# The Optimized Odyssey

by Martin Grötschel and Manfred Padberg

*The problem of the shortest roundtrip is the prototype of a large class of practically important complex minimization or maximization tasks. They are so important that usually one has to content oneself with approximate solutions to them. However, new performing methods have been developed and implemented to find provably optimal solutions for large problems with far greater frequency than before.*

Ulysses, the legendary king of the Greek island of Ithaca, had made a long and eventful journey, when after 20 years of erring around he could finally take his beloved Penelope into his arms again --a journey eventful enough to fill an entire epos, the Odyssey by Homer, the poet of antiquity. Modern interpreters of the epos have determined 16 Mediterranean places which Ulysses has supposedly visited. Undoubtedly, he could have had a shorter journey if destiny, the sea's hostility, the gods' envy and, most importantly, the lack of elementary geography knowledge had not been against him. Figure a Ulysses of modern times: imagine a top-manager or -- more prosaically -- a sales representative (traveling salesman). His objective is to pay a visit to all 16 cities in arbitrary order but only once: time is money. This is the famous Traveling Salesman Problem (TSP): fast to formulate, very important in practice and seemingly easy to solve, but in reality, when the number of cities is large, so difficult to handle that mathematicians still break their teeth on it after decades of hard work. And more: the TSP is just the most popular example of a large class of highly relevant practical problems. These are those problems where among many possible combinations one looks for a best or most favorable combination. Some examples of these numerous practical applications are to be found in: Cargo, truck and air freight loading, Machine scheduling, Crew scheduling in transportation, Microchip design, Capital investment, Layout of communications networks. For Ulysses' problem the shortest roundtrip is 6859 kilometers long. In this calculation the distance between any two of the 16 places is their aerial distance (on a great circle around the globe) and rounded up on kilometer integral value; but even with an

Martin Grötschel

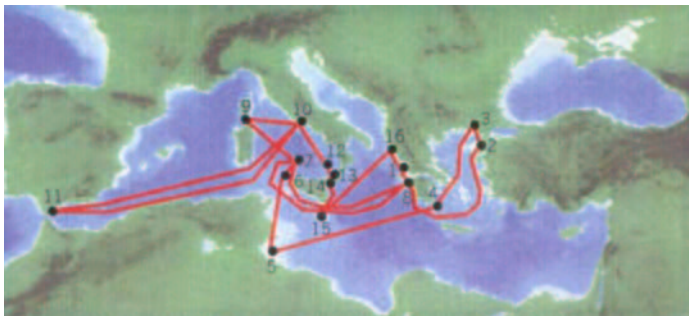
Manfred W. Padberg

airplane, Ulysses would have had to travel 9913 kilometers along his own route. Optimization can bring about enormous savings! How did we find the shortest tour for Ulysses? To find some roundtrip is child's play: there are exactly 653837184000 possible tours. Using a commercial workstation it took 92 hours (almost four days, wow!) to generate all tours, to calculate their lengths, considering them one by one, and then to pick the shortest one. This technique is called enumeration: it is the classical approach to solve combinatorial optimization problems (problems which have finitely many choices). Enumeration is simply the thoughtless application of brute force. But this method (a little old-fashioned) is still alive in some (dusty) heads. Only it fails for problems which are insignificantly bigger than that of Ulysses. This is why, among other reasons, we have chosen Ulysses' problem as an illustration: it can be solved by doing a complete enumeration on currently available computers. Already a TSP with 25 cities would hopelessly overwhelm the most polished enumeration technique even when run on the biggest supercomputer around now or in the future. However, practical problems are much bigger and in spite of their difficulty they have to be and have been solved. In the summer of 1998, David Applegate, Robert Bixby and William Cook of Rice University in Houston (Texas) and Vašek Chvátal of Rutgers University in New Brunswick (New Jersey) have found the shortest tour through more than 13,000 cities with the methods described here, leaving their previous record of 7,397 cities far behind. Problems of this size and far bigger ones arise daily in the production of circuit boards, in the design of very large integrated circuits (VLSI design), in X-ray crystallography and in many other areas. The analysis begins with a mathematical formulation. One replaces every city by a point, called a node, and every direct connection between two cities by a line (mathematicians say: edge) between the corresponding nodes of the TSP. A collection of nodes connected by edges is called a graph. When, like in the case of Ulysses' problem, every node is connected to every other node by some edge --we can fly from every city directly to every other city--, then experts call the graph complete. The complete graph with 16 nodes has exactly 120 edges.

