An Investigation of Multilevel Refinement in Routing and Location Problems

Ph.D.

Demane O'Neil Rodney

School of Computing and Mathematical Sciences
the University of Greenwich
London, England.

A thesis submitted in partial fulfilment of the requirements of the University of Greenwich for the degree of Doctor of Philosophy.

Monday, 16 June 2008

ACKNOWLEDGEMENTS

I must first thank my supervisors Dr. Chris Walshaw and Dr. Alan Soper for the priceless guidance, painfully searching questions, generously apportioned encouragements, all of which were vital to the production of this thesis.

This research has been fully funded by the University of Greenwich and so I must again thank my supervisors along with Dr. Miltos Petridis, Ing. Peter Smith, Ing. Alex Fedorec and Professor Keith Reynolds for the support a research student needs to successfully survive in an academic institution. I must also thank all the lecturers, fellow tutors, and researchers who I have worked with during these last four years; your support has been invaluable.

Finally let me thank all my family and friends for their support in particular; my parents, my brother Andre, sisters Karen and Karlene; Aunts Dawn, Ann, Jean and Yvonne; my girlfriend, Tamar. Thank you!

For:

Mom and Dad

ABSTRACT

Multilevel refinement is a collaborative hierarchical solution technique. The multilevel technique aims to enhance the solution process of optimisation problems by improving the asymptotic convergence in the quality of solutions produced by its underlying local search heuristics and/or improving the convergence rate of these heuristics. To these aims, the central methodologies of the multilevel technique are filtering solutions from the search space (via coarsening), reducing the amount of problem detail considered at each level of the solution process and providing a mechanism to the underlying local search heuristics for efficiently making large moves around the search space. The neighbourhoods accessible by these moves are typically inaccessible if the local search heuristics are applied to the un-coarsened problems. The methodologies combine to meet the multilevel technique's aims, because, as the multilevel technique iteratively coarsens, extends and refines a given problem, it reduces the possibility of the local search heuristic becoming trapped in local optima of poor quality.

The research presented in this thesis investigates the application of multilevel refinement to classes of location and routing problems and develops numerous multilevel algorithms. Some of these algorithms are collaborative techniques for metaheuristics and others are collaborative techniques for local search heuristics. Additionally, new methods of coarsening for location and routing problems and enhancements for the multilevel technique are developed. It is demonstrated that the multilevel technique is suited to a wide array of problems. By extending the investigations of the multilevel technique across routing and location problems, the research was able to present generalisations regarding the multilevel technique's suitability, for these and similar types of problems.

Finally, results on a number of well known benchmarking suites for location and routing problem are presented, comparing equivalent single-level and multilevel algorithms. These results demonstrate that the multilevel technique provides significant gains over its single-level counterparts. In all cases, the multilevel algorithm was able to improve the asymptotic convergence in the quality of solutions produced by the standard (single-level) local search heuristics or metaheuristics. The multilevel technique did not improve the convergence rate of the single-level's local search heuristics in all cases. However, for large-scale problems the multilevel variants scaled in a manner superior to the single-level techniques. The research also demonstrated that for sufficiently large problems, the multilevel technique was able to improve the asymptotic convergence in the quality of solutions at a sufficiently fast rate, such that the multilevel algorithms were able to produce superior results compared to the single-level versions, without refining the solution down to the most detailed level.

CONTENTS

FIGURES

TABLES

Chapter I

1 Multilevel Refinement for Routing and Location Problems

The importance of '*transportation and communication systems*' to the economic success of individuals through to civilisations is a well-documented fact [205], [238]. The areas that can be modelled by the concepts of transportation and communication are vast. From a logistics management point of view, two areas of particular interest are the areas of routing and location. Within the areas of routing and location, the ongoing stories of economic success are a composite of benefits, derived from improvements made in these areas and negative consequences, resulting from inefficiencies within these areas.

The US environmental protection agency, reports that the transportation sector is the second largest source of CO_2 emissions in the US [295] and emission due to transportation is growing faster than other sectors [289]. A recent policy paper highlighted the fact that inefficiency in the transportation systems in Europe costs the EU approximately 1% of GDP each year [14]. These two reports pinpoint two of the main areas of negative consequences experienced from inefficiencies in the transportation sectors: that of financial cost and adverse environmental impact. On the positive side, improvements in the transportation systems over the last couple of decades have saved the US economy 4% of GDP[1] [2] . This has primarily resulted from reduction in inventory cost as the increasing reliability of the transportation systems has facilitated greater reliance on just-in-time manufacturing.

[1] Taken against the backdrop that the US economy is approximately 25% of the world's economy; these are not insignificant figures.

The efficient location of resources is one that is of interest to a number of areas of economic activity. As Current et al. [66] reported, not only do location decisions involve large sums of capital expenditures, they have major impact on the ability of companies to compete in the market place and they play a significant role in stimulating economic activity.

Location problems seek to locate '*resources*' such that the '*cost*' of using the resources is at its most beneficial to both the users and the suppliers of the resources. Routing problems seek to allow suppliers to allocate '*resources*' to sets of users at minimum '*cost*'. The terms cost and resources in these descriptions are euphuisms for a host of measures. Cost describes diverse considerations encompassing environmental impact, time, financial cost, bandwidth requirement, safety, customer satisfaction, profitability, competitiveness etc. Similarly, the considerations referred to as resources run a wide gamut including, but not limited to, warehouses, shipping ports, power plants, hubs, consumable products and services etc. This reflects the fact that routing and location problems model concerns that have direct and indirect commercial applications.

Direct applications of the fields of routing and location are found in the transportation and logistics sector. According to the US department of commerce in 2007, approximately 10 % of U.S. GDP was related to transportation activity [146]. Globally, various companies can be found whose main service is devoted to providing solution techniques for routing problems [246]. Similarly, for location problems, geographical information systems or supply chain management and other technologies, which make use of location analysis commercially, are important parts of the economy. Indirect application of location analysis can be found in the areas of data mining, pattern recognition, bio informatics, cluster analysis [276], portfolio management [59], computer network design [105] etc.

Due to the importance of routing and location problems, both in academia and in various industries, work into finding solution algorithms for such problems continues to be an active area of research. Typically with these types of problems, the goal is to find an optimal (or high quality) solution from a finite or numerable infinite set of possible solutions [28]. The production of routing and location models of increasingly high quality will lead to improved efficiency in many sectors of the economy. However, location and routing models are exceedingly difficulty to solve, as they are intractable for large problems. Hence, heuristic approaches are the predominant solution techniques employed for these problems.

This research centres on a collaborative technique, multilevel refinement, which is capable of aiding the performance of heuristic approaches used in solving routing and location problems.

1.1 Multilevel Refinement

Multilevel refinement is a collaborative hierarchical solution technique that allows a practitioner to vary the level of detail considered when solving a given problem. To achieve this, the technique creates a hierarchy of approximations to the given problem through a recursive coarsening phase. Each of the approximations varies the level of detail at which the problem is viewed. However, solving any one of the approximations results in a feasible solution to the given problem and the solution of an approximation can be projected onto another approximation of greater detail.

Typically, the final approximation created at the end of the coarsening phase is equivalent to an initial solution to the given problem. Starting with this initial solution, the technique applies a refinement phase that attempts to improve the quality of the solution with regard to the objective function. The technique then applies extension operators to transform the solution, to a level of greater detail.

The multilevel technique [25], [285] has been used for a number of years with proven effectiveness across varying problem areas. These include clustering [153], grid computing [187], graph partitioning, graph colouring and the travelling salesman problem [304], [302].

A recent survey shows increased use of the technique, and metaheuristics from simulated annealing through genetic algorithms to tabu search have been incorporated into effective multilevel implementations [305].

1.2 The Problems Studied and the Motivations

The research applies multilevel refinement to classes of routing and location problems[1].

Multilevel refinement greatly aided the solution process when applied to the traveling salesman problem (TSP) [302] and the similarities between the TSP and the vehicle routing problem (VRP) made it a logical decision to question, how a multilevel algorithm for the VRP would perform? This provided the initial motivation for the current research.

However, as research has often gone; this was expanded to include a study of the application of the multilevel technique to facility location problems. The reasons for this were manifold. Firstly, further generalisations were required to validate some of the concepts surrounding the technique. Secondly, facility location problems were chosen

[1] Chapter II presents formal descriptions of the problems to which the technique has been applied.

because the clustering aspect of these problems made them prime candidates for the technique and the researcher wanted the challenge of working with clustering problems. Thirdly, routing and location problems together address numerous concerns faced in the area of logistics management. By demonstrating that the multilevel technique can aid the leading heuristics approaches employed in this area, the research identified an important area of commercial activity in which the technique can potentially make a valuable contribution.

1.3 Research Objectives

The multilevel technique (framework) does not seek to position itself as the answer to the question "*which is likely to be the best algorithm for solving problem y?*" [305]. Consequently, questions relating to how does the multilevel technique compare to tabu search or simulated annealing for instance, are not particularly informative. Instead, a more informative question would be; how does the multilevel version of a particular tabu search algorithm compare to the original tabu search algorithm, for a given problem? This concept, is best summarised in the words of Walshaw "*the multilevel framework is a collaborative framework that acts in concert with some other technique. Therefore the question is: given that I am using technique x for solving problem y, can its performance be boosted by using a multilevel version of technique x?*"[305].

The aim of the research therefore, was to determine the following:

- How would a multilevel algorithm for the VRP compare to an equivalent single-level counterpart, i.e. could the multilevel algorithm improve the asymptotic convergence in the quality of solutions produced by the single-level's local search heuristic(s) and or improve the convergence rate of the single-level's local search heuristic(s).

- The second phase of the research had similarly specific aims but was also complemented by aims that are more general. Namely, could the multilevel algorithm developed for the VRP be extended to a similar but sufficiently different type of problem such that the performance across the problem types could be distilled into generalisations regarding the multilevel technique and its suitability for these and similar types of problems. To this end, the research aim to determine the suitability of the multilevel algorithm for the capacitated p-median problem, comparing it to an equivalent single-level counterpart. Again, determining the impact the multilevel algorithm had on improving the convergence rate and the asymptotic convergence in the quality of solutions produced by the single-level's local search heuristic(s).
- Previously, the multilevel technique was applied to other combinatorial optimisation problems; the one of particular interest was the traveling salesman problem. The research aims to extend this body of work by applying the multilevel technique to other related combinatorial optimisation problems.

1.4 Research Contributions

The research developed and implemented multilevel and single-level algorithms for the capacity vehicle routing problem (CVRP) and the capacitated *p*-Median problem (CPMP). This research constitutes the first application of the multilevel technique in these areas. *Chapter IV* presents detailed treatment of the algorithms implemented and *chapter V* analyses the experimental results.

The research designed and implemented simple ways of constructing solutions to the CVRP acquiescent to the coarsening philosophy of the multilevel technique. The research also investigated other methods of solution construction that had not previously been advocated for multilevel algorithms, centred around separating coarsening and solution construction, this is presented in *chapter IV*. In the case of the CPMP this allowed the leading construction heuristics in the field to seamlessly amalgamate with the coarsening process. The fact that the CPMP is capacitated and the number of medians predetermined, meant the more traditional coarsening approaches faced difficulties in constructing feasible solutions of good quality. It was therefore more appropriate to use the technique of separating construction and coarsening, in creating feasible solutions to the problem. This process added a new technique to the toolkit of future practitioners. It also offered new insights and raised new questions about the coarsening process, the most important part of the multilevel technique.

The research constructed a general framework for solving instances of the CVRP and the CPMP. The similarities between the problems, chiefly the requirement to partitioning of the set of customers into feasible subsets while respecting the problem constraints and minimizing connection costs, was exploited to this end (see *chapter IV*).

The research designed and implemented enhancements to the multilevel technique for the CVRP and the CPMP, which proved productive. These enhancements were *constraint relaxation*, *coarsening homogeneity* and *solution-based recoarsening* (see *chapters IV and V*). In the case of the CPMP two multilevel algorithms were implemented, one using the tabu search metaheuristic. Both multilevel algorithms outperformed their single-level counterpart.

Coarsening is one of the most important areas for multilevel practitioners, numerous simple coarsening algorithms are presented throughout *chapters IV and V*, with experimental analysis done to demonstrate their validity, or lack thereof. Since the multilevel technique is still a new solution approach for combinatorial optimisation problems of the type presented in this thesis, researchers may query whether the technique can be adapted to their areas of interest. *Chapter III* present a wide-ranging discussion of areas where the technique has been applied, possibly increasing the likelihood the technique will further be used.

The research was also able to identify weaknesses, strengths and areas of study future practitioners could pursue: *chapter VI* deals with these subjects.

As an aid to raising the profile of the multilevel technique in the research community, the following dissemination has been undertaken.

1.4.1 Publication and Presentations from the Research

The following publications and presentations have been completed for the research contained here. In all presentations, the speaker was Demane Rodney; all papers were co-authored with Dr. Alan Soper and Dr. Chris Walshaw.

Papers:

- *Rodney*, D. Soper, A. and Walshaw, C. (2007), Multilevel Refinement Strategies for the Capacity Vehicle Routing Problem, Int. J. IT & IC, Vol.2, No.3
- *Rodney*, D. Soper, A. and Walshaw, C. (2007), The application of Multilevel Refinement to the Vehicle Routing Problem, In D. Fogel et al., editors, Proc. CISChed 2007, IEEE Symposium on Computational Intelligence in Scheduling, IEEE, Piscataway, NJ, pp.212 – 219
- *Rodney*, D. Soper, A. and Walshaw, C. (2006), Multilevel Refinement for the Vehicle Routing Problem, In Proc. ODYSSE 2006, Third International Workshop on Freight Transportation and Logistics

- *Rodney*, D. Soper, A. and Walshaw, C. (2005), The Application of Multilevel Refinement to the Vehicle Routing Problem, In Proc. PlanSIG 2005, 24th Annual Workshop of UK Planning & Scheduling Special Interest Group
- *Rodney*, D. Soper, A. and Walshaw, C.(2008) Multilevel Approaches applied to the Capacitated Clustering Problem, In H. R. Arabnia et al., editor, Proc. 2008 Intl. Conf. Scientific Computing, WorldComp'08, pp.271-277.

Presentations:

- *UK PlanSIG 2005*, The 24th Annual Workshop of the UK Planning and Scheduling Special Interest Group, City University of London, England, 15 - 16 December, 2005
- *ODYSSEUS 2006* - Third International Workshop on Freight Transportation and Logistics, Altea, Spain, May 23 - 26, 2006
- *CISched2007* - IEEE Symposium on Computational Intelligence in Scheduling, Honolulu, Hawaii, April 1-5, 2007
- *WORLDCOMP'08* - The 2008 World Congress in Computer Science, Computer Engineering, and Applied Computing, Las Vegas, USA, July 14-17, 2008
- *PhD Seminars* – University of Greenwich PhD Seminars series, June 2005, March 2006, February 2007
- *MPhil to PhD Viva* Presentation - University of Greenwich

The research demonstrated that for the CVRP and the CPMP, the multilevel technique provides significant gains over its single-level counterparts. In all cases, the multilevel algorithm was able to improve the asymptotic convergence in the quality of solutions produced by the single-level's local search heuristics. The multilevel technique did not improve the convergence rate of the single-level's local search heuristics in all cases. However, for large-scale problems the multilevel technique was found to scale in a manner superior to the single-level technique. It was also demonstrated that for sufficiently large problems, the technique was able to improve the asymptotic convergence in the quality of solutions at a sufficiently fast rate, such that the multilevel technique was able to produce superior results compared to the single-level version, without refining the solution down to the most detailed level.

1.5 Outline of the Thesis

Chapter II: **Routing and Location: a Review**. This chapter presents a recent literature survey of the state of research in location and routing. The vast nature of these two subject areas precludes comprehensive treatment. Instead, key contributions are reviewed and it is demonstrated firstly, that the answers to the research questions are valuable to the research community, and secondly, the questions are unanswered.

Chapter III: **The Multilevel Technique.** This chapter presents the central methodologies of the multilevel technique and analysis of the technique through its application to a series of diverse problems. It is demonstrated that the technique is highly adaptable to a vast array of problems and is capable of aiding the solution process in most of the areas it has been applied. The chapter also highlights potential pitfalls to be aware of when using the technique.

Chapter IV: **Multilevel Technique for Routing and Location Problems**. This chapter presents the multilevel framework for the capacity vehicle routing problem (CVRP) and the multilevel framework for the capacitated p-Median problem (CPMP). This chapter along with the following chapter presents the main body of work done during the research. The heuristics and metaheuristics implemented throughout the research are presented and their impact analysed.

Chapter V: **Computational Results for the CVRP and CPMP**. This chapter presents the results of the experimental analysis done for the CVRP and the CPMP. Along with the preceding chapter, this chapter presents the main body of work done during the research.

Chapter VI: **Multilevel Refinement CVRP and CPMP: an Evaluation.** This chapter evaluates issues relevant to the multilevel technique implemented for the classes of location and routing problems studied. This chapter also presents further works and concluding thoughts on the relevance of successfully answering the research questions

Chapter II

2 Routing and Location: a Review

This chapter presents a recent review of the state of the field in the areas of routing and location analysis. It covers the problems studied in this research and the leading solution techniques for those problems.

2.1 Combinatorial Optimisation Problems

In theory and practice, we are often interested in choosing the 'best' solution from a set of finite or numerable infinite solutions. The interest in these combinatorial problems partly arises from the fact that finding the best or a better option than that currently held potentially leads to economic benefits. However due to the exponential growth in the number of possible solutions as the size of the problems increases it is often impractical, if not impossible, to find the best solution.

Formally:

We seek to minimize/maximize the objective function *f(s)* for a combinatorial optimisation (CO) problem $P = (S, f)$ where $s \in S$. S is the set of feasible solutions termed the search space. The objective function is given by $f : S \to R$ where R is the set of reals. An optimal solution $s^* \in S$ for a minimisation problem is then given by $f(s^*) \le f(s)\ \forall s \in S$ and $f(s^*) \ge f(s)\ \forall s \in S$ for a maximization problem. Currently there are no known polynomial time algorithms for finding s^* for combinatorial optimisation problems that are *NP-hard* [102]. This means the practitioner often has to resort to *heuristic* approaches to engage problems of reasonable sizes. Combinatorial optimisation (CO) problems are found in varied problem domains from auctioning [300] to genome theory

[44] however, routing and location are the two areas of CO that are of interest in this thesis.

2.1.1 Routing and Location

Routing and location are two areas of combinatorial optimisation that are of increasing importance. The economic benefits, for example in reduced operating costs to the service and other industries, that can be achieved by more efficient routing have been shown [60]. The adverse environmental and financial impact of poor routing has also been recognised [71]. In worst cases, this can be extreme – a recent study [257] estimated that traffic congestion in a region of New Zealand cost the economy 1% of GDP. Furthermore, as industry moves increasingly towards just-in-time manufacturing, the adverse effect of inefficient routing is magnified [258]. This environment therefore necessitates highly efficient routing models and solutions.

Location science [129] is a cross-disciplinary field, spanning both academia and industry, and encompassing such diverse professionals as engineers, economists, distributions analysts, geographers and computer scientists. It generally addresses the important question of how to optimally locate (position) resources. The location of the resources in question will invariably involve cost implications. In some cases significant costs are involved [235], [309], with huge overheads if poor decisions are made.

Location scientists have many roles. Not only must they decide how best to locate the resources to meet the demands of the known/expected customers (demand nodes) [8], but also they may need to evaluate the effect of other resources already located. For example, Wang et al. [306] model the opening and closing of facilities taking into account budget constraints and operating costs. Furthermore, there can be important environmental considerations involved in location decisions particularly from the effects of routing.

Location scientists generally use a combination of *descriptive* (i.e. the analysis of the factors governing suitable location sites) and *normative* (i.e. the application of quantitative methods for finding good solutions) approaches [75]. From these they can develop rich location models capable of determining the effect of market forces, suitable construction sites, the effects of communication and routing, etc. Such models can then be analysed and solved to provide optimal or near optimal solutions. However, this goes beyond the scope of the thesis and here we concentrate on normative approaches for a specific class of location problems.

These two exciting and interesting areas of research, routing and location, are reviewed.

2.2 Capacity Vehicle Routing Problems (CVRP)

The vehicle routing problem (VRP) [68] describes a group of problems concerned with the collection and or delivery of customers' orders. Design considerations such as delivery frequency, service times and fleet sizes lead to a rich array of problems [155].

The capacity VRP (CVRP) models the situations where a fleet of homogenous vehicles of fixed capacity is used to deliver goods to a set of customers of known demand along routes originating and terminating at one depot. If the requirement that all vehicles are homogenous is eliminated this gives rise to the fleet size and mixed vehicle routing problem (FSMVRP) [123]. This problem is reviewed in [228] and [253].

The multiple depots VRP (MDVRP) [311] extends the CVRP by allowing multiple depots from which to service the customers' demands, however the problem typically specifies that routes should terminate at their starting depot [227]. The open VRP

(OVRP) [259],[265] allows routes to terminate at any depot. This type of problem is often encountered in cases where the 'supplier' of the demand is not responsible for delivering the demands, for example in the cases of subcontractors used to deliver newspapers. Li et al. [171] outline the latest work.

The VRP with time windows (VRPTW) [273] extends the CRVP by specifying a time interval during which customers' demands must be delivered. The VRP with pick-up and delivering (VRPPD) and the vehicle routing problem with backhauls model the situations where the vehicles both deliver and collect from the customers [181]. Recent work includes [21], [185] and [72]. The Stochastic VRP (SVRP) [22] is concerned with problems where aspects of the customers' state e.g. demand or location information is not known during planning.

An assumption inherent to all versions of the problem listed above is that the planning session does not exceed one day. The periodic VRP (PVRP) [19], [7] straddles the boundary between vehicle routing and scheduling such that customers can require delivery at a particular frequency over a stated number of days.

The VRP has been extensively reviewed see [30],[31],[35],[36],[56], with the CVRP and the VRPTW being the most studied members of the group [213].

This thesis concentrates on the CVRP, a formal definition of which follows.

2.2.1.1 Capacity Vehicle Routing Problem

The CVRP requires the creation of a set of vehicle routes originating and ending at a depot O, serving the demands $d_i > 0$ of n customers, for $i = (1,2,3\ldots n)$. The demand of the depot O is zero. A non-negative cost, representing distance or journey time, is defined between any pair of customers $i \neq j$ as C_{ij} and between every customer and O. We assume that the costs are symmetric so that $C_{ij} = C_{ji}$. The depot holds V identical vehicles of capacity Q. The total demand of customers on a route must not exceed this upper capacity Q. An additional requirement is often added that the cost of a route, given by the sum of the costs between customers on the route plus the service cost of the customers and the cost to and from O, should not exceed an upper cost M (perhaps relating to the maximum distance/time a vehicle can travel). If the cost restriction is included the problem is termed the distance constrained VRP (DVRP) [180]. The solution then seeks to minimise the total cost of the routes.

The techniques developed for the VRP by our research were applied directly to both the CVRP and the DVRP. For the remainder of this thesis when speaking about the entire group of problems these will be referred to as the VRP. When speaking about the CVRP and the DVRP these will be referred as the CVRP, but it is understood to mean both problem types.

2.3 Solution Techniques for the CVRP

The VRP is known to be NP-hard [168]. Exact methods, heuristics, approximation algorithms [9], [18] and metaheuristics are the main techniques used for generating solutions to instances of the VRP.

Most of the optimal values obtained for the problem have been through the use of exact methods. These techniques are however outside of the scope of this thesis and the interested reader is directed to the PhD thesis of Ropke [246]. Indeed, there currently exists no exact method capable of routinely providing solutions to instances of over 100 customers [147], [292] . Since there are real world instances of the problem consisting of thousands of customers [42], heuristics and metaheuristics have emerged as the dominant solution techniques.

The research in these two areas for the VRP is enormous. This section outlines some of the main trends and where appropriate surveys and original contributions are highlighted.

2.3.1 Heuristics

Heuristics are solution techniques capable of producing an 'acceptable' solution but unable to guarantee an optimal solution [270], [97]. A*pproximation algorithms* are heuristics capable of providing a worst-case guarantee. For a CO problem as defined in section 2.1, if a solution s_w is returned by an ε- approximation algorithm the following is satisfied: $\left|\frac{f(s_w)-f(s)}{f(s_w)}\right| \le \varepsilon, \quad \forall s \in S$ for some factor ε where S is the search space. In some cases however ε can be very large and therefore be a poor guide to the actual performance of the algorithm [270].

Heuristics provide a superior means of handling the complexity of CO problems when compared to exact methods. The solutions they produce, however, are often local optima which are potentially substantially worse in quality compared to the global optimum. This occurrence, that the local optima can be of very poor quality, termed the central-limit catastrophe [183], [17], is one of the main justification for metaheuristics.

If a local optimum solution s_w is produced by a heuristic for the CO defined above, the solution adheres to: $f(s_w) \le f(s), \quad \forall s \in N(s_w) \subset S$ where $N(s_w)$ is the neighbourhood of s_w and S is the search space. $N(s_w)$ defines all solutions accessible from s_w by the heuristic under consideration, see section 2.3.1.4.

2.3.1.1 Heuristic Characteristics

A good heuristic is one that is *accurate*, *fast*, *simple*, *flexible* [55] and *stable* [100]. A stable heuristic maintains an 'appropriate' degree of separation between the data of the problem being solved and the heuristic. This separation results in heuristics that are capable of adapting to unforeseen changes in the operational environment. If the heuristics take into account none of the features of the problem data this tends to result in solutions of poor quality, however overly coupling the two produces heuristics unable to handle changes [130],[158]. These characteristics, while being subjective, will be used as a guide in analysing the main heuristics currently in use for the VRP.

2.3.1.2 Heuristic Types

Construction heuristics and *improvement* heuristics [226] represent the two main types of heuristics used for routing problems. Laporte et al. [165] and Laporte and Semet [166] refer to these as classical heuristics.

Construction heuristics create a solution without taking any steps to modify the solution with respect to the objective function. *Improvement heuristics* on the other hand systemically modify an existing solution in search of a new local optimum. Where elements of both are combined (and the resulting combination is not recognized as a metaheuristic); it is not uncommon for it to be termed a *composite heuristic* [90],[226]. Laporte et al. [165] provides a survey of heuristic approaches to solving the VRP, Funke et al. [98] provide a survey of local search techniques used for the VRP.

2.3.1.3 Construction Heuristics

For problems that can be modelled as integer programming problems (this includes the classes of routing and location problems), the two dominant heuristic construction philosophies are the greedy *add* and *drop* approaches. The add approach starts with all decision variables equal to zero, then iteratively sets the variable yielding the best return to one until a valid solution is constructed. The drop approach is the opposite. It starts with all variables set to one and if this represents an infeasible solution, the variable yielding the least return is removed until a feasible solution is obtained.

Three of the main types of construction heuristics applied to the VRP are the *savings* method (a drop approach), *insertion* principles (an add approach) and *clustering* heuristics [121], [165]. Clustering heuristics can be constructed using both philosophies.

2.3.1.3.1 Clark-Wright Savings Heuristic

The Clark-Wright savings heuristic (*CWS*) [51] is the most popular of the savings methods. In its original form, it is simple, fast, parameterless [55] and stable. Because of its simplicity and speed, it is one of the most commonly used in commercial routing software [5].

The CWS can be stated as follows: Given a VRP, connect each customer to the depot to form a route. Using the savings criteria $S_{ij} = C_{io} + C_{jo} - C_{ij}$, merge routes in pairs until no feasible merges can be created while respecting the problem constraints. C_{io} represents the cost between a customer i and the depot and C_{jo} represents the cost between a customer j and the depot and C_{ij} represents the cost between customers i and j. S_{ij} represents the change in cost of the solution if the routes serving i and j are merged.

If a route is selected and continually merged with routes yielding the greatest saving (while feasible to do so) this results in the *serial* CWS. If however, at each iteration the two routes yielding the greatest saving if merged are chosen, this results in the *parallel* CWS [5].

A major weakness of the CWS is the preference given to customers the further away they are from the depot. This results in the construction of circumferential routes and routes of poorer quality as the construction process progresses [140], [5]. The introduction of new parameters to the saving criteria has been advocated to handle this weakness: to reduce the likelihood of constructing circumferential routes a positive factor is applied to the cost between each customer to be merged [104], [315]. To limit the dominance of customers further from the depot over those closer, Paessens [211] proposed a modification to the savings criteria that returned higher saving for customers that were similar distance to each other from the depot. Recently Atmel and Oncan [5] proposed another improvement that influences the savings criteria based on the demand of the customers. Each of the proposed improvements adds a parameter to the savings criterion.

These modifications improve the quality of the solution produced by the CWS [1], but the resulting heuristic is no longer parameterless and has lost simplicity. When added to the fact that the CWS is normally used as a construction heuristic for more complicated metaheuristics, the trade-off in added parameters and complexity means the original CWS is often used [140].

A nearest neighbour heuristic [120],[272] can be constructed for the VRP along the same principles as used for the saving heuristic. If the depot is ignored, customers with the least cost between them can be joined to form sub-routes using either a serial or a parallel approach. These are then connected at the end to the depot to form a solution.

2.3.1.3.2 Sweep and Petal Heuristics

Originating from the work of Wren and Holiday [311], and Gillett and Miller [112] the sweep heuristic is the best known and most used of the clustering heuristics. The sweep heuristic is slower than the CWS and more complex.

The heuristic orders customers by increasing polar angle, calculated based on the depot, whose polar coordinates are set at (0, 0). While the problem constraints can be respected, customers are added to a route in order of increasing polar angle. This process is then repeated until all customers are part of a route.

Petal heuristics [96],[252],[225], are a specialization of the sweep heuristic. Customers are ordered as in the case of the sweep heuristic and sets of single routes and double interlocking routes are constructed that respect the problem constraints. A set partitioning problem [29] is solved over the set to select the best combination of routes forming a solution.

2.3.1.3.3 Insertion Heuristics

Fast, simple and stable insertion heuristics have been used for constructing VRP solutions [39]. Early insertion heuristics where proposed by [184] and [49].

The principles used by insertion heuristics to construct a solution for a given VRP can be stated as: while there are customers not served by a route, insert customers on a route while feasible to do so. Parallel insertion heuristics construct multiple routes simultaneously while the serial version constructs one route at a time. The customers selected to be inserted and the route and location of the insertion is typically chosen to have the least effect on the solution cost. However, insertion heuristics are outperformed by savings heuristics [55].

2.3.1.4 Improvement Heuristics/ Local Search Heuristics

For a given solution s in the search space S, the neighborhood of s, $N(s) \subseteq S$ is the set of solutions obtainable from s by modifications applied to s. A move is a series of modifications that transform one feasible solution to another, while improving moves are those that result in solutions of better quality with respect to the objective function.

Local search heuristics are a means of executing moves on s, transforming it to a neighbouring solution s^*. If the net effect of the moves executed is improving, s^* will be better in quality when compared to s.

The set of moves permissible by the local search heuristic and the state of s defines the topology of $N(s)$.Where there exist more than one solution in $N(s)$ of better quality than s, then the quality of s^* will be influenced by the improvement strategy implemented. A *first improvement* strategy selects the first solution better than s found while a *best improvement* strategy selects the best improving solution in $N(s)$. While the first improvement strategy normally results in reduced runtimes, best improvement typically leads to better solutions [166].

Local search heuristics for the VRP principally fall into two main classes; edge exchange heuristics or node exchange heuristics.

2.3.1.4.1 Edge Exchange Heuristics

Edge exchange heuristics remove a given number of edges from a route and reconnect the resulting sub-routes to form a feasible route. There exist many variants on this basic concept. Although complete analysis of each is outside the scope of this section, the main concepts will be analysed with greater details provided by Funke et al. [98] and Irnich et al. [147].

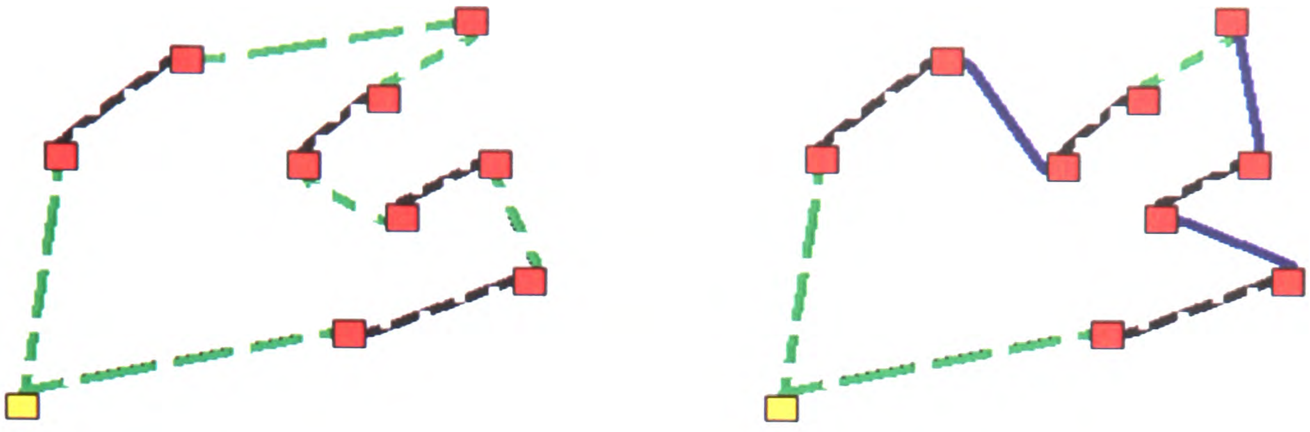

Figure 1 An example of a 3 - Opt Exchange

The *k-opt* edge exchange heuristic [172],[63] removes *k* edges from the route and replaces them with *k* new edges, typically continuing while there exist *k* edges as yet unmoved from the route, or while there is an improvement to be found from executing the procedure. Figure 1 shows an example for *k* equal to 3, termed the 3-opt exchange. Potvin et al. [218],[219] generalised the *k-opt* procedure allowing for a crossover move that breaks an existing giant tour into *k* routes or exchanges *k* sub-routes between *k* routes of a solution. This modification is denoted *k-opt**.

A route is said to be *k*-optimal if no transformation by the *k-opt* heuristics would lead to an improvement. Potvin et al. [218] also demonstrated that a *k-opt** optimal route was also k-optimal but not visa versa.

When a *k-opt* move is executed, a subset of the potential moves requires sub-routes to be inverted. The *Or-opt* moves, originally attributed to Or [203] and again outlined in a VRP context by Potvin and Rousseau [219], is the subset of the *k-opt* moves that preserve the orientation of the original sub-routes, leading to speed ups in the implementation.

In passing, it must be noted that Lin and Kernighan [173] made additional restrictions on the edges removed from the tour, resulting in an edge exchange heuristic of greater efficiency than the generalized cases. Detailed consideration of this involved heuristic and its implementation are provided by [135].

2.3.1.4.2 Node Exchange Heuristics

Node exchange heuristics transfer a set of customers (transferred set) from one location in a route to another within the route or to a different route. If x represents the total number of customers in the route, different node exchange heuristics allow the cardinality of the transferred set to range from one to x inclusive. If the cardinality of the (replacement) set of customers used to replace the set transferred is greater than zero, this is referred to as an *exchange*. If however, the empty set is used as replacement this is termed *relocation or transfer*.

2.3.1.4.3 λ- Interchange Heuristic

The λ- Interchange is a node exchange heuristic due to Osman [206] that allows both exchange and relocation. A positive integer parameter, λ, is used to specify the maximum number of nodes that can be transferred from a single route. The transferred nodes keep their original order and take the position of the nodes being replaced.

For an *exchange*, the cardinality of the transferred and replacement set is greater than zero and less than or equal to λ. All valid combinations of cardinality within the specified range are allowed by an exchange move. In *relocation,* the cardinality of the transferred set is greater than zero and less than or equal to λ.

Relocation and exchange are executed between pairs of routes. The quality of the new solution can be determined from the change in solution cost. This is obtained by comparing the cost of the edges removed from the routes and the edges introduced by the interchange. The λ- Interchange can be implemented using first improvement or best improvement and is typically executed across all possible pairs of routes in the solution.

Funke et al. [98] states the complexity of the heuristic [1] can be $O(n^{2\lambda})$, Where n is the number of nodes. Hence, most implementations use $\lambda = 1$.

2.3.1.4.4 Cyclic Transfer Heuristic

The cyclic transfer heuristic [286] transfers sets of nodes between two or more routes and allows both exchange and relocation. Cyclic transfers are more generalised than λ- Interchanges. The cardinality of the transferred and replacement sets are the same as defined for the λ- Interchange and the nodes keep their original order. However, there are a number of important differences. Cyclic transfers allow the *inserted* nodes (the members of the transferred set) to take positions in the route different from those occupied by the ejected nodes (the members of the replacement set).

A cyclic transfer is specified as '*B*-cyclic *M*-transfer' [286]. *M* states the maximum cardinality of the transferred and replacement sets. B (the cyclic depth) states the number of routes in the transfer, and is equal to or greater than two. A series of exchanges therefore transfers a set of nodes from the first route in the series to the next until a set of nodes is transferred from the last route to the first. The transfer of the empty set terminates the series with relocation. Cyclic transfer results in a larger neighbour search than λ- Interchange.

2.3.1.4.5 Node Ejection Chains

Node ejection chains [117], [223] are node exchange heuristics of almost identical principles to cyclic transfers but typically applied to individual routes. A set of nodes is transferred from one position in the route to another from which a further set is ejected and transferred to another location. If the transfer consists purely of exchanges then the

[1] Variants of the λ- Interchange are used throughout this research with $\lambda = 1$ because of the complexity of the procedure.

last set of nodes is transferred to the position occupied by the first set transferred. A series of exchanges can also end with relocation. The cardinality of the transferred set is typically taken to be one and the nodes keep their original orientation. The nodes ejected and inserted are taken not to be adjacent as this result in cost independent calculations.

2.3.1.5 The Split Procedure

The Split procedure takes an existing VRP solution and constructs a set of routes typically different from the routes in the original solution and in some cases of lower cost.

The procedure constructs a Giant Tour [290] (a Hamiltonian cycle linking all the customers in a CVRP solution, but not the depot), and then finds its optimal partition into routes. The partition to select the optimal combination of routes [225] can be found by solving a set-partitioning problem [29]. The procedure can be performed in $O(n^2)$ time [221], [55] where n is the number of customers in the solution. The routes it produces respect the constraints of the problem. Prins [221] used this procedure to find improvements to VRP solutions.

2.3.2 Metaheuristics

Metaheuristics have produced the best results on medium and large VRPs [55]. A metaheuristic can be viewed as an iterative process that guides the operation of heuristics to produce a more efficient solution than that obtainable by the heuristics acting alone [208].

With this aim, a metaheuristic seeks to find good solutions and efficiently explore the neighbourhood of these solutions, a process called *intensification* [28]. This process is counter-balanced by *diversification,* where the search is moved to unvisited regions of the search space [28], (if these can be identified), to avoid getting trapped in local optima early in the search process.

The top performing metaheuristics often use '*memory of the search process*' to manage the oscillation between intensification and diversification [280], and it is effective use of these three features that drive modern metaheuristics to "higher performance" [28].

Over the last 20 years, much research has focused on metaheuristics. This huge body of work has been extensively surveyed and the interested reader is referred to the following: Gendreau et al. [109], Hertz and Widmer [136], Taillard et al.[280], Blum and Roli [28], Gendreau and Potvin [110], Glover and Kochenberger [119], Pardalos and Resende [209], Walshaw [304], [305].

Metaheuristics can be broadly classified into *single solution metaheuristics* where only one search trajectory is explored at a time and *population metaheuristics* where multiple search trajectories are explored simultaneously. In the following, two *single solution metaheuristics,* Simulating annealing and Tabu search, and two *population metaheuristics,* Evolutionary algorithms and Ant colony search, are reviewed in turn.

2.3.2.1 Simulating Annealing

The Simulating annealing (SA) metaheuristic [43], [156] includes a strategy to escape local optima which is based on using a 'temperature' parameter analogous to the annealing process in solids. Annealing is used to create solids with energy states lower than those initially held, by liquefying the solid through heating then controlling the process of cooling such that energy equilibrium is achieved throughout the solid at each stage of cooling.

With SA, the solution is equivalent to the state of the solid and the solution cost to the energy state. During an iteration a solution in the neighbourhood of the current solution is randomly selected, if the solution is of a better cost than the current solution it is accepted, otherwise the new solution has a probability of being accepted based on the difference in cost between both solutions, and the value of the temperature parameter. If the temperature parameter has a high value and the change in cost is small the probability of acceptance is high. As the solution process progresses the temperature parameter value is lowered until uphill moves (solutions of worse cost) are not accepted. The process then terminates at a local optimum. SA is a memory-less metaheuristic (unlike tabu search) but the allowing of uphill moves makes it a more powerful metaheuristic than simple iterative search.

The performance of the SA metaheuristic is heavily dependent on the process used to control the temperature parameter [28]. It is theoretically possible to devise cooling strategies that converge (the solution) to a global optimum [1], however these strategies are too slow for practical purposes.

2.3.2.2 Tabu Search

Tabu search (TS), proposed by Glover [113],[114], is possibly the most used metaheuristic [106],[107], [115],[116],[118]. TS uses short-term memory, typically in the form of a (tabu) list to record attributes about recently visited solutions. The search is then not allowed to visit solutions whose attributes are currently stored in the list.

The 'best' solution accessible from the neighborhood of the current solution is chosen and added to the list, with a solution currently on the list removed, normally in a first in first out order. This allows the metaheuristic to accept uphill moves, thus navigating away from local optima and prevents cycling between often-visited solutions. The length of the tabu list is a key parameter in the operation of the algorithm and various strategies are employed varying from a static list to dynamically adjusting the length of the list during the search process.

Efficient execution of the algorithm necessitates the storing of solution attributes instead of entire solutions; this means multiple solutions can be tabu at once. To minimise the effect of improving solutions being tabu-ed without having been visited, attributes are typically ascribed an *aspiration criteria* (conditions under which the tabu state can be ignored). If an unvisited solution contains a tabu-ed attribute and this attribute's aspiration criteria are met, it is then possible to visit that solution.

2.3.2.3 Evolutionary Algorithms

Evolutionary algorithms describe a broad class of metaheuristic strategies, the main one being genetic algorithms [142]. They are modelled on the evolutionary processes in nature and operate on a population of solutions simultaneously. Solutions can be thought of as sets of genes with each solution having a level of fitness often related to the quality of the solution with regard to the objective function. Two or more solutions (parents) can be combined using the *mating* process to produce new solutions (offspring) or a single solution can be transformed using the *mutation* procedure. A cycle of mating and mutation represents on iteration of the metaheuristic.

The metaheuristic defines a process to choose solutions currently in the population to form the parents of the next generation of solutions. Solutions with higher levels of fitness have a greater possibility to be chosen, solutions not chosen are discarded. Once the parent solutions are selected, the *mating* process combines them typically using a *crossover* operation in which a point is selected in each solution to be mated and the section of the solution to the left or right of this point is interchanged to produce new solutions. The parents and offspring solutions or just the offspring solutions can be propagated to the next generation. This, population management, is a key consideration with evolutionary algorithms.

Once the population has been constructed for a particular generation the *mutation* process is normally executed. The classic *mutation* process involves small random changes to the individual solutions in the population. More successful mutation strategies use local search algorithms to drive each solution to a local optimum [220].

Evolutionary metaheuristics typically terminate after a number of iterations have been executed without any improvements or, in case where the population is not kept constant but allowed to decrease after each iteration, terminations occur when only one solution is present.

2.3.2.4 Ant Colony Search

Ant colony optimisation [77] is modeled on the operations of ant colonies as they forage for food. In the case of real ants, pheromones are placed along the paths they travel from food source to nest, and since they seek to take the shortest path, this will emerge as the path with the highest concentration of pheromones.

In the ant colony optimisation technique, ants are represented by heuristics and feasible solution components are analogous to food sources and the ants' nest. The connections between components are equivalent to the real world paths. Each connection and solution component defines a pheromone parameter. The pheromone parameter provides the memory for the metaheuristic.

In order to assimilate components into a solution, 'ants' in a probabilistic model use the relationship between solution components with regard to the objective function and the value of the pheromone parameter along a given connection.

For a routing problem, if an 'ant' is located at a customer whose demand has been processed, the next customer to be visited has a probability of being chosen based on the cost between both customers and the number of times the customers have been visited consecutively.

When all 'ants' (an important parameter in the metaheuristic) have completed a solution, the solution with the most traveled connections represents the preferred final solution. An evaporation process that modifies the pheromone parameters is used to change a solution for additional improvement.

2.3.2.5 Hybrid Metaheuristics

There is currently substantial research interest in the field of hybrid metaheuristics [28], [281]. These metaheuristics seek to combine ideas from single solution metaheuristics and/or population metaheuristics in order to optimally address the key issues of *intensification*, *diversification* and the *memory* of the search. These are issues metaheuristics, must effectively address in order to achieve high quality solutions [280]. A number of hybrid metaheuristics have been proposed for optimisation problems [281].

2.3.3 Performance of Solution Techniques for the CVRP

Ideally, different solution techniques should be compared using the same test instances, in the same operating environment. This however is almost never the case in the reported literature, as noted by numerous authors [13], [148], [165]. This fact must be kept in mind when analysing the following comparisons.

The comparisons presented here, were all done using the Christofides [49] test instances.

2.3.3.1 Heuristics Performance for the VRP

The review of Laporte et al. [165] identifies the CWS, the sweep and the petal heuristics as the dominant construction heuristics for the VRP. Their investigations of insertion heuristics suggest they were not comparable to the others.

They implement both parallel and serial versions of the CWS with and without a 3-opt post optimization phase. The 3-opt post optimization phase is tested using both first and best improvement. They conclude that "the best solutions were produced using the parallel CWS combined with best improvement 3-opt". This produced results 6.71% above the best known values.

An implementation of the sweep and the petal heuristics [225] showed that the sweep heuristic returns results an average 7.09% above the best known compared to 5.85% for the petal. The reported implementation of the sweep heuristic includes a 3-opt phase, while the petal algorithm included a 4-opt* phase. An enhanced version of the petal heuristic, also including a 4-opt* phase, allowing for the construction of interlocking routes, was also implemented [225]. This produced results an average 2.38% above the best known. Assuming the improvement heuristics embedded in these construction procedures have similar effect it appears that the petal based approaches are superior to the savings heuristics, but the variance in implementations make this a difficult conclusion to state.

The CWS appears to be the superior heuristic in terms of runtime, regularly outperforming all the others in this area [55], [165].

2.3.3.2 Metaheuristic Performance for the VRP

The conclusions presented in the literature are that tabu search is the "most successful metaheuristic approach" [165], [55] for the VRP. The dominant tabu search metaheuristics have mostly adopted features from other metaheuristics and combine these with the principles of tabu search.

The *adaptive memory procedure* [239] is chief among these. It uses a pool of good solutions to produce new solutions during the search process. The new solutions are produced by selecting routes and assigning them a weight, in favour of routes originating from the best solutions. Steps are taken to ensure the resulting solution's feasibility, as no two routes can share a customer. The improvement phase at each iteration is then driven by a tabu search. The metaheuristic reported finding the best-known values for all the Christofides instances [239]. The process of having multiple solutions however has similarities to the group of population metaheuristics, of which tabu search is not a member.

A number of other equally competitive metaheuristics based around tabu search have been reported. *Tanuroute* [108] and *Taillard tabu search* [279], which returned solutions an average 0.86% and 0.06% above the best known respectively. In the same category of performance is the *granular* [293] and the *unified* [54] tabu search that produced solutions an average of 0.64 % and 0.69 % above the best known respectively.

A number of other metaheuristics have been able to compete with tabu search. Some of the main ones are a genetic algorithm [221] using the CWS during the construction phase, and a 2-optimisation phase at each iteration, reported results 0.23 % above the best known. A simulated annealing variant [81], called record – to – record travel [45] [170], returns results 0.41% above the best-known values [170] for a subset of the Christofides instances. The granular tabu search was applied to the same subset of problems and returned results 0.47% above the best known values [170]. Finally, a hybrid guided local search and evolution strategies metaheuristic [180] produced results 0.03% above the best known values.

2.3.4 Summary for the VRP

As demonstrated, the VRP is an active research area of academic and economic importance. Of the numerous solution techniques for the VRP, the tabu search metaheuristics consistently lead in terms of solution quality.

Savings heuristics are the leading construction approach for the problem while edge and node exchange heuristics form the set of leading improvement approaches. As a result it is of interest to the research community to determine ways of improving their performance. This is an issue addressed in the latter chapters of this thesis.

2.4 Facility Location Problem

This section of the review provides a broad overview of the field of location analysis. It also highlights the areas in the field necessary to appreciate the recent work done on introducing multilevel refinement to the field of location analysis.

Microeconomic and Macroeconomic location planning constitute the field of location planning. *Macroeconomic* location planning is concerned with the optimal distribution of economic sectors and industries [75] and goes beyond the scope of this thesis.

Microeconomic location planning is concerned with the optimal location of resources/ facilities in a spatial context [75], *and* covers the areas of *location* and *layout problems*. The two areas are differentiated by the fact that the facilities to be located in *location problems* are relatively small compared to the space they are located in [234] while *layout problems* are concerned with facilities that are relatively large compared with the space they are located in [234]. Typically, with location problems, we are interested in locating a store, a server, etc., while layout problems would typically be concerned with the layout of departments within a store or of components within a server. We focus on location problems.

Location problems are found in a variety of settings such as the location of production centres [191], emergency services [69], assigning candidates to test centres [60] and storage of hazardous materials [50]. Increasingly they can also be found in less traditional areas such as product positioning [194], candidates' campaign in political sciences [196] and the classification of apparel sizes [294]. Surveys on the class of location problems are provided by [66], [70], [78], [79], [83].

An important application of location analysis is found in the fields of data mining [316] and clustering [260], [132]. Data clustering, a technique central to pattern recognition [195], knowledge discovery [277], image processing [312] and computational biology [269], is concerned with the partitioning of n data points in m-dimensional space into k clusters to maximise similarities between data of the same clusters. The measure of similarity is based either on the similarity of the data in a cluster to some data central to that cluster, or on the similarities between the members of the cluster being greater than similarities with members of other clusters. This conforms to the model of various location problems, chief amongst which are the *p-centre* and *p-median* problems.

2.4.1 Types and Classes of Location Problems

There are numerous ways of classifying location problems. The classifications are dependent on the situations the problems are modeling and so this section describes the main classifications and their relation to each other.

Location problems are differentiated by three main features: their objective functions, where facilities can be located and the problem space. The space consideration typically results in location problems that are either classed as *planar location problems* and solved in 2-dimensional real space or as *network location problems* which are solved on an underlying network [234].

When the facilities can be located anywhere on the network or in the plane, these are termed *continuous* location problems; if there is a restriction to a finite number of possible locations, these problems are termed *discrete* location problems [234].

Planar location problems are typically NP-Hard [234]. For continuous planar location problems, there may exist a global optimum at the demand points as identified by Kuhn [162], and a simplification is to restrict the search for locations to these points. For a survey of these types of problems see [80], [308].

The relations of the predominant objective functions for location problems can be stated as locate the facilities such that customers are:

1 *as close as possible* to their closest facility.

