weights to improve the bound given by the optimal 1-tree. It is also shown that the best bound that can be obtained in this manner is equal to the optimal value of the subtour linear-programming relaxation of the TSP. Two methods are presented for computing the bound, but neither of them were reported to perform well on computational tests. (The efficient algorithm is given in Held and Karp [1971].)

H. Müller-Merbach, *Optimale Reihenfolgen*, Springer, Berlin.

This book includes a long chapter on the traveling salesman. The treatment covers both exact and heuristic algorithms (including computational tests).

1971

M. Bellmore and J. Malone, "Pathology of traveling-salesman subtour-elimination algorithms", *Operations Research* **19,** 278-307.

The authors present an analysis of the effectiveness of using the assignment relaxation for the TSP, partially explaining the success earlier algorithms had in solving random asymmetric instances. They propose to use the 2-factor problem as a relaxation for symmetric problems, and tested this idea on random Euclidean problems having up to 20 nodes (using a b-matching code written by Edmonds and Johnson).

M. Held and R.M. Karp, "The traveling-salesman problem and minimum spanning trees: part II", *Mathematical Programming* **1,** 6-25.

An efficient iterative method for computing a good 1-tree bound (using node weights) is presented. The authors imbed this in a branch-and-bound algorithm and solve, amongst others, the 42-city instance of Dantzig, Fulkerson, and Johnson [1954], the 57-city instance of Karg and Thompson [1964], and a 64-city random Euclidean instance. These computational results were easily the best reported up to that time.

P. Krolak, W. Felts, and G. Marble, "A man-machine approach toward solving the traveling salesman problem", *Communications of the ACM* **14,** 327-334.

A nice paper describing a practical approach to finding good soltions to geometric TSP instances, involving human generated tours, the assignment problem relaxation, node insertion, 2-opt, and a "regional improvement routine" (where the human identifies a region in the current tour that looks suboptimal, and the a subroutine is called to optimize that region and glue it to the full tour).

1972

N. Christofides, "Bounds for the travelling-salesman problem", *Operations Research* **20,** 1044-1056.

S. Hong, "A linear programming approach for the traveling salesman problem", Ph.D. Thesis, The Johns Hopkins University.

Hong's thesis was written under the supervision of M. Bellmore. His work is the most significant (computational) contribution to the linear programming approach to the TSP since the original paper of Dantzig, Fulkerson, and Johnson [1954]. The algorithm presented here goes a long way towards automating Dantzig, Fulkerson, and Johnson's method. Hong uses a dual LP algorithm for solving the linear-programming relaxations; the Ford-Fulkerson max-flow algorithm for finding violated subtour inequalities; a heuristic for finding violated blossom inequalities; and a branch-and-bound scheme that includes the addition of subtour inequalities at the nodes of the branch-and-bound tree (such algorithms are now known as "branch-and-cut" (Padberg and Rinaldi [1991])). In short, Hong had most of the ingredients of the current generation of linear-programming based algorithms for the TSP. His computational tests were carried out on random Euclidean instances having up to 20 cities. On the 60 instances that he tests, 59 were solved without branching and the remaining instance required a single branch. Larger instances were not tested due to difficulties with his LP solver. (We thank Leslie Hall for kindly obtaining this manuscript for us at Johns Hopkins University.)

H. Steckhan and R. Thome, "Vereinfachungen der Eastmanischeen branch-bound-lösung für symmetrische traveling salesman probleme", *Methods of Operations Research* **14,** 360-389.

Some observations on the branch-and-bound algorithm of Eastman [1958]. Computational results on instances having up to 10 cities are given.

1973

S. Lin and B.W. Kernighan, "An effective heuristic algorithm for the traveling-salesman problem", *Operations Research* **21,** 498-516.

1974

K. Helbig Hansen and J. Krarup, "Improvements of the Held-Karp algorithm for the symmetric traveling-salesman problem", *Mathematical Programming* **7,** 87-96.

The authors test their version of Held-Karp on the 57-city instance of Karg and Thompson [1964] and a set of instances having with having random edge lengths.

M. Held, P. Wolfe, and H.P. Crowder, "Validation of subgradient optimization", *Mathematical Programming* **6,** 62-88.

The iterative algorithm of Held and Karp [1971] is examined in detail. The paper includes some computational results on the 42-city instance of Dantzig, Fulkerson, and Johnson [1954].

P.M. Camerini, L. Fratta, and F. Maffioli, "On improving relaxation methods by modified gradient techniques", *Mathematical Programming Study* **3,** 26-34.

1975

W. Schiebel, G. Unger, and J. Terno, "A cutting plane algorithm for the travelling salesman problem", *Report* 07-15-76, Sektion Mathematik, Technische Universität Dresden, GDR.

1976

P. Miliotis, "Integer programming approaches to the travelling salesman problem", *Mathematical Programming* **10,** 367-378.

The author uses a linear programming package written Land and Powell [1973] to implement a branch-and-cut algorithm for the TSP, using subtour inequalities. Computational results for the 42-city instance of Dantzig, Fulkerson, and Johnson [1954], the 48-city instance of Held and Karp [1962], and the 57-city instance of Karg and Thompson [1964].

1977

M.S. Bazaraa and J.J. Goode, "The traveling salesman problem: a duality approach", *Mathematical Programming* **13,** 221-237.

M. Grötschel, *Polyedrische Charackterisierungen kombinatorischer Optimierungsprobleme*, Anton Hain Verlag, Meisenheim/Glan.

This is Grötschel's thesis. It contains much of the polyhedral work on the TSP that was carried out jointly with M. Padberg. It also contains the solution of a 120-city instance by means of a cutting-plane algorithm, where cuts (subtour inequalities and comb inequalities) were detected and added by hand to the linear programming relaxation. This demonstrated the power on linear programming approach on an instance significantly larger than the 42-city instance solved by Dantzig, Fulkerson, and Johnson [1954].

T.H.C. Smith and G.L. Thompson, "A LIFO implicit enumeration search algorithm for the symmetric traveling salesman problem using Held and Karp's 1-tree relaxation", in: *Studies in Integer Programming* (P.L. Hammer, E.L. Johnson, B.H. Korte, and G.L. Nemhauser, eds.), *Annals of Discrete Mathematics***1**, North-Holland, Amsterdam, pp. 479-493.

The authors present some improvements to the Held-Karp algorithm, and test their methods on a variety of examples, including the 57-city instance of Karg and Thompson [1964] and a set of ten 60-city random Euclean instances.