## Mathematical modeling: a roundtrip becomes a graph

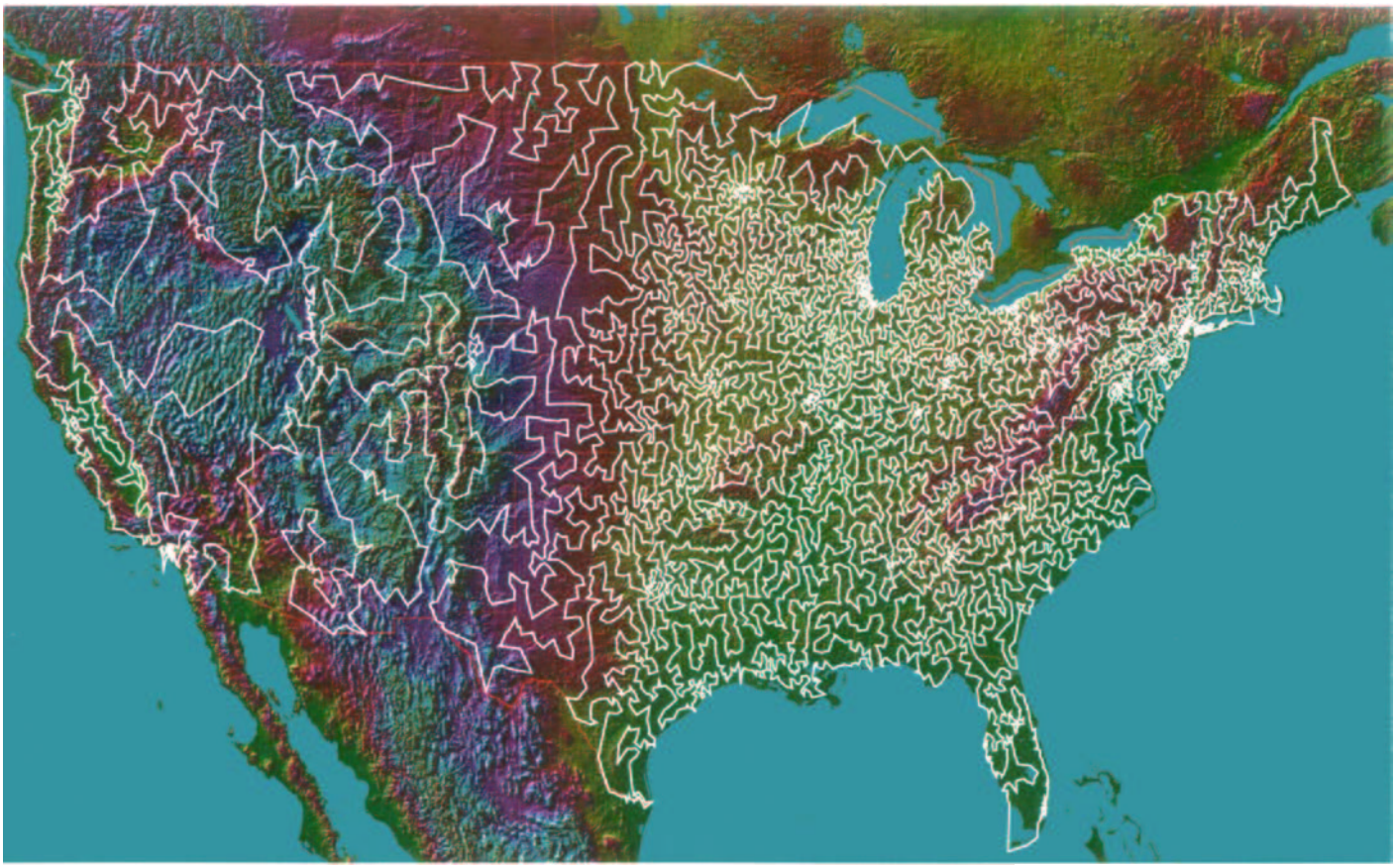
With every edge of the graph we associate a distance or "length": more generally, a number which represents the "cost" (the time, money, material or whatever else is dear and expensive to you) caused by traversing the edge. Every roundtrip corresponds to a subset of the edges of the graph. If such a subset forms a complete roundtrip passing through all nodes exactly once, it is called a tour or a Hamiltonian cycle, after the Irish mathematician Sir William Rowan Hamilton (1805 -- 1865). The length of a tour is the sum of the lengths of all edges in the tour. The TSP, in graph theoretical notation, is thus the problem of finding "the" shortest cycle in a complete graph, more precisely "a" shortest cycle, because a TSP can have different tours with the same minimum length. So why is the TSP so difficult? Does it lie in the fact

