



Phoenix

Non-cooperative bargaining agents deploying
computationally bounded optimization
strategies

Thesis

Master of Media and Knowledge Engineering

Ferdinand de Bakker

Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology, The Netherlands
August 2009



Graduation committee

Prof. Dr. Drs. L.J.M. Rothkrantz

Dr. Ir. C.A.P.G. van der Mast

Ir. H.J.A.M. Geers

Dr. Ir. P. Wiggers


De Bakker, Ferdinand (ferdinand@debakkerconsulting.eu)
Master Thesis, August 2009

Phoenix – non-cooperative bargaining agents deploying computationally bounded optimization strategies

Delft University of Technology, The Netherlands
Faculty of Electrical Engineering, Mathematics and Computer Science
Master programme: Media and Knowledge Engineering

Keywords: distributed artificial intelligence, multi-agents systems, non-cooperative bargaining, optimizing vehicle routing

© Copyright 2006-2009, Ferdinand F de Bakker

 **OpenOffice.org 3** This document was produced with the *OpenOffice.org Productivity Suite*



Contents

Table of Contents

CONTENTS.....	3
0 PREFACE.....	5
0.1 – Motivation.....	5
0.2 – The Phoenix Project.....	5
0.3 – Acknowledgments.....	6
1 ABSTRACT.....	7
2 STRUCTURE OF THESIS.....	9
3 INTRODUCTION.....	13
3.1 – ICT For Networked Businesses.....	13
3.2 – Problem Definition.....	13
3.3 – Relevance.....	14
3.4 – Potential Impact.....	15
3.5 – Research Challenges.....	17
3.6 – Innovation Over State-of-the-art.....	20
3.7 – Methodology.....	20
4 LITERATURE.....	21
4.1 – Foundation.....	21
4.2 – Optimization.....	22
4.3 – Bargaining.....	23
4.4 – Agent Based Computing.....	24
5 ARCHITECTURE.....	27
5.1 – Bargaining Model.....	27
5.2 – Functional Model.....	27
5.3 – Service Platform Architecture.....	28
5.4 – Platform Implementation.....	30
6 IMPLEMENTATION.....	33
6.1 – Project Structure.....	33
6.2 – Risk And Risk Mitigation.....	35
6.3 – Work Packages.....	36
6.4 – Consortium.....	38
6.5 – Work Planning.....	41
6.6 – Detailed Work Description.....	42
6.7 – Project Organization.....	44
7 PROOF OF CONCEPT.....	51
7.1 – Approach.....	51
7.2 – Prototype.....	51
7.3 – Business Case.....	52



7.4 – Rationale.....	53
8 PROTOTYPE ARCHITECTURE.....	55
8.1 – Model.....	55
8.2 – Vehicle Routing Heuristics.....	55
8.3 – Multi-Depot CapVRP.....	58
8.4 – CapVRP Problem Definition.....	58
8.5 – Genetic Representation.....	59
9 CAPVRP IMPLEMENTATION.....	65
9.1 – Requirements.....	65
9.2 – Structural Model.....	65
9.3 – Behavioral Model.....	70
9.4 – Software Architecture.....	72
9.5 – Deployment.....	74
10 EXPERIMENTATION.....	79
10.1 – Introduction.....	79
10.2 – Quality And Robustness.....	79
10.3 – Profitability.....	84
11 CONCLUSIONS.....	87
11.1 – Literature Study.....	87
11.2 – Phoenix Project Proposal.....	87
11.3 – Prototype.....	88
11.4 – Proof Of Concept.....	88
11.5 – Recommendations.....	89
12 REFERENCES.....	91
13 APPENDIX A.....	93
13.1 – CapVRP DTD Schema	93
13.2 – Default Instance File.....	95
13.3 – Augerat A80k10 Instance.....	104
13.4 – Augerat A32k5 Instance.....	105
14 APPENDIX-B – COMPANION CD-ROM.....	107
14.1 – Complete Software API Documentation Including Source Code.....	107
14.2 – Logger Files And Analysis Spreadsheets.....	107
14.3 – Used Instance Files (XML).....	107
15 APPENDIX-C – THE CAPVRP PROBLEM REVISITED – A PAPER.....	109



0 Preface

This master thesis describes the research and development I have done in order to graduate at the master course Media and Knowledge Engineering at the Faculty of Electrical Engineering, Mathematics and Computer Science at Delft University of Technology.

0.1 Motivation

This thesis is an important step in the career switch I initiated in the millennium year 2000. At the time, after a long-standing career in industry and – early – retirement looming on the horizon, I decided to take that prospect as an opportunity and to prepare for a productive and fulfilling “third career”. I started the part-time study Computer Science in order to actualize and deepen my understanding of the fundamentals of computer science, the area in which I have worked for more than a quarter century in business oriented consulting and managerial roles. During my study, my interest was caught by the modeling and computational challenges of artificial intelligence technologies.

Artificial Intelligence technologies have a special significance in relation to agent technologies. While agents within a particular environment form an embodiment, their conduct, however, is determined by the rules and algorithms capable of grasping the – often unpredictable – complexities of the problem domain and producing timely and practical behaviors. Consequently the technology comprises models from cognitive sciences, game theory, computational complexity, optimization and approximation, heuristics, probability theory and software engineering. Here is where the real challenge lies.