1978

M. Grötschel and M. Padberg, "On the symmetric traveling salesman problem: theory and computation", in: *Optimization and Operations Research* (R. Henn, B. Korte, and W. Oettli, eds.), *Lecture Notes in Economics and Mathematical Systems* **157,** Springer, Berlin, pp. 105-115.

The authors give a short survey of their work on cutting-planes for the TSP and present some computational results that demonstrate the effectiveness of the cutting planes.

P. Miliotis, "Using cutting planes to solve the symmetric travelling salesman problem", *Mathematical Programming* **15,** 177-188.

The algorithm uses a combination of subtour cuts and Gomory cuts, similar to the method proposed by Martin [1966]. Tests were made on instances that included the standard 42-city, 48-city, and 57-city examples.

1979

R.E. Burkard, "Travelling salesman and assignment problems: a survey", in: *Discrete Optimization 1* (P.L. Hammer, E.L. Johnson, and B.H. Korte, eds.), *Annals of Discrete Mathematics* **4**, North-Holland, Amsterdam, pp. 193-215.

This survey includes algorithms for the TSP and the asymmetric TSP.

A. Land, "The solution of some 100-city travelling salesman problems", Technical Report, London School of Economics.

The author describes a cutting-plane algorithm for the TSP. The linear-programming relaxations are solved in integer arithmetic, thus avoiding rounding errors in the computations. The separation algorithms include a shrinking heuristic for identifying subtour inequalities and a heuristic for identifying blossom inequalities (looking at the connected components of the graph obtained by deleting the edges assigned the value 0 or 1 in the solution to the linear-programming relaxation). If no subtours or blossoms are found, a Gomory-cut is added to the relaxation. She uses column generation to handle the great number of edges present in larger instances. The algorithm is tested on ten 100-city random Euclidean instances.

1980

H. Crowder and M.W. Padberg, "Solving large-scale symmetric travelling salesman problems to optimality", *Management Science* **26,** 495-509.

This work improves on the earlier study of Padberg and Hong [1980], including the addition of further rounds of cutting planes. The IBM MPSX-MIP/370 integer-programming solver is used to carry out a branch-and-bound search on the final

linear programming relaxation. If the integral solution found by this search is not a tour, then the subtour inequalities violated by the solution are added to the relaxation and branch-and-bound is called again. The computational results obtained in this study are very impressive, including the solution of a 318-city instance described in Lin and Kernighan [1973]. This 318-city instance would remain until 1987 as the largest TSP solved.

M. Grötschel, "On the symmetric travelling salesman problem: solution of a 120-city problem", *Mathematical Programming Study* **12,** 61-77.

This is a journal version of the computational study described in Grötschel [1977]. It is very impressive that this 120-city instance could be so with the addition of so few cutting planes.

D.J. Houck, Jr., J.C. Picard, M. Queyranne, and R.R. Vemuganti, "The travelling salesman problem as a constrained shortest path problem: theory and computational experience", *OPSEARCH* **17,** 93-109.

M.W. Padberg and S. Hong, "On the symmetric travelling salesman problem: a computational study", *Mathematical Programming Study* **12,** 78-107.

A cutting-plane algorithm is described. The algorithm makes use of new separation routines for comb inequalities. Like Land [1979], the linear programming computations were carried out using integer arithmetic to "avoid any problems connected with round-off errors." In their computational study, 54 out of total of 74 instances were solved by the linear programming relaxation. For the 318-city example of Lin and Kernighan [1973], the bound obtained via the relaxation was within a factor of 0.96 of the best tour that was found.

1982

J. Mohan, "A study in parallel computation - the traveling salesman problem", Technical *Report* CMU-CS-82-136, Computer Science Department, Carnegie Mellon University, 1982.

We have not been able to obtain a copy of this report.

T. Volgenant and R. Jonker, "A branch and bound algorithm for the symmetric traveling salesman problem based on the 1-tree relaxation", *European Journal of Operational Research* **9,** 83-89.

A variation of the Held-Karp algorithm is described, together with computational results for a number of small instances, including the ten 60-city examples of Smith and Thompson [1977].

1983

B. Gavish and K.N. Srikanth, "Algorithms for solving large-scale symmetric traveling salesman problems to optimality", *Working Paper Series* Number QM8329, Graduate School of Management, University of Rochester.

We have not been able to obtain this manuscript.

T. Volgenant and R. Jonker, "The symmetric traveling salesman problem and edge exchanges in minimal 1-trees", *European Journal of Operational Research* **12,** 394-403.

1984

R. Jonker and T. Volgenant, "Nonoptimal edges for the symmetric traveling salesman problem", *Operations Research* **32,** 837-846.

1985

B. Fleischmann, "A cutting plane procedure for the travelling salesman problem on road networks", *European Journal of Operational Research* **21,** 307-317.

> A linear-programming relaxation is built using TSP-specific cuts described in the paper. If no such cut is found, then a Gomory-cut will be added. Computational results are given for a number of TSP instances from the literature (including the 120-city example of Grötschel [1977]).

1987

O.A. Holland, "Schnittebenenverfahren für travelling-salesman und verwandte probleme", Dotoral Thesis, Universität Bonn.

M. Grötschel and O. Holland, "A cutting plane algorithm for minimum perfect 2-matchings", *Computing* **39,** 327-344.

M. Padberg and G. Rinaldi, "Optimization of a 532-city symmetric traveling salesman problem by branch and cut", *Operations Research Letters***6,** 1-7.

1988

G. Carpaneto, M. Fischetti, and P. Toth, "New lower bounds for the symmetric travelling salesman problem", *Mathematical Programming* **45,** 223-254.

1989

J. Rost and E. Maehle, "Implementation of a parallel branch-and-bound algorithm for the travelling salesman problem", in: *CONPAR 88: Proceedings*, (C.R. Jesshop and K.D. Reimartz, eds.), *The British Computer Society Workshop Series*, Cambridge University Press, New York, pp. 152-159.

1990

T.H.C. Smith, T.W.S. Meyer, "Lower bounds for the symmetric travelling salesman problem from lagrangean relaxations", *Discrete Applied Mathematics* **26,** 209-217.

T. Volgenant and R. Jonker, "Fictitious upper bounds in an algorithm for the symmetric traveling salesman problem", *Computers and Operations Research* **17,** 113-117.

M. Padberg and G. Rinaldi, "An efficient algorithm for the minimum capacity cut problem", *Mathematical Programming* **47,** 19-36.

M. Padberg and G. Rinaldi, "Facet identification for the symmetric traveling salesman polytope", *Mathematical Programming* **47,** 219-257.

1991

M. Grötschel and O. Holland, "Solution of large-scale symmetric travelling salesman problems", *Mathematical Programming* **51,** 141-202.

