Adam Nicholas Letchford
BA(Hons.), MSc.

Polyhedral Results for Some Constrained Arc-Routing Problems

PhD Thesis, December 1996.
Department of Management Science,
The Management School,
Lancaster University.

Adam Nicholas Letchford, BA(Hons.), MSc.

Polyhedral Results for Some Constrained Arc-Routing Problems

PhD Thesis, December 1996.

Department of Management Science,
The Management School, Lancaster University.

Abstract.

Vehicle Routing Problems (VRPs) arise when routes must be devised for one or more vehicle(s) such that certain standards of efficiency are met and each route obeys one or more given restriction(s). VRPs are complex combinatorial optimisation problems and sophisticated algorithms are required in order to solve them.

In this thesis, four particular VRPs are considered. All four are *Arc Routing Problems*, in which the vehicles are required to traverse certain edges (roads) of a network rather than visit certain vertices (customers). For each problem, an integer programming formulation is given and the convex hull of feasible solutions is studied to yield strong valid inequalities. For two of the problems, detailed optimisation algorithms are presented and tested.

Contents.

3. The Rural Postman Problem.

3.1. Overview.

3.2. The GTSP, SGTSP and RPP Polyhedra.

3.3. HTC Inequalities for RPP(G).

3.4. Paths, Bridges and Necklaces.

3.5. Computational Experiments.

4. The Rural Postman Problem with Deadline Classes.

4.1. Overview.

4.2. Integer Programming Formulation.

4.3. Valid Inequalities from the RPP Subproblem.

4.4. Other Valid Inequalities.

4.5. Separation Algorithms.

4.6. Computational Experiments.

5. Literature on Multi-Vehicle Problems.

5.1. Overview.

5.2. Definitions.

5.3. Complete Formulations.

5.4. 'Sparse' Formulations.

5.5. 'Very Sparse' Formulations.

5.6. Other Formulations.

5.7. Combinatorial Lower Bounds.

5.8. The Bin Packing Problem.

Acknowledgements.

Thanks are due to Richard Eglese, my supervisor. He was invariably available whenever I needed him and he carefully read everything I gave to him. Just as importantly, he helped me to express complex concepts in readable English, thus preventing me from writing an utterly incomprehensible thesis.

I had several interesting discussions, mostly via e-mail, with Angel Corberán, at the University of Valencia, Spain. His work impressed and inspired me. Moreover, he pointed out some mistakes in an early draft of the thesis.

The secretaries in the Department of Management Science were always friendly and helpful, which mattered to me more than they perhaps realise.

My wife, Angela, has endured three years of me working long hours, not to mention occasional financial hardship, yet has been a tremendous support. I love her deeply.

Last, but most of all, thanks are due to God, the creator, sustainer and redeemer of the universe. I am certain that He blessed my intellectual efforts during the past three years. Therefore all honour and glory should go to Him.

1. Introduction.

1.1. Vehicle Routing Problems.

Vehicle Routing Problems (VRPs) arise when routes must be devised for one or more vehicles such that certain standards of efficiency are met and each route obeys one or more given restrictions. Examples of VRPs abound in the real world, since the transportation and distribution of both people and commodities form an integral part of modern life. Even simple VRP instances are often rather difficult to solve and, since large quantities of money are involved, Vehicle Routing has formed a fruitful area of study for Operational Researchers (see Golden & Assad, 1988; Ball et al., 1995b).

One strand of research into VRPs has been concerned with devising *optimisation algorithms*, i.e., procedures for finding an optimal solution and proving that it is optimal. In order for such an algorithm to be applicable, there must be a well-defined function assigning a cost to any given feasible solution. Typically, this function is a weighted sum of two costs; one proportional to the number of vehicles used in the solution and one proportional to the sum of the lengths of the routes ('length' here can mean distance, time, fuel consumption or whatever).

In any solution, a given vehicle is either used or not used and a given road is either traversed or not traversed by any given vehicle. The optimisation problem is therefore discrete or *combinatorial* in nature. Like many other Combinatorial Optimisation Problems or COPs, VRPs are almost always *NP-Hard*. This means (Garey & Johnson, 1979) that it is unlikely that any algorithm can be devised for them which is guaranteed, for any instance, to give the answer in a time which is polynomial in the size of the problem. Even the simplest kind of VRP, the famous Travelling Salesman Problem or TSP, is NP-Hard (see Lawler et al., 1985). The TSP represents the case where a single vehicle must visit all of the vertices of a complete, undirected network exactly once and *no* other complicating constraints are present.

The idea of NP-Hardness was first conceived at the end of the sixties (see the references in Garey & Johnson, 1979). It was an elegant explanation for the fact that nobody had yet devised an efficient algorithm for the TSP and related problems. Unfortunately, many misinterpreted the theory of NP-Hardness to mean that *no* large instance of an NP-Hard problem could ever be solved. This is quite untrue, as witnessed by the recent success of many researchers in solving several TSP instances which were previously regarded as unsolvable (e.g., Padberg & Rinaldi, 1991; Applegate et al., 1995). In fact, Applegate et al. have solved instances with up to 8000 vertices. The key to this rapid progress has been the *polyhedral* approach, which is outlined in the next section.

In this thesis, polyhedral results are given for four particular VRPs, each one of which is at least as useful as the TSP. All four are *Arc-Routing Problems* (ARPs), in which the vehicles are required to traverse certain edges (roads) of a network rather than visit certain vertices (customers). For two of these four cases, the resulting optimisation algorithms are described in detail.

1.2. The Polyhedral Approach.

The text by Nemhauser & Wolsey (1988) is a thorough introduction to almost all of the concepts described in this section.

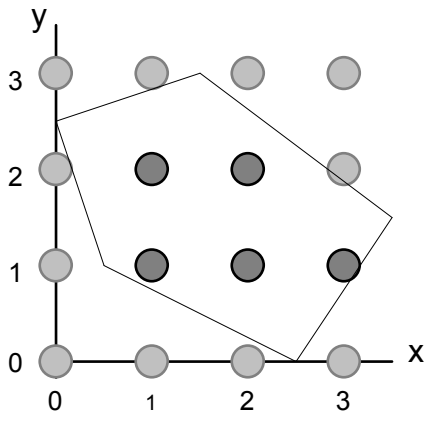
A Linear Programme (LP) is a problem of minimising a linear function of a set of variables subject to linear constraints. If, in addition, some or all of the variables are required to take integer values, the problem is called an Integer Programme or IP. It has been known for a long time (e.g., Dantzig, Fulkerson & Johnson, 1954), that COPs like the TSP can be formulated as IPs.

This knowledge by itself is not too useful since IPs are much harder to solve than LPs. In fact, the most effective known method for solving IPs involves the solution of a (possibly large) number of subsidiary LP relaxations. These LPs are embedded within the enumerative procedure known as *branch-and-bound*.

However, the task of solving a COP as an IP can be made considerably easier by seeking a *tight* formulation. A tight formulation is one in which the solution of the corresponding LP, with integrality conditions relaxed, is still integral or close to integral. A tight formulation is highly desirable since it leads to fewer LPs being solved in the branch-and-bound process.

The way to obtain a tight formulation is to note that the variables in the IP correspond to the dimensions of a Euclidean space. Each feasible solution to the problem corresponds to a point in this space and the convex hull of these points defines a polyhedron. An IP formulation which included all of the linear inequalities which induce *facets* of this polyhedron would be as tight as possible: The resulting LP relaxation would be completely integral and no branch-and-bound would be needed.

This principle is illustrated below. Fig. 1.1(a) displays the feasible region defined by the inequalities $6x - 4y \leq 15$, $2x + 4y \geq 5$, $6x + 2y \geq 5$, $2x - 6y \geq -15$ and $6x + 8y \leq 33$, whereas Fig. 1.1(b) displays the feasible region defined by the inequalities $x \geq 1$, $y \geq 1$, $y \leq 2$ and $x + y \leq 4$. Both regions contain the same five integer solutions, but the latter region is much smaller. This is because $x \geq 1$, etc. induce facets of the convex hull of feasible integer solutions.



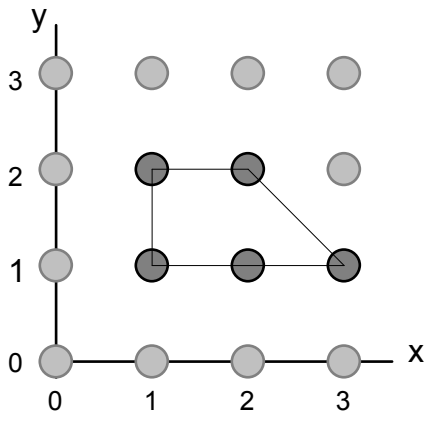


Fig. 1.1(a): A loose IP formulation. **Fig. 1.1(b) A tight IP formulation.**

In practice, however, there may be a vast number of known facet-inducing inequalities and a vast number of unknown ones. Therefore, the *cutting-plane approach* is taken: An initial LP is solved which contains only a tiny fraction of these inequalities. Then, auxiliary *separation routines* are invoked to generate further inequalities which are violated by the current LP solution. These are appended to the LP and the LP is resolved. If no more violated inequalities can be found, but the solution is still not integral, branch-and-bound commences.

In fact, as observed by Padberg & Rinaldi (1987), separation routines can be invoked at *every node* of the branch-and-bound tree, thus tightening the bounds at each stage. Since facet-inducing inequalities are valid *globally* (i.e. over the whole branch-and-bound tree), only one constraint matrix need be stored. This is in contrast to old-fashioned Gomory cuts, which were only valid at a particular node.

Some commercial mathematical programming packages now explicitly support this *branch-and-cut* method. For example, the MINTO package of Nemhauser & Savelsbergh (1994) contains a number of separation routine modules by default and allows the user to append others.

The study of IP formulations of COPs and the associated valid inequalities is known as *Polyhedral Combinatorics* and is developing rapidly. A survey of early work can be found in Lenstra (1985), see also the references in Nemhauser & Wolsey (1988). Relevent papers are reviewed in Chapters 2 and 5.

In this thesis, then, the polyhedral approach is applied to four Arc-Routing Problems. In each case, the author has considered formulations, valid inequalities, separation routines and/or solution algorithms.

1.3. Two Problems with a Single Vehicle.

The first single-vehicle problem under consideration is the *Rural Postman Problem* or RPP (Orloff, 1974). This is defined on a network G (such as a road network) with a set V of vertices (junctions) and a set E of edges (two-way roads). Each edge $e \in E$

has an associated cost c_e which is normally proportional to the length of the road. A subset of edges $R \subseteq E$ requires service. This means that the vehicle *must* traverse each $e \in R$ at least once; the non-required edges, if any, may be traversed any number of times or not at all. Moreover, the vehicle must start and end the route at the same place.

The task is to find a route of minimum cost. Note that the TSP can be transformed into an RPP by replacing each vertex by an infinitessimal required edge and adding a large constant to all other edge costs so that it is never optimal to visit any vertex more than once.

The second single-vehicle problem will be called the *Rural Postman Problem with Deadline Classes* or RPPDC. This is a generalisation of the RPP in which the set R is partitioned into *deadline classes* R^1, ..., R^L according to priority. For each $e \in E$, a time t_e is taken to traverse e *without* servicing it. Similarly, for each $e \in R$, a time s_e is taken to traverse e *while* servicing it. Edges in R^1 must be serviced no later than time T^1, edges in R^2 must be serviced no later than T^2, and so on. For reasons which will be made clear in chapter 4, it is assumed that $L < \ln |R|$.

Note that in the case of the RPPDC it is necessary to specify at which vertex of G the depot is located, since moving the depot can affect the optimal solution or even lead to there being no feasible solution at all.

The RPP is known to be NP-Hard (Lenstra & Rinnooy-Kan, 1976). Since the RPPDC reduces to the RPP when the deadlines become sufficiently large, the RPPDC is NP-Hard also. However, although both of these problems are NP-Hard, there is a sense in which the RPPDC is harder than the ordinary RPP, at least from a practical point of view: Given any particular RPP instance, it is trivial to find a *feasible* solution, even if it will almost certainly be suboptimal. On the other hand, given a particular RPPDC instance, it appears to be NP-Hard to even decide whether a feasible solution exists at all.

Even RPPDC instances in which $L = 1$ are substantially more difficult to solve than ordinary RPP instances, as will become apparent in Chapter 4.

1.4. Two Multi-Vehicle Problems.

The first multi-vehicle problem is the *Capacitated Arc Routing Problem* or CARP (Golden & Wong, 1981). The network G(V, E, R) is defined as in the RPP, but now each $e \in R$ has an associated *demand* q_e, which is assumed to be non-negative. A fleet of identical vehicles is available and the demand of the required edges serviced by any single vehicle cannot exceed Q, the vehicle capacity. A fixed cost of C is incurred for each vehicle used in the solution.

Note that the CARP, like the RPPDC defined in the previous section, is a generalisation of the RPP. It is possible to generalise in both ways at once to obtain the *Capacitated Arc Routing Problem with Deadline Classes* or CARPDC. Although this problem probably has wide applicability in practice, it proved to be too complex to tackle using existing polyhedral techniques. In this thesis, therefore, attention is restricted to the special case of the CARPDC in which L = 1. That is, each vehicle must finish servicing roads before a (single) time deadline T. This will be called the *Capacitated Arc Routing Problem with a Deadline*; it will be denoted by CARP+1D to emphasise the fact that only one deadline is involved.

1.5. Motivation for the Research.

As mentioned in Section 1.1, polyhedral algorithms now exist which are capable of solving many (but not all) large TSP instances to optimality, despite the fact that the TSP is NP-Hard. However, the TSP is rarely an adequate model in practice, because VRPs that occur in real-life may involve any of the following complications: multiple vehicles and/or depots; both pick up and delivery; certain roads requiring treatment; one-way streets; restrictions on the capacity of the vehicles, the length of any route or on the time at which a delivery occurs, etc. There is therefore a need for optimisation algorithms and lower bounding procedures for models of more general applicability.

Such algorithms are slowly beginning to emerge (see Chapters 2 and 5). One problem which has received attention is the *Vehicle Routing Problem with Time Windows* or VRPTW, which allows for vehicle capacity limitations and individual time windows within which each customer must be serviced (see Desrosiers et al., 1995). The best current VRPTW algorithm can solve some instances with up to 100 customers (Kohl & Madsen, 1995).

However, all known VRPTW algorithms fail when the time windows are wide. A time *deadline* is effectively a wide window and therefore VRPTW algorithms can be expected to fail on deadline problems (Nygard et al., 1988 and Thangiah et al., 1994 resort to heuristic approaches). Moreover, there is no optimisation literature on Arc-Routing with either time windows or deadlines, yet work by Eglese and Li (see Section 5.1) has indicated that deadlines are quite likely to occur in real-life ARPs.

For these reasons, then, it seemed that algorithms specifically tailored to ARPs with deadlines might be of use. This led to the decision to study deadline variants of the RPP and CARP. Preliminary attempts to formulate and solve some small single-vehicle problems led to the hope that that fairly large RPPDC instances might be solvable to optimality. Analogous attempts with multi-vehicle problems indicated that good lower bounds might be obtainable for the CARP+1D.

Clearly, a prerequisite for studying deadline variants of the RPP and CARP is a study of the RPP and CARP themselves. During the research, the author in fact found some new results on these problems and also on another problem related to the CARP called the Bin Packing Problem. These results are included in the thesis for the sake of completeness (see Chapters 3 and 6).

1.6. Outline of the Thesis.

i) A thorough survey of the literature on the RPP (and related single-vehicle problems) was conducted. This makes up Chapter 2.

ii) In Chapter 3, a known formulation of the RPP is considered. Large new classes of valid inequalities are introduced for the associated polyhedron, some of which are proved to induce facets. It is shown how to use some of these inequalities as cutting-planes and computational results are given.

iii) A formulation for the RPPDC is given in Chapter 4 which is efficient in that it exploits the natural sparsity of road networks. It is shown how to transfer known valid inequalities for the RPP over to the RPPDC; moreover, further inequalities are given which cannot be obtained in this way. The use of some of these as cutting-planes is given, along with computational results.

iv) A survey of the literature on the CARP (and related multi-vehicle problems) is given in Chapter 5.

v) In Chapter 6, some comments are made concerning the choice of which formulation to use when attempting to solve the CARP. Some new valid inequalities are given for some of these formulations. Also, a lower bound for a related problem, the Bin Packing Problem, is examined and from this it is possible to derive still more valid inequalities for the CARP formulations.

vi) A formulation and valid inequalities for the CARP+1D appear in Chapter 7. It is shown how to use some of these inequalities as cutting-planes to produce a very tight lower bound. The quality of this lower bound is tested on some examples for which good upper bounds are known.

vii) Conclusions and suggestions for further work are given in Chapter 8.

2. Literature on Single-Vehicle Problems.

2.1. Overview.

A vast number of articles and books have been published in the area of Vehicle Routing and it would require several hundred pages to review the field in any depth. In this thesis, therefore, the scope is restricted to *optimisation algorithms for undirected problems with a single depot*. That is, we will not be considering heuristic approaches, nor problems with directed arcs (i.e., one-way streets) or multiple depots.

The literature review is divided into two parts for the sake of convenience. *Single-vehicle problems* are considered in the present chapter, whereas *Multi-vehicle problems* will be considered in Chapter 5. For general surveys of Arc-Routing Problems, see Assad & Golden (1995) or Eiselt, Gendreau & Laporte (1995a, b).

2.2. The GRP and its Special Cases.

The RPP (see Section 1.3) is a special case of the so-called *General Routing Problem* (GRP). In the GRP, the set R may contain required vertices as well as required edges. That is, the vehicle must visit each required vertex at least once as well as traversing each required edge at least once. An instance of the RPP may therefore be regarded as a GRP instance in which $R \subseteq E$.

Both the RPP and GRP were first defined and named in a paper by Orloff (1974), in which he proposed a branch-and-bound algorithm for them. Unfortunately, the paper contained some errors as shown by Lenstra & Rinnooy-Kan (1976). These authors also showed that both the RPP and GRP are strongly NP-Hard.

It is known, however, that the special case of the RPP where R induces a connected subgraph is equivalent to a matching problem and is polynomially solvable (Edmonds & Johnson, 1973). It is called the *Chinese Postman Problem* after its discoverer, Mei-Gu Guan (Guan, 1962).

The other natural special case of the GRP, where $R \subseteq V$, has been termed the *Road Travelling Salesman Problem* (R-TSP) by Fleischmann (1985) and the *Steiner Graphical Travelling Salesman Problem* by Cornuéjols, Fonlupt and Naddef (1985). It will be referred to as the SGTSP. If in addition $R = V$ the *Graphical Travelling Salesman Problem* (GTSP) is obtained (Cornuéjols, Fonlupt & Naddef, 1985). Even the GTSP is strongly NP-Hard, as the standard TSP can be transformed to it.

The standard TSP differs from the GTSP in three ways: in the TSP, vertices must be visited *exactly* once, edges may be traversed *at most* once and the underlying network is *complete* (i.e., there is an edge connecting every pair of vertices). In the GTSP, vertices must be visited *at least* once, edges may be traversed *any number* of times and the network need only be *connected*.

Note also that the GRP, SGTSP and RPP correspond most naturally to real-life situations. Each of these can in fact be transformed to the standard TSP, but only at the price of introducing a large number of redundant variables.

2.3. Time Windows and Time Deadlines.

A *time window* is an interval of time during which a vertex (customer) or edge (road) requires service. Time windows are often present in real-life routing problems (see the survey by Desrosiers et al., 1995). To the author's knowledge, the only single-vehicle problem to receive attention in this area has been the *Travelling Salesman Problem with Time Windows* (TSPTW). The current best algorithm for this appears to be the dynamic programming implementation of Dumas et al. (1995). However, this only works well if the time windows are rather narrow.

Time deadlines (Nygard et al., 1988, Thangiah et al., 1994) are a special case of time windows; namely, the case in which the window starts at time zero. These also frequently occur in practice, for example in parcel delivery, road treatment, waste collection or the delivery of perishable goods. The dynamic programming approach fails in the case of deadlines since deadlines are effectively wide time windows.

The deadline classes described in Section 1.3 are an even more restricted form of time window, which nevertheless present serious difficulties for generic time window algorithms. The reason the author chose to study them is that they arise in practice whenever customers or roads are ordered in terms of priority. Several studies concerning real-life multi-vehicle problems with time deadlines are reviewed in Chapter 5. Of course, these problems contain the single-vehicle case as a subproblem.

Finally, there has been some research on problems with precedence constraints (PCs), i.e. conditions which state that vertex (or edge) i must be serviced before vertex (or edge) j. Balas, Fischetti & Pulleyblank (1995) consider the TSP with PCs and Dror, Stern & Trudeau (1987) and Gélinas (1992) consider the RPP with PCs. However, deadline classes differ from PCs in that there is nothing to prevent a low priority customer from being serviced before a high priority customer if time allows.

Some known polyhedral results are considered in the next three sections.

2.4. The TSP Polyhedron.

Like many COPs, the TSP can be formulated as an IP in a variety of ways. The classical formulation, due to Dantzig, Fulkerson & Johnson (1954), has a zero-one variable x_{ij} for each pair of vertices i and j, taking the value 1 if and only if i and j are adjacent on the tour. If there are N vertices, the formulation can be written as:

$$\textit{Minimise} \quad \sum_{1 \le i < j \le N} c_{ij} x_{ij}$$

Subject to:

$$x(\delta(\{i\})) = 2 \qquad (i = 1, \ldots, N) \qquad (2.1)$$

$$x(\delta(S)) \ge 2 \qquad (\forall S \subset V: 3 \le |S| \le N - 3) \qquad (2.2)$$

$$x_{ij} \in \{0, 1\} \qquad (1 \le i \le j \le N) \qquad (2.3)$$

where $\delta(S)$ represents the set of edges (commonly called the *cutset*) connecting vertices in S to vertices in V - S and, for any set of edges $F \subset E$, x(F) denotes $\sum_{ij \in F} x_{ij}$.

The *Degree* equations (2.1) ensure that the vehicle enters and leaves each vertex exactly once and the *Subtour Elimination* or *Connectivity* inequalities (2.2) ensure that the vehicle enters and leaves each set of vertices at least once.

For a TSP on N vertices, the convex hull of solutions to (2.1) - (2.3) is denoted by Q^N_T (Grötschel & Padberg, 1979). It is a polytope (i.e., a bounded polyhedron) and, due to the degree equations, it is not full-dimensional (that is, it lies within a subspace of the space it is defined in). A great deal of research has been conducted on the structure of Q^N_T and it would be impossible to review all of this here. Jünger, Reinelt & Rinaldi (1995) contains an excellent summary.

Because Q^N_T is not full-dimensional, a given valid inequality for Q^N_T can be expressed in a variety of ways by adding or subtracting multiples of the Degree equations (in this way, the upper bounds of 1 on each variable, implied by (2.3), can be regarded as Connectivity inequalities with $|S| = 2$).

It has been pointed out by Naddef (1990) that most (but not all) of the known valid inequalities for Q^N_T can be expressed in terms of the cutsets of various sets of vertices labelled *handles* and *teeth* (the meaning of these terms will become clear in what follows). In this thesis, we will call such inequalities *Handle-Tooth-Cutset* (HTC) inequalities. Figure 2.1 overleaf, adapted from Naddef (1990), displays every known class of HTC inequalities for Q^N_T at the time of writing. An arrow from one to another means that the former is a special case of the latter.

Comb inequalities (Grötschel & Padberg, 1979) are defined as follows: Let $q \geq 3$ be an odd integer and H, T_1, ..., T_q be sets of vertices such that $H \cap T_i$ and $T_i \setminus H$ are non-empty for $1 \leq i \leq q$ and such that the T_i are mutually disjoint. H is the *handle* of the comb and the T_i are the *teeth*. The associated comb inequality in HTC form is:

$$x(\delta(H)) + \sum_{j=1}^{q} x(\delta(T_j)) \geq 3q + 1 \qquad (2.4).$$

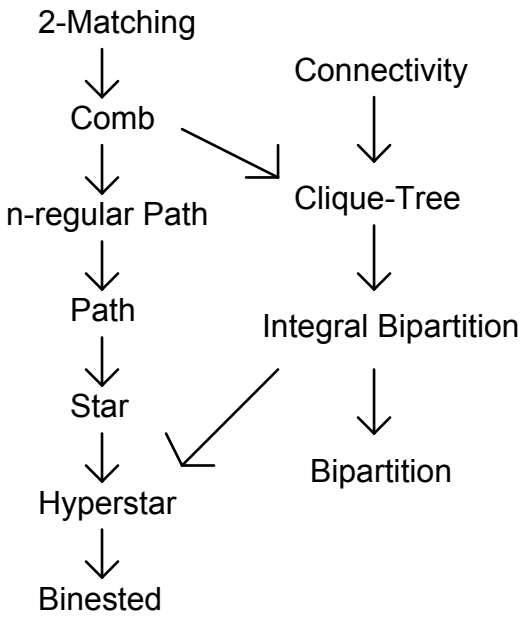


Fig. 2.1. The known classes of HTC inequality.

Figure 2.2 below shows a comb. Throughout this section, bold (respectively, plain) ellipses represent handles (resp., teeth). Small white (resp., black) circles represent sets of vertices which may (resp., may not) be empty. This follows the convention in Naddef (1990).

When each tooth contains only 2 vertices (i.e., each small black circle in fig. 2.2 corresponds to a single vertex), comb inequalities reduce to *2-Matching* inequalities, first discovered by Edmonds (1965).

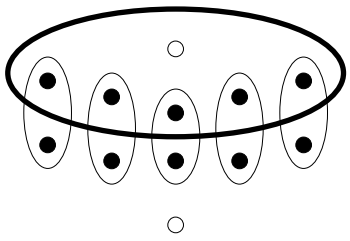


Fig. 2.2: A comb with five teeth.

Historically, the first generalisation of the comb inequalities are the *clique-tree* inequalities (Grötschel & Pulleyblank, 1986). These allow a number of comb configurations to be joined together in a tree-like structure (Fig. 2.3 below). The number of teeth intersecting any given handle must still be odd and at least 3 and each tooth must contain at least one vertex not contained in any handle. If there are p handles and q teeth, the clique tree inequality in HTC form is:

$$\sum_{i=1}^{p} x(\delta(H_i)) + \sum_{i=1}^{q} x(\delta(T_j)) \geq 2p + 3q - 1 \qquad (2.5)$$

As implied by Fig. 2.1, connectivity inequalities are also a kind of clique-tree inequality. In fact, they are those with p = 0 and q = 1.

Clique-trees have been generalised by Boyd & Cunningham (1991) to form *bipartitions*. Handles and teeth are allowed to connect together in any way, subject to the following conditions: all teeth must be disjoint, all handles must be disjoint, the number of teeth intersecting any handle must be odd and at least 3 and no tooth may be contained in any single handle.

Let d_j denote the number of handles intersecting tooth j. In bipartitions, unlike clique-trees, there may be teeth which do not contain any vertices outside of the d_j handles. Such teeth are called *degenerate*. Define $\beta_j = 1$ for non-degenerate teeth, but $\beta_j = d_j / (d_j - 1)$ for degenerate teeth.

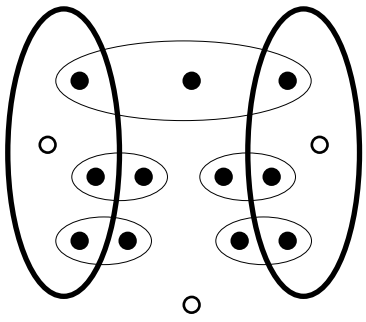


Fig. 2.3: A clique-tree.

The *bipartition* inequality, expressed in HTC form, is (see also Carr, 1995):

$$\sum_{i=1}^{p} x(\delta(H_i)) \;+\; \sum_{j=1}^{q} \beta_j x(\delta(T_j)) \;\geq\; p \;+\; \sum_{j \in ND} (d_j + 2) \;+\; \sum_{j \in D} (2\beta_j - 1)\, d_j \qquad (2.6)$$

where ND and D are the sets of non-degenerate and degenerate teeth, respectively.

When all β_j are integral, i.e., no degenerate teeth intersect three or more handles, the bipartition inequality is referred to as *integral*. Figure 2.4 below shows an integral bipartition with three degenerate teeth. For this instance, all cutsets have a coefficient of 1 apart from the three degenerate teeth on the periphery, which have coefficient 2. Moreover, the rhs is 26.

Path inequalities were first defined by Cornuéjols, Fonlupt & Naddef (1985). If $P \geq 3$ is an odd integer and n_i (i = 1,..., P) are integers greater than one, a *path configuration* is a partition of V into sets A, Z, V^i_j (i = 1, .., P, j = 1, .., n_i) such that the V^i_j are non-empty. Now, for all i, identify V^i_j with A when j = 0 and with Z when $j = n_i + 1$. The path inequality corresponding to the path configuration is:

$$\sum_{1 \leq i < j \leq N} \alpha_{ij}\, x_{ij} \;\geq\; 1 \;+\; \sum_{i=1}^{P} \frac{n_i + 1}{n_i - 1} \qquad (2.7)$$

where α_{ij} is defined as:

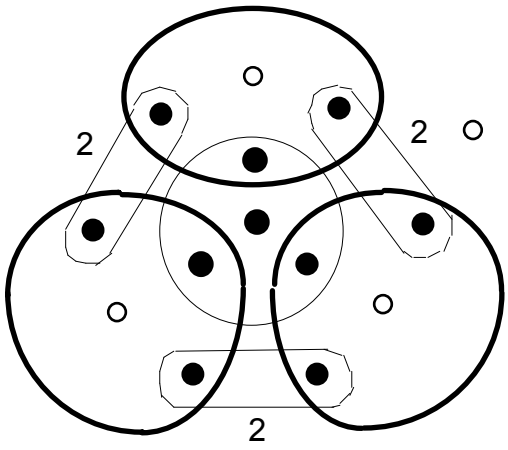


Fig. 2.4: An integral bipartition.

1 if $i \in A,\ j \in Z,$

$(q - p)/(n_r - 1)$ *if* $i \in V^r_p$ *and* $j \in V^r_q$ $(p < q)$, *either* $p \neq 0$ *or* $q \neq n_r + 1;$

$$1/(n_r - 1) + 1/(n_s - 1) + \left| \frac{p-1}{n_r - 1} - \frac{q-1}{n_s - 1} \right| \text{ if } i \in V^r_p,\ j \in V^s_q,\ r \neq s,\ 1 < p < n_r,\ 1 < q < n_s,$$

0 otherwise.

If all n_i are equal to the same number n, then Cornuéjols et al. refer to the path inequality as *n-regular*. In such a case, the inequality can be simplified by multiplying throughout by n - 1. The rhs then becomes P.n + P + n - 1. It can then be shown that the 2-regular path inequalities are precisely the comb inequalities.

A path configuration with P = 3, $n_1 = 3$ and $n_2 = n_3 = 2$ is shown in Fig. 2.5 below.

Although path inequalities do not immediately look like HTC inequalities, they can in fact be expressed in HTC form, as long as handles are permitted to lie inside other handles (Fleischmann, 1988). For example, the inequality corresponding to Fig. 2.5 can be multiplied by 2 and then expressed, as in Fig. 2.6 overleaf, as having two handles and three teeth. All cutsets then receive a coefficient of 1, apart from the two rightmost teeth, which receive a coefficient of 2. The rhs becomes 16.

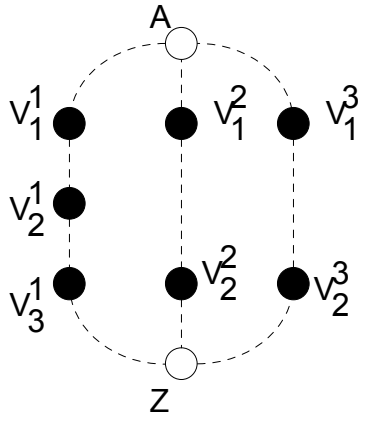


Fig. 2.5. Path configuration with 3 paths.

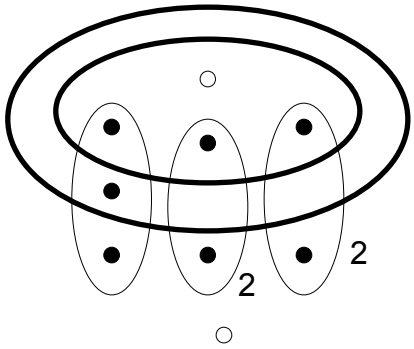


Fig. 2.6. Path configuration in HTC Form.

This idea of *nested* handles led Fleischmann (1988, 1987) to generalise the path inequalities to form the *star* and *hyperstar* inequalities. However, these will not be described here (see Naddef, 1990). Eventually, Naddef (1990, 1992) produced the *binested* inequalities in which, as the name implies, both handles and teeth are permitted to be nested. The conditions on the intersection of the handles and teeth, along with the rules for deriving the cutset coefficients, are extremely complex and will therefore not be described here.