that the number of tours is so tremendously large for large number of cities? Amazingly not. The number of possible solutions does not measure the difficulty of a combinatorial problem adequately. To give you a reason of why this is true, let us look at the following problem. We want to set up a communication network on the 16 cities that connects every city to every other city, not necessarily directly, but possibly via a detour going through some other city or cities. Again the cost of a direct link (running on the edge) between any two cities depends on their distance, e.g., the cost for an optical fiber cable between the two. The most expensive solution to our problem is easy: just connect every city directly to every other one, i.e., choose the complete graph as a solution. There is a very simple method to reduce the cost of this solution: we remove from the graph the most expensive edge that can be removed under the condition that the resulting new graph does not decompose into two disconnected parts. Then we repeat the same procedure as long as it is possible. This method produces for Ulysses' graph with 16 nodes a network of total length of 4540 kilometers. This algorithm chooses at every step the most favorable among all available possibilities without regard to its consequences. Eliminating the edge with the biggest cost reduction does not check if a much better solution could be found. Such a strategy – take what you can get without regard to the delayed effects – is called greedy (or myopic). Usually it is meaningful to accept --temporarily-- a disadvantage in the search for the greatest advantage. But in this case pure greed is exactly right! One can prove that this algorithm is always optimal, i.e., it always finds a minimum cost solution of the communication problem. How many possible solutions are there for this new problem? With $n$ cities every possible solution corresponds to a "unique" graph with exactly $n-1$ edges that link all nodes and that has no cycle. Such graphs are called spanning trees or scaffoldings. Already in 1889, Arthur Cayley has proven that there are exactly $n^{n-2}$ different spanning trees for a graph with $n$ nodes. For 16 cities (nodes) there are exactly 72,057,594,037,927,936 possible spanning trees: this is considerably more than the number of the possible tours in Ulysses' problem and this statement applies generally to an arbitrary (large) number of cities. Nevertheless any PC can calculate a minimum cost spanning tree within a few milliseconds even when $n$ is large. If it is not the number of possible



Martin Grötschel and Manfred Padberg.

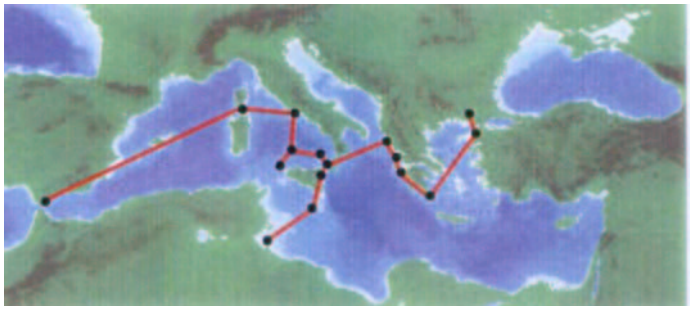
Figure 1 – The legendary Ulysses was almost hopelessly lost on the two-dimensional sea surface and he didn't choose the shortest way home (picture, after the interpretation of E. Bradley). To not just get back home, but to find the shortest roundtrip through all 16 spots, you have found your way around in a 120-dimensional space.



Ray Sterner et al, John Hopkins University, Applied Physics Lab.

Figure 2 – World record: the shortest tour through all 13,509 cities of the USA (without Alaska and Hawaii) with more than 500 inhabitants.

solutions, what is it that makes the TSP such a difficult problem? To be honest, we do not know the answer. In any case, there is a complexity theory which helps to distinguish "easy" problems (those like the minimum cost spanning tree problem) from "difficult" ones like the TSP. But this theory also gives an unpleasant result: many practically relevant combinatorial problems are classified by it as "difficult" ones. In addition, an insignificant complication can change an easy problem into a difficult one. Assume we complicate the matter by requiring a certain "failure safety" for our communication network. For instance, it should work even if a fire, an earthquake or an excavator mistakenly destroys a line. A spanning tree just doesn't offer such safety: as the result of an interruption of any link of it the net "decomposes" into two parts, which can no longer talk to each other. Such obvious safety aspects have not always been taken into account in reality. When in the American regional telephone nets the copper wires were replaced by optical fiber cables, a minimum spanning tree solution was chosen because it gave a minimum cost solution of the high cost project. But afterwards disastrous failures happened: a single edge cut led to the complete collapse of communication networks, just like in 1988 in the Chicago telephone network (and later also in New Jersey and elsewhere) which was highly publicized by the press. Extremely costly improvements of the network were necessary as a result of insufficient planning. The more additional lines and connections there are, the safer the communications network. The complete graph is the best answer to this problem given the safety aspect, but it is also much too expensive. Thus one has to weigh failure safety against construction cost and maintenance. In a real network there are typically vital and less vital connections or cities with priorities which can be quantified and give rise to additional requirements. Such additional conditions can be brought with little effort into the mathematical formulation: this is the beauty of this approach to combinatorial problem solving. Unfortunately, the incorporation of such additional considerations changes our "easy" problem into a difficult one, one that is as difficult as the TSP. A possible (and usual) problem reformulation goes as follows. We allocate weights to each city: 0 for the insignificant, 1 for the more important, 2 to the large cities and so on. Then we demand that a city of weight $k$ shall remain connected to every city having the same or bigger weight even if any $k$ links of the network are cut. Now you can search for a network of lowest costs among all networks satisfying this additional condition. In a similar way safety demands in the case of failure of nodes (and/or nodes and edges) can be taken into account. Problems in practice frequently require many other, additional conditions of this type. For instance, a line or an internetworking processor can contemporarily cope with a certain number of conversations only. Because of these conditions the number of feasible solutions becomes smaller, but --probably contrary to our intuition-- the problem becomes even more difficult. So much about problems. How can we find good or really optimal solutions? And how can we judge the "goodness" of a solution that a computer spits out? Given all differences among different combinatorial optimization problems --even important ones-- there are nevertheless some basic principles that apply across the board.