M. Padberg and G. Rinaldi, "A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems", *SIAM Review* **33,** 60-100.

1992

S. Tschöke, M. Räcke, R. Lüling, and B. Monien, "Solving the traveling salesman problem with a parallel branch-and-bound algorithm on a 1024 processor network", *Technical Report*, Department of Mathematics and Computer Science, University of Paderborn, Germany, 1992.

1993

J.-M. Clochard and D. Naddef, "Using path inequalities in a branch and cut code for the symmetric traveling salesman problem", in *Third IPCO Conference*, (G. Rinaldi and L. Wolsey, eds), pp. 291-311.

1994

M. Jünger, S. Thienel, and G. Reinelt, "Provably good solutions for the traveling salesman problem", *Zeitschrift für Operations Research* **40,** 183-217.

1995

D. Applegate, R. Bixby, V. Chvátal, and W. Cook, "Finding cuts in the TSP (A preliminary report)", *DIMACS Technical Report* 95-05, March.

> A detailed description of several separation algorithms for combs and clique trees. These algorithms were used by the authors to solve a series of TSPLIB test instances, including pcb3038, fnl4461, and pla7392. Certificates of the optimality of these three large instances were made available on the internet by the authors.

Computer Codes

Description

Preface

The TSMP-1 (Traveling SalesMan Problem) program is intended for solving the problem of the precise determination of the length and path of a minimal Hamiltonian cycle (cycles) of a weighed network (the Traveling Salesman Problem, the TSP) in a time that polynomially depends on the network's dimension (the number of its nodes); and it also relates to the construction (on this basis) of algorithms of polynomial complexity to solve so-called NP-complete problems of discrete mathematics.

The essence of this problem is as follows: to find in a network given a cycle sequence of passing the edges in such a way that it includes all the nodes of the network one and only one time (i.e. the passage of the edges is a Hamiltonian cycle), and the sum of the edge weights of the cycle under consideration (the length of the cycle) is minimal among all possible cycles of the network (at least not greater than the length of any other cycles having similar properties).

Premises of Creation of the TSMP-1

A precise method is known to solve the TSP for a network with no bounds on its edge weights. This method is based on constructing a complete set of the network's cycles. Since the number of these cycles is increased as N!, where N is the number of the network's nodes, modern computers are unable to construct and evaluate a solution set within an acceptable time period unless the problem's dimension is not large (for example, the direct solving the TSP by a search all cycles cannot be performed unless the number of the network's nodes is not greater than 30—50). If N is greater than 50, then the continuous computational process to be realized even on a very fast computer for determining the length of all the network's cycles may last for a very long time (months or even years). This method is described both below and in the program body as the Exp. Method.

The fact that the precise solution of the TSP is impossible to be found without constructing a complete set of the Hamiltonian cycles is explained in theoretical cybernetics by the fact that there is a class of NP-complexity problems including the TSP.

However, up to now there is no proof that problems of this class cannot be solved precisely in an acceptable time (by means of algorithms of polynomial complexity). According to Cook's theorem (Cook, S.A. The complexity of theorem-proving procedures, Proc. 3rd Ann. ACM Symp. on Theory of Computing machinery, New York, 151—158 (1.5; 2.6; 3.1.1; 5.2; A1.4; A9.1)), the existence of an algorithm of polynomial complexity to precisely solve the TSP with no bounds on the network's edge weights allows us to affirm that we can construct an algorithm of polynomial complexity which also precisely solves all other NP-complete problems (some of them with the aid of the Traveling SalesMan algorithm). This method is described both below and in the program as the Polyn. Method.

Notions, Definitions, and Notation

An initial weighed non-oriented Hamiltonian network is given by weights' matrix |C| which elements (edge weights) can be positive and negative integer numbers. The network is not Eucledian.

Hamiltonian network
A network is Hamiltonian if it is possible to solve the Hamiltonian problem for it (the determining of Hamiltonian cycles and paths as well as cycles and paths with required properties).

We will differ the following classes of Hamiltonian networks:

strongly-connected undirected weighed network;
quasi-strongly-connected undirected unweighed network;
quasi-strongly-connected directed unweighed network;
strongly-connected directed weighed network.

A network can be Hamiltonian if it is partially directed and also is quasi-strongly-connected and weighed. In the notation of networks *N* is the number of its nodes. Euclidean networks form a subclass of the weighed networks. A characteristic feature of Euclidean networks is relations between the weights of any three interconnected edges. These edge weights satisfy the rule of triangle.

Hamiltonian Cycles
These cycles can be defined in Hamiltonian networks. A cyclic sequence of passing the edges is said to be a Hamiltonian cycle if it includes all nodes of a network. The cycle given passes through all nodes of the network (i.e. includes all nodes).

A Hamiltonian cycle *H* is characterized by its passage and is written in the form of an edge sequence $H(1–2–3–...–(N–1)–N–1)$.

Further, the notation of *H* for a cycle will denote its passage and size. If it is required to point out the passage of a cycle, then the cycle will be denoted as a sequence of edges. In a Hamiltonian network a minimal cycle (H_{min}) and a maximal cycle (H_{max}) can be defined. The length of the cycle is the sum of the weights of all the cycle’s edges.

This book gives no information regarding a computer solving of the TSP and a number of other problems. The author informs with satisfaction that this work has been completed and a package of software programs is available for solving the following problems:

— minimum Hamiltonian cycle in a weighed undirected network (the TSP);
— Hamiltonian cycle in an unweighed undirected network;
— minimum Hamiltonian path in a weighed undirected network;
— minimum Hamiltonian path between two given nodes of a network;

— Hamiltonian path in a network;

— Country Postman;

— minimum Hamiltonian cycle with a bound on the edge weights of the path;

— maximum Hamiltonian cycle in a network;

— maximum Hamiltonian path in a network;

— Hamiltonian cycles of a network which do not exceed the given magnitude C;

— minimum Hamiltonian path with a bound on the edge weights of the path;

— maximum Hamiltonian path between two given nodes of a network;

— edges taking part in the formation of minimum Hamiltonian cycles of a network. Minimum Hamiltonian cycle including a given edge of this set;

— edges belonging to the Hamiltonian cycle of a G_N network. Hamiltonian cycle including a given edge of the set.

The information on timings and other characteristics of the software programs will be given in further publications. The main algorithms described and used for solving the problems given have been registered as an US Patent Application № 09/006,367. For more information see also I.I.G., Inc. Home Page (www.pcgrate.com).

TSMP-1 Documentation

Main Menu Items

File menu commands

The File menu offers the following commands:

New Test Window
Creates a new test window in which you can compare the exp. and polyn. methods using pseudorandom networks.