2 *as far as possible* from their closest facility.

3 *no further than* a given distance from their closest facility.

4 *no closer than* a given distance from their closest facility.

5 *evenly distributed* between the facilities.

Where the facilities offer services that are predominantly '*desirable*' these are referred to as pull objectives [85] and correspond to either relations 1 or 3. The problems corresponding to relation 3 are referred to as *center* or *minmax problems* [236] and model the situations where a minimum guarantee of service is required.

A general objective function, referred to as *minsum* [236], is provided by [99] for the problems corresponding to relation 1. A study of this objective function reveals there to be two main situations these problems model: that of maximizing the demands that a facility serves i.e. the *capture problem* [233]; or minimizing the cost from each facility i.e. the *median problem* [307], [308].

Where the facilities to be located are predominantly '*undesirable*', such as the location of garbage storage facilities or of nuclear sites, these correspond to either relations 2 or 4. While the distance between customers and facilities must be maximized, the effect on cost must also be considered. For example, if a nuclear plant providing electricity is the undesirable facility, the solution of locating the plant some huge distance from the customers would be prohibitively expensive in the resulting cost of transmitting electricity and of course, usually impractical.

Where the facilities to be located are distributed such that the cost between each customer and their designated facility is similar, or such that the number of customers assigned to each facility is equal, these, termed *balancing* or *covering* objectives correspond to relation 5. If the balancing objective is used to locate multiple facilities and the number of facilities is a decision variable, this corresponds to the location set covering model [291].

It was noted by ReVelle et al. [231] that *minsum* objective functions (which minimise the total cost) typically correspond to location modelling in the private sector while *minmax* objective functions (which minimise the maximum cost) were typically used in the public sector. This led to location problems being classified as *private sector* and *public sector* problems, e.g. [231]. It has been noted more recently that both types of objective functions are found in both spheres of business and this is probably not a classification model suitable for today's discussions, ReVelle and Eiselt [234]. However, they are still found in the literature.

2.4.2 Discrete Network Location Problems

This thesis includes new work on a discrete network location problem and so this section reviews some of the main areas in discrete location research (see Brandeau and Chin [24] for an exhaustive review covering approximately 40 additional problems).

Currently four problems characterize and dominate the field of discrete network location research. These are the *p-median* problem, the *uncapacitated facility location* problem, the *p-center* problem, and the *quadratic assignment* problem (QAP). Collectively, they are normally referred to as *location-allocation* problems. The QAP is in fact central to many other domains including scheduling, distributed computing and data analysis – see [174] for a survey of the QAP.

The *p-median* problem [126], [127] is known to be NP-Hard [150] and is a member of the class of *minsum location-allocation* problems. It looks at locating p facilities on a network of n nodes of while attempting to minimize the total cost of connections between facilities and demand nodes. It is sufficient to search for the locations for the facilities among the locations of the nodes, as at least one optimal solution exists within these locations – this is known as the '*Hakimi theorem*' [234].

Given a set of demand nodes, I, and a set of facilities J, where d_{ij} represents the cost of serving one unit of demand for node $i \in I$ from facility $j \in J$ and x_{ij} represent the portion of the demand of node i assigned to facility j. The demand of node i is represented by w_i. The variable y_j is set to one if a facility is located at node j otherwise it is set to zero. The number of facilities to be located is denoted by p. The p-median problem, as formulated by [230] and [234], can be stated as:

$$Min \sum_{i \in I} \sum_{j \in J} w_i \, d_{ij} x_{ij}$$

$$s.t$$

$$\sum_{j \in J} x_{ij} = 1 \quad \forall \ i \in I$$

$$x_{ij} \leq y_j \qquad \forall i \in I, \ j \in J$$

$$\sum_{j \in J} y_j = p$$

$$x_{ij} = 0 \vee 1 \quad \forall i \in I, \ j \in J$$

$$y_j = 0 \vee 1 \quad \forall \ j \in J$$

The variable x_{ij} is set to one if node *i* is assigned to facility *j*, zero otherwise. The capacitated *p*-median problem extends the basic model by enforcing a demand constraint on each facility.

The number of facilities to be located in the *uncapacitated facility location* problem (UFL) [111] is endogenous to the problem, this being the key differentiating feature between the UFL and the *p*-median problem. A cost is incurred for each facility located in the UFL, unlike in the case of the *p*-median problem, and a solution to the UFL seeks to minimise the total cost, which is a summation of the cost of assigning the clients to the nearest facility and the cost of opening each facility. Another difference between the UFL and the *p*-median problem is that representations of the UFL typically separate the set of facilities and demand points. The imposition of a capacity constraint on each opened facility of the UFL results in a harder problem to solve, the *capacitated facility location* problem. The additional difficulty arises because the capacity constraint means some nodes might not be assigned to their nearest facility.

The *p-center* problem [182] is a member of the class of *minmax location-allocation* problems. The problem is NP-Hard [149] and looks at locating *p* facilities on a network of n nodes while attempting to minimize the maximum cost between any demand node and its nearest facility. This contrasts with the p-median type problems that aim to minimise the total cost as outlined above. The two types of problem are also differentiated by the fact that the *Hakimi theorem* typically does not apply for p-center problems [234], as the optimal location of the facilities will occur on the edges of the underlying network.

For p-center problems, to locate one facility, a search for a suitable location involves finding the locations on edges of the network of minimum cost from a given demand node, then finding the maximum of those results. The best solutions then select an edge that minimises the maximum.

2.4.2.1 Other Notable Location Problems

The *location set covering problem* [232], [291] aims to locate facilities such that the maximum cost between any demand node and its nearest facility is within a specified value. The location of emergency services, as an example, can frequently be modelled as a covering problem. A survey of covering problems is provided by [263].

Competitive location problems, *capture problems*, and *location–routing problems* represent some of the future areas of exciting research in the field of location science as well as pinpointing the nexus of interaction between location problems and the other problem of extensive study in the thesis, that of vehicle routing.

Competitive location problems look at locating competing facilities with a view to maximizing their market share [143]. These problems are encountered in varied fields [8],[199],[288] and merge principles of economics, geometrics and game theory. A state of Nash equilibrium [192],[193] can be viewed as a 'solution' to the problem where, once achieved, neither competitor has any incentive to change the current state. It is known however that models based on this assumption are inherently unstable in the face of minor changes to the location rules or parameters [234].

In a competitive environment, a facility will be interested in determining the set of demand nodes that are more attracted to it than to any other facility. This is referred to as its *market area* [4] or its *influence set* [159]. If the demand nodes choose their facility on some distance metric and since the nearest neighbour relation is not symmetric, determining the influence set can be viewed as a reverse facility location problem [38] and solutions generated using reverse nearest neighbor querie*s* [314]. These interesting problems are surveyed by [84],[215],[268].

Sequential competitive location [267] is a modeling of duopolies where a secondary market player will attempt to optimize their location taking into account the location of the market leader. The market leader will then attempt to adjust their position based on the optimised location of the challenger [128]. The maximum *capture problem* [233] is the problem of finding the optimal location for the challenger. This problem is similar to other standard *capture problems* [20] where the aim is to capture as much of the 'flow' of demand as is possible. This is a model suitable for the location of fast food outlets, petrol stations etc.

*Location-routing problem*s incorporate the location of facilities by the desired objective function and then analyse the routing problem of communicating between the facilities and or the demand points. This problem stands at the junction of the two main areas of study of this thesis. It should be noted that an optimal solution to the location problem does not necessarily remain 'location optimal' once the routing considerations are taking into account. A survey of these problems is provided by [190].

2.4.2.2 Capacitated p-median Problem

The multilevel technique is applied to the capacitated p-median problem.

The *Capacitated p-Median problem* (CPMP), also frequently referred to in the literature as the *Capacitated Clustering problem* [188], has direct applications in political districting [34], vehicle routing [88],[160] and communication network design [212] among others. The clustering aspect has applications in fields as diverse as biology, economics, marketing and pattern recognition [207]. The problem extends the *p*-median problem with the addition of capacity constraints [131]. In the case of fixed medians it reduces to the generalised assignment problem [262], [251], [313],[139] and is known to be NP complete [102].

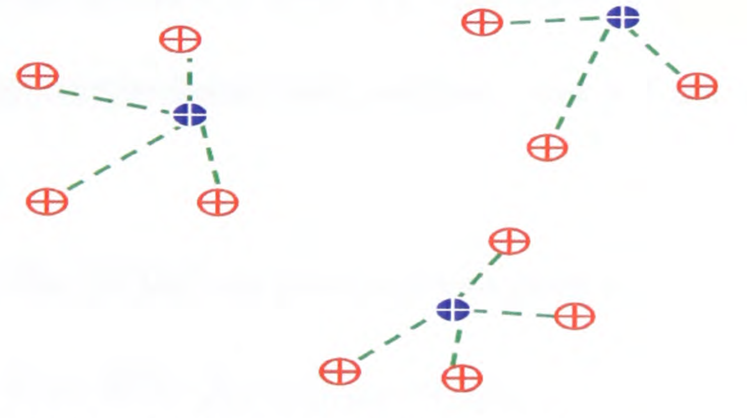

Figure 2 An example solution to a CPMP

The aim of the problem is to partition *n* demand points into *p* disjoint clusters such that a maximum capacity constraint imposed for a cluster is not exceeded and the total cost is minimised. Each demand point must be assigned to exactly one cluster. Each cluster must have exactly one median[1], i.e. the demand point from which the sum of the cost to all other demand points in the cluster is minimized. Figure 2 shows an example solution with $n = 14$ and $p = 3$. Where the medians are located at the geometric centre of the clusters this results in the *capacitated centred clustering problem* [194].

[1] Medians and facilities are used interchangeably.

A standard formal representation of the CPMP [89] is as follows:

Let $N = \{1, \ldots, n\}$ be a set of demand points and the distance between demand points i and j $d_{ij} \geq 0$ and $d_{ii} = 0$, $\forall i, j \in N$. For every demand point there is a positive integer demand q_i. A cluster is a subset of demand nodes $B \subseteq N$. The median of B, α, conforms to, $\alpha(B) \in B \mid \sum_{i \in B} d_{i\alpha(B)} \leq \sum_{i \in B} d_{ij} \ \forall j \in B \setminus \{\alpha(B)\}$. Cluster B is then feasible if the total demand of its nodes does not exceed the maximum capacity constraints for a cluster Q, $\sum_{i \in B} q_i \leq Q$. For an integer p, $2 \leq p < n$, a feasible solution to the CPMP is given by a partition $S = \{B_1, \ldots B_p\}$ of the set of demand nodes N into p feasible clusters. The cost is given by $z(S) = \sum_{k=1}^{p} \sum_{i \in Bk} di\alpha(B_k)$, and an optimal solution is given by a partition of minimum cost. Let $x_{ij} = 1$ if demand node i is a part of the cluster with median j and 0 if not. $y_j = 1$ if a median is located at j and 0 if not.

The CPMP can then be formulated as:

$$z = Min \sum_{i \in N} \sum_{j \in N} d_{ij} x_{ij} \qquad (1)$$

$s.t.$

$$\sum_{j \in N} x_{ij} = 1 \qquad \forall i \in N, \qquad (2)$$

$$\sum_{i \in N} q_i x_{ij} \leq Q y_j \qquad \forall i \in N, \qquad (3)$$

$$\sum_{j \in N} y_j = p, \qquad (4)$$

$$x_{ij}, y_j \in \{0, 1\} \qquad \forall i \in N. \qquad (5)$$

The objective of the problem is given by (1). Constraints (2) ensure each demand node is assigned to exactly one cluster. Constraint (3) ensures the capacity constraint is respected. Constraint (4) ensures that the number of clusters is equal to p and (5) states the variable definitions. The CPMP continues to be an area of active research [3],[89],[262].

In the remainder of this thesis the location problems under consideration are the *p-median* problem in general and the *CPMP* when so stated.

2.5 Solution Techniques for Location Problems

As in the case of the VRP, current approaches for generating solutions to instances of location problems can be divided into three broad categories; exact methods, heuristics and metaheuristics. *Exact methods* are surveyed in [164] and the annotated bibliography of [222] lists numerous sources for research in exact methods. The PhD thesis of Edwards [82] and Guha [124] are devoted to *approximation algorithms* for location problems and the annotated bibliography of [222] provides additional resources. However, these two types of solution methods are outside the scope of this thesis.

2.5.1 Heuristics for p-median problems

Since heuristics form one of the main solution approaches for location problems, this section reviews the leading construction and improvement heuristics in the field. Heuristics have been applied to *p*-median problems for over four decades with varying degrees of success. One of the main factors influencing success is the degree to which the underlying network of a particular *p*-median problem satisfies the triangle inequality. The success of particular heuristics will tend to decrease for an increase in violation of the triangle inequality by the underlying network [264].

2.5.1.1 P-median problems: Construction Heuristics

The common construction heuristics for the p-median problems are the *Greedy* (add), *Drop*, and the *Composite* heuristics.

Greedy: Given a p-median problem consisting of n demand nodes and p facilities where $1 \leq p < n$, the greedy heuristic [161], [310] solves a 1-median problem on the n demand nodes (find the median that minimizes the cost to all other demand nodes). This median is then added to a solution set of facilities and removed from the list of potential facilities. The process is repeated until p medians are chosen. A theoretical and worst case analysis of this heuristic is provided by [58].

Drop: As with the greedy heuristic, we assume the problem consists of n demand points and p facilities. The drop heuristic [87] assigns all n demand points as facility locations, then iteratively removes the facility locations that result in the least increase in the total cost. The process terminates with p assigned facility locations. An efficient modification to this simple concept is provided by [254] where at each iteration only a subset of the demand nodes are considered as facility locations.

Composite: This classification identifies a group of multi-phase construction heuristics that take one of two approaches. The first approach combines more than one construction heuristic: for example, the perturbation heuristic [255] that combines the *drop* and *greedy* heuristics. In this case, the modified drop heuristic is first used to identify a set of facility locations. Then an oscillation phase is added to the greedy heuristic, where, in the search for p medians, q medians can be returned, q being an integer factor whose value is allowed to be less than, equal to or greater than that of p. These two types of infeasible solutions (more than or less than the required number of facilities) are used to filter the search.

A second approach used by composite heuristics is to combine traditional construction heuristics with improvement heuristics. For example [40] combines the greedy and alternate heuristics and [186] combine variants of the drop, greedy and alternate heuristics. Alternate and other improvement heuristics are reviewed in the section immediately following.

2.5.1.2 P-median problems: Improvement Heuristics

The common improvement heuristics for the p-median problems are the *location-allocation (alternate)*, *interchange* (*vertex substitution*) and *exchange* heuristics.

Location-allocation: A local search heuristic [179] that randomly locates p medians then allocates the demand nodes to their nearest facility. For each such cluster formed, a 1-medan problem is solved and the entire process repeated using the new *medians* as the facilities to which customers are allocated. The heuristic, also referred to as the *alternate* method, terminates when a cycle of location-allocation fails to result in an improvement in the solution cost.

Interchange: The interchange method [284], also referenced in the literature as *vertex substitution,* is a more powerful heuristic than the alternate method. Given p facilities, a subset of cardinality less than or equal to p is chosen and the members compared individually with all other demand nodes not currently designated facility locations. Any current facility that would improve the solution cost if it were located at a demand node not currently designated as such has its location changed to this node and the set of facilities is updated. A best improvement strategy is used with the heuristic terminating when no improving location substitution can be found.

Exchange: Exchange heuristics [207] are similar to the class of node exchange heuristics used for routing problems. These heuristics exchange a given number of demand points between clusters, and solve a 1-median problem for each affected cluster while doing so results in an improvement. Exchange heuristics are the main improvement heuristics used for the capacitated p-median problem [91], [207].

2.5.1.3 Summary of heuristics for P-Median Problems

All of the construction heuristics reviewed above can be applied to the CPMP, on which this thesis focuses. However, the interchange heuristic is not suited to the CPMP so well because its operation is dependent on the customers being assigned to their nearest median and the capacity constraints of the problem prevent this from being guaranteed. The alternate and exchange heuristics are therefore the preferred improvement heuristics.

2.5.2 Metaheuristics for p-median Problems

With the exception of heuristic concentration (HC) [248], the leading metaheuristics techniques for location problems (Tabu search (TS) [256], Simulated annealing (SA) [46], Genetic algorithm (GA) [6], and Ant colony optimization (AO) [169]) are the same as those used for routing problems and reviewed above. Hence, conceptual discussion of these will not be attempted in this section. Instead, we focus on how these metaheuristics perform on location problems. Mladenovic et al. [183] provides a recent survey.

2.5.2.1 TS, SA, GA and AO

Arostegui et al. [10] conducted an empirical study on the performance of tabu search, genetic algorithm and simulated annealing for location problems. The conclusions were that TS algorithms showed 'good' performance for all types of facility location problems and while they also appear to perform somewhat better than GA and

SA for increased solution runtime, the parameter setting used for the heuristics could impact those findings. However, they concluded that for location problems TS is the preferred metaheuristic of the three.

Competitive AO metaheuristics have also been reported for location problems [157]. In [169], Levanova and Loresh report being able to solve all problems encoded in a set of standard location problems used in the literature.

2.5.2.2 Heuristic Concentration and other approaches

HC is a two phase metaheuristic that has been used to produce results comparable to TS for a class of location problem [249],[244]. The metaheuristic constructs a set of solutions using a standard construction heuristic such as the *interchange* method or the *drop* heuristic. A subset of the best solutions found are then stored. A *concentration set* is then constructed, representing the facility locations most frequently found in the set of best solutions. This completes the first phase.

The second phase considers facility locations stored in the concentration set only and solves the p-median problem, selecting p facilities from this set to return the best solution possible. One of the weaknesses of the *interchange* method is that it can become trapped in local optima of the same set of facilities location. A HC implementation using the *Interchange* method, because it constructs a set of solutions of different facilities locations, greatly reduces the chances of this happening and in some cases Rosing and Hodgson report being able to find optimal solution values [250].

Other often-cited metaheuristics for location problems include variable neighborhood search [131] and scatter search [101]. Hybrid metaheuristics have also being proposed for location problems [229].

2.5.3 Solution Techniques for the Capacitated *p*-median Problem (CPMP)

Mulver and Beck [188] proposed one of the early heuristics for the CPMP. The heuristic generates a random set of p medians and assigns customers to their nearest median, in order of decreasing *regret* value, while respecting the capacity constraint. A customer's regret value is the difference in cost of assigning the customer to its nearest and second nearest facility. The median of each cluster is calculated at the end of the assignment phase. Where new medians emerge, the assignment phase is repeated, with the entire process continuing until the medians remain unchanged. The heuristic then executes a local search procedure that exchanges customers between clusters whenever this reduces the solution cost.

A similar approach is used in the construction phase of the heuristic proposed by Osman and Christofides [207]. However, instead of choosing the initial medians randomly, initial median sites are selected as the two customers furthest apart. If p is greater than two, then additional medians are chosen until p is satisfied. The new medians are selected such that each selected median maximizes the product of distances between itself and all previously located medians.

The customers are then assigned to their nearest available median in increasing order of distance, while the capacity constraints can be respected. As with the Mulver and Beck heuristic, if the assignment phase produces new medians, the entire process is repeated until a stable set of medians emerges.

A hybrid simulated annealing and tabu search metaheuristic is used to improve the solution. For a set of randomly generated problems [207], in best cases the heuristic returns solutions on average 0.04% above the best known solution [207] .

One weakness of the construction phase of the heuristics outlined above is that they do not guarantee a feasible initial solution. This is likely to arise if the capacity constraints are tight [91] i.e. if the capacity constraints are such that p times the problem stated capacity is approximately equal to the total demand of nodes in the problem. If the nodes are assigned purely based on their distances from the medians, and assuming the node demands are not all equal, it is possible for an overflow to occur involving a node whose demand is larger than the available capacity of any individual median.

The construction phase of the adaptive tabu search metaheuristic proposed by Franca et al.[91], attempts to minimise the possibility of infeasible solutions by assigning customers in increasing order of a quotient relative to available facilities. For each facility, the quotient is given by the distance between the customer and the facility, divided by the demand of the customer. The remainder of the construction phase proceeds in a manner comparable with Osman and Christofides [207]. An adaptive tabu search algorithm then improves the initial solution. For the test instances used in [207], in best cases the heuristic reported results 0.004 % above the best known solutions.

2.5.3.1 Agglomerative Algorithms Clustering and Errors

Multilevel algorithms have being used in graph partitioning [152], which has a number of similarities with clustering [153] (where the multilevel technique has also being applied). As outlined above, the concepts of clustering apply to the p-median problem where the multilevel algorithm is being further extended. However, the coarsening phase tends to be an aggregate process and the issue of errors introduced when agglomerative algorithms are used for clustering must be noted [183].

Most location models assume that demands occurs at the nodes of the network or at specific points in the plane. In some cases, however, this might not be realistic as the point of demand may represent aggregated demands [65],[93]. Aggregation potentially introduces errors due to the use of approximate distances instead of actual distances, the location of aggregate demand nodes at locations occupied by unaggregated demand nodes [138], allocation errors [216], and cost, optimality, and location errors [86]. While solutions have been proposed for these types of errors [64] [141], Mladenvic et al. [183] raised the interesting issue of how effective aggregating the demand nodes for location problems was, when the main source of complexity was the number of facilities to be located.

Aggregation errors are analysed and surveyed by [92] and [94]. With the most recent survey [95] highlighting the benefits of aggregation including decreased data collection cost and increased data confidentiality. Work continues in addressing the issue posed by aggregation as evidenced by the recent works of Plastria and Vanhaverbeke [217].

2.6 Review conclusion

Location and routing are two areas of research that are of importance to industry and academia. That fact that the number of possible solutions to these problems typically increase exponentially for linear increases in the sizes of the problems, means exact solution techniques are limited to small-scale instances. With the desire to find ever-improved solutions and the need to solve larger problems, researchers devote significant amounts of effort to heuristic approaches and, in particular, metaheuristic approaches. These heuristics and metaheuristics are currently the leading solution techniques for routing and location problems.

In the next chapter, it is shown that the multilevel technique can aid the solution process for location and routing problems, for both heuristic and metaheuristic approaches. The multilevel technique had not been applied to routing and location problems before this research.

Chapter III

3 The Multilevel Technique

This chapter provides a review of the multilevel technique, identifying some areas where the technique has been successfully applied. Throughout the chapter, it is shown that the multilevel technique is suitable for a wide array of problem areas. Additionally, potential problem areas where its adaptation might be less successful are highlighted.

The chapter is presented in two parts. *Part I* distils from the existing research some key characteristics of the multilevel technique. The research community has successfully applied multilevel-type techniques across a wide range of problem domains [285]. This includes work on: force-directed graph drawing [133], [303]; multigrid [26]; multi-scale and muti-resolution methods [26], [285]. The combination of aggregation and disaggregation techniques also has similarities to multilevel modeling [243]. Hence, the first part of the chapter draws on these and other areas to present a general review of the technique. *Part II* presents a series of informative case studies from the literature, exemplifying areas where the technique has been applied.

3.1 *Part I* – Concepts of Multilevel Refinement

Multilevel refinement is a collaborative technique, which guides the performance of other solution techniques. As the technique is instantiated for different problems, the multilevel algorithm describing the technique can vary between instances. However, there is an underlying philosophy to multilevel refinement, governing how the multilevel technique operates across varying problem types and implementations. This part of the chapter describes the underlying philosophy along with its aims and methodologies.

3.1.1 Review of the Multilevel Technique

The multilevel technique is a simple one, which uses recursive coarsening to create a hierarchy of approximations to a given problem. In many cases, since the problem is coarsened to the maximum point allowed by the problem constraints, the coarsest approximation can then be used as an initial solution, which is repeatedly extended (coarsest to finest) and iteratively refined, generating a final solution [304]. There is a problem specific element to how a particular multilevel algorithm implements this simple paradigm. However, the technique can be discussed in generic terms and that is the purpose of this section.

The multilevel technique encompasses two main phases *coarsening*, and refinement. Given a problem p, the coarsening phase constructs a sequence of approximations p_0 , p_1, ... p_n; where p_x is a coarser approximation than p_{x-1} but more detailed than p_{x+1} and $0\leq x \leq n$ and p_0 is the original problem.

For each approximation created in the coarsening phase, there is a corresponding solution in the refinement phase. There usually exist a close relationship between solution s_x for approximation p_x, and solution s_{x-1} for p_{x-1}. There also exist a relationship between s_x and s_{x+1}, the solution for approximation p_{x+1}. Generally, if the solutions were generated independently of each other, they could be ordered by their increasing quality and difficulty to arrive at as: s_{x+1}, s_x and s_{x-1}. In the multilevel technique, however, solutions are not generated independently.

Assuming the final approximation created was p_n, the refinement phase starts with an initial solution s_n. The refinement phase then iteratively applies first an extension process to s_n, to unmask sections of the solution approximated when approximation p_n was generated. This intermediate solution is then refined to create solution s_{n-1}. The refinement process is then repeated, for each succeeding solution to s_{n-1}, until solution s_0 is obtained. S_0 being a solution formed on the original problem. Indeed, by being an output from the preceding level, each succeeding solution usually preserves the 'good decisions' made at the previous level and this therefore makes solving the more detailed problems easier, than solving them independently.

The generic multilevel algorithm is shown in Figure 3. The first while loop contains the coarsening phase. Each time through this loop corresponds to a new approximation, or a new *level.* The second while loop of the algorithm, encapsulates the refinement phase. Each time through this loop, the algorithm revisits the levels created in the coarsening phase in reverse order. The *refine* method can constitute the improvement phase of an iterative improvement algorithm(s) or metaheuristic(s) approaches. The differentiating feature of the multilevel technique however, is that the *refine* method is repeated for each approximation created during the coarsening phase.

Figure 3 A generic multilevel algorithm

```
set level counter i := 0
set problem = Pi
while (Pi can be coarsened)
  Pi+1 = coarsen(Pi)
  i := i + 1
end
Set initial solution Si = Pi
while ( i ≥ 0)
  i := i - 1
  Stemp = extend(Si+1)
  Si = refine (Stemp)
end
```

3.1.1.1 Multilevel Technique Methodologies

The multilevel technique aims to aid the solution process of optimisation problems by improving the convergence rate of its underlying local search heuristic(s) and or improving the asymptotic convergence in the quality of solutions produced by these heuristic(s) [304]. To these aims, the central methodologies of the multilevel technique are:

- *Filtering solutions from the search space.*
- *Reducing the amount of problem detail to be considered at each level of the solution process.*
- *Providing a mechanism to the underlying local search heuristic(s), via coarsened problems, for efficiently making large moves around the search space.* The neighbourhoods accessible by these moves would typically be inaccessible if the local search heuristic(s) were applied to un-coarsened problems.

The methodologies coalesce to meet the multilevel technique's aims, because, as the multilevel algorithm iteratively coarsens, extends and refines a given problem, it:

- *Provides a more global view to the local search heuristic(s) than that accessible to the local search heuristic(s) acting alone* [304]. This potentially reduces the possibility of the local search heuristic(s) getting trapped in local optima of poor quality.
- *Is able to improve the quality of the solution while the solution is in a coarsened state to the point where, at the start of applying the technique to the un-coarsened solution, a high quality solution is in place.*

  This typically allows the local search heuristic(s) to refine the un-coarsened solution faster, compared to when the local search heuristic(s) is applied to the un-coarsened problem starting from a poor initial solution. This is because in refining a solution of high quality the local search heuristic(s) will typically become caught in a local optimum of good quality. Secondly, it is usual for the local search heuristic(s) to refine a solution until no improvement in the solution cost can be found. In the case where a high quality solution is in place, this stopping condition tends to be achieved faster than when starting from a solution of poor quality.

3.1.2 Coarsening

Of the two phases of multilevel refinement identified (coarsening and refinement), the coarsening phase possibly plays the key role in determining if the methodologies of the multilevel technique will be successful for a given problem. Determining, how to coarsen a specific problem or if suitable coarsening algorithms can be devised, is largely a problem specific task. However, there are general issues common to coarsening algorithms and a review of these issues is the focus of this section.

It seems that two key questions determine whether a particular coarsening approach for a given problem will lead to success. The first, asks whether the coarsening process approximate the problem in an *exact* or *inexact* manner [305]? Secondly, can a representation be found for the coarsened problem that allows the solution space to be efficiently explored?

Exact coarsening means that for a solution formed on any approximation created during the coarsening phase, evaluating the objective function on that solution before and after the extension process is applied, transforming the solution to a state containing no approximations, would return the same results. Exact coarsening appears to outperform inexact coarsening [305]. Although this is outside the scope of the research here, the following can be theorised. While exact coarsening means feasible solutions can be generated throughout the refinement process, inexact coarsening means that in the upper levels, infeasible solutions are generated and feasibility is not guaranteed until the un-coarsened problem is refined. Hence, in transferring from the inexact solution to an exact solution, the quality of the solution can potentially deteriorate. Secondly, in solving an inexact representation of the problem, it is difficulty to guarantee that the improvements implemented are valid on the actual problem.

When the refinement algorithms are applied to a solution, in the upper levels of refinement, the solution is in a coarsened state. How efficient the refinement algorithms are in finding improvements in the solution is to some extent dependent on the representation of the coarsened solution. Therefore, the coarsening algorithms have to address the issue of how best to represent the coarsened problems such that the refinement algorithms, to be applied in the refinement phase, will find it possible to efficiently improve the quality of the solution.

3.1.3 Refinement

The refinement phase of the multilevel technique seeks to improve the quality of the solution created at the end of the coarsening phase.

The refinement algorithms deployed during the refinement phase, are typically problem specific heuristic(s) or, more general, metaheuristic approaches. From the literature, it can be seen that refinement algorithms have been implemented, running the gamut from tabu search, simulated annealing, genetic algorithms, cooperative search, ant colonies optimisation, to various problem specific approaches [304],[305]. Regardless of the approach taken for a given problem, it is necessary to customise the algorithms to refine the solution in a coarsened state in the upper levels of refinement. Typically, this requires the algorithms to respect the sections of the search space demarcated not for refinement, by the coarsening algorithms, at particular levels.

3.1.4 Multilevel Enhancements

There exist a number of generic enhancements for the multilevel technique. These enhancements can be incorporated into the generic multilevel algorithm to improve performance [305].

In problems where the vertices are weighted, if a large proportion of the total graph weight is concentrated at a few vertices, standard coarsening produces inhomogeneous graphs [305]. **Coarsening homogeneity** is the process of ensuring that the coarsened graphs remain relatively homogeneous [305]. This is achieved either by allowing the matching of more than two vertices at a level, typically these will be the vertices of least weight or by rejecting matches of heavily weighted vertices. Coarsening homogeneity is also applicable in the cases where the edges are weighted, e.g. routing problems [240].

Constraint relaxation is an enhancement that allows the gradual slackening of the constraints at each level of the coarsening process, allowing the construction of higher quality solutions with respect to the objective function (these solution typically will be infeasible). The refinement phase gradually strengthens the constraints with the aim of maintaining the advantages found in the relaxation process. Constraint relaxation is applicable to both vertex-weighted [301] and edge-weighted [240] problems.

Solution-based recoarsening [305], one of the more powerful enhancements, allows for the coarsening of a solution to a given problem. Restrictions are placed on the coarsening process, ensuring that the desirable features of the solution are still present after it is re-coarsened. In the case of a routing solution, coarsening is applied to vertices belonging to the same route. In the case of a k-way partitioning solution, vertices are coarsened if they are part of the same set. The refinement algorithm in place then treats the re-coarsened solution as an initial solution and searches for further improvements. Solution-based recoarsening forms the basis of *iterated multilevel algorithms* [304].

3.2 *Part II* – Multilevel Implementations from the Literature

One aim of this chapter is to identify for the reader those problem areas where the multilevel technique may be successfully applied and those where its application might be less successful. This part of the chapter contributes to that aim, by presenting a series of varied problems where the multilevel technique has been successfully applied and an area where the application has been less successful.

This presentation is by no means exhaustive; however, the recent review by Walshaw [305] provides an overview of numerous other such instances. These include the application of the multilevel technique to: covering design [67], biomedical feature selection [197] and capacitated multicommodity network design [62].

It is often common to discuss multilevel implementations in terms of graphs, so to aid the discussions, the following definitions are provided.

Consider a graph $G = (V, E)$ where V is a set of vertices and E represents a set of edges. A pair of vertices in V is *independent* if they do not share an edge in E. We refer to a collection of independent vertices as an *independent set.* This set is also *maximal* if no new member can be added while it remains independent. A *cycle* of G, is a subset of E forming a path, where the first and last edge of the path incident the same vertex in V. A *Hamiltonian cycle* results if all members of V are a part of a cycle.

3.2.1 Multilevel Technique applied to the Graph Partitioning Problem

The graph partitioning problem (GPP) [102] was the first CO problem for which a multilevel algorithm was developed [305]. Indeed it was its successful application to the GPP that justified employing it as a metaheuristic approach [304].

The graph partitioning problem can be stated as follows. For a given weighted graph, partition the vertices into k disjoint subsets of given sizes (usually equal), while minimizing the sum of the weights of the edges between the subsets. The GPP is known to be NP-hard [102].

Through a number of contributions [12],[37],[136], multilevel implementations for the GPP evolved to provide a 'global perspective' for the leading local search algorithms [151], effectively overcoming their inherent "localized nature" [304]. The multilevel technique typically coarsens the given graph to a desired threshold by *matching* and *contracting* adjacent vertices. Starting with the coarsest representation, for which a feasible solution is found initially, the solution is then iteratively refined and extended to the level below, terminating with the refinement of the solution on the original graph.

Edge contraction [136] is the standard algorithm for coarsening graph partitioning problems. The edge contraction algorithm can be stated as: Given an edge in graph G, $e = (v_1, v_2)$; contract e by removing it from the graph and replacing v_1 and v_2 with a new vertex. Connect all edges, except e, previously incident on v_1 and v_2 on the new vertex.

Using edge contraction, the coarsening process for the GPP iteratively constructs a maximal independent set for a given representation of G and, where possible, collapses all the edges in G incident on vertices in this set, creating a new approximation. Any edges in the previous representation not collapsed are projected to the new approximation and the process is repeated until an approximation of G is obtained containing k vertices. This type of coarsening is known as *set-based* coarsening [305].

The refinement phase seeks to improve the solution quality at each level. Solution techniques ranging from the Kernighan & Lin algorithm [154], simulated annealing [245] and evolutionary algorithms [274] through to tabu search [15] have been implement as refinement schemes for GPP multilevel algorithms. Walshaw reports that these "*have generally been applied with great success*" [305].

3.2.2 Multilevel Technique applied to the Traveling Salesman Problem

The traveling salesman problem (TSP) [266] can be stated as follows. Given a complete graph with weighted edges, find the least weighted Hamiltonian cycle. This NP-hard problem [102] is possibly the most studied CO problem [304], however it has been "dominated by the Lin-Kernighan heuristics" [304]. Walshaw developed a multilevel algorithm for the TSP [42] that was "*shown to enhance considerably the quality of tours*" [305] and had superior performance to its single-level counterpart.

The coarsening phase of the TSP multilevel algorithm iteratively *fixes edges* between (unmatched nearest neighbour) pairs of vertices. This creates a hierarchy of coarser problems each of which has a feasible solution. A path of fixed edges reduces to a single edge and two vertices. The coarsening process terminates when one fixed edge and a pair of vertices represent the entire problem. This type of coarsening is known as *path-based* coarsening [305].

For a TSP problem p, fixing an edge creates approximation p_x. This edge will be contained in solution s_x formed on p_x and in all solutions formed on approximations created from p_x. However, during the refinement phase, solutions preceding s_x, can analyze that edge and other edges fixed at levels above, with regard to the objective function and may reverse the decision of having these edges as a part of a final solution.

The aim therefore is to fix edges in the coarsening process that will be part of a tour of high quality. Where this has not been achieved, the refinement phase applies edges exchange heuristics in an attempt to replace these with edges of less weight.

Another approach to coarsening the TSP is presented by Bouhmala [32] that merges nodes to from a new node at their average location. The final tour is then formed on the smaller approximated problem and refined. While the coarsening procedure produced infeasible solutions in the upper levels (i.e. inexact coarsening), the multilevel technique was still able to outperform comparative single-level versions.

3.2.3 Multilevel Technique applied to Protein- Protein networks

In biological systems, groups of interacting proteins known as protein complexes [189], [275] execute processes at a cellular level. If the proteins in a biological system are modeled as the nodes of a graph and the interaction between proteins as the graph's edges, this results in a so called "*protein-protein interaction (PPI) network*" [200]. The task is then to identify the sets of protein complexes on the network, and these correspond to the most connected areas of the graph. This task typically results in a search for cliques. The PPI network is an unweighted graph as the edges correspond to the interactions between the nodes and the nodes themselves are not weighted.

Oliveria and Seok [201], [202] applied the multilevel technique to a PPI network. They compare their multilevel implementation to a single-level version using a min-max-cut graph clustering algorithm [74]. The multilevel algorithm outperformed the single-level version on solution quality by an average of 10% and proved superior on runtimes. The refinement phase of their algorithm executes a Kernighan-Lin type algorithm [154] at each level.

The multilevel implementation of Oliveria and Seok [200] address two general concepts relevant to multilevel practitioners. The first concept identifies one method for successfully coarsening unweighted graphs, containing cliques. The approach taken was to find cliques of three nodes and coarsen all three into one super node. This procedure replaces three edges and two nodes in the graph. Edges that are incident between the nodes of the clique and nodes external to the cliques, become incident to the super node and the external nodes. Coarsening algorithms typically cluster nodes that are related and normally this is done based on the strength of those relations. However, in the case of the PPI network, because individual edges state that two nodes are related but not how strong that relation is, cliques provide a good method of merging connected nodes [275].

Finding maximal cliques in graphs is an NP-Complete problem [271] while finding cliques of 3 nodes is of an order of $O(|E|^2/|N|)$ [200] where E is the number of edges in the graph and N the number of nodes. For these reasons, Oliveria & Seok's coarsening algorithm merged cliques consisting of three nodes. If a pair of cliques, each of three nodes, share two nodes, both cliques are merged into a super node. If however, only one node is shared between the pair, one of the cliques is chosen arbitrarily and merged, claiming ownership of the shared node. The graph is iteratively coarsened using this process. The researchers concluded that coarsening using cliques of three nodes gave significant advantages over coarsening algorithms merging at most two nodes at once.

Where a feasible initial solution is required, some problems naturally provide a stopping condition for the coarsening process. For example, the p-median problem where the requirement is to identify p medians to serve p clusters, in this case one obvious stopping condition is coarsening the problem until it has been divided into exactly p clusters. With the problem of finding the number of protein complexes on a *PPI* network, there is no obvious stopping condition for the coarsening process. However the implementation of Oliveria and Seok demonstrates that for *PPI* (and similar) networks it is possible for the multilevel algorithm to obtain superior results to the single-level algorithm with one or two levels of coarsening. Hence a practitioner designing a multilevel algorithm can take the approach of using 'some coarsening' as opposed to attempting to coarsen the problem to some 'natural stopping condition' if this is not easily defined.

3.2.4 Multilevel Technique applied to Graph Colouring

The graph coloring problem (*GCP*) [304] is a NP-hard problem [103] with applications in numerous areas including scheduling [167] and computer register allocation [47]. The problem can be stated as: given a graph, color the vertices of graph using the minimum number of colors while ensuring no two adjacent vertices have the same color. Walshaw implemented a multilevel algorithm for the GCP that "*provided some asymptotic convergence of well known algorithms but ... was less impressive than ... when applied to other CO problems*" [304].

Synopsis of the multilevel implementation for the GCP: The coarsening algorithm produces a graph G_{x+1} from G_x by coarsening pairs of vertices in G_x, each matched pair becoming a new vertex in G_{x+1}. Non-adjacent vertices are matched; however this is restricted such that vertices are only allowed to be matched with the neighbour of an immediate neighbour. A tabu search algorithm using an iterated greedy algorithm was used in the refinement phase [304].

Possibly the most relevant lesson for the multilevel practitioner is why the multilevel technique appears to fail for the GCP. Being a collaborative technique, the multilevel algorithm's success or failure largely depends on its ability to aid the asymptotic convergence in the quality of its embedded local search algorithms. Hence, the likelihood of success for multilevel algorithms increases for problems where the objective function is such that executing an improving local change in the solution is reflected in an improved global solution. An example of this is an improving two-opt move for a TSP solution reducing the overall cost of the tour. Where this relationship is found between the objective function and local changes, the multilevel technique has been seen to greatly improve the performance of local search algorithms [304].

The objective function of the GCP however does not appear to exhibit the property of reflecting global changes in response to individual local changes. In other words, as the aim is to minimise the set of colors assigned to the vertices of the graph while meeting the constraint that no two vertices have the same color, changing the colors on two adjacent vertices is not guaranteed to propagate into a reduction of the set of colors. Walshaw provides a detailed analysis of this point [304].

3.3 Conclusions

In designing a multilevel implementation for a given problem, one of the main challenges is in devising a coarsening algorithm. The coarsening algorithm must be capable of constructing approximations that can be efficiently optimised in the refinement phase. This invariably centres on filtering solutions from the search space and reducing the amount of problem detail to be considered in the refinement phase. In various multilevel implementations the approach taken to filtering details from the solution, is embedded in the process of filtering solutions from the search space. This typically results in restricting sections of the problem from considerations at particular levels, as evidenced by the work done on the PPI and GPP. A multilevel implementation for biomedical feature selection Oduntan et al. [198], utilised another approach to filtering details from the solution, that of excluding decision variables from each level.

Against this backdrop and the fact that multilevel algorithms have been successfully employed on weighted and unweighted graphs it can be seen that the multilevel paradigm is potentially suited to a wide array of problems.

Chapter IV

4 Multilevel Technique for Routing and Location Problems

This chapter discusses how multilevel and single-level techniques have been implemented for the capacity vehicle routing problem (CVRP) and the capacitated *p*-median problem (CPMP).

There exist some similarities between both problems, chiefly, the requirement to partition the set of customers into feasible subsets. The partitioning should be achieved while respecting the problem constraints and minimising connection costs.

The method of creation of these subsets, *routes* in the case of the CVRP and *clusters* in the case of the CPMP, are similar for both types of problems. In creating the subsets, groups of customers can be recursively created from single customers or other groups of customers. These groups of customers are representable as a single entity, and can be treated as single customers in generating solutions to the problems. Given a suitable mechanism for recursively producing these groups of customers (segments[1]), the multilevel technique is then able to create a series of coarser approximations to a given problem and refine solutions for them in a standard manner (i.e. using standard heuristics or metaheuristics). Most of the refinement done on solutions to these problems can be viewed as a transferring of segments between subsets with the aim of minimising the total cost.

A multilevel framework was developed for the CVRP, capable of generating high quality solutions for the problem. This framework was then extended to the CPMP

[1] The groups of customers are referred to as segments for both problems. We are aware of the imperfections of this terminology, especially in the case of the CPMP. However, segment(s) is chosen for both problems as is allows a uniformed discussion of the recursive nature of the coarsening process and of the process of inter route and inter cluster refinement.

where it has performed equally well. This chapter presented in five parts, describes the coarsening and refinement heuristics implemented, and enhancements to the multilevel technique utilised for both problems.

Part I presents a synopsis of the work done in applying the technique to the CVRP and the CPMP. *Part II* discusses how the algorithms were applied in the multilevel framework for the CVRP. *Part III* discusses how these algorithms were used in the single-level heuristic for the CVRP. *Part IV* provides a description of the CPMP algorithms and their application in the multilevel framework, while *Part V* provides a discussion of these CPMP algorithms in the single-level format.

4.1 *Part I* – Multilevel Technique and its application in this Research.

The multilevel technique is applied to two problems in this research, the capacity vehicle routing problem (CVRP) and the capacitated p-median problem (CPMP). In this, the first part of the chapter, a description of the methodology used in applying the technique to these problems is presented. This is followed by a synopsis of the application of the multilevel technique to each problem. The synopsis is intended to provide the reader with an overview of the entire solution process, abstracting the details of the algorithms, which are presented in the remainder of chapter.

4.1.1 Research Methodology

This sub-section describes the methodology of applying multilevel refinement to the CVRP and the CPMP.

Leading construction and local search heuristics were used to construct composite heuristics[1] [226] capable of yielding solutions for the CVRP and the CPMP. Multilevel algorithms of these composite heuristics were then implemented. Since a main aim of the research was to investigate if multilevel refinement could aid the solution process for routing and location problems, the construction of multilevel algorithms for these composite heuristics, then facilitated these investigations.

Of the leading heuristics in the field, the ones chosen to construct the composite heuristics were those that were easy to implement, widely used, capable of being adapted to the multilevel technique and generally displayed the characteristics of effective heuristics (see section 2.3.1.1).

The research employed preliminary investigations to determine the heuristics' ability to adapt to the multilevel technique. Of the heuristics rejected, one class of notable rejection was the class of petal heuristics [225]. Our investigations indicated multilevel algorithms using petal heuristics were not particularly effective, as the reliance of petal heuristics on polar coordinates, was particularly at odds with the mechanisms of coarsening. The heuristics chosen are presented throughout parts *II* to *IV* of this chapter.

[1] A composite heuristic is a heuristic consisting of a construction phase that deploys construction heuristic(s) to create an initial solution. This is followed by an improvement phase, deploying local search heuristic(s) to improve the initial solution.

4.1.2 Multilevel Technique applied to the CVRP - a case study

The application of the multilevel technique to the CVRP is analysed against the backdrop of a single instance, namely number 1 of the Christofides et al. test suite [49]. This problem instance is a standard CVRP consisting of 50 customers and has an optimal cost of 524.61 [55],[125]. The cost of 524.61 represents the summation of the distance travelled on all routes in the solution to and from the depot, servicing all 50 customers. Customer locations are represented as (x, y) coordinates and the distance between customers corresponds to the Euclidean distance represented in double precision real numbers. Figure 4 shows the result of the multilevel algorithm applied to this problem instance. The top half of the figure shows the coarsening process, viewed from left to right. The refinement process is shown in the bottom half of the figure, viewing the figure from right to left.

A new coarsened graph is created by fixing edges between selected nodes and representing them as a single node (these single nodes are termed segments - see section 4.2.1). A segment's demand reflects the combined demands of the selected nodes. This process is repeated while there are nodes that can be joined with the resulting segments' demands respecting the capacity constraints allowed for a route. This is one obvious stopping condition; however, the coarsening process can be terminated before or after this point. If the coarsening process is terminated as shown in Figure 4, without steps taken to manipulate the capacity constraints it will be impossible to implement improving moves in the uppermost level. Where steps are taken to manipulate the capacity constraints it is possible to extend the coarsening process beyond this point (see section 4.2.4.1).

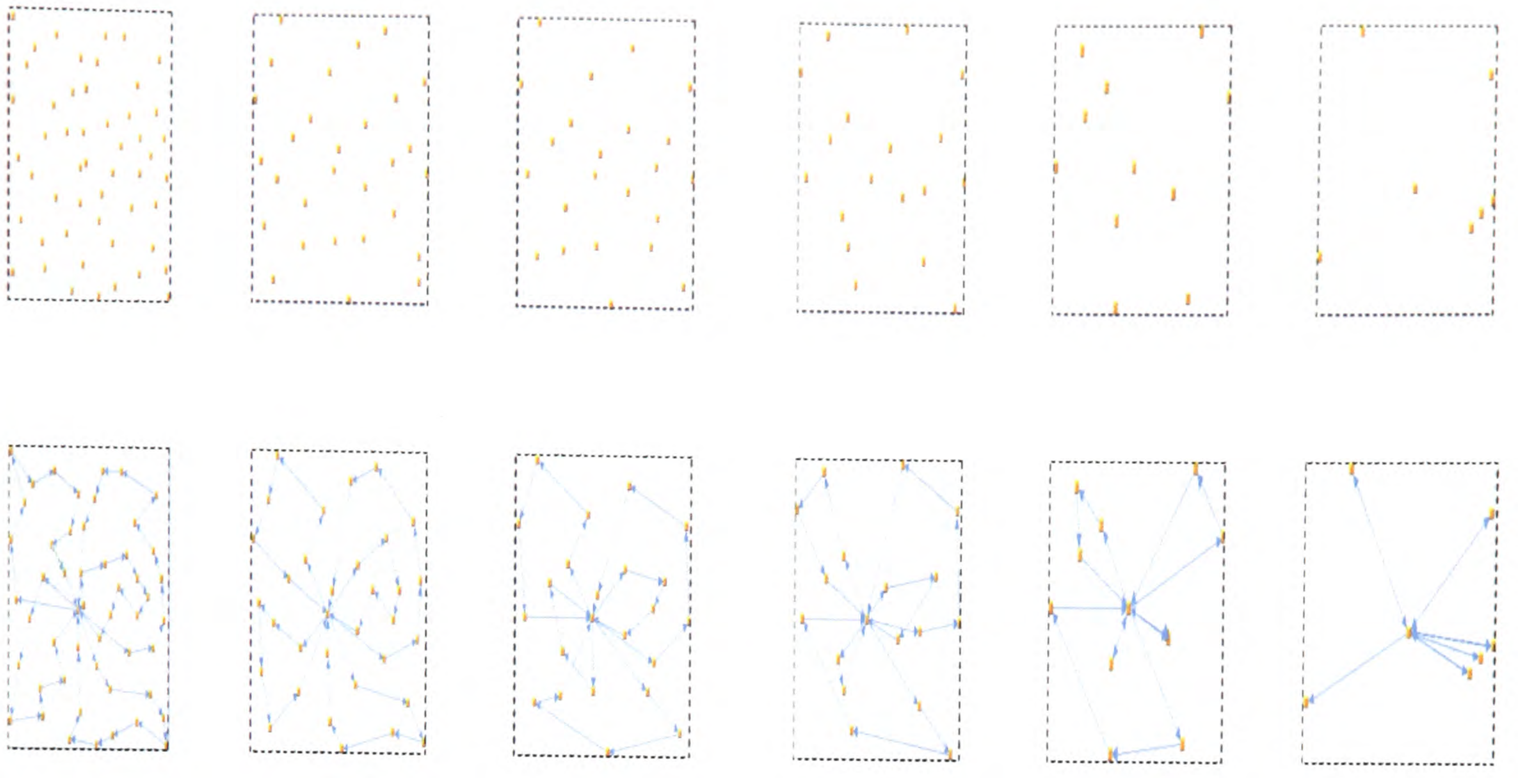

Figure 4 Multilevel refinement of instance 1 of the Christofides et al. instances.

When individual nodes are merged to create a new segment, the new segment's location is represented by their combined sets of (x, y) coordinates as shown in section *B* of Figure 5. As the coarsening process is iteratively applied, it has the effect of filtering some solutions from the search space [304]. These solutions are filtered from the set of solutions formed on graphs in the succeeding levels, and correspond to those solutions that do not include routes serving the merged nodes consecutively. To see this we assume that, in transforming the graph G_x to the coarser representation G_{x+1}, nodes 5 and 6 are coarsened (see Figure 5), creating node 7. This then guarantees that there will be a route formed on G_{x+1} (and all succeeding graphs) that will serve both nodes 5 and 6. Hence, all solutions not including a route serving node 5 followed by 6 are filtered from the search space at these levels.