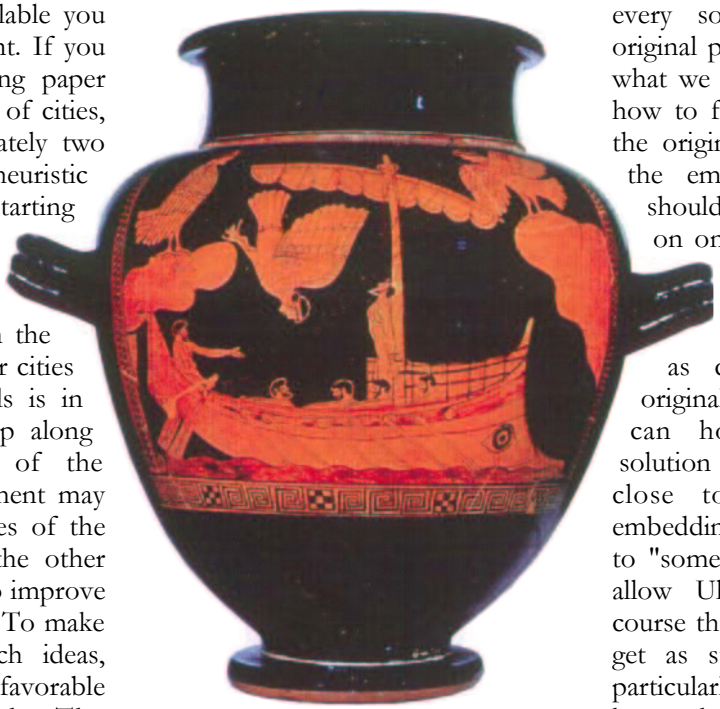
Martin Grötschel and Manfred Padberg.

Figure 3 – The most economical communications network between Ulysses' 16 cities.

## Upper and lower bounds

First of all let us consider a certain objective function --tour length, costs and so on-- that we wish to minimize. (The maximization problems differ only by the signs of the objective function coefficients.) We can now compute for every feasible solution its cost which we can mark in principle on the real line. It is evident that an optimal solution always exists: we get finitely many real numbers and thus the smallest number among all those corresponds to an optimal solution. We only do not know where on the real line that smallest number lies. So we look for a range into which this unknown number falls: we try to find upper and lower bounds for the minimal cost. Without knowing the optimal solution, it will then be possible to say that the objective function should be better than the lower and should not be worse than the upper bound. To find upper bounds is in general not difficult. Every feasible solution gives an upper bound: by definition the optimal solution can not be worse than an arbitrary feasible solution. Of course the solutions that are "close" to the optimum are of greater interest because they give better upper bounds. Algorithms to find "good" feasible solutions are called heuristics in technical jargon. The knowledge that no solution can be better than a certain value, i.e., a lower bound, is also helpful. When the upper and the lower bounds coincide, then we have proven the optimality of the solution corresponding to the upper bound. If this is not the case, then the knowledge of the upper and lower bound values still allows us to estimate how far the solution is from the optimum. If the deviation is small, then we may want to stop the search because further searching, if successful, would not improve the solution by much. It is not evident how to find lower bounds and this will require a suitable mathematical formulation. But it is doable and it is particularly skillful to alternate the search for upper and lower bounds in a way that uses the information of the previous step in the next one. This combined strategy is responsible for the enormous progress in