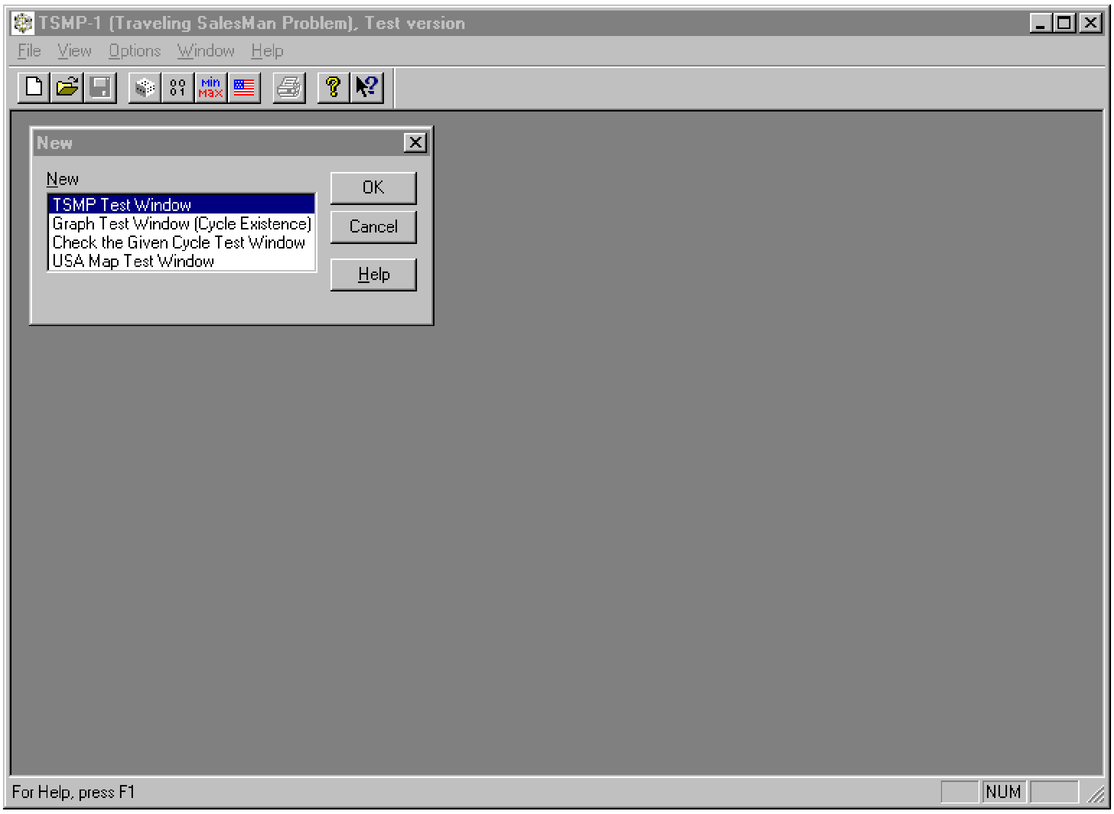
Open
Opens an existing test: creates a test window and loads the data previously saved in the IIG-file.

Close
Closes the open test window.

Save As
Saves the current test data under a specified file name.

USA Demo
Displays a dialog box in which you can find a minimum or maximum route through USA state capital cities.

Exit
Exits the TSMP-1.



View menu commands

The View menu offers the following commands:

Toolbar
Shows or hides the toolbar

Status Bar
Shows or hides the status bar.

Window menu commands

The Window menu offers the following commands, which enable you to arrange multiple views of multiple documents in the application window:

Cascade
Arranges windows in an overlapped fashion.

Tile
Arranges windows in non-overlapped tiles.

Arrange Icons
Arranges icons of closed windows.

Help menu commands

The Help menu offers the following commands, which provide you assistance with this application:

Index
Offers you an index to topics on which you can get help.

Using Help
Provides general instructions on using help.

About
Displays the version number of this application.

Shortcuts

File Open command (File menu)

Toolbar: File Open

Keys: CTRL+O

File Open dialog box

The following options allow you to specify which file to open:

File Name
Type or select the filename you want to open. This box lists files with the extension you select in the List Files of Type box.

List Files of Type
Select the type of file you want to open: *.IIG

Drives
Select the drive in which the TSMP-1 stores the file that you want to open.

Directories
Select the directory in which the TSMP-1 stores the file that you want to open.

Close command (File menu)

Use this command to close all windows containing the active document. The TSMP-1 suggests that you save changes to your document before you close it. If you close a document without saving, you lose all changes made since the last time you saved it. Before closing an untitled document, the TSMP-1 displays the Save As dialog box and suggests that you name and save the document.

You can also close a document by using the Close icon on the document's window, as shown below:
{bml scmenu.bmp}

Save As command (File menu)

Use this command to save and name the active document. The TSMP-1 displays the Save As dialog box so you can name your document.

Shortcuts
Toolbar: File Save
Keys: CTRL+S

File Save As dialog box

The following options allow you to specify the name and location of the file you're about to save:

File Name
Type a new filename to save a document with a different name. A filename can contain up to eight characters and an extension of up to 3 characters. The TSMP-1 adds the extension you specify in the Save File As Type box.

Drives
Select the drive in which you want to store the document.

Directories
Select the directory in which you want to store the document.

1, 2, 3, 4 command (File menu)

Use the numbers and filenames listed at the bottom of the File menu to open the last four documents you closed. Choose the number that corresponds with the document you want to open.

Exit command (File menu)

Use this command to end your TSMP-1 session. You can also use the Close command on the application Control menu.

Shortcuts
Mouse: Double-click the application's Control menu button.

{bmc appexit.bmp}
Keys: ALT+F4

Toolbar command (View menu)

Use this command to display and hide the Toolbar, which includes buttons for some of the most common commands in the TSMP-1, such as File Open. A check mark appears next to the menu item when the Toolbar is displayed.

See Toolbar for help on using the toolbar.

Toolbar
{bml hlptbar.bmp}

The toolbar is displayed across the top of the application window, below the menu bar. The toolbar provides quick mouse access to many tools used in the TSMP-1.

To hide or display the Toolbar, choose Toolbar from the View menu (ALT, V, T).

Add or remove toolbar buttons from the list below according to which ones your application offers.

Click To

{bmc filenew.bmp} Opens a new document.

{bmc fileopen.bmp} Opens an existing document. The TSMP-1 displays the Open dialog box, in which you can locate and open the desired file.

{bmc filesave.bmp} Saves the active document.

