4

Computational Experience in Methods for Finding Tight Lower Bounds for the Sparse Travelling Salesman Problem

Fredrick Mtenzi
Dublin Institute of Technology
School of Computing, DIT Kevin Street, Dublin 8
Ireland

1. Introduction

The Sparse Travelling Salesman Problem (Sparse TSP) which is a variant of the classical Travelling Salesman Problem (TSP) is the problem of finding the shortest route of the salesman when visiting cities in a region making sure that each city is visited at least once and returning home at the end. In the Sparse TSP, the distance between cities may not obey the triangle inequality; this makes the use of algorithms and formulations designed for the TSP to require modifications in order to produce near-optimal results.

When solving the Sparse TSP, our main interest is in computing feasible tours at a reasonable computational cost. In addition because it is not possible to get an optimal solution most of the time, we would like to have some guarantee on the quality of the tours (solutions) found. Such guarantees can most of the time be provided if a lower bound on the length of a shortest possible tour is known.

It is also the case that most algorithms for finding exact solutions for large Standard TSP instances are based on methods for finding upper and lower bounds and an enumeration scheme. For a given instance, lower and upper bounds are computed. In most cases, these bounds will not be equal, and therefore, only a quality guarantee for the feasible solution can be given, but optimality cannot be proved. If the upper and lower bounds coincide, a proof of optimality is achieved.

Therefore, the determination of good tours and derivation of tight lower bounds can be keys to a successful search for optimal solutions. Whereas there have been a lot of work and progress in designing heuristic methods to produce upper bound, the situation for lower bounds is not as satisfying.

In general for the Standard TSP, lower bounds are obtained by solving relaxations of the original problem in the sense that one optimizes over some set containing all feasible solutions of the original problem as a (proper) subset. This then means, for example, that the optimal solution of the relaxed problem gives a lower bound for the value of the optimal solution of the original problem. In practice, the methods usually used for computing lower bound for the Standard TSP are the Held-Karp lower bound (Johnson et al., 1996) and Lagrangian relaxation (Reinelt, 1994).

Since the Sparse TSP is an NP-Hard combinatorial optimization problem as per Fleischmann (Fleischmann, 1985), the standard technique to solve it to optimality is based on an enumeration scheme which for large problems is computationally expensive. Therefore a natural way is to use the Sparse TSP heuristics to obtain a near-optimal solution. Solutions obtained by heuristics for the Sparse TSP provide the upper bounds. Heuristics produce feasible solutions but without any quality guarantees as to how far off they may be from the optimal feasible solution. In order to be able to assess the performance of heuristics we need to find the lower bound of the problem.

Therefore, in this chapter we are interested in exploring methods for computing tight lower bounds for the Sparse TSP. This is the case because we do not have the luxury of comparing with what other researchers have done, since most of the work in the TSP has been focused on the Standard TSP. For example, in the Standard TSP there are sample instances with optimal solutions provided in the TSPLIB for most of the problems (see (Reinelt, 1992)). The results given in TSPLIB include a provable optimal solution if available or an interval given by the best known lower bound and upper bound. As far as we are aware there are no such benchmark results for the Sparse TSP which is studied in this chapter.

A lower bound gives us the quality guarantee of the near-optimal solution obtained by using heuristic methods. The most widely used procedure for finding the lower bound for the Standard TSP is the Held and Karp lower bound (Held & Karp, 1970). Johnson et al (Johnson et al., 1996) provide empirical evidence in support of using the Held and Karp (HK) lower bound as a stand-in for the optimal tour length when evaluating the quality of near-optimal tours. They show that for a wide variety of randomly generated instances the optimal tour length averages less than 0.8% over the HK lower bound, and for the real world instances in TSPLIB the gap is always less than 2%. A tight lower bound for the Sparse TSP will play a key role in developing and assessing the performance of the Sparse TSP heuristic methods.

Definitions

A *relaxation* of an optimization problem P is another optimization problem R, whose set of feasible solutions $\Re$ properly contains all feasible solutions P of P. The objective function of R is an arbitrary extension on $\Re$ of the objective function of P. Consequently, the objective function value of an optimal solution to R (minimisation case) is less than or equal to the objective function value of an optimal solution to P. If P is a hard combinatorial problem and R can be solved efficiently, the optimal value of R can be used as a lower bound in an enumeration scheme to solve P. The closer the optimal value of R to the optimal value of P, the more efficient is the enumeration algorithm.

A *lower bound* of the TSP is the value obtained by solving a relaxation of the original problem or by using heuristics. Its value is in most cases less than the optimal value of the original problem, it is equal to optimal value when the value of lower bound is equal to the value of the upper bound.

In this chapter, we propose and give computational experience on two methods of finding lower bounds for the Sparse TSP, the chapter is organised as follows. In section 2, we give the background and related research in the Travelling Salesman Problem and specifically how this work relates to the Sparse TSP. In section 3, we discuss methods for finding the lower bound of the Sparse TSP, and give the formulation and relaxation for the Linear Programming relaxation of the Sparse TSP. Further, we introduce the Arc-cutset Partial Enumeration Strategy as a strategy for finding the lower bound of the Sparse TSP. Finally, section 4 gives the conclusions and summary.

2. Related works

The Standard TSP has been an area of intensive research since the late 1950's as demonstrated in (Lawler et al., 1985). The first major breakthrough came in 1954, in a seminal paper by (Dantzig et al., 1954) where a description of a method for solving the Standard TSP was given and its power illustrated by solving an instance of 49 cities. From there on there has been a lot of research published on the Standard TSP and its variants.

Most of the methods used for solving the Standard TSP to optimality are of the branch and bound variety where, at each node of the branch and bound tree, lower bounds are computed by solving related problems which are relaxations of the original TSP (see (Camerini et al., 1975), (Gabovich, 1970), Held and Karp (Held & Karp, 1970), Padberg and Hong (Padberg & Hong, 1977), and Rubinshtein (Rubinshtein, 1971). As for all branch and bound methods, the quality of the computed lower bounds at each node has much greater influence on the effectiveness of the algorithm than any branching rules that may be used to generate the subproblems during the search. Branch and Bound techniques have been used successfully in optimization problems since the late 1950's. Several different branch and bound algorithms are possible for the travelling salesman problem. A survey of these is given in Bellmore and Nemhauser (Bellmore & Nemhauser, 1968). A variation of Branch and Bound techniques using cutting plane techniques called Branch-and-Cut by Padberg and Rinaldi((Padberg & Rinaldi, 1989), (Padberg & Rinaldi, 1991)) is a much more powerful technique for computing solutions for the Standard TSP.