combinatorial problem solving that has been made in recent years. We will discuss now the techniques to find upper and lower bounds for the case of Ulysses' problem. For other difficult problems the story sounds mostly only slightly different; but, of course, because of such little technical details the problem can sometimes become substantially more difficult again. For the search of upper bounds, greed --more exactly: myopic search for a cost minimum-- gives a first way of doing things. Choose as the next city the nearest city that you have not yet visited (nearest-neighbor heuristic). If no city is available you go back to the starting point. If you play with this method using paper and pencil on arbitrary sets of cities, then you will find immediately two things. First, the result of a heuristic strongly depends on the starting point. Second, the solution can in general be easily improved. If, for instance, two edges intersect, then in the rectangle formed by the four cities the trip along the diagonals is in general longer than the trip along the two opposite sides of the rectangle. Further improvement may be possible if the two edges of the rectangle are replaced by the other two. There are other ways to improve a nearest-neighbor solution. To make useable methods from such ideas, one has to limit the unfavorable consequences of greed. The insistence on always finding a better solution can get you stuck in a bad "local optimum", which is a solution that is not particularly good but which cannot be improved upon by any of the improvements steps you utilized. To get around this problem, you may e.g. permit --with a certain probability, up to a certain magnitude or under other restrictions-- that in the current "improvement step" the solution actually gets worse. Another possibility is to randomly select, from time to time, a completely new heuristic solution for subsequent improvement. Such methods go by most entertaining names such as Monte-Carlo methods, simulated annealing, genetic or deluge algorithms (see Spektrum der Wissenschaft, July 1987 page 104, and March 1993 page 42). As a rule these heuristic methods require a relatively small programming effort for medium-size problems. For this reason they have become a popular playground for amateurs, which -- unfortunately-- has led to totally incorrect expectations as to their quality and implementability. To apply these techniques to TSPs with tenthousand or hundredthousand cities (these orders of magnitude are not exceptional), complicated data structures and clever tricks from computer science are indispensable if you want to get reasonable solutions in acceptable time. On the other hand, heuristic algorithms that have been adapted to a particular problem structure through long-term experimentation have become --due to present technological limits-- the workhorse for the solution of practical problems and they are also helpful for exact optimization methods in several ways. How do you find valid lower bounds? The -- seemingly paradoxical-- basic idea is to make the problem bigger so as to make it simpler! This means that -- given our problem-- we construct a bigger problem, which has more solutions: every solution to the old problem is also a solution to the new one (but not necessarily the other way round). For example, you could drop or inactivate side conditions or constraints of the original problem. Then the solution set of the original problem is contained ("embedded") in the solution set of the new problem. The experts speak about embedding techniques. The cost of a minimum solution to the bigger problem is by definition lower than (or at best equal to) the cost of every other solution to it, hence also lower than the cost of every solution to the embedded original problem. And that is exactly what we were looking for: we know how to find valid lower bounds for the original problem. To be useful, the embedding (bigger) problem should have two characteristics: on one hand it should be simple so that its minimum solution is easy to find, on the other hand it should be as close as possible to the original problem. Only then we can hope that the minimum solution of the original problem is close to the solution of the embedding problem. It is no big deal to "somehow" embed the TSP. Just allow Ulysses to stay home. Of course then the lower bound that we get as starting information is not particularly effective: every tour is longer than zero kilometers.



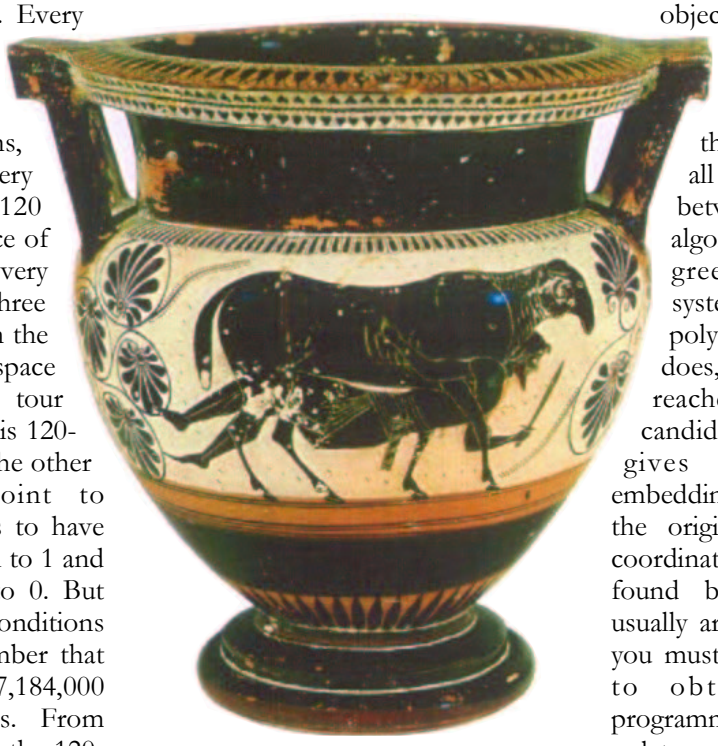
Bildarchiv Preußischer Kulturbesitz, Berlin

Figure 4 - Short-term profitable alternatives are incapable to produce long-term benefits. So are Greedy algorithm advantages. Beware this fault and avoid the sirens songs enchanting that fascinated Odysseus of his companions.
*Attic jug painting, 450 B.C.*

## Lower bounds from linear programming

From among the various embedding techniques for the TSP we will discuss here only the most successful one. It is based on linear programming, one of the most remarkable methods that applied mathematics has produced since the Second World War Two (see "Wirtschaftsfactor lineare Programmierung " by Robert G. Bland, Spektrum der Wissenschaft , August 1981 page 118). The -- traditional-- name is misleading: "programming" here just does not refer to the act of programming a computer. A better name would be "optimization of linear problems" or "linear optimization". It is most frequently interpreted as the art to