{bmc fileprnt.bmp} Prints the active document.

Status Bar command (View menu)

Use this command to display and hide the Status Bar, which describes the action to be executed by the selected menu item or depressed toolbar button, and keyboard latch state. A check mark appears next to the menu item when the Status Bar is displayed.

See Status Bar for help on using the status bar.

Status Bar

{bml hlpsbar.bmp}

The status bar is displayed at the bottom of the TSMP-1 window. To display or hide the status bar, use the Status Bar command in the View menu.

The left area of the status bar describes actions of menu items as you use the arrow keys to navigate through menus. This area similarly shows messages that describe the actions of toolbar buttons as you depress them, before releasing them. If after viewing the description of the toolbar button command you wish not to execute the command, then release the mouse button while the pointer is off the toolbar button.

The right areas of the status bar indicate which of the following keys are latched down:

Indicator	Description
CAP	The Caps Lock key is latched down.
NUM	The Num Lock key is latched down.
SCRL	The Scroll Lock key is latched down.

Cascade command (Window menu*)*

Use this command to arrange multiple open windows in an overlapped fashion.

Tile command (Window menu)

Use this command to arrange multiple open windows in a non-overlapped fashion.

Index command (Help menu)
Use this command to display the opening screen of Help. From the opening screen, you can jump to step-by-step instructions for using the TSMP-1 and various types of reference information.

Once you open Help, you can click the Contents button whenever you want to return to the opening screen.

Using Help command (Help menu)
Use this command for instructions about using Help.

About command (Help menu)
Use this command to display the copyright notice and version number of your copy of the TSMP-1.

Context Help command
{bml curhelp.bmp}

Use the Context Help command to obtain help on some portion of the TSMP-1. When you choose the Toolbar's Context Help button, the mouse pointer will change to an arrow and question mark. Then click somewhere in the TSMP-1 window, such as another Toolbar button. The Help topic will be shown for the item you clicked.

Shortcut
Keys: SHIFT+F1

Title Bar

The title bar is located along the top of a window. It contains the name of the application and document.

To move the window, drag the title bar.
Note: You can also move dialog boxes by dragging their title bars.

A title bar may contain the following elements:
{bmc bullet.bmp}	Application Control-menu button
{bmc bullet.bmp}	Document Control-menu button
{bmc bullet.bmp}	Maximize button
{bmc bullet.bmp}	Minimize button
{bmc bullet.bmp}	Name of the application
{bmc bullet.bmp}	Name of the document
{bmc bullet.bmp}	Restore button

Scroll bars

Displayed at the right and bottom edges of the test window. The scroll boxes inside the scroll bars indicate your vertical and horizontal location in the document. You can use the mouse to scroll to other parts of the document.

Size command (System menu)

Use this command to display a four-headed arrow so you can size the active window with the arrow keys.

After the pointer changes to the four-headed arrow:
1. Press one of the DIRECTION keys (left, right, up, or down arrow key) to move the pointer to the border you want to move.
2.	Press a DIRECTION key to move the border.
3.	Press ENTER when the window is the size you want.

Note: This command is unavailable if you maximize the window.

Shortcut
Mouse:	Drag the size bars at the corners or edges of the window.

Move command (Control menu)

Use this command to display a four-headed arrow so you can move the active window or dialog box with the arrow keys.

Note: This command is unavailable if you maximize the window.

Shortcut
Keys: CTRL+F7

Minimize command (application Control menu)

Use this command to reduce the TSMP-1 window to an icon.

Shortcut
Mouse: Click the minimize icon on the title bar.
Keys: ALT+F9

Maximize command (System menu)

Use this command to enlarge the active window to fill the available space.

Shortcut
Mouse: Click the Maximize icon on the title bar; or double-click the title bar.
Keys: CTRL+F10 enlarges a document window.

Next Window command (Control menu)

Use this command to switch to the next open test window. The TSMP-1 determines which window is next according to the order in which you opened the windows.

Shortcut
Keys: CTRL+F6

Previous Window command (Control menu)

Use this command to switch to the previous open test window. The TSMP-1 determines which window is previous according to the order in which you opened the windows.

Shortcut
Keys: SHIFT+CTRL+F6

Close command (Control menus)

Use this command to close the active window or dialog box.

Double-clicking a Control-menu box is the same as choosing the Close command.
{bml appexit.bmp}

Note: If you have multiple windows open for a single document, the Close command on the document Control menu closes only one window at a time. You can close all windows at once with the Close command on the File menu.

Shortcuts
Keys: CTRL+F4 closes a document window
ALT+F4 closes a window or a dialog box

Restore command (Control menu)

Use this command to return the active window to its size and position before you chose the Maximize or Minimize command.

Switch to command (application Control menu)

Use this command to display a list of all open applications. Use this Task List to switch to or close an application on the list.

Shortcut
Keys: CTRL+ESC

Dialog Box Options

When you choose the Switch To command, you will be presented with a dialog box with the following options:

Task List
Select the application you want to switch to or close.

Switch To
Makes the selected application active.

End Task
Closes the selected application.

Cancel
Closes the Task List box.

Cascade
Arranges open applications so they overlap and you can see each title bar. This option does not affect applications reduced to icons.

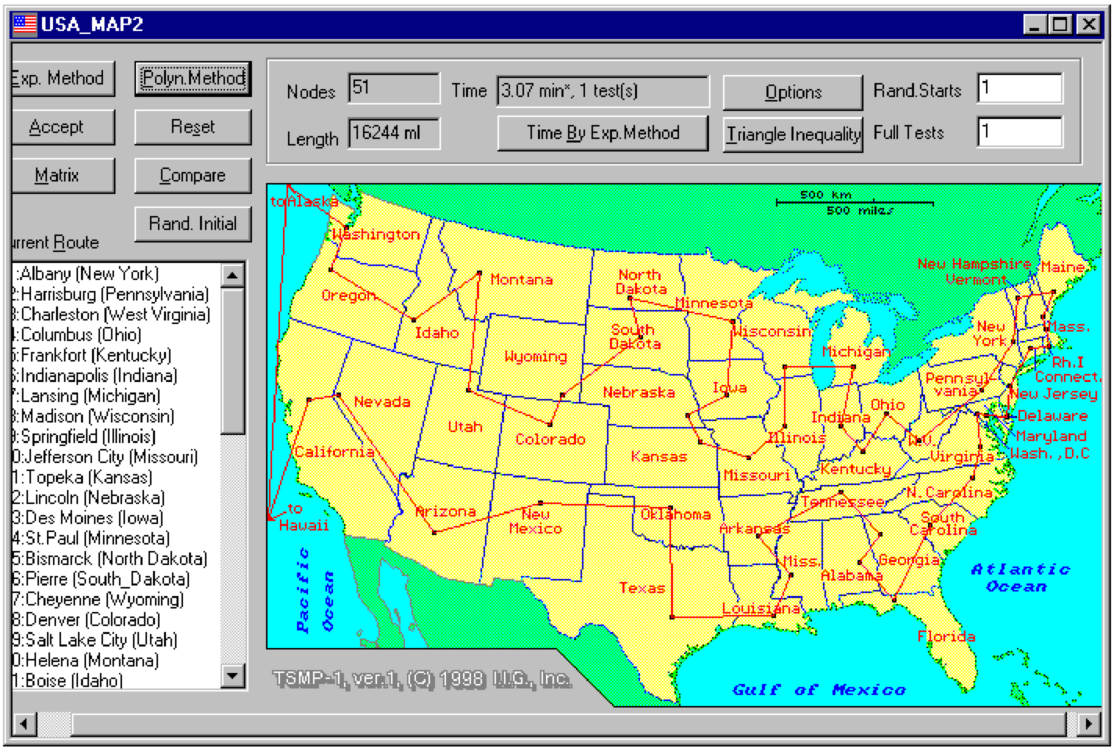
Tile
Arranges open applications into windows that do not overlap. This option does not affect applications reduced to icons.