In addition to filtering solutions from the search space, coarsening also reduces the level of detail in the problem. Section *D* of Figure 5 shows that in serving node 7 the position information available is (x_2, y_2) and (x_4, y_4) corresponding to the location information for nodes 2 and 4. Hence, while nodes 1 and 3 are guaranteed to be served by the same route on all solutions formed on graph G_{x+1} and succeeding graphs, their location information has been filtered from the problem. The effect of filtering detail from the problem is amplified as the problem is further coarsened. A key decision therefore, is determining a suitable mechanism to filter poor solutions from the search space. Since the edges of the CVRP graph are weighted, these weights are used in selecting nodes to be merged. At its simplest, the selection process merges nodes connected by the least weighted edges (see section 4.2.2).

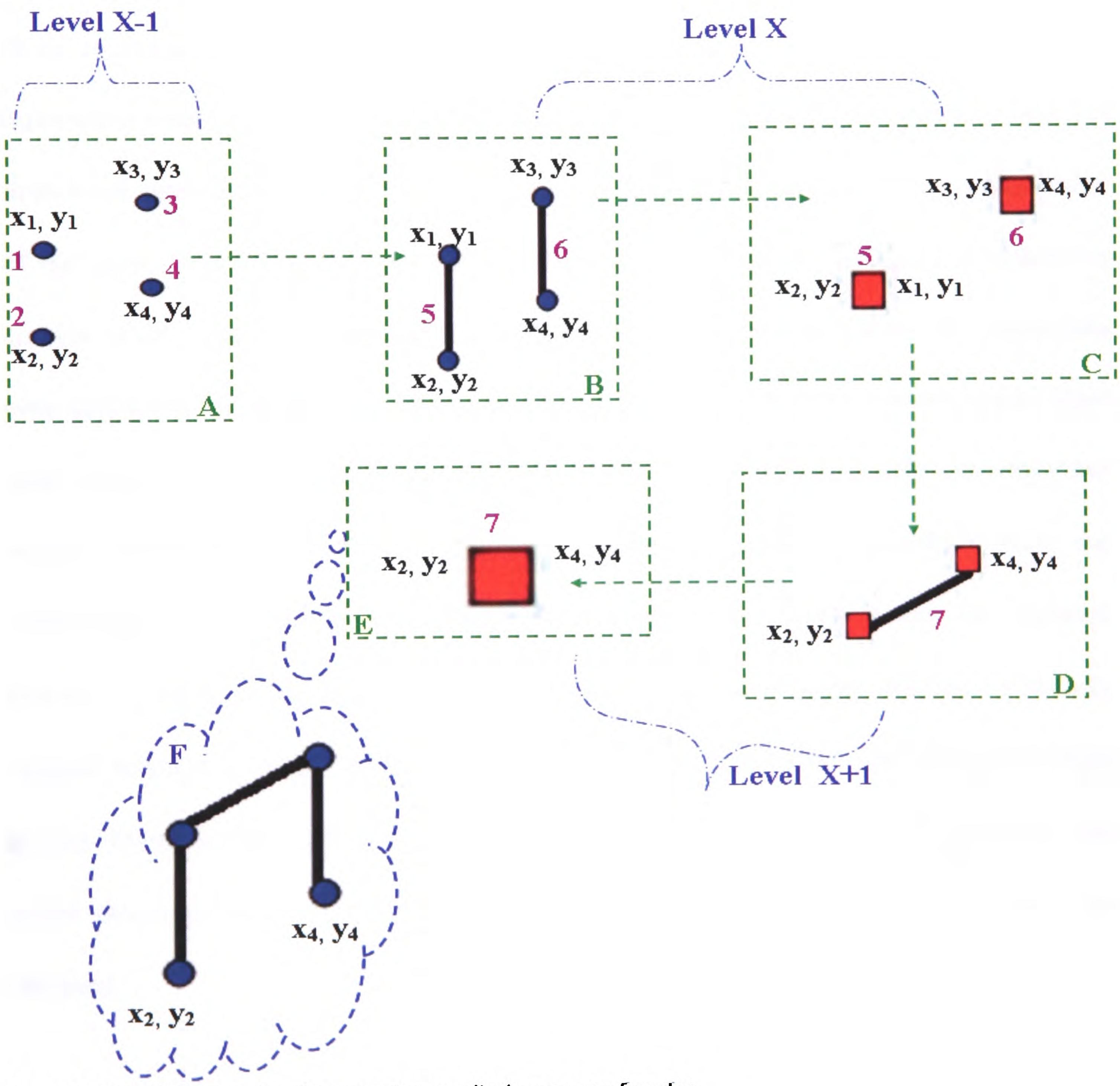

Figure 5 A CPMP example of coarsening applied a group of nodes.

Once the refinement phase frees the fixed edges linking a set of nodes, the refinement algorithm is then allowed to serve the constituting nodes in the manner that best optimises the solution cost. In the case of Figure 5, this means that when the refinement phase revisits the solution formed on graph G_{x+1} and reverses the merge, resulting in node 7, it has the option to serve nodes 5 and 6 on different routes. If they are served on the same route, there is no requirement on the refinement algorithm to serve them consecutively.

Lin-Kernighan type algorithms (sections 4.2.3) are then used in the refinement phase to improve the solution cost. Figure 6 shows the change in solution cost as the refinement phase proceeds through the levels. This is the typical performance sought for multilevel algorithms applied to this type of problem (not always obtained), where most of the improvements in the solution cost is found in the computationally inexpensive coarser levels. As the refinement algorithm uncoarsens the problem, the algorithm potentially has the option of visiting the solutions filtered from the solution space. How many of these solutions can be visited, however, is limited by the neighbourhood of the search space accessible to the algorithm. Therefore, what is desirable from the coarsening process and the refinement phase in the upper levels is that the solution process is taken to an area in the search space where a globally optimal or locally optimal solution of very high quality is located. If this is achieved, the solution of high quality found in the upper levels is propagated to the levels below. If however, the initial decisions are very poor it can be difficult for a high quality solution to be obtained.

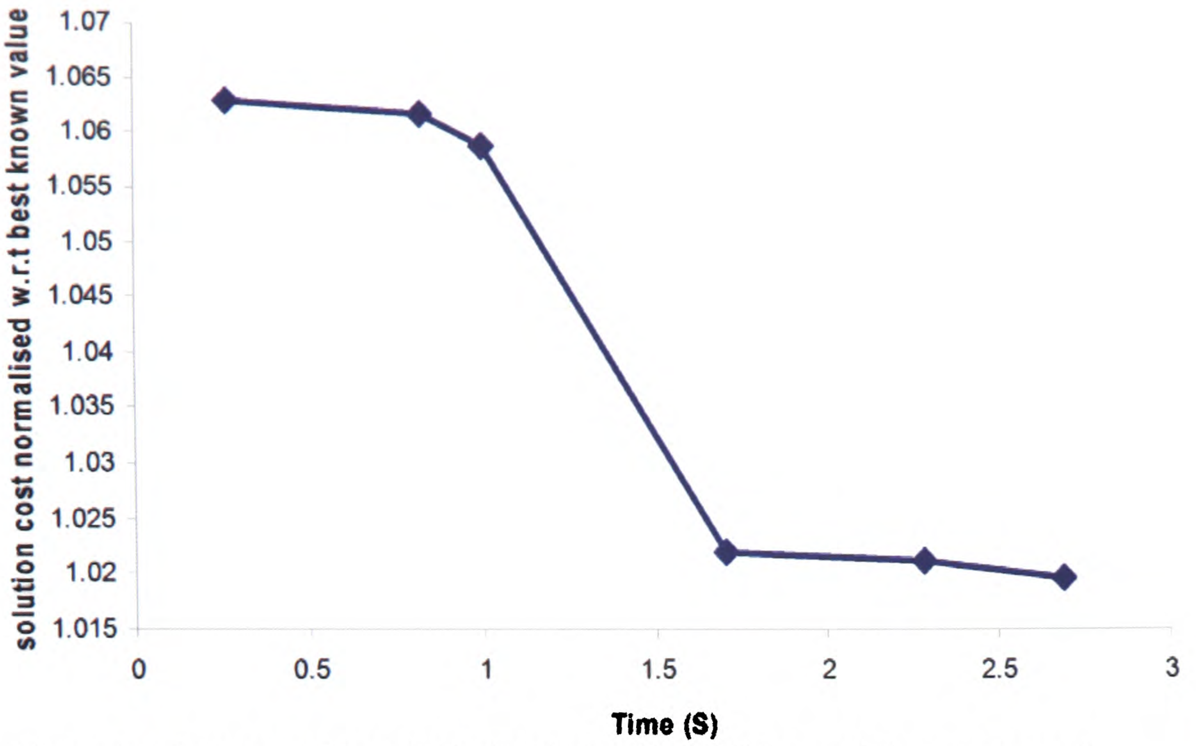

Figure 6 Refinement through the levels for Christofides et al. instances no. 1.

While the multilevel algorithm produced a good solution (for the problem instance under review) the cost is approximately 2% above the optimal value. To improve the solution quality further, an iterated multilevel (It.ML) algorithm was implemented (see section 4.2.4.3). The It.ML algorithm takes as input, the solution produced by the multilevel algorithm and then coarsens and refines this solution for a given number of iterations. It has been found that It.ML algorithms can further improve the asymptotic convergence of multilevel algorithms [305]. Table 1 and Figure 7 show the improvements it produced for this problem.

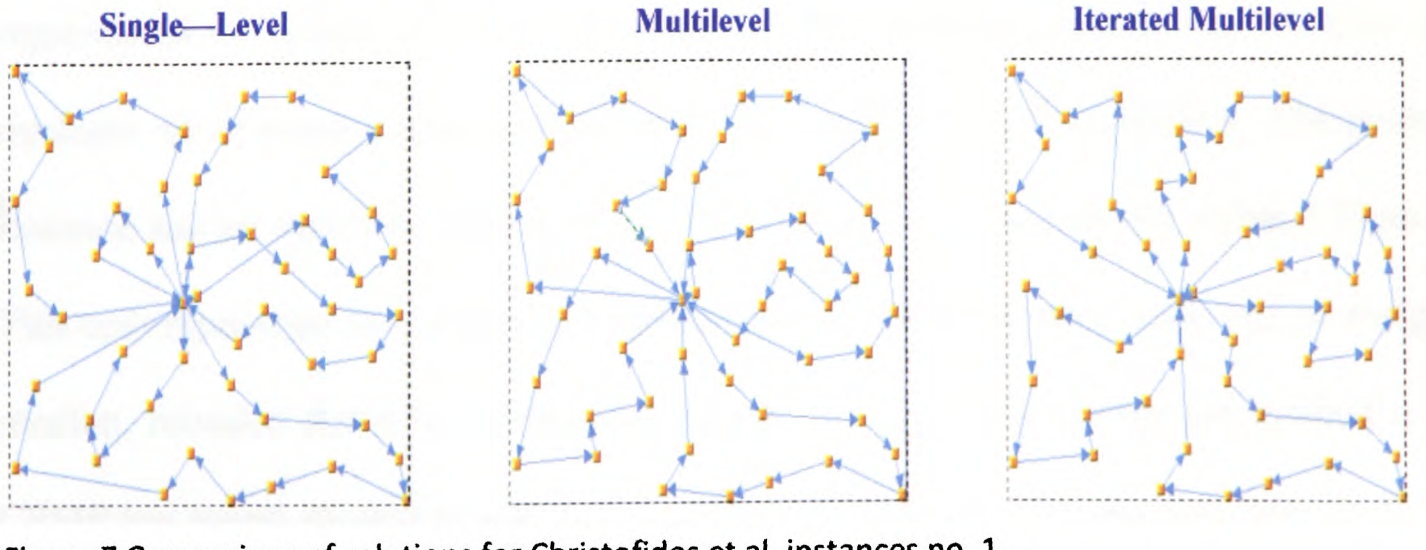

Figure 7 Comparison of solutions for Christofides et al. instances no. 1.

Table 1 Quality of the solutions for instance 1 of the Christofides et al. instances.

	Solution Cost normalised with respect to the optimal value			Runtimes (s)		
	SL	ML	It.ML	SL	ML	It.ML
Coarsening phase	1.114	1.114	-	0.2	0.2	-
Refinement phase	1.086	1.020	1.005	1.08	2.69	8.4

Two characteristics that justified applying the multilevel technique to the VRP were the fact that the nature of the objective function for the VRP is more in keeping with that of the TSP as opposed to say the GCP. Hence, decisions made locally were reflected in the global quality of the solution. Secondly, identifying customers to serve by the same route gave the solution a clustering aspect. Since the multilevel technique has performed well on problems showing varying degrees of these two characteristics, it is unsurprising that the multilevel technique for the VRP has performed reasonably well.

4.1.3 Multilevel Technique applied to the CPMP- a case study

The multilevel technique is capable of finding very good solutions for instances of the CPMP. This is demonstrated in Figure 8 where the solution produced for instance number 1 of the Osman and Christofides instances [207] using the multilevel technique is shown. The instance consists of 50 nodes, all of a stated demand, and each represented by a pair of (x, y) coordinates. The problem requires the location of 5 medians all of equal capacity meeting the demands of all 50 customers. The problem instance has an optimal solution of cost 713, which is shown on the right of Figure 8. This cost represents the sum of Euclidean distances between each node and its assigned median, rounded down to the nearest integer. The multilevel technique applied to the CPMP has found success in filtering solutions from the search space and this is reflected in the plot of Figure 9. The technique is capable of obtaining improvements to the solution cost in the upper levels of the refinement process, when the problem is in a coarsened state and the numbers of possible solutions are restricted.

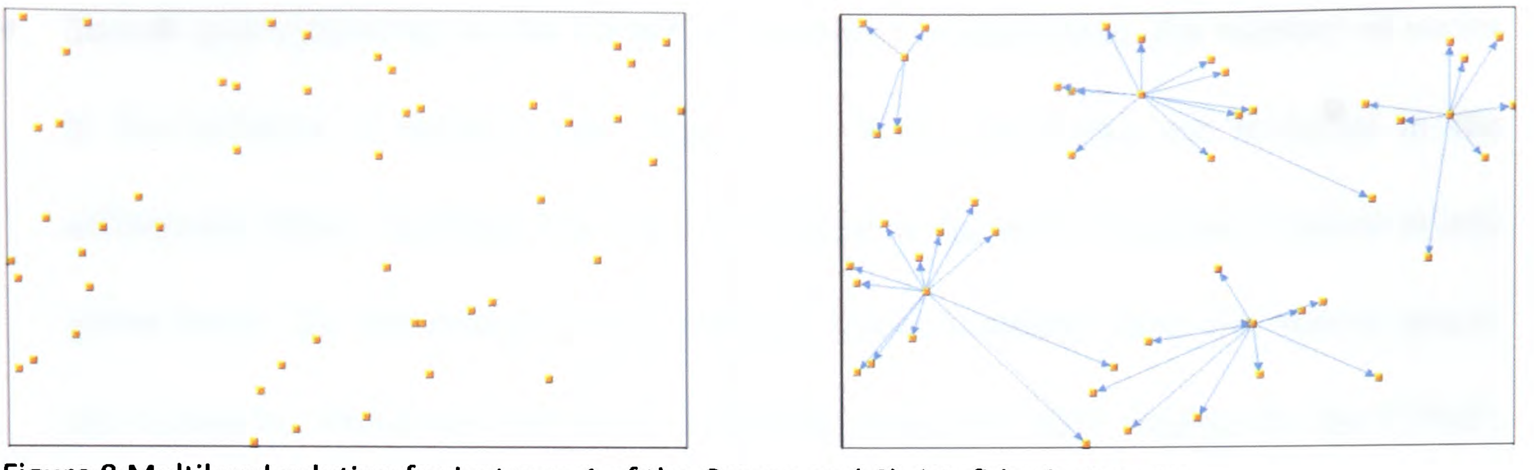

Figure 8 Multilevel solution for instance 1 of the Osman and Christofides instances.

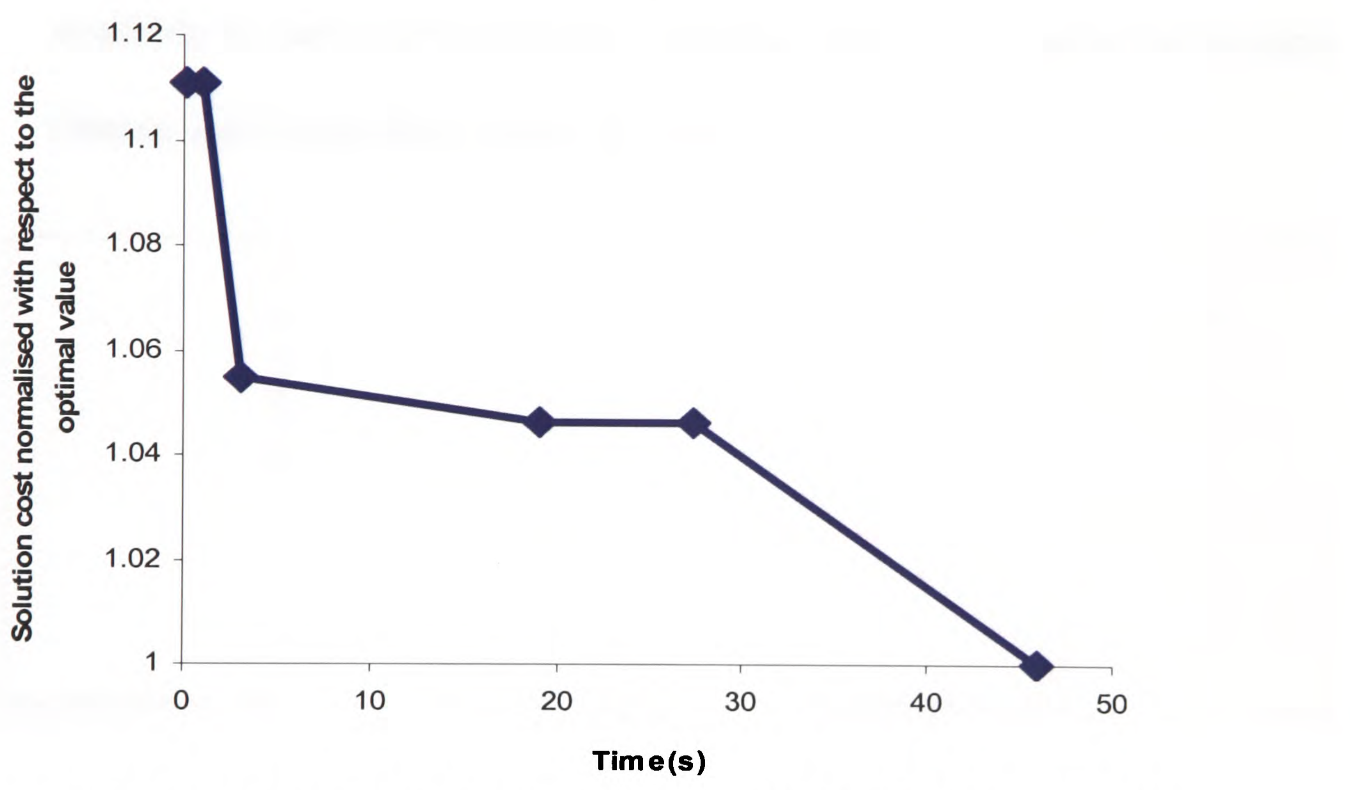

Figure 9 Refinement through the levels for Osman and Christofides instances no. 1.

4.1.3.1 Lessons from the CVRP and CPMP Case Studies

The multilevel technique applied to the CPMP and the CVRP share some similarities. The similarities stem from the fact that both techniques use coarsening to separate the set of nodes into subsets: routes in the case of the CVRP and clusters in the case of the CPMP. The techniques then apply Lin-Kernighan type algorithms in the refinement phase to improve the solution by transferring node(s) between subsets. Due to the difference in nature of the problems however, there are some lessons learnt that are of relevance to the multilevel practitioner.

- *Search space filtering*: as the CPMP is coarsened at each level, the number of nodes in the solution is reduced (see Figure 10). When the levels are revisited in the refinement phase, medians can only be located at the node locations present at any given level. By this process, the technique filters solutions from the search space: the means by which the multilevel technique makes its main impact for the CPMP. Figure 10 shows an example of the reduction in the number of median locations available at each level when the coarsening process is applied to instance 1 of the Osman and Christofides instances [207].

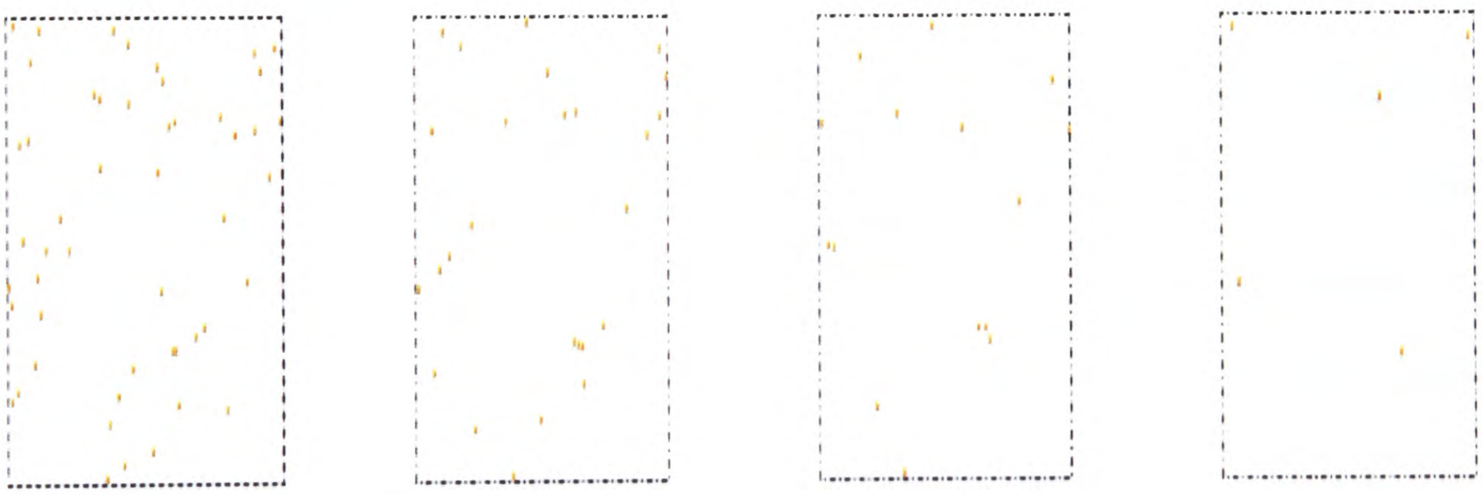

Figure 10 Coarsening instance 1 of the Osman and Christofides instances

- *Reducing the level of detail available at each level*: The multilevel technique approximates the solution space by reducing the number of possible median locations. However, the coarsening process is exact. Hence, the multilevel technique produces accurate solutions to the problem at each level of refinement. Since, in determining the accurate cost of a cluster or determining whether it improves the solution cost to transfer groups of nodes between clusters, it is necessary to calculate the cost between all the nodes and the involved medians. The (x, y) coordinates of the nodes are not filtered from the problem. This can be contrasted with the case of the CVRP, where the cost of a group of connected nodes remains constant, regardless of the route of which they are a part. While in the case of the CPMP, the cost of a group of nodes depends on the cluster of which they are a part. Hence, the only location information needed for accurate cost calculations in the case of the CVRP, assuming the internal cost of the group of nodes is known, are the pair of (x, y) coordinates at the end of the group of nodes. Preliminary tests were done for the CPMP using approximate cost calculations, thus allowing more detail to be filtered from the problem, however these results were not encouraging.

- *Impacts of the node and edge weights during the coarsening process*: The CPMP can be represented by a weighted graph and the solution seeks to minimise the sum of the edges connecting the nodes to their medians. Construction heuristics for this type of problem typically evaluate decisions based on edge weights as opposed to the weights of the nodes. However, in order to produce a feasible initial solution, as the CPMP is capacitated and the number of medians is predetermined, node weights have to be actively considered, ensuring each cluster respects the capacity constraints and exactly *p* clusters are formed. The requirement to actively consider the node weights is especially true for an agglomerative process such as the multilevel technique's coarsening phase, since as the nodes are coarsened the nature of the underlying Bin Packing Problem (BPP) changes and feasibility becomes more difficult to guarantee. This is different from the case of the CVRP, for which the number of vehicles in the solutions can typically be considered endogenous to the solution process. Therefore, the lower bound on the number of vehicles in a feasible solution is given by the number of bins in an optimal solution to the equivalent BPP and the upper bound by the number of nodes in the CVRP. Because of this, the coarsening algorithm for the CVRP only needs to ensure the routes respect the constraints.

4.2 *Part II*- The Multilevel Framework for the CVRP

As stated in section 4.1.1, in designing the multilevel algorithm for the CVRP, leading heuristics in the field of vehicle routing were used to form a composite heuristic capable of yielding high quality solutions. In accordance with multilevel terminology, we refer to this composite heuristic as the single-level algorithm for the CVRP. A multilevel version of this single-level algorithm was then created. This was done by devising various coarsening approaches and customising the local search heuristics, to improve the coarsened approximations in the multilevel algorithm's refinement phase.

The original versions of the construction and improvement heuristics used in the single-level algorithm are described in chapter *II*. This part of the chapter therefore, starts with a general discussion of the multilevel algorithm applied to the CVRP and a description of the coarsening heuristics designed for the CVRP. This is followed by a description of the modified improvement heuristics used in the refinement phase of the multilevel algorithm. This, *Part II* of the chapter, then concludes with a discussion of enhancements developed for the multilevel algorithm.

4.2.1 General concepts of the Multilevel Framework for the CVRP

The multilevel algorithm for the CVRP, developed as part of this research, uses similar techniques to the multilevel algorithm implemented for the TSP [304]. At each level, the coarsening algorithm fixes edges between customer locations to form partial routes. As more and more edges are fixed, the number of free edges decreases, simplifying the problem. This process continues while the partial routes formed, respect the problem constraints. When no more such partial routes can be formed the coarsening process terminates and the partial routes are linked to the depot, forming an initial solution.

Usually edges between nearby customers are fixed early on in the coarsening process so that, during refinement on the coarsest levels, the improvement algorithms can focus on optimising the (longer) edges of greater cost. As the solution is extended to the finer levels (and eventually the original problem) fixed edges are freed and the optimisation can then concentrate on the finer details between nearby customers. To discuss this in detail, it is worth noting the following points.

A *segment,* in the context of the CVRP, is a section of a proposed route. A segment has a cost, a demand, and spans a number of customer location(s). The segment is represented by its cost, demand and the segment's 'end locations' which correspond to either one or two of its customer locations. The cost of the segment is given by the sum of the costs of the edges between customer locations spanned by the segment plus the service cost of those customers. If the segment spans just one customer location, the segment end locations are both equal to this customer's location. If however the segment spans more than one customer location the segment's two end locations correspond to the location of the two customers having only one connecting edge. All other customer locations (those excluding the end locations) spanned by the segment have two connected edges.

At the start of the solution process, referred to as *level zero,* a segment represents a single customer (or a vertex). Hence, the segment's demand is equal to the customer's demand. Additionally at this level, in the case of the CVRP, the cost of the segment is zero and in the case of the DVRP, the cost is equal to the service cost of the customer.

Segments at the upper levels (levels excluding level zero) are created by fixing an edge of least cost between a pair of existing segments. If one or both of the two segments being connected spans more than a single customer, the edge chosen to connect the pair, connects two of the four (or two of the three if one of the segments represent a single customer) available end locations such that its cost is minimised. If more than one edge has cost equal to the minimum cost, then one of these edges is chosen arbitrarily. The two unconnected end locations become the end locations of the created segment.

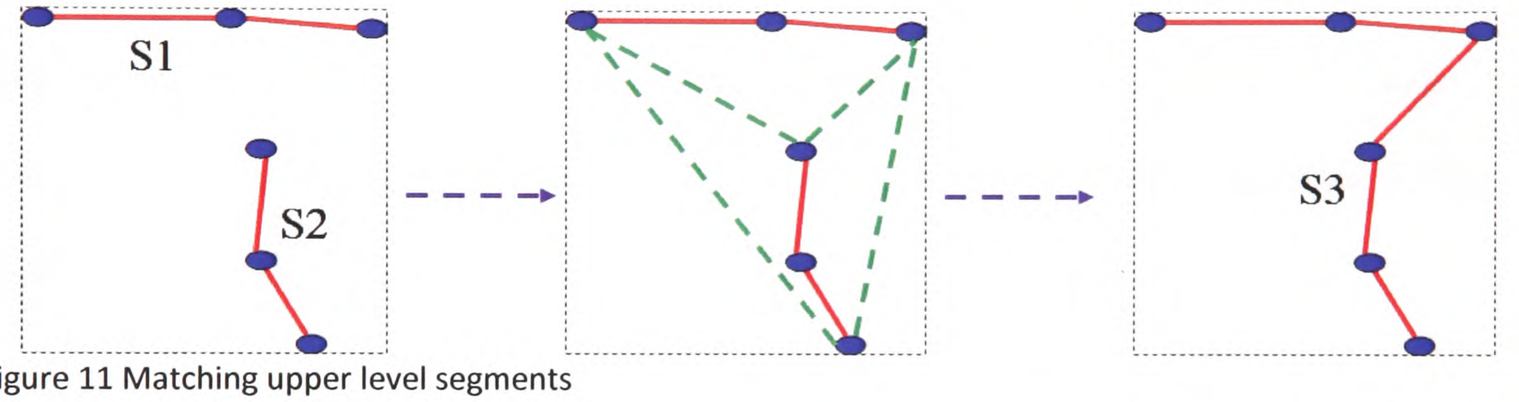

Figure 11 Matching upper level segments

The creation of on an upper level segment is depicted in Figure 11 where the original customer locations are shown by the spheres. The dashed lines indicate the cost calculations done between the four end locations of segments *S1* and *S2* in creating segment *S3*.

4.2.1.1 Fixed Edges and Free Edges

Using *fixed edges* and *free edges*[1], the multilevel algorithm for the CVRP, controls the process of simplifying the problem and filtering solutions from the search space.

---

[1] The segments created during the coarsening phase, generate a series of approximations to the original problem. In the refinement phase, that follows the coarsening phase, these segments are used to form feasible solutions to the problem. *Free edges* are those edges that can be used to transform the segments into feasible solutions. *Free edges* can be introduce into the solution to join the existing segments into feasible routes. *Free edges* can also be direct replacement for fixed edge. These replacements occurs when the fixed edges are freed when the level at which they were fixed are revisited in the refinement phase.

The edges connecting the customer locations within a segment's end locations are fixed into the solution, i.e. fixed edges, while edges linking segments together are considered free edges.

The following example (see Figure 12) illustrates the difference and the purposes they serve. Suppose that at a given level above zero, two segments span a set of customer locations (i, j, k, l, m, n) such that segment $S_1 = (i, j, k)$ and segment $S_2 = (l, m, n)$. The following fixed edges exist in the solution at this level: edges E_{ij} and E_{jk} in the case of S_1 and edges E_{lm} and E_{mn} in the case of S_2. The end locations of S_1 are i and k and the end locations of S_2 are l and n. At the same level the two segments could be joined to the depot o and each other, forming the following vehicle route $r = (o, S_1, S_2, o)$. Route r could be represented as (o, i, j, k, l, m, n, o), however only the following edges in the route would be free for optimisation at the current level E_{oi}, E_{kl} and E_{no}.

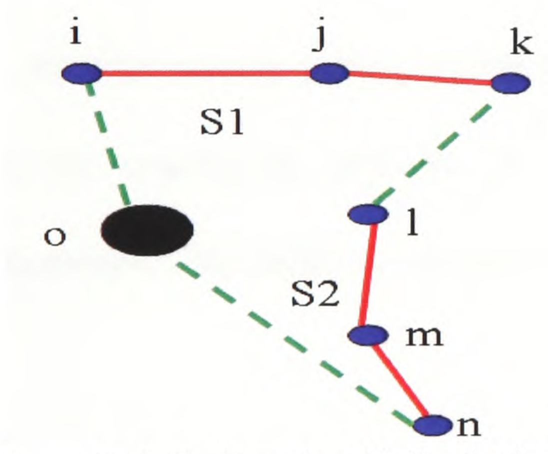

Figure 12 A CVRP route with fixed edges (solid lines) and free edges (dashed lines).

Free edges therefore, identify the only sections of a route that can be optimised at a particular level. The sections of a route within the end locations of an upper level segment demarcate sections of the route that are not available for improvement during the refinement phase, while those edge(s) are fixed. Hence, any solutions that does not contains the customers spanned by that segment and visited in the order they are connected by the segment, is filtered from the search space until the refinement phase frees the fixed edges of the segment.

The refinement algorithms treat segments of level zero (single customers) and segments of the upper levels (group of customers) in a similar manner, as the internal structure of a segment is not accessible.

4.2.1.1.1 Free Edges and the Depot

In addition to the free edges linking segments together, as in the case of the edge E_{kl}, in Figure 10, the other free edges in the solution, are the edges connected to the depot (also as shown in Figure 12).

As the coarsening algorithm fixes edges into the solution, the edges between the depot and the end locations of segments are always kept free. The reasons for not fixing these edges into the solution are manifold. For example, segments are transferred between routes in the refinement phase. An infeasible solution would result if a transfer were done between two routes, where the segment transferred spanned the depot.

Another reason centres on the fact that predominantly savings-type heuristics are used in the coarsening process. In order for these heuristics to operate efficiently and accurately, the ability to change edges connecting segments to the depot is required.

4.2.1.2 Synopsis of the application of the multilevel technique to the CVRP

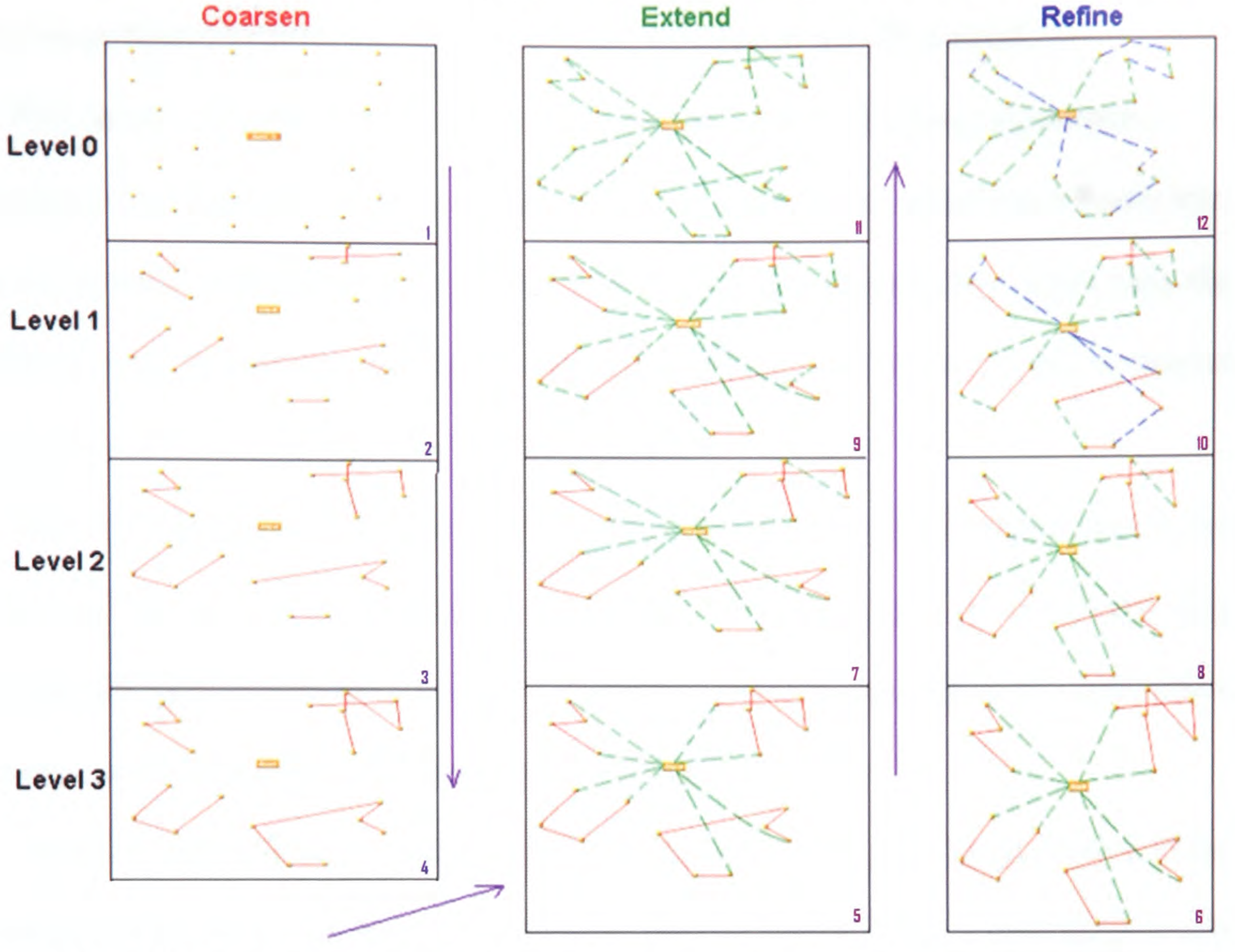

Figure 13 Multilevel refinement applied to a CVRP showing the stages of coarsening, extension and refinement. Continuous lines show fixed edges and broken lines show free edges

Figure 13 shows the multilevel algorithm solution process applied to the CVRP. Coarsening the problem achieves two things. Firstly, it constructs a solution to the problem by fixing edges (represented by solid lines) between segments. It is hoped that these edges will form part of a high quality solution, but they can be changed later as the refinement progresses.

Secondly, it reduces the detail to be considered at each level for when the problem is refined. An edge that is fixed at a given level of the coarsening is freed by extension process [304] when that level is revisited in the refinement phase. The refinement phase optimises the free edges (represented by dotted lines) in the solution[1]. Segments that

[1]This process involves seeking to replace them with edges of lower cost

contain fixed edges are treated as single nodes, capable of being optimised by node exchange heuristics [98], but the internal fixed structure is not considered.

This means a minimal level of detail is presented to the refinement algorithms at the coarsest levels of the refinement process. However, edges fixed at the lower levels tend to be between segments that are 'closer' together, so that typically free edges have the highest cost. Potentially, the largest improvements in cost found by the multilevel algorithm will occur early in the refinement phase.

When the optimisation is completed at a level the solution in place is projected to the level below, the extension process then frees the edges fixed at that level. The refinement process now has a problem of greater detail, but improvements found earlier in the process help make the search faster (see section 3.1.1.1).

For the example shown in Figure 13, no improvement is found at levels 3 and 2 of the refinement process (it can be the case that no improvement is found at the highest levels since there are so few free edges). However refinement does take place at levels 1 and 0. The refinement done at level 1 is on upper level segments. This demonstrates one of the benefits of the multilevel algorithm over the single-level case, that of refining the coarsened problem. As the move is implemented on the coarsened problem the algorithm is required to analyse a smaller subset of the edges in the problem than would be required by the single-level algorithm.

4.2.2 Coarsening for the CVRP

Two types of coarsening are implemented for the CVRP. The first type uses coarsening to construct the initial solution. This is the more traditional way of coarsening a problem in multilevel refinement, e.g. [304] [305]. The coarsening algorithms used in those implementations are different, but coarsening was used to construct the initial solution.

The second type of coarsening implemented for the CVRP is a two-phase coarsening approach in which an initial solution is constructed and then each route in the solution is coarsened. This form of coarsening is new to this research. By separating the construction and the coarsening phase, the two-phase coarsening approach allows a multilevel practitioner to use the solution construction heuristic(s) of their choice while still having the ability to coarsen the problem.

Where coarsening is used to construct the initial solution, two different methods are implemented for selecting the customers to be merged. When two-phase coarsening is employed, two different methods are used for constructing the initial solutions. These lead to four distinct methods of coarsening. Each method is presented in the following sub-sections.

4.2.2.1 Coarsening used to create the initial solution for the CVRP

The process of using coarsening to construct an initial solution for the CVRP is illustrated on the left of Figure 13 (p. 85). At each level, segments are *matched* in pairs, using either a *savings heuristic* or a *nearest neighbour* heuristic. An edge is then fixed between the two segments making up each pair (the pair is *merged*) and a new segment created. The process continues while there are pairs of unmatched segments at the current level from which it is possible to create new segments that respect the problem constraints. Segments are merged once at a given level. The created segments should respect the problem capacity and any cost constraints in place. This requires that the sum of the demands of the customers spanned by a segment should be less than or equal to the maximum route capacity. It also requires that the segment's cost should be less than or equal to any maximum route cost.

When all allowed merges at the current level have been performed, the created segments are included in the next level. Additionally, any segments that could not be merged at the current level are included unchanged in the next level and the process of matching and merging repeated. The last level of coarsening is reached when no new segments can be created respecting the problem constraints. The end locations of the segments are then connected to the depot to form initial routes. The algorithm of Figure 14 outlines the generic coarsening process. A savings heuristic or a nearest neighbor heuristic [247] is used to match and merge the segments.

Figure 14 Generic coarsening algorithm

```
do
  do
      merge pairs of selected segments
  while unmatched pairs of segments
  include merged segments in new level
while new level can be created
```

The ***savings heuristic***, similar to the serial Clark-Wright savings heuristic [1] (see section 2.3.1.3.1), initially assumes each segment forms a route. Each segment can be merged once at a given level, and hence, the heuristic selects a segment at random and merges it to another unmerged segment yielding the greatest savings.

If a route exists, serving segment i and another serving segment j, and there exists a feasible route capable of serving segments i and j, the heuristics calculates the saving

(S_{ij}) obtainable by merging the routes of segment i and j using the formula: $S_{ij} = C_{io} + C_{jo} - C_{ij}$. The cost between segment i and the depot (o) is represented by C_{io}, C_{jo} represents the cost between segment j and the depot and C_{ij} represents the cost between segments i and j. When merging routes consisting of upper-level segments, i and j actually refer to the segments' end locations that are chosen to be merged. These end locations are selected such that S_{ij} is maximized. Any merges violating the problem constraints are given a prohibitively low savings.

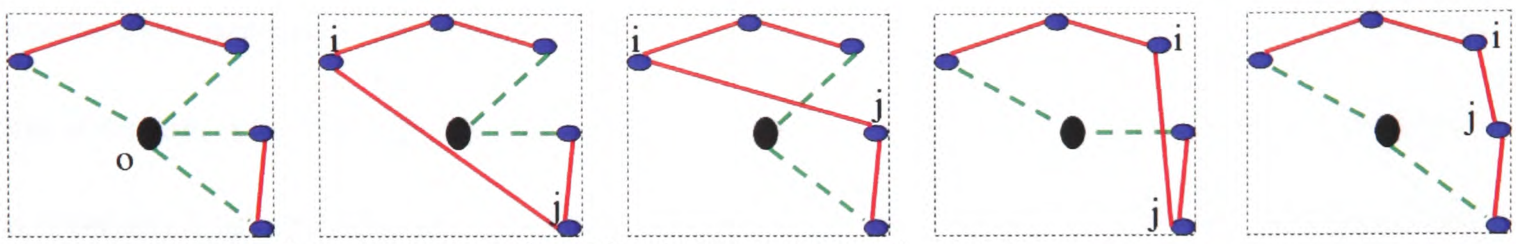

Figure 15 An upper level merger analysed by the saving heuristic

The first screen on the left of Figure 15 shows a solution consisting of two routes with each route consisting of one upper level segment[1]. The other screens show the four moves analysed by the savings heuristic in determining S_{ij}, with i and j labeling the end locations that are merged in each move. The last screen shows the merge that maximizes S_{ij} and hence the one that would be implemented.

The ***nearest neighbor heuristic*** selects a segment at random and merges it with its nearest unmerged neighbour at the current level. Any merges violating the problem constraints are given a prohibitively high cost. Since the cost is calculated between the segments, the depot is ignored during the merging process, unlike for the savings heuristic.

[1] When coarsening used to create the initial solution a route will only ever consist of one segment during the coarsening phase.

Using the savings or nearest neighbor heuristic, the coarsening algorithm of Figure 14 coarsens the problem using the principles applied to the TSP [302], and hence provides a simple and fast construction heuristic [304].

4.2.2.2 Two-phase Coarsening applied to CVRP – Route Construction

The main difference between the two-phase coarsening approach presented in this section and the coarsening approach presented in section 4.2.2.1 is that the two-phase approach constructs a solution and then fixes edges into the solution, while the coarsening approach presented in section 4.2.2.1 uses the fixing of edges to construct the solution. The two-phase coarsening approach uses one of two construction heuristics to construct an initial solution. These two construction heuristics are the parallel Clark-Wright savings heuristic [51],[1],[55] and a parallel Nearest Neighbour heuristic.

The *parallel Clark-Wright savings heuristic* (CWS) (see section 2.3.1.3.1) implemented, uses the formula for calculating savings outlined for the savings heuristics. However, the differentiating feature is that the parallel version ensures that the best feasible merge is always implemented. Hence, the *savings* to be obtained from merging each pair of routes is calculated. The customers served on the pair of routes yielding the greatest savings are combined into a single route, reducing the number of routes in the solution by one. Any merge violating the problem constraints is given a prohibitively low saving. The process is repeated while routes can be merged which leads to savings and while the problem constraints are respected.

The *parallel Nearest Neighbour heuristic* (see section 2.3.1.3.1) proceeds in a similar manner to the parallel CWS, ensuring the best feasible merge is always implemented. However, customers are merged based on their cost relative to each other as opposed to using the savings formula.

4.2.2.2.1 *The Coarsening process in two-phase coarsening*

The coarsening algorithm is applied to the routes of the initial solution in turn, terminating when each route is represented by a single segment whose end points are connected to the depot. If at each level, each route in the solution is represented in the format (*o*, *i*,....., *x*, *o*) where *o* is the depot and *i* and *x* are the end segments. The coarsening heuristic fixes edges between pairs of segments starting with *i* and *i+1* followed by *i+2* and *i+3* and terminating with *x-1* and *x*. All the new segments created are included in the next level. If the segment consisting of *x-1* and *x* can't be formed, then segment *x* is projected to the next level. The process is repeated independently per route until each route is represented by one segment[1]. The last level of coarsening corresponds to the highest level required to coarsen a route in the solution to a single segment.

4.2.3 Refinement for the CVRP

Using the initial solution created at the end of the coarsening phase, the refinement process seeks to improve the quality of the solution at each level by reducing the total cost. The refinement discussed here uses a combination of inter- and intra-route local search heuristics.

These heuristics are based on a number of standard VRP heuristics and are applied in a fixed sequence at each level as shown in Figure 16. In relation to the generic multilevel algorithm of Figure 3 (pp.55) each call to refine() in that algorithm, executes the algorithm of Figure 16.

Intra-route optimisation is performed by the 3-opt heuristic, while inter-route

[1] An alternative heuristic was implemented that sorted the segments of each route by the increasing cost between the segments. Segments were then merged in pairs of by increasing cost, with each segment merged once at each level. However, this heuristic did not performed as well as the one described above.

optimisation is performed by the split procedure, simple and cyclic segment transfer heuristics.

Figure 16. Refinement algorithm executed at each level.

```
Split Procedure
do
        3-opt refinemnet
        do
                Simple Segment Transfer
                Cyclic Segment Transfer (If improvement found, continue)
        while(improvement found)
while(improvement found)
```

4.2.3.1 3-opt Refinement first improvement.

3-opt [172] is an intra-route heuristic that is executed at each level, searching for ways of reconnecting the segment ends (at that level) of a route, to form a route of lower cost. The 3-opt heuristic implemented in the multilevel framework accepts the first improving solution found. Blum and Roli refer to this as *first improvement* [28], Funke et al. as *first search* [98].

The 3-opt heuristic implemented in the multilevel algorithm for the CVRP can be stated as: For a given route *r*, remove three *free edges* (none of which should be adjacent) and reconnect the segments such that the resulting route is feasible. If this results in a solution of lower cost, accept the new solution.

This heuristic therefore, represents a modified version of the standard 3-opt heuristic (see section 2.3.1.4.1). Since the type of solutions for which the standard 3-opt heuristic was originally designed are represented by vertices and edges, the standard 3-opt heuristic needs only check for non-*adjacent edges.* However, since multilevel CVRP solutions require the optimisation of solutions containing upper-level segments, the 3-opt heuristic implemented here works on non-*adjacent free edges*.

Suppose that the provisional route r is of the format $r = (o, i, \ldots\ldots, j, o)$ where i is the first segment on the route, j the last segment and o is the depot. Assuming r contains n free edges, which are numbered one to n starting with the first free edge $E_{\_oi}$. The other free edges, traversing the route from the first segment to the last, are numbered consecutively with the last free edge $E_{\_jo}$ numbered n. Given a free edge in position i, let the next free edge not adjacent to this edge, be in position i+x. Additionally let the first free edge not adjacent to edge i+x be in position i+y where: $1 \leq i < x < y \leq n$; and i , x, y $\in \mathbb{Z}^+$. This 3-opt heuristic as implemented is described in the heuristic of Figure 17.

Figure 17 The 3-Opt heuristic using first improvement

```
do
     set first free edge position i = 1
     while ( i <= n-2) do
          set second free edge position j = i+x
          while ( ( if (i ≠ 1 ) && (j < = n-1)) | | ( if ( i == 1) &&(j < n -1 ) ) do
               set third free edge position k = j+y
               while ( ( if ( i ≠ 1) &&(k <= n)) | | ( if ( i == 1) && (k < n ) ) do
                    remove free edges i , j and k reconnect the segments
                                          constructing all possible feasible solution.
                    If a connection improve solution cost restart the algorithm
                    k := k + 1
               end while
          j = j +1
          end while
     i = i+1
     end while
while (improvements in solution cost found)
```

4.2.3.2 Inter-route optimisation

The immediately following sections (4.2.3.3 to 4.2.3.5) describe the inter-route heuristics used in the refinement phase. While, these heuristics are based on standard VRP heuristics, various modifications have been added to each in order to facilitate the optimising of upper-level segments. When an upper level segment is transferred between two routes, it can be inserted in one of two ways by either reversing its orientation, or not. The main modification to the heuristics therefore, centres on the heuristic checking both orientations to find the one resulting in the connection of least cost.

4.2.3.3 The Split Procedure

The split procedure is a heuristic capable of introducing large changes to CVRP solutions. Given a Giant Tour, the procedure is capable of selecting the optimal combinations of routes from the Giant Tour and, in some cases, returns a solution of improved cost (see section 2.3.1.5). We have implemented the split procedure for the CVRP multilevel algorithm. This was done as a means of introducing large changes to the solutions that were outside the scope of the other inter-route heuristics and as a means of relaxing the problem constraints.

Before the start of the split procedure, a Giant Tour is constructed. This is done by removing all the edges connecting the routes of the CVRP solution to the depot, forming a series of sub-routes. These sub-routes are then used to construct the Giant Tour. The Giant Tour is constructed by randomly selecting a sub-route and arbitrary setting one of its ends as the start of the Giant Tour. The other end of this sub-route is then connected (using the least cost edge) to the nearest unconnected sub-route end which is not yet part of the Giant Tour. This procedure is repeated for the partially completed Giant Tour until all sub-routes are connected. The unconnected end of the last added sub-route is then linked to the unconnected end of the first added sub-route to complete the Giant Tour.