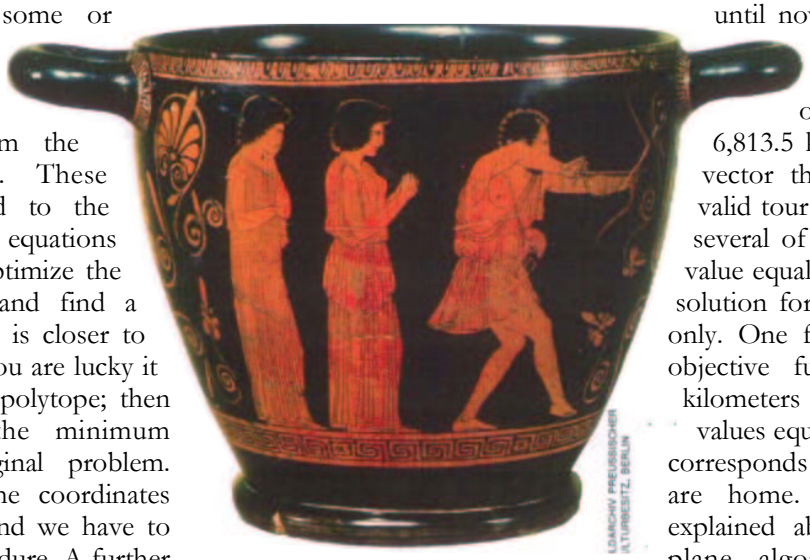
assign scarce resources in an optimal manner to economic activities. Today linear programming is a standard tool for the solution of planning problems in airline companies, oil refineries, banks, automobile industry, telecommunication and in many other industries. To embed Ulysses' problem into a linear programming problem we start by indexing --in some fixed, but arbitrary order-- the 120 edges connecting the 16 cities and write next to a edge the number one if Ulysses uses the edge on his roundtrip, a zero otherwise. Every tour is thus described by 120 numbers. We call this sequence of numbers a vector of 120 dimensions, because we can represent every ordered collection of 120 numbers as a point in a space of 120 dimensions; just like every sequence (x, y, z) of three numbers describes a point in the ordinary three-dimensional space in which we live. Every tour corresponds to a point in this 120-dimensional space, but not the other way around! For a point to correspond to a tour it has to have exactly 16 components equal to 1 and the other 104 ones equal to 0. But that's not enough: further conditions have to be satisfied. Remember that there are exactly 653,837,184,000 possible tours for Ulysses. From among these many points in the 120-dimensional space we have to select one such that the sum of the kilometers of the components having value one is as small as possible. We thus multiply every component of our vector (1 or 0) by the corresponding kilometer value, sum the products and obtain the tour's length. Now comes the decisive step. We just simply interpret the vector components as variables, which may assume arbitrary values between 0 and 1. This may seem rather silly considering the original problem: what does it mean that Ulysses takes half a flight from Ithaca to Troy or 0.17 of a flight from Messina to Gibraltar? However it is meaningful for our purposes because --while we enlarge the set of the feasible solutions tremendously-- we allow the problem to be represented by linear programming. The objective function --the total tour length-- is a linear function of these 120 variables. Every change in one of the variables affects the value of the objective function with a proportionality factor, namely the length of the corresponding edge or inter-city link in kilometers. The side conditions are also linear. To search for the minimum of a linear function within linear side conditions (equations or inequalities) we have an efficient method, the Simplex algorithm. (The interior-point methods, intensively investigated for about ten years now, still do not bring about in any advantage for our purposes). A little geometry is helpful even if our powers of imagination are limited to only three instead of 120 dimensions. If we take all points (vectors) with coordinates between 0 and 1, we form a cube with edge length 1 having a corner in the point zero (the origin of the coordinate system). The corners of this unit cube are the vectors whose components are 0 or 1. This is valid also in 120 dimensions, with the only difference being that now the unit cube has $2^{120}$ corners, which to enumerate and inspect one by one would be stupid. A linear inequality constraint acts like a knife that cuts off a piece of the cube (just think of a piece of cheese). You get some form of a body that is bordered by some planes. In the 120-dimensional space the set of the feasible points (those that satisfy the linear side conditions) generate a more general body of the same shape: it is called a (convex) polytope. The minimum of a linear objective function on a polytope is always assumed at some corner (or if there are several minimal corners the minimum is attained on all of them and everything between them). The Simplex algorithm (following also some greedy principle) searches systematically the corners of polytopes until it stops. When it does, the minimum corner is reached (or one of several candidates). Unfortunately that gives the minimum of the embedding problem only, but not of the original problem. In fact the coordinates of the solution vector found by the Simplex algorithm usually are not all integer. Therefore you must use again the cheese knife to obtain from the linear programming polytope the "true" polytope of the original problem. This is the smallest polytope that contains all zero-one vectors corresponding to the tours. However one single cut will in general not suffice. Even worse: we do not even know exactly how to cut. Explicit linear programs for the TSP are known for problems with only up to 9 cities. More precisely: the polytope associated with the TSP on 9 cities has 9 equations and 42,104,442 inequalities in 36 variables according to the possible 36 direct connections between the 9 nodes of the 9-city TSP. For TSPs with 8 (7 and 6) cities the polytope has 8 (7 and 6) equations and 194,187 (3,437 or 100) inequalities in 28 (21 or 15) variables. For 10 cities there are probably 51,043,900,866 inequalities, certainly not less, but the proof for the exact number is still due. For tour problems