Held and Karp (Held & Karp, 1970), (Held & Karp, 1971) pioneered an iterative approach which uses a technique called Lagrangian Relaxation to produce a sequence of connected graphs which increasingly resemble tours. This technique is based on the notion of a 1-tree and the bound generated is called the Held-Karp lower bound (HK lower bound). Formulating the Standard TSP as an integer linear programming problem (see Dantzig et al (Dantzig et al., 1954), Miller et al (Miller et al., 1960), Fox et al (Fox et al., 1980), and Claus (Claus, 1984) and systematically solving its relaxations is another way of obtaining a lower bound for TSP. Bounds from the solutions of the assignment problem, the matching problem, and the shortest n-path problem have also been suggested and explored by

Christofides (Christofides, 1979) who also gave a brief survey and references. Johnson et al (Johnson et al., 1996) use the HK lower bound as a stand-in for the optimal tour length when evaluating the quality of near-optimal tours in a number of their studies in which they solve problems of up to one million cities.

When the theory of NP-completeness was developed, the Standard TSP was one of the problems shown to be NP-hard by Karp in (Karp, 1972). This remains true even when additional assumptions such as the triangle inequality or Euclidean distances are involved (see Garey (Garey, 1976)). These results imply that a polynomially bounded exact algorithm for the TSP is unlikely to exist. Nevertheless ingenious algorithms for the TSP have been proposed by Little et al (Little et al., 1963), Held and Karp (Held & Karp, 1970), (Held & Karp, 1971), Miliotis (Miliotis, 1976), Crowder and Padberg (Crowder & Padberg, 1980). Over the past four decades the TSP has remained the prototype of a "hard" combinatorial problem. Since the introduction of NP-completeness theory in 1971 and the subsequent inclusion of the TSP in the NP-complete class, some of the mystery has gone out of the TSP. A complete treatment of NP-completeness theory and its relationship to the TSP is given in Garey and Johnson (Garey & Johnson, 1979).

The NP-complete results have given a new impetus to heuristic methods for solving "hard" combinatorial problems. Due to the difficulty of the TSP, many heuristic procedures have been proposed and developed. These heuristics have been compared analytically in the ground-breaking paper of Rosenkrantz, Stearns and Lewis (Rosenkrantz et al., 1977) by studying their worst-case behaviour. Alternatively, computational experimentation may be used to compare the performance of these heuristics, as in the work of Golden et al (Golden et al., 1980) and Stewart (Stewart Jr, 1987). For example, Stewart (Stewart Jr, 1987) describes a number of new algorithms designed specifically to perform well on Euclidean problems.

Much work has been done on fast heuristic algorithms for the Standard TSP. There is a trade-off between the speed of an algorithm and its ability to yield tours which are close to the optimal. The following studies show an enormous interest which researchers have shown to heuristic algorithms. These studies include those of Adrabinski and Syslo (Adrabinski & Syslo, 1983) , Golden et al (Golden et al., 1980), Johnson and McGeoch (Johnson & McGeoch, 1995), Bentley (Bentley, 1993), Reinelt (Reinelt, 1992), and Júnger et al (Júnger et al., 1995). There are two broad classes of heuristics for the travelling salesman problem constructive and improvement heuristics. A typical tour construction heuristic method starts with a node and adds others one by one until the tour is complete. Many other variants are described in Bellmore and Nemhauser (Bellmore & Nemhauser, 1968), Rosenkrantz et al (Rosenkrantz et al., 1977), Johnson (Johnson, 1990), Golden et al (Golden et al., 1980), Golden and Stewart (Golden & Stewart, 1985), Gendreu et al. (Gendreau et al., 1992), and Bentley (Bentley, 1993).

The arc-exchange strategy was first applied to the Standard TSP by Croes (Croes, 1958). He suggested the 2-optimal algorithm for the symmetric TSP. About the same time and independently, a 3-optimal strategy was suggested by Bock in (Bock, 1958). However, it was Lin in (Lin, 1965) who truly established through extensive empirical study that the 3-optimal algorithm was indeed an excellent approximation algorithm for the Standard TSP. A substantial improvement in implementation of the 3-optimal (in general, *r-optimal* where $r \geq 2$) algorithm was given by Christofides and Eilon in (Christofides & Eilon, 1979). Lin and Kernighan (Lin & Kernighan, 1973) added another level of sophistication to the *r*-optimal algorithm. Instead of having a fixed value of 2 or 3, *r* was allowed to vary. Their paper is a classic paper on the local search heuristics for the Standard TSP. In particular, Lin-Kernighan(LK) and its variants are widely recognized as the best (most accurate) local search heuristic for the TSP.

The ejection chains method introduced by Glover (Glover, 1992) have been used to generate compound neighbourhood structures for the Standard TSP by Rego (Rego, 1998). It should be noted that this neighbourhood has been used to solve problems taken from Travelling Salesman Problem LIBrary(TSPLIB) and performed better than the best iterated LK, but it took more time.

In order for the heuristics algorithms to work well efforts should be spent on designing efficient data structures. Data structures play an important role in the design and implementation of the Standard TSP algorithms. The following data structures have been used in the Standard TSP study, the *k-d tree* of Bentley discussed and used in Bentley (Bentley, 1993), (Bentley, 1975), (Bentley, 1990), and (Bentley, 1990). Fredman et al (Fredman et al., 1995) discuss and give implementation details on the following data structure for the Standard TSP, *array-based* and *splay trees*.

We have witnessed remarkable progress in Mathematical Programming, Polyhedral Theory, and significant advances in computer technology that have greatly enhanced the chances of finding exact solutions to reasonably large combinatorial optimization problems. Nevertheless, these problems still pose a real challenge in the sense that it is hard to solve realistically large problem instances in reasonable computational times. This challenge has led to the development of many approximate algorithms and has made the science of heuristics a fast growing area of research. For more details on combinatorial optimization problems we refer the reader to Nemhauser and Wolsey (Nemhauser & Wolsey, 1988), Papadimitriou and Steiglitz (Papadimitriou & Steiglitz, 1982), Garey and Johnson (Garey & Johnson, 1979).

Metaheuristics are a class of approximate solution methods, which have developed dramatically since their inception in the early 1980s. They are designed to solve complex optimization problems where classical heuristics and other optimization methods have failed to be effective and efficient. A *metaheuristic* can be defined as an iterative generation process which guides a subordinate heuristic by combining intelligently different concepts for exploring and exploiting the search space, learning strategies are used to structure information in order to find efficiently near-optimal solutions (see Osman (Osman, 1995), and Osman and Kelly (Osman & Kelly, 1996)). Hence, metaheuristics include, but are not limited to: Constraint Logic Programming; Genetic Algorithms; Greedy Random Adaptive Search Procedures; Neural Networks; Non-monotonic search strategies; Problem and heuristic search-space; Simulated Annealing; Tabu Search; Threshold Algorithms and their hybrids. These techniques are based on concepts borrowed from biological evolution, intelligent problem solving, Mathematical and Physical Sciences, Nervous System, and

Statistical Mechanics. For details we refer the reader to Reeves (C.R.Reeves, 1993; Reeves, 1993). However, Meta-heuristics have not been very successful in solving the Standard TSP according to Johnson and McGeoch (Johnson & McGeoch, 1995). They have been very effective in other areas such as industrial applications and communications networks.