A simple example of a binested set is given in Fig. 2.7 below. This has a single handle, but six teeth. In the corresponding binested inequality, each cutset has a coefficient of 1 and the rhs is 18.

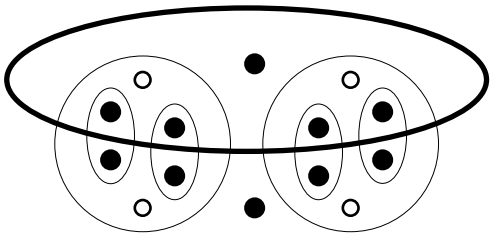


Fig. 2.7: A binested set; rhs = 18.

2.5. The GTSP and SGTSP Polyhedra.

The first paper published on the GTSP was that of Cornuéjols, Fonlupt & Naddef (1985), though a few of their results were simultaneously discovered by Fleischmann (see Fleischmann, 1985, 1988). Cornuéjols et al. associate a general integer variable x_e with each e ∈ E, representing the number of times e is traversed, and formulate the GTSP as follows:

$$\textit{Minimise} \quad \sum_{e \in E} c_e x_e$$

Subject to:

$$x(\delta(i)) \text{ is a positive even integer} \qquad (\forall i \in V) \qquad (2.8)$$

$$x(\delta(S)) \geq 2 \qquad (\forall S \subset V) \qquad (2.9)$$

$$x_e \geq 0 \text{ and integer} \qquad (\forall e \in E) \qquad (2.10)$$

For a given graph G, let **GTSP(G)** denote the convex hull in $\Re^{|E|}$ of solutions to (2.8) - (2.10). Notice that **GTSP(G)** is always a full-dimensional, unbounded polyhedron, which makes it easier to study than Q^N_T. Yet, from a comparison of (2.1) - (2.3) with (2.8) - (2.10), it can be seen that Q^N_T is a face of **GTSP(K_N)**, where K_N is the complete graph on N vertices. Hence, new facets of **GTSP(K_N)** frequently lead to new facets of Q^N_T.

It was these observations which led Cornuéjols et al. to discover the path inequalities. They showed that the path inequalities are valid for **GTSP(G)** as well as for Q^N_T.

It has been shown (see Naddef, 1990, Jünger, Reinelt & Rinaldi, 1995 and the references therein), that *all HTC inequalities for Q^N_T are also valid for **GTSP(G)***. Of course, they will not necessarily induce facets. Indeed, the sparser the graph G is (i.e., the fewer edges it has), the less likely it is for a HTC inequality to induce a facet.

For example, when Cornuéjols, Fonlupt and Naddef (1985) define path inequalities for the GTSP, they require the subgraphs induced in G by each of the sets V^i_j to be connected and, moreover, that there be an edge in G connecting V^i_j to V^i_{j+1} for i = 1, ..., P and j = 1, ..., n_i - 1. Path inequalities do not induce facets of **GTSP(G)** if these conditions are not met.

The SGTSP polyhedron has received little attention. Cornuéjols, Fonlupt & Naddef (1985) only consider analogues of the connectivity inequalities (2.9): These remain valid provided that $| S \cap R | \geq 1$ and $| (V \setminus S) \cap R | \geq 1$. Fleischmann (1985) describes what he calls "3-star" inequalities. In the terminology of Cornuéjols et al. (1985), the 3-star inequalities are in fact 2-regular path inequalities with P = 3. These inequalities remain valid provided that $| V^i_j \cap R | \geq 1$ for all i and j.

In the next section, attention is turned to known polyhedral results for the two problems in which required edges are permitted; namely, the RPP and GRP.

2.6. The RPP and GRP Polyhedra.

In this section some more notation will be needed: recall that a RPP instance is given by a graph G(V, E, R), a set of edge costs $c_e \geq 0$ for each $e \in E$ and a set $R \subseteq E$ of required edges. We let R(S) and $\delta_R(S)$ denote the set of required edges with both end-vertices in S, or one end-vertex in S, respectively.

Corberán & Sanchis (1994) formulate the RPP in the following way: associate a general integer variable x_e with each $e \in E$ representing the number of times e is traversed (if $e \notin R$), or one less than this number (if $e \in R$). Then we have:

$$\textit{Minimise} \quad \sum_{e \in E} c_e x_e$$

Subject to:

$$x(\delta(S)) \geq 2 \qquad (\forall\, S \subset V:\ \delta_R(S) = \varnothing,\ R(S) \neq \varnothing,\ R(S) \neq R) \qquad (2.11)$$

$$x(\delta(i)) \equiv |\delta_R(i)| \bmod 2 \qquad (\forall\, i \in V) \qquad (2.12)$$

$$x_e \geq 0 \text{ and integer} \qquad (\forall\, e \in E) \qquad (2.13)$$

Now, for a given graph G, let **RPP(G)** denote the convex hull in $\Re^{|E|}$ of solutions to (2.11) - (2.13). It is a full-dimensional, unbounded polyhedron. Corberán & Sanchis show that the *connectivity* inequalities (2.11) and the non-negativity conditions induce facets of **RPP(G)** under very mild conditions. However, because of the non-linear degree constraints (2.12) and the integrality conditions, more inequalities are required to define **RPP(G)**.

Observe that a vehicle must leave a set $S \subset V$ as many times as it enters it and therefore that $|\delta_R(S)| + x(\delta(S))$ must be even. This leads to the *R-odd cut* inequalities:

$$x(\delta(S)) \geq 1 \qquad (\forall\, S \subset V:\ |\delta_R(S)| \text{ odd}) \qquad (2.14)$$

Two other polyhedral results are presented in Corberán & Sanchis (1994). The first of these is the following: Suppose that the set of vertices V is partitioned into subsets in such a way that each subset contains at least one required edge and no required edge crosses between subsets. Now, if each subset is shrunk into a single vertex, one can define a GTSP instance on the resulting network G'. It can be shown that every facet-inducing inequality for **GTSP(G')** is also facet-inducing for **RPP(G)**. Note that the connectivity inequalities are a special case of this. The procedure also leads to path, star, etc., inequalities for **RPP(G)**.

The second result is that the following *K-component* (K-C) constraints are facet-inducing: Let k be an integer greater than one. Consider a partition of V into the non-empty sets V_j (j = 0, ..., k+1) such that:

- each induced subgraph $G(V_j)$ is connected;

- each required edge either lies within some $G(V_j)$ or crosses from V_0 to V_{k+1} ;

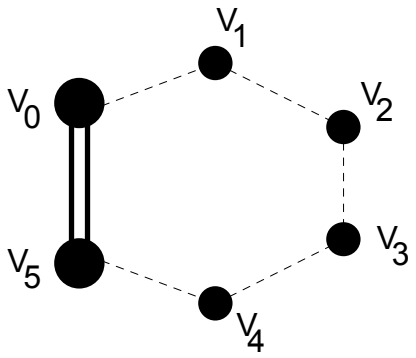


Fig. 2.8: K-C structure.

- at least one edge in E connects $G(V_j)$ to $G(V_{j+1})$, for $j = 0,.., k$;

- there are an even number of required edges crossing from V_0 to V_{k+1}.

A diagram of the resulting *K-C configuration* is shown in Figure 2.8 above. Letting (p:q) denote the set of edges with one end-vertex in each of V_p and V_q, the corresponding K-C inequality takes the form:

$$(k-1)\ x(0{:}k+1) + \sum_{\substack{0 \le p \le q \le k+1, \\ p \ge 1 \text{ or } q \le k}} (q-p)\ x(p{:}q) \ \ge \ 2k \qquad (2.15)$$

Now to consider the GRP. The recent paper by Corberán & Sanchis (1996) extends the above results on the RPP to the GRP and also significantly generalises them. They show that the formulation (2.11) - (2.13) transfers to the GRP provided that R(S) is interpreted as containing the required vertices in S as well as the required edges having both end-vertices within S. The validity of the R-Odd, K-C and GTSP-type inequalities then transfers directly to the GRP.

Corberán & Sanchis (1996) then present the *honeycomb* inequalities, valid for the RPP as well as the GRP. It would take up too much space to formally define these here. Instead, we simply note that honeycomb configurations can be formed by

'gluing' K-C configurations together by identifying edges. The gluing must be done in such a way that certain conditions are met, such as the condition that some of the non-required edges form a spanning tree.

K-C configurations themselves can be regarded as very simple honeycomb configurations; namely, those in which the spanning tree is a mere path.

Three example honeycomb configurations are shown in Fig. 2.9 below. In the corresponding inequalities, the edges shown have a coefficient of 1 unless otherwise indicated. The coefficient of any edge not shown is equal to the number of edges traversed in the spanning tree to get from one end-vertex of the edge to the other.

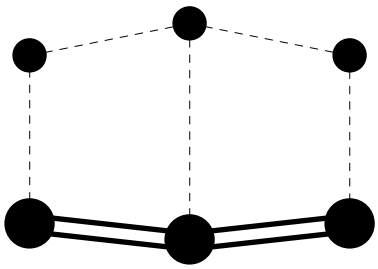


Fig. 2.9a: First honeycomb, rhs = 6.

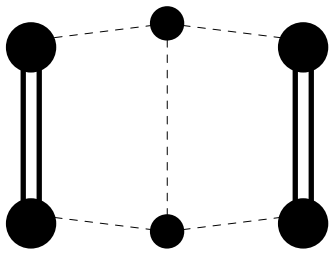


Fig. 2.9b: Second honeycomb, rhs = 6.

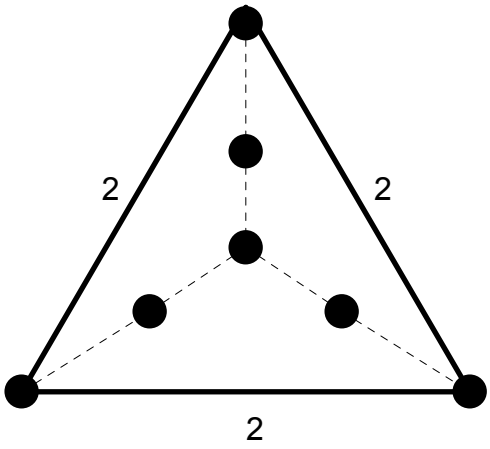


Fig. 2.9c: Third honeycomb, rhs = 8.

2.7. Separation Algorithms for the TSP.

A separation algorithm is a routine which takes an LP relaxation as input and outputs one or more violated inequalities in a given class (e.g. 2-matching, comb, etc.), if any exist. In the case of the TSP, the input will typically be a weighted *support graph* G*(V, E*), where e $\in$ E* if and only if the variable associated with that edge is positive, and the weight of e is equal to the current value of x_e. An algorithm which guarantees to find a violated inequality in a given class (if one exists), is called *exact*, otherwise it is called *heuristic*. It is also desirable for such an algorithm to run in polynomial time.

At present, three exact polynomial separation algorithms are known:

(i) A Connectivity inequality (2.2) is violated if and only if a minimum cut in G* takes a value less than 2. If the minimum cut algorithm of Nagamochi, Ono & Ibaraki (1994) is used, this means that separation for (2.2) can be performed in time O(NM + N^2 log N), where M is the number of edges in G*.

(ii) Padberg & Rao (1982) show how to construct an auxiliary graph G', by splitting edges of G* and labelling certain vertices odd, with the property that a 2-matching inequality is violated if and only if a minimum odd cut in G' has a weight less than 1. They also show that a minimum odd cut can be found in polynomial time by invoking a maximum flow routine at most N_{ODD} times, where N_{ODD} is the number of odd vertices.

(iii) Carr (1995, 1996) has shown that bipartition and binested inequalities with a fixed number of handles and teeth can be separated in polynomial time. However, the order of the polynomial in Carr's algorithm increases rapidly as the number of handles and teeth increases. At present, the only practical use of the algorithm is for the identification of comb inequalities with three teeth.

We now consider heuristic separation routines. Padberg & Rinaldi (1990) present fast separation heuristics for connectivity, 2-matching, comb and clique-tree inequalities. The comb heuristic relies mainly on the idea of shrinking sets of vertices in G*, in such a way that a violated 2-matching inequality in the shrunk graph leads to a violated comb inequality in G* (*shrinking* a set S of vertices means identifying all vertices in S, eliminating any consequent loops, and merging each consequent set of parallel edges into a single edge, summing the weights in the process). This comb heuristic is extended in Clochard & Naddef (1993) and Naddef & Clochard (1994). In these papers, not only is the comb heuristic made more effective, but it is extended to find violated path inequalities as well.

Note that if all connectivity inequalities are satisfied, then there is a limit on the amount by which a comb inequality can be violated. It is not hard to show that the maximum violation occurs when $x(\delta(T_j \cap H)) = x(\delta(T_j \setminus H)) = x(\delta(T_j)) = 2$ for all teeth, when the lhs in (2.4) is 3q. Such *maximally violated* combs often arise in LP relaxations. This led Applegate et al. (1995) to concentrate on producing a separation algorithm for this extremal case.

Applegate et al. call a set of vertices S *tight* if $x(\delta(S)) = 2$ in G*. They show that tight sets can be conveniently represented by O(N) *necklaces*. A necklace is a partition of V into vertex sets $S^1, ..., S^m$, called *beads*, such that each bead is tight, the union of any consecutive sequence of beads is tight, and $S^m \cup S^1$ is also tight. The union of a pair of consecutive beads is called a *domino*.

Fig. 2.10, overleaf, shows a possible G* arising within a branch-and-cut procedure. The solid lines represent edges with $x_e = 1$ and the dotted lines represent edges with $x_e = 1/2$. One necklace is given by the beads {1}, {2} and {3, ..., 16}. Since there are only 3 beads in this necklace, each domino is the complement of a single bead and therefore tight. A second necklace is given by the beads {1, 2}, {3}, {4} and {5, ..., 16}. Note that the dominoes {1, 2, 3}, {3, 4}, {4, ..., 16} and {1, 2, 5, ..., 16} are tight, as required.

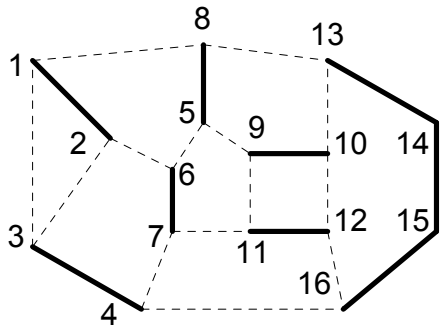


Fig. 2.10: Possible G*.

Since each tooth of a maximally violated comb satisfies $x(\delta(T_j \cap H)) = x(\delta(T_j \setminus H)) = x(\delta(T_j)) = 2$, each domino is a potential T_j and the two beads making up the domino are candidates for $T_j \cap H$ and $T_j \setminus H$. Applegate et al. show that, if there is a maximally violated comb in G* at all, then there is one which uses at most one domino from each necklace. They associate a {0, 1} variable with each necklace, taking the value 1 if some domino in that necklace is to be a tooth. They then set up a system of linear congruences modulo 2 in these variables and show that the existence of a maximally violated comb in G* is equivalent to the existence of a solution to these congruences with a particular property.

Applegate et al. do not give a polynomial-time algorithm for finding suitable solutions to the congruences. Instead, they resort to a (sophisticated) heuristic.

Fleischer & Tardos (1996) note that the set of necklaces can be obtained in O(N.M.logN) time, using an algorithm of Benczúr (1994). However, they also note that the remainder of the necklace algorithm has not been shown to run in polynomial time in the worst case. They show, however, that when G* is a planar graph, the whole algorithm can be made to run in polynomial time. This is because a necklace can be represented by a series of parallel edges running from one face of G* to another. Fig. 2.11, overleaf, illustrates this for the necklace with beads {1, 2}, {3}, {4} and {5, ..., 16} discussed above.

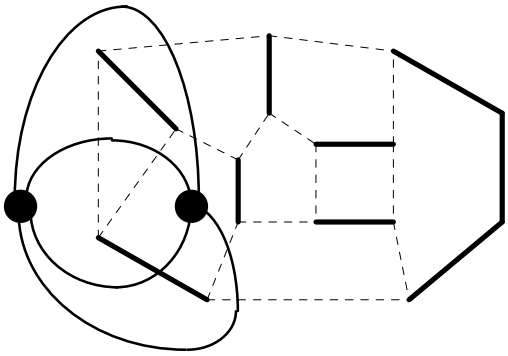


Fig. 2.11: Representing a planar necklace.

Construct an auxiliary graph G', with a vertex corresponding to each face of G*, and one of the parallel edges for each necklace. It is possible for G' to not be planar even though G* is planar. Fleischer & Tardos (1996) show that if a comb inequality is maximally violated in G*, then G' contains an odd cycle. The converse is false, but they show that if G' contains an odd cycle, then one or both of the following conditions holds:

(i) At least one maximally violated comb inequality exists.

(ii) At least two violated comb inequalities exist which are not maximally violated.

Since there are only O(N) necklaces, G' contains O(N) edges. Therefore an odd cycle in G' can be detected in O(N) time by breadth-first search. This time is far outweighed by the O(N.M.logN) time taken to list the necklaces initially. Finally, it can be shown that the whole approach takes $O(N^2.\log N)$ time, since M is O(N) for planar graphs.

Finally, Christoff & Reinelt (1996) have very recently given complete linear descriptions of Q^N_T for $n \leq 10$, using a computer program. They are also working on producing heuristic separation routines for some of the newly discovered classes of facets.

2.8. Other Separation Algorithms.

The success of polyhedral theory in tackling large instances of the TSP has not yet been duplicated with the GTSP, SGTSP, RPP or GRP. However, this is probably merely due to the fact that fewer researchers pay attention to these problems. The only papers known to the author in which computational results are presented are those of Fleischmann (1985) and Corberán & Sanchis (1994).

As mentioned in Section 2.5, Fleischmann (1985) concerns the SGTSP, although the problem is termed the R-TSP in that paper. Fleischmann gives exact and heuristic polynomial separation routines for the connectivity inequalities and gives a separation heuristic for the 3-star inequalities. Instances with up to 200 vertices were solved to optimality with a combination of connectivity and 3-star inequalities, traditional Gomory cuts, and branch-and-bound.

As mentioned in Section 2.6, Corberán & Sanchis (1994) concerns the RPP. Violated connectivity, R-odd cut and K-C inequalities were identified by eye. RPP instances with up to 184 edges were solved to optimality without the use of branch-and-bound.

Finally, Corberán (1996) gives the following theoretical results for the GRP:

i) The separation problem for connectivity inequalities can be solved exactly in polynomial time: First, give each $e \in E$ in G a weight equal to x_e. Shrink all required edges in G to yield a smaller graph G'. Call the new vertices thus created *special*. Each special vertex corresponds to a cluster of required edges in G. Pick one special vertex and send maximum flows in G' to each one of the other special vertices. If a Connectivity inequality is violated, at least one of these maximum flows will have a value less than 2.

ii) The separation problem for R-odd cut inequalities can also be solved exactly in polynomial time: Give each $e \in E$ in G a weight equal to x_e, label vertex i odd if and only if $|\delta_R(i)|$ is odd, then find a minimum weight odd cut (see Padberg & Rao, 1982).

Corberán also claims to have fast and effective separation heuristics for connectivity, R-odd cut, K-C and honeycomb inequalities.

In the following chapter, many new classes of valid inequality will be given for the RPP, along with an effective separation heuristic for one of these classes.

3. The Rural Postman Problem.

3.1. Overview.

In this chapter, the polyhedral approach to the RPP is extended, both theoretically and computationally. In Section 3.2, it is shown how to "borrow" most of the polyhedral results known for the GTSP (Section 2.5), first to give new inequalities for the SGTSP and then to give new inequalities for the RPP. This leads to an extremely detailed (though not complete) description of **RPP(G)**. In Section 3.3, some of the inequalities thus obtained are discussed in detail. They are shown to generalise most of the inequalities in Corberán & Sanchis (1994, 1996). In Section 3.4, the *path-bridge* inequalities are defined and the necklace method of Fleischer & Tardos (1996) (see Section 2.7) is adapted to yield a good separation heuristic for them. The chapter concludes with some computational experiments and comments in Section 3.5.

3.2. The GTSP, SGTSP and RPP Polyhedra.

In Section 2.5, it was stated that all of the HTC inequalities discussed in Section 2.4 are valid for the GTSP as well as the standard TSP. Each class of HTC inequalities is defined in terms of the way in which the handles and teeth are permitted to intersect and also whether certain vertex sets are permitted to be empty. Naddef (1990, 1992) notes that, given a set of handles and teeth, one can define an equivalence relation on the vertices, putting two vertices in the same class if they belong to the same handles and teeth. Naddef calls these classes *HT-classes*.

When viewed in this way, the small black circles in Figs. 2.2 - 2.7 (pp. 13-17) represent HT-classes which are not permitted to be empty and the small white circles represent HT-classes which may be empty without affecting the validity of the inequality. We will call the latter kind of HT-class *emptiable*.

Armed with these definitions, the first theorem in this thesis can be stated:

*Theorem 3.1: All HTC-inequalities are valid for **SGTSP(G)**, exactly as defined for **GTSP(G)**, provided that every non-emptiable HT-class contains at least one required vertex.*

*Proof: Note that, if the theorem is true for complete graphs G, then it is true for all graphs. This is because removing edges of G can only cause a shrinking of **SGTSP(G)**. Thus we assume that G is complete. Recall that, by definition, the lhs of an HTC inequality is equal to the weighted sum of a number of cutsets. This implies that the coefficients of the edges obey the triangle inequality.*

Suppose that a SGTSP tour exists that violates a suitable HTC inequality. This tour can be written as a circular sequence of vertices, possibly with repetitions, representing the order in which the vertices are visited. Now consider the new SGTSP tour formed by taking "shortcuts"; that is, by observing the same sequence but omitting the visits to non-required vertices. By the triangle inequality, this must also violate the HTC inequality. Yet this is impossible, since this would imply that the HTC inequality is not valid for the GTSP.

The following result is also easy to obtain:

*Lemma 3.2: An HTC inequality which is valid for **SGTSP(G)** induces a facet of **SGTSP(G)** if it induces a facet of **GTSP(G)**.*

*Proof: Immediate from the fact that the SGTSP on a given graph is a relaxation of the GTSP on the same graph and therefore that **SGTSP(G)** contains **GTSP(G)**.*

This means that all of the classes of inequality in Fig. 2.1 (p. 13) have SGTSP analogues. This significantly extends the observation by Fleischmann (1985) that the star inequalities are valid for **SGTSP(G)**.

It will now be shown that the RPP and SGTSP polyhedra are also strongly related, by showing how to transform any RPP instance into a SGTSP instance with only a minor modification of G and the cost function. The transformation is of interest in its own right, since it preserves useful properties, such as sparsity or planarity, that G might have. However, its main use will be to provide new valid inequalities for **RPP(G)**. The transformation involves doing the following for each required edge e = {u, v} (that is, u and v are the end-vertices of e):

- add two new *required vertices* (u' and v', say).
- add three new *edges* {u, u'}, {u', v'} and {v', v}.
- give these new edges a very large cost.
- make e non-required.

We will call the new vertices (resp., edges) *fixing vertices (edges)* and denote by G' the transformed network. Fig. 3.1 overleaf illustrates the transformation. Note that G' has |E| + 3 |R| edges.

Lemma 3.3: A solution vector for the SGTSP produces a solution vector to the original RPP when the variables associated with the fixing edges are dropped.

Proof: For any required edge e in G, consider the new vertices u' and v' in G'. The following three connectivity inequalities are valid for the SGTSP instance:

$$x\{u, u'\} + x\{u', v'\} \geq 2, \quad x\{u, u'\} + x\{v', v\} \geq 2, \quad x\{u', v'\} + x\{v', v\} \geq 2.$$

Since the three fixing edges have all been given a very large cost, these inequalities will be satisfied at equality in any optimal SGTSP solution if possible. Solving the simultaneous equations yields x{u, u'} = x{u', v'} = x{v', v} = 1. Hence, the path u - u' - v' - v plays the role of a required edge and the result follows.

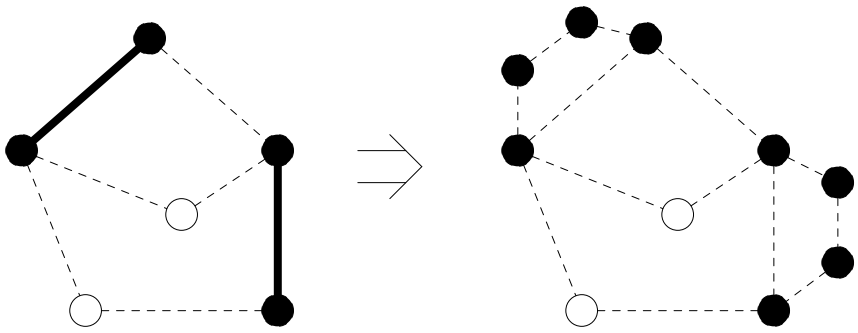


Fig. 3.1: Transforming an RPP into a SGTSP.

A possible SGTSP solution for the graph G' of Fig. 3.1 is shown in Fig. 3.2a. The corresponding RPP solution is shown in Fig. 3.2b. It can be seen that this is a solution to the RPP of Fig. 3.1. In general, there is a precise one-to-one correspondence between RPP solutions and appropriate SGTSP solutions.

Now let **SGTSP(G')** denote the convex hull in $\Re^{|E|+3|R|}$ of all feasible SGTSP tours in G'. Consider the face of **SGTSP(G')** containing all tours in which the connectivity inequalities described in Lemma 3.3 hold as equalities (that is, all variables corresponding to fixing edges take the value 1). Call this face **P***. The one-to-one correspondence between SGTSP tours in G' and GRP tours in G immediately implies the following theorem:

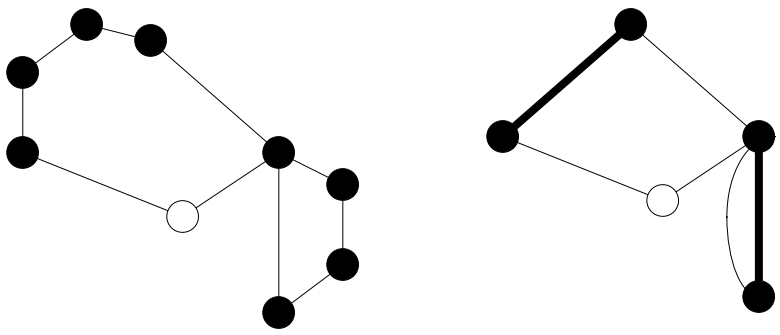


Fig. 3.2a: SGTSP solution in G'. **3.2b: RPP solution in G.**

Theorem 3.4: ***RPP(G)*** *is equal to the projection of* ***P**** *onto the |E| dimensional subspace defined by the original (i.e., non-fixing) edges.*

Elementary properties of polyhedra yield the following result:

Corollary 3.5: Any facet-inducing inequality for ***SGTSP(G')*** *can be reduced, by elimination of the variables associated with the fixing edges, to a valid (not necessarily facet-inducing) inequality for* ***RPP(G)****. Moreover, any facet-inducing inequality for* ***RPP(G)*** *can be obtained in this way.*

These results imply that each class of HTC inequality in Fig. 2.1 has a generalisation in terms of the RPP. The author terms these *generalised bipartition*, *generalised binested*, etc., inequalities. These inequalities are discussed in the next section.

3.3. HTC Inequalities for RPP(G).

In this section, the generalised versions of many of the classes of inequality listed in Fig. 2.1 are discussed in detail. It is assumed throughout that the reader understands the transformation from G to G', along with the idea of eliminating variables associated with fixing edges, as outlined in the previous section.

First consider the inequalities of Corberán & Sanchis (1994):

Lemma 3.6: Connectivity inequalities (2.11) are generalised connectivity inequalities.

Proof: This is the trivial case in which no fixing edges lie in the cut in G'.

Lemma 3.7: R-odd cut inequalities (2.14) with $|\delta_R(S)| = 1$ *are generalised connectivity inequalities.*

Proof: If S is such that $|\delta_R(S)| = 1$*, then* $\delta(S)$ *in the transformed graph G′ contains exactly one fixing edge. Since the variable associated with this edge receives a coefficient of 1 in the connectivity inequality, the rhs drops by 1 when it is eliminated.*

Remark 3.8: Any generalised connectivity inequality not covered by Lemmata 3.6 and 3.7 is redundant for ***RPP(G)****: If there are two or more fixing edges in a cut in G′, the rhs drops to zero or less when variables are eliminated.*

Theorem 3.9: R-odd cut inequalities (2.14) in which $|\delta_R(S)| \geq 3$ *are generalised 2-matching inequalities.*

Proof: Each of the $|\delta_R(S)|$ *required edges in an R-odd cut yields a pair of fixing vertices in G′. One vertex in the pair is connected to S by a fixing edge; the other vertex in the pair is connected to V - S by a fixing edge. We say that the former type is "close" to S. Define a 2-matching inequality on G′ as follows: let the handle be the union of S and the fixing vertices which are close to S; let each pair of fixing vertices be a tooth. The rhs of this 2-matching inequality is* $3\ |\delta_R(S)| + 1$*. Each original (i.e., non-fixing) edge receives the same coefficient in the 2-matching inequality as it did in the original R-odd cut inequality. However, the variables associated with the three fixing edges in each new path also appear on the lhs of the 2-matching inequality, each with a coefficient of 1. Since there are* $3\ |\delta_R(S)|$ *such edges, the rhs of the 2-matching inequality drops down to 1 when eliminating the variables associated with the fixing edges. In this way, the original R-odd cut inequality is obtained.*

Theorem 3.10: K-C inequalities (2.15) are generalised path inequalities.

Proof: Recall (Section 2.6) that a K-C configuration on G is defined by sets of vertices $V_0, \ldots, V_{k+1}$*. Suppose there are B required edges crossing from* V_0 *to* V_{k+1} *(B must be even). Then a path configuration can be defined on G′ as follows (see Fig.*

3.3 below): Let $P = B + 1$ and let $n_i = 2$ for $i = 1, ..., B$, $n_i = k$ for $i = B + 1$. Let $A = V_0$, $Z = V_{k+1}$, $V^P_j = V_j$ for $j = 1, ..., k$. Finally, for $i = 1, ..., B$, let V^i_1 be the fixing vertex of the ith crossing required edge which is close to A and let V^i_2 be the fixing vertex of the ith crossing required edge which is close to Z.

The rhs of this path inequality, see (2.7), is $1 + 3B + (k + 1)/(k - 1)$. An edge in G' going from V^P_s to V^P_t $(0 \le s < t \le k + 1)$ receives a coefficient of $(t - s) / (k - 1)$ in the path inequality, unless $s = 0$ and $t = k + 1$, when it recieves a coefficient of 1.

Now, corresponding to each of the B crossing required edges in G there is a path of three fixing edges in G', each with a coefficient of 1 in the path inequality. Hence, when eliminating variables, the rhs drops to $1 + (k + 1)/(k - 1) = 2k / (k - 1)$.

We have established that the generalised path inequality gives a coefficient of $(t - s) / (k - 1)$ to an edge in G going from V_s to V_t $(0 \le s < t \le k + 1)$ unless $s = 0$ and $t = k+1$, and that the rhs is $2k / (k - 1)$. Multiplying throughout by $k - 1$ yields the original K-C inequality (2.15).

As we shall see, there are useful, non-redundant generalised path inequalities which are neither K-C inequalities nor ordinary path inequalities. Before presenting these, however, the honeycomb inequalities of Corberán & Sanchis (1996) are discussed.

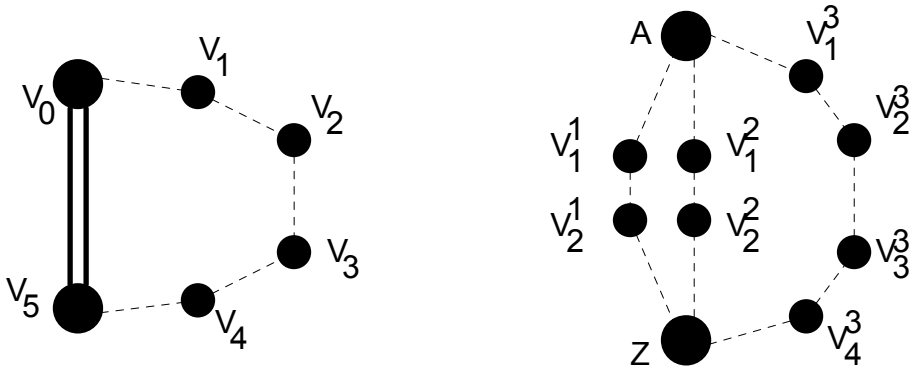


Fig. 3.3: (a) K-C configuration in G. **(b) Path configuration in G'.**

It is obvious that there are honeycomb inequalities of HTC type, since K-C inequalities are honeycomb inequalities (Section 2.6). Some other honeycomb inequalities, not of the K-C type, can also be expressed in HTC form: It is not hard to show that the honeycomb inequality of Fig. 2.9a is a *generalised clique-tree* inequality (cf. Fig. 2.3), the honeycomb inequality in Fig. 2.9b is a *generalised binested* inequality (cf. Fig 2.7) and the honeycomb inequality in Fig. 2.9c is a *generalised integral bipartition* inequality (cf. Fig. 2.4).