Bildarchiv Preußischer Kulturbesitz, Berlin

Figure 5 - Polyphem and the Polytop: the searching algorithm has a blind man behavior. To quickly move on he has no general vision but short sight perception. For time reasons he doesn't analyze every side condition. Then he must accept violations. So Odysseus gets out clutched at the ram belly of the gluttonous Cyclop: an impossible mission!
*Sapphic painter portrayal on a vessel, 510 B.C.*

on graphs with more than 10 nodes, a complete set of inequalities for the TSP polytope is not yet known even theoretically. And even if it would be known, it would certainly be too big to be of any help. It is possible that, for big numbers of cities, it will never be found, and this would be an additional proof of the enormous difficulty of the traveling salesman problem. For the 16-city problem of Ulysses we have 16 equations in 120 variables and the number of inequalities is in any case astronomically big. We do not know all of them; however a partial subset of them is known. We can indeed almost always find some or several linear inequalities or cuts, which separate the minimum found from the true TSP polytope. These inequalities are added to the present system of the equations and inequalities, we optimize the new linear program and find a minimum point which is closer to the TSP polytope. If you are lucky it even lies on the true polytope; then we have reached the minimum solution of the original problem. Otherwise it has some coordinates that are not integer, and we have to repeat the whole procedure. A further complication arises. Generally, we cannot even start with polytope of the embedding linear optimization problem. To do so, we would have to start with its complete set of inequalities which for a large number of cities is also astronomically large! Nevertheless we can solve big problems using linear programming to optimality. How is this possible? Once again we replace the embedding problem by a bigger and simpler one. We use only a very small subset of the astronomically many side conditions (we cut the cheese cube only slightly), we find a solution vector that is an optimal corner of this big polytope -- which is too big-- and then we improve it. We then look for a side condition which cuts off not only the optimal corner just found, but also as big a slice of the cheese as possible. Back to Ulysses' problem: as an initial embedding we can first consider the 16 side conditions (equations) modeling the fact every city may participate in the roundtrip with exactly two edges: an edge going into the city and an edge going out of it. In terms of the edge variables this means that the sum over all edges meeting a city must equal 2. (Different from an inequality, an equation reduces the dimension of the polytope, and, unfortunately, that we can no longer illustrate on a cube of cheese.) The solution of the corresponding linear program gives a minimum objective function value of 6,113 kilometers. This is after all already a useable lower bound. Unfortunately the corresponding solution vector consists of several subtours. Since this



Bildarchiv Preußischer Kulturbesitz, Berlin

Figure 6 - Integer optimization achieved by arrow and bends. When after twenty year wandering Odysseus got in his court again, many unfulfilled side conditions still separate him from the his objectives, namely the Penelope suitors. By fixing one side condition after the other if he eliminates all inadmissible points and finally reaches the optimum.
*Red paint on Attic bowl*

is not a feasible solution to the original problem, you must add some inequalities that cut off this point from the current polytope, i.e., we have to improve the original embedding by adding some cuts. There is a lot of choice among a large number of such cuts. Among those are "best possible" cuts, that is those that have as much as possible in common with the facets of the true polytope. (In large dimensions, facets are the generalization of the boundary surfaces of a three-dimensional polytope). A large part of the theoretical work of the last 20 years has concentrated on identifying such facet cuts. In Ulysses' case we add the side conditions which exclude the four subtours of the solution just obtained. Now we have to solve the linear program augmented by these linear inequalities. Modern software packages take care of this quickly and efficiently because they make clever use of the information contained in the previous solution. The objective function value rises to 6,228 kilometers, but the corresponding solution vector does not give a suitable roundtrip for Ulysses. However the algorithmic idea now is clear: we go on this way, i.e., we iterate. We activate a further small subset of the astronomically large set of inequalities which we have ignored until now. Doing so and solving the third linear program we get an optimal objective function value of 6,813.5 kilometers and a solution vector that still does not give a valid tour for Ulysses. Namely in it several of the 120 variables have a value equal to 0.5 which is a feasible solution for the embedding problem only. One further iteration gives an objective function value of 6,859 kilometers and a solution with all values equal to 0 or 1. This solution corresponds to the optimal tour. We are home. The iterative method explained above is called a cutting plane algorithm. If the set of inequalities, which describe the true polytope, is complete, then the cutting plane algorithm terminates after a finite number of steps. In our case four steps sufficed, but that does of course not apply in general. At present we know, however, all of these inequalities neither for the traveling salesman problem nor for most other combinatorial optimization problems of practical importance. A large part of today's research work concentrates on augmenting this knowledge for more and more combinatorial problems. As a result of this work, the limits of mathematical computability have been driven far beyond levels that, only few years ago, scientists could not imagine overcoming. On the other hand it is possible that the difficulty of the TSP or of other combinatorial optimization problems excludes forever a complete knowledge of the required inequalities. We simply do not know this, and at present we