The stages of a Giant Tour construction are shown in Figure 18. It is worth noting that some Giant Tours constructed by the multilevel algorithm will contain upper-level segments (depicted in the diagram with the use of solid lines). This is as opposed to containing only vertices, the scenario for which the original split procedure was designed. The split procedure used in this research is therefore modified to optimise upper-level segments. The modification ensures that when a segment is added to a route, the orientation of the segment in the route is chosen such that the cost of the new edges added to the route are minimised.

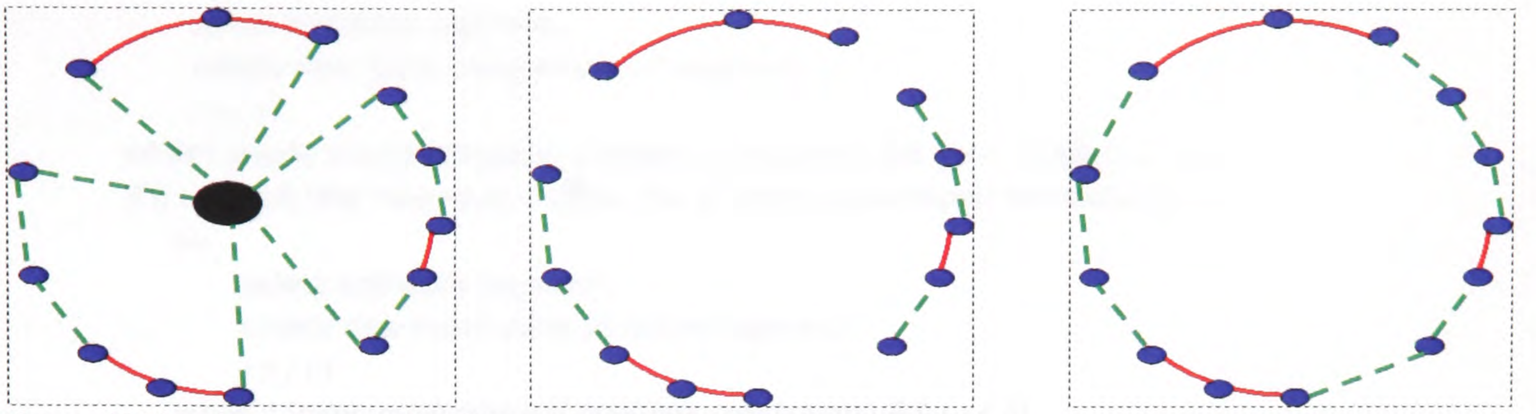

Figure 18 The three stages of a Giant Tour construction.

The split procedure is applied to the Giant Tour to recover a CVRP solution, satisfying any constraints, and hopefully of lower cost. The algorithm for the split procedure implemented is shown in Figure 19, and assumes the Giant Tour contains n segments where $n > 1$. Since the Giant Tour may contain upper levels segments this effectively means the split procedure is only allowed to splits the Giant Tour at the points corresponding to the free edges. This is another modification to the split procedure that was implemented for its use with the multilevel technique.

Figure 19 The Split Procedure

```
set starting segment position i = 1
set problem constraints
do
    select and store segment i
    create a route comprising only segment i
    if (i < n ) set next segment position j = i+1
    do
        select and store segment j
        create new route using all stored segments
        j = j +1
    while ( (route created respects problem constraints) && (j <= n) && ( j > i) )
    if (j > n)  && (the new route respect the problem constraints) reinitialize j = 1
      do
          select and store segment j
          create new route using all stored segments
          j = j +1
      while  ( (new route respects problem constraints) && ( j < i))
    reset stored segments to zero
    i = i+1
while (i <= n)
solve set portioning problem to select optimal combination of routes[1]
```

The split procedure has also been useful for implementing constraint relaxation (See section 4.2.4.1). A CVRP solution that violates the constraints can be used to form a Giant-Tour as above, and then the split procedure is applied, but now with the desired relaxation of problem constraints at that level enforced by the procedure. The formation of a Giant Tour followed by a split can be performed at any level.

4.2.3.4 Simple Segment Transfers

As with Osman's λ- Interchange [206], the simple segment transfer heuristic searches for improvements by looking at the effects of moving segments between every pair of routes in a solution. For all segments in routes q and p, the heuristic considers all possible insertions in a route q of a segment from route p and vice-versa. This move is

[1] No modifications to the algorithm for solving the set-partitioning problem were necessary in order for the algorithm to work with the multilevel technique. Consequently, this algorithm is not reproduced here. However, its original specifications are provided by [29] and [225].

called a *transfer* and symbolized by the *(0, 1)* and *(1, 0)* operators. These operators indicate that only one segment is relocated from a route with each transfer.

The heuristic also implements exchanges where two segments on different routes exchange places i.e. each is inserted into the other route with the same neighbours as the removed segments, this is termed an *interchange* and again the move is attempted for all segments in the routes. The interchange move is symbolized by the *(1, 1)* operator.

Osman's λ- Interchange considers additional transfers involving groups of neighbouring vertices. We however restrict our attention to the case where only one segment at any level is removed from a route i.e. λ is equal to one in Osman's terminology, for the following reasons.

The purpose of increasing the value of λ, is to introduce increasingly larger changes to the solution than that achievable using $\lambda = 1$. However, this increases the runtime of the heuristic. A measure of the likely effects on runtime from increased λ values can be garnered from looking at the moves permitted by $\lambda = 1$ and $\lambda = 2$. When λ is equal to one, the following moves are allowed between routes q and p; *(0, 1)*, *(1, 0)* and *(1, 1)*. However when $\lambda = 2$, the following moves are permitted *(0, 1)*, *(1, 0)*, *(1, 1)*, *(0, 2)*, *(2, 0)*, *(1, 2)*, *(2, 1)* and *(2, 2)*.

The transfer or interchange of upper level segments in the multilevel algorithm results in the transfer of groups of neighbouring vertices. Therefore the multilevel algorithm can achieve a measure of these larger changes associated with $\lambda > 1$ by implementing moves of upper level segments, with $\lambda = 1$. This avoids the large increases in runtime associated with increasing values of λ.

Simple segment transfers are a special case of cyclic segment transfers - described next - and have been implemented as such.

4.2.3.5 Cyclic Segment Transfers

Low-cost solutions to the CVRP are often composed of routes that are close to the maximum route capacity [240]. Thus, when searching for inter-route improvements in a high quality solution, the transfer of a customer from one route to another will normally require the removal of one or more customers from the route into which it has been inserted. The ejected customer(s) will then have to be inserted into another route and so on. This has led to the construction of cyclic transfer algorithms, in which the set of allowed transfers between routes forms a cycle – the customers ejected from the last route are inserted into the first.

The number of possible cycles when using cyclic transfer algorithms is influenced by three main factors. These are, the number of customers in the solution (for the multilevel algorithm, we are concerned with the number of segments at each level), the number of routes in the solution (as the number of routes decreases for a constant number of customers the number of possible cycles decreases); and the maximum number of routes allowed in a cycle (i.e. the cycle depth). Increasing the cycle depth, for a constant number of routes and customers, increases the number of possible cycles.

The cyclic segment transfer heuristic implemented here follows the cyclic transfer algorithm of Thompson and Psaraftis [286]. However, it is restricted to the case where only one segment is transferred at a time. Since, segments represent groups of customers; the heuristic is capable of executing large changes to the solution while transferring one segment at a time. The heuristic is modified to handle the transfer of segments, ensuring the orientation of each transferred segment is such that the cost of each insertion and ejection is minimised. The cycle depth is varied during the algorithm's execution.

The scheme works as follows: For each segment i in a given CVRP solution, the cyclic transfer algorithm identifies all segments $j \neq i$, that can be feasibly ejected from their route R_j with segment i inserted in its place. It is feasible to eject segment j and insert segment i if they are served by different routes and R_j will remain a feasible route if i is added and j is removed. The algorithm then notes the change in cost of R_j for each feasible insertion and ejection. A cycle that improves the solution cost then corresponds to a series of changes in cost whose summation is negative. The heuristic of Figure 20 searches for these improving cycles utilising efficient techniques for finding elementary circuits in graphs [287].

If the insertion of a segment into a route and a corresponding ejection results in an infeasible route, no cycle consisting of that move is considered. If an insertion into a route is feasible without a corresponding ejection and it leads to an improvement in the solution cost this move is accepted and the series of transfers is terminated.

When segment i is inserted into R_j and segment j ejected, it is possible for the new neighbouring segments of segment i to be different from those shared by segment j. The position segment i takes in R_j is termed the *least cost insertion point*. The least cost insertion point determines, when inserting segment i into route R_j while ejecting segment j, which location for i reduces the cost of R_j by the largest amount compared to its cost before j was ejected. If inserting i increases the cost of R_j compared to the cost before j was ejected, the least cost insertion point is that position in R_j that increases the cost by the least amount. This point can be efficiently pre-calculated [286] and Thomspson and Psarafits [286] demonstrated this will be in one of four locations. It will either be in one of the three locations in R_j (excluding the position occupied by j) where inserting i before ejecting j, increases the route cost of R_j by the least amount.

Alternately, the least cost insertion point will be the position occupied by j before j is ejected.

In calculating the change in cost of inserting segment i and ejecting segment j, where segment i is an upper level segment, both possibilities for connecting segment i to reform a feasible route, after the ejecting of segment j, are analysed. The one yielding the route of best cost is implemented. If the least cost insertion point is different from the position occupied by j, the change in cost of R_j is given by subtracting the sum cost of the three edges removed from R_j and the sum cost of three edges added to R_j. If least cost insertion point is the same as the position occupied by j, the change in cost of R_j is given by subtracting the sum cost of the two edges removed from R_j and the sum cost of the two edges added to R_j. It should also be noted that for each insertion and ejection the changes in costs are independent for each route. Hence, the change in cost of R_j for inserting i and ejecting j is independent of the change in cost to be incurred when removing i from its route and also that of adding j to another route.

As the insertion location of i can be independent of the one vacated by j, cyclic segment transfer produces a more powerful search than that provided by λ-interchanges, even for cycle depths of two. Following Thompson and Psaraftis [286] we use iterative deepening, searching for cycles of depth 2, then 3 and so on until the maximum cycle depth allowed. Whenever an improvement is found, the search for cyclic transfers is halted and simple segment transfers are again sought (see Figure 16 , p.92).

Nomenclature Cyclic Segment Transfers heuristic:

S_i, S_j, S_q - Segments i, j , q respectively

R_i, R_j - Routes serving segments S_i and S_j respectively

S[i][j] – Feasible change in cost of inserting segment *i* into R_j while simultaneously ejecting segment *j*

S[i][j] is feasible if: $i \neq j$; $R_i \neq R_j$; R_j demand <= max route demand ; R_j cost <= max route cost

n, i, j, q, x, y $\in \mathbb{Z}^+$

$j + x$ next feasible ejection for inserting segment *i* ;

q + y next feasible ejection for inserting segment *j*

cycle cost $\in \mathbb{R}$; cycle depth > = 2

Figure 20 Cyclic Segment Transfers heuristic

```
Function Transfer ()
    set number of segments in solution at level n
    set segment to be inserted position, i = 1
        set cycle cost = 0
        set cycle depth counter, d = 0
    Double Array S = change in cost for feasible insertion of i while ejecting j
    Array P = segments making up a cycle
    while ( i < = n ) do
            set j to position of first feasible insertion for Si
        while ( j < = n ) do
            add Si and Sj to P
                    d = d + 1
                    cycle cost = cycle cost + S[i][j]
            Search (j)
            If( no improving cycle found ) j = j + x
            else update S restart search from i = 1
              start new cycle - removing all elements from P; cycle cost = 0; d = 0 .
        end while
        i = i+1
    end while
End Function
Function Search (j ∈ Z+)
        set q to position of first feasible insertion for Sj
    while ( (q <= n ) && ( d < cycle depth)) do
            if( q = = i)
                if (cycle cost < 0), accept cycle , End Function
            If(q ∉ P )
                Add Sq to P
                d = d + 1
                cycle cost = cycle cost + S[j][q]
        Search (q)
    q = q + y
    end while
End Function
```

4.2.4 Multilevel Enhancements

Problem constraints often prevent moves that would otherwise lower solution costs. This is particularly restrictive at higher levels in multilevel algorithms and, for example in the CVRP, segments can have demands which are a substantial fraction of the maximum route capacity, thus preventing inter-route moves.

With this in mind, two enhancements were implemented for the algorithms which were found to work well together: constraint relaxation (section 4.2.4.1) and coarsening homogeneity (section 4.2.4.2). Both enhancements appear to complement each other since, coarsening homogeneity is targeted at managing the effect the multilevel technique has on the problem, while constraint relaxation is targeted at managing the effect the problem constraints have on the solution process.

We have also implemented an iterated version of the multilevel algorithm (section 4.2.4.3) which is able to find significantly better results, although taking longer to do so.

4.2.4.1 Constraint relaxation

Constraint relaxation [305] is the process of gradually relaxing the problem constraints [108] at each level.

There are two goals desired from constraint relaxation. The first is the creation of a set of infeasible routes, of very good cost, from which it will be possible to obtain a set of feasible routes. The set of feasible routes should be obtained while maintaining some of the cost improvement found from relaxing the constraint.

The second goal is to produce solutions more amenable to improvement by inter-route heuristics. The amenability of routes to inter-route heuristic improvement is increased as the ratio between the total capacity in the problem (number of routes * capacity per route) and the total demand of customers in the problem, is increased. This ratio is referred to as the tightness of the capacity constraints.

For the CVRP, constraint relaxation was applied to the demand constraints. Two methods of constraint relaxation were implemented. For both methods a *capacity overload factor* (*cof*), an experimentally calculated real number is specified (see section 5.2.3.2). The *cof* acts as an upper bound on the amount the capacity constraints (*RC*) can be relaxed in relation to the original route capacity (*ORC*).

4.2.4.1.1 First method of Constraint relaxation

Using the following relaxation, the first method gradually relaxes the *RC* at each level during coarsening: $RC_{(i+1)} = RC_{(i)} + (cof - 1) * 0.1 * ORC$. The relaxation is applicable at level i if the resulting $RC_{(i)} <= ORC * cof$. For level $i = 0$, $RC_{(i)} = ORC$.

During refinement, the constraints are gradually brought back into line with the original values imposed by the problem while attempting to preserve the improvement in cost found by relaxing the constraints. A feasible solution is regained when the original problem constraints are satisfied, which occurs at level zero, although feasible solutions may be found at higher levels.

The Split Procedure is used to tighten the constraints. This is achieved by inputting to the procedure the route capacity allowed at level i. The procedure is then able to construct a set of routes of capacities less than or equal to the input value, from a given set of routes with capacity values equal to the route capacities allowed at level $i+1$.

4.2.4.1.2 Second method of Constraint relaxation

The second method of constraint relaxation is implemented entirely in the refinement phase. For this method, a feasible solution exists at the start and end of the refinement phase, with the constraints relaxed and strengthen at each level of the solution between these two points.

This method uses the relaxation: $RC_{(i-1)} = RC_{(i)} - (cof - 1) * 0.1 * ORC$. The relaxation is applicable at level i if the resulting $RC_{(i)} >= ORC$. When i = first level of refinement, $RC_{(i)} = ORC * cof$ and for $i = 0$, $RC_{(i)} = ORC$.

The relaxation determines the allowed route capacity at a given level and again the Split Procedure is used to generate a solution consisting of a set of routes, of capacities up to and including the targeted value. The routes are then refined in accordance with the refinement algorithm executed at each level (see Figure 16, pp. 92).

4.2.4.1.3 Evaluation of the methods of Constraint relaxation

The merging of selected segments during the coarsening process is allowed once they meet the constraints in place. If the capacity constraint is relaxed during coarsening, this means segments are potentially created having demands close in value to the relaxed capacity value. Because of this feature, the two methods of constraint relaxation influence the refinement process in different ways. As the first method facilitates the construction of individual segments close to the relaxed constraints it was found that the goal of finding more improving moves in the upper levels was not greatly achieved (see section 5.2.4). Conversely, as the second method relaxes the constraints during refinement, there were no segments in the solution with a demand value exceeding the problem stated route capacity. This meant that the second method was better able to provide the additional capacity in the solution required for inter-route heuristic improvement at the higher levels. Based on our experimentation this seems to be the best method of constraint relaxation for the multilevel algorithm (see section 5.2.4).

4.2.4.2 Coarsening Homogeneity

Coarsening homogeneity [305], the second of the enhancements implemented, is the process of creating segments during the coarsening phase of approximately the same demand and/or cost. This increases the likelihood that an insertion of a segment and simultaneous deletion of a different segment from a route will be allowed by the problem constraints. Cyclic transfers and simple segment transfers are composed of such operations and so should benefit from 'homogeneous' segments.

At each level, a targeted level demand is specified and, in the case of the DVRP, a targeted cost is also specified. The homogeneity enhancement then seeks to enforce two things: firstly, that no segment created at a given level can exceed the constraints set at that level; and secondly, if a segment is created at a given level and it is possible to merge it with another segment at the current level while respecting the constraints at that level, the merge is implemented. Consequently, unlike standard coarsening, when coarsening homogeneity is employed a segment can be merged more than once at a given level.

Figure 21 is used to illustrate these effects. Assume all the initial segments are of the same demand and the lengths of the edges are proportional to the costs of the segments they represent. In *part A* of Figure 21 sections 1 and 2 show the normal coarsening process while sections 3 and 4 shows the process employing coarsening homogeneity. The diagram assumes the level constraint values are such that segment *S1* is not allowed to be merged in section 3, and hence fairly homogenous segments are created in section 4. In *part B* the second effect is demonstrated where sections 7 and 8 show a pair of segments involved in multiple merges at the same level. This is possible if the level constraints allow segments of the cost and demand values of segment *S2*.

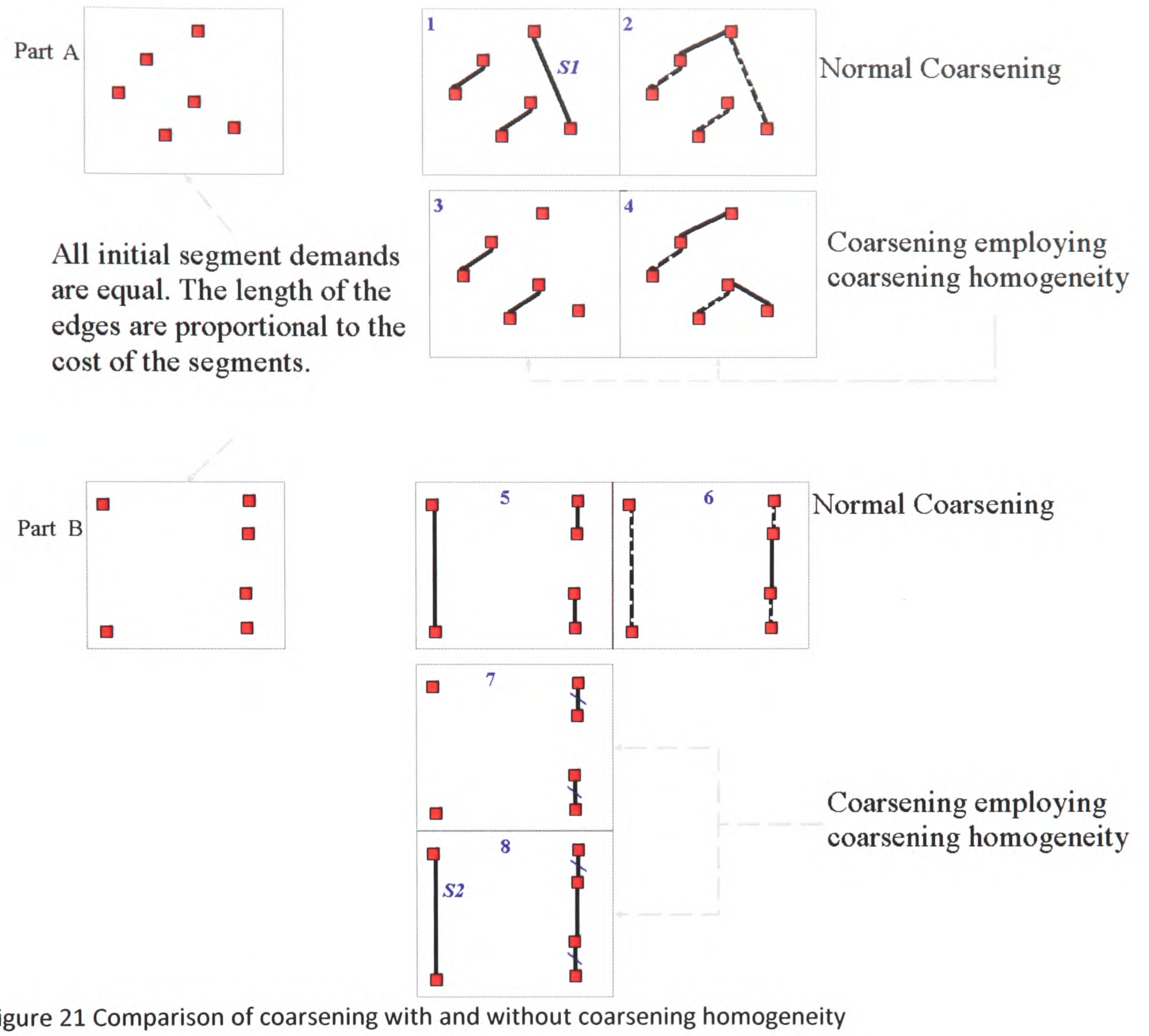

Figure 21 Comparison of coarsening with and without coarsening homogeneity

4.2.4.2.1 Implementation of Coarsening Homogeneity

Coarsening homogeneity is implemented as follows: The average demand of customers in the problem is calculated and multiplied by a *segment balancing factor (sbf)* to give the *base demand value at level zero*. The *sbf* is an experimentally established real number value of an order of one (see section 5.2.3.3).

For a given pair of segments to be matched at level *i* during coarsening, the targeted demand for the resulting segment is calculated as 1.2^i times the base demand value at level zero. The rate of change of the targeted demand value was experimentally devised as 1.2^i.

The formulation seeks to ensure that segments merged in pairs result in new segments of approximately equal demand. It can be seen that appropriate values for the heuristic are dependent on the distribution of the demands in the problem and the tightness of the capacity constraints. The targeted demand values at a given level cannot exceed the allowed problem constraints at that level. When coarsening homogeneity is employed with constraint relaxation during coarsening, the targeted level demand of segments at a level is increased at a rate of 1.3^i. This was decided after experimentation to manage the effect on the coarsening homogeneity heuristic, of relaxing the problem constraints at each level. However, the preferred combination of coarsening homogeneity and constraint relaxation, occurs when constraint relaxation is implemented purely in the refinement phase (see section 5.2.4).

When coarsening homogeneity is employed to create segments of uniform cost at each level of coarsening, the targeted level cost for segments is increased by 20% of the maximum route cost. This is done for each level of coarsening while the targeted cost is less than the maximum route cost. While segments can be matched more than once if their demand is less than the targeted level demand value, this is not applied to the cost constraint. This is because the coarsening heuristics merge segments based on their cost relative to each other, hence the homogeneity heuristic seeks primarily to balance demand considerations.

Section 5.2.5 discusses experimental results, which indicate that the use of homogeneity can be an effective enhancement, particularly when the problem instances are clustered.

4.2.4.3 Iterated Multilevel Algorithm

An iterated multilevel scheme is one in which the multilevel procedure is iterated via repeated coarsening and refinement cycles [304]. Iterated multilevel schemes are built on the concepts of solution based recoarsening (see section 3.1.4) and form a powerful enhancement to the generic multilevel paradigm. Using an iterated multilevel process, the research has been able to improve the performance of the multilevel algorithm implemented for the CVRP.

The iterated process starts with a solution constructed by the multilevel algorithm. The solution is coarsened, constructing a new hierarchy of approximations to the problem, and refined. This is repeated for a given number of iterations with the algorithm keeping a record of the best solution found

Two strategies were used for choosing the starting solution for an iteration. These were using either the current elite solution or the last generated solution. Preliminary testing revealed that the implemented multilevel framework found more improving solutions if the last generated solution was used as the starting solution for the next iteration. Using the last generated solution as the starting solution provides the framework with a means of accepting uphill moves. Uphill moves have been noted to improve the performance of heuristics [27].

In previous iterated multilevel algorithm implementations (for example the iterated algorithm applied to the TSP [304]), the coarsening phase ensures that only the edges in the existing solution are fixed in the new approximations for the problem. This creates a coarsened solution with cost equivalent to the initial solution. If the refinement process only accepts improving moves, this type of iterative algorithm guarantees not to return a solution worse in cost than the one initially held. A random element is added to the fixing of edges. This makes it likely that each iteration will give a different hierarchy of approximations to the problem and hence allows the refinement algorithm to visit different solutions in the search space [304].

The research in this thesis, however, has taken a slightly different approach to recoarsening the solutions. The savings heuristic (see section 4.2.2.1), is applied to each route in the last generated solution, creating a new initial solution. This means the grouping of segments i.e. the routes, is respected but the edges connecting the segments can be disregarded. The new coarsened solution will be similar in quality to the initial one but can be worse. Thus for each iteration, approximations of greater differences are created than would have been done had a scheme similar to the one employed for the TSP been implemented. Experimentation revealed that the strategy used in this research yields solutions of better quality. The refinement phase then proceeds as described for the multilevel implementation.

Figure 22 Iterated multilevel algorithm

```
set level counter i := 0
set problem = P_i
set number of times to iterate solution t > 0
set best solution = ∞
do
  while (P_i can be coarsened)
    P_{i+1} = coarsen(P_i)
    i := i + 1
  end
  Set initial solution S_i = P_i
  while ( i ≥ 0)
    i := i - 1
    S_temp = extend(S_{i+1})
    S_i = refine (S_temp)
  end
  if( S_i < best solution ) best solution = S_i
  set P_i = S_i
  t := t - 1
while (t > 0)
```

The iterated multilevel algorithm implemented is outlined in Figure 22. Figure 23 shows an example of the operation of the algorithm, where the coarsening phase is shown creating a new hierarchy of approximations. These approximations can then be refined by the multilevel algorithm in the standard manner.

A single-level iterated algorithm is implemented on the same principles. As a general multilevel enhancement, iterated multilevel algorithms have been used in a number of implementations, generally with great success [305]. Iterated multilevel results are discussed throughout chapter *V* and generally deliver much better solutions than the multilevel algorithm, although typically at the expense of greater runtimes.

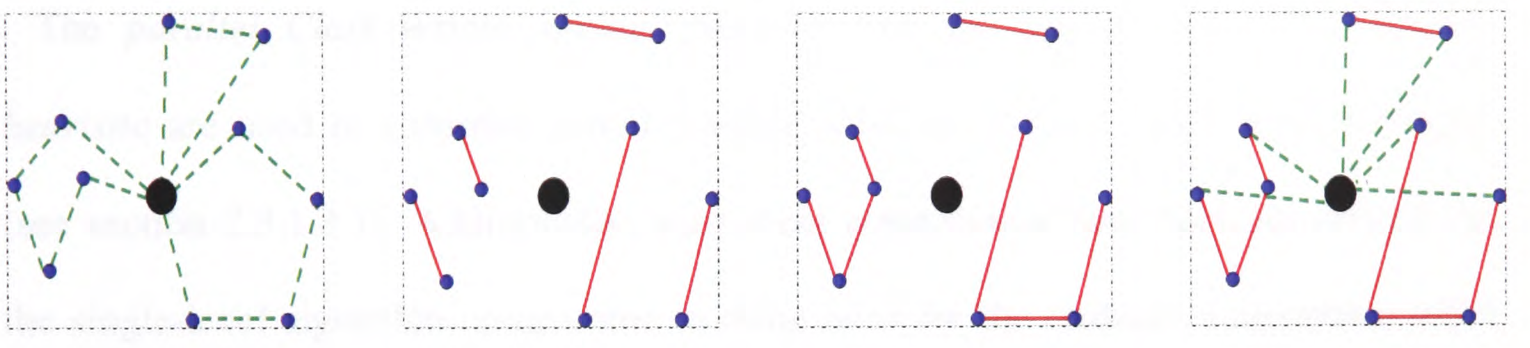

Figure 23 The iterated multilevel algorithm applied to coarsen a solution

4.3 *Part III* - The Single-level Framework for the CVRP

The single-level algorithm constructed for the CVRP provides a reference point from which to judge the effectiveness of the multilevel framework on the problem.

A composite heuristic, utilising an iterative improvement algorithm in the improvement phase, is implemented for the single-level algorithm as outlined in Figure 24. For a given instance of the CVRP, a solution is constructed and then refined using the refinement algorithm of Figure 16, section 4.2.3.

Figure 24 Single-level algorithm

set problem = P
construct initial solution S for P
while (improvement found)
  refine (S)
end

Figure 25 gives an illustration of the equivalent states during the solution process for both single-level and multilevel algorithms, when the multilevel algorithm utilises two-phase coarsening. The diagram shows that the solution construction processes for both algorithms are the same. Hence, the initial solutions to both algorithms are of the same quality. The multilevel solution process then diverges, executing coarsening and refinement in the upper-levels. This is followed by refinement on the entire problem as is done by the single-level algorithm. However, all single-level refinement is done at this stage.

The *parallel Clark-Wright savings heuristic* and the *parallel nearest neighbour heuristic* are used to construct initial feasible solutions for the single-level algorithm (see section 2.3.1.3.1). Additionally, equivalent construction heuristics are created for the single-level algorithm comparable to those used for the multilevel algorithm when coarsening is used to create the solution. Both set of heuristics, those used for the single-level and those used for the multilevel produce initial solutions of the same quality.

In the case of the multilevel algorithm, these latter heuristics, of which there are two, create a solution by merging segments at each level using either the savings or the nearest neighbour criterion to select the segments. Since the single-level algorithm does not include coarsening, the heuristics are modified such that the customers are not merged into upper level segments. The first case is a savings heuristic which assumes each customer is served by a route. An arbitrary route is chosen, and merged to another yielding the best saving. This is repeated for the other unmerged routes while there are pairs of routes as yet unmerged. The entire process is then repeated in this manner while routes can be merged respecting the problem constraints.

This heuristic differs from the serial Clark-Wright savings heuristic which selects one route and executes all feasible merges before moving to another route. It is also different from the parallel Clark-Wright savings heuristic which always executes the best feasible merge amongst all possibilities.

The second case implements a nearest neighbour version. This version progresses in the same manner as the savings version, however the merges are done on the basis of end customers on the routes being nearest neighbours.

The refinement algorithm of section 4.2.3 is then applied. Since the segments in this case are all vertices, the refinement algorithm is exactly the one applied at level zero in the multilevel case.

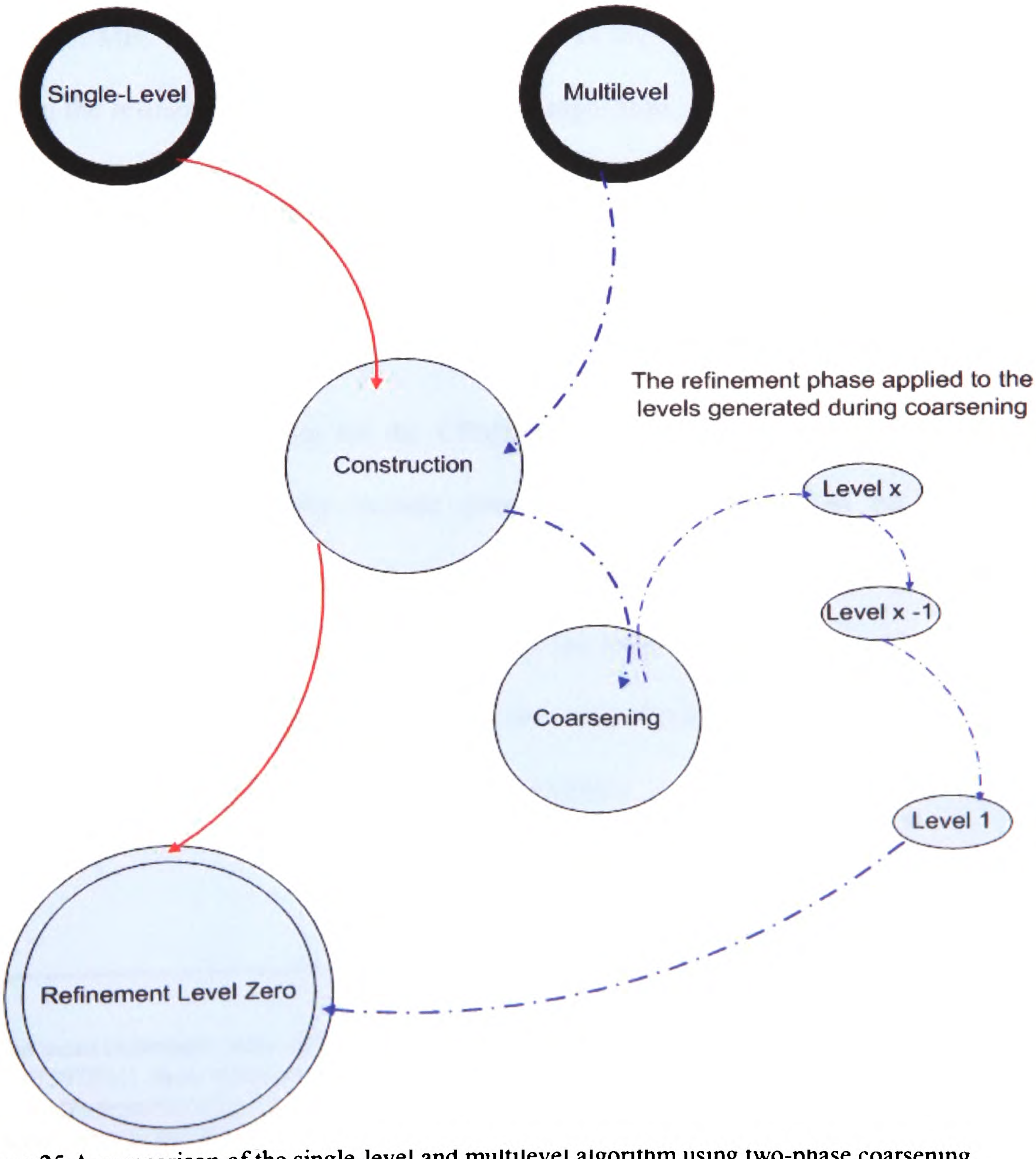

Figure 25 A comparison of the single-level and multilevel algorithm using two-phase coarsening.

4.4 *Part IV* - A Multilevel Framework for the CPMP

Similar to the approach taken for the CVRP, in designing the multilevel algorithm for the capacitated p-median problem (CPMP), leading heuristics in the field of location analysis were used to form composite methods capable of yielding high quality solutions. Multilevel versions of these composites were then created. For these multilevel algorithms, numerous coarsening approaches were devised for the CPMP.

This, part III of the chapter, starts with a general discussion of the multilevel algorithm applied to the CPMP and a description of the coarsening heuristics devised for the CPMP. This is followed by a description of the modified improvement heuristics used in the refinement phase of the multilevel algorithm.

4.4.1 General concepts of the Multilevel Framework for the CPMP

This section provides an overview of the application of the multilevel technique to the CPMP.

The multilevel algorithm for the CPMP contains similar features to the algorithm employed for the capacity vehicle routing problem (CVRP). At each level, the coarsening algorithm iteratively merges customer locations[1] to form partial clusters. The refinement process then extends and refines the initial solution, created at the end of coarsening, until an optimised solution to the original problem is obtained. As with the CVRP, further discussions require some definitions.

[1] The terms customers, users, demand points are used interchangeably in the literature [229][207][91], these terms refer to the weighted vertices of the graph and as the formal description of the CPMP states (section 2.4.2.2), the problem models many situations not involving customers. The use of these terms in this thesis is therefore understood to represent the general context of the weighted vertices of the graph.

In the context of the CPMP we redefine a *segment*[1] as a part of a cluster having a demand and a location (represented by x, y coordinates). At level zero, a segment represents a single customer. A segment in an upper level is created by merging a pair of existing segments. The new segment represents its constituting segments as a single location.

When two segments are merged one of these segments' locations is randomly chosen as the location of the new segment. The location is randomly chosen to aid the speed of the procedure. By assigning the new segment one of the original segments' locations, the heuristic ensures that, the search for median locations in all levels of the refinement stage occurs at locations corresponding to customer locations in the original problem. Since the formulation of the CPMP requires the set of median locations to be chosen from the set of customer locations, the decision to always locate the medians at customer locations means exact coarsening is used and guarantees feasible solutions throughout the refinement phase.

Using the average location (of the segments being merged) as the location of the new segment is in some sense a more intuitive approach. However, this produces an inexact coarsening process. Preliminary testing revealed that this inexact coarsening approach was less suited for the multilevel algorithm compared to the approach implemented.

The demand of an upper level segment is equal to the total demand of all its constituting segments. As with the CVRP, the refinement algorithms treat segments of level zero and segments of the upper levels in a similar manner.

[1] We use segments here for the convenience of the terminology, as is allows a uniformed discussion of the recursive nature of the coarsening process and of the process inter cluster refinement. This is similar to the case of the CVRP.

4.4.1.1 Synopsis of the Solution Construction and Refinement

The coarsening process applied to the CPMP executes two main tasks. Firstly, it reduces the number of potential median locations available at each level. Secondly, by creating segments, it identifies the groups of customers that can be transferred together by inter-cluster refinement moves during the refinement phase. This section looks at the main stages in the coarsening process and how they influence the refinement phase.

Figure 26 shows an example of the multilevel algorithm applied to a CPMP, with the aim of locating one median. The top row of the figure shows the coarsening process from left to right. The segments are matched in pairs and one of the segments locations' prohibited from being a median location at the given level. This location is filtered from the problem. The other segment's location becomes the location of the new segment and the process repeated.

The refinement process (second row of the figure from right to left) treats the coarsest graph as an initial solution and locates a median at the only available location at that level. An iterative process then ensues, expanding the segments and revealing the median locations available at each level. From these available median locations, optimisation algorithms can search for a better median location than that currently held.

The third row of the figure shows the internal structure of the upper level segments. The spherical dashed lines demarcate the constituting segments. The customers locations within these spheres are transferred together in improving inter cluster moves. The dashed lines connecting customer locations to the median are the *edges* in the problem. Edges in the multilevel algorithm for the CPMP, refer to the cost between the medians and customer locations.

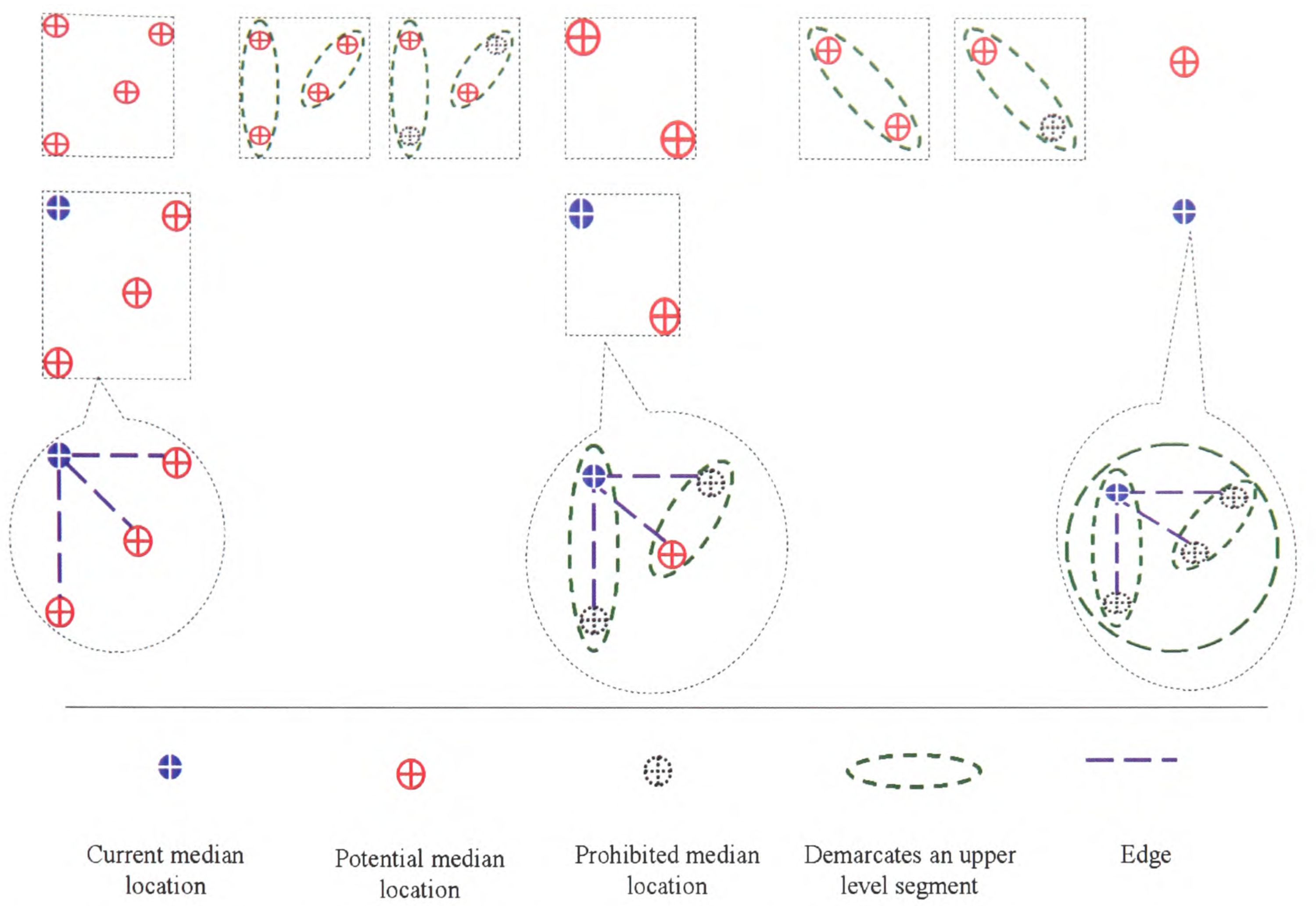

Figure 26 Coarsening and refinement process applied to a group of CPMP nodes.

4.4.2 Coarsening for the CPMP

Two types of coarsening are implemented for the CPMP. The first type uses coarsening to construct the initial solution while the second type uses a two-phase coarsening approach. The following sub-sections describe both approaches. Additionally, both approaches are evaluated experimentally and the results are discussed in section 5.3.3.

4.4.2.1 Coarsening used to create the initial solution for the CPMP

In the vein of previous multilevel implementations [304], and as was done for the CVRP, the more typical way of using coarsening in the multilevel framework, that of coarsening the problem to construct an initial solution, was implemented for the CPMP. However, this approach to coarsening faced a unique set of challenges not found in the case of the CVRP. This resulted from the fact that the CPMP is capacitated and the number of medians predetermined. Hence, as the problem is coarsened and the nature of the underlying bin-packing problem changed, it became increasingly difficult to guarantee feasibility. This section describes the coarsening heuristic implemented and the challenges faced due to the nature of the problem.

Coarsening heuristic: At each level, segments are matched in pairs and a new segment is created to replace each pair of matched segments. This continues while there are pairs of unmatched segments at the current level that can be used to create new segments that respect the constraints. The new segments plus segments that could not be matched are included in the next level and the process repeated, until the problem is represented by p segments. This stopping condition is chosen, since p clusters are required to be served by p medians with each median belonging to exactly one cluster. These p segments therefore become the initial solution that is passed to the refinement phase.

Because the problem is capacity constrained and the number of medians is predetermined, the segments chosen to be merged into a new segment are not chosen on the basis of cost. Instead, the segments are ordered by decreasing demands at each level and consecutive pairs are merged, starting with the two segments of largest demands. The solution produced by the heuristic may then be an infeasible one, i.e. there may exist more than p medians. In this case feasibility is enforced during the refinement phases using the inter-cluster heuristics. However merging the segments on the basis of demand as opposed to cost, means the coarsening heuristic is in the mould of "*first-fit-decreasing*" heuristics [33] and reduces the possibility of constructing infeasible solutions.

4.4.2.2 Two-phase coarsening for the CPMP.

Two-phase coarsening for the CPMP offered two advantages. Firstly, using two-phase coarsening, the challenges faced using the coarsening approach of section 4.2.2.1 were not encountered to the same degree, due to the separation of construction and coarsening. Secondly, two-phase coarsening offered the advantage of using the leading construction heuristic(s) in the field as part of the multilevel solution approach.

When two-phase coarsening is employed, an initial solution is created and then coarsened while respecting the clusters. For the creation of initial solutions, two construction heuristics were implemented for the CPMP.

The first of these, termed the *grouping heuristic*, is modelled on the three-phase heuristic proposed by Osman and Christofides [207], and modified by Franca et al. [91].

The grouping heuristic commences by selecting two initial median locations, these being the locations of the two customers farthest apart. If the number of medians p is equal to two, the heuristic terminates. If however p is greater than two, additional medians are chosen until $p-1$ medians are obtained, such that each new median *maximizes* the product of the distances between itself and all previously located medians. The last median is then chosen, satisfying p, such that the product of the distance between itself and all previously located medians is *minimised* [91].

In the second phase, the customers are assigned to medians in increasing order of a calculated *quotient*, while the capacity constraints can be respected. The quotients are calculated by dividing the distance between each customer and each median by the customer's demand. If an assignment of a customer to a median is prevented due to the capacity constraint, the affected customer is assigned to the next available median. In this case the medians are ordered by the increasing quotient value, relative to the affected customer. The third phase recalculates the median of each cluster at the end of all assignments. If a new set of medians is found they become the initial medians, and the second and third phase repeated until a stable set of medians emerges.

The criteria used for locating the last median in the first phase and for assigning the customers are two of the modifications proposed by Franca et al.[91], and adopted in this work. They are implemented as a means of minimising the possibility that, in the assignment phase, the demand constraints will lead to infeasible clusters. These infeasible clusters could arise depending on the "*tightness of the capacity constraints*" [207] and the demand distribution. The tightness of the capacity constraints is a ratio of the total capacity in the problem (p * capacity allowed per median) over the total demand in the problem. As this ratio decreases, the possibility of obtaining infeasible initial solutions increases. For a given tightness of the capacity constraints, ensuring a 'good[1]' spread of initial medians and taking the customers' demands into account during the assignment phase reduces the chances of infeasibility arising.

Since the heuristic does not update the median location until all assignments have been made it has the effect of *grouping* the customers of larger demands around the chosen median locations. This is then followed by assigning the customers of smaller demands to their nearest medians as the capacity constraints allow.

The second construction heuristic implemented, termed the *greedy heuristic*, selects p initial medians randomly, as investigated by [188]. The customers are assigned by increasing cost from their nearest available median, as described by Osman and

[1] A good spread of the medians i.e. distributing the median location so that they are evenly distributed throughout the problem space; can aid in reducing infeasibility. The heuristic seeks to assigned customers to the nearest median. Hence, if all the medians are located near to each other and most the customers are located in the near proximity to the medians; the assignments phase would assign the outlying customers last. The demand on the medians and the capacity constraints might prevent their assignments at that point.

Christofides [207]. However, we have modified the heuristic, such that after each insertion the median locations are updated for the affected clusters, producing a more '*greedy*' heuristic than the first. Where capacity overflow occurs, the customers in the affected cluster are reassigned to the cluster by decreasing order of demand, with the unassigned customer(s), assigned to their nearest available median in increasing order of cost.

4.4.2.2.1 Coarsening the clusters.

Since the construction of the solution guarantees each cluster to be feasible, the coarsening algorithm is able to merge segments based on cost. The coarsening algorithm is applied to each cluster in the solution in turn, calculating the cost[1] between all unique pairs of segments in the cluster. The segments are merged in pairs starting with the pair of closest segments (based on the cost just calculated), then the next pair of closest segments, and so on, while there are pairs of unmerged segments at the current level. Segments are merged once at a given level. The new segments are then included in the next level along with any unmerged segment and the process repeated until the cluster is represented by one segment.

4.4.2.3 Concluding Analysis of Coarsening for the CPMP

The CPMP has been a more difficult problem to coarsen than the CVRP. This difficulty arises from two sources, the nature of the problem and the manner in which solutions to the problem change during refinement.

[1] Each segment is assigned one unique location. The cost between two segments is calculated between their unique locations.

The issues relating to the problem being capacitated and the number of medians predetermined, has been handled both from the viewpoint of coarsening the problem based on the demands of the segments and utilising two-phase coarsening. The results (see section 5.3.3) demonstrate the effectiveness of the heuristics in this regard.

Since, the cost of a cluster is calculated from a single median, each transfer of a segment to/from a cluster can require the cost of the entire cluster to be recalculated. This is different from the case of the CVRP where, adding a new segment to a route, requires one edge to be removed and two edges added. The new cost of the route, assuming the cost before the move was known, can be calculated by using the old cost and the difference in cost of the three mentioned edges. Figure 27 gives an illustration of the difference; from left the figure shows a cluster before and after the addition of a new segment. The equivalent is shown for a route on the right of the figure. The new edges added to each are shown in solid lines, from which can be seen the potentially large changes to the cluster.

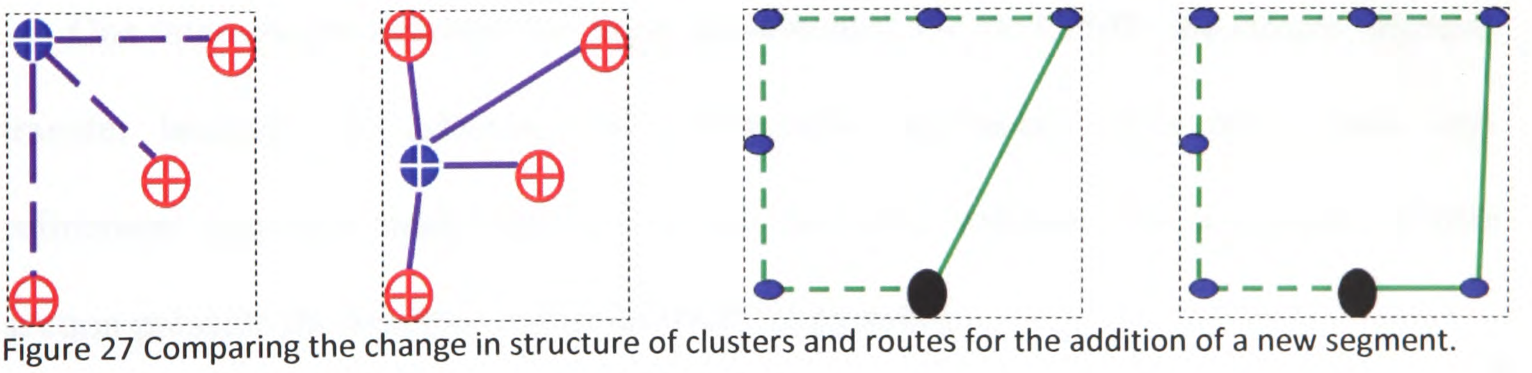

Figure 27 Comparing the change in structure of clusters and routes for the addition of a new segment.