The author has been unable, however, to prove that *all* honeycomb inequalities are derivable in this way. In fact, it is conjectured that they cannot. Just as there are facets of the GTSP which are not of the HTC type (Jünger, Reinelt & Rinaldi, 1995), it may be that some honeycomb inequalities are not of HTC type either.

Some hitherto unknown inequalities, which can also be derived by the transformation from G to G', are now examined. Consider the simple RPP instance in Fig 3.4a below. When this is formulated as in section 2.6, and violated connectivity and K-C inequalities have been appended, the LP relaxation shown in Fig. 3.4b is obtained with cost 6. Although it is has integral x values, it violates constraints (2.12) (there are 'odd' vertices). Furthermore, no R-odd cut, honeycomb or GTSP-type inequalities are violated. Thus, the cutting-planes of Corberán & Sanchis (1994, 1996) cannot ensure that a valid tour is obtained, even if integrality is achieved by branch-and-bound.

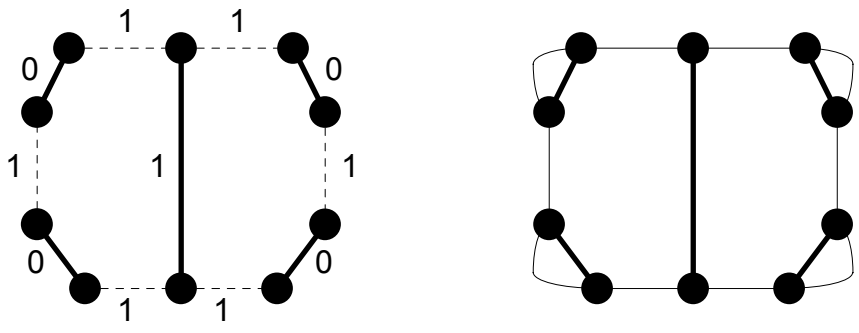


Fig. 3.4 (a) RPP instance, c_e shown. **(b) an LP relaxation.**

Now consider the graph G' for this RPP instance (Fig. 3.5a below). We can construct an LP relaxation of the SGTSP instance on G' (Fig. 3.5b), which corresponds to the LP relaxation of the RPP in Fig. 3.4b. This new LP relaxation violates a comb inequality with 3 teeth and a rhs of 10. Three fixing edges have a coefficient of 1 in this inequality; the other fixing edges have a coefficient of 0. By eliminating the corresponding variables, one obtains a *generalised comb* inequality, valid for **RPP(G)**, with a rhs of 7. This inequality is in fact precisely

$$\sum_{e \in E} c_e x_e \geq 7 \qquad (3.1).$$

It turns out that (3.1) induces a facet of **RPP(G)**, as can be shown by enumerating affinely independent tours. Moreover, the addition of (3.1) to the LP is sufficient to obtain an optimal solution to the RPP instance, which consequently has a cost of 7. Thus, the generalised comb inequalities differ from all other known inequalities, and at least some of them can cut off integral LP relaxations with 'odd' vertices. This also means that the *generalised path* inequalities properly contain the path, comb, K-C and R-odd cut inequalities.

However, there are still other RPP instances, such that the polyhedron defined by all GTSP-type, generalised path and honeycomb inequalities still has invalid integral extreme points with 'odd' vertices. The simplest example known to the author is shown in Fig. 3.6 overleaf.

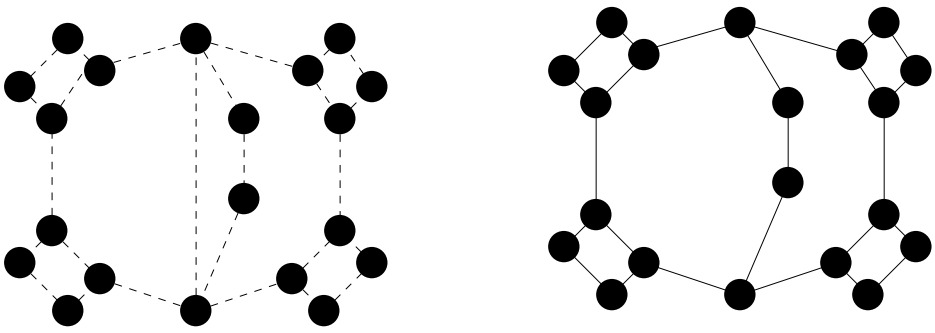


Fig. 3.5(a) corresponding G'. **(b) corresponding LP relaxation.**

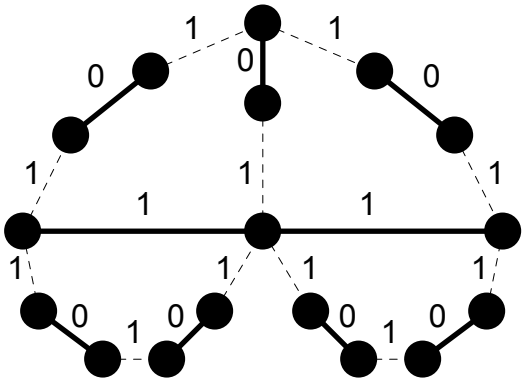


Fig. 3.6: RPP instance, c_e shown.

A little experimentation should convince the reader that all optimal solutions to the RPP in Fig. 3.6 have a cost of 12. Yet, when connectivity, R-odd cut and generalised path inequalities have been added to the LP for this instance, the invalid integral LP relaxation of Fig. 3.7 below is obtained, with a cost of 11. It has 'odd' vertices; yet no GTSP-type or honeycomb inequalities are violated. Thus, yet another new inequality is necessary to cut off this extreme point. To find this inequality, we proceed in the same way as for the extreme point shown in Fig. 3.4b.

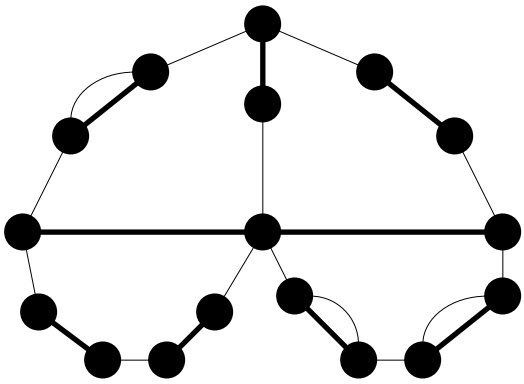


Fig. 3.7. Invalid LP relaxation.

Again, we draw the corresponding graph G' (Fig. 3.8a below), along with the analogous LP relaxation of the SGTSP instance on G' (Fig. 3.8b). It is apparent that this LP relaxation violates a clique-tree inequality, with 2 handles and 5 teeth, of the form shown in Fig. 2.3 in Chapter 2. This has a rhs of 18. Six fixing edges have a coefficient of 1 in this inequality; the other fixing edges have a coefficient of 0. By eliminating the corresponding variables, one obtains a new *generalised clique-tree* inequality, valid for **RPP(G)**, with a rhs of 12. This inequality is in fact precisely:

$$\sum_{e \in E} c_e x_e \geq 12 \qquad (3.2).$$

As in the earlier case of (3.1), (3.2) induces a facet of **RPP(G)** and the addition of (3.2) to the LP is sufficient to obtain an optimal solution to the RPP instance (with a cost of 12). Thus, the class of generalised clique-tree inequalities also contains inequalities in no hitherto known class, and these too can cut off integral LP relaxations with 'odd' vertices.

To summarise the results of this section: At least two classes of Generalised HTC inequalities contain hitherto unknown facet-inducing members, and these are sometimes necessary to cut off invalid integral extreme points. The HTC approach is therefore a significant new development in the quest for a description of **RPP(G)**.

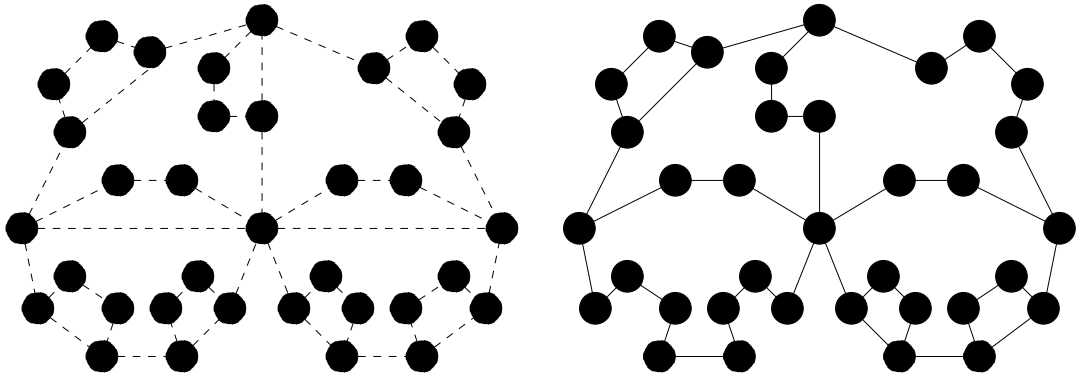


Fig. 3.8(a) corresponding G'. **(b) corresponding LP relaxation.**

3.4. Paths, Bridges and Necklaces.

In the previous section, the generalised comb inequality (3.1) was shown to be both facet-inducing and useful for one particular RPP instance. This particular inequality, which is also a generalised path inequality, has the property that the only required edge to receive a non-zero coefficient is an edge which crosses from A to Z in G'. This made the derivation of coefficients rather straightforward, since the rhs had merely to be reduced by 3 when eliminating variables.

The K-C inequalities, which are also generalised path inequalities (Theorem 3.10), have a similar property: The only required edges to receive a non-zero coefficient are the B edges crossing from A to Z in G'. This facilitated the proof of Theorem 3.10 since, when converting the path inequality in G' to a K-C inequality in G, the rhs had merely to be reduced by 3 B.

In the light of these observations, we will give special attention to the generalised path inequalities in which the only required edges receiving a non-zero coefficient cross from A to Z in G'. Associated with these inequalities is a configuration of the form shown in Fig. 3.9 overleaf. It is similar to an ordinary path configuration (see Fig. 2.5, p. 16), but there are two important differences:

i) There may now be required edges crossing from A to Z. These required edges will collectively be referred to as the *bridge*. The bridge is empty in an ordinary path configuration.

ii) The condition that P be odd and at least 3 is replaced by the condition that, if B is the number of required edges in the bridge, then P + B must be odd and at least 3.

We will call a partition of vertices with the structure shown in Fig. 3.9 a *path-bridge configuration* and the resulting inequality a *path-bridge inequality* (PBI). This section is entirely concerned with PBIs. They are important for at least five reasons:

i) R-odd cut, K-C, path (and therefore comb) inequalities are all PBIs,

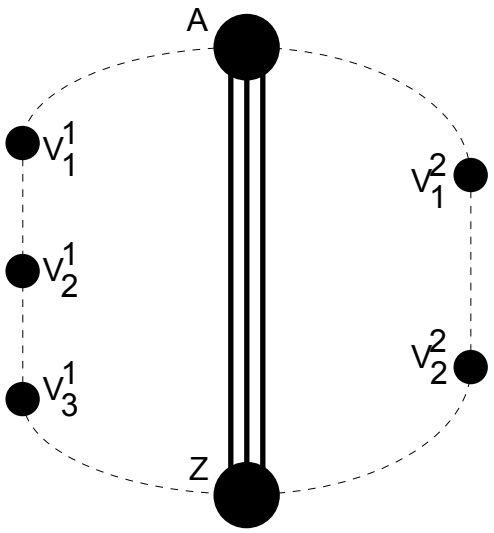


Fig. 3.9: Path-bridge configuration.

ii) The coefficients of a PBI can be calculated easily without invoking G', the SGTSP and so forth.

iii) PBIs appeared to be very useful in initial computational experiments,

iv) No other generalised path inequalities appeared to be of use computationally.

v) The necklace methods of Applegate et al. (1995) and Fleischer & Tardos (1996) (see Section 2.7) can be extended to produce a heuristic separation algorithm for PBIs.

Point (i) has already been explained. As for point (ii), it turns out that the coefficients and rhs of a PBI are given precisely by the formula (2.7) given in Section 2.4 for the ordinary path inequalities: each of the B extra paths of 3 fixing edges in G', corresponding to the B edges in the bridge, contributes 3 to the rhs of the path inequality in G'; yet, each of the variables associated with the 3B fixing edges has a coefficient of 1 in the path inequality and the rhs drops by 3B when these are eliminated.

Point (iv) could be explained if it was shown that generalised path inequalities which are not PBIs are redundant for **RPP(G)**. The author conjectures that this is so, but has been unable to prove it. The rest of this section is devoted to demonstrating point (v). The results given also have implications for the standard TSP.

By analogy with Section 2.7, we say that a PBI is *maximally violated* if it is violated by as much as possible given that all connectivity inequalities are already satisfied. Due to Theorem 3.4, a PBI is maximally violated in G if and only if an ordinary path inequality is maximally violated in G' (recall Figs. 3.4b and 3.5b). Thus, we can restrict our search to maximally violated path inequalities in G'.

Now, Goemans (1995) has shown (in the context of the GTSP) that a maximally violated path inequality has the following property: The sum of the x_e over the edges going from V^i_j to V^i_{j+1} is 1, for all i and for j = 0, ..., n_i. In the terminology of section 2.7, this is equivalent to saying that all V^i_j except A and Z are tight. Hence, each V^i_j apart from A and Z is either a bead in some necklace or the union of several beads.

Theorem 3.11. If a path inequality is maximally violated in G', then at least

$$\prod_{i=1}^{P}(n_i - 1)$$

comb inequalities are simultaneously violated in G'.

Proof: Let r_i, for i = 1, ..., n_i, satisfy $1 \le r_i \le n_i - 1$. Set:

$$T_i = V^i_{r_i} \cup V^i_{r_i+1}, \quad H = \bigcup_{i=1}^{P} \bigcup_{j=1}^{r_i} V^i_j.$$

The resulting comb inequality is maximally violated, since Goemans result implies that T_i, $H \cap T_i$ and $T_i \setminus H$ are tight for all i.

Now assume that G' is planar. It is possible (see Fleischer & Tardos, 1996 and the references therein) to obtain a list of all necklaces and beads in G' in O(N.M.logN)

time. Since G' is planar, this is $O(|V|^2.\log|V|)$ time. Moreover, the result of Fleischer & Tardos, that a necklace can be represented by a series of parallel edges running from one face of G' to another, still holds. If the Fleischer & Tardos algorithm is applied with G' as input, one of three things happens:

(i) The algorithm returns a maximally violated comb inequality in which each tooth is a domino (the union of two consecutive beads in a necklace).

(ii) The algorithm returns two (not maximally) violated comb inequalities.

(iii) The algorithm fails to find any violated comb inequality.

Now suppose that outcome (i) occurs. The comb inequality is (see Section 2.4) a 2-regular path inequality. To obtain a more general (not necessarily 2-regular) maximally violated path inequality, simply pick any particular tooth of the comb, made up, say, of beads b_1 and b_2, with b_1 adjacent to A in the path and b_2 adjacent to Z. In the necklace, b_1 will be adjacent not only to b_2, but also to some other bead b_0. If b_0 lies wholly within A, then we can extend the path b_1 - b_2 to b_0 - b_1 - b_2 and remove b_0 from A. This process can be performed repeatedly until A and Z no longer contain any suitable beads.

Preliminary computational testing showed that it is always a good strategy to extend the paths in this way, rather than using the 'raw' comb inequalities. A possible explanation for this is that path inequalities are *stronger* than comb inequalities: Goemans (1995) has shown that, in the case of the GTSP, the addition of comb inequalities to the system (2.9), (2.10) cannot increase the lower bound by more than a factor of 1/9. Even the clique-tree inequalities cannot increase it by more than a factor of 1/7. In contrast, path inequalities can increase it by a factor of 1/3 in some cases. Moreover, no other class of inequalities is known that will do better.

It seems likely that, in the context of the RPP, some similar result can be shown for the path-bridge inequalities.

3.5. Computational Experiments.

In Christofides et al. (1981), 24 RPP instances were constructed to test their Lagrangean relaxation algorithm. Corberán & Sanchis (1994) also solved these 24 instances, plus two more of their own. Out of the whole 26 instances, 11 could not be solved to optimality using only connectivity and R-odd cut inequalities. The data for these 11 particularly hard instances are given in Appendix 1, and these are the instances on which the new PBI separation routine shall be tested.

Table 3.1 below gives a summary of these eleven instances. The last three columns in the table show, respectively, the number of connected components in the graph obtained by dropping all non-required edges, the optimal solution value, and the number of branch-and-bound nodes needed in the Lagrangean relaxation algorithm.

Instance	|V|	|E|	|R|	Comps	Optm	LRB
I2	14	33	12	4	72	57
I4	17	35	22	3	29	31
I17	19	44	17	5	49	401
I5	20	35	16	5	55	79
I14	28	79	31	6	57	857
I16	31	94	34	7	64	4293
I24	41	125	55	7	113	5751
I21	49	110	67	6	78	8923
I20	50	98	63	7	116	1209
I23	50	158	78	6	95	8537
I22	50	184	74	6	122	14791

Table 3.1: Problem Characteristics.

There are three points to note here: first, the optimum of I21 is 78, rather than 76 as printed in Corberán & Sanchis (1994); similarly, the optimum of I16 is 64 rather than 63 (Corberán, 1996). Second, none of these instances can in any sense be regarded as easy. Third, none of the instances is planar. Despite this, however, it was found that (a) the graphs obtained by dropping non-basic, non-required edges were always planar throughout the cutting-plane procedure when connectivity and R-odd cut inequalities were all satisfied, and (b) the Fleischer-Tardos approach never returned a violated comb inequality which was not maximally violated. The author has no explanation for either of these facts.

The LP solver used was CPLEX 3.0. Minimum cut algorithms were kindly donated to the author by G. Skorobohatyj at the Zentrum fur Informationstechnik, Berlin (ZIB); the author had to write some C code also. Connectivity and R-odd cut separation routines were invoked alternately until no more violated inequalities could be found, at which point the PBI routine was invoked. The LP was always resolved as soon as a single violated inequality was found.

Table 3.2, overleaf, shows the performance of the cutting-plane procedure. The first 4 columns show the number of inequalities of each type in the final LP: R-odd cut, connectivity, K-C and (other) path-bridge. The next 3 columns show the lower bounds obtained by adding only R-odd cut inequalities (LB1), both R-odd cut and connectivity inequalities (LB2), and all three classes of inequality (LB3), respectively. As implied by the final column, all instances apart from I21 could be solved without branching. When the procedure terminated for I21, 13 branch-and-bound nodes produce an integer LP relaxation. This had no 'odd' vertices and so was an optimal solution.

As a further indication of performance, note that the average of LB1/OPT, LB2/OPT and LB3/OPT over the 11 instances come to 51.91%, 96.26% and 99.72%, respectively. Moreover, for I21, the PBIs closed precisely 2/3 of the gap between LB2 and the optimum. The PBI separation routine thus appears to be extremely useful in the solution of difficult RPP instances.

Instance	R-Odd	Con	KC	PBI	LB1	LB2	LB3	B&B
I2	9	6	1	0	24	67.5	72	-
I4	10	3	3	0	9	27	29	-
I17	15	5	1	0	27	48.5	49	-
I5	13	6	1	0	35	54.5	55	-
I14	31	8	3	0	26	53	57	-
I16	25	7	2	1	41	62.5	64	-
I24	33	9	1	0	60	111.5	113	-
I21	36	7	5	0	36	71	75 2/3	13
I20	43	9	3	0	41	111	116	-
I23	49	8	1	0	64	93	95	-
I22	38	5	0	0	93	122	122	-

Table 3.2: Computational Results.

Before closing this chapter, one anomalous result should be noted. Although Corberán & Sanchis reported that a single K-C inequality was needed to solve problem I22, the author found that it could be solved using only connectivity and R-odd cut inequalities. In order to shed light on this discrepancy, the author examined the simplex tableau for the final LP relaxation. It transpires that, for the particular cost function involved, the polyhedron described by the connectivity and R-odd cut inequalities has multiple optima. Many of these multiple optima represent valid solutions to the RPP instance, but there is at least one which does not. For this particular optimum, a single K-C inequality is maximally violated.

4. The Rural Postman Problem with Deadline Classes.

4.1. Overview.

In this chapter, the cutting-plane procedure for the RPP outlined in the previous chapter is adapted to the RPPDC, which was defined in Section 1.3. In Section 4.2, an integer programming formulation is given which has far fewer variables than another, more obvious, formulation. In Section 4.3, the fact that the RPPDC contains the RPP as a subproblem is exploited to give valid inequalities for the RPPDC. In Section 4.4, other valid inequalities are given which cannot be found in this way. In Section 4.5, separation algorithms are given for some of the inequalities discussed. Finally, the results of some computational experiments are discussed in Section 4.6.

For ease of exposition, it is assumed throughout that the depot is located at vertex 1.

4.2. Integer Programming Formulation.

Like many combinatorial optimisation problems, the RPPDC can be formulated as an integer programme in a variety of ways. One of the most common ways to formulate problems involving time-constraints is to treat time as a commodity which is consumed during the vehicle's journey; see, e.g., Desrosiers et al. (1995). This leads to a large number of zero-one variables representing the travel of the vehicle from one vertex to another, along with an equally large number of continuous variables representing the flow of the commodity.

It is not difficult to produce such a commodity-flow formulation for the RPPDC. However, there are three reasons why this approach is undesirable:

i) The formulation does not exploit the special structure of deadline classes.

ii) The formulation ignores any special structure that the network G may have. In practice, G may be a planar (or near-planar) road network, with low vertex degrees.

iii) Flow formulations are known to produce poor bounds when the time windows are large (Desrosiers et al., 1995). As mentioned in Section 1.5, windows of the deadline type are typically large.

These considerations led the author to avoid flow variables in favour of an approach which can exploit sparsity in G. This is done by regarding a feasible RPPDC solution as being composed of L *time-phases*, as explained below:

Let R_A^B, with $1 \le A \le B \le L$, denote $R^A \cup ... \cup R^B$. In phase 1, the vehicle leaves the depot and services the edges in R^1 plus (optionally) some edges in R_2^L. There is then a *phase transition* to phase 2, which must occur no later than time T^1. The vehicle begins phase 2 at the vertex at which it ended phase 1, and services any remaining edges in R^2 plus (optionally) some remaining edges in R_3^L. And so on. When the final phase, L, is finished (no later than T^L), all edges have been serviced and the vehicle returns to the depot via a shortest path.

The above observations lead to the following variable definitions:

x_{ep} = no. times $e \in E$ is traversed *without* servicing in time phase p.

y_{ep} = 1 if $e \in R$ is serviced in time phase p, 0 otherwise.

z_{vp} = 1 if time phase p ends at $v \in V$, 0 otherwise.

The x_{ep} are defined for all $e \in E$, p = 1, ..., L. For a given $e \in R^m$, y_{ep} is defined for p = 1, ..., m. When $e \in R^1$, however, y_{ep} need not be defined since it must equal 1. The z_{vp} are defined for all $v \in V$, p = 1, ..., L and represent the phase transitions (when p = 1,..., L-1), or the return trip to the depot from the last edge serviced (when p = L). Certain z_{vp} can in fact be fixed to zero without losing generality (e.g., by assuming that the vehicle finishes phase 1 immediately after servicing an edge in R^1), but we define all z_{vp} for notational convenience.

We will also need the following notation: Let c^*_v denote the cost of the shortest path from vertex v to the depot. For any $S \subset V$, let $\delta(S)$ (respectively, E(S)) denote the set of edges having exactly one end-vertex (exactly two end-vertices) in S. For

any F $\subseteq$ E, let $R_A^B(F) = R_A^B \cap F$ and $x_A^B(F) = \sum_{p=A}^{B} \sum_{e \in F} x_{ep}$. For any F $\subseteq R_A^L$, let $y_A^B(F)$ $= \sum_{p=A}^{B} \sum_{e \in F} y_{ep}$. If A = B in any of the above expressions, we drop the subscript, writing $R^B(E(S))$, $R^B(\delta(S))$ and so on. Finally, for any $\kappa \subseteq V$, let $z^p(\kappa) = \sum_{v \in \kappa} z_{vp}$.

In an optimal solution, no edge will be traversed more than twice in any particular phase: if the vehicle traverses an edge n times, n ≥ 3, it may just as well traverse it n - 2 times, since the route remains connected and Eulerian. Thus, no edge need be traversed more than 2L + 1 times overall (twice in each phase and once on the return trip to the depot). The RPPDC can now be formulated as follows:

$$\textit{Minimise} \quad \sum_{p=1}^{L} \sum_{e \in E} c_e x_{ep} \; + \; \sum_{v \in V} c_v^* z_{vL}$$

Subject to:

$$y_1^m(e) = 1 \qquad (m = 2, \ldots, L,\ e \in R^m) \qquad (4.1)$$

$$\sum_{p=1}^{m} \left[\sum_{e \in E} t_e x_{ep} + \sum_{e \in R_{m+1}^L} s_e y_{ep} \right] \leq T^m - \sum_{e \in R_1^m} s_e \qquad (m = 1, \ldots, L) \qquad (4.2)$$

$$z^p(V) = 1 \qquad (p = 1, \ldots, L) \qquad (4.3)$$

$$x^1(\delta(S)) + y^1(R_2^L(\delta(S))) + z^1(S) \geq 2$$
$$(\forall S \subseteq V \setminus \{1\}: R^1(E(S)) \neq \varnothing,\ R^1(\delta(S)) = \varnothing) \qquad (4.4a)$$

$$x^1(\delta(S)) + y^1(R_2^L(\delta(S))) + z^1(S) \geq 2\, y_{e1}$$
$$(\forall S \subseteq V \setminus \{1\}: R^1(\delta(S) \cup E(S)) = \varnothing,\ e \in R_2^L(E(S))) \qquad (4.4b)$$

$$x^1(\delta(S)) + y^1(R_2^L(\delta(S))) + z^1(S) \geq x_{e1}$$
$$(\forall S \subseteq V \setminus \{1\}: R^1(\delta(S) \cup E(S)) = \varnothing,\ e \in E(S)) \qquad (4.4c)$$

$$x^p(\delta(S)) + y^p(R_p^L(\delta(S))) + z^{p-1}(S) + z^p(S) \geq 2\, y_{ep}$$
$$(\forall S \subset V,\ p = 2, .., L,\ e \in R_p^L(E(S))) \qquad (4.4d)$$

$$x^p(\delta(S)) + y^p(R_p^L(\delta(S))) + z^{p-1}(S) + z^p(S) \geq x_{ep}$$
$$(\forall S \subset V,\ p = 2,..,L,\ e \in R_p^L(E(S))) \qquad (4.4e)$$

$$x^1(\delta(v)) + y^1(R_2^L(\delta(v))) + z_{v1} \equiv |R^1(\delta(v))| \mod 2 \quad (\forall v \in V \setminus \{1\}) \qquad (4.5a)$$

$$x^p(\delta(v)) + y^p(R_p^L(\delta(v))) + z_{v,p-1} + z_{vp} \equiv 0 \bmod 2$$
$$(\forall v \subset V,\ p = 2, ..., L) \qquad (4.5b)$$

$$x_{ep} \in \{0, 1, 2\};\ y_{ep}, z_{vp} \in \{0, 1\} \qquad (4.6)$$

The y variables do not appear in the objective function since the cost of servicing the required edges is fixed. The second component in the objective represents the cost of returning to the depot after servicing is completed. Constraints (4.1) state that each edge must be serviced exactly once, (4.2) are the time deadlines (after some simplification) and (4.3) ensure that there is only one phase transition at the end of each phase and one trip back to the depot after servicing the last required edge. Constraints (4.4a, b, c) ensure that the route is connected in phase 1 and (4.4d, e) do the same for the other phases. Constraints (4.5) ensure that the vehicle leaves each vertex as many times as it enters: (4.5a) concerns phase 1 and (4.5b) the other phases. Finally, (4.6) are the integrality conditions and bounds. Note that LP software with provision for Special Ordered Sets can exploit the structure of (4.3).

Under the assumption that $c_e > 0$ for all e, (4.4c), (4.4e) and the bounds of 2 on the x variables in (4.6) are redundant, since they will never be violated by any LP relaxation satisfying the other constraints. However, they are included in the formulation in the belief that a tighter formulation can lead to tighter valid inequalities (see Section 4.4).

The convex hull of solutions to (4.1) - (4.6) is a bounded polyhedron, i.e., a polytope. It will be denoted by **DC(G)**.

4.3. Valid Inequalities from the RPP Subproblem.

The fact that the RPPDC reduces to the ordinary RPP, when the deadlines are sufficiently large, implies that the inequalities for the RPP given in Chapter 3 have RPPDC analogues. This idea is explored in detail in this section. Some of the resulting inequalities have proved to be effective when used as cutting-planes for the RPPDC, suggesting that they induce high-dimensional faces of **DC(G)**. However, no attempt is made to find conditions under which they induce facets, for three reasons: First, little is yet known even about multidimensional Knapsack polyhedra such as implied by constraints (4.2) and (4.6); second, **DC(G)** is not full-dimensional due to constraints (4.1) and (4.3); third, it is clearly NP-Hard to calculate the dimension of **DC(G)** or even to determine if it is non-empty.

A very large number of inequalities, which are termed *strong cumulative* (SC) inequalities, are found from a consideration of the part of the vehicle route lying within the first B phases, where $1 \leq B \leq L$. Define a graph $G^B(V^B, E^B)$, with $V^B = V \cup \{1^*\}$ and $E^B = E \cup \{1, 1^*\} \cup \{\{v, 1^*\} \; \forall \; v \in V\}$. The vertex $\{1^*\}$ may be thought of as a copy of the depot. Now consider an RPP defined on G^B, with arbitrary costs, in which the required edges are $R_1^B \cup \{1, 1^*\}$. By using the formulation (2.11) - (2.13) given in Chapter 2, the polyhedron **RPP(G^B)** can be defined. All of the classes of valid inequalities and facets given in Chapter 3 are then valid for **RPP(G^B)**.

Theorem 4.1: If any inequality of the form

$$\sum_{e \in E} \alpha_e x_e \; + \; \sum_{v \in V} \beta_v x_{\{v,1^*\}} \; + \; \gamma \, x_{\{1,1^*\}} \; \geq \; \delta$$

*is valid for **RPP(G^B)**, then the inequality*

$$\sum_{e \in E} \alpha_e \, x_1^B(e) \; + \; \sum_{e \in R_{B+1}^L} \alpha_e \, y_1^B(e) \; + \; \sum_{v \in V} \beta_v \, z_{vB} \; \geq \; \delta$$

*is valid for **DC(G)**.*

Proof: Given any feasible solution to (4.1) - (4.6), construct a multigraph G(V*, E*) as follows: Let V* = V^B and let E* consist of the edge {1, 1*}, $x_1^B(e)$ copies of each edge $e \in E \setminus R$, $x_1^B(e) + 1$ copies of each edge in R^1, z_{vB} copies of edge {v, 1*} and, for $2 \leq m \leq L$, $x_1^B(e) + y_1^{\min(m,B)}(e)$ copies of each $e \in R^m$. By construction, G* is Eulerian and E* contains at least one copy of each $e \in R_1^B$. But this means that G* represents a feasible solution to the RPP defined on G^B. The inequality follows from the construction of G* and the fact that $x'_{\{1, 1*\}} = 0$ in this feasible solution.*

Three specific classes of SC inequality have proven to be particularly useful as cutting-planes, producing large increases in the objective function value. Unsurprisingly, these are the SC versions of the connectivity, R-odd cut and path-bridge inequalities (see Sections 2.6 and 3.4). For completeness, these are defined explicitly in the following three corollaries:

Corollary 4.2: For some $1 \leq B \leq L$, let $S \subseteq V - \{1\}$ be such that $R_1^B(E(S)) \neq \emptyset$ and $R_1^B(\delta(S)) = \emptyset$. Then the strong cumulative connectivity (SCC) inequality:

$$x_1^B(\delta(S)) + y_1^B(R_{B+1}^L(\delta(S))) + z^B(S) \geq 2 \qquad (4.7)$$

*is valid for **DC(G)**.*

Corollary 4.3: If $1 \leq B \leq L$ and $S \subseteq V - \{1\}$, then the inequality

$$x_1^B(\delta(S)) + y_1^B(R_{B+1}^L(\delta(S))) + z^B(S) \geq 1 \qquad (4.8a)$$

*is valid for **DC(G)** when $|R_1^B(\delta(S))|$ is odd and*

$$x_1^B(\delta(S)) + y_1^B(R_{B+1}^L(\delta(S))) + z^B(V - S) \geq 1 \qquad (4.8b)$$

*is valid for **DC(G)** when $|R_1^B(\delta(S))|$ is zero or even. These will be called strong cumulative parity (SCP) inequalities.*

Corollary 4.4: Let B, P and W be integers with $1 \le B \le L$, $P \ge 1$, $W \ge 0$, $P + W \ge 3$ and odd. Let n_i (i = 1,..., P) be integers greater than one. Partition V into sets A, Z, V^i_j (i = 1, ..., P, j = 1, ..., n_i). For convenience, for all i we identify V^i_j with A when j = 0 and with Z when $j = n_i + 1$. If the partition is such that:

- for each edge $\{u, v\} \in R^B_1$, either {u} and {v} lie within the same $G(V^i_j)$, or one of {u} and {v} is in A and the other is in Z;

- for i = 1, ..., P, j = 1, ..., n_i, either $R^B_1(E(V^i_j)) \neq \emptyset$, or $\{1\} \in V^i_j$, or both;

- there are W required edges in R^B_1 with one end-vertex in A and the other in Z;

then the strong cumulative path-bridge (SCPB) inequality:

$$\sum_{e \in E} \alpha_e x_e + \sum_{e \in R^L_{B+1}} \alpha_e y_e + \sum_{v \in V} \beta_v z_{vB} \ge 1 + \sum_{i=1}^{P} \frac{n_i + 1}{n_i - 1}$$

*is valid for **DC(G)**, where the α_e are given as in Section 2.4 and the coefficient β_v equals the coefficient an edge would have if it connected v to {1}.*

The SCC inequalities (4.7) generalise (4.4a).