The solution produced by using local search procedures can be very far off from optimality. In order to avoid such disadvantages while maintaining the simplicity and generality of the approach, the following concepts which form the basis of most metaheuristics are considered see Reeves (Reeves, 1993). Start from good initial solutions which can be generated intelligently using a greedy random adaptive search designed by Feo and Resende (Feo & Resende, 1995), or space-search methods in Storer, Wu and Vaccari (Storer et al., 1995). Use the learning strategies of neural networks discussed in Sharda (Sharda, 1994), and Tabu search in Glover (Glover, 1995) that gather information during the algorithms execution in order to guide the search to find possibly better solutions. Employ non-monotonic search strategies that sample/accept neighbours based on hybrid modifications of simulated annealing and Tabu Search methods or others see (Glover (Glover, 1995), (Glover, 1995)), (Hu, Khang and Tsao (Hu et al., 1995)), (Osman (Osman, 1993), (Osman, 1995), (Osman, 1995)) and (Osman and Christofides (Osman & Christofides, 1994)).

Sahni and Gonzales in (Sahni & Gonzales, 1976) showed that if a triangle inequality is not satisfied, the problem of finding an approximate solution to the TSP within any fixed bound ratio of the optimum is as hard as finding an exact solution. A comprehensive treatment of various aspects of the TSP can be found in the collection of research papers in Lawler et al (Lawler et al., 1985).

3. Methods for finding Lower bound for the Sparse TSP

The standard technique for obtaining lower bounds on the Standard TSP is to use a relaxation that is easier to solve than the original problem. These relaxations can have either discrete or continuous feasible sets. Several relaxations have been considered over years for the Standard TSP. We are going to introduce modifications to these relaxations, so that they can be used to find lower bounds for the Sparse TSP at a reasonable computational effort.

We can illustrate the relaxation of an optimization problem by figure 1 in which the region ABCD is the feasible region of the relaxed problem, while the region MNOPQRS is the feasible region of the original “true” problem, which in this case happens to contain integer points. Solution ABCD may be obtained when we relax integrality constraints. Relaxation of the combinatorial optimization has become so popular because in most cases the problems can be solved with reasonable computational effort, if not easier than the original problem.

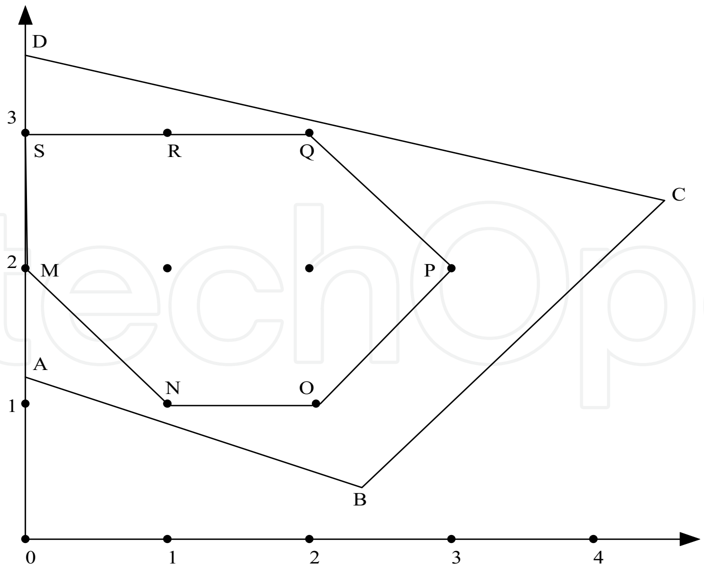


Fig. 1. Region MNOPQRS shows a feasible solution to the original "true" problem while region ABCD shows a feasible solution to the relaxed problem

The HK lower bound is the solution to the LP relaxation of the integer programming formulation of the Standard TSP (see Dantzig et al (Dantzig et al., 1954), Reinelt (Reinelt, 1994) and Johnson et al (Johnson et al., 1996). That is, it is the Integer Linear Programming with the integrality constraints relaxed. The HK lower bound provides a very good estimate of optimal tour length for the Standard TSP. This measure has enormous practical value when evaluating the quality of near optimal solutions for large problems where the true optimal solutions are not known or are computationally expensive to find. The HK lower bound has been used as a stand-in for the optimal tour length when evaluating the quality of near-optimal tours in a lot of studies (for example, in Johnson et al (Johnson et al., 1996)).

Although the HK lower bound can be evaluated exactly by Linear Programming techniques, code for doing this efficiently for problems larger than a few hundred cities is not readily available or easy to produce (see Valenzuela and Jones (Valenzuela & Jones, 1996)). In addition linear programming implementations (even efficient ones) do not scale well and rapidly become impractical for problems with many thousands of cities. To be able to find the HK lower bound, a procedure for finding violated inequalities must be provided. This is not a simple matter of automatically generating violated inequalities. It is because of the above mentioned difficulties that most researchers have preferred to use the iterative estimation approach for finding lower bound for the Standard TSP proposed by Held and Karp (Held & Karp, 1970), (Held & Karp, 1971). In this chapter we use this method and modify it to solve the Sparse TSP problems.

3.1 The LP Relaxations for the Sparse TSP

The formulation for the Integer Linear Programming (ILP) Sparse TSP is given as:

$$\min_{x} \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} x_{ij} \tag{1.1}$$

Subject to

$$\sum_{j:(i<j)\in A} x_{ij} + \sum_{j:(j<i)\in A} x_{ji} = 2m_i, \text{ for all } i \in N \tag{1.2}$$

$$\sum_{i\in S, j\in N-S, i<j} x_{ij} + \sum_{i\in S, j\in N-S, j<i} x_{ji} \geq 2, \text{ for all } \mathrm{S} \subset \mathrm{N}, S \neq \varnothing \tag{1.3}$$

$$m_i \geq 0 \text{ for all } i \in N \tag{1.4}$$

$$x_{ij} \in \{0,1\} \text{ for all } i, j = 1, \cdots, N \tag{1.5}$$

By relaxing the integrality constraint (1.5) we get the Linear Programming (LP) relaxation for the ILP Sparse TSP where equations (1.1, 1.2, 1.3 and 1.4) remains the same and constraint (1.5) becomes,

$$x_{ij} \geq 0 \text{ for all } i, j = 1, \cdots, N \tag{1.6}$$

Note: that any integral solution to the LP relaxation is a tour.
We solved the LP relaxation for the modified ILP Sparse TSP formulation problems, results given in figure 7 were obtained by using violated arc-cutset constraints.