The fact that the CPMP solution changes in this way during the refinement process affects the coarsening process, because one of the aims of the coarsening heuristics deployed in multilevel refinement, is to filter detail from the problem at each level. This is achieved for the CPMP in an important context in that the coarsening heuristics reduce the number of median locations at each level.

However, in the refinement phase, cost calculations require location information about the customers. Thus, from the point of view of reducing details about the customers available at each level, the customer's location information is not filtered from the problem in the coarsening phase. However, preliminary tests were done on filtering the customer details from upper level segments then using approximate cost calculations. These approaches proved inferior to the approaches presented here.

4.4.3 Refinement for the CPMP

The refinement process uses inter- and intra-cluster optimisation at each level. Intra-cluster optimisation calculates, for each cluster, the segment location (median) within that cluster, from which the sum of the cost to all other segments in the cluster is minimised. The median locations are chosen from the segment locations available at the given level. The complexity of the procedure is $O(n^2)$ [59],[282] where n is the number of segment locations available in the solution. In the refinement algorithms to follow this is referred to as *compute median locations.*

One inter-cluster heuristic has been implemented for the CPMP, the simple segment transfer heuristic. In addition, two refinement approaches have been used, one refinement approach using tabu search and the other without. The remainder of this section presents the heuristic and refinement approaches.

4.4.3.1 Simple segment transfers.

The simple segment transfer is a inter-cluster heuristic (section 4.2.3.4) [206],[207],[91] that defines two move types for generating neighbourhoods, namely *transfer* and *interchange*. Unlike the case of the CVRP, where routes can be viewed as an ordered set, clusters for the CPMP are unordered sets. This mean there is no requirement to consider the position of segments in the clusters when implementing transfers or interchanges.

Transfer moves consider the insertion of segment(s) from one cluster into another. Interchange moves consider the exchange of segment(s) between pairs of clusters i.e. segments from one cluster are transferred to another cluster from which a different set of segments is returned. All segments in the clusters are considered for transfer or interchange and a parameter, λ, specifies how many segments can be removed or added to a cluster at once.

An arbitrary ordering is defined on all the clusters in a solution S, and the heuristic then sequentially searches all unique pairs of clusters in S. The search is conducted first for improving interchange moves, and then repeated for improving transfer moves. A first improvement strategy is used. Since the heuristic searches the clusters in unique pairs, the size of the search neighbourhood is determined by the number of clusters in S and the value of λ. The solution S, contain c clusters and hence $c\ (c\ \text{-}1)/2$ unique pairs of clusters. After preliminary experimentations, λ was set to one, to reduce the size of the search neighbourhood. λ equal to one is effective for this heuristic in the multilevel framework since, in the upper levels of refinement segments represent groups of customers.

4.4.3.1.1 Edges as they relate to the moves of simple segments transfers.

In the case of the CPMP edges relate to the cost between segments and median locations. However, not all of these edges are treated in the same manner. A tabu search mechanism is used in one of the refinement approaches (see section 4.4.3.3) and the tabu-ed attributes are selected edges in the solutions. Therefore, the edges linking or that linked segments directly transferred by the simple segments transfer heuristic are treated differently to edges that link other segments in the clusters changed by the moves performed by the heuristic.

The former set of edges are said to be recorded by the heuristic, i.e. they are used to store moves implemented, while the latter are only used in the cost calculations assessing the feasibility of those moves. This section discusses how these edges are handled in the refinement phase.

Figure 28 shows a proposed transfer, relocating segment *S1* from cluster 1 to cluster 2. If edge *E2* were of a lower cost than edge *E1*, the transfer would be accepted. Failing that, Figure 29 and Figure 30 show the two other combinations that would be assessed. These combinations are to determine if a cluster could be formed, including segments of cluster 2 and segment *S1*, of cost less than that of the original two clusters.

These series of diagrams highlight one of the main features of the multilevel algorithm for the CPMP. The number of cost calculations is reduced by the number of median locations available at any given level. They also show that, in analysing the validity of a proposed move, if the move is not immediately improving (based on the edges removed and added to the solution connecting the segment (s) of the move), then the medians of the affected clusters have to be updated and the move reanalysed (as exemplified by Figure 29 and Figure 30).

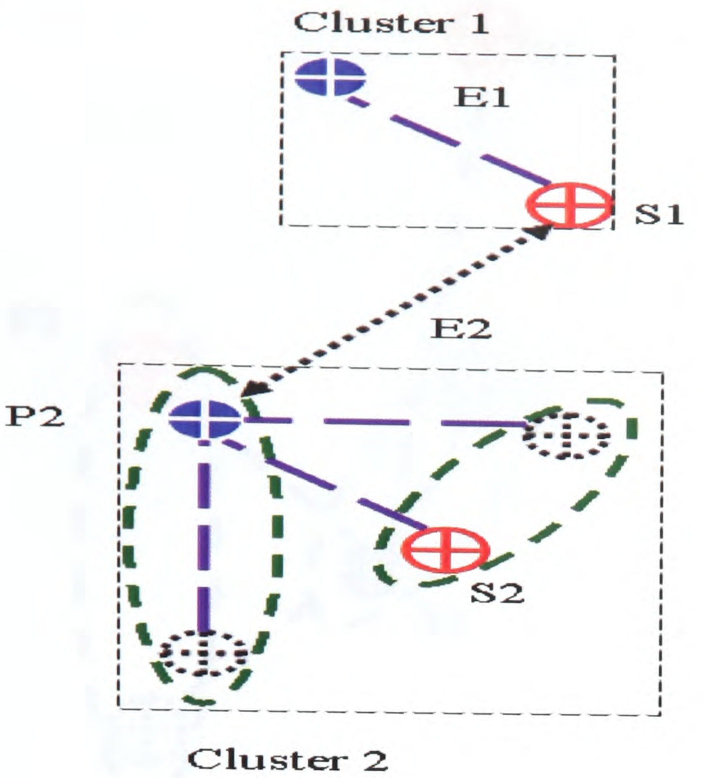

Figure 28 Transfer move executed by the simple segments transfer heuristic.

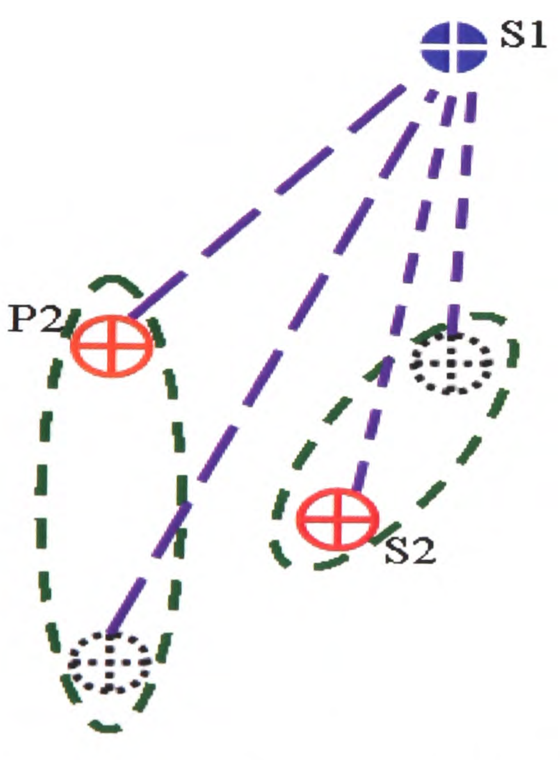

Figure 29 Alternative configuration to the transfer move of figure 26.

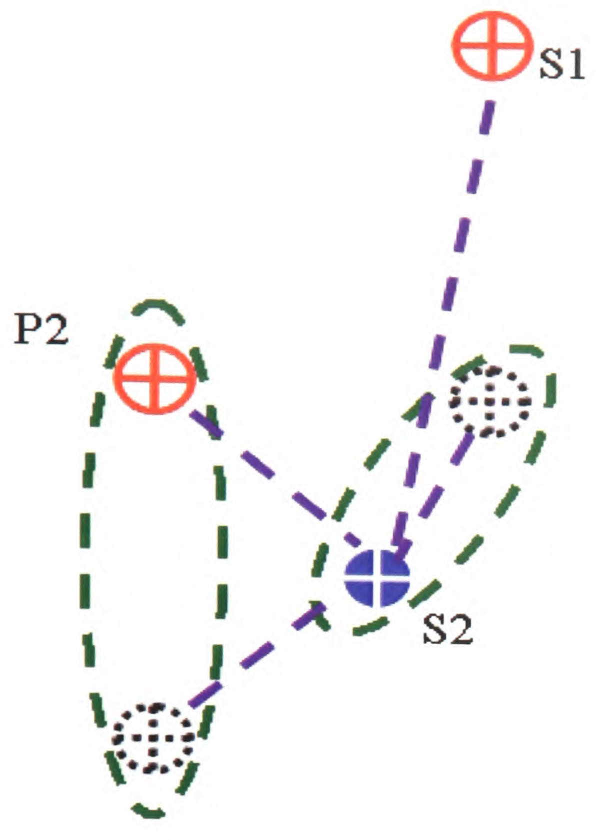

Figure 30 Alternative configuration to the transfer move of figure 26 and figure 27.

The number of edges involved in the cost calculations for a transfer or interchange move is equal to two times the number of customer locations contained in the segments transferred, in the case where the medians of both clusters are the same before and after the moves. However, since a segment is assigned one location, in recording a move, it is sufficient to record the edges in the move connecting the location of the segment(s) of the move to the relevant medians. In the case of a transfer move, an example of which is shown in Figure 31[1] transferring segment *S1* from cluster one to two, there are two such edges recorded. These are labeled *E3* and *E4* in the figure. For interchange moves, there are four such edges. These edges are used to record moves when the tabu search algorithm is utilised (see section 4.4.3.3).

[1] The same key applies for Figure 28 to Figure 31 where it is displayed.

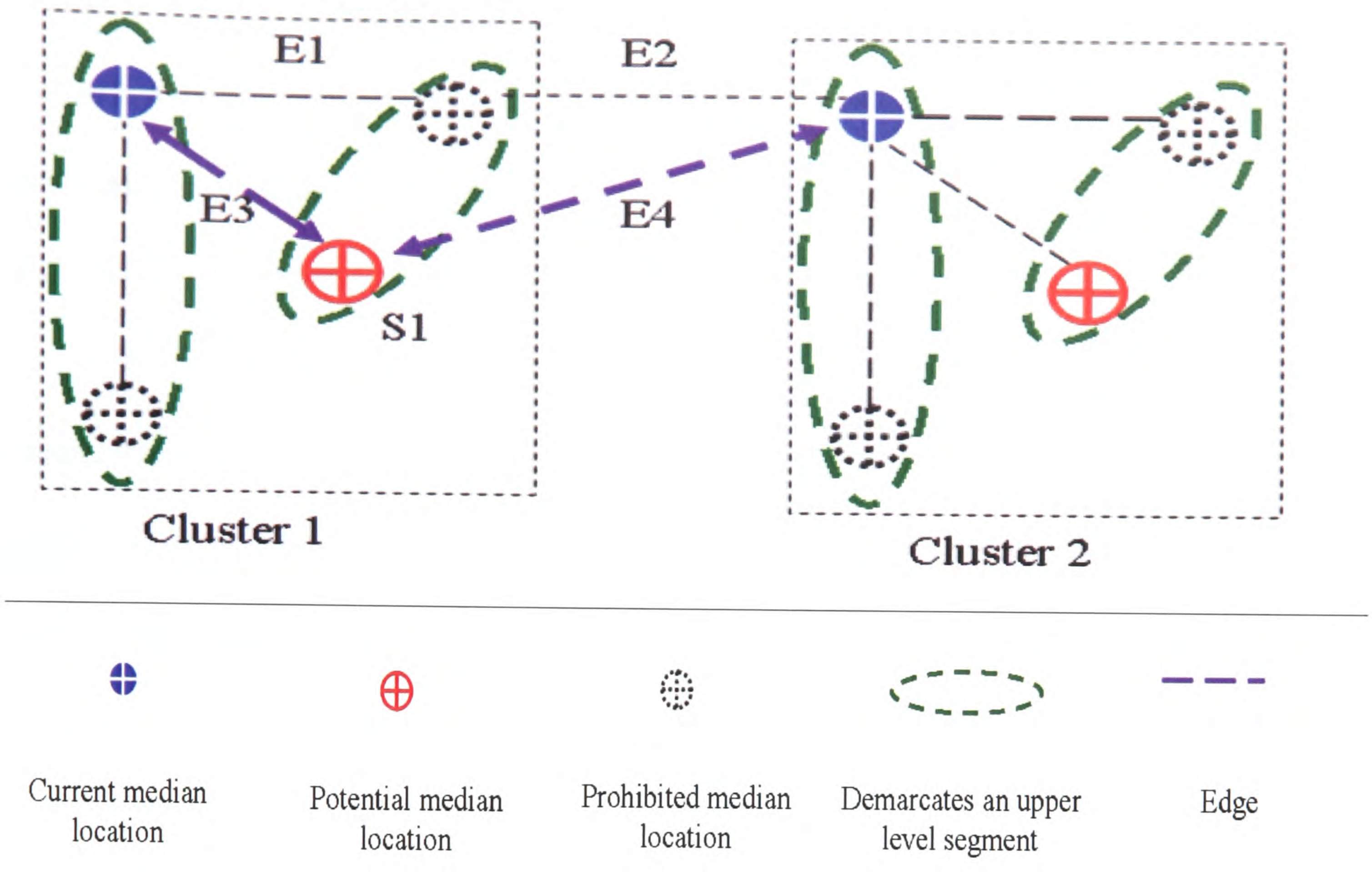

Figure 31 Transfer move for an upper level segment executed by the simple segments transfer heuristic.

When a move changes the median of one or both of the affected clusters (example Figure 29), there are more involved edges than the two or four described above. In clusters where the median has changed, the edges linking segments that have remained in the same cluster[1] are needed for assessing the cost of the moves. However, these edges are not recorded as a part of the move as is done for the edges linking the segment(s) that have changed clusters. These edges are not recorded to limit the area of the search space tabu-ed when the tabu search mechanism is deployed.

4.4.3.1.2 Assessing the cost of moves for simple segments transfers.

The following describes the process of assessing the impact on the solution of moves performed by the simple segment transfer heuristic.

[1] We refer to the customers remaining in the same cluster if they were not actively transferred by the heuristic.

Let $S = \{1, ..., n\}$ index the set of clusters forming a solution. Given two clusters C_i, $C_j \in S, \forall i, j \in S \;\; i \neq j$ let P_i be the median of C_i from which the cost of C_i is minimised and P_j be the median of C_j from which C_j's cost is minimised. V_i is a member segment of C_i and V_j is a member segment of C_j.

Without loss of generality the following assumes all the moves attempted respect the capacity constraint and further, the criteria to be outlined for improving moves holds $\forall V_i \in C_i$ and $\forall C_i \in S$. An improving move is a transfer or interchange that reduces the solution cost of S.

If a transfer move attempts to relocate segment V_i from C_i to C_j and the location of V_i is different to that of P_i, the move is an improving move, if the cost of connecting V_i to P_j minus the cost of connecting V_i to P_i is less than zero. However, if this cost is not less than zero, or the location of V_i is the same as that of P_i, in order to determine if the move is an improving move, requires the identification of the median $P_y \in C_j$ that minimises the cost of C_j with V_i added to the cluster. Secondly, the median $P_x \in C_i$ that minimises the cost of C_i with V_i removed from the cluster has to be identified. An improving move is then found if the cost of the new clusters are less than the cost of the old clusters.

If an *interchange* move is attempted for $V_i \in C_i$ and $V_j \in C_j$, and V_i and V_j 's locations are different from those of P_i and P_j, an improving move is found if the following relation applies: *cost* V_i *to* P_j + *cost* V_j *to* P_i < *the cost* V_i *to* P_i + *cost* V_j *to* P_j. If however, this cost is not less than zero, or one of the segments being interchanged shares a location with P_i or P_j, the effect of the moves across both clusters has to be determined. This requires the identification of median $P_x \in C_i$ that minimises the cost of C_i with V_i removed and V_j added to the cluster. It also requires the identification of median $P_y \in C_j$ that minimises the cost of C_j with V_i added to the cluster and V_j removed from the cluster. Again, an improving move is then found if the costs of the new clusters are less than the costs of the old clusters.

4.4.3.2 Refinement strategies for the CPMP

Two refinement strategies have been implemented for the CPMP. The first, termed *simple search*, outlined in Figure 32 iteratively expands and then refines the clusters at each level using the simple segment transfer heuristic. When an improving move is performed, the affected clusters are optimised[1], determining the best median locations within those clusters at the given level.

The second strategy guides the refinement process using a *tabu search* heuristic. In relation to the generic multilevel algorithm (Figure 3, section 3.1.1) each call to refine() in that algorithm, executes one of these refinement strategies. Both strategies are experimentally evaluated and reported on (see section 5.3.3.2).

Figure 32 The simple search refinement algorithm executed at each level

```
expand clusters
  do
       simple segments transfers
       If improvement compute median locations
  while( improvement found)
```

4.4.3.3 Tabu Search Refinement Strategies

The multilevel algorithm using tabu search refinement is outlined in Figure 33. Similar to the algorithm of Figure 32, simple segment transfers are used to iteratively refine the solutions at each level, however a tabu search mechanism is added which allows the acceptance of non improving moves and the rejection of tabu-ed moves. This

[1] Using the *compute median locations* algorithm

guides the refinement process to areas of the search space inaccessible to the simple search algorithm. The tabu search concepts of the algorithm of Figure 33 are modelled on the work of France et al. [91] but the algorithm has been modified for the multilevel framework.

Synopsis of the tabu search mechanism: A tabu list is defined at the start of the refinement process and a given number of the edges that are part of moves performed during the refinement phase are added to the tabu list (*tabu-ed edges*) for a stated number of iterations [91]. The solution is then prevented from visiting solutions containing a given number of tabu-ed edges (where this occurs the heuristic has encountered a tabu-ed move) [91]. If no improving move was found during the last iteration, the least non-improving move, from the list of potential moves that has not been tabu-ed, is accepted. These non-improving moves are allowed for a stated number of iterations [91].

The simple segment transfer heuristic is modified to include an added step to check the tabu state of moves and to tabu those moves that are performed.

Figure 33 Tabu search refinement algorithm executed at each level

```
set iteration counter x := 0
set maximum number of iterations  tni := an integer value.
     expand clusters
     do
          do
               simple segments transfer
               If improvement found compute median locations
          while( improvement found)
          store best solution found
          if( no improvement found)
               accept least non improving move.
               compute median locations
               x := x + 1
     while( x < tni) // each time though this loop is one iteration
```

4.4.3.3.1 Analysis of the Tabu search mechanism

At each level, an experimentally devised *tabu range* is defined. The tabu range states the lower and upper bounds for the number of iterations that move attributes can be tabu-ed and is devised based on the number of segments at a given level. A suitable range allows the tabu-ed attributes to be maintained in their tabu-ed state, sufficiently long for the algorithm to explore areas of the search space different from the areas the tabu moves have been performed in, but not so long as to prevent the search from being able to analyse further moves in those areas at the current level.

Since, the number of edges involved in the cost calculations for a move is at least equal to two times the number of customers locations involved in the move, at an upper level, this could be a significant number of edges. Therefore, the heuristic does not tabu all the edges involved in a move as this would result in large sections of the search space being inaccessible for optimisation. Instead, the edges connecting the location of the segments to the affected medians are tabu-ed. Where the segments in the moves are upper level segments, the moves tabu-ed relate only to these segments and not their constituting segments. This means that a transfer move produces two tabu-ed edges while an interchange move produces four tabu-ed edges. Each tabu-ed edge is assigned a random value chosen between the tabu range and corresponds to the number of iterations for which the edge is to be stored on the tabu list. The edges stored in the tabu list are then propagated through all the levels of refinement.

The storing of attributes as opposed to entire solutions has been recognized as aiding the efficiency of tabu search algorithms [256]. In this heuristic the tabu-ed attributes are edges and each edge records the segments transferred or interchanged in moves that have been performed. Where these segments are upper level segments, while the edges are tabu-ed and the segments have not been extended, the relevant moves are tabu-ed. If however, the segments are extended before the edges are removed from the tabu-ed list, the previously tabu-ed moves are no longer tabu-ed since the segments they relate to are no longer part of the solution.

Intensification and diversification of the search process is driven by: the tabu range; the acceptance of uphill and downhill moves; and the controlling of when moves are tabu-ed. An integer tolerance parameter [91] is defined stating the maximum number of tabu-ed edges allowed in a move. Moves containing a number of tabu-ed edges exceeding this tolerance parameter value are tabu-ed. The following section discusses how the tolerance parameter is used to control when moves are tabu-ed.

4.4.3.3.1.1 *The effect of the tolerance parameter*

For transfer moves, a tolerance parameter value between zero and two was chosen and for interchange moves, a value between zero and four was chosen. If the tolerance parameter is set to zero, no moves involving tabu-ed edges are permitted; if it is set to the maximum value for either range all moves of that particular type are possible. Franca et al. [91] outlines that, setting the values to zero, is likely to result in diversification "*as most active moves in the current neighbourhood will be forbidden*" while setting the value to four for interchange or two for a transfer "*should promote intensification*". Values in between will effect a trade-off between the intensification and diversification processes.

The diagram in Figure 34 uses a series of three interchange moves to analyse the effect of varying the tolerance parameter. Assuming all three moves are performed during the same iteration, the values selected from the tabu range for all edges are greater than zero and the original state of the solution is depicted in screen one. The tabu-ed edges at the end of each move are shown on the tabu list. Each edge is represented by listing the two segments it connected, in the move of which it was a part.

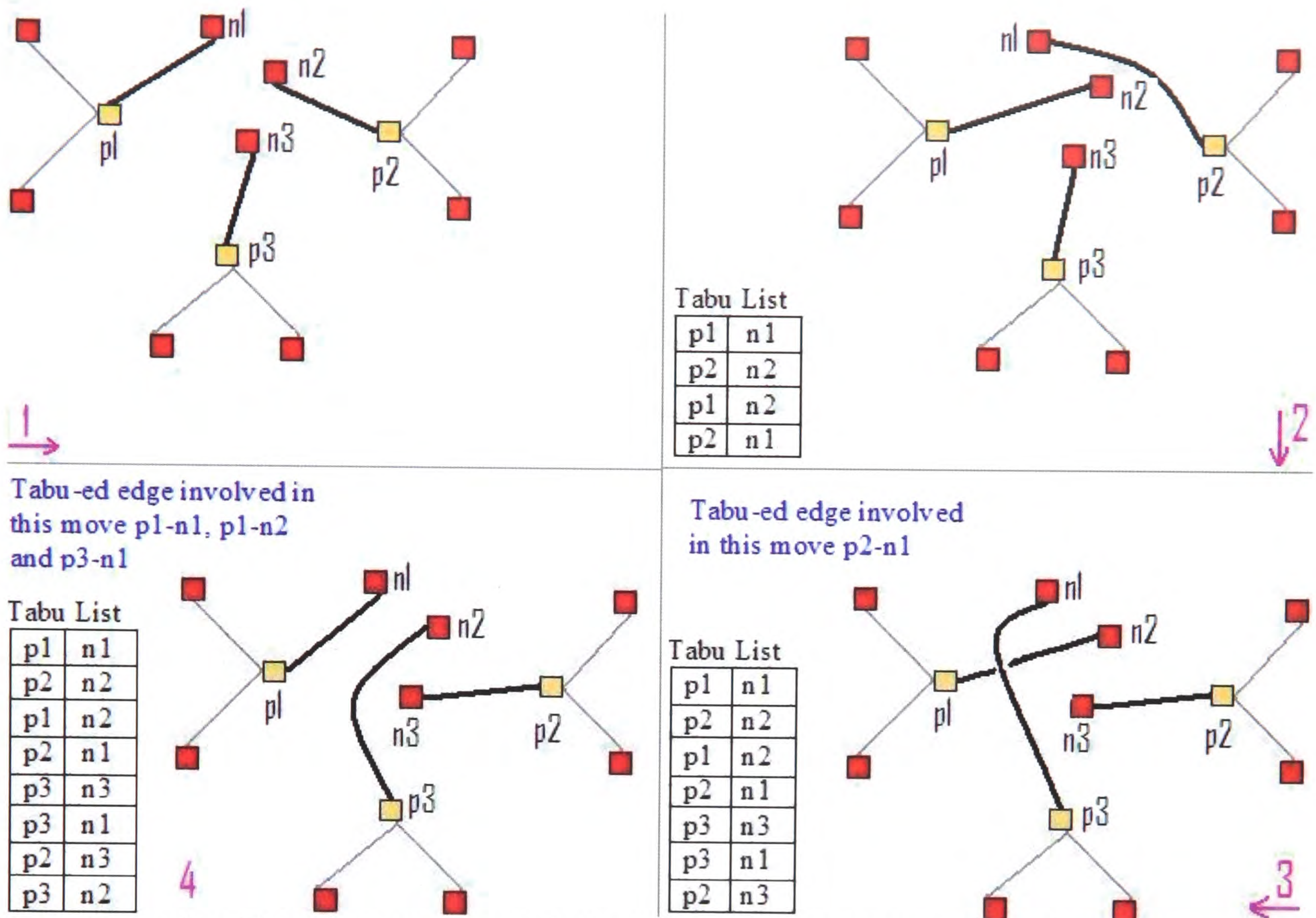

Figure 34 A series of interchange moves, analysing the effect of the tolerance parameter.

The interchange move transferring the solution from screen one to that shown in screen two is permissible irrespective of the tolerance parameter's value. The second interchange move, the result of which is shown in screen three, involves one tabu-ed edge (p2 – n1). Hence, if the tolerance parameter value were set at zero this move would be prevented. Edge p2 – n1 would be a tabu-ed edge in the second move, as it had previously been involved in the first move. The interchange move resulting in the solution shown in screen four involves three tabu-ed edges and consequently would require a tolerance parameter of three or greater to be implemented.

4.4.3.4 A comparison of the two refinement strategies

At each level, the simple segment transfer heuristic attempts to relocate each segment in the solution. When used with the simple search algorithm, since there is no history of moves made at previous levels, computational resources could potentially be wasted. This can occur when expensive cost calculations are made, in attempting to transfer or interchange segments that had previously been exchanged as a part of an improving move, and the move currently being attempted would return their respective clusters to the states they were in before the improving moves were performed. This type of move will ultimately not be implemented once the calculations reveal it is not an improving move. The tabu search algorithm potentially reduces the waste of computational resources due to these types of calculations. The edges involved in each new move are checked against the tabu list. If the tolerance parameter is not set to its maximum value this type of move just described would be marked as tabu-ed and the cost calculations that would have been performed in assessing the move would not be performed.

The experimental analysis of both strategies (section 5.3.3.2) shows the advantages the tabu search algorithm offers over the simple search approach.

4.4.4 Iterated Multilevel Algorithm for the CPMP

An iterated multilevel algorithm is implemented as outlined in Figure 22 (p. 109). The last generated solution is recoarsened respecting the clusters in place and refined. The coarsening algorithm applied is the same as described for the coarsening phase of two-phase coarsening (see section 4.4.2.2). For the tabu search multilevel algorithm the tabu list is propagated through all iterations, this has the effect of improving the use of memory of the search process. It may be possible to increase this effect by increasing the tabu range and allowing the tabu attributes to be tabu-ed for more iterations.

As demonstrated by the experimental results (see section 5.3.3.3) the iterated multilevel algorithm significantly improves the quality of the solutions obtained from the multilevel algorithm, regardless of the refinement approach used.

4.5 *Part V* - The CPMP Single-level Algorithm

As was the case for the CVRP, the single-level algorithm constructed for the CPMP provided a reference point from which to judge the effectiveness of the multilevel framework on the CPMP.

The single-level algorithm for the CPMP is outlined in Figure 24 (p. 110). The construction phase follows the solution construction process outlined in section 4.4.2.2, and both refinement algorithms (see section 4.4.3.2) are implemented.

The experimental testing of the single-level algorithm (section 5.3.3.3) shows the algorithm implemented to be an effective solution technique for the CPMP.

Chapter V

5 Computational Results for the CVRP and CPMP

5.1 Introduction

This chapter presents the results of the experimentation done to determine whether the multilevel technique could aid the solution process for routing and location problems. The heuristics implemented throughout the research are tested using standard benchmarking suites from the literature and results presented both for the capacity vehicle routing problem (CVRP) and the capacitated p-median problem (CPMP). The results demonstrate that the multilevel technique can aid the solution process for routing and location problems. Additionally, through testing done across both problem types, it is shown that two-phase coarsening can significantly aid the performance of multilevel algorithms.

The results demonstrate that when the multilevel technique is coupled with a tabu search mechanism it is not only able to significantly outperform its single-level counterpart (a feat it also achieves without the tabu search mechanism), but it is also able to produce results comparable with the leading solution techniques in the field. Throughout the chapter, it is also demonstrated that the iterated multilevel algorithms produce results comparable to the standard in the literature for both routing and location problems.

The chapter is divided into two parts. *Part I* is devoted to the results for the CVRP. This part starts by laying out the suites of problem instances used for experimentation and the conditions under which those experiments were conducted. This is followed by results justifying the algorithmic configuration and parameter settings used for the main results presented for the CVRP. The remainder of this part of the chapter analyses the results produced by the multilevel, single-level and their iterated versions for each suite of problem instances used for the CVRP.

Part II of the chapter presents the results for the CPMP and follows a similar format to that used in *Part I.*

5.1.1 Platform

The algorithms designed for the CVRP and the CPMP and described in *Chapter IV* have been implemented in Java (jdk1.5.0_09) using an object oriented design. The development and testing was done on a Pentium-4, 3GHz PC with 1 GB memory operating windows XP Professional SP2. For small to medium problems, the Java Virtual Machine (JVM) [134] is assigned 256MB of memory and up to 800 MB of memory for large and very large-scale problems.

5.2 *Part I* - Experiments for the CVRP

5.2.1 CVRP Instance Sizes

The CVRP test instances in the literature can be classified by size as small-medium, large and very large instances. Small-medium problems range in sizes from 20 customers to 200 customers. Classic representatives of the class of small-medium problems are the instances of Christofides and Eilon [48] and Christofides et al. [49]. Large CVRPs range in sizes from 200 customers to 500 customers and the test suite proposed by Golden et al [122] is a classic representative of this class of problem size. Very large-scale instances are still emerging in the literature. The size of these problems typically range from around 500 to 20000 customers. Li et al. [170] and Irnich et al. [147] provide large-scale instances of varying sizes. The grouping of problems according to these sizes may evolve in response to more powerful computers and better algorithms, but this classification is typical of the current field. In their recent works, Kytöjoki et al. [163] and Mester and Bräysy [180] use a similar classification.

5.2.2 CVRP Instance Types

The algorithms have been tested extensively on a number of standard suites of problem instances used by other authors for benchmarking VRP algorithms. This section introduces the five suites of problem instances used.

The first set contains the problems proposed by Christofides et al. [49]. Of this set, instances 1 – 5 are CVRPs, instances 6 – 10 and 13 & 14 are distance-constrained VRPs (DVRPs) and in instances 11 – 14 the nodes are clustered. When we refer to DVRPs we refer only to instances 6 – 10, when we refer to clustered problems we refer to instances 11 - 14. In addition to looking at the performance of the heuristics across the entire test suite, the heuristics' performance are analysed across the different problem groupings just identified. The problem instances in this test suite range in size from 50 – 199 customers.

The second suite of CVRP instances used for testing contains the instances provided by Christofides and Eilon [48]. The instances in this test suite range in sizes from 20 – 100 customers and are all CVRPs.

The test suite, provided by Golden et al. [122], contains 20 classical large-scale problem instances ranging in size from 200 to 483 customers. The first 8 instances are distance-constrained with zero service times and the remaining problems are CVRPs. These instances are symmetrical, with the customers located in concentric circles, squares, or a star formation. The depot is located in the centre of the customers or in one corner of the Euclidean plane. This is the third suite of instances used. Internet addresses are provided in [296] and [299] for the above three sets of test cases.

The fourth suite used is that of Li et al. [170] and the fifth that of Irnich et al. [147]. The suite of Li et al. [170] contains 12 very large-scale instances ranging in size from 560 customers to 1200 customers. These instances are symmetrical as the customers are located in concentric circles around the depot. An internet address is provided in [297] for these instances.

Irnich et al. [147] provides a diverse test suite of large scale to very large-scale instances grouped in series. They are, however, not much used in the literature. Five series are used for testing (series 1 – 5). Among these series, the customers are uniformly distributed from $[-100, +100]^2$ with their coordinates represented by integers. The demand distribution is different for each series (section 5.2.12.1). Within each series, the capacity constraint is varied across a number of problem sizes. An internet address is provided in [298] for these instances.

5.2.2.1 Computational standard CVRP

All computations for the CVRP are done with double precision real numbers and the provided solution values are rounded up to two decimal places. The runtimes provided include the input and processing times. The processing time is measured to the termination of the process at the end of level zero in the refinement process.

5.2.2.2 Best Known Solution value (bks) for the CVRP

When solution costs for individual problems are given, where possible, they are represented as a percentage above the *bks* for the problem. The use of the *bks* in this way provides a standard way of rating the current quality of a particular solution. Since, most of the *bks* are not proven optimal, the rating given to solutions in this way is potentially a moving criterion (one of the many difficulties of analysing heuristics [13]).

It is also worth highlighting that across all the test suites the *bks* are produced by many different methods. Section 1 (p. 232) provides a detailed listing of the *bks* presented in the following results.

5.2.3 Parameter Tuning for the CVRP

As noted by Cordeau et al. [55], the proliferation of parameters in heuristic design results in algorithms that are difficult to use. In this section, the main parameters used are justified and experiments carried out in order to determine suitable values are presented.

We have kept the parameters used in the multilevel algorithm for the CVRP to a core set of four. These are:

- The *segment balancing factor (sbf)* which governs the targeted demand of segments created at each level of coarsening. The *sbf* is based on the average demand of customers in the problem and is used in the cases where coarsening homogeneity is employed (see section 4.2.4.2).
- The *capacity overload factor* (*cof*). When constraint relaxation is employed (see section 4.2.4.1), the allowed capacity for a route is determined at each level. The allowed capacity cannot exceed a maximum limit. This maximum limit, is given by the *cof* times the stated capacity of the problem.
- The *cycle depth*, used in the cases where the cyclic segment transfer heuristic is employed (see section 4.2.3.5). The cycle depth states the maximum number of routes allowed in cycles formed by the heuristic.
- The λ-interchange parameter (λ). λ states the maximum number of segments transferable between pairs of routes in a solution when the simple segment transfer heuristic is employed (see section 4.2.3.4). Preliminary testing for λ

showed that the runtime of the simple segment transfer heuristic increased for increasing λ values. However, there were no corresponding improvements in the quality of the solution. The value of λ was therefore set to one.

5.2.3.1 Justification for the cycle depth parameter range

In the case of the cyclic depth, integer values 2 to 6 inclusive were chosen. The operation of the cyclic segment transfer heuristic requires cycle depth values of 2 or greater since cycles cannot be formed consisting of less than two routes. The cyclic depth plays a key role in determining the runtime of the cyclic segment transfer heuristic, with increased cyclic depth corresponding to increased runtime. Hence, the upper limit of 6 routes was chosen to allow large neighbourhoods to be searched by the heuristic while limiting the computational resources devoted to the heuristic. Additionally, experimentation revealed that increasing the cyclic depth beyond this upper limit yielded no additional improvements in the solution, for the additional computation resources (see section 5.2.3.4).

5.2.3.2 Justification for the capacity overload factor (cof) range

The *cof* parameter requires real number values. The range chosen was between 1.0 and 1.5 inclusive. Within this range the parameter value were then increased in increments of 0.1. This range encompasses the values that yielded the best results (we found) for the heuristic.

Values of less than 1 for the *cof*, strengthen the capacity constraints and therefore has the opposite effect to that desired from the constraint relaxation heuristic: that of relaxing the capacity constraints. These values were therefore not considered. A *cof* of 1, indicates that the problem stated capacity is applied at all levels i.e. constraint relaxation has no effect on the solutions produced. As the *cof* is increased, the feasibility of the solution decreases, for instance, a *cof* of 2 indicates routes can be formed having capacity twice the problem specified values. The experimentation revealed that forming infeasible solutions beyond a certain point failed to result in improved final solutions. This point proved to be values above 1.5 for the *cof*. The increment of 0.1 was the smallest increment at which consistent changes in the solution quality could be discerned due to changes in the parameter (see section 5.2.3.4).

5.2.3.3 Justification for the segment balancing factor (sbf) range

The *sbf* has been a particularly difficult parameter to tune. As with the *cof*, the *sbf* requires real number values. Since the *sbf* is used along with the average demand of customers in a problem to determine the targeted demand values of segments created by the coarsening homogeneity heuristic, values below 0 for the *sbf* are invalid. Values between 0 and 1.0 were found to produce a 'drawn out' coarsening process consisting of many levels and almost no coarsening in the lower levels. This resulted in wasted computational resources in the lower levels of the refinement phase. Values far in excess of 1.5 produced a 'compressed' coarsening process that coarsened the problems in very few levels. The segments so produced, did not offer the benefits to inter-route heuristics sought and consequently the final solutions were of poor quality.

The range chosen (between 1.0 and 1.5 inclusive) contains the values for the parameter that gave the coarsening homogeneity heuristic the best chance to produce the effect desired, that of facilitating additional improvements due to inter-route heuristics (see section 5.2.3.4). The increment of 0.1 was chosen for the same reason as that given for choosing the increment for the *cof*.

5.2.3.4 Experimentation for parameter tuning

The results for tuning the *cof* and *sbf* are summarized in the Table 2 and the results for the tuning of the cyclic depth are summarized in Table 3. In both tables the solution quality is expressed as the cost of each instance normalised with respect to the *bks* for each instance and averaged over all instances in the test suite. This is followed by the standard deviation of the solution quality. The runtimes for Table 3 are normalised with respect to instance size for each instance and averaged over all instances.

Table 2 Results of tuning the capacity overload factor and segment balancing factor for a constant cycle depth of 4. Two-phase coarsening is applied with the parallel *CWS* used to produce feasible initial solutions. The multilevel algorithm is applied to these for the Christofides instances

cof	*sbf*	Average normalised cost	Standard deviation
1.0	1.1	1.043	0.028
	1.5	1.051	0.033
1.1	1.1	1.041	0.032
	1.5	1.051	0.035
1.2	1.1	1.042	0.027
	1.5	1.055	0.031
1.3	***1.1***	***1.041***	***0.027***
	1.4	1.058	0.033
1.4	1.0	1.050	0.028
	1.2	1.059	0.028
1.5	1.0	1.066	0.031
	1.1	1.047	0.024

In tuning the parameters, each cycle depth value in the range is fixed during testing one at a time. For each cycle depth value, the *cof* is fixed one at a time and the *sbf* varied. In total, 180 sets of test were conducted. The results summarised in both of these tables were done around the cycle depth value that produced the best results.

The best values were found when the cyclic depth was set to 4. The minimum and maximum normalised cost values found for each *cof,* while varying the *sbf,* and the cycle depth is set to 4, is shown in the table above. It can be seen that as the *cof* is increased beyond 30% of the problem stated capacity, the quality of the results decreases. The best values found are highlighted.

All combinations of the *cof* and *sbf* within the range of 1.0 and 1.5 in increments of

0.1 where tested with the cycle depth varied between 2 and 6 as described above. The results of Table 3 show the effect on the results for varying the cycle depth with the *sbf* and *cof* set to the best values found for these two parameters.

The best combination of values for the *sbf*, *cof*, λ and cycle depth that were found for the Christofides instances were 1.1, 1.3, 1 and 4 respectively. These parameters were kept constant for the CVRP results provided below.

Table 3 Results for tuning the cyclic depth with sbf set to 1.1 and cof set to 1.3. Two-phase coarsening is applied with the parallel *CWS* used to produce feasible initial solutions. The multilevel algorithm is applied to these for the Christofides instances

Cycle depth	Average normalised cost	Standard deviation	Average normalised Time (min)
2	1.057	0.036	0.001
3	1.042	0.028	0.002
4	**1.041**	**0.027**	**0.003**
5	1.041	0.027	0.004
6	1.041	0.027	0.016

5.2.4 Coarsening Heuristics Testing

One of the contributions of this research has been the creation of new ways of coarsening for the multilevel technique. In this section, we look at the results produced from these varied coarsening approaches.

Section 4.2.2 described four methods of coarsening that were implemented for the CVRP (savings, nearest neighbour, parallel nearest neighbour and parallel Clark-Wright Savings). These four methods were tested across the Christofides test suite, both with and without the use of coarsening homogeneity and constraint relaxation. When constraint relaxation was employed, each method was tested in the case where constraint relaxation is used to produce infeasible solutions at the end of the coarsening phase. Additional tests were done for the situations where constraint relaxation was applied by the split procedure during the refinement phase, starting from feasible initial solutions.

Column 1 of the following three tables presents the coarsening methods. When coarsening is used to create the initial solutions, the merges are implemented by the savings and the neighbour heuristics and when two-phase coarsening is used, the solutions are constructed by the parallel nearest neighbour heuristic and the parallel CWS. Columns 2 and 3, present the average solution cost above the best known at the end of the coarsening and refinement phases respectively. Columns 4 and 5 provide the runtimes averaged over all instances in the test suite, again at the end of the coarsening and refinement phases respectively.

Table 4 The multilevel algorithm applied to the Christofides instances with varying coarsening strategies. Coarsening homogeneity and constraint relaxation *were not* used in the solution process. The initial solutions are feasible.

Method of Coarsening	Average % cost above *bks* end of coarsening	Average % cost above *bks* end of the solution	Average time - end of coarsening (Min)	Average time end of solution (Min)
Coarsening used to create initial solution (savings heuristic)	81.31	9.15	0.02	0.27
Coarsening used to create initial solution (nearest neighbour heuristic)	70.51	7.97	0.002	0.24
Two phase Coarsening (parallel nearest neighbour heuristic)	32.13	**5.46**	0.03	0.26
Two phase Coarsening (parallel CWS)	7.63	5.55	0.05	0.18

The results shown in Table 4 looks at the case where coarsening homogeneity and constraint relaxation are not used in the solution process. From these results, it can be seen that without the enhancements, two-phase coarsening using the parallel nearest neighbour heuristic produces the best final results. However, two-phase coarsening using the parallel CWS produced the best initial results and the final results produced starting from this method of coarsening is comparable to the best results found for these experiments. The two-phase coarsening methods outperformed the others in all areas.

The results shown in Table 5 look at the case where coarsening homogeneity and constraint relaxation are used in the solution process. In this case, the enhancements are employed in the refinement phase only. Hence, the initial solutions are feasible and are equivalent to the initial solutions presented in Table 4. With the addition of the enhancements, two-phase coarsening using the parallel CWS, produced the best final results and the best results found for all the testing on the coarsening heuristics, that of 4.11 % above *bks.* This is as opposed to the previous case where the parallel neighbour heuristics produce the best final results. Again, the two-phase coarsening methods outperformed the others in all areas.

Table 5 The multilevel algorithm applied to the Christofides instances with varying coarsening strategies. Coarsening homogeneity and constraint relaxation *are used* in the solution process. The initial solutions are feasible

Method of Coarsening	Average % cost above *bks* end of coarsening	Average % cost above *bks* end of the solution	Average Time end of coarsening (Min)	Average Time end of solution (Min)
Coarsening used to create initial solution (savings heuristic)	81.31	8.03	0.02	0.55
Coarsening used to create initial solution (nearest neighbour heuristic)	70.51	7.88	0.002	0.51
Two phase Coarsening (parallel nearest neighbour heuristic)	32.13	4.96	0.03	0.55
Two phase Coarsening (parallel CWS)	7.63	**4.11**	0.05	0.61

For the results shown in Table 6 coarsening homogeneity and constraint relaxation are used in the solution process. In this case however, the enhancements are used to create infeasible initial solutions. This is different from the two previous sets of results. From these results the parallel nearest neighbour heuristic led to the best final solutions, but the result of 4.88% above the *bks* is outperformed by the best found in the case shown in Table 5, that of 4.11 % above the *bks.*

Table 6 The multilevel algorithm applied to the Christofides instances with varying coarsening strategies, Overloading and Segment Balancing are used in the solution process. The initial solutions are infeasible.

Method of Coarsening	Average % cost above *bks* end of coarsening	Average % cost above *bks* end of the solution	Average Time end of coarsening	Average Time end of solution
Coarsening used to create initial solution (savings heuristic)	75.32	7.64	0.01	0.41
Coarsening used to create initial solution (nearest neighbour heuristic)	65.41	7.44	0.001	0.42
Two phase Coarsening (parallel nearest neighbour heuristic)	30.60	**4.88**	0.02	0.45
Two phase Coarsening (parallel CWS)	7.18	5.34	0.06	0.51

5.2.4.1 Conclusions from the Coarsening Heuristics Testing

The following conclusions can be drawn from the results presented for the four methods of coarsening.

The methods of coarsening relying on the savings and nearest neighbour heuristic, are consistently outperformed by the two-phase coarsening methods, and can therefore be disregarded as suitable coarsening approaches for the CVRP. The quality of the solutions produced, at the end of coarsening, by the parallel nearest neighbour heuristic is on average 4 times worse than that produced by the parallel Clark-Wright Savings heuristic. However, the quality of both sets of solutions at the end of refinement compares favourably. This indicates that the parallel nearest neighbour heuristic can be used as a suitable construction heuristic for the CVRP.

The solutions produced by the parallel Clark-Wright Savings heuristic were close to local optima with regards to the accessible neighbourhoods [28] for the refinement algorithms as incorporated in the multilevel framework. Since, initial solutions produced by the parallel nearest neighbour heuristic were poorer, the refinement algorithms were able to obtain larger percentage improvements.

For the preferred means of coarsening, that of constructing feasible solutions, the average time spent during coarsening was always less than 28% of the overall solution time (see Table 4 and Table 5). It can also be seen that the runtimes for two-phase coarsening exceeded the runtimes where coarsening was used to create the initial solutions by a factor of 2.5 in best cases and a factor of 15 in worst cases (see Table 4 and Table 5). However, the final runtimes obtained using either type of coarsening are almost identical, with two-phase coarsening producing superior solutions in all cases. The tradeoffs between constructing a solution very quickly, which typically requires longer to improve or constructing a solution of better quality and spending less time in refinement, is one the multilevel algorithm has to balance. For the CVRP two-phase coarsening, appear to provide an acceptable balance.

The use of constraint relaxation to produce infeasible routes at the end of coarsening proved less effective, compared to the cases where the initial routes were feasible and constraint relaxation implemented during the refinement phase. Where constraint relaxation is reported on in the remainder of the results, this refers to constraint relaxation implemented during the refinement phase.

In summary, it can be seen that the best results were obtained when the following conditions existed:

- Two-phase coarsening was used along with coarsening homogeneity and constraint relaxation.
- The initial routes, constructed by parallel CWS were feasible.

This is the method of coarsening used for all further results produced for the CVRP unless otherwise stated.

Note additionally that for the single-level algorithm, the parallel CWS proved the dominant heuristic in all cases. Also coarsening without constraint relaxation predominantly produced better results. This occurred because using constraint relaxation with the single-level algorithm meant the constraints had to be brought back in line immediately at level zero. This contrasts with the case of the multilevel algorithm where the process of satisfying the constraints can be gradually implemented throughout the refinement levels. This implies that the multilevel algorithm is a platform potentially more suited to cases where constraint relaxation is desired.

5.2.5 Testing Enhancements and Problem Types

In the previous section (5.2.4) the preferred method of coarsening was identified as two-phase coarsening using the parallel CWS to construct feasible initial solutions. In this section, that method of coarsening is used with the multilevel algorithm, and coarsening homogeneity and constraint relaxation are varied, to assess their influence. It is of interest to look at how coarsening homogeneity and constraint relaxation influences the results, as in the experimental testing of the algorithms these heuristic approaches appear to offer significant improvements on some of the problem instances, most notably the ones in the Christofides test suite.

In the following results of Table 7, a *segment balancing factor* (*sbf*) of 1.1 indicates that the targeted level demand at the first level of coarsening is 1.1 times the average demand of customers in the problem. Similarly, a capacity overload factor (*cof*) of 1.3 indicates that the problem stated route capacity can be increased by a maximum of 30% when constraint relaxation is employed. The values used for the parameters *sbf* and *cof* are the best values found for these parameters, as outlined in the section on parameter tuning (section 5.2.3). N/A indicates the heuristic was not used. The last four columns list the solution values normalized with respect to the *bks* for each problem and averaged over all problems in the test suite or sections of the test suite as outlined.

Table 7 A comparison of the different heuristics approaches applied to the Christofides et al. instances

sbf	cof	Average normalised cost			
		All instances	Clustered VRPs	DVRPs	VRPs
N/A	N/A	1.056	1.008	1.080	1.069
N/A	1.3	1.054	1.014	1.078	1.061
1.1	N/A	1.043	1.006	1.061	1.056
1.1	1.3	**1.041**	1.018	1.059	1.042

From a comparison of the four different configurations of *sbf* and *cof* shown in Table 7 it can be seen that using either coarsening homogeneity or constraint relaxation leads to improvements in the results, compared to the cases where these heuristics are not used. Additionally we see that of the two, coarsening homogeneity has the greatest effect and offers improvements for all problems types.

5.2.5.1 Comparison of the clustered and non-clustered instances

Table 8 shows solution quality for the clustered Christofides instances compared with the non-clustered instances.

Table 8 Solution quality for the clustered vs. non-clustered Christofides et al. instances

Instances	Average normalised cost	
	End of coarsening	End of solution
Non-clustered VRPs (instances 1 -10)	1.098	1.050
Clustered VRPs (instances 11 -14)	**1.022**	**1.018**

The results indicate that where the problems are clustered there is a greater likelihood of improvements being found in the solution cost. Whilst more extensive testing needs to be done using larger datasets of clustered problems, the results appear to arise for the following reason. When the problems are clustered, there is a high probability that the nodes can be partitioned into disjoint clusters with the nodes of each cluster belong to the same route, allowing for the capacity constraints. The coarsening phase is well adapted at identifying these clusters, as can be seen from the results shown in Table 8. The refinement phase then needs only improve the solution by optimising the order the customers of each route are visited. The refinement phase is then equivalent to solving a TSP utilising Lin-Kernighan type heuristics [304]. It has been demonstrated in the literature [304] that the multilevel technique performs well in these circumstances.

5.2.5.2 Conclusions from the Testing of Enhancements and Problem Types

The following conclusions are drawn from the testing carried out for coarsening homogeneity and constraint relaxation.

Firstly, in a situation where there is no constraint relaxation, at the start of the refinement phase the total capacity in the solution (equal to the problem capacity multiplied by the number of routes) is close in value to the total demand of the solution. This means that there is a high possibility that inter route moves will be rejected because they violate the capacity constraint on the routes. Conversely, when the solution uses constraint relaxation the total capacity of the solution[1] at the start of the refinement phase exceeds the total demand of the solution. This additional capacity increases the possibility of implementing improving inter-route moves at the upper levels.