Despite the fact that there are vast numbers of possible SC inequalities, there are other inequalities, also concerning the first B phases, which are not of the SC type. Consider the following results:

Theorem 4.5: For some $1 \le B \le L$, let $S \subseteq V - \{1\}$ be such that $R^B_1(E(S) \cup \delta(S)) = \emptyset$, but $R^L_{B+1}(E(S)) \neq \emptyset$. Then, for any $e \in R^m(E(S))$, with $B < m \le L$,

$$x^B_1(\delta(S)) + y^B_1(R^L_{B+1}(\delta(S))) + z^B(S) \ge 2\, y^m_1(e) \qquad (4$$

*is valid for **DC(G)**.*

Proof: If e is not serviced in phases 1 to B, the rhs becomes 0 and the inequality is trivially true. On the other hand, if e is serviced in phases 1 to B, the rhs becomes 2, which is valid since the vehicle must enter S at least once during phases 1 to B, and must then either leave S (possibly servicing an edge in R^L_{B+1} *as it does so), or end phase B while still within S.*

Theorem 4.6: If $1 \leq B \leq L$ *and* $S \subseteq V \setminus \{1\}$*, then:*

$$x_1^B(\delta(S)) + y_1^B(R^L_{B+1}(\delta(S) - F)) + z^B(S) \geq y_1^B(F) - |F| + 1$$

is valid for any $F \subseteq R^L_{B+1}(\delta(S))$ *such that* $|F \cup R_1^B(\delta(S))|$ *is odd and*

$$x_1^B(\delta(S)) + y_1^B(R^L_{B+1}(\delta(S) - F)) + z^B(\mathrm{V} - S) \geq y_1^B(F) - |F| + 1$$

is valid for any $F \subseteq R^L_{B+1}(\delta(S))$ *such that* $|F \cup R_1^B(\delta(S))|$ *is zero or even.*

Proof: If $|F| - 1$ *or less of the edges in F are serviced in phases 1 to B, the rhs becomes zero or less and the inequality is trivially true. On the other hand, if all of the edges in F are serviced in phases 1 to B, the rhs becomes 1, which is valid since the vehicle must enter and leave S an even number of times.*

The inequalities (4.9), which generalise (4.4b), can be obtained by the following three step procedure: (I) Temporarily fix y_{ep} to zero for $B < p \leq m$, which is equivalent to moving e from R^m to R^B. (II) Invoke Corollary 4.2 for the new RPPDC instance thus formed to give an SC inequality. (III) Lift the resulting inequality back into the space of the original variables (for an introduction to lifting see, e.g., Nemhauser & Wolsey, 1988). For this reason, (4.9) will be termed *lifted cumulative connectivity* (LCC) inequalities. Similarly, the inequalities (4.10a, b) can be obtained by fixing variables associated to the edges in F, invoking Corollary 4.3 and lifting. They will be called *lifted cumulative parity* (LCP) inequalities.

Other *lifted cumulative* inequalities can be found in a similar way, but only the LCC and LCP inequalities have proven to be useful in the cutting-plane algorithm. Even these do not tend to produce as large an increase in the objective function as the SC inequalities, although they are necessary to obtain feasibility.

Unfortunately, **DC(G)** is even more complicated than implied by the results given so far. The inequalities presented so far have all concerned parts of the route involving phase 1. It is possible to derive inequalities which concern only later phases, as illustrated below.

Theorem 4.7: If $S \subseteq V$ *is such that* $R_2^L(S) \neq \emptyset$ *and e is any edge in* $R^m(S)$, $m \geq 2$, *then*

$$x_A^B(\delta(S)) + y_A^B(R_A^L(\delta(S))) + z^{A-1}(S) + z^B(S) \geq 2\, y_A^B(e) \qquad (4$$

is valid for any $2 \leq A \leq B \leq m$.

Proof: This is analogous to the proof of Theorem 4.5. If e is not serviced during phases A to B, the inequality is trivially true. On the other hand, if e is serviced during these phases, the rhs becomes 2 which is valid since: (a) the vehicle must either enter S during phases A to B, or finish phase A-1 while within S, and (b) the vehicle must either leave S during phases A to B, or finish phase B while within S.

Theorem 4.8: for any $S \subseteq V$, $2 \leq A \leq B \leq L$ *and* $F \subseteq R_A^L(\delta(S))$,

$$x_A^B(\delta(S)) + y_A^B(R_A^L(\delta(S) - F)) + z^{A-1}(S) + z^B(S) \geq y_A^B(F) - |F| + 1 \qquad (4.1$$

is valid when $|F|$ *is odd and*

$$x_A^B(\delta(S)) + y_A^B(R_A^L(\delta(S) - F)) + z^{A-1}(S) + z^B(V - S) \geq y_A^B(F) - |F| + 1$$

is valid when $|F|$ *is even.*

Proof: This is analogous to the proof of Theorem 4.6. If |F| - 1 or less of the edges in F are serviced during phases A to B, the rhs becomes zero or less and the inequality is trivially true. On the other hand, if all of the edges in F are serviced during phases A to B, the rhs becomes 1, which is valid since the vehicle must enter and leave S an even number of times.

For obvious reasons, (4.11) and (4.12a, b) are called *non-cumulative connectivity* (NC) and *non-cumulative parity* (NP) inequalities, respectively. They tend to produce little increase, if any, in the objective function when used as cutting-planes. However, they are required in order to gain feasibility. Note that (4.10) generalise (4.4d).

4.4. Other Valid Inequalities.

Many of the inequalities presented in the previous section have proved to be useful as cutting-planes. In this section, some other possible valid inequalities are discussed. However, they are probably of theoretical interest only, since no good separation algorithms are known.

Suppose that for some RPPDC instance with L = 4, $S \subseteq V\backslash\{1\}$ is such that $R_1^3(\delta(S)) = \varnothing$ and $|R^4(\delta(S))|$ is odd. If, in some feasible solution, $y^1(R^4(\delta(S))) = y^3(R^4(\delta(S))) = 0$, then precisely one of the quantities $y^2(R^4(\delta(S)))$ and $y^4(R^4(\delta(S)))$ must be odd. If the first is odd, Theorem 4.8 yields $x^2(\delta(S)) + z^1(S) + z^2(S) \geq 1$. If the second is odd, Theorem 4.8 yields $x^4(\delta(S)) + z^3(S) + z^4(S) \geq 1$. Either way, $x^2(\delta(S)) + z^1(S) + z^2(S) + x^4(\delta(S)) + z^3(S) + z^4(S) \geq 1$ holds. Hence, for the RPPDC instance under discussion, the following inequality is valid:

$$x^2(\delta(S)) + z^1(S) + z^2(S) + x^4(\delta(S)) + z^3(S) + z^4(S) \geq 1 - y^1(R^2(\delta(S))) - y^3(R^2(\delta(S))).$$

It can be shown that this inequality is not equal to, nor dominated by, any of the inequalities in Section 4.3. In order to make the nature of this inequality more apparent, it is helpful to use equations (4.1) to rewrite the right-hand-side as:

$$y^2(R^2(\delta(S))) + y^4(R^2(\delta(S))) - |R^2(\delta(S))| + 1.$$

In this form, the inequality resembles an NP inequality of type (4.12a). This result may be generalised, via a consideration of *all possible subsets* of {2, ..., L}, to obtain a class of valid inequalities which subsumes the class of NPs (4.12a):

Theorem 4.9: Let A_i and B_i, for i = 1, ..., h, be integers satisfying $A_1 \geq 2$, $B_t \leq L$, $B_i \geq A_i$ for all i, and $A_{i+1} \geq B_i + 2$ for i = 1, ..., h. Let $S \subseteq V$ be such that $R^L_{A_1}(\delta(S)) \neq \varnothing$ and let $F \subseteq R^L_{A_1}(\delta(S))$ be such that $|F|$ is odd. Then the following inequality is valid:

$$\sum_{i=1}^{h}\left(x^{B_i}_{A_i}(\delta(S)) + y^{B_i}_{A_i}(R^L_{A_1}(\delta(S) - F)) + z^{A_i-1}(S) + z^{B_i}(S)\right) \geq \sum_{i=1}^{h} x^{B_i}_{A_i}(F) - |F| + 1.$$

Proof: If a feasible solution satisfies $\sum_{i=1}^{h} x^{B_i}_{A_i}(F) < |F|$, the rhs becomes zero or less and the inequality is trivially true. Now suppose a feasible solution has $\sum_{i=1}^{h} x^{B_i}_{A_i}(F) = |F|$; the rhs reduces to one in this case. Since $|F|$ is odd, $x^{B_i}_{A_i}(F)$ must be odd for at least one $1 \leq i \leq h$. Then, Theorem 4.8 implies that

$$x^{B_i}_{A_i}(\delta(S)) + y^{B_i}_{A_i}(R^L_{A_1}(\delta(S) - F)) + z^{A_i-1}(S) + z^{B_i}(S) \geq 1$$

for at least one $1 \leq i \leq h$. The result follows from the non-negativity of the variables.

The author calls these inequalities *generalised non-cumulative parity* (GNP) inequalities. It may be the case that further GNP inequalities can be derived by analogy with the inequalities (4.12b). In addition, it is probable that a similar generalisation of the LCP inequalities (4.10a, b) is possible, yielding *generalised lifted cumulative parity* (GLCP) inequalities. Hence, the number of different inequalities of the parity type is immense.

These new inequalities will not be examined further, for the following reason: The author has attempted to solve the corresponding separation problem and has not been able to arrive at an algorithm with a running-time which is polynomial in L. It may in fact be the case that the separation problem is strongly NP-Hard. If so, the RPPDC formulation will only be effective when L is small, say $L < \ln |R|$ as recommended in Section 1.3. In Section 6.3, a similar argument will be used to conclude that a particular formulation of the Capacitated Arc-Routing Problem (CARP), due to Belenguer & Benavent (1996), will only be effective when the number of vehicles is small.

Even this is not the last word on the parity question. Due to the presence of the deadlines (4.2), it is conceivable that constraints (4.5) could be violated even after all known inequalities have been added and integrality has been achieved through branch-and-bound. Some unknown inequalities would then be necessary to cut off such invalid solutions. Fortunately, this situation did not arise during computational experiments.

Now note that constraints (4.1), (4.2) and (4.6) form a multidimensional Knapsack subproblem with upper bounds of 2 on the x variables and generalised upper bounds on the y variables. It may be possible to produce valid inequalities for **DC(G)** based on a consideration of this structure. Some polyhedral results are available for the 0-1 Knapsack problem (see, e.g., Balas, 1975; Nemhauser & Wolsey, 1988), and also for the 0-1 Knapsack Problem with generalised upper bounds (see, e.g., Johnson & Padberg, 1981; Nemhauser & Vance, 1994). However, to the author's knowledge, no nice results are presently known for multidimensional Knapsack polyhedra. For this reason, this area was not explored further.

Finally, it might be possible to produce inequalities which take into account the Knapsack and routing aspects simultaneously. Belenguer & Benavent (1996) do something similar to this in the context of the CARP (see Section 5.4). Again, the author did not explore this any further.

4.5. Separation Algorithms.

In this section, polynomial separation algorithms are given for the various connectivity and parity inequalities presented in Section 4.3. Moreover, the separation problem for SCPB inequalities is considered briefly. It is assumed that the reader is familiar with the concept of *shrinking* a set of vertices in a weighted graph, which is used by many authors (e.g., Fleischmann, 1985; Padberg & Rinaldi, 1990; Nagomochi et al., 1994). Shrinking a set S of vertices means deleting E(S), moving the vertices in S together until they coincide, then replacing each set of parallel edges thus formed with a single edge, summing weights in the process.

The vertex in the shrunk graph which corresponds to S will be called a *supernode*. Shrinking can be done repeatedly in that a set S' of vertices in a shrunk graph may also be shrunk, even if it contains supernodes from a previous shrinking.

The separation problem for the strong cumulative connectivity (SCC) inequalities can be solved in the following way: For a fixed value of B, construct a graph as follows: Copy G(V, E), give each $e \in R_{B+1}^{L}$ a weight of $x_1^B(e) + y_1^B(e)$ and each $e \in E \setminus R$ a weight of $x_1^B(e)$. For each $v \in V$, add an extra edge $\{v, 1\}$ with weight z_{VB}. Call the resulting graph the *B-graph*. Now shrink all edges in R_1^B to obtain what will be called the *SB-graph*. There is now a many-to-one correspondence between edges in R_1^B in the original graph and supernodes in the SB-graph.

The weights of the edges in the SB-graph have been contrived in such a way that an SCC inequality is violated if and only if there is a cut in the SB-graph of weight less than 2, such that at least one supernode lies on the opposite shore of the cut to the depot vertex $\{1\}$. To test if such a cut exists, it suffices to solve a series of maximum flow problems, one for each supernode. In each case, the flow is sent from one particular supernode to $\{1\}$.

Now, for fixed B, $O(\min\{|V|, |R|\})$ maximum flows will be needed and the SB-graph contains $O(|V|)$ vertices and $O(|E|)$ edges. Therefore:

Theorem 4.9: The separation problem for SCC inequalities can be solved in the time taken to perform O(L.min{|V|, |R|}) maximum flow problems in graphs with O(|V|) nodes and O(|E|) edges.

In practice, the running-time of the algorithm is often much better than implied by Theorem 4.9. The number of supernodes may be very small.

Now consider the separation of lifted cumulative connectivity inequalities (4.9). Assume once more that B is fixed and construct the B-graph once more. Note that one of the conditions on the set S in (4.9) is that $R_1^B(E(S) \cup \delta(S)) = \varnothing$. This permits a more radical shrinking of the B-graph than was used to form the SB-graph: *All of the vertices in the B-graph which are incident to edges in R_1^B, plus {1} if this is not already included, can be shrunk into a single supernode*. Call the resulting graph the TB-graph (T for 'tiny'). Since the conditions on S and e in (4.9) imply that e is not adjacent to any edge in R_1^B, this means that, for fixed B, the only edges in R_{B+1}^L which need to be considered as candidates for e in (4.9) are those which are not adjacent to the supernode in the TB-graph.

For fixed B and a fixed suitable candidate e = {u, v}, choose an arbitrary end-vertex {u} and send a maximum flow from {u} to the supernode in the TB-graph. If the weight of the resulting cut is $\geq 2\, y_1^B(e)$, no LCC inequality is violated. If the weight is $< 2\, y_1^B(e)$ and {u} and {v} are on the same shore of the cut, then an LCC inequality is violated for the given B and e. The third possibility is that the weight is $< 2\, y_1^B(e)$, but {u} and {v} are on opposite shores of the cut. In such a situation, it turns out that a lifted cumulative parity inequality of the form (4.10a) is violated. To see this, set S equal to the shore of the cut containing {u} and set F = {e}. The resulting LCP inequality (4.10a) can be rewritten as:

$$x_1^B(\delta(S) + y_1^B(R_{B+1}^L(\delta(S)) + z^B(S) \geq 2\, y_1^B(e),$$

which is clearly violated.

Once again, for fixed B, O(min{|V|, |R|}) maximum flows will be needed. Therefore:

Theorem 4.10: If all LCP inequalities are already satisfied, the separation problem for LCC inequalities can be solved in the time taken to perform O(L.min{|V|, |R|}) maximum flow problems in graphs with O(|V|) nodes and O(|E|) edges.

In practice, however, the running-time is much faster, since the TB-graph can be very small and there may only be a handful of candidates for e.

In fact, the routine can be made even faster, due to the following result:

Theorem 4.11: Suppose all LCP inequalities are satisfied, B is fixed and a LCC inequality is violated for some S and e. If an $f \in R^L_{B+1}$ *is adjacent to e, such that* $y_1^B(f) \geq y_1^B(e)$*, then the LCC inequality with f replacing e is violated by at least as much.*

Proof: As e and f are adjacent, f is in either R(E(S)) or R(δ(S)). If f is in R(E(S)), the result is immediate. Suppose f lies in R(δ(S)). Because all LCP inequalities are satisfied, we have

$$x_1^B(\delta(S)) + y_1^B(R^L_{B+1}(\delta(S))) + z^B(S) \geq 2\, y_1^B(f) \geq 2\, y_1^B(e),$$

which contradicts the assumption that the LCC inequality is violated for the given S and e.

Thus only a few edges in R^L_{B+1} (the ones with locally maximal y values), need to be considered as candidates for the edge e. This leads to fewer maximum flows being needed.

The separation problem for non-cumulative connectivity (NC) inequalities is similar to that for LCC inequalities, but a little more tricky. Assume that A and B are both

fixed. Construct the following *parachute graph*: Copy G(V, E), give each edge in R_A^L a weight of $x_A^B(e) + y_A^B(e)$ and each other edge a weight of $x_A^B(e)$. Add a *virtual depot* vertex {1*} and connect each v ∈ V to {1*} with an edge of weight $z_{V,A-1} + z_{VB}$. Now, pick any edge e = {u, v} from R_A^L. A minimum cut in the parachute graph such that {u} and {1*} lie on opposite shores can be found by sending a maximum flow from {u} to (1*}.

The parachute graph has been constructed so that, if the weight of the resulting cut is ≥ 2 $y_1^B(e)$, no NC inequality is violated for the particular values of B and e. If the weight is < 2 $y_1^B(e)$ and {u} and {v} are on the same shore of the cut, then an NC inequality is violated. Finally, if the weight is < 2 $y_1^B(e)$, but {u} and {v} are on opposite shores of the cut, a non-cumulative parity inequality is violated by an argument similar to the one given above.

The maximum possible number of times the maximum flow routine needs to be invoked is again O(min{V, R}). Since the number of edges in the parachute graph is |E| + |V| = O(|E|), we have:

Theorem 4.12: If all NP inequalities are already satisfied, the separation problem for NC inequalities can be solved in the time taken to perform O(L^2.min{V, R}) maximum flow problems in a graph with O(|V|) nodes and O(|E|) edges.

Moreover, a result similar to that of Theorem 4.11 could allow the algorithm to be improved substantially.

We now consider the strong cumulative parity (SCP) and lifted cumulative parity (LCP) inequalities simultaneously. The reason this can be done is because (4.8a,b) can be regarded as a special case of (4.10a, b); namely, the case in which F = ∅.

Now note that (4.10a) and (4.10b) can be rewritten in the following form:

$$x_1^B(\delta(S)) + y_1^B(R_{B+1}^L(\delta(S)-F) + \sum(1-y_1^B(e)) + z^B(S) \geq 1 \qquad (4.10$$

$$x_1^B(\delta(S)) + y_1^B(R_{B+1}^L(\delta(S)-F) + \sum(1-y_1^B(e)) + z^B(V \setminus S) \geq 1 \qquad (4.10b'$$

We use the well-known edge-splitting strategy (see Padberg & Rao, 1982 and Section 2.7). Create a copy of G(V, E) and label vertices *even* unless they meet an odd number of edges in R_1^B, when they should be labelled *odd*. Give each e $\in R_{B+1}^L$ a weight of $x_1^B(e) + y_1^B(e)$, other edges a weight of $x_1^B(e)$. Now *split* each e $\in R_{B+1}^L$ into two new edges, called *halves,* by inserting a new vertex, labelled odd, in the middle. Let one half (the *normal* half) retain the previous weight, but let the other (the *flipped* half) have weight $1 + x_1^B(e) - y_1^B(e)$. Now add an extra odd vertex {1*} and reverse the label of the depot. For each v $\in$ V, add an edge {v, 1*} of weight z_{iB}. Finally, reverse the label of any v $\in$ V which is adjacent to an odd number of flipped halves.

A cut in the resulting *split graph* will have weight equal to the left-hand-side of 4.10a' (resp., 4.10b') if {1}and {1*} lie on the same (opposite) shores of the cut; moreover, the cut will be *odd* iff the edge-cutset contains an *odd* (*even*) number of flipped halves. Therefore we have immediately that a SCP or LCP is violated iff there is an odd cutset in the split graph with a weight less than 1. If {1} and {1*} are on the same shore of the cut, it is of the form 4.10a'; but if they are on opposite shores, it is of the form 4.10b'. The set F consists of those e $\in R_{B+1}^L$ whose flipped halves lie in the edge-cutset.

The minimum weight odd-cut routine of Padberg & Rao (1982) entails the solution of N_{ODD} maximum flow problems, where N_{ODD} is the number of odd vertices. Since the split graphs contains O(max{|V|, |R|}) vertices, O(|E|) edges and O(|R|) odd vertices, we have:

Theorem 4.13: The separation problem for SCP and LCP inequalities, combined, can be solved in the time taken to perform O(L.|R|) maximum flow problems in graphs with O(max{|V|, |R|}) vertices and O(|E|) edges.

A similar approach works for the non-cumulative parity (NP) inequalities. Assume A and B are fixed. Create a copy of G(V, E) and label all vertices *even*. Give each e $\in$

R^L_A a weight of $x^B_A(e) + y^B_A(e)$, other edges a weight of $x^B_A(e)$. Now split each $e \in R^L_A$ into two halves, by inserting a new odd vertex in the middle. Let the normal half retains the previous weight, but let the flipped half have weight $1 + x^B_A(e) - y^B_A(e)$. Now add *two* extra odd vertices, {1*} and {1**}. From each $v \in V$, add an edge to {1*} of weight $z_{i,A-1}$ and an edge to {1**} of weight z_{iB}. Finally, reverse the label of any $v \in V$ which is adjacent to an odd number of flipped halves.

By the same arguments as for the LCP inequalities, a NC is violated iff there is an odd cutset in the split graph with a weight less than 1. If {1*} and {1**} are on the same shore of the cut, it is of the form (4.12a); but if they are on opposite shores, it is of the form (4.12b). The set F consists of those $e \in R^L_A$ whose flipped halves lie in the edge-cutset. We also have:

Theorem 4.14: The separation problem for NP inequalities can be solved in the time taken to perform $O(L^2.|R|)$ maximum flow problems in graphs with $O(\max\{|V|, |R|\})$ vertices and $O(|E|)$ edges.

Finally, consider the separation problem for the strong cumulative path-bridge (SCPB) inequalities. Initially, the author attempted to look for *maximally violated* SCPB inequalities by analogy with Section 3.4. However, preliminary experiments showed that SCPB inequalities were rarely, if ever, maximally violated. This is because constraints (4.2) tend to destroy integrality entirely in LP relaxations.

For this reason a simpler heuristic is used: Before solving an RPPDC instance, the instance is first treated as an ordinary RPP and the cutting-plane algorithm of Chapter 3 is used. In the final LP relaxation of this RPP instance, there will typically be one or more binding PB inequalities. These can then be transformed into their strong cumulative counterpart via Theorem 4.1 and used as cutting-planes for the RPPDC instance proper.

Despite the fact that this is an extremely naive approach, it worked well in limited computational experiments, as will be seen in the next section.

4.6. Computational Experiments.

In this section, some computational results are given for a number of RPPDC problem instances, using the separation algorithms outlined in the previous section. The initial LP relaxation is found by solving the RPPDC instance as an ordinary RPP, by the method of Chapter 3, then converting the binding constraints in the last LP relaxation into their SC form. These are then added to (4.1) - (4.3). Then, the SCC, LCC, SCP/LCP, NC and NP separation algorithms are invoked in cyclic order until a violated inequality is found. Each time an inequality is added, the LP is resolved using the dual simplex method. When no more violated inequalities can be found, branch-and-bound is invoked to obtain integrality.

It may happen that the resulting integer solution violates one or more known inequalities. If this happens, the inequalities are appended to the LP and the cutting-plane procedure continues again, followed by branch-and-bound once more, and so on. Such restarts, which were done manually, were necessary since branch-and-cut software was unavailable at the time of writing. The number of restarts needed ranged from zero to about fifty in the problems we tested, so branch-and-cut software would speed up the algorithm significantly.

As in Chapter 3, CPLEX version 3.0 along with some C routines for cut problems due to Georg Skorobohatyj were run on a Hewlett Packard HP-9000 workstation. Additional C routines for constructing the parachute and split graphs and forming inequalities were written by the author.

The test problems are based on the five hardest RPP instances solved in Section 3.5: I4, I14, I16, I20 and I21. All of these required at least 3 path-bridge inequalities to be solved. Table 4.1 (overleaf) shows the number of *binding* connectivity, R-odd cut, K-C and (other) path-bridge inequalities in the final LP relaxation when solved as an ordinary RPP.

Two RPPDC versions of each problem were solved: one with L = 1 and one with L = 2. In each case, t_e was set to c_e and s_e was set to $3c_e/2$ rounded down to the nearest

Instance	Connectivity	R-Odd Cut	K-C	Path-Bridge
I4	1	7	3	0
I14	5	21	2	0
I16	6	18	1	1
I20	9	20	2	0
I21	4	22	5	0

Table 4.1: Binding constraints.

number. Such a high correlation between times and costs is often encountered in practice. The versions with L = 1 were formed by making the deadline just tight enough to make the ordinary RPP solution invalid. The versions with L = 2 were formed by a slightly more complicated procedure as follows:

Partition R into equivalence classes, putting two edges in the same class if they have the same cost. List the classes in decreasing order of cost and set $R^1 = \varnothing$ and $R^2 = R$. Then do the following until $|R^1| \geq |R^2|$: remove a class C from the head of the list, set $R^1 := R^1 \cup C$ and $R^2 := R^2 \setminus C$. This led to $|R^1|$ values of 11, 20, 19, 40 and 38, respectively, for the five problems. "Plausible" phases 1 and 2 were then formed by hand, and the deadlines set accordingly.

The computational results are given in Tables 4.2 and 4.3 overleaf. For each problem the following is listed: the time deadline(s), the number of cutting-planes of each type generated in addition to those in the initial relaxation, the cost of the final LP relaxation, the number of branch-and-bound nodes required to solve the final relaxation to optimality and the cost of the optimal solution. It will be seen that all ten instances could be solved to optimality by the cutting-plane method.

Instance	T^1	Con	R-Odd	LP	B&B	Optimum
I4	105	0	3	29.67	20	33
I14	260	0	0	57.00	15	59
I16	263	0	3	64.43	27	67
I20	522	0	4	116.49	28	118
I21	490	0	2	75.67	180	84

Table 4.2: Results for single deadline class.

Instance	T1	T2	Con	R-Odd	LP	B&B	Optimum
I4	92	110	28	50	30.35	25	33
I14	236	321	33	42	57.23	63	62
I16	229	268	29	27	67.65	147	72
I20	498	624	37	52	116.17	131	118
I21	468	500	56	81	75.84	2657	84

Table 4.3: Results for two deadline classes.

Although the final column of Table 4.2 is identical to the final column of Table 4.3 in three positions, this should not be taken to mean that the corresponding solutions were identical. Indeed, a comparison of the solutions showed that this was not the case. For two of the three pairs, there were edges traversed in the L = 1 version which were not traversed in the L = 2 version and vice-versa.

Due to the manual restarts, it is not too meaningful to talk in terms of computation time. It is worth mentioning, however, that the time taken by the LP solver and the separation routines, combined, never exceeded 3 minutes for any problem; also, each single invocation of branch-and-bound took less than 20 seconds. The author is confident that each of the problems could be solved within a few minutes if branch-and-cut software was available.

A final point: When solving practical problems, the running time of the algorithm will probably be improved if the edge costs are measured very accurately (e.g. to 3 significant figures), perhaps from a Geographical Information System (GIS). This acts as a perturbation scheme which reduces the chance of alternate optima. In contrast, it is only worthwhile measuring *times* to 1/2 significant figures. This is for two reasons: First, LP packages perform better if all constraints have similar precision; second, if a viable separation algorithm ever emerges for "Knapsack-type" valid inequalities (Section 4.4), it is likely to be pseudopolynomial in the coefficients of constraints (4.2) (see, e.g., Boyd, 1992). Fortunately, real-life time constraints are often of a 'soft' nature.

5. Literature on Multi-Vehicle Problems.

5.1. Overview.

In Chapter 2, the relevant literature on single-vehicle problems was reviewed. In this chapter, multi-vehicle problems are considered. As in Chapter 2, the scope is mainly restricted to *optimisation algorithms for single-depot problems*. Special attention will be given to Arc-Routing Problems (ARPs), but, as more results are known for their Node Routing counterparts, some mention will be made of these also. Formal definitions are given in the next section.

A variety of different formulations have been attempted for multi-vehicle problems. In this chapter, these will be put into one of the following four categories: *complete*, *sparse*, *very sparse* and *other*. Sections 5.3 - 5.6, respectively, describe these formulations and review polyhedral results and algorithms where applicable. In Section 5.7, some lower bounding procedures are briefly reviewed. In Section 5.8, some literature is reviewed on a strongly NP-Hard problem known as the *Bin Packing Problem* (BPP). An understanding of the BPP turns out to be useful for deriving valid inequalities for the CARP, as will be seen in Chapter 6.

5.2. Definitions.

The *Capacitated Arc Routing Problem* or CARP has already been defined in Section 1.4. Golden and Wong (1981) provided the first formulation of the CARP and noted that merely determining the minimum possible number of vehicles is equivalent to the BPP. A consequence of this is that even finding a CARP solution within 50% of the optimum is also strongly NP-Hard.

The CARP with R = E is called the *Capacitated Chinese Postman Problem* (CCPP) and was first considered in Christophides (1973). Unlike the ordinary Chinese Postman Problem (see Section 2.2), the CCPP is strongly NP-Hard.

The node-routing counterpart of the CARP (i.e., the version in which R contains vertices instead of edges), has been the subject of intense research and for this reason is often referred to as *the* Vehicle Routing Problem (VRP). Laporte & Osman (1995) provides a comprehensive survey on this and other problems.

Because the CARP+1D (see Section 2.2) will be studied in Chapter 7, it is worth mentioning a little about time windows. Although no specialised optimisation algorithms are currently available for ARPs with time windows, much work has been done on the *Vehicle Routing Problem with Time Windows* (VRPTW). The best current solution method for the VRPTW appears to be that of Kohl & Madsen (1995). However, as in the case of the single-vehicle TSPTW (Section 2.3), VRPTWs with as few as fifty customers can defeat even the most recent methods unless the time windows are rather narrow.

For this reason, current VRPTW algorithms cannot be applied to problems with only *time deadline*s (though some heuristics appear in Nygard et al., 1988, Thangiah et al., 1994). It is unlikely that they could even be adapted to the CARP+1D. Yet, as mentioned in Section 2.3, deadlines do in fact occur in real-life. Eglese & Li (1992) study a multi-vehicle, multi-depot problem, in which roads must be treated with salt within 2, 4 or 6 hours of a callout, depending on their importance. The problem was complicated by the presence of one-way roads and the possibility of vehicles refilling at various points. Various heuristics for this problem are described in Eglese & Li (1992), Howie & Aktin (1993), Eglese (1994) and Li & Eglese (1996).

The CARP+1D is a simplified version of this problem. Eglese & Li (1996) have produced an effective Tabu Search-based heuristic for the CARP+1D, called LOCSAR. This was used to obtain upper bounds for the test problems in Chapter 7.