Arrange Icons
Arranges the icons of all minimized applications across the bottom of the screen.

USA Demo command
It can be started from the File Menu, or by clicking the USA flag icon on the toolbar.

USA map

Displays a USA map on which all state capital cities and Washington, DC are shown. Some or all state capital cities (depending on a selection, see the Current Route listbox description below) are linked to each other by current route lines.



The Current Route listbox
This listbox contains a list of USA capital cities to select from. To accept a current selection, choose the Accept button. To return to the full set, choose Reset button.

Buttons:

Exp. Method
Polyn. Method
Starts searching for a minimum (or maximum, see the Maximum Route option description) route through a current set of USA capital cities by the exp. or polyn. method (see General Information/Preface).

Accept
Creates an initial route by selecting items from the capital cities list given in the Current Route listbox. The minimum number of the cities to make up the route is 4, the maximum one is all 51 items from the listbox (see Reset below).

Reset
Forms a full route that includes all 50 capital cities and Washington, DC. By using this command, the alphabetical list of the cities is formed.

Compare
Subsequently searches for a minimum/maximum route. The search is executed by both methods with the following comparison of the results found by the polyn. method with the ones found by the exp. method.

Matrix
Displays the window containing the distance matrix text (the distances are between all the cities of a current set). Distances are given in miles or kilometers according to a current selected condition of the Use Kilometers option in the Option dialog box.

The By Exp. Method button and Time field
Estimates an approximate time of the search for a minimum or maximum route by the exp. method. The time is displayed in the field Time. After either method has completed its work, the real search time is displayed in the field Time. The real time is marked with symbol *.

Triangle Rule
This command opens a dialog box with the Triangle Rule Violations list box and two buttons: Start Testing and Cancel. Clicking the first button starts searching for all violations of the triangle rule for all city triads of the current route.

Options
This command opens a dialog box with 3 checkbuttons:

Show Distances
Toggles whether to show or not the route distances in miles or kilometers (see the Use Kilometers option below).

Use Kilometers
Changes the between-cities distance units. Default units are miles. Checking this option toggles current units to kilometers.

Maximum Cycle
Specifies the route you want to search for: the minimum (default) or the maximum one (by checking this checkbox).

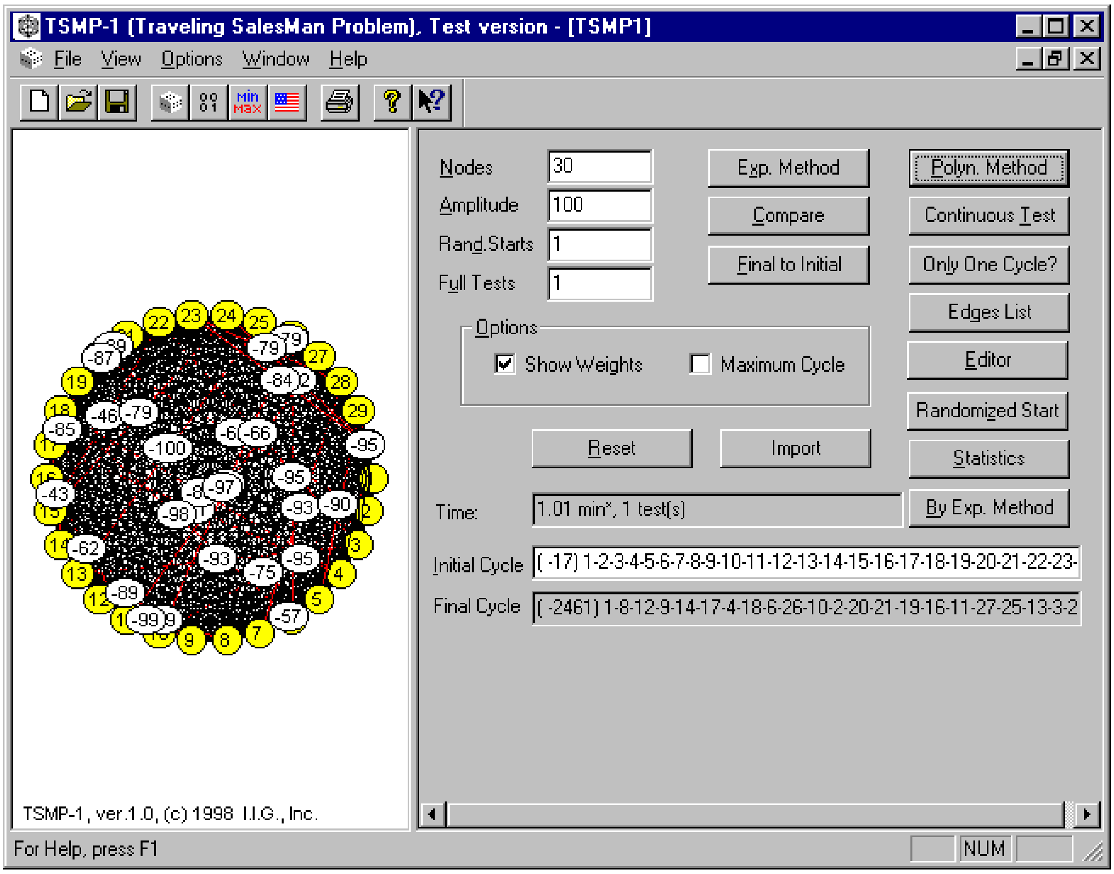
The Nodes field.
The number of capital cities for the current route is indicated in this field.