know for sure that we do not know all required inequalities. Differently from Ulysses' problem, the iterative cutting plane method can thus terminate before an optimal tour is found. What are our choices in such a case? We could stop and be content with the knowledge of a good lower bound. We also could use an enumeration method as follows: if the cutting plane algorithm ends undecidedly (without a feasible solution for our true problem), then there are edges, which are assigned values different from 0 and 1. We pick out such an edge, arbitrarily put the value 1 (meaning "we use this link") on it and try solve the more restricted and therefore simpler problem. We try the same with the edge specification of 0 instead of 1 (meaning "we do not use this link"). We take the better solution of the two possibilities. For one of the two possibilities --take that edge or do not take it-- applies to any optimal tour. Of course the solution attempt can end undecidedly for these subproblems once again. Again the solution process branches into two alternatives (branching) and we get a whole tree of solution possibilities. We must search through this tree as skillful as possible. The corresponding tree search method is known as Branch and Cut. This method builds on the method known as Branch and Bound of the 1960's, but it is fundamentally more efficient. At present the most successful algorithms for big combinatorial optimization problems are all based on Branch and Cut and so are all algorithms with the best quality guarantees. Various heuristic and enumerative techniques have been added. To implement a successful method you need thorough mathematical analysis, interdisciplinary knowledge in optimization and computer science, including knowledge of efficient data structures, data processing and computer programming. And only development and testing on real computers will prove the goodness of a method. The TSP illustrations for this article have been computed with a computer code that was developed by Manfred Padberg and Giovanni Rinaldi of the Italian research center IASI-CNR (Istituto di Analisi dei Sistemi ed Informatica) in Rome. There are other successful applications of Branch and Cut by us and other researchers, far more than we can possibly describe in this article. A Branch and Cut solver on a modern workstation takes only a few milliseconds to solve the various problems that we have discussed here for the 16 cities of Ulysses: a very short time compared with the 92 hours needed to solve the problem by mindless enumeration! Add to this the rapid performance-growth of computer hardware, solid applied mathematics and intelligent software development. Through all of that, the exact solution of far bigger problems than we can imagine today may very well become soon a reality. In conclusion, once again the question: why is the traveling salesman problem so difficult? For the solution method described here the biggest hurdle is undoubtedly the astronomically high number of inequalities, which we must take into account. Does this cause the difficulty of the problem? Surely it is part of the reason. Amazingly, however, this not a complete explanation either. There are problems in combinatorial optimization which can be solved easily both in theory and practice, the polytopes of which, however, necessitate far more inequalities (in the same dimensions) than the TSP polytope. If it is neither the number of possible solutions nor the number of the required inequalities, then what makes the problem "difficult"? At present nobody has an answer to this question, yet the algorithms discussed here yield excellent results nevertheless.

*"I'm sorry, but nine Greek islands in seven days is just too much."*



*Martin Grötschel*
*groetschel@zib.de*

*Manfred W. Padberg*
*manfred@padberg.com*

From **Spektrum der Wissenschaft** 4/1999 (pages 76 – 85).

**Acknowledgments**

The AIROnews Editors are deeply indebted with Dr. Christoph Pöppe for his support to obtain the permission to publish the English translation of the "Spektrum der Wissenschaft" paper. They also thank Ms. Alice Krüßmann for her fruitful information of illustrations' license references.

Authors:
**Martin Grötschel** and **Manfred Padberg** started their common work on combinatorial optimization problems in 1974 at the University of Bonn where Padberg was guest professor and Grötschel a PhD student. At present Grötschel is professor for mathematics at the technological University of Berlin and vice-president of the Konrad-Zuse center for information technology. Padberg has been a visiting professor at many prestigious universities in Europe and the United States of America and is a professor at New York University's Stern School of Business, but lives now in Marseille, France.

## References

· D.Applegate et al., "On the solution of traveling salesman problems", in: *Documenta Mathematica* Extra Volume, Proceedings of the ICM 98, 1998 (645-656); http://www.mathematikuni-bielefeld.deldocumental xvol-icm/ICM.html

· E.Bradford, *Ulysses Found,* Harcourt, Brace & World, New York, 1963.

· R.E.Burkard et al., "Well-solvable special cases of the travelling salesman problem: A survey", *SIAM Review*, 40, 1998 (496-546).

· M.Jünger, G.Reinelt and G.Rinaldi, "The travelling salesman problem", in: *Annotated bibliographies in combinatorial optimization*, M.Dell'Amico et al. (Eds.). Wiley, Chichester, 1997 (199-221).

· E.L.Lawler et al. (Eds.), *The traveling salesman problem. A guided tour of combinatorial optimization,* Wiley, Chichester, 1985.

· M.Padberg, *Linear Optimization and Extensions,* Springer, Berlin 1995.

· M.Stoer, "Design of Survivable Networks", *Lecture Notes in Mathematics*, 1531, Springer, Berlin 1992.