In Stephenson (1996), a parcel delivery problem is studied which is somewhat similar. This is a node-routing problem (like the VRP), but with the added complication that parcels must be delivered by 9.30 a.m., 11 a.m., 1.00 p.m. or 5.30 p.m., depending upon the rate paid by the customer. Stephenson (1996) also resorts to an heuristic approach.

5.3. Complete Formulations.

One way to formulate multi-vehicle problems such as the VRP and CARP is to adapt the TSP formulation given in Section 2.4. The author calls such formulations 'complete' since they are effectively defined on a complete graph, with $O(|R|^2)$ edges, and do not exploit sparsity in G. With the VRP, the standard complete formulation is as follows (see, e.g., Gouveia, 1995): let the depot be vertex 1 and the other customers be located at vertices 2, ..., N+1. Let the cost of travel from i to j be c_{ij} and let the {0, 1} variable x_{ij} take the value one if and only if some vehicle moves between i and j in the solution. Then, assuming that use of a vehicle incurs a fixed cost C, we obtain:

$$\textit{Minimise} \quad \frac{C}{2} x(\delta(1)) + \sum_{i=1}^{N} \sum_{j=i+1}^{N+1} c_{ij} x_{ij}$$

Subject to:

$$x(\delta(i)) = 2 \qquad (i = 2, ..., N + 1) \qquad (5.1)$$

$$x(\delta(S)) \geq 2 K(S) \qquad (\forall S \subseteq V \setminus \{1\}: |S| \geq 2) \qquad (5.2)$$

$$x_{ij} \in \{0, 1, 2\} \text{ if } \{i, j\} \in \delta(\{1\}), \ \{0, 1\} \text{ otherwise} \qquad (5.3)$$

Constraints (5.1) are known as *degree* constraints. Constraints (5.2), known as *capacity* inequalities, are strengthened subtour elimination inequalities. In these, the term K(S) represents the minimum number of vehicles required to service S. For a particular S, K(S) can be found by solving a Bin-Packing Problem (see Section 5.7 and also Martello & Toth, 1990). The integrality conditions and bounds are given in (5.3). Note that x_{ij} is permitted to take the value 2 when ij is incident with the depot. This corresponds to a trip in which a vehicle leaves the depot, visits a single customer and returns immediately to the depot.

In the case of the CARP, the complete formulation is a little more complicated (see, e.g., Welz, 1994). Recall that the CARP is defined in terms of a graph G(V, E, R). Assume, without loss of generality, that the edges in E are ordered such that first |R| edges in E are the members of R. Define a complete graph G'(V', E'), with 1 + 2 |R| vertices. The depot is represented by vertex {1} in G', as it was in G. For i = 1, ..., |R|, let vertex i + 1 in V' be a copy of one particular end-vertex of the ith edge in E. Similarly, let vertex i + |R| + 1 in V' be a copy of the other end-vertex of the ith edge in E.

In E', for i = 1, ..., |R|, the edge {i + 1, i + |R| + 1} now represents the ith required edge in E. The other edges in E' represent shortest paths between the corresponding vertices in G. The set of these other edges will be denoted by E*. A {0, 1} variable x_{ij} is now defined for every edge in E*. Now let $S \subseteq V'\backslash\{1\}$ be called *unbroken* if it has the property that, for i = 2, ..., |R| + 1, $i \in S$ if and only if $i + |R| \in S$. That is, S corresponds to a set of required edges in R. The formulation becomes:

$$\textit{Minimise} \quad \frac{C}{2} x(\delta(1)) + \sum_{ij \in E^*} c_{ij}\, x_{ij}$$

Subject to:

$$x(\delta(i)) = 1 \qquad (i = 2, ..., 2\,|R| + 1) \qquad (5.4)$$

$$x(\delta(S)) \geq 2\,K(S) \qquad (\forall S \subseteq V'\backslash\{1\},\ S \text{ unbroken}) \qquad (5.5)$$

$$x_{ij} \in \{0, 1\} \qquad (ij \in E^*) \qquad (5.6)$$

Constraints (5.4) ensure that a vehicle arrives or departs from each end-vertex of each required edge exactly once. Constraints (5.5) are analogous to (5.2). Here, K(S) is the minimum number of vehicles required to service the required edges represented by the unbroken set S.

Constraints (5.6) ensure that all variables are binary. Unlike the case of the VRP, there is no need to allow variables to take the value 2. This is because, even if a vehicle only services a single edge, it will leave and return to the depot via different shortest paths.

Branch-and-bound will be necessary to solve either of the complete formulations (5.1) - (5.3) and (5.4) - (5.6). In both cases, the important issue is what kind of relaxation to use during the branch-and-bound procedure. This thesis is mainly concerned with the use of linear programming relaxations, but it is possible to use a *discrete* or *combinatorial* relaxation. In such a relaxation, the integrality condition is retained but some of the other constraints in the problem are relaxed.

For example, Fisher (1995) solves the VRP by relaxing (5.1) and some of (5.2), to obtain a minimum weight degree-constrained spanning K-tree subproblem. Miller (1995) solves the VRP by relaxing (5.2) to obtain a minimum weight b-matching subproblem. In both these cases, other constraints are incorporated via Lagrangean relaxation and subgradient optimisation. Hirabayashi, Saruwatari & Nishida (1992) solve the CARP by relaxing (5.5) to obtain a minimum weight perfect matching subproblem.

This thesis is mainly concerned with LP-based cutting-plane approaches. In the case of the CARP, only Welz (1994) attempts to solve the complete formulation (5.4) - (5.6). He does not study the integer polyhedron, but does present a simple separation heuristic for (5.5).

Far more time has been devoted to studying the complete formulation of the VRP. It is known (Harche & Rinaldi, 1995; Corberán, 1996) that the LP relaxation using constraints (5.2) tends to be extremely tight in practice, but that the separation problem for (5.2) is strongly NP-Hard. Therefore, not even a pseudopolynomial exact separation algorithm exists. Harche & Rinaldi (1995) resort to a heuristic. They use a max-flow algorithm to find violated constraints of the weaker form:

$$x(\delta(S)) \geq 2 \left(\sum_{i \in S} q_i / Q \right) \qquad (S \subseteq V / \{1\}: |S| \geq 2) \qquad (5.7)$$

and then replace any violated constraints found with their tighter equivalent.

Gouveia (1995) notes that as well as tightening (5.7) to become (5.2), it is possible to tighten (5.7) in a different direction to yield a new class of inequalities which in general neither dominates nor is dominated by the class (5.2):

$$x(\delta(S)) \geq 2 \left(\sum_{i \in S} q_i + \qquad \right) / Q \qquad (S \subseteq V / \{1\}: |S| \geq 2) \qquad (5.8)$$
$$i \in S, j \in$$

These can be seen as a kind of multistar inequality (see Araque, Hall & Magnanti, 1993 and Gouveia, 1995). We will call them *weak multistar* (WM) inequalities in Chapter 6, for reasons which will become clear in that chapter.

Gouveia (1995) gives an extended LP formulation of the VRP, with $O(N^2)$ variables and constraints, with the property that the projection of the associated polyhedron onto the space of the x variables is equal to the polytope defined by (5.1), (5.8) and the bounds in (5.3). Hall (1993) uses this extended formulation to show that the separation problem for (5.8) can be solved exactly in polynomial time, via a maximum flow problem.

An obvious application of the Hall (1993) result is to improve the Harche & Rinaldi (1995) heuristic: If a set S is found such that the WM (5.8) is violated, test to see whether the capacity inequality (5.2) is also violated (apparently, Harche & Rinaldi were unaware of the Hall result, which is due to be published in 1997). However, it is possible to do even better than this, as will become apparent in Section 6.4.

We now consider comb inequalities (see Section 2.4). Since the subtour elimination inequalities (2.2) can be strengthened in different ways by taking capacity considerations into account, it might be expected that the comb inequalities (2.4) could be strengthened in a similar way. This is indeed the case. The tightest variants known at the time of writing, due to Araque (1990), have the same form as (2.4) but have a larger right-hand-side. There is also a condition that the depot must be outside the handles and teeth. The details are not given here as the formulae for calculating the correct rhs are very complicated.

Other more exotic valid inequalities are known for the complete VRP formulation. For instance, Araque, Hall & Magnanti (1993) consider the special case in which all customers have identical demands. They define *multistar*, *partial multistar*, *ladybug* and *clique cluster* inequalities. Some of these were used by Araque et al. (1994) to give an effective branch-and-cut algorithm for this special case. Araque et al. used heuristic separation routines based on *shrinking* (defined in Section 2.7).

5.4. 'Sparse' Formulations.

Complete formulations have a very large number of variables, which in practice means that some kind of pricing / column generation scheme is necessary during the branch-and-bound routine. However, many multi-vehicle problems of interest are defined on road networks, which are highly sparse. In such cases, it seems desirable to exploit this sparsity by using a different formulation, since the number of variables required can be much smaller.

If the number of vehicles, K, to be used in the solution is specified beforehand, and K is sufficiently small, then it may be useful to use the following approach: For each edge e, let the general integer variable y_{ek} represent the number of times vehicle k traverses edge e (without servicing it). For each required vertex i (or required edge e), let the {0, 1} variable x_{ik} (or x_{ek}) take the value 1 if vehicle k services i (or e), 0 otherwise.

The resulting *sparse* formulation has K.|E| general integer variables and K.|R| binary variables. If K is sufficiently small, the number of variables will be far below that of the complete formulation. In practical applications, K will be small and fixed if, e.g., few vehicles are available and/or the cost C of each vehicle is very large.

One such sparse formulation has been proposed for the CARP by Belenguer & Benavent (1996). Let S denote a set of *non-depot* vertices, R(S) the set of required edges which have both end-vertices within S and $x_k(S)$, $x_k(\delta_R(S))$ and $y_k(\delta(S))$ denote $\sum_{e \in R(S)} x_{ek}$, $\sum_{e \in \delta_R(S)} x_{ek}$ and $\sum_{e \in \delta(S)} y_{ek}$ respectively. The formulation is:

Minimise $\sum_{k} \sum_{e \in E} c_e$

Subject to:

$$\sum_{k} x_{ek} = 1 \qquad (\forall e \in R) \qquad (5.9)$$

$$x_k(\delta_R(S)) + y_k(\delta(S)) \geq 2\, x_{fk} \qquad (\forall k, S, f \in R(S)) \qquad (5.10)$$

$$x_k(\delta_R(i)) + y_k(\delta(i)) \equiv 0 \bmod 2 \qquad (\forall k, i \in V) \qquad (5.11)$$

$$\sum_{e \in R} q_e\, x_{ek} \leq Q \qquad (\forall k) \qquad (5.12)$$

$$x_{ek} \in \{0, 1\}, \quad y_{ek} \geq 0 \quad \text{and integer} \qquad (5.13).$$

Belenguer & Benavent (1996) show that (5.9) are implicit equations of the associated polyhedron **CARP(G)** and that the *connectivity* inequalities (5.10) and the non-negativity conditions in (5.13) induce facets under mild conditions. They also introduce six other classes of valid inequality:

i) (5.9), (5.12) and the conditions on the x variables in (5.13) are the constraints of a *Generalised Assignment Problem* (GAP). Therefore any facets of the GAP polyhedron (Gottlieb & Rao, 1990) are also valid for **CARP(G)**. In fact they can be shown to induce facets of **CARP(G)**. There might even be extra implicit equations from the associated GAP.

ii) If S is such that $|\delta_R(S)|$ is odd, then $\sum_{k} y_k\ (\delta(S)) \geq 1$ is valid.

iii) If S is such that $\delta_R(S) \neq \varnothing$ and $F \subseteq \delta_R(S)$ is such that |F| is odd, then $x_k(\delta_R(S)\text{-}F) + y_k(\delta(S)) \geq x_k(F) - |F| + 1$ is valid for all k.

iv) If K(S) denotes the minimum number of vehicles required to service $R(S) \cup \delta_R(S)$, due to the capacity constraints, then the inequality $\sum_k y_k(\delta(S)) \geq 2\ K(S) - |\delta_R(S)|$ is valid. It will frequently induce a facet when $1 < K(S) < K$. When $K(S) = 1$, it is dominated by the connectivity inequalities (5.10). When $K(S) = K$, it is dominated by the following set of inequalities.

v) If, for some S, $K(S) = K$, then all vehicles must enter S. As a result, the inequalities $x_k(\delta_R(S)) + y_k(\delta(S)) \geq 2$ are valid for all k. These frequently induce facets.

vi) if $\sum_{e \in R(S) \cup \delta_R(S)} \alpha_e x_e \leq \beta$ is any valid inequality due to the GAP aspect of the problem, then $x_k(\delta_R(S)) + y_k(\delta(S)) \geq \frac{2}{\beta}\left[\sum_{e \in R(S) \cup \delta_R(S)} \alpha_e x_e\right]$ is also valid.

Although Belenguer and Benavent (1996) do not explicitly mention it, the connectivity inequalities (5.10) are a special case of (vi). To see this, note that the inequality $x_{fk} \leq 1$ is valid not only for **CARP(G)**, but also for the associated GAP.

When it comes to the issue of separation algorithms, Benavent (1995) reports the following results: Connectivity inequalities (5.10) can be separated by solving a series of maximum flow problems for each vehicle. The inequalities (ii) can be separated by solving a minimum odd cut problem. The edge-splitting strategy of Padberg & Rao (1982) (see Section 2.7) can be used to reduce the separation problem for the inequalities (iii) to a set of k minimum odd cut problems. Benavent (1995) is also currently working on heuristic separation algorithms for inequalities (i) and (iv) - (vi).

Belenguer & Benavent (1996) report that the inequalities (iv) are extremely important when attempting to increase the LP lower bound. They have adapted the maximum flow routine of Harche & Rinaldi (1995), described in the previous section, to separate (iv). They use this in conjunction with simple shrinking heuristics.

5.5. 'Very Sparse' Formulations.

If the number of vehicles K is fixed once more, even more economical formulations of the VRP and CARP are possible. These have only one general variable x_e for each $e \in E$ and will be termed *very sparse*. For the VRP, x_e represents the total number of times e is traversed (by any vehicle). A feasible x vector now represents a number of feasible vehicle routes superimposed on each other, what one might call a *K-route-set*. The formulation is, trivially,

Minimise $\sum_{e \in E} x_e$

Subject to:

x must represent a K-route-set. *(5.14)*

Cornuéjols & Harche (1993) examine this VRP formulation. They show that the associated polyhedron is full-dimensional and that the non-negativity inequalities induce facets. They also show that the capacity inequalities (5.2) are valid here and give conditions under which they induce facets.

A similar formulation would clearly be possible for the CARP. In the case of the CARP, however, it would be more sensible to interpret x_e as the total number of times e is traversed *without servicing*, by any vehicle. Then, if any required edge was such that the only time it was traversed was when it was being serviced, the associated variable would have a zero value. This would reduce the number of basic variables, making the LP relaxations easier to solve. To the author's knowledge, this CARP formulation has not been examined.

Since very sparse formulations have so few variables, especially for problems defined on road networks, they initially seem very promising. However, there are two serious drawbacks:

(i) To the author's knowledge, no explicit integer programme is known which has the K-route-sets as its feasible solutions. That is, the capacity inequalities are not sufficient to cut off integral x vectors which do not represent a K-route-set (see Cornuéjols & Harche, 1993, for a counter-example).

(ii) It appears to be strongly NP-hard to "untangle" a feasible solution to construct individual routes (the Bin Packing Problem could probably be reduced to it).

These difficulties imply that very sparse formulations are of little use for optimisation. However, if it is only a good *lower bound* which is required, they are potentially very useful. A very sparse formulation is used in Chapter 7 to obtain good lower bounds for the CARP + 1D.

5.6. Other Formulations.

Other formulations have been attempted for the VRP, which could also easily be adapted to the CARP. Only two will be mentioned here, one only briefly.

The first is the Set Partitioning formulation (see, e.g., Desrosiers et al., 1995). By the use of Dantzig-Wolfe decomposition, the integer programme (5.1) - (5.3) is reformulated as a Set Partitioning Problem (SPP). In the SPP, there is a variable for each feasible single-vehicle route, and a constraint for each customer ensuring that exactly one route visits that customer. The resulting SPP therefore has a huge number of variables but a small number of constraints, unlike the polyhedral approach which uses a small number of variables but a huge number of constraints.

To deal with the huge number of variables, an initial SPP relaxation is solved with only a small number of variables present. New variables (columns) are then generated as and when needed. To do this, a *column generation* algorithm is needed, which can utilise the values of the dual variables in the current SCP relaxation. This is analogous to a separation algorithm in the polyhedral approach, which utilises the values of the

primal variables. In this sense, the column generation approach can be regarded as dual to the polyhedral approach.

If the number of vehicles is too small, the SPP becomes highly degenerate. On the other hand, if the constraints in the problem are tight enough, it may be possible to solve the column generation problem by dynamic programming (Desrosiers et al., 1995). The SPP formulation therefore seems to be most appropriate for highly constrained problems involving a fairly large number of vehicles.

The second approach is to use Benders reformulation to convert (5.1) - (5.3) into a Generalised Assignment Problem (GAP) with a non-linear objective function (Fisher & Jaikumar, 1981). The objective function is then approximated by a linear function, which is successively improved by the addition of Benders cuts.

This approach appeared to be promising initially. However, the GAP is strongly NP-Hard and difficult to solve to optimality in practice. Moreover, the addition of Benders cuts results in a problem which is even harder to solve than the GAP. The method has not been as successful as the polyhedral and SPP approaches.

5.7. Combinatorial Lower Bounds.

All of the above formulations and valid inequalities for the CARP could be used to derive LP-based lower bounds, and, as will be shown in Chapter 7, it is not hard to adapt these to the CARP+1D also. However, there has also some been some work done on *combinatorial* lower bounding techniques. These rely on discrete relaxations rather than LP relaxations (see Section 5.3). For the sake of brevity, these will not be explored in detail here. However, a brief description of them is given, since one of them will relevant in Chapter 7.

To the knowledge of the author, all of the known combinatorial bounds for the CARP are summarised (and extended) in Benavent et al. (1992). The basic idea behind most of these bounds is that the optimal cost of a CARP solution must be at least as great as the sum of the following two components:

i) a matching of those vertices in G which are adjacent to an odd number of required edges, and

ii) an extra factor to account for the fact that each vehicle has to leave and return to the depot in order to service any required edge.

Hence, most of these bounding techniques involve the computation of a minimum cost perfect matching, on a network derived from G by adding extra edges and introducing extra copies of the depot vertex. They differ merely in the precise nature of the derived network, and some have refinements to take the demand of certain induced subgraphs into account also. Hirabayashi, Saruwatari & Nishida (1992) describe another variant of this general scheme, independently of Benavent et al.

Besides reviewing and extending these bounds, Benavent et al. (1992) also examine a combinatorial bound based on state-space relaxation, which can be easily computed using dynamic programming. However, this bound performed uniformly poorly compared to the matching-based bounds in their thorough computational experiments.

Li (1992) reviewed the best of the Benavent et al. bounds, and addressed the question of producing an analogous bound for the CARP+1D. This proved to be more tricky than might be expected. The difficulty is that, prior to invoking a perfect-matching algorithm, it is first necessary to have a good estimate of K_{min}, the minimum number of vehicles required to service the required edges. If the time deadline was not present, it would merely be necessary to solve a Bin Packing Problem. However, due to the deadline, it is not so easy.

For this reason, Li produced the following iterative procedure: an initial lower bound on K_{min} is formed by summing s_e over all $e \in R$ and dividing the result by T. A matching problem is then solved, assuming this fleet size, to obtain a lower bound on *the total time required, excluding return trips to the depot*. This may in turn imply that additional vehicles are needed. If so, the lower bound on K_{min} is updated and a new matching problem is solved, and so on.

Eventually, the lower bound on K_{min} cannot be further improved. At this point, a final matching problem is solved which, like the Benavent et al. CARP bounds, is in terms of cost rather than time. This yields the Li bound.

In Chapter 7, which concerns the CARP+1D, an experimental comparison will be made between the Li lower bound, a new LP-based lower bound due to the author, and an upper bound obtained from a tabu-search heuristic due to Eglese & Li (1996).

5.8. The Bin Packing Problem.

The Bin Packing Problem (BPP) is a well-known combinatorial optimisation problem which has been shown to be strongly NP-Hard by Garey & Johnson (1979) (that is, not even a pseudopolynomial algorithm exists for the BPP unless P = NP). An instance is given by a positive integer Q, representing the capacity of a single *bin*, and a set of positive integers q_1, ..., q_n, such that $q_i \leq Q$ for all $i \in N = \{1, ..., n\}$, representing the weights of n *items*. The task is to pack the items into as few bins as possible, given that the sum of the q_i of the items in any single bin cannot exceed Q.

Clearly, finding the minimum number of vehicles necessary to service the required edges in a CARP instance is equivalent to solving a BPP. Less obvious connections between the two problems, of a polyhedral nature, will be demonstrated in Chapter 6. References on heuristics, lower bounds and optimisation algorithms for the BPP can be found in Martello & Toth (1990) and Chao, Harper & Quong (1995). We will be primarily interested in lower bounding procedures.

An obvious lower bound on the number of bins required is the following quantity:

$$LB_0 = \left\lceil \sum_{i \in N} q_i \ / \ Q \right\rceil ,$$

where $\lceil U \rceil$ denotes the least integer greater than or equal to U. LB_0 is easily computed in O(n) time and, despite its simplicitly, works suprisingly well (Chao, Harper &

Quong, 1995), provided that there are very few "large" items (e.g., items with $q_i \geq Q/4$). A simple example where it fails is when $Q = n = 3$ and $q_1 = q_2 = q_3 = 2$. Here, $LB_0 = 2$, whereas 3 bins are required.

A refinement of LB_0, which we shall call LB_{MT}, can be found in Martello & Toth (1990). For any given λ, $0 \leq \lambda \leq Q/2$, define BIG(λ) as $\{i \in N: q_i > Q/2 + \lambda\}$ and MEDIUM(λ) as $\{i \in N: Q/2 - \lambda \leq q_i \leq Q/2 + \lambda\}$. Then each item in BIG($\lambda$) must be placed in a separate bin and the items in MEDIUM(λ) require still further bins. Hence, the quantity

$$f(\lambda) = |BIG(\lambda)| + \left\lceil \sum_{i \in MEDIUM(\lambda)} q_i / Q \right\rceil$$

is a valid lower bound. LB_{MT} is then defined as the maximum of $f(\lambda)$ over all λ. Since $f(Q/2) = LB_0$, LB_{MT} dominates LB_0. Nevertheless, Martello & Toth show that LB_{MT} can also be computed in O(n) time.

Although LB_{MT} dominates LB_0, there are still situations in which LB_{MT} performs badly. In fact, for any small $\varepsilon > 0$, there are examples in which the ratio of the optimum to LB_{MT} is at least $1.5 - \varepsilon$. For example, if $Q = 1000$, $q_i = 334$ for all i, and $n = 999$, the optimal packing uses 500 bins, yet $LB_{MT} = 334$.

A third lower bound is that of Leuker (1983), which will be denoted by LB_L. Leuker shows that if f(x) is a function such that the following holds:

$$\sum_{i \in N} q_i / Q \leq 1 \quad \textit{if and only if} \quad \sum_{i \in N} f(q_i/Q) \leq 1,$$

then $\left\lceil \sum f(q_i / Q) \right\rceil$ is a valid lower bound.

Several such functions are then defined and LB_L is defined as the maximum of the corresponding lower bounds. It also can be computed in linear time. To the author's knowledge, the worst-case behaviour of LB_L has not been analysed.

A fourth lower bound, which will be termed LB_{CHQ}, is found in Chao, Harper & Quong (1995). It is defined as the maximum of LB_0, LB_L and a third term which is specially defined to cope with "large" items; specifically, items with $q_i \geq Q/4$. LB_{CHQ} can be computed in O(n log n) time. The worst-case behaviour of LB_{CHQ} also appears to be unknown.

Finally, another lower bound is obtained by formulating the BPP as a Set Partitioning Problem (see Section 5.6), in which there is one variable for every possible filling of a single bin and one constraint for each item to ensure that it appears in a bin. The LP relaxation of this SPP can be solved via a column generation algorithm as outlined in the classic paper by Gilmore & Gomory (1961). The lower bound is then obtained by rounding up the optimal solution cost of this LP relaxation to the nearest integer. We will denote this lower bound by LB_{SPP}.

The worst-case behaviour of LB_{SPP} is also unknown, although it is extremely tight in practice. Unfortunately, it is also not known whether the column generation approach can be implemented to run in polynomial, or even pseudopolynomial, time.

6. The Capacitated Arc-Routing Problem.

6.1. Overview.

This chapter is a theoretical study of the three integer programming formulations of the CARP outlined in Sections 5.3 - 5.5. It is theoretical in that no computational experiments have been conducted. Rather, new valid inequalities and separation routines are given for each formulation (along with some comments which help to unify some of the known results).

Crucial to the derivation of many of the new valid inequalities is an understanding of a lower bounding procedure for the Bin Packing Problem devised by the author. This is outlined in the next section. The new valid inequalities are presented in Section 6.3. Separation algorithms for some of these inequalities are given in Section 6.4. The chapter concludes in Section 6.5 with some thoughts on which formulation(s) are appropriate to use in given situations.

6.2. On a Lower Bound for the Bin Packing Problem.

The Bin Packing Problem (BPP) was formally defined in Section 5.7, where the known lower bounds were described. As mentioned in Section 5.3, the BPP often arises as a subproblem when attempting to solve VRP or CARP instances. In this section, another lower bound is given for the BPP based on linear programming. The derivation of this lower bound will be utilised in the next section to yield new inequalities for the complete, sparse and very sparse CARP formulations outlined in Sections 5.4 - 5.5.

With any BPP instance, it is possible to define an associated *knapsack polytope* (see, e.g., Balas, 1975; Nemhauser & Wolsey, 1988; Zemel, 1989), by considering the feasible packings of any single bin. Pick one such bin and define the {0, 1}

variables y_i ($i \in N$) such that $y_i = 1$ if and only if item i is placed in that bin. Clearly, all feasible solutions to the BPP must satisfy $\sum_{i \in N} q_i y_i \leq Q$.

Accordingly, define the Knapsack Polytope **KP(Q)** as the convex hull of the incidence vectors of feasible packings of the single bin, that is

$$\mathbf{KP(Q)} = \text{Conv}\left\{ y \in \Re^n : \sum_{i \in N} q_i y_i \leq Q,\ y_i \in \{0,1\}\ (\forall i \in N) \right\}.$$

The following result is a corollary of a result of Leuker (1983). For completeness, a proof is given here:

Theorem 6.1: If $\sum_{i \in N} \alpha_i y_i \leq \beta$ *is any valid inequality for* ***KP(Q)****, with* $\beta > 0$*, then* $\left\lceil \sum_{i \in N} \alpha_i \ / \ \beta \right\rceil$ *is a valid lower bound on the minimum number of bins.*

Proof: Call α_i *the pseudo-weight of item i. Any single bin can contain a set of items with a pseudo-weight at most* β*. Yet the whole set of items has pseudo-weight equal to* $\sum_{i \in N} \alpha_i$.

Leuker's lower bound (LB_L, see Section 5.7) is obtained by invoking Theorem 6.1 for a small number of valid inequalities. But Theorem 6.1 yields many other lower bounds in addition to LB_L. For example, it yields the Martello-Toth (1990) bound, LB_{MT} (Section 5.7). This is because the inequality

$$\sum_{i \in BIG(\lambda)} Q y_i \ + \sum_{i \in MEDIUM(\lambda)} q_i y_i \ \leq \ Q$$

is valid for **KP(Q)**. To see this, note that if $y_p = 1$ for any $p \in BIG(\lambda)$, y_i must be zero for all $i \in MEDIUM(\lambda)$.

Hence, if LB* denotes the maximum lower bound obtainable by an application of Theorem 6.1, $LB^* \geq LB_L$ and $LB^* \geq LB_{MT} \geq LB_0$ both hold.

To see that at least the second inequality may be strict, consider the BPP instance with Q = 10, n = 5 and q_1, ..., q_5 equal to 6, 6, 3, 3 and 2 respectively. For this instance, $LB_{MT} = LB_0 = 2$. Yet, it can be shown that the inequality $2\,y_1 + 2\,y_2 + y_3 + y_4 + y_5 \leq 3$ is a facet of **KP(Q)** in this case. Then an application of Theorem 6.1 yields a bound of $\lceil 7 / 3 \rceil = 3$, which is in fact optimal.

The author does not know whether LB* can be computed in polynomial time. However, the following is true:

Theorem 6.2: LB can be computed in pseudopolynomial time.*

In order to prove this, we need the definition of a *1-polar* (see, e.g., Nemhauser & Wolsey, 1988). Given a polyhedron $P = \{x \in \Re^n : Ax \leq b\}$ containing the origin, the 1-polar of P is the polyhedron $\pi^1(P) = \{\pi \in \Re^n : \pi x \leq 1 \ \forall \ x \in P\}$. That is, for every inequality $\pi x \leq 1$ valid for P, there is a corresponding point in $\pi^1(P)$. It is known (see, e.g., Grötschel, Lovasz & Schrijver, 1988) that linear optimisation problems over $\pi^1(P)$ can be solved in polynomial (respectively, pseudopolynomial) time if and only if linear optimisation problems over P can be solved in polynomial (resp., pseudopolynomial) time. The proof of this relies on the ellipsoid method.

Proof of Theorem 6.2: Maximising $\sum_{i \in N} \alpha_i \ / \beta$ *subject to the condition that* $\sum_{i \in N} \alpha_i y_i \leq \beta$ *is valid for* ***KP(Q)*** *is equivalent to maximising* $\sum_{i \in N} \alpha_i$ *subject to the condition that* $\sum_{i \in N} \alpha_i y_i \leq 1$ *is valid for* ***KP(Q)****. This is a linear programming problem over* π^1(***KP(Q)***)*. This can be solved in pseudopolynomial time, since the Knapsack problem can be solved in pseudopolynomial time.*

Instead of using the ellipsoid method to optimise over π^1(**KP(Q)**), the algorithm of Boyd (1992) could be used. This involves the solution of a side-constrained network flow problem and will probably be more efficient in practice.

The author has wondered whether any dominance relation exists between LB* and the other two bounds LB_{CHQ} and LB_{SPP} mentioned in Section 5.8. In fact, it was recently shown by Wolsey (1997) that LB* = LB_{SPP}. To prove this, one uses the fact that $\pi^1(\pi^1(\mathbf{KP(Q)})) = \mathbf{KP(Q)}$ to formulate the optimisation problem over the 1-polar as a linear programming problem with a constraint for every feasible packing of a single bin. The dual of this LP has a variable for every feasible packing, and a constraint for each item. It turns out to be the Set Partitioning formulation.

6.3. New Valid Inequalities for the CARP.

In this section, new valid inequalities are presented for the complete, sparse and very sparse formulations outlined in Sections 5.3 - 5.5. A thorough reading of those sections is a prerequisite for understanding the results given in this section.

To start with, inequalities for the complete formulation (5.4) - (5.6) are presented. First, note that constraints (5.4) and (5.6) resemble the constraints of a perfect matching problem (Edmonds, 1965). Hence, an analogue of *blossom* inequalities can be defined: Let a given $S \subset V'\backslash\{1\}$ be called *broken* if it is not unbroken. If S is broken, then some set $F \neq \varnothing$ of required edges lies within $\delta(S)$. It is clear that the blossom inequality:

$$x(\delta(S)) \geq 1 \qquad (\forall S \subset V'\backslash\{1\},\ S \text{ broken},\ |F| \text{ odd}) \qquad (6.1)$$

is valid for the complete CARP formulation.

Now define the *enlargement* of a broken set S, *en(S)*, to be the minimal unbroken set $S' \subseteq V$ such that $S \subseteq S'$ holds. Letting K(*en(S)*) represent the minimum number of vehicles required to service the edges within *en(S)*, the capacity inequality (5.5) corresponding to the set *en(S)* is

$$x(\delta(en(S))) \geq 2\,K(en(S)).$$

Using the degree equations (5.1) for the vertices in *en(S)* \ S, this capacity inequality can be re-written as:

$$x(\delta(S)) \geq 2\,K(en(S)) - |F| + 2 \sum_{i \in S, j \in en(S) \setminus S} x_{ij} + \sum_{i,j \in en(S), i \neq j} x_{ij} .$$

This will dominate the blossom inequality (6.1) if $|F| \leq 2\,K(en(S)) - 1$. Thus, a necessary condition for (6.1) to be facet-defining is that $|F| \geq 2\,K(en(S)) + 1$. The author does not know whether this condition is also sufficient.

Now recall the definition of the graph G'. Define a 'shrunk graph', SG', by shrinking each edge in E' \ E*, until its end-vertices coincide. The required edges have effectively become required vertices and a solution to (5.4) - (5.6) is now analogous to a solution to the VRP formulation (5.1) - (5.3). In a natural way, valid inequalities for the resulting VRP yield valid inequalities for the CARP. The author calls these *VRP-derived* inequalities. Viewed in this way, (5.5) are VRP-derived capacity inequalities.