The Route Length field.
The length of the current route in miles or kilometers (see the Use Kilometers option) is indicated in this field.

Random Test window

Nodes

You can type here the number of the network's nodes you want to set. The minimum number is 4, the maximum one is 512 in the research version.



Amplitude/Max.Degree
If the Random Matrix mode is chosen (see below) and Cycle Existence option is OFF, you can type here the amplitude you want for the network's edge weights range. The valid range for the value is 1:32767. The actual edge weights values are shifted by a half of the amplitude towards the area of

negative numbers. For example, if the amplitude is given to be equal to 100, the edge weights values vary from –50 to +50.
If the Cycle Existence option is ON, this field shows the DEGREE OF NODE, which is the maximum number of a graph's edges incident to a given node. The graph's edges are selected at random.
If the From Editor Window mode is on, this field is in the read-only state.

Buttons:

Exp. Method
Polyn. Method
Searches for a minimum (or maximum, see the Maximum Cycle option) cycle in a current network by the exponential or polynomial method (see Preface).

Compare
Compares the final results obtained by the Polyn. And Exp. Methods for a current network. If the network has already been processed by one of the methods, then the Compare command runs the other.
If both methods have been applied to the current network, the program displays a dialog box with the appropriate information.

Continuous Test
Continuous testing of the Polyn. Method results by the Exp. Method until the user aborts the process (by pressing the Esc key). The process can encompass as many as about 2 billions testings (the maximum integer value on the Win32 platform). This command is not available if the From Editor Window mode is ON, i.e. when the matrix is manually entered.

Reset
Changes the matrix at random.

Editor
See below the From Editor Window option description.

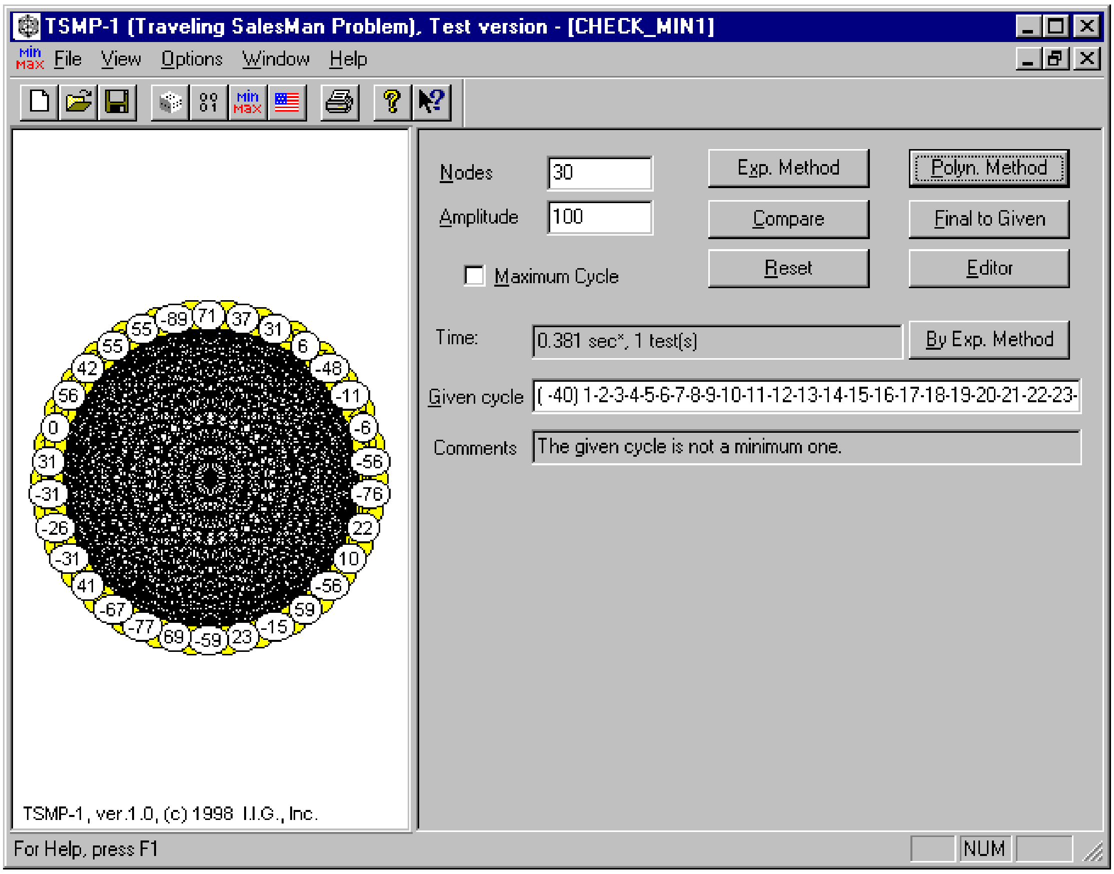
Randomized Start
Forms an initial cycle at random. The default initial cycle is always that in the form of 1–2–3–4–...–N–1, where N is the number of the network's nodes.

The By Exp. Method button and Time field
Estimates an approximate time of the search for a minimum or maximum cycle by the exp. method. The time is displayed in the Time field. When either method completes its work, the actual search time is displayed in the Time field. The actual time is marked with symbol *.

When the Continuous Test process runs, this field shows the time passed since the first tests started and the number of the tests. The values displayed are renewed after each 10 tests.

Edges List

Shows the edges, the weights of which can be increased (and the value of increase) without changing the status of the cycle (its minimality). You will see a list of the found edges along with the corresponding weight changes in a dialog box displayed.



Only One Cycle

Finds one more minimum cycle in addition to that already found if there exist more than one minimum cycle. The passage of only one extra cycle is displayed irrespective of how many more minimum cycle exist.