3.2 The LP relaxation for the EFF for the Sparse TSP

From the single commodity flow formulation, we present its modification which we call the Embedded Flow Formulation (EFF) for the Sparse TSP. This formulation involves a polynomial number of constraints, even though the number of variables is increased considerably. The EFF for the Sparse TSP is given as:

$$\min_x \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} x_{ij} \tag{1.7}$$

Subject to

$$\sum_{j:(i<j)\in A} x_{ij} + \sum_{j:(j<i)\in A} x_{ji} = 2m_i, \text{ for all } i \in N \tag{1.8}$$

$$y_{ij} - (n-1)x_{ij} \leq 0, \text{ for all } (i, j) \in A \tag{1.9}$$

$$\sum_{j:(i<j)\in A} y_{ij} - \sum_{j:(j<i)\in A} y_{ji} = -1, \text{ for all } i = 2, 3, \cdots, n \tag{1.10}$$

$$\sum_{i\in A} (y_{1i} - y_{i1}) = n - 1 \tag{1.11}$$

$$m_i \geq 0, \text{ for all } i \in N \tag{1.12}$$

$$x_{ij} = 0 \text{ or } 1, \text{ for all } (i, j) \in A \tag{1.13}$$

$$y_{ij} \geq 0, \text{ for all } (i, j) \in A \tag{1.14}$$

The LP relaxation for the EFF for the Sparse TSP formulation is obtained by relaxing the integrality constraint (1.13). All other equations remain the same except constraint (1.13) changes to:

$$x_{ij} \geq 0, \text{ for all } (i, j) \in A \tag{1.15}$$

It was interesting to know how close the lower bound Z_{LB} obtained by solving the subtour relaxation is to the length of an optimal tour Z_{opt} . Worst-case analysis of HK lower bound by

Wolsey (Wolsey, 1980) and Shmoy and Williamson (Shmoys & Williamson, 1990) show that for any cost function C satisfying the triangle inequality, the ratio Z_{LB}/Z_{opt} is at least $2/3$. The $2/3$ lower bound is not shown to be tight and actually it is conjectured by Goemans (Goemans, 1995) that $(Z_{LB}/Z_{opt}) \geq 3/4$. Our computational results show that for many instances the above ratio is very close to 1.

The results we obtained for the EFF for the Sparse TSP are presented in figure 8. In general the LP relaxation is not equal to the minimum tour length but it is very close. LP relaxation for the ILP Sparse TSP gives a much tighter lower bound than the LP relaxation for the EFF for the Sparse TSP and requires less iterations.

3.3 An Arc-cutset Partial Enumeration Strategy (APES)

In this section we are proposing a strategy for quickly generating some of the violated arc-cutset constraints which we call an *Arc-cutset Partial Enumeration Strategy* (APES). The APES is based on the following observation, using the formulation for the ILP Sparse TSP, we can drop the connectivity constraints. When the resulting formulation is solved the solution produces a lot of disconnected components, all have at least two nodes connected by two arcs. That is to say each component is a subtour, and the obtained solution is not a tour.

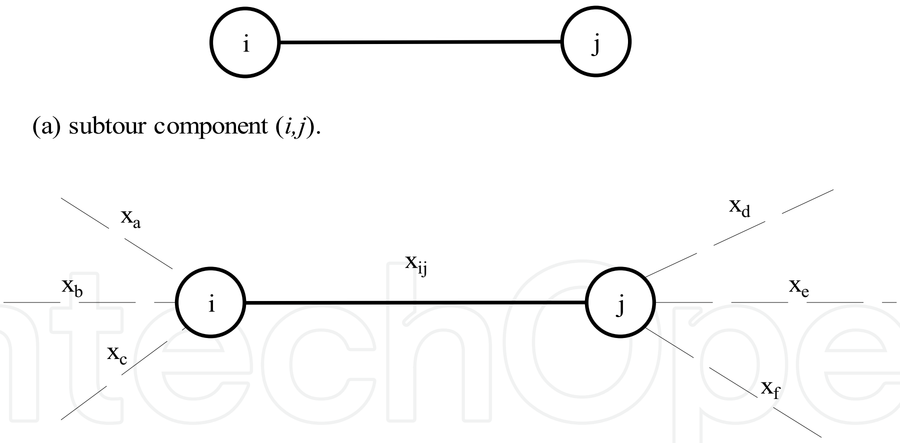
These components needed to be connected with other components to produce a single connected component, a tour. To be able to achieve this, we generated an arc-cutset constraint for each component. In other words we generated an arc-cutset constraint for each arc in the graph. This approach is reasonable to the Sparse TSP because the number of arcs m in the sparse graph is $O(n)$ as opposed to $O(n^2)$ in the complete graph. Arc cutset identified between two components resulted in those components being connected plus several others. Therefore, even if when the APES is applied at first, there are may be say five disconnected components, it does not mean that five arc cutset will have to be identified. Our computational study showed that the number of arc cutset to be identified is always less than five. However, we were not able to come up with the relationship between the size of the problem and the number of arc cutset to be identified before obtaining a tour.

The arc-cutset constraints generated this way are all valid inequalities. Naddef and Rinaldi (Naddef, 1992), Cornuéjols et al (Cornuéjols et al., 1985), and Swamy and Thulasiraman (Swamy & Thulasiraman, 1981) have shown the validity of the arc-cutset constraints. They say that once the components are connected then the violated arc-cutset constraints are valid inequalities.

Our algorithm for the APES is given below.

```
An Arc-cutset Partial Enumeration Strategy algorithm
Step 1: Formulate the problem using evenness condition
          Constraints and integrality constraints only.
          Let nodes be the number nodes in the starting path
          Let nodesv be the number of nodes to be visited
          Let k := 2
Step 2: For nodes := k to n do
          For nodesv := 3 to n – k do
                    List all arcs incident to the path
                    with end node 1 and nodesv
                    add the arc-cutset constraint to the formulation
          EndFor
          EndFor
Step 3: Solve the new formulation using any LP solver
Step 4: stop
```

In forming the arc-cutset constraints, we first used a subtour component consisting only of two end nodes i and j with (i,j) as a connected component. The violated arc-cutset constraints were constructed by listing all arcs incident to node i and node j to form one violated arc-cutset constraint, i.e., all arcs incident with a subtour component. The arc connecting node i and node j was not included in the arc-cutset constraints. Figure 2 shows how using component (i,j) arc-cutset constraints was formed. This is what takes place in step 2 of the APES algorithm.



(a) subtour component (*i,j*).

(b) subtour component with incident arcs to node *i* and node *j*.

Fig. 2. Formulation of the arc-cutset constraint

$$x_a + x_b + x_c + x_d + x_e + x_f \ge 2 \tag{1.16}$$

The arc-cutset constraints which are generated by the APES are used to connect components. Since the APES starts by using the evenness condition constraints and integrality constraints, while omitting the connectivity constraints. For example the twenty nodes problem shown in figure 3, is used to demonstrate how the APES works.