In this way, it is not hard to adapt the weak multistar (WM) inequalities (5.8) to the CARP. However, it is possible to obtain a stronger class of inequalities. To show this, it suffices to give a class of inequalities for the complete VRP formulation which dominates (5.8). The VRP-derived version of these new inequalities will then automatically dominate the VRP-derived WM inequalities in an obvious way.

Theorem 6.3: Given a VRP instance, define

$$\textbf{KP(Q)} = conv\left\{ y \in \Re^N : \sum_{i=2}^{N+1} q_i y_i \leq Q,\ y_i \in \{0,1\} \right\}.$$

Note that ***KP(Q)*** *is a Knapsack polyhedron whose extreme points correspond to feasible assignments of customers to a single vehicle.*

If $\sum_{i=2}^{N+1} \alpha_i y_i \leq \beta$, *with* β *and all* $\alpha_i > 0$, *is valid for* ***KP(Q)***, *then the inequalities*

$$x(\delta(S)) \geq 2\,(\sum_{i \in S} \alpha_i + \sum_{i \in S, j \in V \setminus (S \cup \{1\})} \alpha_j x_{ij}) / \beta \qquad (S \subseteq V \setminus \{1\}: |S| \geq 2) \qquad (6.2)$$

are valid for the complete VRP formulation.

Proof: Since no single vehicle can visit a set of vertices whose α *coefficients sum to more than* β*, any feasible VRP solution must also be feasible for a transformed VRP in which demands are* α_i *and vehicle capacity is* β*. Validity of (6.2) then follows from validity of (5.8).*

The discovery of inequalities (6.2) was inspired by the derivation of the Bin Packing lower bound given in the previous section. There is also a slight resemblance to the sixth class of valid inequalities for the sparse CARP formulation (Section 5.4), due to Belenguer & Benavent (1996). The author calls (6.2) *knapsack-tightened multistar* (KTM) inequalities.

The class of KTM inequalities dominates the WM class, but still neither dominates nor is dominated by the class of capacity inequalities (5.2). It is clear that a KTM inequality cannot induce a facet of the VRP polyhedron unless it is derived from a facet of **KP(Q)**, since linear combinations of **KP(Q)** inequalities lead to KTM inequalities which are linear combinations of other KTM inequalities. However, the converse may not apply; it is conceivable that a KTM derived from a facet of **KP(Q)** may nevertheless fail to induce a facet. The author believes that, in general, it will be difficult to derive sufficient conditions for KTM inequalities to induce facets.

An interesting observation is that the standard subtour elimination inequalities (2.1) are in fact a special case of (6.2), since, for i = 2, ..., N+1, $y_i \leq 1$ is valid for **KP(Q)**. Hence (5.2) and (6.2) are distinct valid generalisations of (2.1), neither of which subsumes the other.

All of these results transfer over to the CARP. The resulting *VRP-derived KTM* inequalities have similar properties to their VRP counterparts.

Now we consider the sparse formulation of the CARP, due to Belenguer & Benavent (1996). Three new classes of valid inequality will be presented. Each of the first two classes subsumes two of the six classes outlined in Section 5.4.

Theorem 6.4: Given $S \subseteq V \setminus \{1\}$ and $F \subseteq \delta_R(S)$, $|F|$ odd, along with a set of vehicles $H \subseteq \{1, ..., K\}$, the following inequality is valid for ***CARP(G)****:*

$$\sum_{k \in H} \left[x_k(\delta_R(S) - F) + y_k(\delta(S)) \right] \geq \sum_{k \in H} x_k(F) - |F| + 1 \qquad (6.3).$$

Proof: If any vehicle not in H services any edge in F, the right-hand-side becomes zero or less and the inequality is trivially valid. On the other hand, if only the vehicles in H service the edges in F, the rhs becomes 1, which is valid since the vehicles in H must cross $\delta(S)$ an even number of times.

The discovery of (6.3) was inspired by the discovery of the various parity inequalities for the RPPDC given in Chapter 4. The author calls (6.3) *general parity* (GP) inequalities. Note that when $|H| = K$ and $F = \delta_R(S)$, the x variables can be eliminated due to equations (5.9) and one obtains

$$\sum_{k \in K} y_k(\delta(S)) \geq 1 ;$$

which is the second set of valid inequalities described in Section 5.4. On the other hand, when $H = \{k\}$, i.e. $|H| = 1$, one obtains $x_k(\delta_R(S) - F) + y_k(\delta(S)) \geq x_k(F) - |F| + 1$ for all k; i.e., the third set of valid inequalities in Section 5.4.

Theorem 6.5: let $S \subseteq V \setminus \{1\}$ be such that at least K(S) vehicles are required to cover $R(S) \cup \delta_R(S)$ due to the vehicle capacity constraints, and let $H \subseteq \{1, ..., K\}$ be any subset of vehicles such that $K - K(S) < |H| \leq K$. Then the following inequality is valid for ***CARP(G)****:*

$$\sum_{k \in H} \left[x_k(\delta_R(S)) + y_k(\delta(S)) \right] \geq 2 \, [\, |H| - K + K(S) \,] \qquad (6.4).$$

Proof: At least K(S) vehicles must have $x_k(\delta_R(S)) + y_k(\delta(S)) \geq 2$ *in any feasible solution. At least* $|H| - K + K(S)$ *of the vehicles in H must enter S since there are only* $K - |H|$ *other vehicles available. Hence at least* $|H| - K + K(S)$ *vehicles in H must satisfy* $x_k(\delta_R(S)) + y_k(\delta(S)) \geq 2$ *in any feasible solution and the result follows.*

The author calls (6.4) *minimum crossing* inequalities. Note that when |H| = K, the x variables can be eliminated due to equations (5.9) and one obtains

$$\sum_k y_k\,(\delta(S)) \;\geq\; 2\,K(S) - |\,\delta_R(S)\,|,$$

the fourth set of valid inequalities described in Section 5.4. On the other hand, when K(S) = K and |H| = 1, one obtains $x_k(\delta_R(S)) + y_k(\delta(S)) \geq 2$ for all k; i.e., the fifth set of valid inequalities in Section 5.4.

Finally, note that the general parity inequalities with |H| = K resemble the R-odd cut inequalities (2.14) for the RPP, apart from the summation over the K routes. This is due to the fact that, if all of the K routes are superimposed on one another, then an RPP tour must result. Hence, it is possible to adapt all of the inequalities discussed in Chapter 3 to the CARP. More formally:

Theorem 6.6: If any inequality of the form $\sum_{e \in E} \alpha_e x_e \geq \beta$ *is valid for* ***RPP(G)****, then* $\sum_{k \in K} \sum_{e \in E} \alpha_e y_{ek} \geq \beta$ *is valid for* ***CARP(G)****.*

However, these RPP-derived inequalities cannot be expected to induce facets unless the demands of the edges are small relative to the vehicle capacity.

It is helpful to summarise and consolidate the above results on the sparse formulation in the following proposition:

Proposition 6.7: Every single known valid inequality for ***CARP(G)*** *falls into one of the following six classes:*

(a) The non-negativity inequalities for the y variables in (5.13),

(b) The inequalities derived from the Generalised Assignment Problem (the first set of valid inequalities of Belenguer and Benavent (1996)),

(c) The inequalities derived from the interaction of the GAP and routing subproblems (the sixth class of inequalities of Belenguer & Benavent (1996).

(d) Generalised parity inequalities (6.3),

(e) Minimum crossing inequalities (6.4),

(f) RPP-derived inequalities.

To see this, note that the equations (5.9), the inequalities (5.12) and the non-negativity conditions on the x variables in (5.13) are a special case of (b). Also, it was already shown (in Section 5.4) that the connectivity inequalities (5.10) are a special case of (c).

Finally, we briefly consider a very sparse CARP formulation. For each $e \in E$, define a general integer variable x_e, representing the total number of times e is traversed, *without servicing*, by any vehicle. A feasible x vector now represents a *K-route-set*, which in the case of the CARP means a set of K feasible vehicle routes superimposed on each other, but not including servicing. The formulation is, trivially,

Minimise $\sum_{e \in E} x_e$

Subject to:

x must represent a K-route-set. *(6.5)*

This very sparse formulation has a strong relationship with the RPP formulation of Corberán & Sanchis (1994), discussed in Section 2.6: Since each individual vehicle route must define an Eulerian multigraph, the routes superimposed upon each other must also define an Eulerian multigraph. Therefore, any solution to (6.5) must also be a solution to (2.11) - (2.13). Hence, it can immediately be said that all of the valid inequalities discussed in Chapter 3 (path-bridge, generalised binested, etc.) are also valid for the very sparse CARP formulation. In addition, the following result holds:

Theorem 6.8: Given an $S \subseteq V \setminus \{1\}$, let K(S) denote the minimum number of vehicles required to service the edges having at least one end-vertex within S in G. Then:

$$x(\delta(S)) \geq 2\,K(S) - |\delta_R(S)| \qquad (\forall S \subseteq V \setminus \{1\}: |S| \geq 2) \qquad (6.6)$$

is valid for the very sparse CARP formulation.

Proof: At least K(S) vehicles must enter S. That is, $\delta(S)$ must be crossed at least 2 K(S) times. Hence, the vehicles must cross $\delta(S)$ without servicing an edge in $\delta_R(S)$ on at least 2 K(S) - $|\delta_R(S)|$ occasions.

There is a strong link between (6.6) and the minimum crossing inequalities (6.4) with |G| = K, valid for the sparse formulation.

The main theoretical conclusion from the results of this section is that there is a complicated system of relationships between the three different formulations of the CARP; and, moreover, all three are also related to the Bin Packing Problem and the Rural Postman Problem. With more work, it might be possible to describe these connections with greater precision, but this is outside of the scope of this thesis.

6.4. Separation Routines.

In this section, various exact separation algorithms are given for some of the inequalities discussed in the previous section. Where the author has been unable to find useful separation algorithms, some comments are given which may aid the search for them.

We begin with the complete formulation. The easiest class of inequalities to deal with are the blossom inequalities (6.1). The result of Padberg & Rao (1982) implies that these can be separated in polynomial time by solving a minimum weight odd cut problem.

All of the other known valid inequalities for the complete CARP formulation are *VRP-derived* (see the previous section); that is, they can be derived by shrinking required edges to obtain a corresponding VRP instance. A careful rereading of the argument in the previous section shows that this result carries over to the separation problem: An LP relaxation of the CARP violates an inequality in a given class of VRP-derived inequalities if and only if the corresponding LP relaxation of the corresponding VRP violates the corresponding inequality. Hence, any separation algorithm for a given class of VRP inequalities automatically yields a separation algorithm for corresponding class of CARP inequalities. We can therefore restrict our attention to separation algorithms for the VRP.

As mentioned in Section 5.3, comb inequalities are valid for the complete VRP formulation (5.1) - (5.3). Therefore, any heuristic for comb inequalities, such as the necklace method of Section 2.7, can be used with the VRP (and therefore CARP). Moreover, if a comb inequality is violated or nearly violated, the rhs can then be strengthened according to the results of Araque (1990).

Since 2-matching inequalities are a special case of comb inequalities (see Section 2.4), it follows that 2-matching inequalities are valid for the VRP. These can be separated exactly in polynomial time as outlined in Padberg & Rao (1982). Again, it might be possible to strengthen the resulting inequality.

The capacity (5.2) and KTM (6.2) inequalities will be treated simultaneously, and three heuristic separation routines will be suggested. First, however, consider the simpler (still apparently NP-Hard) problem of detecting violation of (5.2) or (6.2) when the set S has already been fixed beforehand. In the case of (5.2), it is recommended that the Bin Packing lower bound of Section 6.2 be used unless the demands are measured to a very high precision, when one of the simpler lower bounds reviewed in Section 5.7 should be used. In the case of (6.2), the problem is to find a valid inequality for **KP(Q)** which maximises the violation for the given S. This is an optimisation problem over π^1(**KP(Q)**), which can be solved in pseudo-polynomial time (See Section 6.2).

Now consider the general problem once more. The first heuristic separation algorithm is to shrink all edges with $x_{ij} = 1$, then compute a minimum cut in the resulting graph. The set S on the opposite shore to the depot is then a candidate to check for violation of (5.2) or (6.2) as explained in the previous paragraph. This may work well in the initial stages of a cutting-plane algorithm, but will fail later on.

The second heuristic is to use the Hall (1993) max-flow algorithm to find a most violated weak multistar inequality (5.8). The resulting set S is again a candidate for violating (5.2) and (6.2). The third heuristic is to fix the valid inequality beforehand (perhaps by optimising over π^1(**KP(Q)**)), and then run the Hall algorithm with this particular inequality taking the place of the capacity constraint as implied by the proof of Theorem 6.3.

The sparse CARP formulation will now be briefly considered, using the classification (a) - (f) given in Proposition 9.7. The case (a) is trivial. Any known separation algorithms for the Generalised Assignment Problem (GAP) could be used for case (b). Since Gottlieb & Rao (1990) show that facets of the individual knapsack constraints are also facets of the GAP polyhedron, this includes any known separation routines for the Knapsack problem. As for case (c), the strong resemblance between between these inequalities and the KTM inequalities (6.2) suggests that the heuristics suggested above for (6.2) could be adapted to them.

The main difficulty lies with case (d), the GP inequalities (6.3). A plausible approach might be to examine each possible set G of vehicles, using the edge-splitting strategy of Padberg & Rao (1982) with each set. The problem with this is that there are 2^K-1 possible sets of vehicles, which leads to an running time which is exponential in K. The author is not aware of any way around this difficulty (a similar problem may exist for the RPPDC formulation in Chapter 4, as alluded to in section 4.4). Worse still, a similar problem seems to occur with case (e), the MC inequalities (6.4).

Fortunately, when it comes to case (f), the RPP-derived inequalities, all of the separation results of Section 2.8 and Chapter 3 carry over to the sparse CARP formulation.

Finally, when it comes to the very sparse formulation, the situation is much simpler. Any known separation algorithm for the RPP (Section 2.8 and Chapter 3 again) can be used directly to separate RPP-derived inequalities, and the only other important class of inequalities is (6.6), the analogue of the capacity inequalities. Some simple but effective heuristics for the separation of (6.6) will be considered in Section 7.4.

6.5. Comparing Formulations.

In this final section, an attempt is made to compare the three different CARP formulations based on the preceding discussion.

The complete formulation has the disadvantage that it requires a large number of variables, therefore making a column generation scheme necessary. However, it has two advantages:

(i) very good bounds can be obtained merely by adding capacity inequalities to the LP relaxation (Welz, 1994)

(ii) any work on separation algorithms for the VRP transfer directly to the CARP as outlined in the previous section.

The sparse formulation (Belenguer & Benavent, 1996) has two advantages:

(i) it has few variables provided that the number of vehicles K is fairly small;

(ii) it is not too difficult to adapt it to handle heterogeneous vehicle fleets, i.e. vehicles with different capacities, costs, etc.

However, it has three drawbacks:

(i) K, or at least a tight upper bound on K, must be determined beforehand;

(ii) a bewildering variety of cutting-planes appear necessary to obtain a feasible solution;

(iii) there may not be a separation algorithm for the GP and MC inequalities which runs in a time polynomial in K, and there is no obvious heuristic either.

The very sparse formulation, being an aggregation of the sparse formulation over all vehicles, circumvents the second and third difficulties presented by the sparse formulation. The LP relaxations are also very easy to solve as they have so few variables. However, it appears to be strongly NP-Hard to "untangle" the optimal K-route-set into separate routes, or even to recognise when an integral LP relaxation is a feasible K-route-set.

A tentative conclusion, based on the above analysis, is as follows: if K is large and an optimal or near-optimal solution is required, the complete formulation is preferable. If K is small (e.g., less than 5), but an optimal or near-optimal solution is still required, then the sparse formulation is preferable. Finally, no matter what the value of K, if all that is required is a very good lower bound, then the very sparse formulation is appropriate.

7. The Capacitated Arc Routing Problem with a Deadline.

7.1. Overview.

This chapter is concerned with the CARP+1D as defined in Section 1.4. It should be clear from Chapters 5 and 6 that a variety of different formulations and solution techniques could be applied to the CARP+1D. It should also be clear that, whichever formulation is chosen, extremely sophisticated software will be needed if the goal is to solve realistic CARP+1D instances to optimality.

In this chapter, a less ambitious approach is taken which is easier to implement. In Section 7.2, a formulation is given for the CARP+1D which, while not facilitating the search for an optimal solution, does allow good lower bounds to be obtained fairly easily. This formulation is of the *very sparse* type (see the previous chapter). In Section 7.3, some (fairly) simple valid inequalities are presented for this formulation. Some heuristic separation algorithms are described in Section 7.4, along with the novel idea of *dynamic tightening*, which has implications for many other problems beside the CARP+1D. Finally, in Section 7.5, the lower bounding procedure is tested on a few test problems from Eglese & Li (1996). The resulting lower bounds are then compared with the lower bounds of Li (1992), and also with the best known upper bounds found by the LOCSAR heuristic program as reported by Eglese (1996).

7.2. A Very Sparse Formulation.

It would be possible to formulate the CARP+1D using only |E| general integer variables, one for each edge of the network. However, when the author attempted this, it proved extremely difficult to derive any classes of valid inequality which took the time deadline into account. It also proved impossible to derive a useful lower bound on the minimum number of vehicles required, or to take vehicle utilisation costs into account.

It turned out to be a great deal easier to borrow the *time phase* concept from Section 4.2. In the context of the CARP+1D, it is best to regard each vehicle route as made up of *two* phases as follows: in phase 1, the vehicle leaves the depot and services a number of required edges; in phase 2, which must begin no later than time T, the vehicle returns to the depot via a shortest path.

Defining c^*_v as the cost of the shortest path from vertex v to the depot, and a *route-set* as a set of feasible single-vehicle routes superimposed on each other, this viewpoint leads naturally to the following very sparse formulation:

x_e = no. times edge e is traversed without servicing, by any vehicle, during phase 1.

z_v = no. vehicles beginning phase 2 at vertex v.

$$\textit{Minimise} \quad \sum_{e \in E} c_e x_e \; + \; \sum_{v \in V} (c^*_v + C) z_v$$

Subject to:

x and z represent a route set *(7.1)*

Note that the objective function incorporates not only the travel costs, but also a fixed cost C of utilising any single vehicle. It is also easy to modify this formulation to allow for variants of the CARP+1D in which the number of vehicles is limited/fixed a priori: simply add an additional equality/inequality with the sum of the z variables on the left-hand-side.

Just as for the very sparse VRP and CARP formulations (see Section 5.5), the author was unable to produce an explicit representation of (7.1) in terms of inequalities and integrality conditions. However, if all that is needed is a *lower bound on the optimum solution value*, then an explicit formulation is not necessary. All that is needed is a useful set of valid inequalities.

7.3. Valid Inequalities.

Throughout this section, just as in Chapters 4 to 6, it is assumed that the depot is located at vertex 1 for ease of exposition.

The author has discovered a large number of different valid inequalities implied by the condition (7.1). To present these, the following notation will be needed: Let $\delta(S)$ denote the set of edges in G having exactly one end-vertex in S and E(S) denote the set of edges in G having exactly two end-vertices in S. Also let $\delta_R(S)$ denote $\delta(S) \cap R$ and $E_R(S)$ denote $E(S) \cap R$.

For any $F \subseteq E$, let x(F) denote $\sum_{e \in F} x_e$; for any $S \subseteq V$, let z(S) denote $\sum_{v \in S} z_v$.

The first class of inequalities are the *trivial* inequalities, $x_e \geq 0$ and $z_v \geq 0$. These clearly hold for all e and v, respectively.

A second, less obvious class, is based on the fact that a vehicle cannot end phase 1 at a vertex unless it has already reached that vertex.

Theorem 7.1: For any $S \subseteq V \setminus \{1\}$, the following 'Balancing' inequality is valid for the CARP+1D:

$$x(\delta(S)) \geq z(S) - |\delta_R(S)| \qquad (7.2)$$

Proof: Consider any feasible solution to (7.1). If $z(S) \leq |\delta_R(S)|$, the rhs of (7.2) reduces to zero or less and the inequality is trivially true. Now suppose $z(S) > |\delta_R(S)|$. Since z(S) vehicles finish phase 1 while within S, at least z(S) vehicles must have entered S during phase 1. This implies that the cutset $\delta(S)$ must have been crossed on at least z(S) occasions in phase 1. On at most $|\delta_R(S)|$ of these occasions, this was by a vehicle servicing an edge. This leaves at least $z(S) - |\delta_R(S)|$ occasions on which a vehicle crossed $\delta(S)$ without servicing in phase 1. Thus, $x(\delta(S))$ must be at least $z(S) - |\delta_R(S)|$.

The third class of inequalities is simply an analogue of the capacity inequalities (5.2) for the ordinary CARP (see Sections 5.3 and 5.5):

Theorem 7.2: For any $S \subseteq V \setminus \{1\}$, let $K_{CT}(S)$ represent the minimum number of vehicles required to service the required edges having at least one end-vertex within S, taking into account the capacity and time deadline restrictions. Then the following 'minimum crossing' inequality is valid for the CARP+1D:

$$x(\delta(S)) + z(S) \geq 2\,K_{CT}(S) - |\delta_R(S)| \qquad (7.3)$$

Proof: If $2\,K_{CT}(S) \leq |\delta_R(S)|$, the rhs reduces to zero or less and the inequality is trivially true. Now assume that $2\,K_{CT}(S) > |\delta_R(S)|$. At least $K_{CT}(S)$ vehicles must enter and leave S. This implies that the cutset $\delta(S)$ must be crossed on at least $2\,K_{CT}(S)$ occasions in total. On at most $|\delta_R(S)|$ occasions, this could be by a vehicle servicing an edge in phase 1. This leaves at least $2\,K_{CT}(S) - |\delta_R(S)|$ occasions on which a vehicle must cross $\delta(S)$ without servicing. This could be done during phase 1 or phase 2.

The author calls these *minimum crossing* inequalities, rather than *capacity* inequalities, to emphasise the fact that time as well as capacity is being taken into account. Note that, for a given S, the correct computation of $K_{CT}(S)$ is extremely difficult. This implies that the separation problem is also likely to be extremely difficult. A possible way around this difficulty will be discussed in the next section.

The next theorem shows that any known inequalities for the RPP can be adapted to the CARP+1D:

Theorem 7.3: Let G' be identical to G, but with an additional non-required edge {v, 1} for each vertex v. Define an RPP instance on G' and formulate it as in Chapter 3. If any inequality in the form:

$$\sum_{e \in E} \alpha_e x_e \; + \; \sum_{v \in V} \beta_v x_{\{v,1\}} \; \geq \; \gamma$$

is valid for the resulting polyhedron ***RPP(G')****, then the inequality*

$$\sum_{e \in E} \alpha_e \, x_e \; + \; \sum_{v \in V} \beta_v \, z_v \; \geq \; \gamma$$

is valid for the CARP+1D.

Proof: Given any feasible solution to (7.1), construct a multigraph G(V, E*) as follows: Let E* consist of* x_e *copies of each* $e \in E \setminus R$, $x_e + 1$ *copies of each* $e \in R$ *and, for each* $v \in V$, z_v *copies of edge {v, 1}. Note that G* may contain one or more 'loops', viz., edges having {1} at both ends. By construction, G* is Eulerian and E* contains at least one copy of each* $e \in R$. *But this means that G* represents a feasible solution to the RPP defined on G'. The inequality follows from the construction of G*.*

There is a strong similarity between the proof of Theorem 7.3 and the proof of Theorem 4.1 in Section 4.3. As an application of Theorem 7.3, we have:

Corollary 7.4: If $S \subseteq V \setminus \{1\}$ *is such that* $|\delta_R(S)|$ *is odd, then the R-odd cut inequality*

$$x(\delta(S)) \; + \; z(S) \; \geq \; 1 \qquad (7.4)$$

is valid for the CARP+1D.

For sets S with $|\delta_R(S)|$ odd, the inequalities (7.3) and (7.4) will both be applicable. If $2\,K(S) \leq |\delta_R(S)|$, the latter will dominate the former; if $2\,K(S) \geq |\delta_R(S)| + 2$, the former will dominate the latter; if $2\,K(S) = |\delta_R(S)| + 1$, they will be identical. Whatever the case, however, the balancing inequality (7.2) will always apply and will not be dominated by either.

Due to the difficulty of finding an appropriate right-hand-side for (7.3), the author was led to search for another useful class of inequalities which take the time deadline into account. This led to the discovery of the following set of inequalities:

Theorem 7.5: For a given t, $0 < t < T$, let V^t be the set of all vertices i such that the time taken for a vehicle leaving the depot to reach i is t or more. For any $S \subseteq V^t$, let $\delta^t(S)$ represent the (possibly empty) set of edges with one end-vertex in S and the other in $V^t \setminus S$ and let $\delta^t_R(S)$ represent $\delta^t(S) \cap R$. The 'T-radius' (TR) inequality:

$$x(\delta(S)) + z(S) \geq \frac{2}{T-t} \left(\sum_{e \in E_R(S) \cup \delta^t_R(S)} t_e + \sum_{e \in E(S) \cup \delta^t(S)} t_e x_e \right) - |\delta_R(S)| \qquad (7.5)$$

is valid for the CARP+1D.

Proof: Since t units of time must have elapsed before any given vehicle can enter V^t, the time taken by any such vehicle while within $E(V^t)$ can be at most T - t. Hence, if exactly k vehicles enter V^t in some feasible solution to (7.1), the total time spent by the k vehicles within V^t must be no more than k(T-t). This implies that the inequality

$$\sum_{e \in E_R(S) \cup \delta^t_R(S)} t_e + \sum_{e \in E(S) \cup \delta^t(S)} t_e x_e \leq k(T - t)$$

must hold for this feasible solution. Now observe that, for this same feasible solution, the inequality $x(\delta(S)) + z(S) \geq 2k - |\delta_R(S)|$ must also hold (see the proof of Theorem 7.2). These two inequalities imply that (7.5) holds for the feasible solution in question and therefore that (7.5) is valid in general.

By analogy with Theorem 6.3 of Chapter 6, the inequalities in (7.5) could be tightened by defining an auxiliary polytope for the CARP+1D instance, the extreme points of which represented feasible routes for a single vehicle. The resulting

tightened T-radius (TTR) inequalities would dominate (7.5), but have a much more complex separation problem. This is not pursued here in detail because a) the ordinary TR inequalities appear to perform adequately, b) the minimum crossing inequalities (7.3) appear to be far more important for improving the LP lower bound and c) ordinary TR inequalities can be separated in polynomial time. Point (c) is discussed in the next section.

7.4. Separation Algorithms.

In this section, some exact and heuristic separation algorithms are given for the inequalities mentioned in the previous section.

The separation problem for the balancing inequalities (7.2) is fairly easy to solve. Recall that for these to be valid, the depot is not permitted to be in the vertex set S. Now, rewrite the inequality as:

$$x(\delta(S)) + |\delta_R(S)| + z(V - S) \geq z(V) \qquad (7.6).$$

For a given LP relaxation, the rhs of (7.6) is a constant. Hence, a set S yielding a most violated balancing inequality is one which minimises the lhs of (7.6). Now construct a graph G' with vertex set $V \cup \{1^*\}$, where $\{1^*\}$ represents a copy of the depot, and an edge set equal to E plus an additional edge $\{v, 1^*\}$ for every $v \in V$. Give the edges in R a weight of $x_e + 1$, the edges in $E \setminus R$ a weight of x_e and each new edge $\{v, 1^*\}$ a weight z_v. By sending a maximum flow from $\{1\}$ to $\{1^*\}$, a minimum weight cut such that $\{1\}$ and $\{1^*\}$ lie on opposite shores can be found, and G' has been constructed so that the set of vertices lying on the same shore of the cut as $\{1^*\}$ is a set S minimising the lhs of (7.6).

Because the graph G' has $|V| + 1$ vertices and $|E| + |V|$ edges, this yields the following result:

Theorem 7.6: The separation problem for balancing inequalities can be solved in the time taken to solve a maximum flow problem on a graph with O(|V|) vertices and O(|E|) edges.

Now consider the minimum crossing (MC) inequalities (7.3). These are very important inequalities for obtaining strong lower bounds but, as mentioned previously, the appropriate rhs is hard to compute. Therefore exact separation is out of the question. However, call an inequality a *weak* MC inequality if it is like (7.3) but has a lower bound on $K_{CT}(S)$ in the rhs expression. At least three heuristics present themselves for finding weak MC inequalities.

The first heuristic takes the capacity constraint into account, but not the time deadline. Note that the quantity

$$\sum_{e \in E_R(S) \cup \delta_R(S)} q_e \; / Q \qquad (7.7)$$

is a lower bound on $K_{CT}(S)$. The maximum flow algorithm of Harche & Rinaldi (1995) could then be adapted to find a violated inequality of the weaker form:

$$x(\delta(S)) + z(S) \geq 2 \sum_{e \in E_R(S) \cup \delta_R(S)} q_e \; /Q - |\delta_R(S)|$$

Once the set S is found, any valid lower bound on the Bin Packing Problem could be used to generate a weak MC inequality.

As discussed in Section 6.2, the lower bound (7.7) could be strengthened by considering an appropriate Knapsack polyhedron. This would lead to a different variant of the first heuristic.

The second heuristic is much easier, but requires that at least one weak MC inequality has already been added to the LP:

i) Pick any vertex set $S \subset V \setminus \{1\}$ for which a weak MC is currently binding or near-binding.

ii) Put the elements of $V \setminus (S \cup \{1\})$, in any order, into a list L.

iii) Add head(L) to S. Calculate a lower bound on $K_{CT}(S)$ for the new S.

iv) If the weak MC for the new S has a smaller surplus than, or is more violated than, the weak MC for the previous set S, remove head(L) from L and go to (v). Otherwise, remove head(L) from both L and S and go to (vi).

v) If $S = V \setminus \{1\}$, stop. Otherwise, go to (ii).

vi) If L is empty, stop. Otherwise, go to (iii).

When calculating the lower bound on $K_{CT}(S)$ in step (iii), one could merely use a Bin Packing lower bound. However, since S has *grown*, and $K_{CT}(S)$ clearly increases as S grows, one could use the previous value of $K_{CT}(S)$ if this is larger than the Bin Packing bound.

Other variants on this second heuristic include a version in which S shrinks at each step, or even a sophisticated local search technique which allows the insertion and deletion of vertices according to various rules.

Note that the second heuristic utilises the surplus values in the current LP relaxation, rather than the x and z vectors alone. The third heuristic does the same, but in a more sophisticated fashion: Choose, from the current LP tableau, a binding or near-binding T-radius inequality, or a weak MC inequality with a surplus not equal to an integer multiple of two. Calculate a lower bound K* (such as a Bin Packing bound), on $K_{CT}(S)$. Temporarily add the following equality to the LP relaxation:

$$x(\delta(S)) + z(S) = 2K^* - |\delta_R(S)|.$$

If the resulting LP is feasible, remove the equality from the relaxation and stop. No weak MC was found. Otherwise, keep incrementing K* by 1, adjusting the new

equality accordingly, until feasibility is attained. At this point, the equality can be changed into a "greater-than-or-equal-to" inequality. It is now a new weak MC.

This third heuristic differs radically from most known separation algorithms for routing problems. The author calls the underlying technique *dynamic tightening*. It is conceivable that a similar approach would work for classes of inequality known for other hard routing problems. Note also that the procedure can be further extended if a good feasible solution to the routing problem is known. In this situation, it is not necessary for the LP to become infeasible; it is only necessary for the cost of the LP relaxation to exceed that of the upper bound.

No such sophisticated ideas are necessary for the R-odd cut inequalities (7.4). Simply construct a graph G' with vertex set V and an edge set equal to E plus an additional edge {v, 1} for every v ∈ V. Give the edges in E a weight of x_e and each new edge {v, 1} a weight z_v. For each v ∈ V, label v odd if and only if $|\delta_R(\{v\})|$ is odd, otherwise label it even. Find a minimum weight odd cut in G' (Padberg & Rao, 1982). If the weight of the cut is less than 1, then setting S to the shore of the cut not containing the depot yields a violated R-odd cut inequality.

Let N_{ODD} be the number of odd vertices in G'. The Padberg & Rao (1982) minimum weight odd cut algorithm involves the solution of N_{ODD} maximum flow problems and therefore:

Theorem 7.7: The separation problem for R-odd cut inequalities can be solved in the time taken to solve N_{ODD} maximum flow problems on a graph with O(|V|) vertices and O(|E|) edges.