Options Window

You can choose one from the three initial network matrix modes available:

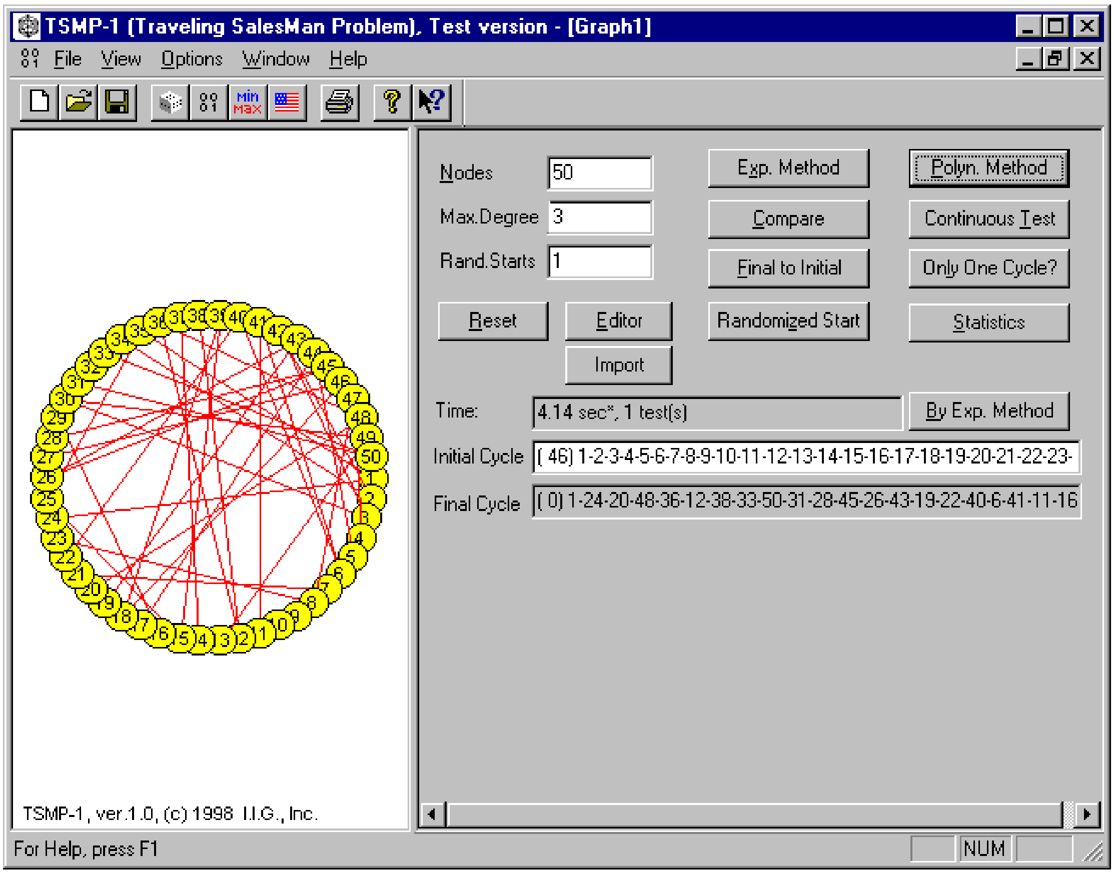
1. Random Matrix

The formation of the initial network matrix is fulfilled by a generator of pseudorandom numbers. The network's weights range can be changed in the Amplitude field if Zero Cycle Existence option is OFF.

2. From Editor Window (and the Editor button).

The matrix (i.e. all the network's weights) is entered manually. To enter the matrix, you should gain access to the text editor dialog box by clicking the

Editor button. This command loads the current matrix (that you can change) into the editor window. Note: if, while clicking the Editor button, the From Editor Window option is not checked, matrix editing is disabled (i.e. the editor is in the read-only state). The matrix weight changes are not approved unless you exit the editor by pressing O.K. The program makes sure that the weights values lie in the limit range of –32768:32767. In case that a weight value is beyond the range, it is equated to an appropriate limit without informing the user.



3. Maximum Cycle

Specifies the cycle you want to search for: the minimum (default) or maximum (by checking this checkbox) one.

4. Show Weights

Toggles whether to show or not the edge weights of a current cycle of the graph.

5. Cycle Existence

Determines a minimum cycle in the graph that is given by the maximum degree of its nodes.
The graph's edges are weighed as follows:
zero weight is assigned to the existing edges;
the weight of 1 is assigned to the edges complemented to build up a strongly-connected graph.
If a cycle of zero length is found, in the graph there is a Hamiltonian cycle.

Computations

TSPLIB Instances

The TSPLIB is Gerhard Reinelt's library of some 110 instances of the traveling salesman problem. Some of these instances arise from the task of drilling holes in printed circuit boards; others have been constructed artificially. (A popular way of constructing a TSP instance is to choose a set of actual cities and to define the cost of travel from X to Y as the distance between X and Y.) None of them (with a single exception) is contrived to be hard and none of them is contrived to be easy; their sizes range from 17 to 85,900 cities. TSMP-1 solver has solved each instance up to size 299 because of the problem with memory of the Win32 platform; some of the remaining instances are still open problems.

TSMP-1 Benchmarks

We report below the running times for the TSMP-1 solver on TSPLIB instances. The tests were carried out on a 200 MHz Intel® Pentium Pro™ workstation under Windows NT™ ver. 4.0 and with 64 MB of RAM.

Name	Attempts in Search Procedure	Running Time (seconds)
burma14	1	6
ulysses16	1	22
gr17	1	8
gr21	1	3
ulysses22	1	53
gr24	1	7
fri26	1	7
bayg29	1	9
bays29	1	13
dantzig42	1	23
swiss42	1	13
att48	1	56

gr48	1	67
hk48	1	17
eil51	1	73
berlin52	1	29
brazil58	1	68
st70	1	50
eil76	1	30
pr76	1	186
gr96	1	67
rat99	1	95
kroA100	1	100
kroB100	1	236
kroC100	1	96
kroD100	1	100
kroE100	1	244
rd100	1	67
eil101	1	74
lin105	1	59
pr107	1	103
gr120	1	223
pr124	1	364
bier127	1	165
ch130	1	213
pr136	1	397
gr137	1	342

pr144	1	258
ch150	1	303
kroA150	1	500
kroB150	1	423
pr152	1	793
u159	1	100
si175	3	1309
brg180	1	146
rat195	5	2223
d198	3	1182
kroA200	1	659
kroB200	1	391
gr202	1	501
ts225	1	2052
tsp225	1	1501
pr226	1	435
gr229	3	3861
gil262	1	1306
pr264	1	267
a280	3	537
pr299	3	1749

References & Traveling Salesman Problem Links

Publications

1. Ivan I. Goray, "On polynomial solvability of the Hamiltonian cycle problem for graphs of degree less than or equal to 3," arXiv: 1007.0235v1 [math.OC], 1 Jul, 2010, 27 p.).
2. US Patent Application № 09/006,367.

Links

- TSPLIB - Gerd Reinelt's library of TSP instances.
- TSPBIB - Pablo Moscato's listing of TSP references.
- I.I.G., Inc. Home Page (www.pcgrate.com).