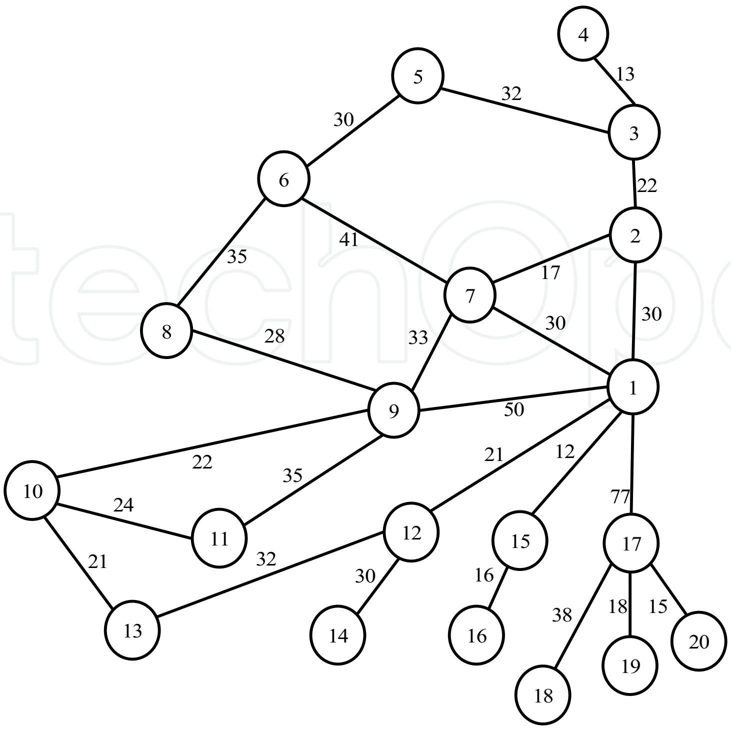


Fig. 3. A twenty nodes problem to demonstrate how APES works

Solving the twenty nodes problems before adding the violated arc-cutset constraints generated by the APES gives the disconnected tours illustrated in figure 4 and whose objective function is 524.

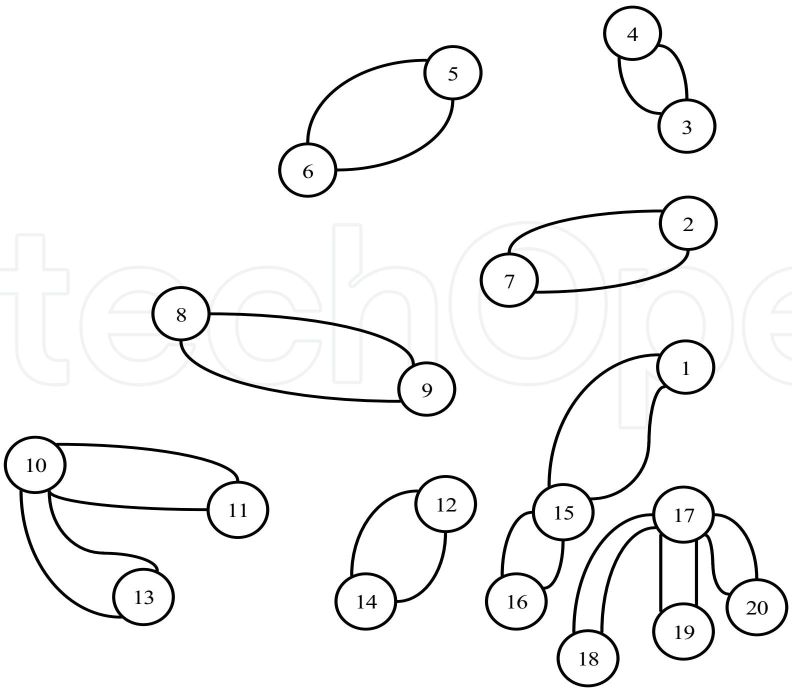


Fig. 4. A twenty nodes problems with disconnected subtour components

After adding the violated arc-cutset constraints generated by the APES we got a tour as illustrated in figure 5 and its objective function was 765, which in this case happened to be the optimal tour.

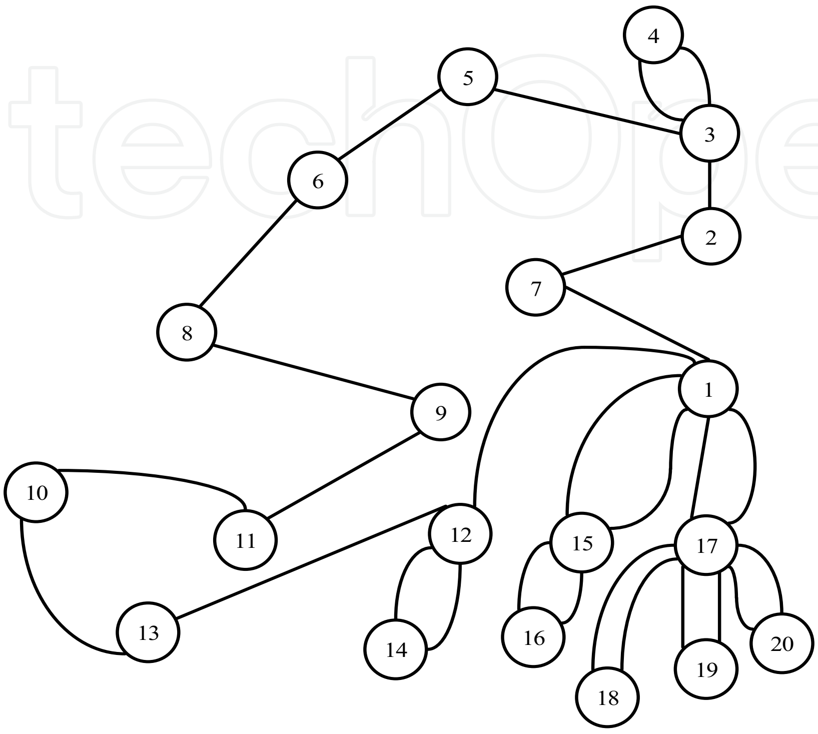


Fig. 5. A twenty nodes problem tour

For all the problems reported in this study, we used the APES to generate violated arc-cutset constraints from each arc. This strategy was used to ensure that disconnected components were connected to form a tour. The APES was able to produce optimal tours for all small problems we solved of up to 1000 nodes. It is interesting to note how the algorithm performed in some graphs. For example, in the case of the 30 nodes problem we had to add eleven violated arc-cutset constraints before getting a tour, while we had to add only two violated arc-cutset constraints for the 67 nodes problem to get a tour. As pointed out earlier the number of violated arc-cutset required to be identified seem to be a function of the nature of the graph and not its size.