Finally, the T-radius inequalities (7.5) are considered. These can be re-written as:

$$x(\delta(S)) + z(S) + |\delta_R(S)| + \frac{2}{T-t}\left(\sum_{e\in E_R(V^t-S)} s_e + \sum_{e\in E(V^t-S)} t_e x_e\right)$$

$$\geq \frac{2}{T-t}\left(\sum_{e\in E_R(V^t)} s_e + \sum_{e\in E(V^t)} t_e x_e\right) \qquad (7.8)$$

Now, for a *fixed* value of t, the rhs of (7.8) is a constant. Hence, a set S minimising the lhs of (7.8) will yield a most violated TR inequality for this particular t. To find such a set, construct a graph G' as follows: copy G and add a copy of the depot, {1*}. Give each $e \in R$ a weight of $x_e + 1$, each $e \in E \setminus R$ a weight of x_e. For each $v \in V$, add an edge {v, 1} of weight z_v. Split each $e \in E(V^t)$ into two halves (see Sections 2.7 and 4.5) by inserting a new *splitting vertex* in the middle. Each half should retain the previous weight. For each splitting vertex v thus created, corresponding to some original $e \in E(V^t)$, add an edge {v, 1*} of weight $2 (s_e + t_e x_e) / (T - t)$. Now, *shrink* $V \setminus V^t$ (see Section 2.7). Finally, send a maximum flow from {1} to {1*} to find a minimum weight cut separating {1} from {1*}. The graph G' has been constructed so that the set of vertices on the same shore of the cut as {1*} is the desired set S.

Now recall that $0 < t < T$. Within this range, $2 / (T - t)$ increases monotonically with t. Therefore, it is only worth looking for violated TR inequalities for values of t such that the vehicle takes exactly t units of time to travel from {1} to some vertex in G. That is, only O(|V|) values of t need be considered. Hence:

Theorem 7.8: The separation problem for T-radius inequalities can be solved in the time taken to solve O(|V|) maximum flow problems on a graph with O(|E|) vertices and O(|E|) edges.

Since this running time is unattractive, the author also recommends a local search heuristic along the same lines as that for the weak MC inequalities outlined above.

It was said in Section 7.3 that it might be possible to tighten the TR inequalities using advanced polyhedral arguments. It may not, however, be worthwhile seeking separation algorithms for tightened TR inequalities, since the potential weakness of TRs can be alleviated by dynamic tightening as outlined above.

7.5. Computational Experiments.

In this section, the above results are used to find lower bounds for a number of CARP+1D instances. Since, to the author's knowledge, the only previous work done on the CARP+1D is that cited by Eglese & Li (1996), it was thought useful to use some of the CARP+1D examples which were examined therein. The advantage of this is that there are already useful lower and upper bounds available for these instances.

These problems are based on real-life *winter gritting* problems, on rural road networks. Two road networks were used to define the CARP+1D instances, labelled EAST and SOUTH. The EAST network has 77 vertices and 111 edges, of which 84 are required. The SOUTH network has 140 vertices and 203 edges, of which 160 are required. It is also larger physically.

In reality, the EAST and SOUTH networks contain a few one-way streets. In Eglese & Li (1996), as in this chapter, these were treated as two-way for simplicity.

Two main problems are defined on each network: EASTA and EASTB differ only in the edge lengths and times, SOUTHA and SOUTHB differ only in the location of the depot. Each of these four problems in turn has two variants depending upon whether the capacity constraint is included or ignored. Furthermore, for each of the eight resulting versions, two different objective functions were minimised: First, the total number of vehicles and second, the total distance travelled. This led to sixteen subproblems in total.

The precise data for these problems are given in Appendix 2. To give the reader a feel for these problems, the following characteristic features are noted:

- There is a high proportion of vertices of degree 3, compared to urban road networks. Other vertices have degrees ranging from 1 (on the periphery) to 8 (a dual carriageway intersection).

- There are a few parallel edges, i.e. edges having identical end-vertices. This does not affect the validity of any of the inequalities discussed.

- The network in the SOUTH examples is planar. The network in the EAST examples would be planar if one edge (representing an underpass) was removed.

- There are a small number of components in the subgraph induced by the required edges and the depot. This is because it is the motorways and A-roads which tend to require service, and these tend to be connected to each other. For the EAST problems, there are only 2 components; for SOUTHA, there are 6 and for SOUTHB there are 5.

- Edge costs c_e are measured in metres. They normally, but not always, obey the triangle inequality. Moreover, they follow a roughly lognormal distribution.

- The time deadline, T, is set at precisely 1 hr 55 minutes ≅ 1.917 hrs.

- Times are correlated with costs. To be precise, $t_e = c_e/40000$ and $s_e = c_e/25000$. That is, the vehicle travels at 40kph when traversing, but 25kph when servicing an edge.

- Demands too are correlated with costs. For simplicity, it is assumed that $q_e = c_e$ if e $\in$ R, 0 otherwise. This means that the vehicle capacity, Q, can be expressed as 40 km.

The cutting-plane lower bounding procedure uses all of the inequalities (balancing, minimum crossing, R-odd cut, T-radius), described in Section 7.3. However, as mentioned in Section 7.4, only a 'weak' variant of the minimum crossing inequalities could be used. It was not found necessary to use other classes of inequality implied by Theorem 7.3, due to the small number of connected components of required edges.

When attempting to minimise the number of vehicles used, a single additional inequality, not discussed in Section 7.3, was found to be essential and was therefore added to the LP relaxation. The author calls it the *vehicle* inequality. It is:

$$z(V) \geq \left(\sum_{e \in R} s_e + \sum_{e \in E} t_e x_e \right) / T \quad .$$

The quantity in the brackets represents the total amount of time spent in phase 1, summed over all vehicle routes. The validity of the inequality follows from the fact that each vehicle has only T time units available in phase 1.

In the cutting-plane algorithm, the initial LP relaxation consists of a subset of the R-odd cut inequalities - those in which i is merely a single vertex - plus the vehicle inequality where appropriate. For the uncapacitated problem variants, a subset of the TR inequalities is also added, by randomly selecting some large vertex sets S. On the other hand, for the capacitated problem variants, a subset of the weak MC inequalities is added, by randomly selecting some large vertex sets S.

For the problems under consideration, it turned out to be easy to identify violated balancing and R-odd cut inequalities by eye. For the weak MC and TR inequalities, the local search separation heuristics outlined above appeared to perform adequately.

The results of the experiments are displayed in Tables 7.1 - 7.4 on pages 113 & 114. Table 7.1 shows the results for the uncapacitated versions, when minimising the number of vehicles. The first column shows the Li (1992) lower bound. The next four columns show the number of *binding* balancing (7.2), minimum crossing (7.3), R-odd cut (7.4) and T-radius (7.5) inequalities at the optimum solution of the LP relaxation. The next column gives the lower bound found by the cutting-plane approach and the final column gives the best known upper bound from Eglese (1996).

Table 7.2 is analogous to Table 7.1, but for the capacitated versions. An additional column gives the Bin Packing lower bound (see Section 5.8). Table 7.3 is equivalent to Table 7.1, but the objective is now to minimise total distance travelled. There is an additional column on the right, showing the percentage of the gap between the Li bound and the upper bound closed by the cutting-planes. Finally, Table 7.4 is like Table 7.3, except that it concerns the capacitated versions.

The total number of inequalities actually generated during the algorithm was normally between two or three times the number of binding inequalities, apart from the case of the T-radius inequalities, as explained below.

Instance	Li LB	Bal	MC	Odd	TR	LP LB	UB
EASTA	6	4	8	43	1	6	6
EASTB	6	4	8	44	1	6	7
SOUTHA	9	9	11	111	0	10	10
SOUTHB	9	15	8	103	0	9	10

Table 7.1: Number of vehicles: uncapacitated variants.

Instance	Li LB	BPP LB	Bal	MC	Odd	TR	LP LB	UB
EASTA	6	6	4	10	45	0	6	7
EASTB	6	6	4	11	44	0	6	7
SOUTHA	9	9	6	25	109	0	10	11
SOUTHB	9	9	14	16	101	0	9	10

Table 7.2: Number of vehicles: capacitated variants.

Three things are apparent from Tables 7.1 and 7.2. First, the Bin Packing lower bound equalled the Li (1992) bound in all cases. This must be merely coincidendental, as different capacities or deadlines would give different results. Second, few T-radius inequalities were binding at the LP optimum. This is because dynamic tightening converted some of them into minimum crossing inequalities. Third, the cutting-planes did little to improve the lower bounds on the number of vehicles required. It is hard to know, however, whether this is due to a poor quality lower bound or a poor quality upper bound. The author has no opinion either way.

Instance	Li LB	Bal	MC	Odd	TR	LP LB	UB	%Closed
EASTA	89371	3	9	42	3	133584	159671	62.89
EASTB	93074	4	9	45	3	138936	170413	59.30
SOUTHA	209207	2	17	86	7	287609	381944	45.39
SOUTHB	154815	1	10	93	15	274760	319770	72.71

Table 7.3: Distance: uncapacitated variants.

Instance	Li LB	Bal	MC	Odd	TR	LP LB	UB	%Closed
EASTA	89371	1	17	45	0	147144	166233	75.16
EASTB	93074	1	14	41	0	153982	170413	78.75
SOUTHA	209207	8	24	84	5	293260	395776	45.05
SOUTHB	154815	2	11	89	15	276641	319770	73.85

Table 7.4: Distance: capacitated variants.

The results from Tables 7.3 and 7.4 are rather more encouraging. In all cases it was possible to make significant improvements to the Li (1992) bound.

Because EAST is physically small, inequalities derived from the capacity aspect of the problem were more important than those derived from the time aspect. For the SOUTH problems, the reverse holds. Thus, the difference between the LP lower bounds of the capacitated and uncapacitated versions is much larger for the EAST problems than for the SOUTH problems. This also explains why many more T-radius inequalities are binding for the SOUTH problems than for the EAST problems.

8. Conclusions and Suggestions.

8.1. Comments on Previous Chapters.

The scope of this thesis has been fairly wide: Four different (though related) Arc-Routing problems have been examined, and an attempt has been made to present formulations, valid inequalities and/or separation algorithms for each of them, along with (limited) computational experiments.

From a theoretical perspective, the research effort has been very fruitful, not least in terms of papers accepted for publication. The author was particularly pleased with the considerably increased understanding of the RPP polyhedron presented in Chapter 3, particularly the discovery of the path-bridge inequalities (PBIs). Moreover, the separation algorithm for PBIs worked better than expected. This has led the author to believe that truly large-scale RPP instances will become solvable in the very near future.

The results found on the RPPDC and CARP (Chapters 4 and 6), were almost as encouraging. The RPPDC formulation worked very well and an automated branch-and-cut algorithm would probably have no problem solving realistic instances to optimality. The discovery of an array of valid inequalities for the various CARP formulations, along with the analysis of the Bin Packing lower bound, in Chapter 6, led to an increased understanding of these problems and also shed light on the standard Vehicle Routing Problem.

The results obtained for the CARP+1D were more mixed. Although the LP-based lower bound on distance was invariably a strong improvement over the Li (1992) bound, it had been hoped that the T-radius inequalities would frequently lead to an improved lower bound on the number of vehicles required. The remaining gaps between the LP-based lower bounds and the LOCSAR solutions indicate that there is room for improvement either to the lower bound or to LOCSAR. The author believes that it is likely to be the lower bound that is underperforming here.

The other drawback to the research is the somewhat limited nature of the computational experiments. Given more time, and the availability of commercial branch-and-cut software, it might have been possible to devise fully automated branch-and-cut algorithms for all four problems. This would have permitted a thorough testing of the formulations on more comprehensive sets of test problems.

Inevitably, due to time limitations, some theoretically interesting issues were left untouched. These are the subject of the next section.

8.2. Some Open Research Problems.

During the research, the author had to make decisions as to which issues were worth exploring in detail. Many other apparently less important questions were therefore left unanswered. Some of these are nevertheless of theoretical interest, and are therefore mentioned in this section. Presented below is a list of open questions, for each of the four routing problems, which seem most relevant from the author's point of view:

The RPP:

i) Are there polynomial-time exact separation algorithms for any of the new classes of inequality outlined for the RPP in Chapter 3?

ii) Can the recent results of Carr (1996), which essentially provide polynomial separation routines for a plethora of TSP inequalities, be transferred to the RPP? This may be possible via the RPP to SGTSP transformation given in Chapter 3.

iii) Can the different RPP inequalities be ranked according to their relative power to improve the LP lower bound, as Goemans (1995) did for various TSP inequalities? Here, the appropriate measure of the strength of a class of inequalities appears to be the maximum percentage improvement obtainable by adding the inequalities to the LP relaxation consisting of non-negativity, connectivity and R-odd cut inequalities.

The RPPDC:

i) Is there a truly polynomial (i.e., not exponential in L) separation routine for the most general form of the parity inequalities for the RPPDC (Section 4.4)? If the answer is no, can the separation problem be proved to be NP-Hard?

ii) Alternatively, is there a different formulation for the RPPDC which avoids the pitfalls of the author's formulation, yet still exploits the structure of the problem?

The CARP:

i) Under what conditions do blossom inequalities (6.1) induce facets of the complete CARP formulation? Is it sufficient for $|F| \geq 2\,K(en(S)) + 1$ to hold?

ii) Under what conditions do knapsack-tightened multistar (KTM) inequalities induce facets of the complete CARP formulation?

iii) Is there a pseudopolynomial-time exact separation algorithm for the KTM inequalities? The author believes this may be possible using convex quadratic programming.

iv) Is it true that all facets of the polyhedron associated with the complete CARP formulation are either blossom inequalities or VRP-derived inequalities?

v) Is there some way of adapting the very sparse formulation so as to avoid the problem of "tangled" routes, yet retain the advantage of having few variables? Perhaps it could be possible to add new variables (and constraints) dynamically, to progressively untangle the routes.

vi) Could the CARP be solved more effectively by a Set Partitioning approach as done for the VRPTW in Desrosiers et al. (1995)? It does seem certain that the column generation subproblem could not be solved effectively by dynamic programming.

The CARP+1D:

i) Is there a separation routine for the T-radius inequalities (Section 7.3) with a smaller worst-case running time?

ii) The T-radius inequalities are rather weak, in a theoretical sense. Can a useful separation routine be found for any of the tightened forms of the T-radius inequalities?

iii) Is there some other useful formulation and/or class of valid inequalities which could lead to stronger LP lower bounds?

8.3. More General Routing Problems.

Another limitation of the research was the (deliberate) restriction to four particular undirected arc-routing problems: the RPP, RPPDC, CARP and CARP+1D. One direction for further research is to attempt to extend the results to routing problems of greater generality.

A common generalisation of all four problems, as mentioned in Section 1.4, is the *Capacitated Arc-Routing Problem with Deadline Classes* or CARPDC. The formulations for the RPPDC (Chapter 4) and the CARP+1D (Chapter 7) suggest a formulation for the CARPDC which has separate variables for each route and each time phase. The author in fact attempted this during the research but found that the resulting formulation had a vast array of possible valid inequalities, none of which appeared to have an easily solvable separation problem. Hence this line of research was abandoned. Some other formulation, however, might prove more viable.

Another obvious generalisation is to permit a mixture of vertices and edges to require service, rather than just edges. The four problems would then become the *GRP* (General Routing Problem), *GRPDC*, *CGRP* and *CGRP+1D*. The first of these has been studied by Orloff (1974), Lenstra & Rinnooy-Kan (1976) and Corberán & Sanchis (1996). A variant of the last is studied in Laporte, Nobert & Desrochers

(1985). It is in fact not difficult to generalise all of the polyhedral results in this thesis to these problems. This has been demonstrated in Letchford (1997, 1996a) for the case of the GRP and in Letchford & Eglese (1996b) for the CGRP and CGRP+1D.

A third generalisation would be to allow *mixed networks*, i.e., networks in which there is a mixture of one-way and two-way streets. Nobert & Picard (1996) have devised an effective algorithm for the *Mixed Chinese Postman Problem*; the more general *Mixed RPP* is currently being studied by Corberán & Sanchis (1997).

Finally, and most challenging, it would be interesting to study problems in which general time deadlines are allowed. That is, problems in which each vertex or edge requiring service is permitted to have its own deadline. The author believes that a *complete* formulation (see Section 5.1 ff.) would be most appropriate here, and that it may be possible to produce a useful generalisation of the T-radius inequalities for such a formulation.

8.4. A Recent Breakthrough.

Very recently (at the time of writing), Caprara & Fischetti (1996) have made an important breakthrough in the search for efficient separation algorithms for some special classes of valid inequalities. It is worth describing this in some detail here, as the author believes that it will lead to a significant increase in the size of routing problems which will be solvable in the coming years.

Suppose a problem is formulated as an integer programme (IP) with integral coefficients. Without loss of generality, it can be assumed that the constraint set is of the form $AX \leq B$, where A is an n by m matrix, X is the column vector of variables, and B is the column vector of right-hand sides (a greater-than-or-equal-to inequality can be multiplied by -1, and an equality written as two inequalities). Assume also that the non-negativity conditions are included explicitly in the constraint set. It is often possible to produce useful valid inequalities, or even facets, in the following way: Choose some of the original inequalities in the formulation, sum them together

anddivide the resulting inequality by 2. If the resulting inequality has the property that all coefficients on the left-hand-side are integral, but the right-hand-side is equal to an integer plus 1/2, then it can be strengthened by rounding down the rhs.

For example, summing the inequalities $x_1 + x_2 \leq 1$, $x_1 + x_3 \leq 1$ and $x_2 + x_3 \leq 1$ yields $2\ x_1 + 2\ x_2 + 2\ x_3 \leq 3$; dividing this by two yields $x_1 + x_2 + x_3 \leq 3/2$, and rounding down yields $x_1 + x_2 + x_3 \leq 1$, which is clearly valid if the variables are restricted to be integral.

Caprara & Fischetti call a valid inequality which can be found by the above method a *{0, 1/2}-Chvátal-Gomory cut*, or *{0, 1/2}-cut* for short (as explained in the paper, the procedure for yielding a {0, 1/2}-cut is a special case of a more general technique for producing valid inequalities, known as Chvátal-Gomory rounding). They then review a variety of integer programming formulations of combinatorial optimisation problems, and note that in many cases, there are known facet-inducing inequalities which are {0, 1/2}-cuts.

They show that the set of all possible {0, 1/2}-cuts corresponds to the set of solutions to the following system of linear congruences modulo 2:

$$\begin{bmatrix} A^T \\ B^T \end{bmatrix} Y \equiv L$$

where Y is a column vector of 0-1 variables, one for each of the m inequalities in the formulation, and D is a column vector with n + 1 entries, the first n being 0 and the last one being 1.

Now suppose that one is also given an LP relaxation for the IP, i.e., an X vector satisfying $AX \leq B$. From this it is easy to construct a column vector S representing the values of the m slack variables. The problem of finding a most violated {0, 1/2}-cut then becomes equivalent to the problem of finding a minimum weight solution to the system of congruences (that is, the problem of minimising S^TY subject to (8.1)).

Using this result, Caprara & Fischetti prove that the separation problem for {0, 1/2}-cuts is NP-Hard in general (it contains the max-cut problem as a special case). However, it can be solved in polynomial time for some special cases, yielding new polynomial-time separation routines, for certain valid inequalities and facets, for a wide variety of IP problems.

Now say that a {0, 1/2}-cut is *maximally violated* if all of the inequalities used in its derivation are binding in the LP relaxation. That is, it is violated by as much as possible given that the LP relaxation satisfies $AX \leq B$. Although not explicitly observed by Caprara & Fischetti, it is clear that a {0, 1/2}-cut is maximally violated if and only if the reduced system

$$\begin{bmatrix} A'^T \\ B'^T \end{bmatrix} Y' \equiv D' \mod 2, \qquad (8.2)$$

which only has variables for *binding* inequalities, has a solution. Provided that there are only a polynomial number of such binding inequalities, the system (8.2) can be easily solved in polynomial time by a form of Gaussian elimination.

The relevance of this to routing problems is as follows: it has been known for some time (see, e.g., Nemhauser & Wolsey, 1988), that the comb inequalities for the TSP polytope are {0, 1/2} cuts with respect to the subtour elimination inequalities, degree conditions and bounds. Recall from Section 2.7 that it is known that there are at most $O(N^2)$ sets S of vertices satisfying $x(\delta(S)) = 2$ in an LP relaxation of a TSP on N vertices. Since there are only $O(N^2)$ bounds and O(N) degree conditions, this implies that the matrix A' has $O(N^2)$ rows. Therefore we can conclude that:

Theorem 8.1: If W is the set of all {0, 1/2}-cuts derivable from the TSP formulation (2.1) - (2.3), then:

a) W contains all comb inequalities, and

b) Given an LP relaxation, the presence of a maximally violated inequality in W can be detected in polynomial time.

This result significantly generalises the result of Fleischer & Tardos (1996), which only guarantees the detection of a maximally violated comb inequality when the support graph of the LP relaxation is planar. The author conjectures that a similar technique will work to identify maximally violated 2-regular PBIs for the RPP. This would immediately lead to a polynomial algorithm to detect the presence of maximally violated general PBIs, as outlined in Section 3.4.

Just as exciting would be the application of a similar idea to the VRP and CARP: One of the arguments given by Araque (1990) to derive strengthened comb inequalities for the complete VRP formulation is in fact an application of the {0, 1/2}approach. Hence, one might be able to find *maximally violated*, *strengthened* comb inequalities for the VRP in polynomial time. From the results in Section 6.3, this would immediately apply to the complete CARP formulation also.

The author would thus like to end on an optimistic note: it seems entirely probable that the coming years will lead to exciting breakthroughs in the search for new formulations, valid inequalities and separation algorithms. Together with the growing availability of branch-and-cut software, it seems that the range of real-life problems which are amenable to solution to optimality will be significantly extended in the near future.

References.

Applegate, D., Bixby, R.E., Chvátal, V. & Cook, W. (1995) Finding cuts in the TSP. **DIMACS Technical Report 95-05**, Bell Communications Research, Morristown, New Jersey, 07690, USA.

Araque, J.R. (1990) Lots of combs of different sizes for vehicle routing. **Discussion paper 9074**, Centre for Operations Research and Econometrics, Catholic University of Louvain, Belgium.

Araque, J.R., Hall, L.A. & Magnanti, T.L. (1993) Capacitated trees, capacitated routing and associated polyhedra. **Working paper**, Dept. of Math. Sciences, John Hopkins University.

Araque, J.R., Kudva, G., Morin, T.L. & Pekny, J.F. (1994) A branch-and-cut algorithm for vehicle routing problems. **Ann. of Ops Res.**, **50**, 37-59.

Assad, A.A. & Golden, B.L. (1995) Arc-routing methods and applications. In **Ball et al. (1995b)**.

Balas, E. (1975) Facets of the knapsack polytope. **Math. Prog., 8**, 146-64.

Balas, E., Fischetti, M. & Pulleyblank, W.R. (1995) The precedence-constrained asymmetric travelling salesman polytope. **Math. Prog.**, **68**, 241-265.

Ball, M.O., Magnanti, T.L., Monma, C.L. & Nemhauser, G.L. (Eds.) (1995a) **Network Models**. Handbooks on Ops Research and Man. Science, Vol. 7. Amsterdam: Elsevier.

Ball, M.O., Magnanti, T.L., Monma, C.L. & Nemhauser, G.L. (Eds.) (1995b) **Network Routing**. Handbooks on Ops Research and Man. Science, Vol. 8. Amsterdam: Elsevier.

Benavent, E. (1995) **Private communication**.

Belenguer, J.M. & Benavent, E. (1996) The capacitated arc-routing problem: valid inequalities and facets. **Eur. J. Opl Res.**, forthcoming.

Benavent, E., Campos, V., Corberán, A. & Mota, E. (1992) The capacitated arc-routing problem: lower bounds. **Networks**, **22**, 669-690.

Benczúr, A.A. (1994) Augmenting undirected connectivity in RNC and in randomised $O(n^3)$ time. **Proc. 26th Ann. Symp. Th. of Comp.**, 658-667.

Boyd, E.A. (1992) A pseudopolynomial network flow formulation for exact knapsack separation. **Networks**, **22**, 503-14.

Boyd, S.C. & Cunningham, W.H. (1991) Small travelling salesman polytopes. **Math. of Ops Res.**, **16**, 259-271.

Caprara, A. & Fischetti, M. (1996) {0, 1/2}-Chvátal-Gomory cuts. **Math. Prog.**, **74**, 221-235.

Carr, R.D. (1995) Separating clique-tree and bipartition inequalities having a fixed number of handles and teeth in polynomial time. To appear in **Math. of Ops Res.**

Carr, R.D. (1996) Separating over classes of TSP inequalities defined by 0 node-lifting in polynomial time. In W.H. Cunningham, S.T. McCormick and M. Queyranne (Eds.) **Proc. of the 5th Int. IPCO Conference**. Lecture Notes in Computer Science, Vol. 1084. Springer-Verlag, NY.

Chao, H.-Y., Harper, M.P. & Quong, R.W. (1995) A tight lower bound for optimal bin packing. **Ops Res. Letts**, **18**, 133-138.

Christophides, N. (1973) The optimal traversal of a graph. **Omega, 1**, 719-32.

Christofides, N., Campos, V., Corberán, A. & Mota, E. (1981) An algorithm for the rural postman problem. **Technical Report**, Imperial College, London.

Christoff, T. & Reinelt, G. (1996) Combinatorial optimisation and small polytopes. **Top (J. of the Spanish Stats and Opl Res. Soc.)**, **4**, 1-64.

Clochard, J.M. & Naddef, D. (1993) Using path inequalities in a branch-and-cut code for the symmetric travelling salesman problem. In G. Rinaldi & L. Wolsey (Eds.) **Proc. of the 3rd Int. IPCO Conference**. Centro Ettore Majorana; Erice, Italy.

Corberán, A. (1996) **Private communication**.

Corberán, A. & Sanchis, J.M. (1994) A polyhedral approach to the rural postman problem. **Eur. J. of Opl Res.**, **79**, 95-114.

Corberán, A. & Sanchis, J.M. (1996) The general routing problem polyhedron: facets from the RPP and GTSP polyhedra. Submitted to the **Eur. J. of Opl Res.**

Corberán, A. & Sanchis, J.M. (1997) The mixed rural postman problem polyhedron. **Working paper**, Dept. of Stats and O.R., University of Valencia.

Cornuéjols, G., Fonlupt, J. & Naddef, D. (1985) The traveling salesman on a graph and some related integer polyhedra. **Math. Prog.**, **33**, 1-27.

Cornuéjols, G. & Harche, F. (1993) Polyhedral study of the capacitated vehicle routing problem. **Math. Prog.**, **60**, 21-52.

Dantzig, G., Fulkerson, D.R. & Johnson, S. (1954) Solution of a large-scale travelling salesman problem. **Ops Research**, **2**, 393-410.

Desrosiers, J., Dumas, Y., Solomon, M.M. & Soumis, F. (1995) Time-constrained routing and scheduling**.** In **Ball et al. (1995b)**.

Dror, M., Stern, H. & Trudeau, P. (1987) Postman tour on a graph with precedence relations on arcs. **Networks**, **17**, 283-94.

Dumas, Y., Desrosiers, J., Gélinas, E. & Solomon, M.M. (1995) An optimisation algorithm for the travelling salesman problem with time windows. **Ops Research**, **43**, 367-371.

Edmonds, J. (1965) Maximum matching and a polyhedron with 0-1 vertices. **J. of Res. of the Nat. Bur. of Standards**, **69B**, 125-130.

Edmonds, J. & Johnson, G.L. (1973) Matchings, Euler tours and the Chinese postman problem. **Math. Prog.**, **5**, 88-124.

Eglese, R.W. (1994) Routing winter gritting vehicles. **Disc. App. Math.**, **48**, 231-244.

Eglese, R.W. (1996) Private communication.

Eglese, R.W. & Li, L.Y.O. (1992) Efficient routing for winter gritting. **J. Opl. Res.Soc., 43**, 1031-4.

Eglese R.W. & Li L.Y.O. (1996) A tabu search based heuristic for arc routing with a capacity constraint and time deadline. In I.H. Osman & J.P. Kelly (Eds.) **Metaheuristics: Theory and Applications**. Kluwer, Boston, pp.633-650.

Eiselt, H.A., Gendreau, M. & Laporte, G. (1995a) Arc routing problems, part 1: the Chinese postman problem. **Ops Research**, **43**, 231-242.

Eiselt, H.A., Gendreau, M. & Laporte, G. (1995b) Arc routing problems, part 2: the rural postman problem. **Ops Research**, **43**, 399-414.

Fisher, M.L. (1995) Vehicle routing. In **Ball et al. (1995)**.

Fischer, M.L. & Jaikumar, R. (1981) A generalised assignment heuristic for vehicle routing. **Networks**, **11**, 109-124.

Fleischer, L. & Tardos, E. (1996) Separating maximally violated comb inequalities in planar graphs. **Technical Report TR1150**, School of Ops. res. & Ind. Eng., Cornell University, US.

Fleischmann, B. (1985) A cutting-plane procedure for the travelling salesman problem on a road network. **Eur. J. Opl Res.**, **21**, 307-17.

Fleischmann, B. (1987) Cutting-planes for the symmetric travelling salesman problem. **Research report**, Hamburg University.

Fleischmann, B. (1988) A new class of cutting-planes for the symmetric travelling salesman problem. **Math. Prog.**, **40**, 225-246.

Garey, M.R. & Johnson, D.S. (1979) **Computers and Intractibility: a guide to the theory of NP-Completeness**. San Fr.; Freeman.

Gélinas, E. (1992) Le problème du postier chinois avec contraintes générales de préséance. **M.Sc.A. dissertation**, École Polytechnique de Montréal.

Gilmore, P.C. & Gomory, R.E. (1961) A linear programming approach to the cutting-stock problem. **Ops Research**, **9**, 849-859.

Goemans, M.X. (1995) Worst-case comparison of valid inequalities for the TSP. **Math. Prog.**, **69**, 335-349.

Golden, B.L. & Assad, A.A. (eds.) (1988) **Vehicle Routing: Methods and Studies**. Amsterdam: North Holland.

Golden, B.L. & Wong, R.T. (1981) Capacitated arc-routing problems. **Networks**, **11**, 305-15.

Gottlieb, E.S. & Rao, M.R. (1990) The generalised assignment problem: valid inequalities and facets. **Math. Prog.**, **46**, 31-52.

Gouveia, L. (1995) A result on projection for the vehicle routing problem. **Eur. J. Opl Res.**, **85**, 610-624.

Grötschel, M., Lovasz, L. & Schrijver J. (1988) **Geometric Algorithms in Combinatorial Optimisation**. Springer.

Grötschel, M. & Padberg, M.W. (1979) On the symmetric travelling salesman problem I: inequalities. **Math. Prog.**, **16**, 265-280.

Grötschel, M. & Pulleyblank, W.R. (1986) Clique tree inequalities and the symmetric travelling salesman problem. **Math. Ops Res.**, **11**, 537-569.

Guan, M. (1962) Graphic programming using odd or even points. **Chinese Math.**, **1**, 273-277.

Hall, L.A. (1993) Integer programming formulations for designing a centralised processing network. Submitted to **Ops Research**.

Harche, F. & Rinaldi, G. (1995) The capacity inequalities for the capacitated vehicle routing problem. In preparation.

Hirabayashi, R., Saruwatari, Y. & Nishida, N. (1992) Tour construction algorithm for the capacitated arc-routing problem. **Asia-Pac. J. of O.R.**, **9**, 155-75.

Howie, N. & Aktin, H. (1993) Precautionary winter gritting in Cumbria. **MSc. Dissertation**, Lancaster University.

Johnson, E.L. & Padberg, M.W. (1981) A note on the knapsack problem with special ordered sets. **Ops Res. Letts.**, 1, 18-22.

Jünger, M., Reinelt, G. & Rinaldi, G. (1995) The travelling salesman problem. In **Ball et al. (1995a)**.

Kohl, N. & Madsen, O. (1995) An optimisation algorithm for the vehicle routing problem with time windows based on lagrangean relaxation.. **Working paper**, Institute of Mathematical Modelling, Technical University of Denmark.

Laporte G., Nobert Y. & Desrochers M. (1985) Optimal routing under capacity and distance restrictions. **Ops Research**, **33**, 1050-1073.

Laporte, G. & Osman, I.H. (1995) Routing problems: a bibliography. **Annals of O.R.**, **61**, 227-262.

Lawler, E.L., Lenstra, J.K., Rinnooy-Kan, A.H.G. & Shmoys, D.B. (Eds.) (1985) **The Travelling Salesman Problem: A guided tour of combinatorial optimisation**. New York: Wiley.

Lenstra, J.K. (1985) Polyhedral combinatorics. In M. O'hEigartaigh, J.K. Lenstra & A.H.G. Rinnooy-Kan (Eds.), **Combinatorial Optimisation: Annotated Bibliographies**. New York: Wiley.