Coarsening homogeneity lead to solutions of better quality as it enhanced the possibility of improvements being found by inter-route heuristics. Where coarsening homogeneity is not employed, improving moves can be prevented due to the capacity constraints. Example, an improving move involving the exchange of two segments of vastly differing demand values, between two routes both of capacity equal to the maximum allowed capacity at a given level, would be prevented as one of the routes would violate the capacity constraints. In the cases where coarsening homogeneity is

[1] The total capacity of the solution is equal to the (relaxed) capacity of a route times the number of routes in the solution.

employed, the segments are likely to be approximately equal in demand and moves similar to the one just described would be allowed. Therefore, we can conclude that the use of coarsening homogeneity yields improvements as the performance of inter–route heuristics improves when exchanging segments that are approximately equal in demand. This is so as the effect of the capacity constraints is minimised and negated if the segments are equal in demand.

It is interesting to note the cases where the use of constraint relaxation returns results worse than those obtained when it was not employed. This highlights a weakness with the multilevel algorithm that can occur even in cases where constraint relaxation is not employed and for all types of problems. This weakness stems from the fact that the algorithm can accept improving moves in the upper levels that results in a local optimum from where the algorithm can find no further improvement. This can result in situations where, for a given problem instance, an equivalent single-level algorithm could return results superior to its multilevel counterpart. This effect is more likely to surface when constraint relaxation is used as it permits a wider array of improving moves in the upper levels. This can then result in a solution that in some cases cannot be improved to the same quality as that possible in the case where constraint relaxation was not used and the corresponding moves not permitted.

However, across all the testing we have conducted it has been found that coarsening homogeneity and constraint relaxation are two enhancements to the multilevel technique capable of aiding the multilevel solution process.

5.2.6 Component Testing

The refinement algorithm descried in section 4.2.3 (p.91), used a combination of inter and intra route heuristics. It is of some interest to know how significant the individual heuristics or components of that algorithm are to the solutions produced by the algorithm. Using the Christofides et al. instances [49], this section analyses the effectiveness of the components of the multilevel algorithm, acting with and without the enhancements implemented for the algorithm.

The four components of the multilevel algorithm are shown in columns 2 – 5 of Table 9. Column 6 shows the two enhancements, constraint relaxation (cr) and coarsening homogeneity (ch). A dot indicates that the components or enhancements were used in the solution process. The last three columns list the solution values normalized with respect to the *bks* for each problem and averaged over all problems in the test suite, the standard deviation of the solution values, and the runtime normalised with respect to problem sizes and averaged over the test suite.

Table 9 Varying the combination of components that make up the multilevel algorithm.

Configuration	Components				Enhancements	Measurements		
	3-opt	Split Procedure	Simple Segment Transfer	Cyclic segment Transfers	cr and ch	Average normalised cost	Standard deviation	Average normalised Time
1	▪					1.070	0.043	0.0003
2	▪	▪				1.070	0.043	0.0003
3	▪		▪			1.068	0.042	0.0003
4	▪			▪		1.060	0.037	0.001
5	▪		▪	▪		1.054	0.034	0.001
6	▪	▪	▪			1.068	0.042	0.0003
7	▪	▪		▪		1.059	0.037	0.001
8	▪	▪	▪	▪		1.056	0.035	0.001
9	▪	▪			▪	1.089	0.046	0.0003
10	▪	▪		▪	▪	1.060	0.035	0.002
11	▪	▪	▪		▪	1.083	0.044	0.0004
12	▪	▪	▪	▪	▪	**1.041**	0.027	0.003

The 3-opt exchange is always used, as coarsening followed by expansion and 3-opt refinement is the most basic multilevel configuration that can be applied to the CVRP for the multilevel algorithm that was implemented. The best configuration is 12, which uses all four components and both enhancements. Simple segment transfers and cyclic segment transfers are the two most significant components, as demonstrated by the results of configurations 3 and 4. Based on these results, the full multilevel algorithm, as shown in configuration 12 is used for CVRP.

5.2.7 Experiments and algorithms

The remainder of this chapter on the CVRP presents results for the single-level (SL) and multilevel (ML) algorithms applied to all the CVRP test suites used in the research. Results are also presented for the iterated single-level (It.SL) and iterated multilevel (It.ML) algorithms applied to the CVRP test suites (except for the very-large scale instances, where runtime precluded this analysis). Where the iterated versions of the algorithms are employed, the number of iterations for which the solutions are explored is varied based on the size of the problem instances in the test suite. For the Christofides et al. and Christofides and Eilon test suites, the number of iterations is set to 10 iterations and for the Golden test suite, it is set to 5 iterations.

For the tables of results presented in the following three sections (section 5.2.8, section 5.2.9 and section 5.2.10), the first two columns give the problem instance number and problem size. The next four columns compare the solution quality produced by the four algorithms, whilst the last four columns present the corresponding runtimes in minutes.

5.2.8 Results for the Christofides et al. test suites

This section presents the detailed results for the Christofides et al. [49] instances.

Table 10 Results for the Christofides et al. instances.

Instance		Quality (% above bks)				Time (min)			
		SL	ML	It.SL	It.ML	SL	ML	It.SL	It.ML
	N								
1	50	8.62	1.96	8.62	0.49	0.01	0.05	0.01	0.14
2	75	5.75	2.42	5.75	0.76	0.03	0.14	0.01	1.72
3	100	6.78	4.20	6.67	1.92	0.05	0.27	0.03	2.67
4	150	9.70	5.47	8.20	2.25	0.16	0.67	0.18	3.16
5	199	7.29	6.98	6.94	3.39	0.38	1.40	0.55	10.19
6	50	11.56	1.79	11.56	1.70	0.01	0.11	0.01	0.08
7	75	5.10	5.08	4.13	4.49	0.09	0.19	0.09	0.30
8	100	11.01	7.29	8.15	6.71	0.16	0.54	0.42	0.55
9	150	10.56	7.13	10.56	5.72	0.27	1.35	0.16	2.00
10	199	9.42	7.97	8.44	4.12	0.45	2.25	0.81	26.14
11	120	0.71	5.27	0.52	3.29	0.07	0.40	0.06	5.95
12	100	0.30	0	0.13	0	0.05	0.18	0.05	0.11
13	120	2.29	1.29	1.36	1.26	0.17	0.78	0.53	0.70
14	100	0.25	0.71	0.25	0.21	0.06	0.19	0.04	0.23
Average		**6.38**	**4.11**	**5.81**	**2.59**	**0.14**	**0.61**	**0.21**	**3.85**

As can be seen, the multilevel algorithm can get within an average of 4.11% of the best known values (see section 8.2) with an average runtime of 0.61 minutes. The single-level algorithm is considerably faster (by around a factor of 4, as predicted by Walshaw in a general discussion of multilevel schemes [304]) but gives results approximately 1.5 times worse in quality compared to the multilevel algorithm results.

The iterated multilevel algorithm provides a means of obtaining very good results (2.59% above best known) but with a corresponding compromise made on the computational runtime. Indeed, the quality of the solutions found by the iterated multilevel version is comparable to the improved petal heuristic [225], which reports an average cost of 2.43% above the best known values but with faster runtimes (see section 5.2.13).

The results indicate that where the problems are clustered (problems 11, 12, 13, 14) there is a greater likelihood of improvements being found in the solution cost. This can be explained by the following: when inter-route heuristics transfer segments of equal cost and demand between routes, an improvement is found once the costs of the new edges are less than the costs of the edges being replaced in the affected routes. In a clustered problem, new edges generated by exchanges of segments between the routes should be similar in cost and similar to the cost of old edges since the segments are close to each other.

On average, therefore, most potential moves attempted should return a cost approximately equal to zero (no improvement or deterioration in the cost function). In some cases, there may be slight differences in the cost of the new edges compared to the old edges. In the cases where these differences correspond to improvements, the solution is changed. This allows the algorithm to explore areas of the search space that might not have been possible in the case where the problems are not clustered. This effect can be further seen when we look at solving the clustered problems with and without coarsening homogeneity. The results were worst in quality in the case where coarsening homogeneity was not used (see section 5.2.5).

5.2.9 Results for the Christofides and Eilon test suites

Results for selected instances from the Christofides and Eilon instances are presented in this section.

The multilevel algorithm for these instances (see Table 11) returns results 2.88% above the *bks* compared with 3.29 % above the *bks* (see section 8.1) for the single-level. The general trends observed for small to medium scale instances are confirmed by these results: The multilevel algorithm has the ability to improve on the single-level results and this is further improved by the use of the iterated version. Again, the results for the iterated multilevel algorithm are highly competitive with the standard in the published literature.

Table 11 Results for the selected Christofides and Eilon instances

Instance	N	Quality (% above bks) SL	ML	It.SL	It.ML	Time (min) SL	ML	It.SL	It.ML
E-n22-k4	21	2.47	2.47	2.41	0.07	0.001	0.006	0.001	0.006
E-n23-k3	22	0	0.18	0	0.18	0.002	0.003	0.001	0.003
E-n33-k4	32	0.80	0.69	0.80	0.36	0.002	0.01	0.002	0.03
E-n76-k14	75	6.25	4.69	6.12	2.68	0.02	0.083	0.05	0.32
E-n76-k8	75	4.50	2.57	4.49	2.06	0.02	0.083	0.02	0.48
E-n76-k7	75	2.89	4.43	2.89	1.35	0.02	0.12	0.02	3.05
E-n101-k14	100	6.14	5.16	6.08	2.52	0.03	0.15	0.05	2.06
Average		**3.29**	**2.88**	**3.26**	**1.32**	**0.014**	**0.07**	**0.20**	**0.85**

5.2.10 Results for the Golden test suite

This section presents the results for the Golden test suite [122] of large-scale problem instances (see section 8.3 for *bks*). For the Christofides instances, the average problem size is approximately 114 nodes while for these, the Golden instances, it is approximately 348 nodes. Although the average problem size has been increased by a factor of three, the average single-level runtime has increased at a rate approximately twice that experienced for the multilevel runtime (see runtimes Table 10 and Table 12). This finding is consistent with the expectation of the multilevel algorithm as a framework more suitable for larger scale problems [304] and arises because most of the large improvements in the solution quality are made at the (computationally cheap) coarsest levels. Thus, when the multilevel scheme reaches the (computationally expensive) finer levels it already has a high quality solution to work with, in contrast to the single-level algorithm which must do all its computations on the entire problem, working with what initially may be a very poor solution.

Table 12 Results for the Golden test suite.

Instance	N	Quality (% above bks) SL	ML	It.SL	It.ML	Time (min) SL	ML	It.SL	It.ML
1	240	4.61	4.58	4.61	3.22	2.31	1.99	2.24	4.97
2	320	8.54	7.33	7.05	6.64	1.61	4.64	5.69	14.31
3	400	11.54	11.34	11.52	11.28	4.71	8.34	6.47	11.87
4	480	14.35	14.09	14.35	9.24	8.97	27.48	8.71	313.87
5	200	7.86	7.29	7.46	4.78	0.75	1.83	2.01	24.08
6	280	11.56	11.38	10.98	11.19	1.33	2.88	7.89	11.58
7	360	15.97	13.91	15.97	13.32	3.24	6.83	3.17	40.56
8	440	11.24	10.55	11.24	10.47	7.66	16.33	7.44	33.08
9	255	11.78	11.33	8.94	2.98	0.32	1.01	1.96	13.24
10	323	13.89	14.11	12.57	6.35	0.74	1.80	1.92	29.07
11	399	13.71	12.82	11.26	3.91	1.36	2.90	5.68	55.98
12	483	12.17	13.10	8.74	7.04	2.18	4.82	11.85	62.00
13	252	10.10	7.21	6.20	2.72	0.42	0.89	2.42	18.04
14	320	9.59	9.63	8.57	5.56	1.07	1.78	4.20	18.00
15	396	9.08	8.20	5.31	2.77	1.05	2.78	7.96	54.52
16	480	11.02	8.61	5.98	3.93	2.12	5.98	14.39	89.29
17	240	6.56	6.66	6.36	6.56	0.48	0.74	0.60	1.15
18	300	4.28	3.98	4.14	3.98	0.60	1.07	1.53	1.07
19	360	4.65	5.54	4.42	4.95	1.33	2.21	2.58	8.58
20	420	5.18	5.11	5.03	3.69	2.17	3.54	4.60	26.62
Average		**9.88**	**9.34**	**8.54**	**6.23**	**2.22**	**4.99**	**5.17**	**41.59**

For these larger problem instances, the multilevel algorithm again outperforms the single-level version on solution quality. The results of the multilevel algorithm are a factor of 1.06 better then the single-level results based on a comparison of average result of both algorithms (see Table 12). The iterated multilevel algorithm further improves the multilevel results, but we can also see that the improvements found are not as pronounced as those for the Christofides test suite. However, we expect that if the iterated multilevel algorithm were allowed more runtime, it would continue to make gradual improvements to the solutions. This expectation is supported by the algorithmic performance plot of Figure 35 which shows a convergence towards the best-known values.

5.2.11 Algorithmic comparisons across all three test suites

Figure 35 shows the performance of the iterated multilevel (It.ML) and iterated single level (It.SL) algorithms applied to the instances of Golden et al. [122], Christofides et al. [49] and Christofides and Eilon [48], referred to as Eilon in the figure. Starting with the values obtained from the single-level and multilevel algorithm, each plot shows the quality of the solution with respects to best known as the algorithms are iterated.

Overall, the iterated multilevel algorithm is the dominant algorithm over all the test suites. Indeed, the iterated multilevel algorithm does not yet appear to have reached its asymptotic performance and, given sufficient runtime, could well approach closer to the best known values. In contrast, the iterated single-level algorithm produces negligible improvements in the solution cost as compared with the single-level algorithm. Finally the performance of both these algorithms deteriorates for increased problem size.

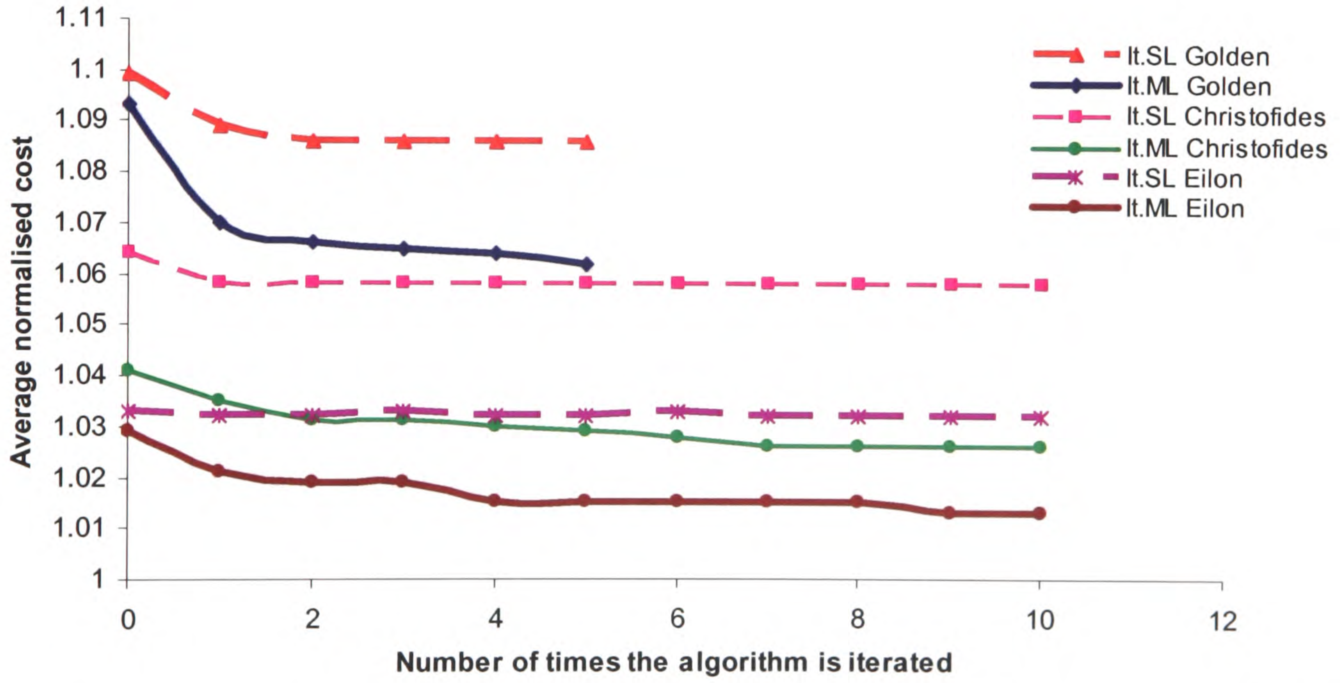

Figure 35 Comparisons of the algorithms applied to the Golden, Christofides and Christofides and Eilon instances. Iteration value zero indicates the result for running the single-level or multilevel algorithm.

5.2.12 Results for the Very Large Scale CVRP instances.

This section presents the results for the multilevel and single-level algorithm applied to the very large-scale Li et al. [170] instances. Also, this section presents results showing that the multilevel algorithm is able to produce superior results to the single-level algorithm in the upper levels of refinement. This is a feature of the performance of the multilevel algorithm than can be exploited in solving large problems. This is of particular importance for large problems, as in some instances a fairly significant percentage of the runtime incurred by the multilevel algorithm occurs when refining the solution at level zero.

Table 13 Results for the Li et al. instances.

Instance	N	Quality (% above BKS) SL	ML	*ML¤*	Time (min) SL	ML	*ML¤*
21	560	13.37	12.83	12.90	14.79	32.64	25.12
22	600	6.96	6.90	6.37	18.59	34.16	15.50
23	640	11.50	11.07	11.33	57.80	51.67	45.19
24	720	11.56	10.53	11.42	44.60	79.10	16.99
25	760	2.59	2.48	2.483	35.81	66.49	32.32
26	800	10.50	10.34	10.47	58.69	104.22	53.00
27	840	7.80	8.12	7.425	26.94	74.10	36.46
28	880	12.69	12.47	12.66	89.80	177.73	106.05
29	960	12.39	11.84	12.32	64.97	337.13	162.93
30	1040	12.62	12.48	12.60	287.74	451.15	203.61
31	1120	12.82	12.74	12.82	336.59	846.64	353.20
32	1200	12.98	12.91	12.99	413.43	853.68	401.43
Average		**10.65**	**10.39**	**10.48**	**120.81**	**259.05**	**120.98**

The problem instances of Li et al. [170] are solved by both multilevel and single-level algorithms and the results are presented in Table 13. The first two columns give the problem instances as they appear in [170] and the problem sizes. Columns 3 and 4 compare the solution quality as a percentage above the bks (see section 8.4) for the single-level and multilevel algorithms, whilst columns 6 and 7 present the corresponding runtimes in minutes. Columns 5 and 8 presents results for a shortened run of the multilevel algorithm (ML¤). In this run the problem constraints are strengthened during the refinement phase such that they are all satisfied at (what would normally be) the end of the penultimate level of refinement i.e. level 1.

The multilevel algorithm again outperformed the single-level algorithm, in this instance by a factor of 1.03 based on the average results of both algorithms in the preceding table. This confirms the multilevel algorithm's ability to aid the performance of local search heuristics deployed for the CVRP. This has been demonstrated over all problem suites used.

One of the features the multilevel algorithm offers is that solutions in the upper levels (once feasibility is ensured) are often better than those found by the single-level algorithm. This feature can be exploited when working with very large-scale instances as demonstrated in this case. Starting from feasible initial solutions, all problem constraints were satisfied by the end of the penultimate level of refinement and the solution cost and runtime attributed to the *ML¤* algorithm were measured at this level. As can be seen from the results of Table 13, the *ML¤* algorithm slightly outperforms the single-level algorithm when not refining the entire problem. The problem is still in a partially coarsened state at level 1. This feature can be further exploited in multilevel algorithm implementations. Note that in the upper levels, the runtime of the *ML¤* algorithm is comparable to that of the single-level algorithm. A practitioner could therefore satisfy the problem constraints and take a solution at some level of the multilevel refinement above zero. This might be useful in a situation where a quick solution is needed and the decisions made on that solution can be adapted later if a better solution is found.

5.2.12.1Result analysis for the Irnich et al. instances

The final set of instances tested for the CVRP, were the instances of Irnich et al. This section analyses the performance of the multilevel and single-level algorithms for these instances.

From the instances of Irnich et al. [147], 140 problem instances are chosen from series 1 – 5. Table 14 shows the distributions for the customer locations, demand, and capacity for each series. The problems chosen are of the following sizes 250, 300, 400, 500, 600 and 700 customers. Each problem size is represented in all the series. For each problem size in a series, a problem instance of that size is tested for each capacity value; consequently, the demand distribution is varied for each such problem instance.

The results (see section 7.1, p.230) looked at the solutions obtained for all 140 problem instances grouped by problem sizes. Within each grouping, the results looked at the effect of varying the vehicle capacity and consequently the average number of customers per route and the effect of varying the demand distribution of the customers.

The multilevel algorithm performed better where there was a 'good' spread in the distribution of the customer demand as in series 2. Where the initial demands are fairly large, and with a tight distribution, the solution quality is poorer compared to the cases where the initial demand is smaller. This is consistent with the effect of coarsening, as the chances of refinement in the upper level will be limited in these instances. The runtimes were analysed for all problem sizes. While confirming that the multilevel algorithm will take more runtime for these instance sizes, the results were not particularly revealing. This however is a test suit worthy of further usage in CVRP research.

Table 14 The series of the Irnich et al. instances.

Series	Location (x y)	Demand	Capacity
1	$[-100, +100]^2$	[10, 30]	{500, 1000,1500,2000}
2	$[-100, +100]^2$	[10, 50]	{750, 1500,2250,3000}
3	$[-100, +100]^2$	[10, 90]	{1250, 2500,3750,5000}
4	$[-100, +100]^2$	[1, 99]	{1250, 2500,3750,5000}
5	$[-100, +100]^2$	[90, 110]	{2500, 5000,7500,10000}

5.2.13 Comparison of Multilevel results with other solution techniques.

Some of the leading solution techniques for the Christofides et al. instances [49] are presented in the Table 15. Since these problems have been extensively reviewed through the years a complete comparison is beyond this thesis. However, additional comparisons can be found in [283].

Those chosen are: A genetic algorithm (*PGA*) due to Prins [221] implemented in Delphi 5 on a Pentium-3 1 GHz PC. A tabu search heuristic termed *Tanuroute* [108] by Gendreau, Hertz and Laporte, implemented on a Silicon Graphics workstation, 36 MHz, 5.7Mflops. A hybrid guided local search and evolution strategies (*HES*) metaheuristic [180] due to Mester and Bräysy implemented in VB6.0 on a Pentium IV Net Vista 2800 MHz PC. An improved petal composite heuristic (*IPH*) by Renaud et al. [225] implemented in Pascal on a Sun Sparcstation 242 Mflops. A granular tabu search (*GTS*) by Toth and Vigo [293] implemented on a Pentium 200 MHz PC, 15 Mflops.

Table 15 Comparison of solution approaches for the Christofides et al. instances.

Solution Approach	Quality (% above *bks*)	Runtimes (Min)	Runtimes Scaled to *PC4* (Min)	TTP (MFlops)
PGA	0.23	5.19	-	N/A
Tanuroute	0.86	46.8	0.09	5.7
HES	0.03	2.8	-	N/A
IPH	2.38	3.48	0.05	42
GTS	0.64	3.8	0.02	15
It.ML	2.59	3.85	3.85	2880

It is difficult to compare the runtimes of different algorithms implemented on different platforms. The Toward Peak Performance (*TPP*) [76] value is used in attempting to do this. The *TPP* value for a computer gives an indication of the "number of millions of floating-point operations the computer can execute per second (*Mflop/s*) when solving a systems of equations of order 1000 in a Fortran environment" [76]. The

TTP rating [76] of the computer used in this research (*PC4*) is 2880 MFlops. The runtimes in Table 15 are scaled to the *PC4*.

The multilevel algorithm implemented, collaborated the solution process of local search heuristics, this therefore means the final results were not expected to outperform the leading metaheuristics in the field. However, from the above comparisons it can be seen that the iterated multilevel algorithm can produce results comparable to the current standard in the field.

5.2.14 CVRP Conclusion

Across all the test suites used for the CVRP, the multilevel algorithm improved on the quality of solutions produced by the single-level algorithm. This demonstrates the ability of the multilevel framework to aid the solution process of the leading local search heuristics employed for the CVRP. The iterated versions of the multilevel algorithm in all instances were able to further improve the quality of the multilevel solutions.

The multilevel enhancements of constraint relaxation and coarsening homogeneity play a role in improving the performance of the multilevel algorithm deployed for the CVRP. The results also demonstrated that using a two-phase coarsening approach can significantly aid the performance of multilevel algorithms for the CVRP.

5.3 *Part II*- Experiments for the Capacitated P-median Problem

The concepts of the multilevel algorithms used for the CVRP, that of coarsening the graph and improving the resulting solutions primarily through the use of inter-route heuristics, were extended to the CPMP. In this case, the graph is coarsened and the solution improved primarily using inter-cluster heuristics. Throughout this part of the chapter the results obtained for the CPMP are presented.

Initially the problem instances used for CPMP experiments are introduced. This is followed by results analysing the coarsening methods implemented for the CPMP. Finally the results produced by the multilevel, single-level and their iterated versions, for each suite of problem instances, are presented.

The results presented in this part of the chapter show that, as in the case for the CVRP, the multilevel algorithms implemented for the CPMP aid the solution process. Therefore, multilevel refinement has a role to play in the field of location analysis.

5.3.1 CPMP Instance Types

A number of standard benchmark suites from the literature, have been used to test the algorithms developed for the CPMP. This section introduces each suite used.

The Osman and Christofides instances [207], encompass two sets (A and B) of problems totalling 20 instances. Set A contains 10 instances of 50 nodes (n) and 5 medians (p) while set B contain 10 instances with $n = 100$ and $p = 10$. The node locations are randomly generated using a uniform distribution in the range [1, 100]. The Euclidean distances d_{ij} between nodes i and j are rounded down to the nearest integer. The demand value for node i , w_i, is generated from the uniform distribution with range [1, 20]. The capacity constraints imposed on each instance are chosen, such that the ratio between the total demand of nodes in an instance divided by the total capacity of the instance (equal to p * the capacity constraint) ranges between [0.82, 0.96]. While this range was not justified by the original proponent of the test instances [207] or by subsequent researchers [91], it can be seen from the nature of the CPMP that increasing the ratio closer to 1 increases the difficulty of constructing feasible solutions and reducing the ratio much pass 0.8 makes the problem simpler to solve. An Internet address is provided by [204] for the above test instances. Maniezzo et al. [178] provided optimal solution values (*opt*) for these instances using a branch and bound algorithm.

A set of larger instances (the San Jose dos Campos instances [175]), available from [145] use real data collected from the Brazilian city of San Jose dos Campos. The six instances have dimensions (n, p) of (100, 10), (200, 15), (300, 30), (402, 30) and (402, 40). Each demand node represents one city block, with the demand estimated based on the number of living quarters in each block. For blocks with no living quarters, the demand is set to one. The capacity constraints are the quotient of dividing the total demand of an instance by a fraction of the number of medians in the instance. The divisor is given by the number of medians times 0.9 or 0.8. These values again influence the difficulty of solving the problem instances.

5.3.1.1 Computational standard CPMP

All computations for the Osman and Christofides (OS) instances [207] are rounded down to the nearest integer, in accordance with the standard in the published literature [207],[91]. Computations for the CPMP San Jose dos Campos instances [175] are done with double precision real numbers and the provided solution values are rounded up to two decimal places. For all test instances the runtimes are measured from the start of the solution process to termination at level zero in the refinement process and includes input times.

5.3.2 Parameter Tuning for the CPMP

The two types of refinement implemented for the CPMP, simple search and tabu search, are tested and reported on. The λ-interchange parameter, $\lambda = 1$, is employed for both types of refinement. It is used to handle the complexity issues as the discussions for the CVRP identified.

Three other parameters are employed when tabu search refinement is used; these are:

- The *number of iterations* available to the tabu search algorithm to search for improving solutions (see Figure 33, p.130). A suitable value for this parameter was determined experimentally to be 4 and kept constant for all the results produced.
- The *tabu range* defining the upper and lower bounds for the number of iterations that solution attributes can be tabu-ed. France et al. [91], proposed setting the minimum tabu range to be the number of customers divided by 4 and the maximum to be equal to the number of customers divided by 2. This has been utilised to determine the tabu range at each level as the number of segments at the level divided by 2 and 4 respectively. These values are kept constant for the results produced.
- The *tolerance parameter*, governing how many tabu edges can be part of a move (see section 4.4.3.3). The work of France et al. [91] advocated using the tolerance parameter equal to 1 for transfers and 3 for exchanges. Experimentation showed these values to be suitable for this thesis.

5.3.3 Results for the Osman and Christofides Instances

5.3.3.1 Construction and Coarsening Heuristics Testing

Two types of coarsening were implemented for the CPMP. These were two-phase coarsening and coarsening used to construct the initial solutions. Where two-phase coarsening was used the initial solutions were constructed by the grouping and greedy heuristics. All the methods of coarsening are experimentally evaluated for both *tabu search* and *simple search* algorithms and the important results presented in the remainder of this sub-section. These results show that the preferred method of solution construction for the multilevel algorithm for the CPMP is two-phase coarsening using the greedy heuristic.

For these results, the average normalised cost values presented, represent the solution costs normalised with respect to the optimal solution values (*opt*) for each problem and averaged over all problems in the test suite. The standard deviations presented are the standard deviations of the average normalised cost. Runtimes are normalised with respect to the number of nodes for each problem and averaged over all problems in the test suite.

5.3.3.1.1 Results for the construction phase of two-phase coarsening

The results for the two construction heuristics, the grouping and greedy heuristics, are presented in this section. These results show the quality of the solution initially passed to the single-level algorithm and to the coarsening phase of the multilevel algorithm when two-phase coarsening is used.

Table 16 Results across both test suits of the Osman and Christofides instances, at the end of *the construction phase.*

Construction Heuristics	Average normalised cost	Standard deviation	Average normalised time (S)
Grouping heuristic	**1.1923**	0.1156	0.003
Greedy heuristic	1.2031	0.1913	0.017

5.3.3.1.2 Results for the coarsening phase

Two types of coarsening results are shown in this section. The first are the results for two-phase coarsening, the second are the results obtained from using coarsening to construct the initial solution. These results show the quality of the solution initially passed to the refinement phase of the multilevel algorithm.

Table 17 Results across both test suits of the Osman and Christofides instances, at the end of *the coarsening phase.*

Method of Coarsening	Average normalised cost	Standard deviation	Average normalised time. (S)
Two-phase Coarsening (Grouping heuristic)	**1.2468**	0.131	0.004
Two-phase Coarsening (Greedy heuristic)	1.2545	0.200	0.018
Coarsening used to create the initial solution	2.9046	0.6039	0.002

In the cases where construction and coarsening are separated (two-phase coarsening), at the end of the construction phase the grouping heuristic returned results an average 19.23% above the optimal values compare with 20.31 % for the greedy heuristic (see Table 16). Since the single-level algorithm does not comprise a coarsening phase, this is the quality of the results at the start of its refinement phase. Two-phase coarsening for the multilevel algorithm is applied to these initial results. At the end of this coarsening phase the differences in the quality of solutions produced were minimal (see Table 17). The increase in cost above that initially produced by a construction heuristic arises from the fact that the final locations for the upper level segments in some instances will be different to the initial median locations assigned to the clusters by the construction heuristic.

5.3.3.1.3 Results at the end of the solution process for the different methods of coarsening

The refinement phase[1] is applied to the results produced by all the methods of coarsening and the final solution results are presented in Table 18, for the OS instances. At the end of the multilevel solution process there are negligible differences between the methods employing two-phase coarsening (see Table 18). The final solutions produced, starting from two-phase coarsening, are outperformed by the solutions produced when coarsening is used to create the initial solutions. However, when coarsening is used to create the initial solution, the solution process requires approximately 40% additional computational effort (compared to when two-phase coarsening is used) based on a comparison of the runtimes for the multilevel algorithms (see Table 18).

Table 18 Results across both sets of the Osman and Christofides test suits, at the end of the *solution* process using tabu search refinement.

Method of Coarsening	Average normalised cost		Standard deviation		Average normalised time (S)	
	SL	ML	SL	ML	SL	ML
Two phase Coarsening (Grouping heuristic)	1.0294	1.0233	0.035	0.028	9.757	21.768
Two phase Coarsening (Greedy heuristic)	1.0325	*1.0233*	0.047	0.031	8.9	21.256
Coarsening used to create the initial solution	1.0326	**1.0200**	0.033	0.020	13.340	30.549

[1] The refinement phase uses tabu search refinement as tabu search refinement outperforms the simple search refinement on all occasions.

While the result for the multilevel algorithm shows the final solutions starting from either two-phase approaches to be equal in quality (see Table 18), when the tests are extend to the iterated multilevel versions and to the set of larger test instances, solutions starting from the initial configuration of two-phase coarsening with the greedy heuristic were better in quality (see Table 19 and Table 20).

Table 19 Results at the end of the iterated multilevel solution process using tabu search refinement across both sets of the Osman and Christofides test suites. For the iterated multilevel algorithm, the number of iterations is equal to 5.

Method of Coarsening	Average normalised cost.	standard deviation	Average normalised time. (S)
	It.ML	It.ML	It.ML
Two-phase Coarsening (Grouping heuristic)	1.0078	0.011	82.99
Two phase Coarsening (Greedy heuristic)	1.0049	0.007	98.17

As the algorithms are extended to the larger test cases, the differences in performance between the two configurations is magnified (see Table 20) and here it can be seen that two-phase coarsening using the greed heuristic is the better of the two configurations.

Table 20 Results at the end of the solution process using tabu search refinement for the Osman and Christofides instances and the San dos Campos instances.

Method of Coarsening	Average normalised cost.	
	Osman and Christofides	San dos Campos
	ML	ML
Two-phase Coarsening (Grouping heuristic)	1.0233	1.0713
Two-phase Coarsening (Greedy heuristic)	**1.0233**	**1.0488**

After reviewing all the results, two-phase coarsening with the greedy heuristic is the preferred means of creating initial solutions for the CPMP multilevel algorithm. It outperforms the other two-phase coarsening methods when viewed across the iterated multilevel algorithm and on the larger set of problem instances. This is the method of coarsening used for all the multilevel results for the CPMP, presented in the remainder of this thesis.

We also note that while the grouping heuristic gave better results than the greedy heuristic for the single-level algorithm for the OS instances, when the investigations were extended to the larger test cases, results starting from the greedy heuristic proved to be of better quality. At the end of the solution process, solutions starting from the greedy heuristic were 7.73% above the optimal values compared to 9.12% for the grouping heuristic. The greedy heuristic is therefore used to produce all the initial solutions for the single-level algorithm, for the remainder of this thesis.

5.3.3.1.4 Coarsening for the CPMP: an analysis

This section discusses why single-phase coarsening, used to create the initial solution, is not advocated as the coarsening method of choice for the CPMP.

When coarsening is used to create the initial solutions, the quality of these initial solutions are an average 190.46% above the optimal values. However, the results produced at the end of the refinement phase of the multilevel algorithm starting from these initial solutions, are better than the best produced when the other coarsening methods were employed, these results being 2 % above the optimal values compared to 2.33% for the recommended configuration (two-phase coarsening using the greedy heuristic, see Table 18). However, the runtime incurred by the heuristic to recover from these poor starting results is approximately 40% more than the runtime experienced for the better starting results, as can be seen for the multilevel runtimes in Table 18.

Similar findings also held for the San Jose dos Campos instances. In this case, superior results were obtained in less runtime starting from two-phase coarsening and then refining using the iterated multilevel algorithm. These results were superior to those produced using coarsening to construct the initial solutions and refining using the multilevel algorithm.

The fact that the problem is capacity constrained and the number of medians specified a priori, means that using coarsening to create the initial solution is unnaturally biased towards the demand considerations as opposed to the cost considerations. For this reason, using coarsening to construct the initial solution results in the solution process consuming more runtime than that used by the two-phase coarsening process which produces better results overall. Two-phase coarsening is therefore advocated for this problem.

5.3.3.2 Refinement approaches applied to the Osman and Christofides (OS) Instances

Two refinement approaches were implemented for the CPMP, simple search and tabu search. This section analyses their performance for both the single-level and multilevel algorithms. The results to be presented show firstly that the multilevel algorithms outperform the single-level algorithms and secondly that configurations using the tabu search approach outperform the simple search versions.

The performance of the multilevel and single-level algorithms using both simple search and tabu search are compared in Table 21 and Table 22. Both sets of results show that, irrespective of refinement type, the multilevel algorithms outperform the single-level versions. The difference in performance of the algorithms is most profound when the simple search is used. In this case, the average single-level results across both test suites are 64% worse than those of the multilevel scheme (see Table 22). In the case of the tabu search, the average single-level results across both test suites are 40% worse when compared to the multilevel results (see Table 21). These results demonstrate the multilevel technique's ability to aid the solution process for the CPMP.

The results also showed that the tabu search algorithms outperform their equivalent simple search counterparts. The multilevel simple search algorithm returns results 21.89% above the tabu search multilevel algorithm when the average results of both algorithms are compared. In the case of the single-level algorithms, the simple search average results were 42.94% above the tabu search average results (see Table 21 and Table 22 where these comparison are done based on the average figures across both test suites). The general findings presented above also hold if the results for each set are analysed individually, where it can be seen, both for the smaller problems of set A, and the larger problems of set B. The multilevel algorithms outperform the single-level versions (see Table 21 and Table 22).

Table 21 Results for the Osman and Christofides instances. The refinement phase uses the tabu search algorithm.

Instance	N	P	Average quality (% above opt)		Average runtime (S)	
			SL	ML	SL	ML
Set A	50	5	1.20	1.09	20.97	63.37
Set B	100	10	5.32	3.57	136.20	295.34
Average across both sets			**3.26**	**2.33**	**78.59**	**179.36**

Table 22 Results for the Osman and Christofides instances. The refinement phase uses the simple search algorithm.

Instance	N	P	Average quality (% above opt)		Average runtime (S)	
			SL	ML	SL	ML
Set A	50	5	3.55	1.43	6.53	11.71
Set B	100	10	5.76	4.25	46.82	63.09
Average across both sets			**4.66**	**2.84**	**26.68**	**37.4**

Runtime analysis: Consistent with the experience of multilevel performance in numerous areas [304], the single-level algorithm outperforms the multilevel on runtimes, in this instance by approximately a factor of two. This is true for the Osman and Christofides test cases as well as the larger San dos Campos instances (see section 5.3.4).

However, if the results are looked at across the OS instances, when the tabu search approach is used they reveal that, as the size of the instances are increased from set A to set B, the runtime for the single-level tabu search scheme increases by a factor of 6.49 (see Table 21 single-level runtime) compared to an increase by a factor of 4.66 for the multilevel version (see Table 21 multilevel runtime). The simple search algorithm sees the single-level runtime increase by a factor of 7.16 for the larger test cases of set B over that experience for set A, whereas the multilevel runtimes increase by a factor of 5.38 (see Table 21 and Table 22 runtimes). These results suggest that for increasing problem sizes the multilevel algorithm potentially scales in a manner superior to the single-level algorithm.

5.3.3.3 Detailed results for the Osman and Christofides (OS) Instances

This section presents the detailed results for the OS instances for the single-level, multilevel and their iterated versions. The solutions are refined using the tabu search approach.

The results are shown in Table 23. The first two columns give the problem instance and the optimal values (see section 9.1, p.236). Columns 3, 4, 5 and 6 compare the solution quality, represented as percentage of the solution cost above the optimal values for the SL, ML, It.SL and It.ML algorithms. The last 4 columns present the corresponding runtimes. For the It.ML algorithm the solution is iterated 5 times. For the It.SL algorithm the solution is iterated 3 times, as no improvement in the solution quality is obtained for increased iterations. Instances 1 to 10 are the instances of set A, i.e. $n = 50, p = 5$, and instances 11 to 20 are the instances of set B, i.e. $n = 100, p = 10$.

Table 23 Detailed results for the Osman and Christofides instances.

Instance	opt	Quality (% above opt) SL	ML	It.SL	It.ML	Time (s) SL	ML	It.SL	It.ML
1	713	0	0	0	0	19.26	85.79	20.36	92.48
2	740	0	0	0	0	18.09	75.10	19.34	80.10
3	751	0.26	0.26	0.26	0.26	23.65	74.87	112.34	77.71
4	651	0	0.46	0	0	9.5	64.53	10.64	230.91
5	664	1.65	1.65	1.20	1.20	26.09	65.23	223.71	233.91
6	778	0	0	0	0	8.6	61.81	9.23	64.51
7	787	8.25	0.50	5.97	0	18.10	77.54	116.71	681.45
8	820	0.85	6.82	0.85	0	25.76	36.34	71.92	560.26
9	715	0	0	0	0	32.56	47.76	33.49	50.67
10	829	0.96	1.20	0.24	0.24	28.10	44.89	116.51	97.52
11	1006	3.67	2.58	3.37	0.29	225.01	320.23	675.72	917.60
12	966	2.38	3.62	2.38	0	59.92	212.12	286.31	3556.56
13	1026	10.23	10.91	10.23	2.63	173.98	345.26	531.81	2319.78
14	982	3.15	1.52	1.12	0.61	200.37	262.59	619.19	1190.08
15	1091	3.24	0.54	3.02	0.45	139.50	276.29	413.49	703.59
16	954	3.03	3.98	3.03	0.41	185.43	269.07	475.81	3656.01
17	1034	4.15	0.19	4.15	0.19	61.20	301.15	350.80	305.63
18	1043	1.82	2.68	0.47	0.38	144.35	341.63	644.41	1152.27
19	1031	1.93	0.87	1.93	0.77	132.38	354.26	390.96	895.52
20	1005	19.60	8.85	16.81	2.38	39.90	270.76	187.82	1403.92
Average		**3.26**	**2.33**	**2.75**	**0.49**	**78.59**	**179.36**	**265.53**	**913.52**

While the iterated single-level algorithm did not lead to significant improvements in the single-level results, the multilevel results are further improved by the iterated multilevel algorithm that produces results 0.49% above the optimal values. This is comparable to the performance of some metaheuristics applied to the stated test cases [52].These results again demonstrate the success the multilevel technique had in aiding the solution process.

5.3.3.4 Algorithmic comparisons across Osman and Christofides test instances

Figure 36 shows the performance of the iterated multilevel and iterated single-level algorithms applied to the Osman and Christofides instances. The cost values are normalized with respect to the optimal values for each problem and averaged over all problems in the test suite and the runtime normalised with respect to the number of nodes for each problem and averaged over all problems in the test suite The first data point on each plot, viewing the plots from left to right, represents the cost value for the single-level and multilevel algorithm on their respective plots. Starting with these values, obtained from the single-level and multilevel algorithms, each plot then shows the quality of the solution with respects to the optimal values as the algorithms are iterated. As was the case for the CVRP, the iterated multilevel algorithm given sufficient runtime has an asymptotic performance approaching the optimal values.

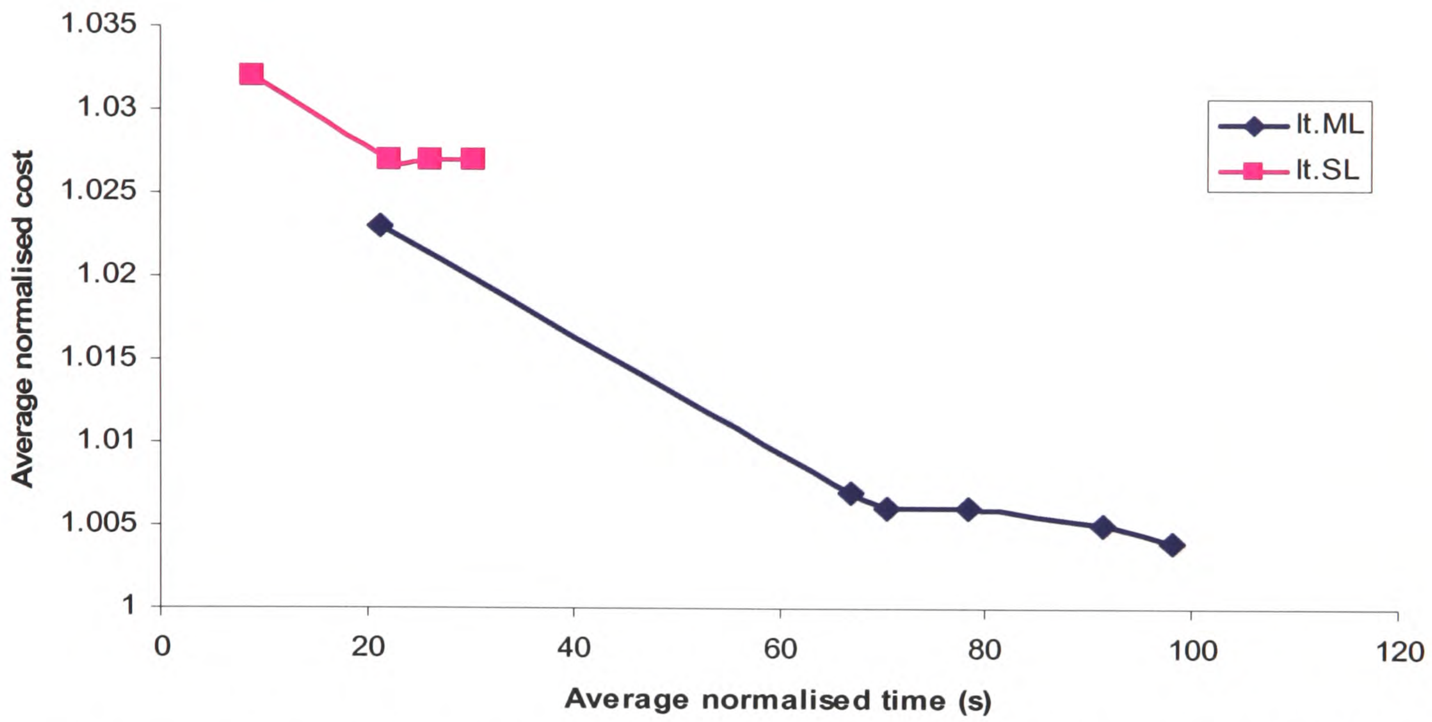

Figure 36 A performance comparisons of the iterated algorithms applied to the Osman and Christofides instances.

5.3.4 Results for San Jose dos Campos city test cases

The San Jose dos Campos city instances are the second suit of test instances used for testing the algorithms developed for the CPMP. This section presents the results of those tests for the single-level, multilevel and iterated multilevel versions. The results for the iterated single-level algorithms did not enhance the single-level results and were outperformed by the iterated multilevel results. The iterated single-level results are therefore omitted from this analysis.

The results for applying the algorithms to the San Jose dos Campos city instances are shown in see Table 24. The refinement phase uses the tabu search algorithm. The first four columns give the problem instance, the number of nodes, the number of medians and the best known solution values (*bks*) (see section 9.2, p.237). Columns 5, 6 and 7 compare the solution quality, represented as a percentage of the solution cost above the *bks* for the SL, ML and It.ML algorithms, whilst the last 3 columns present the corresponding runtimes. For the It.ML algorithms the number of iterations to search for improvements is set to 1.

Table 24 Detailed results for the San Jose dos Campos instances.

Instance	*N*	*P*	*BKS*	Quality (% above BKS)			Time (min)		
				SL	ML	It.ML	SL	ML	It.ML
SJC1	100	10	17288.99	2.45	2.45	0.79	5.99	10.05	59.76
SJC2	200	15	33270.94	4.45	4.27	0.76	27.94	85.92	487.35
SJC3a	300	25	45338.01	16.39	7.17	2.99	96.61	261.65	1067.74
SJC3b	300	30	40635.90	10.58	7.10	6.75	45.37	124.08	300.37
SJC4a	402	30	61928.91	7.26	2.74	1.79	225.10	539.99	1061.98
SJC4b	402	40	52541.72	5.23	3.22	1.66	161.75	274.85	956.25
Average				**7.73**	**4.49**	**2.46**	**93.79**	**216.09**	**655.58**

As the algorithms are applied to the larger scale instances, the multilevel algorithm outperforms the single-level algorithm by a factor of 1.72 (based on the average cost values of each algorithm see Table 24). This compares to the case of the smaller Osman and Christofides instances where the multilevel algorithm outperformed the single-level algorithm by a factor of 1.4.

For these larger scale instances, the iterated multilevel algorithm was only executed for one iteration due to runtime considerations; however, the algorithm was still able to improve upon the multilevel results. The average values returned of 2.46 % above the best known values compared with 4.49 % for the multilevel algorithm. These results show that the multilevel technique aids the solution process for these large scale CPMP instance as it did for the smaller OS instances.

5.3.5 Multilevel results compared with other solution techniques

This section gives an indication of how the quality of the solutions produced in this research compares numerically to the literature, keeping in mind the differing nature of multilevel algorithms and metaheuristics as discussed in section 1.3.

5.3.5.1 Comparisons for the Osman and Christofides instances

Some of the leading solution techniques for the Osman and Christofides instances are presented in Table 25 alongside some of the results from this research. From these results it can be seen that the multilevel technique is highly competitive with the standard in the field.

The optimal values are produced by the branch and bound procedure due to Maniezzo et al. [178] termed a Bionomic Approach to the Capacitated p-Median Problem (*BAC*). The algorithms were implemented on an IBM PC with a Pentium 166 MHz CPU in FORTRAN 77.

The following heuristic techniques from the literature included are:

- A tabu search implementation (TS), an advanced tabu search implementation (*ATS*) and a further advanced tabu search implementation (*ATS*$^+$) due to Franca et. al [91]. Franca's algorithms are implemented on a SUN Soarc20 workstation in C. The tabu search implementation (TS) uses the same local search heuristic used in this research. From the results in the table below it can be seen that the multilevel technique can significantly improve the heuristic's performance
- A hybrid simulated annealing and tabu search algorithm (*HSATS*) by Osman and Christofides [207] implemented on a VAX 860 computer in FORTRAN 77. Computation times are measured for processing only and so exclude input and output operations.

- A simulated annealing procedure (*SAP*) by Connolly [52] implemented on a VAX 860 computer in FORTRAN 77.
- A hybrid scatter search (*HSS*) heuristic due to Diaz and Fernandez [73] implemented on a Sun Blade 1000/750 coded in C.

The iterated multilevel results (It.ML) and the single level results are also presented. These algorithms are implemented in JAVA on a Pentium-4, 3GHz PC (*PC4*).

Apart from the result for *HSATS,* the computation times reported for the other algorithms do not indicate how the times are measured. Times measured for the multilevel heuristics include input and processing times. The Toward Peak Performance values [76] were not available for the platforms the solution techniques have been implemented on. However, we can see that the solution qualities of the multilevel algorithms are competitive with the standard in the literature.

Table 25 Comparison of solution approaches for the Osman and Christofides instances.

Solution Approach	Quality (% above *BAC*)	Runtimes (Min)
BAC	0	92.66
TS	3.27	-
ATS	0.047	2.65
ATS^{+}	0.004	4.52
HSATS	0.049	6.02
SAP	0.36	3.28
HSS	0.04	4.32
It.ML	0.49	15.22
SL	3.26	1.31

5.3.5.2 Comparisons for the San Jose dos Campos instances

Currently the San Jose dos Campos instances are not as much used in the literature as the OS instances. However, a couple of the leading solution techniques for the San Jose dos Campos instances are presented in Table 26. A Column generation approach using Lagrangean surrogate relaxation (*CLS*) due to Lorena and Senne [175] implemented in C on a Sun Ultra 30 WorkStation. A hybrid scatter search (*HSS*) heuristic due to Diaz and Fernandez [73] implemented on a Sun Blade 1000/750 coded in C.