We then extended this technique of identifying violated arc-cutset constraints. These new violated arc-cutset constraints were formed by visiting a path of three or more nodes together. The first violated arc-cutset constraint was formed by visiting any three nodes which formed a path. When forming these constraints, we included all arcs which were incident to nodes 1 or 2 or 3 and ignore arcs connecting nodes 1, 2, and 3. The next constraint was formed by adding the fourth node to the path and the violated arc-cutset constraint was identified by listing all nodes incident to the path consisting of four nodes ignoring the arcs which form the path. The process continued until we got a tour. Figure 6 shows how these violated arc-cutset constraints were formed.

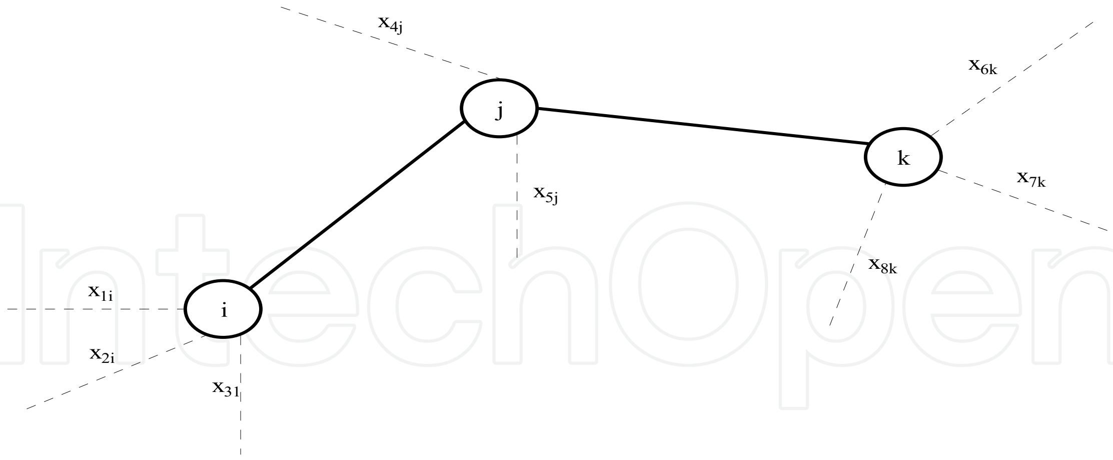


Fig. 6. An example of how the extended technique for identifying violated arc-cutset constraints works

From figure 6 the following violated arc-cutset constraint (1.17) will be formed:

$$x_{1i} + x_{2i} + x_{3i} + x_{4j} + x_{5j} + x_{6k} + x_{7k} + x_{8k} \geq 2 \tag{1.17}$$

The results in figure 6 shows how the APES method performed when the starting path visited 3, 4, … , 10 nodes together. In other words at first the starting path had 3 nodes and we extended the path by adding one node at a time. The second time the starting path had 4 nodes and we extended the path by adding one node at a time. We continued increasing the number of nodes in a starting path, until at last our starting path had 10 nodes to start with. We got in a good number of cases substantial improvements in the lower bound as we increased the number of nodes in the starting path. However, as the number of nodes in the starting path went beyond 10 we got marginal improvement.

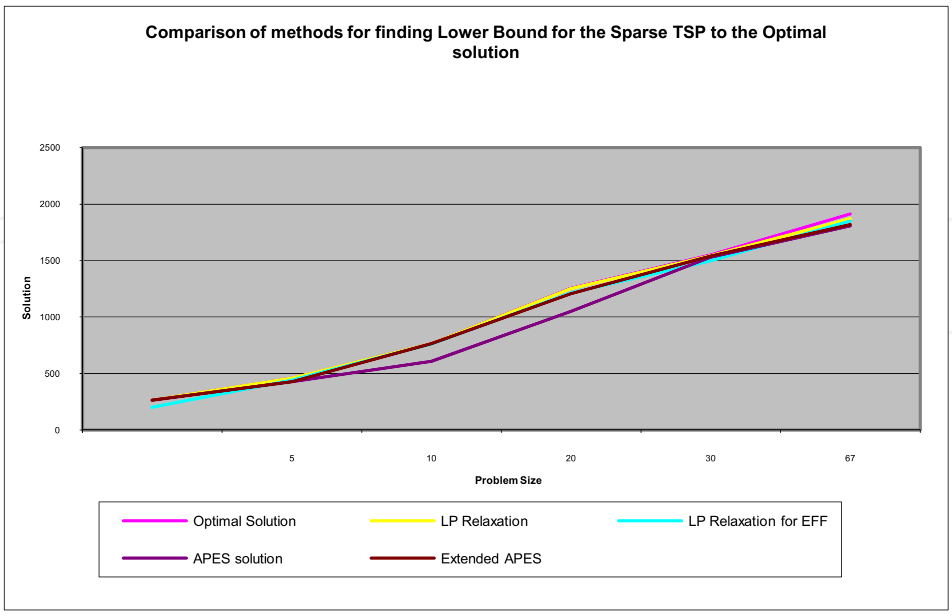


Fig. 7. Comparison of methods for finding Lower Bound for the Sparse TSP to the optimal solution

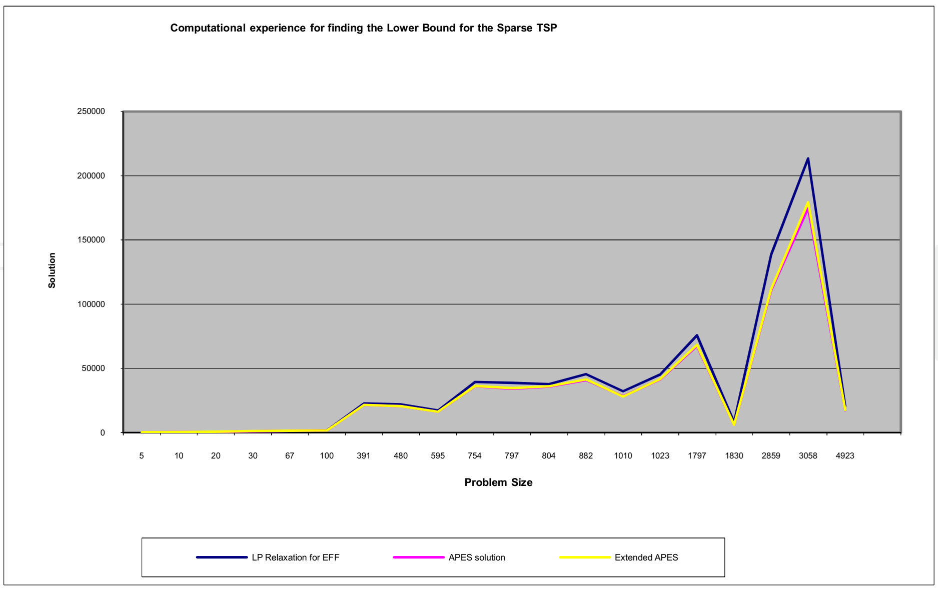


Fig. 8. Computational experience for finding the Lower Bound for the Sparse TSP

The order of complexity of our APES is $O(n)$. This is far less than the connectivity constraints with the order of complexity of $O(n^2)$. This makes our strategy much easier to use.