Lenstra, J.K. & Rinnooy Kan, A.H.G. (1976) On general routing problems. **Networks, 6**, 273-80.

Letchford, A.N. (1997) New inequalities for the general routing problem. **Eur. J. Opl Res.**, **96**, 317-322.

Letchford, A.N. (1996a) Further inequalities for the general routing problem. Submitted to the **Eur. J. Opl Res.**

Letchford, A.N. (1996b) On lower bounds for the bin packing problem. **Submitted**.

Letchford, A.N. & Eglese, R.W. (1996a) The rural postman problem with deadline classes. To appear in **Eur. J. Opl Res.**

Letchford, A.N. & Eglese, R.W. (1996b) Valid inequalities for resource-constrained vehicle routing problems. **In preparation**.

Leuker, G.S. (1983) Bin-packing with items uniformly distributed over interval [a, b]. **Proc. 24th Ann. FOCS**, 289-297.

Li, L.Y.O. (1992) Vehicle Routing for Winter Gritting. **PhD Thesis**, Dept. of Man. Science, Lancaster University.

Li, L.Y.O. & Eglese, R.W. (1996) An interactive algorithm for winter gritting. **J. of the Opl Res. Soc.**, **47**, 217-228.

Martello, S. & Toth, P. (1990). **Knapsack Problems: Algorithms and Computer Implementations**. New York: Wiley.

Miller D.L. (1995) A matching based exact algorithm for capacitated vehicle routing problems. **ORSA J. of Comp.**, **7**, 1-9.

Naddef, D. (1990) Handles and teeth in the symmetric travelling salesman polytope. In W. Cook & P.D. Seymour (Eds.), **Polyhedral Combinatorics**. DIMACS Series in Discrete Mathematics and Theoretical Computer Science Vol. 1, Amer. Math. Soc.

Naddef, D. (1992) The binested inequalities for the symmetric travelling salesman polytope. **Math. Ops. Res.**, **17**, 882-900.

Naddef, D. & Clochard, J.M. (1994) Some fast and efficient heuristics for comb separation in the symmetric travelling salesman problem. Research report RR941-M, Institut IMAG, Universite Joseph Fourier.

Nagamochi, H., Ono, T. & Ibaraki, T. (1994) Implementing an efficient minimum capacity cut algorithm. **Math. Prog., 67**, 325-341.

Nemhauser G.L. & Savelsbergh M.W.P. (1994) Functional description of MINTO, a Mixed INTeger Optimizer. **Report COC-91-03A**, Georgia Institute of Technology.

Nemhauser, G.L. & Vance, P.H. (1994) Lifted cover facets of the 0-1 knapsack polytope with GUB constraints. **Ops Res. Letts**, **16**, 253-263.

Nemhauser, G.L. & Wolsey, L.A. (1988) **Integer and Combinatorial Optimisation**. New York: Wiley.

Nobert, Y. & Picard, J.-C. (1996) An optimal algorithm for the mixed chinese postman problem. **Networks**, **27**, 95-108.

Nygard, K.E., Greenberg, P., Bolkan, W.E. & Swenson, J.E. (1988) Generalised assignment methods for the deadline vehicle routing problem. In **Golden & Assad (Eds.)**, 1988.

Orloff, C.S. (1974) A fundamental problem in vehicle routing. **Networks**, **4**, 35-64.

Padberg, M. W. & Rao, M. R. (1982) Odd minimum cut-sets and b-matchings. **Math. Ops Res.**, **7**, 67-80.

Padberg, M. W. & Rinaldi, G. (1987) Optimisation of a 532-city symmetric travelling salesman problem by branch-and-cut. **Ops Res. Letts.**, **6**, 1-7.

Padberg, M. W. & Rinaldi, G. (1990) Facet identification for the symmetric travelling salesman polytope. **Math. Prog.**, **47**, 219-257.

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

Stephenson M. (1996) Devising skeletal routes for a parcel delivery company. **Draft PhD Thesis**, Dept. of Management Science, Lancaster University.

Thangiah, S.R., Osman, I.H., Vinayagamoorthy, R. & Sun, T. (1994) Algorithms for the vehicle routing problem with time deadlines. **Amer. J. of Math. and Man. Sci.**, **13**, 323-384.

Welz, S.A. (1994) Optimal solutions for the capacitated arc-routing problem using integer programming. **PhD thesis**, University of Cincinnati.

Wolsey, L.A. (1997) Private communication.

Zemel, E. (1989) Easily computable facets of the knapsack polytope. **Math. of Ops Res.**, **14**, 760-765.

Appendix 1: RPP and RPPDC Instances.

The eleven RPP instances solved in Chapter 3 were previously described and solved in Christofides et al. (1981) and Corberán & Sanchis (1994). The data, reproduced below, was obtained from Angel Corberán. However, the ordering of the edges has been altered somewhat, for reasons outlined at the end of this appendix.

For each problem, the first line gives the instance number and the number of vertices, required and non-required edges. The following lines list the required and non-required edges in ascending order of label. For each edge, the two end-vertices are given in ascending order, followed by the cost c_e.

In five of the problem listings, an asterisk appears in brackets. This is of no relevance to the RPP instances, but is of use when converting them to RPPDC instances as explained at the end of the appendix.

```
RPP I2: V = 14 R = 12 NR = 21

R:    2  3  9     3  4 11     5  6  8     5  7 10     9 10  8
      9 12 13     1  3  5     6  7  2     8  9  4    10 11  1
     11 12  7    13 14  2

NR:   1  4  9     1  5 17     1  7 19     1  8 27     1 10 22
      1 13 13     2  5 14     2  7 21     2  8 29     2 10 24
      3  5 18     3  7 23     3  8 31     3 10 26     5 10 20
      5 13 26     7  8 10     7 10 13    10 12  3    10 13 10
     10 14 10
```

```
RPP I4: V = 17 R = 22 NR = 13

R:    1  2  3     1  4  3     7 12  3    13 14  3    13 17  3
     14 15  3    14 16  4    14 17  3    15 17  4    15 16  3
     16 17  3 (*) 2  3  1     2  4  2     3  4  1     4  5  2
      6  7  2     6  8  2     7  8  2     7 10  2     9 10  2
      9 12  2    11 12  2

NR:   1 16 12     3  6  6     4 17  9     5  6  4     5  8  3
      5 13 11     5 17  9     6 13  8     7  9  1     8 10  1
      9 11  1    10 13  8    12 13  7
```

RPP I5: V = 20 R = 16 NR = 19

```
R:    1  2  5     1  6  7     4  5  3     7  8  3     8  9  7
     12 13  6    14 15  3    15 16  9    17 18  5    18 19  3
     19 20 10     2  3  2     3  4  1     5  6  1     8 10  2
     10 11  2

NR:   2  5  2     4  7  7     4 12 19     4 14 17     5 16  8
      5 20 10     7 11  5     7 14 17     9 10  4     9 12  9
      9 14 15     9 16 24    12 14 10    12 16 19    13 16 14
     13 17 10    16 17 12    16 20  8    18 20  3
```

RPP I14: V = 28 R = 31 NR = 48

```
R:    1  4  4     2  3  4     2  4  5     6  7  7     6  8  6
      9 14  8    10 11  5    11 12  4    11 14 11    12 13  9
     13 14  6    15 16  4    18 19  7    18 20  6    20 21  4
     22 23  4    22 24  8    23 24  5    24 26  6    25 28  4
 (*)  1  2  2     3  4  3     5  6  3     9 10  3    12 14  3
     16 17  3    19 20  3    24 25  2    25 26  3    26 27  2
     27 28  1

NR:   1 28  3     2 28  3     3  5  7     3  8  5     3 22 11
      3 25  9     4 22 11     4 25  9     5  8  3     5 22 12
      5 25 10     6  9  9     6 10  8     6 13 19     6 19 19
      6 21 16     7  8  2     7  9  8     7 10  7     7 13 14
      7 14 13     7 19 14     7 21 11     7 22 11     8 22 10
      8 25  8     9 19 16     9 21 13     9 22 13    13 15  1
     13 19 12    13 21 17    13 22 17    14 19 11    14 21 16
     14 22 16    15 17  5    15 19 12    16 19 10    17 18  8
     17 19 11    17 20  9    17 26 20    18 26 17    20 23  8
     20 26 13    21 23  5    22 25  8
```

RPP I16: V = 31 R = 34 NR = 60

```
R:    1  3  4     2  3  5     6  7  6     7  8  9    10 12  4
     12 14  5    16 17  7    16 20  4    17 18  6    18 19  4
     19 20  6    22 23  7    24 25  5    25 26  8    27 28  6
     28 29  5    28 30  5    29 30  8    30 31  4 (*) 1  2  1
      3  4  3     4  5  2     8  9  3    10 11  2    11 12  1
     12 13  3    13 14  2    15 16  2    15 17  3    17 19  1
     17 20  1    19 21  3    20 21  2    28 31  2

NR:   1  6 12     1  7 10     1 23 12     2  5  8     2 30  9
      2 31  7     3 30 11     3 31  9     4  7  9     5  6  7
      5 23  7     5 30  9     5 31  7     6  8  1     6 23  5
      6 29 18     6 30 11     7  9  2     8 11  8     8 14  9
      8 18 14     8 21 15     8 22  7     9 10  6     9 14  9
      9 18 14     9 21 15     9 22  7    10 13  3    11 14  3
     11 18  8    11 22  7    13 15  3    14 18  7    14 22  8
     18 22  9    18 24 12    18 26 12    18 29 14    18 30 19
     21 22 10    21 24 13    21 26 13    21 29 15    21 30 20
     22 24 11    22 26  8    22 29 10    22 30 15    23 24 14
     23 26 11    23 29 13    23 30 11    24 26  9    24 29 11
     24 30 16    26 27  8    26 29  7    26 30 13    27 29  3
```

RPP I17 V = 19 R = 17 NR = 27

```
R:    1  2  3     2  4  5     3  4  4     5  8  4     6  7  6
      7  8  3     8  9  5    10 12  3    13 14  6    14 15  4
     14 16  3    17 18  9     2  3  1     5  6  2     7  9  1
     10 11  2    18 19  2

NR:   1 19  7     3  5  7     3  6  8     3 14  5     3 17  5
      4 14  6     4 17  6     4 19 13     5 13  9     5 14  7
      6 14  8     8 13  6     9 10  4     9 12  5     9 15 10
      9 16 10     9 17 16    11 12  4    11 16 14    11 17 20
     12 16 13    12 17 19    13 15  4    14 17  6    14 19 13
     15 16  5    17 19 10
```

RPP I20: V = 50 R = 63 NR = 35

```
R:    1  2  4     1  7  5     5  7  4     8 15  5     9 14  6
     10 11  8    11 12  6    12 13  7    14 15  4    15 16  4
     17 18  9    17 20  5    18 19 10    19 21  6    20 21  4
     21 22  4    22 23  8    25 26  4    26 27  5    26 31  6
     26 32  6    27 28  4    28 31  9    29 30  7    31 32  5
     33 35  5    35 38  6    36 37  7    37 38  6    39 42  4
     41 42  5    41 43  8    41 46  6    43 46  4    44 45  5
     45 46  8    47 48  4    47 49  7    48 50  4    49 50  8
(*)   1  4  2     2  3  3     3  4  1     4  5  3     5  6  1
      6  7  2     8  9  3     9 10  2    10 13  1    13 14  3
     14 16  2    20 23  3    20 24  2    23 24  1    28 29  3
     30 31  2    34 35  3    35 36  2    36 38  1    39 40  2
     40 41  3    43 44  2    44 46  3

NR:   4 15  7     4 39 21     4 46 18     5 15  5     5 39 19
      5 45 17     5 46 16     5 48  9    12 17  9    12 24 10
     12 33 24    12 36 18    12 39 27    13 33 24    13 36 18
     15 39 15    15 45 19    15 46 12    16 17 17    16 24 18
     16 25 30    16 33 22    16 36 16    17 36 23    17 39 32
     23 27  3    24 36 21    24 39 33    26 33  6    33 34  4
     36 39 24    36 40 25    39 48 23    45 48 14    46 48 19
```

RPP I21: V = 49 R = 67 NR = 43

```
R:    1  6  4     2  3  6     2  6  5     4  6  9     5  8  5
      6  7  7     7  8  4     9 10  8     9 13  6    10 13  5
     11 12  7    12 13  9    12 16  6    13 14  4    17 18  6
     18 22  4    20 21  5    23 24  6    23 25  9    24 26  5
     25 26  4    27 28  8    27 29  6    28 29  7    28 31  4
     29 32  4    30 32  7    31 32  6    32 34  5    33 34  4
     39 40  7    40 41  4    43 49  9    44 49  5    45 46  4
     46 48  5    47 48  5    48 49  7 (*) 1  2  3     3  4  3
      4  5  2     6  8  1    10 11  3    10 12  3    11 16  2
     14 15  3    15 16  2    17 20  2    19 21  1    20 23  2
     21 22  3    28 30  2    29 30  3    30 31  3    31 33  2
     32 33  1    35 37  2    36 38  1    37 38  3    38 39  3
     41 42  1    43 44  2    43 47  2    44 45  3    45 48  3
     46 47  3    47 49  3

NR:   1  7  8     1 17 17     1 18 15     7 17 18     7 18 13
      8  9  8     8 18 12     8 22 15     8 26 11     8 27 10
      8 29 10     8 34 18     8 45 17     9 18 17     9 22 20
      9 26 16     9 27  8     9 35 14     9 45 22    15 35  7
```

```
      18 26  9     18 29  8     18 34 16     18 45 15     21 24  2
      22 24  3     22 29 11     22 34 15     22 45 14     26 29  7
      26 34 11     26 43  6     26 45 10     28 35  4     29 45 11
      31 35  2     31 39  3     34 41 11     34 45  3     35 36  6
      38 40  8     39 41  7     41 46 17
```

RPP I22: V = 50 R = 74 NR = 110

```
R:     2 10  8      3 10  9      4  7  7      6  7 10      7  8  8
      11 14 11     13 15  7     13 16  9     15 16  8     17 19  7
      20 21  8     20 26  9     21 25 11     22 25  9     23 24  7
      23 25  8     25 26  7     28 31 10     28 32  8     31 32  7
      31 33 11     32 34 10     33 34  8     34 35  7     36 37  7
      36 39 15     37 39 11     38 39  9     39 41  9     40 41  8
      42 43  7     43 44  8     43 46  9     45 48  7     45 50  9
      46 48  8     48 49  7     48 50 12      1  2  6      1 10  3
       2  3  5      3  4  5      4  5  3      4  9  6      5  6  4
       8  9  6      9 10  4     11 12  5     11 15  6     12 13  4
      14 15  2     14 17  6     16 17  5     17 18  5     20 27  4
      21 22  5     21 26  6     22 23  2     24 25  4     26 27  6
      28 29  5     28 35  6     29 30  6     30 31  3     32 33  5
      32 35  4     36 40  3     37 38  6     42 45  6     43 45  4
      45 46  5     45 47  5     47 50  3     49 50  6

NR:    7 11 16      7 14 17      7 18 19      7 20 29      7 31 19
       7 33 27      7 39 32      7 42 38      7 43 39      7 45 40
       7 47 39      8 11 15      8 14 15      8 18 15      8 20 24
       8 30 15      8 31 14      8 33 22      8 39 27      8 42 33
       8 43 34      8 44 38      8 45 35      8 47 34      9 14 17
       9 18 17      9 20 23      9 29  6      9 31 13      9 39 26
       9 42 32      9 43 33      9 44 37      9 45 34      9 47 33
      10 28  3     11 29 26     11 31 19     11 33 27     11 34 34
      11 39 32     14 18  6     14 31 12     14 33 20     18 19  8
      18 21  6     18 31 12     18 33 18     18 34 21     18 39 19
      19 21 10     19 23 16     20 31 18     20 33 12     20 34 15
      20 39 13     25 33 26     25 34 25     25 39 23     25 42 15
      25 43 14     25 44 11     25 45 17     25 47 16     26 33 20
      26 34 19     26 39 17     26 42  9     26 43  8     26 44 12
      26 45 11     26 47 10     27 34 16     27 39 14     27 42 10
      27 43 11     27 44 15     27 45 12     27 47 11     28 37 12
      31 39 21     31 42 27     31 43 28     31 44 32     31 45 29
      31 47 28     33 39 14     33 42 21     33 43 22     33 44 26
      33 45 23     33 47 22     34 38 11     34 39  7     34 42 19
      34 43 21     34 44 25     34 45 21     34 47 20     35 37  8
      39 42 16     39 43 19     39 44 23     39 45 18     39 47 17
      41 47 17     41 50 16     42 47  5     44 46 13     44 47 16
```

RPP I23: V = 50 R = 78 NR = 80

```
R:     1  2  5      1  8  9      2  3  4      2  5  4      3  4  6
       3  6  7      4  5  9      5  6  5      5  7  7      6  8  6
       7  8  8     10 11  5     10 12  4     10 13  7     11 13  9
      13 14  6     13 16 10     14 16  4     15 17  8     16 17  6
      16 18  4     19 20  6     19 24  6     20 21  9     21 22  4
      22 23  5     23 24  6     23 26  5     24 25  7     25 26  4
      27 28  9     27 31 11     28 29  8     28 30  9     30 31  4
      30 33  6     34 35  6     35 36  9     35 37  5     36 38  4
      37 40  7     39 40  6     40 41  4     42 43  9     42 45  5
```

```
      44 49  4     45 46  9     45 49  6     47 48  4     47 50  6
      48 50  4      1  6  3      2  6  1      3  5  3      7  9  2
       8  9  1     12 13  3     13 15  1     15 16  3     17 18  3
      19 25  3     20 25  2     20 26  1     21 26  2     29 30  3
      31 32  2     32 33  1     34 37  2     35 38  3     37 39  3
      39 41  2     43 44  2     43 49  3     44 50  2     46 47  3
      46 49  1     47 49  2     49 50  3

NR:    1 19 22      1 23 25      1 34 21      1 36 28      1 43 34
       4  7  9      4 12  5      4 19  8      4 21 15      4 23 13
       4 27 22      4 32 23      4 34  9      4 36 16      4 43 22
       4 44 22      4 48 27      5  8  3      7 19 11      7 23 16
       7 34 12      7 36 19      7 43 25     12 21 16     14 18  6
      18 21 11     18 22 11     18 27 10     18 32 11     18 36 15
      18 38 14     18 40 15     18 42 17     18 43 10     18 44 10
      18 48 15     19 34  7     19 36 14     19 43 20     21 27 13
      21 32 14     21 43 13     21 44 13     21 48 18     22 27 11
      22 32 12     22 36 13     22 38 12     22 40 13     22 42 15
      22 43 10     22 44 11     22 48 16     23 25  6     23 34  8
      23 36 15     27 30  5     27 32  3     27 43  2     27 44  2
      27 48  7     29 33  3     30 32  4     32 43  3     32 44  3
      32 48  8     33 48  8     34 36 11     36 43  7     38 40  3
      38 42  7     38 43  6     38 46  9     40 42  8     40 43  7
      40 46 10     41 42  8     41 46 10     42 49  7     43 48  7
```

RPP I24: V = 41 R = 55 NR = 70

```
R:     1  3  7      2  3  6      2  4  7      6  7  5      6  9  8
       7  8  6     12 13  6     12 14  5     14 16  9     15 16  6
      16 17  7     19 21  6     19 22  8     20 22  5     24 25 13
      24 26  9     25 27  7     25 28  8     26 27 11     26 29  6
      27 28  5     28 30  6     31 32  5     32 35  6     33 34  5
      33 35  6     34 35  9     36 39  5     37 39  8     39 40  7
      40 41  8      1  2  3      3  5  4      8  9  4      8 10  2
       9 10  3     11 12  4     13 14  3     14 15  4     18 19  4
      18 21  3     19 20  3     21 22  1     21 23  3     22 23  4
      27 29  4     29 30  3     31 33  3     32 33  2     32 34  4
      36 37  4     37 38  3     37 40  3     38 40  4     38 41  2

NR:    3 18  9      4  5  9      4  8 19      5 20  5      7  9  4
       7 11 10      7 13 14      7 16 14      7 17 14      7 25 12
       7 30 23      7 31 15      7 38 27      8 20  9      8 24 14
       8 29 17      8 36 19     10 11 11     10 16  9     10 17  9
      10 25  7     10 30 18     10 31 10     10 38 22     11 13  8
      11 16 14     11 17 14     11 20 18     11 24 23     11 25 12
      11 30 23     11 31 15     11 36 28     11 38 27     13 15  3
      15 17  5     16 20 16     16 24 21     16 25 10     16 30 21
      16 31 13     16 36 26     16 38 25     17 20 16     17 24 21
      17 25 10     17 29 16     17 30 15     17 31  7     17 36 21
      17 38 19     20 24 13     20 25 14     20 29  9     20 31 17
      20 36 11     20 38 16     23 29 13     23 31 23     23 36 15
      23 38 20     24 31 22     24 36 23     25 31 11     29 31 12
      29 36 11     29 38  9     30 31 11     31 36 17     33 38 10
```

The RPPDC test problems were based on five of these eleven RPP instances: I4, I14, I16, I20 and I21. As explained in Section 4.6, each of these five instances yielded two RPPDC instances; one with L = 1 and one with L = 2.

The L = 1 versions are formed by letting $t_e = c_e$ ($\forall e \in E$) and $s_e = \lfloor 3\ c_e / 2 \rfloor$ ($\forall e \in R$), and setting the deadlines to 105, 260, 263, 522 and 490, respectively, for the five problems.

The L = 2 versions have the same costs and times as the L = 1 versions. The first deadline, T^1, was 92, 236, 229, 498 and 468, respectively. The second deadline, T^2, was 110, 321, 268, 624 and 500. In the above listings for RPP instances I4, I14, I16, I20 and I21, the required edges have been ordered so that those edges which become members of R^1 in the L = 2 version precede those which become members of R^2. The small asterisk indicates the division between the two classes of edges.

Appendix 2: CARP+1D Instances.

The CARP+1D instances examined in Chapter 7 were previously described by Li (1992); see also Eglese & Li (1996). The data was obtained from Richard Eglese. The four problem instances are called EASTA, EASTB, SOUTHA and SOUTHB. The networks on which these problems are defined are given below (further information is given at the end of the appendix). For each problem, the first line gives the instance name, the number of vertices, required and non-required edges, and the sum of the costs (lengths) of the required edges in metres. The following lines list the edges in ascending order of label. For each edge, the two end-vertices are given in ascending order, followed by the cost (length) c_e in metres and an indication of whether the edge is required or not. Note that SOUTHA and SOUTHB are defined on the same network; they only differ in the location of the depot.

EASTA: V = 77, R = 84, NR = 27, servicing distance = 231429 m.

```
 1  2 3166 R     2  1 3166 R     2  3 1389 R     3  2 1389 R
 2  4 1741 R     4  2 1741 R     4  5 5565 R     5  6  759 N
 5  7  636 R     7  8 1798 R     8  9 2613 R     9 10 2008 R
10 11 1157 N    11 12 3212 R    12 11 3212 R    12 16 2913 R
16 12 2913 R    16 13 1343 R    13 14  675 R    14 15  745 N
13 77 1201 N    15 17 2585 R    15 18 3817 R    18 19 4130 R
19 20 3247 R    20 18 2092 N    21 19 3819 R    21 22 1743 R
22 75 2396 R    22 24  446 R    24 25 2705 R    75 25 1628 R
25 26 1720 R    26 23 1241 R    26 27 1540 R    26 28 1754 R
28 29 1600 N    29 25 2690 R    30 31 1406 N    23 31 3989 R
31 32 5835 R    32 33 3044 R    32 34 4329 R    32 35 8581 R
33 36 2726 R    33 37 7500 N    37 38  759 N    37 39 2101 N
39 40  457 N    39 35  743 N    35 41 1456 R    40 41  904 N
41 42 1769 N    42 43 1099 N    43 44 7777 R    44 45 1186 R
45 46 1191 N    44 46 1257 R    46 47 3269 R    47 48 1020 R
48 11 7886 N    47 49  938 R    49 50  828 R    49 51 1012 R
50 52  241 R    51 53  405 R    53 52  433 R    52 54  352 R
53 24 2416 R    19 50 2986 R    56 55  648 R    56 42 2586 N
42 57 1420 R    57 42 1420 R    57 58 7782 R    58 57 7777 R
58 59 2476 R    59 58 2476 R    59 11 7783 R    11 59 7783 R
60 58 3349 R    60 61 3904 R    60 62 2292 R    62 63 3012 R
63 64  924 R    63 65 2077 R    62 66 1603 R    66 68 4512 R
58 69 3231 R    59 69 1095 R    69 59 1095 R    69  4 7459 R
 4 69 7459 R    70 71 5589 N    71 72  588 N    73 71  493 R
73 72  500 R    73 74 2536 R    72 18 9150 R    56 67 3573 N
75 23 1668 R    12 76  742 N    76 20  752 R    15 77 1341 N
77 76 3543 N    21 51  202 R    44 59 2765 R    44 59 2765 R
 7  8 1984 N    60 67 2853 N    62 67 3120 N
```

EASTB: V = 77, R = 84, NR = 27, servicing distance = 238374 m.

```
 1  2 3261 R     2  1 3261 R     2  3 1431 R     3  2 1431 R
 2  4 1793 R     4  2 1793 R     4  5 5732 R     5  6  782 N
 5  7  655 R     7  8 1852 R     8  9 2691 R     9 10 2068 R
10 11 1192 N    11 12 3309 R    12 11 3309 R    12 16 3000 R
16 12 3000 R    16 13 1383 R    13 14  696 R    14 15  768 N
13 77 1237 N    15 17 2662 R    15 18 3932 R    18 19 4254 R
19 20 3345 R    20 18 2155 N    21 19 3933 R    21 22 1795 R
22 75 2468 R    22 24  459 R    24 25 2786 R    75 25 1677 R
25 26 1771 R    26 23 1278 R    26 27 1586 R    26 28 1807 R
28 29 1648 N    29 25 2771 R    30 31 1448 N    23 31 4109 R
31 32 6010 R    32 33 3135 R    32 34 4459 R    32 35 8838 R
33 36 2808 R    33 37 7725 N    37 38  782 N    37 39 2164 N
39 40  471 N    39 35  766 N    35 41 1500 R    40 41  931 N
41 42 1822 N    42 43 1132 N    43 44 8011 R    44 45 1222 R
45 46 1226 N    44 46 1295 R    46 47 3367 R    47 48 1051 R
48 11 8123 N    47 49  966 R    49 50  852 R    49 51 1042 R
50 52  248 R    51 53  417 R    53 52  446 R    52 54  363 R
53 24 2489 R    19 50 3076 R    56 55  667 R    56 42 2664 N
42 57 1463 R    57 42 1463 R    57 58 8016 R    58 57 8010 R
58 59 2551 R    59 58 2551 R    59 11 8016 R    11 59 8016 R
60 58 3449 R    60 61 4022 R    60 62 2361 R    62 63 3102 R
63 64  951 R    63 65 2139 R    62 66 1651 R    66 68 4647 R
58 69 3328 R    59 69 1127 R    69 59 1127 R    69  4 7683 R
 4 69 7683 R    70 71 5756 N    71 72  606 N    73 71  507 R
73 72  515 R    73 74 2613 R    72 18 9425 R    56 67 3680 N
75 23 1718 R    12 76  765 N    76 20  775 R    15 77 1381 N
77 76 3649 N    21 51  209 R    44 59 2848 R    44 59 2848 R
 7  8 2044 N    60 67 2939 N    62 67 3214 N
```

SOUTHA / SOUTHB: V = 140, R = 160, NR = 43, servicing distance = 347351 m.

```
 5   6 2728 R     6   5 2728 R     6  7 2135 N     6  8  1032 R
 8   9 3797 R     9  10 1188 N     8 11 4910 R    11 12  2559 R
12  13 1236 R    13  12 1236 R    13 14 1856 R    14 13  1856 R
10  15 1354 N    15  16  866 N    15 17 1375 N    17 13   632 N
16  13 1282 R    17  18  850 N    18 19  567 N    19 10  2923 R
12  20 3529 R    20  21 5933 R    20 22 1002 R    21 23  5714 N
20  24 3389 R    23  24 5802 R    24 25 1046 R    25 26  4966 R
23  26 2092 N    25  27 1987 R    27 28  620 R    28 29   495 R
28  30  540 R    30  31 4212 R    26 31 1247 N    30 32  2912 R
29  33 5340 R    11  33 2484 R   139 34 5002 R    34 35  6111 R
35  36  414 R    35  37 1719 R    36 37 2051 R    37 36  2051 R
36  38 2562 R    38  36 2562 R    38 39  635 R    39 38   635 R
39  40 1858 R    40  41 1231 R    41 38 1006 R    41 47 10147 R
42  37 1797 R    42  43  522 R    43 37 1649 R    43 44  3188 R
44  45 1411 R    45  34  429 R    47 42  396 R    47 46   700 R
43  46  513 R    33 139 1301 R   139 48 5086 N    48 49  1049 R
48  50  811 R    50  51 3109 R    51 52 1255 N    52 53  1215 R
53 140 2489 R    49 140 2493 R    54 55 1015 R    55 56   404 R
56  57 5023 R    57  58 2027 R    58 59 4167 R    59 60  1848 R
60  45 1496 R    60  61 3620 R    59 61 3194 R    61 49  1375 R
57  62 1461 R    62  63  678 R    63 64  880 R    64 65  1527 R
```

```
 56  65  3082 N    62  66 3768 R     66  67 3423 R     67 68   974 R
 67  69   425 R    69  70  826 R     70  71  919 R     69 71   637 R
 71  72   918 R    72  73 1861 R     72  74 1524 R     73 74   525 R
 73  44  3690 R    71  75 3663 N     75  76 4855 R     74 76  2802 R
 76  77   983 R    46  77 1642 R     77  78 1558 R     78 79  1231 R
 78  79  1343 R    79  80 1190 R     80  79 1190 R     79  81  952 R
 80  81  1065 R    80  82 1666 R     82  80 1666 R     82  83 1064 R
 81  83  1794 R    82  84 1295 R     84  82 1295 R     83  84 1846 R
 84  85  1034 R    85  84 1034 R     85  86 2011 R     86  87  449 R
 86  88  3355 N    88  89 1391 N     88  90 1209 N     90  91  736 N
 90  68  6941 N    91  66 7096 N     89  92 1010 R     92  82 2722 R
 92  84  3028 N    64  93 2870 R     63  93 2562 R     93  94 1720 N
 94  95  4995 N    95  96  818 R     96  97 1702 R     97  98  320 R
 96  99  1052 R    97  99 3770 R     99 100 3365 R    100  94 1798 N
 94 101   460 N    66 101 2844 N    101 102 1756 N    102 103 4332 N
103  91  2784 N   102 104 2142 R    104 105  583 R    104 105  797 R
105 106  1556 R   106 107 2886 R    107 108 2590 R    106 108 3899 R
108 109 10576 R   107 110  848 R    110 111  752 R    110 112 1375 R
107 112  1296 R   112 113  453 R    113 103 4732 R    113 114 1259 R
114 115  3734 N   115 116 1013 N    116   1 1834 R    116 117 1956 R
115 117  2038 N   117   2 1661 R    117 118 5019 R    117 119 2445 R
119 117  2501 R   119 120 1417 N    114 118 5972 R     91 118 3593 N
118 121  1763 N   120 121 1670 N    121 122  802 N    122  87 1554 N
122 123  1558 N   123 124 3026 N    123 120 3288 N    123 125 3289 N
124   4  6924 R   124 126 1166 R    126 127 1591 R    127 128 2223 R
128   3  4271 R   128 129 2339 R    129 127 1156 R    126 130 3571 R
126 131  2649 R   131 132  466 R    132 133  754 R    133 134 1029 R
133 135  1912 R   135 134 1394 R    134 136 3189 R    136 137 2972 R
136 138   530 R   131 138 4829 R    138 125 2021 R     55 140 1422 R
140  54  1805 R    11  27 5108 R     58  70 1081 R
```

For all of the four problems, the time deadline T is set at 115 minutes. The vehicle is assumed to travel at a speed of 40 km/hr when not servicing an edge, but only 25 km/hr when servicing. The demand of a required edge is assumed to be equal to its cost. That is: $q_e = c_e$ (if $e \in R$)

$= 0$ (if $e \in E \setminus R$).

Because of this, the vehicle capacity can be expressed in terms of distance. It is set at 40km.

EASTB is identical to EASTA except that the edge costs, times and demands are somewhat larger. The only difference between SOUTHA and SOUTHB is that the depot vertex is 91 for SOUTHA, but 73 for SOUTHB. Vertex 91 is not near any required edges, whereas vertex 73 is incident on several. This means that SOUTHA is likely to require more vehicles than SOUTHB and have a higher optimal solution cost.