Table 26 Comparisons of solution techniques for the San Jose dos Campos instances.

Solution Approach	Quality (% above HSS)	Runtimes (Min)
HSS	0	582.47
CLS	0.13	105.53
It.ML	2.46	655.58

For these instances the multilevel approach is not as competitive as the standard in the literature, but the throughout the research, results demonstrated the multilevel technique's ability to aid the solution process for the embedded local search heuristics.

5.3.6 CPMP Conclusion

As was experienced for the CVRP, across all of the test suites for the CPMP, the multilevel algorithm improved on the quality of solutions produced by the single-level algorithm. Again this demonstrates the multilevel technique's ability to aid the solution process of the local search heuristics employed. It was also demonstrated that the multilevel technique was able to further improve the performance of the local search heuristics when a tabu search configuration was employed. The results produced when the tabu search configuration was used were highly competitive with those in the literature.

Chapter VI

6 Multilevel Refinement CVRP and CPMP: an Evaluation

From the results presented in the preceding chapter, it is clear that the multilevel technique can aid the solution process for routing and location problems. In some cases, this effect is quite significant. This chapter, presented in five parts, looks at issues arising from the work done and projects the research forward.

The research objectives were accomplished and *Part I* addresses this. *Part II* addresses some general issues arising from the application of the multilevel technique to the two problem areas. *Part III* highlights the main contributions and summaries the main results. *Part IV* presents a further works section. This section outlines some novel ideas resulting from the research that are outside the scope of the current work. Conclusions are presented in *Part V*.

6.1 *Part I* – Review of research objectives

Location and routing are two areas of combinatorial optimisation that are of importance to industry and academia. Because the number of possible solutions to the types of problems encountered in these areas typically increases exponentially for linear increases in the problem sizes, exact solution techniques are limited to small-scale instances.

Industry and academia devote significant resources to improving existing solutions and solving larger problem instances. From a business perspective, being able to improve the routing or location solutions employed, provides the possibility of deriving economic benefits. In the case of direct transportation, or location analysis related services, the benefits can be derived from a service provider being able to more efficiently serve their customer base, serving a larger customer base, or being able to allocate resources more efficiently etc.

The academic interests in these problems are manifold. In addition to meeting one of academia's responsibilities, that of responding to valid commercial needs, the theoretical understanding garnered from studying these problems are, in and of themselves, worthy pursuits. Additionally, the abstract nature of the models developed in academia offers invaluable knowledge across a number of fields. The *p*-median problem, for instance, models the location of physical facilities while at the same time offering insights into data clustering, among other disciplines.

There are also societal considerations influencing the efforts devoted to these problem areas by both commercial parties and academic institutions. This is because improvements in location and routing models tend to have positive impacts on environmental and other social factors. Transportation, for instance, has been identified as a leading producer of CO_2, a major contributor to global warming. Where society has faced challenges on these scales, industry and academia has typically sought to engineer solutions. The fields of routing and location, because of the benefits accruable from improvements and the adverse effects resulting from inefficiencies, have been, and will continue to be, areas where practitioners strive for advancements. This research is concerned with advancements in solution techniques to routing and location models.

Because exact solution is impractical, heuristics and metaheuristics are the leading approaches deployed for location and routing problems. The multilevel technique has been demonstrated, throughout the research literature, to aid the solution process for combinational optimisation problems. Significantly, it has been found to be able to aid the solution process, when coupled with both heuristic and metaheuristic approaches.

However, the multilevel technique had not been applied to routing and location problems before this research.

The research was conducted against this backdrop and the results validate the conclusions that:

- Multilevel refinement has a role to play in the fields of routing and location, and also in the wider combinatorial optimisation problem areas surrounding these fields.
- The multilevel technique is capable of significantly aiding the solution process of local search procedures and metaheuristic approaches used in the fields of routing and location.
- The multilevel technique, may provide a valuable solution technique for increasing problem sizes, in the areas of vehicle routing and facility location.

The key research questions proposed at the start of the research have been answered.

The first of these questions was to determine: could the multilevel technique aid the solution process for the vehicle routing problem? The research has demonstrated that the multilevel technique does improve the solution process for the vehicle routing problem.

The other key research question was: could the multilevel technique aid the solution process for facilities location problems? Again, the research has demonstrated that this could be successfully done.

Thirdly, the research has shed new light on the multilevel technique and provided researchers with new means of constructing solutions and enhancements to the refinement process (for example, two-phase coarsening and coarsening homogeneity).

The research has also given some insights on how the multilevel technique could be utilised in other areas.

6.2 *Part II* – An evaluation of some general issues

This part of the chapter evaluates issues that relate to the multilevel technique for both routing and location problems. These issues may also be of interest to multilevel practitioners in other problems areas.

6.2.1 The effect of the quality of the initial results.

This section advocates an additional area, where a practitioner should consider using the multilevel technique. This is in the cases where, for a given problem, the existing construction heuristics return poor quality solutions. This recommendation is based on the observation that the multilevel technique outperforms the single-level algorithm by even wider margins when they are both started from initial solutions of poor quality.

There exist notable differences in the quality of the solutions produced by the multilevel and single-level algorithms when the quality of the initial solution is poor. In the case where coarsening is used to construct the initial solution for the multilevel algorithm, and an equivalent algorithm is used for the construction phase of the single-level algorithm, the initial and final solutions are typically worse in quality compared to the case where two-phase coarsening is employed[1]. This is true for both multilevel and single-level results, but in both cases the multilevel algorithm outperforms the single level versions.

[1] In the case of the single-level algorithm, only the construction phase of the two-phase coarsening approach is employed.

Table 27 Multilevel and single-level algorithms applied to the Christofides et al. instances

Method of Coarsening	Average % cost above *bks* end of coarsening		Average % cost above *bks* end of the solution		Factor By which the ML out performs SL
	SL	ML	SL	ML	
Coarsening used to create initial solution (merges implemented by savings heuristic)	81.31	81.31	20.97	8.03	2.6
Two phase Coarsening (initial solution constructed with parallel CWS)	7.63	7.63	6.37	4.11	1.6

However, in the case where the initial results are poor, the multilevel algorithm outperforms the single-level by greater margins. Table 27 shows a comparison for the CVRP where it can be seen that, starting from initial results of poor quality, the multilevel algorithm returns results 2.6 times better then the single-level algorithm. This compares to the case where the initial results were of a better quality and the multilevel results were better by a factor of 1.6.

A somewhat similar trend was seen for the CPMP, exemplified by the San Jose dos Campos instances. Where the coarsening algorithm was used to produce an initial solution for the multilevel algorithm and the equivalent construction algorithm used for the single-level version, the initial solutions were approximately 392% above the best known values, while the better performing two-phase approach produced results, approximately 27.36 % above the best known values at the end of the construction phase. In the case where the initial solutions were poorer, the multilevel results were three times better than the single-level results. This compared to the case where the initial solutions are of a much higher quality and the multilevel results were 1.7 times better than the single-level result.

The trends demonstrated by these results imply that the multilevel technique potentially possesses the ability to improve initially poor solutions better than does an equivalent single-level counterpart. This could be related to the global perspective that the technique imparts to the underlying local search algorithms. It is well known that local search algorithms are susceptible to becoming trapped in local optima [28] which are far from the global optimum. In the case where both sets of starting solutions are close to the global optimum, the effect of the single-level algorithm's weakness, that of becoming trapped in low-quality local optima, is not as obvious. The potential for the multilevel algorithm to overcome the same weakness is not as noticeable.

However, in the cases where both sets of solutions are far from the global optimum, what has been seen through repeated testing is that the multilevel technique is able to get closer to the global optimum values compared to the single-level algorithm's performance. This does not mean that the multilevel technique should be restricted to the case where suitable construction heuristics do not exist, although if it were clearly demonstrated that the technique was superior in these cases, this would be a major endorsement of the technique. The results demonstrate the ability of the multilevel technique to improve the performance of local search heuristics by shaping the search space.

6.2.2 The effect of coarsening on restricting areas of the search space

The coarsening process demarcates sections of the search space that are prohibited from being refined at given levels. It is of inertest to note that the restricted spaces are distributed (across the routes or clusters) differently when two-phase coarsening is used as opposed to the cases when coarsening is used to construct the initial solution.

When coarsening is used to construct the initial solution, the coarsening process is applied fairly evenly across the search space. For a routing problem, for example, the highest level of coarsening is typically experienced across all routes i.e. if there were y levels of coarsening there will likely be a segment coarsened at level y on each route in the solution. Additionally, there are likely to exist some routes with no segments coarsened at certain levels below y.

When two-phase coarsening is used, a number of different sets of circumstances are experienced. The highest level of coarsening is typically not experienced on each route. If the last level of coarsening is y, and for each route i in the solution the last level of coarsening is measured as x_i, then set level x^* equal to the x_i value of maximum difference between x_i and y for all values of x_i. In this case, on all routes in the solution there will exist segments coarsened at level x^* and levels below. The levels above x^* up to y and inclusive, where there is a difference between the values of x^* and y, will be experienced only by some routes in the solution.

The difference in the number of levels between x^* and y can be significant and while it means that more effort could be spent refining in some parts of the search space as opposed to others, no adverse effects on performance have been established as a result of this.

6.2.3 Approximate Refinement: a strategy for large scale problems

The issue of how to efficiently refine large problems is one of constant interest to practitioners. What has been seen in this research, and discussed in the remainder of this section, is that the multilevel technique can enhance the search space reduction techniques offered by coarsening to tackle this issue, by not fully expanding the problem at any stage during refinement. This concept, of not fully expanding the problem, we term *approximate refinement.*

Typically, the runtime used by a multilevel algorithm at level zero is less then the runtime used by the refinement phase of an equivalent single-level algorithm. However, the runtime used by the multilevel algorithm at level zero and in the immediately preceding levels is a significant amount of the runtime incurred by the multilevel algorithm during the refinement phase.

One feature that has been observed of the multilevel technique and demonstrated on the large scale routing problems is that the solution found by the multilevel technique in the upper levels of refinement is often of a quality superior to that produced by the single-level algorithm.

This suggests that an approximate refinement strategy, whereby the problem is never fully expanded during the refinement phase, could hold some benefit for the technique. This might be most effective when two-phase coarsening is deployed, ensuring that a good solution is in place and the sections of the solution fixed at the lower levels are of a high quality. Additionally, if the new method of relating levels and approximation (where the fixed edges of all levels are continually interrogated throughout the levels of refinement - see section 6.4.1) is coupled with the process of approximate refinement, this researcher expects that this will aid the multilevel technique in the solution of large-scale problems.

6.2.4 Representations effect on efficiency and accuracy

In applying the coarsening algorithm to a given problem, the way in which the coarsened approximations are represented plays a role in determining the quality of solutions produced by the multilevel technique. This sections looks at desirable characteristics for the coarsened approximations.

The series of approximations created by a coarsening algorithm for routing and location problems are required to be easier to solve than the original problem and, consequently, are represented in a manner which differentiates each approximation from the problem in its original form. The representation chosen for an approximation plays a significant role in determining how efficiently the search space can be explored during the refinement phase. In computational experimental analysis, since the efficiency of exploring solutions for given problems will have implementation dependencies, it is probably not possible to specify generic efficient representations. However, there are characteristics that seem desirable in all representations.

The representation should be such that detail is filtered from the search space, where groups of nodes are coarsened together. For example in a routing problem, location information should be filtered from the problem. The representations should also be such that they can be easily manipulated by the local search procedures being used. The representations should preserve the properties of high quality solutions over solutions of poorer quality. An analogy is the representation presenting metadata about the problem as opposed to raw data. For example, in coarsening any given number of nodes of a routing problem, the presentation chosen for this research gives one measure for their total demand, one measure for the cost of connecting the nodes in one chosen formation and two pairs of (x, y) coordinates. Representation also plays a role in influencing the accuracy of the search - in other words, the metadata presented by the chosen representation should facilitate the construction of accurate solutions to the original problem. This typically means distortions to the original problem should not be introduced by the representation: hence, exact coarsening is preferred over inexact coarsening [305].

6.2.5 Failures of the multilevel technique

This section looks at a weakness of the generic multilevel technique that practitioners should be cognisant of when applying the technique.

The multilevel technique is susceptible to accepting improvements in the upper levels that can result in a final solution of poor quality. In some instances, this solution can be worse in quality than a solution produced by an equivalent single-level heuristic. In the experiments conducted, primarily on the Irnich et al. instances [147], while the multilevel technique was observed to return worse results than the single level results on a number of occasions, overall the multilevel algorithm outperformed the single-level version.

Outside of a research environment, a practitioner typically will not engage a multilevel and single-level version simultaneously. Therefore, without best known solutions, verifying if the solutions produced by the multilevel technique suffer from being caught in local optima of poor quality in the upper levels will be difficult. One strategy that could potentially aid in reducing the infrequent (as experienced in our research) occurrence of this weakness, is projecting all proposed moves in the upper levels onto the original problem and only accepting those moves corresponding to improvements on the original problem. A similar idea has been proposed in the field of covering design, as a part of a multilevel algorithm that has produced impressive results [67].

6.3 *Part III* – Conclusions from the main results

The research developed and implemented multilevel and single-level algorithms for the capacity vehicle routing problem (CVRP) and the capacitated p-median problem (CPMP). This research constitutes the first application of the multilevel technique in these areas. The research demonstrated that for the CVRP and the CPMP, the multilevel technique provides significant gains over its single-level counterparts. In all cases, the multilevel technique was able to improve the asymptotic convergence in the quality of solutions produced by the single-level's local search heuristics.

The following two tables show the improvements the multilevel algorithms offer over their single-level counterparts, first for the CVRP and then the CPMP. The first column lists standard test suites from the literature used for the benchmarking of heuristic approaches for the relevant problem type. The second column in each table shows the factor by which the multilevel technique's results are an improvement on the results produced by the single-level heuristic. The third column of each table shows the equivalent comparisons for the iterated multilevel algorithm results, relative to the single-level heuristic results.

Table 28 A comparison of the classes of algorithms applied to CVRP test suites.

Problem suites	Factor of multilevel improvement over single-level	Factor of iterated multilevel improvement over single-level
Christofides and Eilon instances	1.14	2.49
Christofides et al. Instances	1.55	2.46
Golden Instances	1.06	1.58
Li et al. instances	1.03	-

Table 29 A comparison of the classes of algorithms applied to CPMP test suites.

Problem suites	Factor of multilevel improvement over single-level	Factor of iterated multilevel improvement over single-level
Osman and Christofides instances	1.40	6.65
San Jose dos Campos instances	1.72	3.14

From the above results, it can be seen that for the CVRP and the CPMP the multilevel technique is able to aid the solution process, in some cases significantly.

The research constructed a general framework for solving instances of the CVRP and the CPMP. The similarities between the problems, chiefly the requirement to partition the set of customers into feasible subsets while respecting the problem constraints and minimising connection costs, was exploited to this end. From these results, we can conclude that practitioners in the areas of location and routing should consider the multilevel technique and what role it can play in their attempts at devising efficient solutions.

6.3.1 Coarsening

The research designed and implemented simple ways of constructing solutions to the CVRP and the CPMP that are acquiescent to the coarsening philosophy of the multilevel technique.

The research also developed new methods of solution construction that had not previously been advocated for multilevel algorithms. These were centred on separating coarsening and solution construction. It has been demonstrated that, in many instances, solutions starting from the new methods of coarsening developed in the research were superior to solutions found based on the coarsening philosophy of using coarsening to construct the initial solutions. For example, in the case of the Christofides et al. [49] instances the new type of coarsening outperformed the old by a factor of 1.95 times. In the case of the CPMP, the new types of coarsening developed allowed the leading construction heuristics in the field to seamlessly amalgamate with the coarsening process. The fact that the CPMP is capacitated and the number of medians predetermined meant the more traditional coarsening approaches faced difficulties in constructing feasible solutions of good quality. It was therefore possible to use the technique of separating construction and coarsening in creating feasible solutions to the problem.

The work done in this research on expanding the applicability of the coarsening process is significant. Significant from the viewpoint that coarsening has posed problems for practitioners seeking to devise coarsening strategies capable of constructing feasible solutions to their problems. In some cases, practitioners could fail in their quest to use coarsening as a suitable construction technique. With the new approach, the practitioner is free to construct solutions utilising the leading construction techniques available for a given problem, thereby ensuring very good initial results and accruing the resulting decrease in computational resources extended on refinement.

Additionally, it allows the practitioner to concentrate on the representational aspects of coarsening as opposed to the solution construction aspects. This then means that the main advantages the multilevel technique offers (search space reduction, global perspective to the local search heuristics etc.) can be brought to bear on a problem without excluding the best construction heuristics in the field. It is expected that the development of two-phase coarsening in this research will play a role in increasing the uptake of multilevel refinement[1].

6.3.2 Enhancements

The research also designed and implemented effective enhancements to the multilevel technique for the CVRP and the CPMP. These enhancements were *constraint relaxation*, *coarsening homogeneity* and *solution-based recoarsening*.

For example, for the (CVRP) Christofides et al. instances [49], with constraint relaxation and coarsening homogeneity, the multilevel technique was able to produce results 1.36 times superior to the case where these enhancements were not used. The effect of solution-based recoarsening is displayed in the Table 28 and Table 29 (p. 192), illustrated by the results for the iterated multilevel algorithms[2].

---

[1] This is complemented by the fact that the concepts of two phase coarsening (that given a solution it can be coarsened) is shared by solution based recoarsening; and the success that solution based recoarsening has had seemingly on all occasions it has been utilised. This suggests that a multilevel practitioner should investigate this form of coarsening as their default option.

[2] Solution-based recoarsening forming the basis of iterated multilevel algorithms.

These enhancements to the technique demonstrated that the technique could be adapted to aid the solution process of heuristics and metaheuristics in varied settings, particularly in areas requiring added side constraints. They also demonstrated that the technique could aid its underlying local search procedures to produce results comparable to the standard in the published literature.

6.3.3 Collaborative technique for metaheuristics

In the case of the CPMP, two multilevel algorithms were implemented – one using a tabu search metaheuristic, the other relying on a simple local search procedure. Both multilevel algorithms outperformed their single-level counterparts. However, the multilevel algorithm using the tabu search metaheuristic outperformed the multilevel algorithm executing the local search procedure. This reinforced the concept that the multilevel technique can aid the solution process for metaheuristics. It is the hope of this researcher that the technique will be explored in this regard, both in academia and in industry.

6.4 *Part IV* - Further works: Multilevel refinement

The multilevel technique is a new solution approach for combinatorial optimisation problems of the type reported on in this thesis. The research presented a discussion on the wide range of areas where the technique has been successfully applied. This is with the aim of demonstrating that the technique can be customised to aid the solution process in varied settings.

While the technique, in some circumstances, aided the asymptotic convergence in runtimes and reduced the possibility of its underlying local search procedures being trapped in local optima of poor quality, there are occasions where the technique failed at these pursuits. Where these failings occur, a practitioner can be faced with two of the major weaknesses of the technique. The first is the case where the technique accepts improvements in the upper levels that result in final solutions of poorer quality than that produced by an equivalent single-level algorithm. The second occurs when the improvements produced by the technique in the upper levels of refinement are insufficient to enable the technique to sufficiently reduce the runtime expended in the lower levels. While the research has advocated strategies for tackling these weaknesses, these areas demand further research.

Other weaknesses of the technique are analysed throughout the thesis, but most important seems to be the relationship between the successful application of the technique and the objective functions of the problems that it is being applied to. It has been recognised in the literature that problems where local changes in the solution are not reflected in the global quality of the solution, are potentiality not suited to the multilevel technique [304].

However, there is also a need for further analysis on the types of problems the technique is suited for, and if there are problems it is not suited to handle. In some discrete problems conforming to graph models, it has been identified that the successful application of some heuristics can be affected by the degree to which these models satisfy the triangle inequality [234]. The multilevel algorithm, being an amalgamative process, impacts on the structure of the graphs it is applied to. It is of interest to determine: can a multilevel algorithm change the underlying structure of a problem in a manner that makes it impossible for local search heuristics to operate efficiently on the resulting graphs? This issue requires further analysis, particularly in the areas where the multilevel algorithm fails to return results superior to the single-level version.

There are numerous other new areas of research identified because of the work done during our research. These are presented in the remainder of the chapter.

6.4.1 Levels and Approximations

This section presents a new way of relating levels and approximations. The proposed method is viewed as one way of finding greater improvements in the upper levels and could possibly improve the solution process for large-scaled problems, when coupled with approximate refinement.

The coarsening algorithms implemented in this research, and as used in other multilevel implementations [302], present one main approximation per level. Each approximation specifies a maximum number of free edges allowed in a feasible solution during the refinement phase. The approximation also demarcates the only areas in the search space these free edges can be optimised for improvements. This is enforced by fixing edges in the other areas.

In the refinement phase at a given level, undoing the steps that created a given approximation then project the solution to the level below. In the case of a routing problem this corresponds to expanding all the edges fixed at level x and transforming the solution to the approximation of level x-1. Therefore, as levels are created and reversed by the fixing and freeing of edges, a level can be characterised by the number of free edges present in a feasible solution. Additionally, edges are only freed in the process of reversing a level. Since the fixed edges in the solution demarcate sections of the search space not available for refinement, and these fixed edges cannot be freed until the levels at which they were fixed are visited in the refinement phase, this can lead to a situation in which improvements are often not found in the upper levels. The tight coupling of the approximations available to the refinement algorithm and the freeing of edges means that the refinement algorithms only 'see' the main approximation created for a given level.

In light of the situation just outlined and the advocacy of two-phase coarsening presented in this thesis (two-phase coarsening emphasising the fact that the approximations are created to demarcate sections of the search space that are not to be optimised at particular levels), we are proposing a new heuristic for relating approximations and levels. The new heuristic would be employed in the refinement phase. The difference between the new heuristic and the old method of relating approximations and levels is simple. Similar to the old approach the new one adheres to the maximum number of free edges allowed at a given level. However, the new method disregards the concept that at a given level, free edges can only be experienced in the areas of the search space demarcated during coarsening.

The new method defines a shifting process such that *after all improving moves at a given level have been performed, the free edges at that level can be fixed and an equivalent number of fixed edges are freed and further improvements sought.* This is done for a given number of attempts before a new level is explored. A new level is explored by increasing the number of free edges allowed in the solution. Two key areas for the heuristic to address, therefore, are a means of determining the edges to be fixed and freed at a given level (the edges to be shifted) and a means of projecting the solution to a new level by introducing additional free edges into the solution.

The following is proposed as one means of determining how to fix and free edges at a given level: at each level, let the number of free edges allowed in a feasible solution represent the maximum number of edges that can be shifted at that level. All feasible combinations of edges in the solution can be attempted to be shifted. The size of a combination, however, must be less than or equal to the maximum number of free edges allowed at that level. Where the number of free edges in the solution exceeds the number of fixed edges, the maximum number of edges allowed to be shifted is equal to the number of fixed edges in the solution. The coarsening process would determine *the number* of fixed and free edges allowed at each level. After each shift, the refinement algorithms in place would be applied to the solution.

In projecting the solution to a new level it is proposed that if the solution is currently at level *x*, in transferring to level *x-1*, the number of edges fixed at level *x-1* in the coarsening phase is determined and an equivalent number of fixed edges in the current solution is arbitrarily chosen and freed.

In the forms of refinement used throughout this research and others [304], the transferring of the solution from one level of refinement to another results in the sections of the solution demarcated as not for optimising at that level being the same as those sections demarcated as not for optimising in the equivalent level of coarsening. In the new form of refinement outlined, the number of areas of the search space demarcated as not for optimising in the refinement and coarsening phase at a given level would be the same, but the actual areas may be different. This method of refinement should allow more improving moves to be implemented in the upper levels and is an area the researcher is excited to take forward. Coupled with the work on approximate refinement (see section 6.2.3) this should give the multilevel technique added impetus to handle large-scale problems.

6.4.2 Multilevel Refinement as part of another Refinement Strategy

Past and current research for multilevel algorithms have focused on the multilevel technique being used as a collaborative technique, but providing the over-arching philosophy of the collaboration. Another area potentially worthy of exploration is one in which a multilevel approach may be embedded as part of another metaheuristic strategy with that other strategy providing the over-arching philosophy. This might be worthwhile in the case where a metaheuristic approach is dominant for a particular problem and although a multilevel version of this metaheuristic does not outperform the metaheuristic in all instances, the search space reduction techniques of coarsening is found to be beneficial in some circumstances.

A practitioner using another technique altogether could incorporate a phase of coarsening and refinement during the improvement phase to aid the solution process. In this case the coarsening and refinement is being used more in the form of a refinement tactic as opposed to being the over-arching refinement approach. This could be applied to the entire problem or a part of the problem. Related research in applying the technique to parts of a solution has been investigated by Oduntan [198] who considered partial solution spaces.

6.4.3 Rich Routing Problems

The multilevel technique has been applied to the CVRP, the DVRP and the CPMP. While the technique remains to be applied to routing problems with additional side-constraints, the technique does demonstrate it has the capability to work with rich or complex routing and location problems, especially when coupled with constraint relaxation. Additionally, the feature selection technique proposed by Oduntan [197], where different features of the problems are solved on different approximations of the problems, presents another promising addition to the technique that potentially aids in the solution of rich problems.

A lot of the research on routing problems assumes a static operation environment. However, in some real life situations, the requirement is often for vehicle routes that change during a planning period [177]. Here again the multilevel technique possesses scope for performing well on dynamic problems, as, once feasibility is guaranteed, the technique presents a succession of solutions throughout the levels of refinement. Updating these to take into consideration a changing environment seems eminently plausible.

6.4.4 Multilevel Technique applied to other Routing and Location problems

From the work done on the VRP, a prime candidate for further investigations of the multilevel technique in routing problems appear to be the multi-depot vehicle routing problem. The multi-depot vehicle routing problem appears appropriate because of the level and clustering aspects of additional depots. These features should make the problem fit the methodology of coarsening.

A number of interesting location problems present potential opportunities for research centered on the multilevel technique. They include the competitive location problems, the uncapacitated p-median problem and capture problems. The future problems of most interest to the researcher however, are location-routing problems: a combination of the work done in the thesis.

6.4.5 Extracting data from the problem being solved.

The multilevel algorithms implemented for both routing and location problems can be described as 'stable heuristics' [100], i.e. the algorithms are decoupled from the data of the problem they are applied to. An iterative search procedure by Taillard [279] demonstrated, however, the improvements in the quality of solutions found when characteristics of the problem data are taken into account. The Taillard procedure was applied to the Christofides et al. instances and made use of the fact that the test instances are symmetric in solving the instances. While the researcher is mindful of solving test instances, as opposed to developing algorithms capable of solving classes of problems (a concerned shared by Ropke in his PhD thesis [246]), the other side of the argument has some validity. If a company is given a set of customers for whom a vehicle schedule is required and there exists stability in the customer information, the company will quite possibly take into account the customers' data in developing a suitable solution. One possible approach out of this conflict, therefore, is for a researcher to produce two versions of the algorithms, one coupled with the data, one without. Further applications of the multilevel technique should address how much consideration should be afforded to the data of the problems being solved.

6.4.6 Multilevel Technique and Self-adaptation

A number of implementation and experimental analysis issues have influenced this research. It is not possible to offer comprehensive treatment for some of these issues in the thesis as they relate primarily to optimising the implementations of the heuristics. They are therefore cases for further research.

One area however deserving of special mention is the issue of self-adaptation of the algorithms. Automatic parameter selection or tuning [144][16][210] is an area of considerable importance in the literature with proven effect on influencing the performance of algorithms. The PhD theses of Sullivan [278] and Ridge [237] offer comprehensive treatment of a number of the issues. An automatic schedule has not been developed for this research and instead a heuristics testing suite was developed that fixed each parameter individually and searched for the best setting around those until a suitable set of parameters emerged. It would be of interest for further research, however, to investigate the effects of tuning on the performance of multilevel algorithms, applied in routing and location.

6.4.7 Representation and the expansion process

An area where the multilevel technique can be improved is how it makes use of *memory of the search* process. There is no provision in the generic multilevel algorithm to explicitly make decisions based on past search history. This can lead to wasted computational resources among other problems. This section proposes that a tabu search mechanism should be added to the generic multilevel algorithm to provide the techniques with explicit means of exploiting memory of the search process.

As the solution process moves from one approximation to the next in the refinement phase, typically the expansion process imparts a diversification effect on the solution. As can be deduced from the section on levels and approximation (section 6.2.2), in transferring from one approximation to another, the expansion process is not necessarily applied to all areas of the coarsened problem. Consequently, the diversification effect of expansion is not felt in all areas of the search space and in the areas where expansion is not applied, potentially wasted computational effort will accrue.

In instances like this, a tabu search element to the multilevel technique can have the effect of reducing this wasted computational effort by noting areas of the search space where expansion has not taken place and restricting expansion dependent improvement in these areas. This would be similar to the effect described for location problems (see section 4.4.3.3), where it was outlined that a tabu search mechanism can play a significant role in reducing computational effort as the solution moved from one approximation to another.

6.4.7.1 Tabu search mechanism: expansion and intensification

The generic multilevel algorithm does not include a mechanism for explicitly handling intensification. The tabu search mechanism, used as a part of the multilevel algorithm deployed for the CPMP, provided a means for managing diversification and intensification. From the performance of this algorithm compared to the case where it was not used (see section 5.3.3.2) it seems a worthwhile idea to investigate, coupling a tabu search mechanism with the generic multilevel algorithm as a means for managing diversification and intensification.

A tabu search mechanism presents one means of managing diversification and intensification in a controlled manner. However, the expansion process can have the effect of moving the search back to a tabu-ed area of the search space. This result from the expansion process replacing tabu-ed attributes in the solution with the segments from which they are made. To overcome this problem a partial score mechanism could be added to the tabu search, such that when coarsened attributes that are tabu-ed are expanded, the constituting attributes become partially tabu-ed. The weight applied to the partially tabu-ed attributes can be used in a manner similar to the tolerance parameter used for the CPMP, to control diversification and intensification.

6.4.8 Closing thoughts on further works

Across both problems both routing and location problems, two-phase coarsening lead to better initial and final solutions compared to the standard form of coarsening presented in this thesis. However, where the initial solutions are of poor quality, the multilevel technique outperforms the single-level technique by larger margins. In cases where the initial solutions are poor in quality therefore, a practitioner should consider investigating the multilevel technique.

While the multilevel technique has emerged as an attractive solution approach, there are still areas in the technique that future research should focus on. The one with perhaps the most significance is investigation into a new approach to relating levels and approximations. This has the potential to fundamentally change the dynamics of the technique and make it even more suited to solving large-scale problems.

6.5 *Part V* - Concluding

Routing and location analysis are important economic and academic pursuits. Efficient solutions to routing and location models can lead to economic benefits and significant theoretical understanding. The leading methods for solving these models use heuristic approaches.

The multilevel technique is a collaborative technique capable of aiding the solution process of heuristic approaches. We have demonstrated that the multilevel technique can aid the solution process for heuristic approaches employed in the fields of routing and location. We have further demonstrated that the collaborative process can be sufficiently decoupled, allowing the leading construction heuristic(s) in an area where the technique is being applied, to be incorporated into the technique. Consequently, the multilevel technique has a role to play in industry and academia in producing high quality solutions to routing and location models. This could potentially lead to added economic benefits and provide new ways of understanding prevailing solution techniques.

REFERENCES

[1] Aarts, E. Korst, J. and Laarhoven, P.(1997) Simulated annealing. In Local Search in Combinatorial Optimization, Aarts, E and Lenstra, J. Eds. Wiley-Interscience, Chichester, England, pp. 91–120.

[2] Adams, T. Ebeling, M. Gardner, R. Lindquist, P. Stewart, R. Szymkowski, T. Hecke, S. Voderembse, M and Wittwer, E. (2007) Upper Midwest Freight Corridor Study - Phase II, Midwest Regional University Transportation Center University of Wisconsin, Ohio Department of Transportation and the US Department of Transportation, Federal Highway Administration, pp. 1 - 161

[3] Ahmadi, S. and Osman, I. (2005) Greedy random adaptive memory programming search for the capacitated clustering problem, European Journal of Operational Research, Vol. 162, pp. 30 – 44

[4] Ahn, H. Cheng, S. Cheong, C. Golin, M and Oostrum, R (2003) Competitive facility location the Voronoi game, Theoretical computer science, Vol. 310, pp. 457 – 467

[5] Alinel, IK. And Oncan, T (2005) A new enhancement of the Clark and Wright savings heuristics for the capacitated vehicle routing problem, Journal of the Operational Research Society, Vol.56, pp. 954–961.

[6] Alp, O. Erkut, E. and Drezner, D. (2003) An efficient genetic algorithm for the p-median problem, Annals of Operations Research, Vol. 122, pp. 21 – 42.

[7] Alonso, F. Alvarez, M. and Beasley, J. (2007) A tabu search algorithm for the periodic vehicle routing problem with multiple vehicle trips and accessibility restrictions, Journal of the Operational Research Society, advance online publication; doi: 10.1057/palgrave.jors.2602405

[8] Anderson, S. (1988) Equilibrium existence in a linear model of spatial competition, Economica, Vol. 55, pp. 479 – 491

[9] Angel, E. (2006) A Survey of Approximation Results for Local Search Algorithms, In: Efficient Approximation and Online Algorithms: Lecture Notes in Computer Science, Springer Berlin, pp. 30-73, DOI 10.1007/11671541_2

[10] Arostegui, M. Kadipasaoglu, S. and Khumawal, B. (2006) An empirical comparison of Tabu Search Simulated Annealing and Genetic Algorithms for facilities location problems, Int. J. Production Economics, Vol. 103, pp. 742 – 754

[11] Augerat, P. Belenguer, J.M. Benavent, E. Corberan, A. Naddef, D. and Rinaldi, G. (1995) Computational results with a branch-and-cut code for the capacitated vehicle routing problem, Research report RR949-M. ARTEMIS-IMAG, France

[12] Barnard, S. and Simon, H. (1994) A fast multilevel implementation of recursive spectral bisection for partitioning unstructured problem, Concurrency practice and experience, Vol. 6, No. 2, pp. 101 - 117

[13] Barr, R. Golden, B. Kelly, J. Resende, M. and Stewart, W.(1995) Designing and reporting on computational experiments with heuristic methods, Journal of heuristics, Vol. 1, No.1, pp. 9 – 32

[14] Barrot, J. (2006) Fear of change is Europe's transport roadblock, Europe'S World, Autumn 2006 Issue

[15] Battiti, R. Bertossi, A. and Cappelletti, A. (1999) Multilevel Reactive Tabu Search for Graph Partitioning, Preprint UTM 554, Dip. Mat., Univ. Trento, Italy.

[16] Battiti, R. (1996) Reactive Search: Toward Self--Tuning Heuristics In. Modern Heuristic Search Methods, editor V. J. Rayward Smith and I. H. Osman and C. R. Reeves and G. D. Smith, John Wiley & Sons Ltd., Chichester, pp. 61 - 83

[17] Baum, E. (1986) Toward practical neural computation for combinatorial optimization problems, In: Denker, J. (Ed.), Neural Networks for Computing, American Institute of Physics.

[18] Bazgan, C. Hassin, R and Monnot, J. (2005) Approximation algorithms for some vehicle routing problems, Discrete Applied Mathematics, Vol. 146, pp. 27 – 42

[19] Beltrami, E. and Bodin, L. (1974) Networks and vehicle routing for municipal waste collection, Networks Vol. 4, No. 1, 65 – 94.

[20] Berman, O. Hodgson, M. and Krass, D. (1995) Flow intercepting models. In: Drezner, Z. (Ed.), Facility Location: A Survey of Applications and Methods, Springer - Verlag, pp. 427– 452.

[21] Bianchessi, N. and Righini, G. (2007) Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery, Computers & Operations Research, Vol. 34, No. 2, pp. 578 – 594

[22] Bianchi, L. Birattari, M. Chiar, M. Manfrin, M. Paquete, L. Rossi-doria, O. and Schiavinotto, T. (2004) Metaheuristics for the vehicle routing problem with stochastic demands, Proceedings of the 8th International Conference on Parallel Problem Solving from Nature, Vol. 3242 Lecture Notes in Computer Science, pp. 450 – 460

[23] Blasum, U. Hochstattler,W. (2002) Application of the branch-and-cut method to the vehicle routing problem, Technical report, Universitat zu Koln.

[24] Brandeau, M. and Chin, S. (1989) An overview of representative problems in location research, Management Science, Vol. 35. No.6, pp. 654 - 674

[25] Brandt, A. (1977) Multilevel Adaptive Solutions to Boundary Value Problems, Math. Comp, vol.31, pp.333-390.

[26] Brandt, A. (1988). "Multilevel Computations: Review and Recent Developments." In S.F. McCormick (ed.), Multigrid Methods: Theory, Applications, and Supercomputing, Proc. of 3rd Copper Mountain Conf. Multigrid Methods, Lecture Notes in Pure and Applied Mathematics, Vol. 110. New York: Marcel Dekker, pp. 35–62.

[27] Breedam, A. (1995) Improvement heuristics for the Vehicle Routing Problem based on simulated annealing, European Journal of Operational Research, Vol. 86, No. 3, PP. 480-490

[28] Blum, C. and Roli, A. (2003) Metaheuristics in combinatorial optimization overview and conceptual comparison, ACM Computing Surveys, Vol. 35, No. 3, pp. 268 - 308.

[29] Boctor, F.F. and Renaud, J. (2000), The column-circular subsets-selection problem: complexity and solutions, Computers & Operations Research, vol.27, pp. 383 – 398.

[30] Bodin, L. Golden, B. Assad, A. and Ball, M (1983) Routing and Scheduling of Vehicles and Crews - The State of the Art, Computers and Operations Research, Special Issue, Vol. 10, No. 2, pp. 63 – 212

[31] Bodin, L. (1990) Twenty Years of Routing and Scheduling, Operations Research, Vol. 38, No. 4, pp. 571-579.

[32] Bouhmala, N. (2004) Combining local search and genetic algorithms with the multilevel paradigm for the traveling salesman problem. In Blum, C. et al., editors, Proc. Ist Intl workshop on hybrid Metaheuristics

[33] Boyar, J. Epstein, L. Favrholdt, L. Kohrt, J. Larsen, K. Pedersen, M. WZhlk, S (2006) The maximum resource bin packing problem, Theoretical Computer Science, Vol. 362 , pp. 127 – 139

[34] Bozkaya, B. Erkut, E. and Laporte, G. (2003) A tabu search heuristic and adaptive memory procedure for political districting, European Journal of Operational Research, Vol. 144, No. 1, pp. 12 – 26.

[35] Bräysy, O. and Gendreau M. (2005) Vehicle routing problem with time windows, part II: Metaheuristics, Transportation Science, Vol. 39, No. 1, pp.119 – 139.

[36] Bräysy, O. and Gendreau M. (2005) Vehicle routing problem with time windows, part I: Route Construction and Local Search Algorithms. Transportation Science, Vol. 39, No. 1, pp.104 – 118.

[37] Bui, T. and Jones, C. (1993) A heuristic for reducing fill-in sparse matrix factorization, In Sincovec, R. et al., editors, Parallel processing for scientific computing, pp. 445 – 452.

[38] Cabello, S. Díaz-Báñez, J. M. Langerman, S. Seara, C. and Ventura, I. (2005) Reverse facility location problems, 17th Canadian Conference on Computational Geometry, Windsor (Canada).

[39] Campbell, A. and Savelsbergh, M. (2004) Efficient Insertion Heuristics for Vehicle Routing and Scheduling Problems, TRANSPORTATION SCIENCE, Vol. 38, No. 3, pp. 369–378

[40] Captivo, E. (1991) Fast primal and dual heuristics for the p-median location problem, European Journal of Operational Research, Vol. 52, pp. 65 – 74.

[41] Cardoso, A. (2000) Wage Differentials across Firms: An Application of Multilevel Modelling, Journal of Applied Econometrics, Vol. 15, No. 4. pp. 343-354.

[42] Caseau, Y. and Laburthe, F. (1999), Heuristics for Large Constrained Vehicle Routing Problems, Journal of Heuristics, vol.5, pp. 281-303.

[43] Cerny, V. (1985), Thermodynamical Approach to the Traveling Salesman Problem: An Efficient Simulation Algorithm, Journal of Optimization Theory and Applications, Vol. 45, pp. 41–51.

[44] Chan, W. Zhang, Y. Fung, S. Ye, D. and Zhu, H. (2007) Efficient algorithms for finding a longest common increasing subsequence, Journal of Combinatorial Optimization, Vol. 13, No. 3, pp. 277-288.

[45] Chao, I. Golden, B. and Wasil, E. (1993) new heuristic for the multi-depot vehicle routing problem that improves upon best-known solutions, American Journal of Mathematical and Management Sciences, Vol.13, No.3-4, pp. 371 - 406.

[46] Chiyoshi, F. and Galvao, D. (2000) A statistical analysis of simulated annealing applied to the p-median problem, Annals of Operations Research, Vol. 96, pp. 61 – 74.

[47] Chow, F. and Hennessy, J. (1990) The priority-based coloring approach to register allocation, ACM Transactions on Programming Languages and Systems, Vol. 12, pp. 501 – 36

[48] Christofides, N. and Eilon, S. (1969) An algorithm for the vehicle dispatching problem. Operational Research Quarterly, Vol. 20, No. 3, pp. 309 - 318

[49] Christofides, N. Mingozzi, A. and Toth, P. (1979) The Vehicle routing problem. In Combinatorial Optimization. pp. 315-338. Wiley, Chichester.

[50] Church, R. and Garfinkel (1978) Locating an Obnoxious Facility on a network, Transportation Sci., Vol.12, pp. 107 - 118

[51] Clark, G. and Wright, J.W. (1964) Scheduling vehicles from a central depot to a number of delivery points, Oper. Res. vol. 12, pp. 568 – 581.

[52] Connolly, D. (1990) An improved annealing scheme for the QAP. European Journal of Operational Research, Vol. 46, pp. 93-100.

[53] Connolly, D. (1992) General Purpose Simulated Annealing, The Journal of the Operational Research Society, Vol. 43, No. 5, Mathematical Programming in Honour of Ailsa Land. pp. 495 - 505

[54] Cordeau, J. Laporte, G. and Mercier, A. (2001) A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows, The Journal of the Operational Research Society, Vol. 52, No. 8, pp. 928 - 936.

[55] Cordeau, J. Gendreau, M. Laporte, G. Potvin, J. and Semet, F. (2002) A guide to vehicle routing heuristics, Journal of Operational Research Society, Vol. 53, pp. 512–522.

[56] Cordeau, J. Desaulniers, G. Desrosiers, J. Solomon, M, and Soumis F. (2002) VRP with time windows. In: Toth, P. and Vigo, D. editors: The vehicle routing problem, SIAM monographs on discrete mathematics and applications, Vol. 9. Philadelphia: SIAM; pp. 157–193.

[57] Cormen, T. Leiserson, C. and Rivest, R. (1990) Introduction to algorithms. Cambridge, MA: MIT Press.

[58] Cornuejols, G. Fisher, M. and Nemhauser, G. (1977) Location of bank accounts to optimize float: an analytical study of exact and approximate algorithms, Management Science, Vol. 23, pp. 789 - 810

[59] Cornuejols, G. Nemhauser, G. and Wolsey, L. (1990) The uncapacitated facility location problem. In: Mirchandani, P. Francis, R. (Eds.), Discrete Location Theory, Wiley- Interscience, New York, pp. 119 – 171.

[60] Correa, E. Steiner, M. Freitas, A. and Carnieri, C. (2004) A genetic algorithm for solving a capacitated p-median problem, Numerical Algorithms, Vol. 35, pp. 373 – 388

[61] Crainic, T. and Laporte, G. (1997) Planning Models for Freight Transportation, European Journal of Operational Research, Vol. 97, No. 3, pp. 409 — 438

[62] Crainic, T. Li, Y. and Toulouse, M. (2006) A first multilevel cooperative algorithm for capacitated multicommodity network design, Computers and operations research, Vol. 33, No. 9

[63] Croes, G. (1958) A Method for Solving Traveling Salesman Problems, Operations Research, Vol. 6, No. 6, pp. 791 - 812.

[64] Current, J. and Schilling, D. (1987) Elimination of source A and B errors in p-median location problems, Geographical Analysis, Vol. 19, pp. 95 – 110.

[65] Current, J. and Schilling, D. (1990) Analysis of errors due to demand data aggregation in the set covering and maximal covering location-problems, Geographical Analysis, Vol. 22, pp. 116 – 126.

[66] Current, J. Daskin, M, and Schilling, D. (2002) Discrete network location models. In: Drezner, Z. Hamacher, H. (Eds.), Facility Location: Applications and Theory. Springer-Verlag, Berlin, pp. 81 –118.

[67] Dai, C. Li, P. and Toulouse, M. (2005) A Cooperative Multilevel Tabu Search Algorithm for the Covering Design Problem, Artificial Evolution, Vol. 3871, pp. 119 - 130

[68] Dantzig, G. and Ramser, J. (1959) The Truck Dispatching Problem, Management Science, Vol. 6, No. 1, pp. 80-91.

[69] Daskin, M. and Stern, E. (1981) A hierarchical objective set covering model for emergency medical service vehicle deployment, Transportation Sci., Vol. 15, pp. 137 - 152

[70] Daskin, M. (1995) Network and Discrete Location. Wiley, New York.

[71] Dessouky, M. Rahimi, M. and Weidner, M (2003) Jointly optimizing cost service and environmental performance in demand-responsive transit scheduling, Transportation Research Part D, Vol. 8 pp. 433 – 465

[72] Dethloff, J. (2002) Relation between vehicle routing problems: an insertion heuristic for the vehicle routing problem with simultaneous delivery and pick-up applied to the vehicle routing problem with backhauls, Journal of the Operational Research Society, Vol. 53, pp 115 – 118

[73] Diaz, J. and Fernandez, E. (2006) Hybrid scatter search and path relinking for the capacitated p-median problem, European Journal of Operational Research, Vol. 169 pp. 570 – 585

[74] Ding, C. He, X. Zha, H. Gu, M. and Simon, H. (2001) A min-max cut algorithm for graph partitioning data clustering, In ICDM 2001, Proceedings IEEE International Conference on Data Mining, pp. 107–114.

[75] Domschke, W. and Krispin, G. (1997) Location and layout planning A survey, OR Spektrum, Vol. 19 pp. 181 – 194.

[76] Dongarra, J. (2007) Performance of Various Computers Using Standard Linear Equations, Software Electrical Engineering and Computer Science Department University of Tennessee Knoxville, CS - 89 – 85

[77] Dorigo, M. and Gambardella, L. (1997). Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem, IEEE Transactions on Evolutionary Computation Vol. 1, pp. 53–66.

[78] Drezner, Z. (1995) Facility Location. A Survey of Applications and Methods. Springer, New York.

[79] Drezner, Z. and Hamacher, H. (2002) Facility Location: Applications and Theory. Springer, Berlin.

[80] Drezner, Z. Klamroth, K. Schobel, A. and Wesolowsky, G. (2002) The Weber problem. In: Drezner, Z. Hamacher, H. (Eds.), Facility Location: Applications and Theory, Springer-Verlag, Berlin, pp. 1 – 36.

[81] Dueck, G. (1993) New optimization heuristics: the great deluge algorithm and the record-to-record travel, Journal of Computational Physics, Vol.104, pp. 86 – 92.

[82] Edwards, N. (2001) Approximation algorithms for the multilevel facility location problem, PhD Thesis, Faculty of the Graduate School, Cornell University.

[83] Eiselt, H. (1992) Location Modeling in Practice, American Journal of Mathematical and Management Sciences, Vol. 12, pp. 3 – 18.

[84] Eiselt, H. Laporte, G. and Thisse, J. (1993) Competitive location models: A framework and bibliography, Transportation Science, Vol. 27, pp. 44 – 54.

[85] Eiselt, H. Laporte, G. (1995) Objectives in location problems. In: Drezner, Z. (Ed.) Facility Location: A Survey of Applications and Methods. Springer-Verlag, Berlin, pp. 151 – 180.

[86] Erkut, E. and Bozkaya, B. (1999) Analysis of aggregation errors for the p-median problem, Computers & Operations Research, Vol. 26, pp. 1075 – 1096

[87] Feldman, E. Lehrer, F. and Ray, T. (1966) Warehouse locations under continuous economies of scale, Management Science, Vol. 12, pp. 670 – 684.

[88] Fisher, M. and Jaikumar, R. (1981) A generalized assignment heuristic for the vehicle routing problem, Networks, Vol. 11, No. 2, pp. 109 – 124.

[89] Fleszar, K. and Hindi, K. (2007) An effective VNS for the capacitated p-median problem, European Journal of Operational Research, doi:10.1016/j.ejor.2006.12.055

[90] Framinan, J. M. Leisten, R. and Ruiz- Usano, R. (2005) Comparison of heuristics for flowtime minimisation in permutation flowshops. Computer and Oper. Res. vol. 32, pp. 1237 – 1254

[91] Franca, P. Sosa, N. and Pureza, V. (1999) An adaptive tabu search algorithm for the capacitated clustering problem, International transaction in operation research, Vol. 6, pp. 665 – 678.

[92] Francis, R. Lowe, T. and Tamir, A. (2000) Aggregation error bounds for a class of location models, Operations Research , Vol. 48, pp. 294 – 307.

[93] Francis, R. Lowe, T. and Tamir, A. (2002) Demand point aggregation for location models, In: Drezner, Z. Hamacher, H. (Eds.), Facility Location, Applications and Theory, pp. 207 – 232

[94] Francis, R. Lowe, T. Tamir, A. (2003) Worst-case incremental analysis for a class of p-facility location problems, Networks, Vol. 39, pp. 139 – 143.

[95] Francis, R. Lowe, T., Rayco, B. and Tamir. A, (2005) Aggregation error for location models: survey and analysis. Working paper. Department of Industrial and Systems Engineering, University of Florida, Gainesville. http://www.ise.ufl.edu/francis/download/Survey.pdf

[96] Foster, B. and Ryan, D. (1976) An Integer Programming Approach to the Vehicle Scheduling Problem, Operational Research Quarterly, Vol. 27, No. 2, pp. 367-384.

[97] Foulds, L. (1983) The Heuristic Problem-Solving Approach, The Journal of the Operational Research Society, Vol. 34, No. 10, pp. 927-934.

[98] Funke, B. Grunert, T. and Irnich, S. (2005) Local Search for Vehicle Routing and Scheduling Problems: Review and conceptual integration, Journal of Heuristics, vol. 11, pp. 267 -306.

[99] Gabzewicz, J. and Thisse, J. (1989) Spatial competition and location of firms, Fundamentals of Pure and Applied Economics, Vol. 5, pp. 2 – 37.

[100] Gan, H. and Wirth, A (2007) Heuristic stability: A permutation disarray measure, Computers & Operations Research, Vol.34, No. 11, pp. 3187 – 3208

[101] Garcia-Lopez, F. Melian Batista, B. Moreno Perez, J. and Moreno Vega, J. (2003) Parallelization of the scatter search for the p-median problem, Parallel Computing, Vol. 29 No. 5, pp. 575– 589.