4. Summary and future work

When optimization problems arise in practice we want to have confidence in the quality of the solutions. Quality guarantees are required because in most cases and especially for large problems, it is not always possible to find an optimal solution. Quality guarantees become possible by being able to compute tight lower bounds at a reasonable computational cost. In this chapter we have proposed two methods for finding tight lower bound for the Sparse TSP using the modified Linear Programming relaxation method and the Arc-cutset Partial Enumeration Strategy.

When the LP relaxation method is used to find a lower bound for the ILP Sparse TSP, finding arc-cutset constraints is a headache especially for large problems. There are procedures for identifying violated arc-cutset constraints automatically in practice, such as the separation routines. These procedures are computational expensive and therefore were not used in this study.

The Arc-cutset Partial Enumeration Strategy proposed is a simple and fast way of getting a lower bound without spending time in a separation algorithm. Computational results show that the lower bounds obtained by using this method are as comparable as those obtained using Separation algorithms.

A lower bound on the optimal value (assuming a minimization problem) is obtained from a relaxation of the integer program. In the past fifteen years attention has shifted from

Lagrangian Relaxation to Linear Programming relaxation, since the latter type of relaxation can be strengthened more easily by using cutting planes. Combining cutting planes and Lagrangian relaxation usually causes convergence problems as discussed by Aardal et al (Aardal et al., 1997). Utilising various modified LP relaxation to suit the Sparse TSP has given us the tightest lower bound of all lower bound techniques we have discussed in this chapter.

5. References

Aardal, K. et al. (1997). A decade of Combinatorial Optimization, Working paper, Uetrecht University, Computer Science Department, Netherlands.

Adrabinski, s. A. & Syslo, M. M. (1983). Computational experiments with some approximation algorithms for the travelling salesman problem. *Zastos. mat.*, 18, 1, 91-95.

Bellmore, M. & Nemhauser, G. L. (1968). The Traveling salesman problem: A survey. *Operations Research*, 16, 538-558.

Bentley, J. J. (1993). Fast Algorithms for Geometric Traveling Salesman Problems. *ORSA Journal on Computing*, 4, 4, 387-411.

Bentley, J. L. (1990). Experiments on traveling salesman heuristics, *The 1st Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, Philadelphia, PA.*, 91-99.

Bentley, J. L. (1990). K-d trees for semidynamic point sets, *The 6th Annual Symposium on Computational Geometry, ACM, New York*, 187-197.

Bentley, J. L. (1975). Multidimensional binary search trees used for associative search. *Comm. ACM*, 18, 309-517.

Bock, F. (1958). An algorithm for solving Traveling Salesman and related network optimization problems, *inproceedings 14th National meeting of the ORSA*, St. Louis, Mo., St. Louis, Mo.

Reeves C.R. (1993). *Modern Heuristic Techniques for Combinatorial Problems*, McGraw-Hill Book Company.

Camerini, P. et al. (1975). Travelling salesman problem: Heuristically guided search and modified gradient techniques.

Christofides, N. (1979). The Travelling Salesman Problem, *Combinatorial Optimization*, N. Christofides, Mingozzi, A., Toth, P. and Sandi, C., 131-149, John Wiley and Sons. John Wiley and Sons.

Christofides, N. & Eilon, S. (1979). Algorithm for large-scale Traveling Salesman Problems. *Operational Research Quarterly*, 23, 4, 511-518.

Claus, A. (1984). A new formulation for the traveling salesman problem. *SIAM Journal of Algebraic Discrete Methods*, 5, 21-25.

Cornuéjols, G. et al. (1985). The travelling salesman problem on a graph and some related integer polyhedra. *Mathematical Programming*, 33, 1-27.

Croes, G. A. (1958). A method for solving traveling salesman problems. *Operations Research*, 6, 791-812.

Crowder, H. & Padberg, M. (1980). Solving large-scale symmetric travelling salesman problem to optimality. *Management Science*, 26, 495-509.

Dantzig, G. et al. (1954). Solution of a large-scale Traveling-Salesman Problem. *Operations Research*, 2, 393-410.

Feo, T. A. & Resende, M. G. C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, 6, 109-118.

Fleischmann, B. (1985). A cutting plane procedure for travelling salesman problem on road networks. *European Journal of Operational Research*, 21, 307-317.

Fox, K. et al. (1980). An n-constraint formulation of the (time-dependent) traveling salesman problem. *Operations Research*, 28, 1018-1021.

Fredman, M. L. et al. (1995). Data Structures for Travelling Salesmen. *Journal of Algorithms*, 18, 432-475.

Gabovich, E. J. (1970). The small travelling salesman problem. 19, 27-51.

Garey, M. (1976). Some NP-complete geometric problems, *inproceedings Proceedings of the 8th SIGACT Symp. on the theory of computing*, 10-22.

Garey, M. R. & Johnson, D. S. (1979). *Computers and Intractability: A Guide to the theory of NP-Completeness*, W.H. Freeman, San Francisco.

Gendreau, M. et al. (1992). New Insertion and Postoptimization procedure for the traveling salesman problem. *Operations Research*, 40, 1086-1094.

Glover, F. (1995). Tabu thresholding: improved search by non-monotonic trajectories. *ORSA Journal on computing*, 7, 426-436.

Glover, F. (1995). Tabu search fundamentals and uses, Graduate School of Business, University of Colorado, Boulder.

Glover, F. (1992). New ejection chain and alternating path methods for the traveling salesman problems, *Operations research and computer science: new developemnts in their interfaces*, O. Balci, Sharda, R. and Zenios, S. A., 27-49, Pergamon Press, Oxford., Pergamon Press, Oxford.

Goemans, M. X. (1995). Worst-case Comparison of valid Inequalities for the TSP. *Mathematical Programming*, 69, 335-349.

Golden, B. et al. (1980). Approximate travelling salesman algorithms. *Operations Research*, 28, 3, 694-711.

Golden, B. L. & Stewart, W. R. (1985). Empirical analysis of heuristics, *The Traveling Salesman Problem*, E. L. Lawler, Lenstra, J. K., Kan, A. H. G. R. and Shmoys, D. B., 207-249, John Wiley and Sons Ltd., John Wiley and Sons Ltd.

Held, M. & Karp, R. M. (1970). The Travelling Salesman Problem and Minimum spanning Trees, Part I. *Operations Research*, 18, 1138-1162.

Held, M. & Karp, R. M. (1971). The Travelling Salesman Problem and Minimum spanning Trees, Part II. *Mathematical Programming*, 1, 6-25.

Hu, T. C. et al. (1995). Old bachelor acceptance: A new class of non-monotonic threshold accepting methods. *ORSA Journal on computing*, 7, 417-425.

Johnson, D. S. & McGeoch, L. A. (1995). The Traveling Salesman Problem: A case study, *Local search in Combinatorial Optimization*, E. Aarts and Lenstra, J. K., 215-310, John Wiley and Sons Ltd., John Wiley and Sons Ltd.

Johnson, D. S. et al. (1996). Asymptotic Experimental Analysis for the Held-Karp Traveling Salesman Bound, *Proceedings of the 7th Annual ACM-SIAM symposium on Discrete Algorithms*, 341-350, January 1996.

Johnson, D. S. (1990). Local optimization and the traveling salesman problem, *ICALP '90*, 446-461, Springer-Verlag.

Júnger, M. et al. (1995). The traveling salesman problem, *Network Models, Handbooks in Operations Research and Management Science*, M. O. Ball, Magnanti, T. L., Monma, C. L. and Nemhauser, G. L., 225-330, North-Holland. Amsterdam, North-Holland, Amsterdam.

Karp, R. M. (1972). Reducibility among Combinatorial Problems, *Complexity of Computer Computations*, R. E. Miller and Thatcher, J. W., 85-103, Plenum Press, New York., Plenum Press, New York.

Lawler, E. L. et al. (1985). *The Traveling Salesman Problem*, John Wiley and Sons Ltd.

Lin, S. (1965). Computer solutions of the traveling salesman problem. *The Bell system technical journal*, 44, 2245-2269.

Lin, S. & Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling salesman problem. *Operations Research*, 21, 498-516.

Little, J. et al. (1963). An algorithm for the traveling salesman problem. *Operations Research*, 11, 6, 972-989.

Miliotis, P. (1976). Integer programming approaches to the traveling salesman problem. *Mathematical Programming*, 15, 367-378.

Miller, C. et al. (1960). Integer programming formulations and traveling salesman problem. *Journal of the Association for Computing Machinery*, 7, 326-329.

Naddef, D. (1992). The binested inequalities for the symmetric traveling salesman Polytope. *Mathematics of Operations Research*, 17, 4, 882-900.

Nemhauser, G. L. & Wolsey, L. A. (1988). *Integer and Combinatorial Optimization*, John Wiley and Sons Ltd.

Osman, I. H. & Kelly, J. P. (1996). *Meta-heuristics : Theory and Applications*, Kluwer Academic Publishers.

Osman, I. H. (1995). An Introduction to Meta-Heuristics, *Operational Research Tutorial papers*, M. Lawerence and Wilson, C., 92-122, Operational Research Society Press. Birmigham, U.K., Operational Research Society Press, Birmigham, U.K.

Osman, I. H. (1993). Metastrategy simulated annealing and Tabu search for the vehicle routing problem. *Annals of Operations Research*, 41, 421-432.

Osman, I. H. (1995). Heuristics for generalized assignment problem, simulated annealing and Tabu search approaches. *OR Spektrum*, 17, 211-228.

Osman, I. H. & Christofides, N. (1994). Capacitated clustering problems by hybrid simulated annealing and tabu search. *International Transactions in Operational Research*, 1, 317-329.

Padberg, M. & Rinaldi, G. (1989). A branch-and-cut approach to a traveling salesman problem with side constraints. *Management Science*, 35, 11, 1393-1414.

Padberg, M. & Rinaldi, G. (1991). A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM review*, 33, 1, 60-100.

Padberg, M. W. & Hong, S. (1977). On the symmetric travelling salesman problem: A computational study.

Papadimitriou, C. H. & Steiglitz, K. (1982). *Combinatorial Optimization : Algorithms and Complexity*, Prentice-Hall, Inc.

Reeves, C. R. (1993). *Modern Heuristic Techniques for Combinatorial Problems*, McGraw-Hill Book Company.

Rego, C. (1998). Relaxed tours and path ejections for the traveling salesman problem. *European Journal of operational research*, 2, 522-538.

Reinelt, G. (1992). Fast Heuristics for Large Geometric Traveling Salesman Problems. *ORSA Journal on Computing*, 4, 2, 206-217.

Reinelt, G. (1994). *The traveling salesman: Computational solutions for TSP applications*, Springer Verlag, 0-387-58334-3.

Rosenkrantz, D. J. et al. (1977). An analysis of several heuristics for the travelling salesman problem. *Siam Journal of Computing*, 6,3, 1977.

Rubinshtein, M. I. (1971). On the symmetric TSP. *Automatika I Telemekhanika*, 6, 126-126.

Sahni, S. & Gonzales, T. (1976). P-complete approximation problem. *J. ACM*, 23, 555-565.

Sharda, R. (1994). Neural networks for the MS/OR analyst. An application bibliography. *Interfaces*, 24, 116-125.

Shmoys, D. A. & Williamson, D. P. (1990). Analyzing the Held-Karp TSP bound: A monotonicity property with application. *Information Processing Letters*, 35, 281-285.

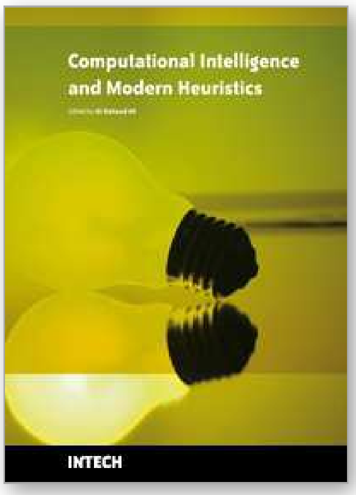
Stewart Jr, W. R. (1987). Accelerated branch exchange heuristics for symmetric travelling salesman problems. *Networks*, 17, 423-437.

Storer, R. H. et al. (1995). Local search in problem and heuristic space for job shop scheduling. *ORSA Journal on Computing*, 7, 453-458.

Swamy, M. N. S. & Thulasiraman, K. (1981). *Graphs, Networks, and Algorithms*, John Wiley and Sons Ltd.

Valenzuela, C. L. & Jones, A. J. (1996). Estimating the Held-Karp lower bound for the geometric TSP. *European Journal of Operational Research*, 102, 157-175.

Wolsey, L. A. (1980). Heuristic analysis, linear programming and Branch and Bound. *Mathematical Programming Study*, 13, 121-134.



Computational Intelligence and Modern Heuristics
Edited by Al-Dahoud Ali

ISBN 978-953-7619-28-2
Hard cover, 348 pages
Publisher InTech
Published online 01, February, 2010
Published in print edition February, 2010

The chapters of this book are collected mainly from the best selected papers that have been published in the 4th International conference on Information Technology ICIT 2009, that has been held in Al-Zaytoonah University, Jordan in the period 3-5/6/2009. The other chapters have been collected as related works to the topics of the book.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821