[102] Garey, M and Johnson, D. (1979) Computers and intractability: a guide to the theory of NP-completeness, New York: W. H. Freeman.

[103] Garey, M and Johnson, D. (1999) Computers and Intractability: A Guide to the Theory of NP-Completeness, 21st ed.,W. H. Freeman and Company, New York

[104] Gaskell, T. (1967) Bases for vehicle fleet scheduling, Operation Research Quarterly, Vol.18, No. 3, pp. 281 – 295.

[105] Gen, M. Tsujimura, Y. and Ishizaki, S. (1996) Optimal design of a star-LAN using neural networks, Computers and Industrial Engineering, Vol. 31, pp. 855–859.

[106] Gendreau, M. (2002), Recent Advances in Tabu Search In C.C. Ribeiro and P. Hansen (eds.), Essays and Surveys in Metaheuristics, Boston: Kluwer, pp. 369–378.

[107] Gendreau, M. (2003), An Introduction to Tabu Search. In F. Glover and G. Kochenberger (eds.), Handbook of Metaheuristics. Boston: Kluwer, pp. 37–54.

[108] Gendreau, M. Hertz, A. and Laporte, G (1994) A Tabu Search Heuristic for the Vehicle Routing Problem, Management Science, Vol. 40, No. 10, pp. 1276 - 1290.

[109] Gendreau, M. Laporte, G. and Potvin, J. (2002) Metaheuristics for the capacitated VRP. In: Toth P and Vigo D (eds). The Vehicle Routing Problem. SIAM Monographs on Discrete Mathematics and Applications. SIAM Publishing: Philadelphia, PA, pp 129–154.

[110] Gendreau, M. and Potvin, J. (2005) Metaheuristics in Combinatorial Optimization, Annals of Operations Research, Vol. 140, No. 1, pp. 189 – 213

[111] Ghosh, D. (2003) Neighborhood search heuristics for the uncapacitated facility location problem, Journal of Operational Research, Vol. 150, pp. 150 – 162

[112] Gillett, B. and Miller, L (1974) A Heuristic Algorithm for the Vehicle-Dispatch Problem, Operations Research, Vol. 22, No. 2, pp. 340-349.

[113] Glover, F. (1977), Heuristics for Integer Programming using Surrogate Constraints, Decision Sciences, Vol. 8, pp. 156–166.

[114] Glover, F. (1986), Future Paths for Integer Programming and Links to Artificial Intelligence, Computers & Operations Research, Vol. 13, pp. 533–549.

[115] Glover, F. (1989), Tabu Search—Part I, ORSA Journal on Computing Vol. 1, pp. 190–206.

[116] Glover, F. (1990), Tabu Search—Part II, ORSA Journal on Computing Vol. 2, pp. 4–32.

[117] Glover, F. (1996) Ejection chains, reference structures and alternating path methods for traveling salesman problems, Discrete Applied Mathematics, Vol. 65, No. 1, pp. 223-253

[118] Glover, F. (1997), Tabu Search and Adaptive Memory Programming—Advances, Applications and Challenges. In R.S. Barr, R.V. Helgason and J.L.Kennington (eds.), Advances in Metaheuristics, Optimization and Stochastic Modeling Technologies. Boston: Kluwer, pp. 1–75.

[119] Glover, F. and Kochenberger, G. (eds.) (2003). Handbook of Metaheuristics, Boston: Kluwer.

[120] Golden, B. Bodin, L. Doyle, T. and Stewart, W. (1980) Approximate Traveling Salesman Algorithms, Operations Research, Vol. 28, No. 3, pp. 694 - 711

[121] Golden, B. and Assad, A (1986) Perspectives on Vehicle Routing: Exciting New Developments, Operations Research, Vol. 34, No. 5, pp. 803-810.

[122] Golden, BL. Wasil, EA. Kelly JP, and Chao, IM. (1998) The impact of metaheuristics on solving the vehicle routing problem: algorithms, problem set, and computational results, In: Crainic, T. Laporte, G. (Eds.), Fleet management and logistics, Kluwer, Boston, MA. pp. 33-56.

[123] Golden, B. Assad, A. Levy, L. and Gheysens, F. (1984) The fleet size and mix vehicle routing problem, Computers & Operations Research, Vol. 11, No.1, pp. 49-66

[124] Guha, S. (2000) Approximation algorithms for facility location problems, PhD Thesis, Department of computer science and the committee on graduate studies, Stanford University.

[125] Hadjiconstantinou, E. Christofides, N. and Mingozzi, A. (1995). A new exact algorithm for the vehicle routing problem based on q-paths and k-shortest paths relaxations. In: Gendreau, M .and Laporte, G. (eds). Freight Transportation. Baltzer: Amsterdam, pp 21 – 43.

[126] Hakimi, S. (1964) Optimal locations of switching centres and the absolute centres and medians of a graph, Operations Research, Vol. 12, pp. 450 – 459.

[127] Hakimi, S. (1965) Optimal distribution of switching centres in a communication network and some related theoretic graph theoretic problems, Operations Research Vol. 13, pp. 462 – 475

[128] Hakimi, S. (1983) On locating new facilities in a competitive environment, European Journal of Operational Research, Vol. 12, pp. 29 – 35.

[129] Hale, T. and Moberg, C. (2003) Location Science Research: A Review, Annals of Operations Research, Vol. 123, pp. 21 – 35

[130] Hall, N and Posner, M (2004) Sensitivity Analysis for Scheduling Problems, Journal of Scheduling, Vol. 7, No. 1, pp. 49 – 83

[131] Hansen, P. and Mladenovic´, N. (1997) Variable neighbourhood search for the p-median, Location Science, Vol. 5, pp. 207 – 226.

[132] Hansen, P. and Jaumard, B. (1997) Cluster analysis and mathematical programming, Mathematical Programming, Vol. 79, pp. 191 – 215.

[133] Harel, D. and Koren, Y. (2002) A fast Multi-Scale Algorithm for Drawing Large Graphs, Journal of Graph Algorithms and Applications, Vol. 6, No. 3, pp. 179 – 202.

[134] Hartel, P. and Moreau, L. (2001) Formalizing the Safety of Java, the Java Virtual Machine and Java Card, ACM Computing Surveys, Vol. 33, No. 4, pp. 517 – 558

[135] Helsgaun, K. (2002) An effective implementation of the Lin-Kernighan traveling salesman heuristic, European Journal of Operational Research, Vol. 126, No. 1, pp. 106 – 130.

[136] Hendrickson, B. and Leland, R (1995) A multilevel algorithm for partitioning graphs, In Karin, S, editor, proceedings supercomputing '95, San Diego, ACM Press, New York.

[137] Hertz, A. and Widmer, M. (2003) Guidelines for the use of meta-heuristics in combinatorial optimization, European Journal of Operational Research, Vol. 151, No. 2, pp. 247 – 252

[138] Hillsman, E. and Rhoda, R. (1978) Errors in measuring distances from populations to service centers, Annals of Regional Science Association, Vol. 12, pp. 74 – 88.

[139] Hindi, K. and Pienkosz, K. (1999). Efficient solution of large scale single-source capacitated plant location problems, Journal of the Operational Research Society, Vol. 50, pp. 268 –274.

[140] Hjorring, C. (1995) The vehicle routing problem local search metaheuristics, PhD thesis, Department of Engineering Science, The University of Auckland.

[141] Hodgson, M. and Neuman, S. (1993) A GIS approach to eliminating source C aggregation error in p-median models, Location Science, Vol. 1, pp. 155 – 170

[142] Holland, J. (1975), Adaptation in Natural and Artificial Systems, The University of Michigan Press.

[143] Hotelling, H. (1929) Stability in Competition, Economics Journal, Vol. 39, pp. 41 – 57

[144] Hutter, F. Hoos, H. Stützle, T. (2007). Automatic Algorithm Configuration based on Local Search. In Proc. of the Twenty-Second Conference on Artifical Intelligence (AAAI '07), pp. 1152 - 1157.

[145] INPE – (last update 2007) http://www.lac.inpe.br/~lorena/instancias.html. (last accessed march 2008)

[146] International TRADE Administration, Department of commerce USA, (Updated March 2008) http://trade.gov/investamerica/logistics.asp (Accessed March 2008)

[147] Irnich, S. Funke, B. and Grunert, T.(2006) Sequential search and its application to vehicle routing problems, Computer and operations research, Vol. 33, No. 8, pp. 2405- 2429

[148] Johnson, D. (2002) A theoretician's guide to the experimental analysis of algorithms, In Goldwasser, M. Johnson, D. and McGeoch, C. editors, Data Structures, Near Neighbor Searches, and Methodology: Fifth and Sixth DIMACS Implementation Challenges, pp. 215 - 250. American Mathematical Society, Providence.

[149] Kariv, O. and Hakimi, S. (1979). An algorithmic approach to network location problems. Part I: The p-centers, SIAM Journal of Applied Mathematics, Vol. 37, pp. 513 – 538.

[150] Kariv, O. and Hakimi, S. (1979) An algorithmic approach to network location problems Part II: The p-median, SIAM Journal of Applied Mathematics, Vol. 37, pp. 539 – 560.

[151] Karypis, G. and Kumar, V. (1998) A fast and high quality multilevel Scheme for partitioning Irregular Graphs, Siam journal on scientific computing, Vol. 20, No. 1, pp. 359 - 392

[152] Karypis, G. and Kumar, V. (1998) Multilevel k-way Partitioning Scheme for Irregular Graphs, Journal of Parallel and Distributed Computing, Vol. 48, pp. 86 - 129

[153] Karypis, G. Han, E. and Kumar,V. (1999) Multilevel refinement for hierarchical clustering. Technical Report TR-99-020, Department of Computer Science, University of Minnesota, Minneapolis

[154] Kernighan, B. and Lin, S. (1970) An efficient heuristic for partitioning graphs, Bell Systems Technical Journal, Vol. 49, pp. 291 – 308.

[155] Kilby, P. Prosser, P. and Shaw, P. (2000) A Comparison of Traditional and Constraint-based Heuristic Methods on Vehicle Routing Problems with Side Constraints, Constraints, vol,. 5, pp. 389–414.

[156] Kirkpatrick, S. Gelatt, C. and Vecchi, M. (1983), Optimization by Simulated Annealing, Science, Vol. 220, pp. 671 – 680.

[157] Kochetov, Y. Alekseeva, E. Levanova, T. and Loresh, N. (2005) Large neighborhood search for the p-median problem, Yugoslav Journal of Operations Research, Vol. 15, No. 1, pp. 53 – 63.

[158] Kolen, A. Rinnooy Kan, A. van Hoesel, C. and Wagelmans, A (1994) Sensitivity analysis of list scheduling heuristics, Discrete Applied Mathematics, Vol. 55, No. 2, pp. 145 – 162.

[159] Korn, F. and Muthukrishnan, S (2000) Influence sets based on reverse nearest neighbor queries, Proceedings of the ACM SIGMOD International Conference on Management of Data, Vol. 29. No. 2 pp. 201-212.

[160] Koskosidis, Y. and Powell, W. (1992) Clustering algorithms for consolidation of customer orders into vehicle shipments, Transportation Research, Vol. 26 , No. 5, pp. 365 – 379.

[161] Kuehn, A. and Hamburger, M. (1963) A heuristic program for locating warehouses, Management Science, Vol. 9, No. 4, pp. 643 – 666

[162] Kuhn, H. (1973) A note on Fermat's problem, Mathematical Programming, Vol. 4, pp. 98 – 107.

[163] Kytöjoki, J. Nuortio, T. Bräysy, O and Gendreau, M. (2007) An efficient variable neighborhood search heuristic for very large scale vehicle routing problems, Computer and operations research, Vol. 34, No. 19, pp. 2743- 2757

[164] Labbé, M. Peeters, D. and Thisse, J. (1995) Location on networks, Handbooks in operations research and management science, No. 8, Ball, M. Magnanti, T Monma, C and Nemhauser, G. (Editors), Network routing, Amsterdam, pp. 551–624.

[165] Laporte, G. Gendreau, M. Potvin, J-Y. and Semet, F. (2000) Classical and modern heuristics for the vehicle routing problem. International Transactions in operational Research, vol.7, pp.285- 300.

[166] Laporte, G. and Semet, F. (2002) Classical heuristics for the capacitated VRP. In Paulo Toth and Daniele Vigo, editors, The Vehicle Routing Problem, volume 9 of SIAM Monographs on Discrete Mathematics and Applications, chapter 5, pp 109–128. SIAM, Philadelphia.

[167] Leighton, F. (1979) A graph coloring algorithm for large scheduling problems, Journal of Research of the National Bureau of Standards, Vol. 84 pp. 489 – 503

[168] Lenstra, J. K. and Rinnooy, K.(1981) Complexity of vehicle routing and scheduling problem. Networks, Vol. 11, pp. 221-227.

[169] Levanova, T. and Loresh, M. (2004) Algorithms of ant system and simulated annealing for the p-median problem, Automation and Remote Control, Vol. 65, pp. 431 – 438

[170] Li, F. Golden, B. and Wasil, E. (2005) Very large-scale vehicle routing: new test problems algorithms, and results, Computers and operations research, Vol. 32, No. 5, pp. 1165- 1179

[171] Li, F. Golden, B. and Wasil, E. (2007) The open vehicle routing problem: Algorithms, large-scale test problems and computational results, Computers and operations research, Vol. 34, No. 10, pp. 2918- 2930

[172] Lin, S. (1965) Computer solutions if the travelling salesman problem, bell Syst. Tech. J. vol. 44, pp. 2245 – 2269

[173] Lin, S. and Kernighan, B (1973) An Effective Heuristic Algorithm for the Traveling-Salesman Problem, Operations Research, Vol. 21, No. 2, pp. 498 - 516.

[174] Loiola, E. Maia de Abreu, N. Boaventura-Netto, P. Hahn, P. and Querido, T. (2007) A survey for the quadratic assignment problem, European Journal of Operational Research, Vol. 176, pp. 657 – 690

[175] Lorena, L. and Senne, E. (2004) A column generation approach to capacitated p-median problems, Computers & Operations Research, Vol. 31, pp. 863 – 876

[176] Lysgaard, J. Letchford, A.N. and Eglese, R.W. (2004) A new branch-and-cut algorithm for the capacitated vehicle routing problem, Mathematical Programming, Vol. 100, No. 2, pp. 423 – 445

[177] Magalhães, J. and Pinho de Sousa, J. (2006) Dynamic VRP in pharmaceutical distribution — a case study, Journal Central European Journal of Operations Research, Vol.14, No. 2 , pp. 177 – 192.

[178] Maniezzo, V. Mingozzi, A. Baldacci, R. (1998) A Bionomic Approach to the Capacitated p-Median Problem, Journal of Heuristics, Vol. 4. No. 3, pp. 263 – 280.

[179] Maranzana, F. (1964) On the location of supply points to minimize transportation costs, Operations Research Quarterly, Vol. 12, pp.138 – 139.

[180] Mester, D. and Bräysy, O. (2007) Active-guided evolution strategies for large-scale capacitated vehicle routing problems, Computer and operations research, Vol. 34, No. 10, pp. 2964- 2975

[181] Min, H. (1989) The Multiple Vehicle Routing Problem with Simultaneous Delivery and Pickup Points, Transportation Research-A, Vol. 23A, No. 5, pp. 377 - 386.

[182] Minieka, E. (1977). The Centres and Medians of a Graph, Operations Research, Vol. 25, pp. 641 – 650.

[183] Mladenovic, N. Brimberg, J. Hansen, P. and Moreno-Perez, J (2007) The p-median problem: A survey of metaheuristic approaches, European Journal of Operational Research, Vol. 179, pp. 927 – 939

[184] Mole, R and Jameson, S. (1976) A Sequential Route-Building Algorithm Employing a Generalised Savings Criterion, Operational Research Quarterly, Vol. 27, No. 2, pp. 503-511.

[185] Montané, F. and Galvão, R. (2006) A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service, Computers & Operations Research, Vol. 33, No. 3, pp. 595 – 619

[186] Moreno-Pe´rez, J. Rodrı´guez, C. and Jimenez, N. (1991) Heuristic cluster algorithm for multiple facility location-allocation problem, Operations Research, Vol. 25, pp. 97 – 107.

[187] Moulitsas,I. and Karypis, G (2001) Multilevel Algorithms for Generating Coarse Grids for Multigrid Methods, sc, ACM/IEEE SC 2001 Conference (SC'01), pp. 15-24.

[188] Mulvey, J. and Beck, M. (1984) Solving capacitated clustering problems, European Journal of Operational Research, Vol. 18, No. 3, pp. 339 – 348.

[189] Myers, C. Robson, D. Wible, A. Hibbs, M. Chiriac, C. Theesfeld, C. Dolinski, K. and Troyanskaya, O. (2005) Discovery of biological networks from diverse functional genomic data, Genome Biology, Vol.6, No. 13

[190] Nagy, G. and Salhi, S. (2007) Location-routing: Issues models and methods, European Journal of Operational Research, Vol. 177, pp. 649 – 672

[191] Nambiar, J. Gelders, L. and Van Wassenhove, L. (1981) A large scale location allocation problem in natural rubber industry, European Journal of Operations research, Vol. 6, pp. 183 – 189

[192] Nash, J. (1950) Equilibrium points in n-person games, Proc Natl Acad Sci, Vol. 36 pp. 48–49

[193] Nash, J. (1951) Non-cooperative games, Ann Math, Vol. 54, pp. 286 – 295

[194] Negreiros, M. and Palhano, A. (2006) The capacitated centred clustering problem, Computers & Operations Research, Vol. 33, pp.1639–1663

[195] Nieddu, L. and Patrizi, G. (2000) Formal methods in pattern recognition: A review, European Journal of Operational Research, Vol, 120, pp. 459 - 495

[196] Niemi, R. and Weisberg, H, (1976), Controversies in American Voting Behaviour. W.H. Freeman & Sons, San Francisco, CA.

[197] Obuntan, I. (2005) A multilevel search algorithm for feature selection in biomedical data, Masters thesis, Department of computer science, University of Manitoba

[198] Oduntan, I. Toulouse, M. Baumgartner, R. Bowman, C. Somorjai, R. and Crainic, T. (2008) A multilevel tabu search algorithm for the feature selection problem in biomedical data, Computers and Mathematics with Applications, Vol. 55 No. 5, pp. 1019 – 1033

[199] Okabe, A. and Aoyagy, M. (1991) Existence of equilibrium configurations of competitive firms on an infinite two-dimensional space, J. Urban Economics, Vol. 29, pp. 349 – 370.

[200] Oliveria, S. and Seok, S. (2007) Multilevel approaches for large-scale proteomic networks, International Journal of Computer Mathematics, Vol. 84, No. 5, pp. 683–695

[201] Oliveira, S. and Seok, S. (2006) A multilevel approach for identifying functional modules in protein–protein interaction networks, Proceedings of IWBRA, Lecture Notes in Computer Science, 3992 (Berlin: Springer- Verlag), pp. 726–773.

[202] Oliveira, S. and Seok, S. (To appear) A matrix-based multilevel approach to identify functional protein models. International Journal of Bioinformatics Research and Applications.

[203] Or, I. (1976) Traveling salesman-type combinatorial problems and their relation to the logistics of blood banking, PhD Thesis, Department of Industrial Engineering and Management Sciences, Northwestern University.

[204] OR-Library, (date last update 2007) http://people.brunel.ac.uk/~mastjjb/jeb/orlib/pmedcapinfo.html. (date last accessed 2008)

[205] Ormerod, P (2006), Why most things fail, London: Faber and Faber Limited.

[206] Osman, I.H. (1993), Metastrategy simulated annealing and tabu search algorithms for vehicle routing problem, Annals of Oper. Res. vol. 41, pp. 421–451.

[207] Osman, I. and Christofides, N (1994) Capacitated Clustering Problems by Hybrid Simulated Annealing and Tabu Search, International Transactions in Operational Research, Vol.1, No. 3, pp. 317 – 336

[208] Osman, I. and Laporte, G (1996) Metaheuristics: A bibliography, Annals of Operations Research, Vol. 63, No. 5, pp. 511-623

[209] Pardalos, P and Resende, M (eds.) (2002). Handbook of Applied Optimization, New York: Oxford University Press.

[210] Park, H. Grings, A. Vinicius dos Santos, M. and Santos Soares, A. (2008) Parallel hybrid evolutionary computation: Automatic tuning of parameters for parallel gene expression programming, Applied Mathematics and Computation

[211] Paessens, H. (1988) The savings algorithm for the vehicle routing problem, European Journal of Operational Research, Vol. 34, No. 3, pp. 336 - 344.

[212] Pirkul, H. (1987) Efficient algorithms for the capacitated concentrator location problem, Computers and Operations Research, Vol. 14, No. 3, pp.197 – 208.

[213] Pisinger, D. and Ropke, S. (2007) A general heuristic for the vehicle routing problem, Computer and operations research, Vol. 34, No. 8, pp. 2403- 2435

[214] Pisinger, D. and Röpke, S. (2005) Technical Report: A general heuristic for vehicle routing problems, Computers & Operations Research, in press, doi:10.1016/j.cor.2005.09.012.http://www.diku.dk/~sropke/Papers/GeneralVRP_TechRep.pdf

[215] Plastria, F. (2001) Static competitive facility location: An overview of optimization approaches, European Journal of Operational Research, Vol. 129, pp. 461 – 470.

[216] Plastria, F. (2001) On the choice of aggregation points for continuous p-median problems: A case for the gravity center, TOP Vol.9, No. 2, pp. 217 – 242.

[217] Plastria, F and Vanhaverbeke, L (2007) Aggregation without Loss of Optimality in Competitive Location Models, Netw Spat Econ, Vol. 7, pp. 3 – 18

[218] Potvin, J. Lapalme, G. and Rousseau, J. (1989) generalized k-opt exchange procedure for the MTSP, Information Systems and Operations Research, Vol.27, No.4, pp. 474 – 481

[219] Potvin, J. and Rousseau, J. (1995) An Exchange Heuristic for Routeing Problems with Time Windows, The Journal of the Operational Research Society, Vol. 46, No. 12, pp. 1433 - 1446.

[220] Preux, P. and Talbi, E. (1999), Towards hybrid evolutionary algorithms, International Transactions in Operational Research, Vol. 6, pp. 557–570.

[221] Prins, C. (2004) A simple and effective evolutionary algorithm for the vehicle routing problem, Computers & Operations Research, vol.31, pp. 1985–2002.

[222] Reese, J. (2006) Solution methods for the p-median problem: An annotated bibliography, Networks, Vol. 48, No. 3, pp. 125 – 142

[223] Rego, C. (1998) A Subpath Ejection Method for the Vehicle Routing Problem, Management Science, Vol. 44, No. 10, pp. 1447 - 1459.

[224] Reinelt, G. (1991), TSPLIB: A traveling salesman problem library, ORSA Journal on Computing, Vol. 3, pp. 376 – 384 URL: http://www.iwr.uni-heidelberg.de/groups/comopt/software/TSPLIB95/

[225] Renaud, J. Boctor, F.F. and Laporte, G. (1996), An Improved Petal Heuristic for the Vehicle Routing Problem, Journal of Operational Research Society, Vol. 47, pp. 329 – 336.

[226] Renaud, J. Boctor, F.F. and Laporte, G. (1996) A fast composite heuristics for the symmetric travelling salesman problem. INFORMS Journal on computing, Vol. 8, no.2, pp.134 – 143

[227] Renaud, J. Laporte, G. and Boctor, F. (1996) A tabu search heuristic for the multi-depot vehicle routing problem, Computers & Operations Research, vol.23, No. 3, pp. 229–235.

[228] Renaud, J. Boctor, F. (2002) A sweep-based algorithm for the fleet size and mix vehicle routing problem, European Journal of Operational Research Vol. 140, pp. 618–628.

[229] Resende, M. and Werneck, R. (2004) A hybrid heuristic for the p-median problem. Journal of Heuristics, Vol. 10, No .1, pp. 59 – 88.

[230] ReVelle, C. Swain, R. (1970) Central facilities location, Geographical Analysis, Vol. 2, pp. 30 – 42.

[231] ReVelle, C. Marks, D. and Liebman, J. (1970) Analysis of private and public sector location problems, Management Science, Vol. 16 pp. 692 – 707.

[232] ReVelle, C. Toregas, C. and Falkson, L. (1976) Applications of the location set covering problem, Geographical Analysis, Vol. 8, pp. 67 – 76

[233] ReVelle, C. (1986) The maximum capture or sphere of influence location problem hotelling revisited on a network, Journal of Regional Science, Vol. 26, pp. 343 – 358.

[234] ReVelle, C. and Eiselt, H. (2005) Location analysis: A synthesis and survey, European Journal of Operational Research, Vol. 165, pp. 1 – 19

[235] ReVelle, C. Murray, A. and Serra, D. (2007) Location models for ceding market share and shrinking services, Omega, Vol. 35, pp. 533 – 540

[236] ReVelle, C. Eiselt, H. Daskin, M. (2008) A bibliography for some fundamental problem categories in discrete location science, European Journal of Operational Research, Vol. 184, pp. 817 – 848

[237] Ridge, E (2007) Design of Experiments for the Tuning of Optimisation Algorithms, PhD Thesis, University of York

[238] Roberts, P (2004), The new penguin history of the world, London: Penguin Group

[239] Rochat, Y. Taillard, E. (1995) Probabilistic diversification and intensification in local search for vehicle routing, Journal of Heuristics Vol.1, No. 1, pp. 147-167

[240] Rodney, D. Soper, A. and Walshaw, C. (2007), Multilevel Refinement Strategies for the Capacity Vehicle Routing Problem, Int. J. IT & IC, Vol.2, No.3

[241] Rodney, D. Soper, A. and Walshaw, C. (2007), The application of Multilevel Refinement to the Vehicle Routing Problem, In D. Fogel et al., editors, Proc. CISChed 2007, IEEE Symposium on Computational Intelligence in Scheduling, pp.212 – 219, IEEE, Piscataway, NJ, 2007

[242] Rodney, D. Soper, A. and Walshaw, C. (2006), Multilevel Refinement for the Vehicle Routing Problem, In Proc. ODYSSE 2006, Third International Workshop on Freight Transportation and Logistics

[243] Rogers, D. Plante, R. Wong, R. and Evans, J. (1991) Aggregation and disaggregation Techniques and Methodology in Optimization, Operations research, vol.39, No. 4, pp. 553–582.

[244] Rolland, E. Schilling, D. and Current, J. (1996) An efficient tabu search procedure for the p-median problem, European Journal of Operational Research, Vol. 96, pp. 329–342.

[245] Romen, Y. Rudolph, L and Stein, J.(1995) Adapting multilevel simulated annealing for mapping dynamic irregular problem, In Ranks, S. editor , Proc. Intl Parallel Processing Symp., pp. 65 - 72

[246] Ropke, S. (2005) Heuristic and exact algorithms for the vehicle routing problems, PhD Thesis, Department of Computer Science University of Copenhagen

[247] Rosenkrantz, D. Stearns, R. and Lewis, P. (1974) Approximate Algorithms for the Traveling Salesperson Problem, Proceedings of the 15th Annual IEEE Symposium of Switching and Automata Theory, pp. 33 – 42.

[248] Rosing, K. and ReVelle, C. (1997) Heuristic concentration: Two stage solution construction, European Journal of Operational Research, Vol. 97, pp. 75 – 86.

[249] Rosing, K. ReVelle, C. Rolland, E. Schilling, D. and Current, J. (1998) Heuristic concentration and tabu search: A head to head comparison, European Journal of Operational Research, Vol. 104, pp. 93 – 99.

[250] Rosing, K. and Hodgson, M. (2002) Heuristic concentration for the p-median: An example demonstrating how and why it works, Computer Operations Research, Vol. 29, pp. 1317 – 1330.

[251] Ross, G. and Soland, R. (1977) Modelling Facility Location Problems As Generalized Assignment Problems, Management Science, Vol. 24, No. 3, pp. 345 - 357.

[252] Ryan, D. Hjorring, C. and Glover, F. (1993) Extensions of the Petal Method for Vehicle Routeing, The Journal of the Operational Research Society, Vol. 44, No. 3, pp. 289 - 296.

[253] Salhi, S. and Rand, G. (1993) Incorporating vehicle routing into the vehicle fleet composition problem, European Journal of Operational Research, Vol.66 pp. 313–330.

[254] Salhi, S. and Atkinson, R. (1995) Subdrop: A modified drop heuristic for location problems, Location Science, Vol. 3, pp. 267 – 273.

[255] Salhi, S. (1997) A perturbation heuristic for a class of location problems, Journal of Operational Research Society, Vol. 48, pp. 1233– 1240.

[256] Salhi, S. (2002) Defining tabu list size and aspiration criterion within tabu search methods, Computers and Operations Research, Vol. 29, pp. 67 – 86.

[257] Sankaran, J. Gore, A. and Coldwell, B. (2005) The impact of road traffic congestion on supply chains: insights from Auckland New Zealand, International Journal of Logistics: Research and Applications, Vol. 8, No. 2, pp. 159 - 180

[258] Sankaran, J. and Wood, L. (2007) The relative impact of consignee behaviour and road traffic congestion on distribution costs Transportation Research Part B, doi:10.1016/j.trb.2007.04.005

[259] Sariklis, D. and Powell, S. (2000) A heuristic method for the open vehicle routing Problem, Journal of the Operational Research Society, Vol. 51, pp. 564-753

[260] Schaeffer, S. (2007) Survey Graph clustering, COMPUTER SCIENCE REVIEW, Vol. 1 pp. 2 7 – 6 4

[261] Schenke J. K-mart eyes more store closings. Atlanta Business Chronicle, 2002.

[262] Scheuerer, S. and Wendolsky, R. (2006) A scatter search heuristic for the capacitated clustering problem, European Journal of Operational Research, Vol.169, pp. 533–547

[263] Schilling, D. Jayaraman, V. Barkhi, R. (1993) A review of covering problems in facility location, Location Science, Vol. 1, pp. 22 – 55

[264] Schilling, D. Rosing, K. and ReVelle, C. (2000) Network distance characteristics that affect computational effort in the p-median problem, European Journal of Operational Research, Vol. 127, pp. 525 – 536.

[265] Schrage, L. (1981) Formulation and structure of more complex/realistic routing and scheduling problems, Networks, Vol. 11, No. 2, pp. 229-232

[266] Schrijver, A. (2005) On the history of combinatorial optimization (till 1960), In Aardal, K. Nemhauser, G. Weismantel, R. editor, Handbook of Discrete Optimization , Elsevier, Amsterdam, 2005, pp. 1 - 68.

[267] Serra, D. ReVelle, C. (1994) Market Capture by 2 competitors preemptive location problem, Journal of Regional Science, Vol. 34, pp. 549 – 561.

[268] Serra, D. and ReVelle, C. (1995) Competitive location in discrete space. In: Drezner, Z. (Ed.), Facility Location, Springer-Verlag, pp. 337–356

[269] Sharan, R. Maron-Katz, A. and Shamir, R. (2003) CLICK and EXPANDER: a system for clustering and visualizing gene expression data, Bioinformatics, Vol. 19, pp. 1787–1799

[270] Silver, E. (2004) An overview of heuristic solution methods, Journal of the Operational Research Society, Vol.55, pp. 936–956.

[271] Skiena, S, (1998) The Algorithm Design Manual (NewYork: Springer-Verlag)

[272] Smid, M. (2002) Closest-point problems in computational geometry, Sack, J. and Urrutia, J. (eds), Handbook of Computational Geometry, Elsevier, Amsterdam, pp. 877 – 935

[273] Solomon, M. (1987) Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints, Operations Research, Vol. 35, No. 2. pp. 254-265.

[274] Soper, A. Walshaw, C and Cross, M (2004) A Combined Evolutionary Search and Multilevel Optimisation Approach to Graph Partitioning, J. Global Optimization, Vol. 29, No. 2, pp. 225 - 241.

[275] Spirin, V. and Mirny, L. (2003) :Protein complexes and functional modules in molecular networks, Proc. Natl. Acad. Sci. USA, Vol. 100, pp. 12123 - 12128

[276] Stuetzle, W. (2003) Estimating the cluster tree of a density by analyzing the minimal spanning tree of a sample, J. Classif. Vol. 20, pp. 25 – 47

[277] Stumme, G. (2003) Off to newshores: conceptual knowledge discovery and processing, Int. J. Human-Computer Studies, Vol. 59, pp. 287 – 325

[278] Sullivan, D. (2003) Using probabilistic reasoning to automate software tuning, Ph.D. thesis, Harvard University

[279] Taillard, E. (1993). Parallel iterative search methods for vehicle routing problems, Networks, Vol. 23, pp. 661 – 673.

[280] Taillard, E. Gambardella, L. Gendreau, M. and Potvin, J. (2001) Adaptive memory programming: a unified view of metaheuristics, European Journal of Operational Research, Vol. 135, pp. 1 - 16.

[281] Talbi, E. (2002) A Taxonomy of Hybrid Metaheuristics, Journal of Heuristics, Vol. 8, No. 5, pp. 541 – 564.

[282] Tamir, A. (1996) An $O(pn^2)$ algorithm for the p-median and related problems on tree graphs, Operations Research Letters, Vol. 19 pp. 59 - 64

[283] Tarantilis, C.D. (2005) Solving the vehicle routing problem with adaptive memory programming methodology, Computers & Operations Research, Vol. 32 pp. 2309–2327

[284] Teitz, M. and Bart, P. (1968) Heuristic methods for estimating the generalized vertex median of a weighted graph, Operations Research, Vol. 16, pp. 955 – 961

[285] Teng, S. (1999). Coarsening, sampling and smoothing: elements of the multilevel method. In: Heath MT, Ranade A and Schreiber RS (eds). Algorithms for Parallel Processing. IMA Volumes in Mathematics and its Applications, Vol. 105, Springer Publishing Co. Inc., New York, pp 247–276.

[286] Thompson, P.M. and Psaraftis, H.N. (1993), Cyclic transfer Algorithms for Multivehicle Routing and Scheduling Problems, Oper. Res. Vol. 41, No.5, pp. 935-946.

[287] Tiernan, J.C. (1970) An efficient search algorithm to find elementary circuits of a graph, Communications of the ACM, Vol.13, No. 12, pp. 722-726.

[288] Tobin, R. Friesz, T. and Miller, T. (1989) Existence theory for spatially competitive network facility location models, Annals Operation Research, Vol. 18, pp. 267 – 276.

[289] Tol, R. (2007) The impact of a carbon tax on international tourism, Transportation Research, Part D, Vol. 12, pp. 129 – 142

[290] Tore Grünert, B.F. and Irnich, S. (2005) Local Search for Vehicle Routing and Scheduling Problems: Review and Conceptual Integration, Journal of Heuristics, Vol. 11, pp. 267-306.

[291] Toregas, C. Swain, R. ReVelle, C. and Bergman, L. (1971) The Location of emergency service facilities, Operations Research, Vol. 19, pp. 1363 – 1373.

[292] Toth, P. and Vigo, D. (1998) Exact solutions of the vehicle routing problem, In: Crainic, T. Laporte, G. (Eds.), Fleet management and logistics, Kluwer, Boston, MA. pp. 33-56

[293] Toth, P. and Vigo, D. (2003) The Granular Tabu Search and Its Application to the Vehicle-Routing Problem, Journal on Computing, Vol. 15, No. 4, pp. 333 - 346

[294] Tryfos, P. (1986) A integer programming approach to the apparel sizing problem, Journal of the Operational Research Society, Vol. 37, pp. 1001 – 1006.

[295] U.S. Environmental Protection Agency (Updated March 2008) http://www.epa.gov/climatechange/emissions/co2_human.html (accessed March 2008)

[296] Vehicle Routing Data Sets, (Updated October 2003), http://branchandcut.org/VRP/data/ (accessed June 2007)

[297] Vehicle Routing Problem Data, http://www.rhsmith.umd.edu/faculty/bgolden/vrp_data.htm (accessed June 2007)

[298] VRP instances, http://www.dpor.rwth-aachen.de/vrp-instances/ (accessed June 2007)

[299] VRP Web, (Updated March 2007), http://neo.lcc.uma.es/radi-aeb/WebVRP/index.html (accessed May 2007)

[300] Vries, S and Vohra, R. (2003) Combinatorial Auctions: A Survey, INFORMS Journal on Computing Vol.15, No. 3, pp. 284 – 309

[301] Walshaw, C. and Cross, M. (2000) Mesh Partitioning: A Multilevel balancing and refinement algorithm, SIAM J. SCI. COMPUT, Vol. 22, No. 1, pp. 63-80.

[302] Walshaw, C (2002) A Multilevel Approach to the Travelling Salesman Problem, Operations Research, Vol.50, no. 5, pp.862–877.

[303] Walshaw, C. (2003) A Multilevel algorithm for Fource-Directed Graph Drawing, Journal of Graph Algorithms and Applications, Vol. 7, No. 3, pp. 253 - 285.

[304] Walshaw, C. (2004) Multilevel Refinement for Combinatorial Optimisation Problems, Annals of Operations Research. Vol. 131, pp. 325–3

[305] Walshaw, C. (2007) Multilevel Refinement for Combinatorial Optimisation: Boosting Metaheuristic Performance. In C. Blum et al., editor, Hybrid Metaheuristics – An emergent approach for optimization. Springer. (Invited Chapter).

[306] Wang, Q. Batta, R. Badhury, J. and Rump, C. (2003) Budget constrained location problem with opening and closing of facilities, Computers and Operations Research, Vol. 30, No. 13, pp. 2047 – 69.

[307] Weber, A.(1909) Uber den Standort der Industrien, Tubingen (translated by C.J. Friedrich as Theory of the Location of Industries, University of Chicago Press, Chicago, IL).

[308] Wesolowsky, G. (1993) The Weber problem: History and procedures, Location Science, Vol. 1, pp. 5 – 23

[309] Weston, F. (1982) Optimal configuration of telephone answering sites in a service industry, European Journal of Operational Research, Vol. 10, No. 4, pp. 395-405

[310] Whitaker, R. (1983) A fast algorithm for the greedy interchange for large-scale clustering and median location problems, INFOR, Vol. 21, pp. 95 – 108

[311] Wren, A. and Holliday, A. (1972) Computer Scheduling of Vehicles from One or More Depots to a Number of Delivery Points, Operational Research Quarterly, Vol. 23, No. 3, pp. 333-344.

[312] Wu, Z. and Leahy, R. (1993) An optimal graph theoretic approach to data clustering: theory and its application to image segmentation, IEEE Trans. Pattern Analysis Mach. Intell. Vol. 15, No. 11, pp. 1101 – 1113.

[313] Yagiura, M. Ibaraki, T. and Glover, F. (2006) A path relinking approach with ejection chains for the generalized assignment problem, European Journal of Operational Research, Vol. 169 No. 2, pp. 548 – 569.

[314] Yang, C. and Lin, K. (2001) An index structure for efficient reverse nearest neighbour queries. In Proceedings of the IEEE International Conference on Data Engineering, Heidelberg Germany, pp. 485 - 492

[315] Yellow, P. (1970) A computational modification to the savings method of vehicle scheduling, Operation Research Quarterly, Vol. 21, No. 2, pp. 281–283.

[316] Zhou, Z. (2003) Three perspectives of data mining, Artificial Intelligence, Vol. 143, pp. 139–146

7 Appendices A – Irnich Results

7.1 Detailed results for the Irnich et al. instances

This section presents the detailed results for the experiments of applying the multilevel technique to selected instances from the Irnich et al. [147] test instances for the CVRP (see section 5.2.12.1).

Each table shows the results for problems of one instance size across all five series. Within each table, the results look at the effect of varying the vehicle capacity and consequently the average number of customers per route and the effect of varying the demand distribution of the customers. The costs are normalised with respect to the best cost found for each problem size.

Table 30 Varying vehicle capacity and demand distribution for instance size 250 of the Irnich et al. instances for a constant customer distribution (series 1 – 5)

Varying demand distribution by series	Normalised solution cost for Instance Size 250 Increasing average number of customers per route							
	25		50		75		100	
	SL	ML	SL	ML	SL	ML	SL	ML
1	1.373	1.366	1.140	1.126	1.091	1.070	1.035	1.038
2	1.365	1.346	1.081	1.080	1.027	1.029	1.011	**1**
3	1.381	1.374	1.140	1.125	1.103	1.095	1.040	1.038
4	1.365	1.384	1.128	1.130	1.056	1.055	1.030	1.030
5	1.417	1.381	1.159	1.147	1.106	1.103	1.075	1.068

Table 31 Varying vehicle capacity and demand distribution for instance size 300 of the Irnich et al. instances for a constant customer distribution (series 1 – 5)

Varying demand distribution by series	Normalised solution cost for Instance Size 300 Increasing average number of customers per route							
	25		50		75		100	
	SL	ML	SL	ML	SL	ML	SL	ML
1	1.477	1.461	1.149	1.143	1.079	1.068	1.029	1.027
2	1.400	1.395	1.115	1.114	1.065	1.063	1.005	**1**
3	1.453	1.444	1.150	1.159	1.073	1.051	1.059	1.062
4	1.411	1.400	1.147	1.132	1.052	1.050	1.040	1.036
5	1.500	1.473	1.125	1.118	1.071	1.068	1.055	1.027

Table 32 Varying vehicle capacity and demand distribution for instance size 400 of the Irnich et al. instances for a constant customer distribution (series 1 – 5)

Varying demand distribution by series	Normalised solution cost for Instance Size 400 Increasing average number of customers per route							
	25		50		75		100	
	SL	ML	SL	ML	SL	ML	SL	ML
1	1.520	1.517	1.154	1.151	1.076	1.070	1.011	1.009
2	1.460	1.460	1.133	1.124	1.045	1.046	1.025	1.014
3	1.544	1.490	1.149	1.139	1.095	1.071	1.022	1.023
4	1.431	1.426	1.143	1.140	1.040	1.039	1.001	**1**
5	1.571	1.536	1.168	1.161	1.071	1.071	1.045	1.052

Table 33 Varying vehicle capacity and demand distribution for instance size 500 of the Irnich et al. instances for a constant customer distribution (series 1 – 5)

Varying demand distribution by series	Normalised solution cost for Instance Size 500 Increasing average number of customers per route							
	25		50		75		100	
	SL	ML	SL	ML	SL	ML	SL	ML
1	1.617	1.596	1.232	1.206	1.099	1.074	1.012	1.004
2	1.596	1.573	1.214	1.194	1.075	1.055	1	**1**
3	1.593	1.572	1.224	1.199	1.103	1.084	1.077	1.074
4	1.541	1.546	1.133	1.131	1.059	1.06	1.010	1.010
5	1.667	1.631	1.245	1.232	1.120	1.113	1.029	1.024

Table 34 Varying vehicle capacity and demand distribution for instance size 600 of the Irnich et al. instances for a constant customer distribution (series 1 – 5)

Varying demand distribution by series	Normalised solution cost for Instance Size 600 Increasing average number of customers per route							
	25		50		75		100	
	SL	ML	SL	ML	SL	ML	SL	ML
1	1.670	1.662	1.189	1.183	1.075	1.081	1.050	1.041
2	1.662	1.663	1.217	1.20	1.077	1.068	1.010	**1**
3	1.658	1.628	1.238	1.211	1.113	1.109	1.068	1.057
4	1.568	1.555	1.213	1.209	1.084	1.074	1.011	1.011
5	1.667	1.644	1.246	1.244	1.097	1.096	1.057	1.051

Table 35 Varying vehicle capacity and demand distribution for instance size 700 of the Irnich et al. instances for a constant customer distribution (series 1 – 5)

Varying demand distribution by series	Normalised solution cost for Instance Size 700 Increasing average number of customers per route							
	25		50		75		100	
	SL	ML	SL	ML	SL	ML	SL	ML
1	1.696	1.680	1.238	1.223	1.112	1.086	1.024	1.014
2	1.655	1.647	1.219	1.196	1.095	1.083	1.005	**1**
3	1.682	1.673	1.241	1.227	1.121	1.118	1.029	1.028
4	1.648	1.637	1.217	1.216	1.091	1.094	1.049	1.037
5	1.698	1.672	1.226	1.224	1.100	1.079	1.047	1.031

8 Appendices B - Best known solution and methods CVRP

This section presents the best known solution for the CVRP instances used in the research.

8.1 Best known solution for the Christofides and Eilon instances

This section presents the Best Known Solution (BKS) for the Christofides and Eilon instances [48] for the CVRP (see section 5.2.9).

Table 36 Best Known Solution for the Christofides and Eilon instances

Instance	N	BKS	Source	Notes
				All the BKS values rounded up to the nearest integer as specified by [224]. * Denotes values that are proven optimal
E-n22-k4	21	375*	BCA	Reported runtime of 0.1 s on a 400 MHz Sun-Ultrasparc II
E-n23-k3	22	569*	BCA	Reported runtime of 0.1 s on a 400 MHz Sun-Ultrasparc II
E-n33-k4	32	835*	BCA	Reported runtime of 0.7 s on a 400 MHz Sun-Ultrasparc II
E-n76-k14	75	1021	ABCA	Reported runtime of 986 s on a 700 MHz Intel Celeron processor and 256 MB of RAM running under Microsoft Windows 98
E-n76-k8	75	735*	BCA	Reported runtime of 126 s on a 400 MHz Sun-Ultrasparc II
E-n76-k7	75	682*	BCA	Reported runtime of 95 s on a 400 MHz Sun-Ultrasparc II
E-n101-k14	100	1071	ABCA	Reported runtime of 1040 s on a 700 MHz Intel Celeron processor and 256 MB of RAM running under Microsoft Windows 98

BCA: Branch and cut algorithm [23]. The work in [23] is based on the heuristics of [11]
ABCA: Branch and cut algorithm [176]

8.2 Best known solution for the Christofides et al. instances

This section presents the Best Known Solution for the Christofides et al. instances [49] for the CVRP (see section 5.2.8)

Table 37 Best Known Solution for the Christofides et al. instances

Instance	N	BKS	Source	Notes
				* Denotes values that are proven optimal
1	50	524.61*	PTS	++Run times for individual cost not provide explanation provided in [279], were the goal of the algorithm was to obtain the best results possible for each instance
2	75	835.26	PTS	
3	100	826.14	PTS	
4	150	1028.42	PTS	
5	199	1291.45	PLS	
6	50	555.43	PTS	++
7	75	909.68	PTS	
8	100	865.94	PTS	
9	150	1162.55	PTS	
10	199	1395.85	PLS	
11	120	1042.11	PTS	++
12	100	819.56	PTS	
13	120	1541.14	PTS	
14	100	866.37	PTS	

PTS: Parallel iterative search [279]
PLS: Probabilistic local search [239]

8.3 Best known solution for the Golden instances

This section presents the Best Known Solution for the Golden instances [122] for the CVRP (see section 5.2.10)

Table 38 Best Known Solution for the Golden instances

Instance	N	BKS	Source	Notes
				Runtime in minutes
1	240	5627.54	*AGES*	3.12
2	320	8447.92	*AGES*	16.93
3	400	11036.22	*AGES*	14.33
4	480	13624.52	*AGES*	357.03
5	200	6460.98	*AGES*	1.27
6	280	8412.80	*GA*	9.97
7	360	10181.75	*ALNS*	Runtime not published
8	440	11663.55	*AGES*	12.25
9	255	583.39	*AGES*	6
10	323	741.56	*AGES*	1.25
11	399	918.45	*AGES*	7.33
12	483	1107.19	*AGES*	10.78
13	252	859.11	*AGES*	6.67
14	320	1081.31	*AGES*	0.8
15	396	1345.23	*AGES*	0.45
16	480	1622.69	*AGES*	13.33
17	240	707.79	*AGES*	0.5
18	300	998.73	*AGES*	2.5
19	360	1366.86	*AGES*	0.39
20	420	1820.09	*AGES*	3.83

AGES: Active-guided evolution strategies [180]. Implemented in Visual basic 6.0 and executed on a Pentium 4 Net Vista PC 2.8 GHz. 512 MB RAM.
ALNS: Adaptive large neighborhood search [213]. Value taken from [214]
GA: Evolutionary algorithm [221] implemented in Delphi 5 and executed on a 1 GHZ Pentium -3 PC running windows 98.

8.4 Best known solution for the Li et al. instances

This section presents the Best Known Solution for the Li et al. instances [170] for the CVRP (see section 5.2.12)

Table 39 Best Known Solution for the Li et al. instances

Instance [1]	N	BKS	Source	Notes
				Runtime in minutes
21	560	16212.74	*AGES*	2.98
22	600	14597.18	*AGES*	0.58
23	640	18801.12	*AGES*	666.67
24	720	21389.33	*AGES*	1.67
25	760	16902.16	*ALNS*	Runtime not available
26	800	23971.74	*AGES*	2.17
27	840	17488.74	*AGES*	4.88
28	880	26565.92	*AGES*	333.33
29	960	29154.34	*EST*	Runtime not applicable
30	1040	31742.51	*AGES*	15
31	1120	34330.84	*AGES*	83.33
32	1200	36919.24	*EST*	Runtime not applicable

AGES: Active-guided evolution strategies [180]. Implemented in Visual basic 6.0 and executed on a Pentium 4 Net Vista PC 2.8 GHz. 512 MB RAM.
ALNS: Adaptive large neighborhood search [213]. Value taken from [214]
EST: Estimated solution from [170]

[1] Instances are giving the original identifying number used in [170]

9 Appendices C - Best known solution and methods CPMP

This section presents the best-known solution for the CPMP instances used in the research.

9.1 Best known solution for the Osman and Christofides instances

Optimal solution values (*opt*) for the Osman and Christofides instances [207] for the CPMP (see section 5.3.3.3).

Instances 1 to 10, N = 50, P = 5. Instances 11 to 20, N = 100, P = 10.

Table 40 Optimal solution values for the Osman and Christofides instances.

Instance	*opt*	Runtimes (s)	Notes
1	713	23	All solution values are optimal values produced by the branch and bound procedure of Maniezzo et al. The algorithms were coded in Fortran 77 and run on a IBM PC equipped with a Pentium 166 MHz CPU.
2	740	2	
3	751	6	
4	651	4	
5	664	5	
6	778	9	
7	787	49	
8	820	3186	
9	715	17	
10	829	255	
11	1006	722	
12	966	3485	
13	1026	274	
14	982	11042	
15	1091	8414	
16	954	1908	
17	1034	12315	
18	1043	3239	
19	1031	11894	
20	1005	54345	

9.2 Best known solution for the San Jose dos Campos instances

Best known solution for the San Jose dos Campos instances [175] for the CPMP (see section 5.3.4).

Table 41 Best Known solutions for the San Jose dos Campos instances.

Instance	*N*	*P*	*bks*	Time (min)	Notes
SJC1	100	10	17288.99	3.57	The best known values were produced by A hybrid scatter search (HSS) heuristic due to Diaz and Fernandez [73] implemented on a Sun Blade 1000/750 coded in C
SJC2	200	15	33270.94	28.45	
SJC3a	300	25	45338.01	177.87	
SJC3b	300	30	40635.90	443.33	
SJC4a	402	30	61928.91	1265.44	
SJC4b	402	40	52541.72	1576.21	