Distributed Artificial Intelligence or Computational Intelligence – the combination of Artificial Intelligence and Agent technologies – I consider as the next generation delivery vehicle for advanced functionality and autonomous problem solving.

0.2 The Phoenix Project

In search for realistic and value-added applications of these technologies, I encountered a seminal article by Tuomas Sandholm from Carnegie Mellon University, Pittsburgh PA USA [1] in which a comprehensive case was presented, potentially bringing together optimizing heuristics for intractable problems and bargaining based on game theoretic models and for which I could envision an agent-based implementation. The article titled “Bargaining with limited computation: deliberation equilibrium” – though quite flawed in its elaboration – presented the case of logistic service providers negotiating on a case per case basis to pool their orders and fleets, including the distribution of profits resulting from the efficiency gained through cooperation.

I was impressed by the business potential of the concept if it could be researched in depth and developed into a service package based on a comprehensive set of components enabling rapid solution development and implementation.

In July 2005, I took the initiative to form an international consortium and prepared a project proposal for IST Call 5: ICT for Networked Businesses, within the EU Sixth Framework Programme.



0.3 Acknowledgments

At this place I would like to mention the following persons, who made valuable contributions or have been supportive in reaching the result before you.

- My son Simon de Bakker MA, for helping me out on the more esoteric intricacies of C, C++, Linux and Unix style software development and many other technical issues, which not only provided me with invaluable insights but also saved me LOTS OF TIME.
- My consortium partners, who were extremely supportive in getting the project proposal at the European Commission in Brussels on time. In that respect I would especially like to thank Dr. Geir Hasle, Chief Scientist at Sintef Information and Communication Technology, Department of Applied Mathematics, Oslo, for his valuable – and candid – feedback when nearing the deadline. I would also like to thank Ir. Bert van Werkhoven, department head at SenterNovem, for his unrelenting guidance in the labyrinth of European Procedures and Guidelines around Framework Program research projects.
- I am especially grateful of my mentor Prof. Dr. Drs. Leon Rothkrantz, for his pleasant and open manner in helping me through the various hurdles and setbacks inherent to a part time study, keeping me focused and ultimately getting me to the results at hand.
- Last but not least I would like thank my partner and life companion Elles Loenen, who had to keep up with broken holidays and numerous other inconveniences on which I'd rather not elaborate here, and – almost – always kept her supportive attitude and probably more importantly so, her good humor.



1 Abstract

Title project	Phoenix – Non-cooperative bargaining agents deploying computationally bounded optimization strategies.
Name	Ferdinand F. de Bakker
Student number	1063634
Graduation committee	Prof. Dr. Drs. L.J.M. Rothkrantz, Dr. Ir. C.A.P.G. van der Mast, Ir. H.J.A.M. Geers, Dr. Ir. P. Wiggers.
Faculty	Faculty of Electrical Engineering, Mathematics and Computer Science.
Study	Master of Media and Knowledge Engineering.
Keywords	Distributed artificial intelligence, multi-agents systems, non-cooperative bargaining, optimizing vehicle routing, genetic algorithm.

Abstract. The project addresses cost savings and productivity improvement by pooling of resources between independent companies through multi-agent, bargaining-based decision methods. Cost savings are realized by optimizing plans across companies, while the pooling decision and the benefit distribution is done by bargaining. Applications with these characteristics are wide-spread including logistics (pooling of fleets) and manufacturing (joint manufacturing plans).

This thesis comprises the project proposal as a research project submitted to the Sixth Framework Programme, Information Society Technologies, and the realization of the proof-of-concept. This proof is provided by the design and implementation a prototype optimizer for the Capacitated Vehicle Routing Problem (CapVRP) in order to simulated pooling versus non-pooling strategies based on a – simplified – business case. The optimizer uses the genetic algorithm meta-heuristic. Extensive experimentation using published CapVRP instances and benchmarks leads to the delivery of the proof-of-concept.



This page is intentionally left blank.



2 Structure Of Thesis

This thesis is based on the project proposal for IST Call 5: ICT for Networked Businesses, within the EU Sixth Framework Programme¹, and the design and development of a small part of the heuristics package. As a result this thesis consists of two main parts:

- Part 1: The Phoenix project proposal, providing the full scope of the project including research objectives, societal and commercial rationale, project structure and delivery strategy.
- Part 2: Based on the full project scope we are zooming in on a proof of concept. As the scope of the project is quite large, we first draw the overall architecture in broad lines and subsequently focus on the design and realization of the first essential building block: optimizing a computationally intractable problem – the Capacitated Vehicle Routing Problem (CapVRP).

¹ Calls for proposals for actions under the specific programme for Research, Technology development and Demonstration: “Integrating and strengthening the European research Area” in the Community Sixth Framework Programme. Thematic priority area: Information Society Technologies, 2005-2006 Work programme.
Call identifier: FP6-2005-IST-5 – Strategic Objective 2.5.8: ICT for Networked Businesses



This page is intentionally left blank.



PART 1

PHOENIX PROJECT PROPOSAL



This page is intentionally left blank.



3 Introduction

3.1 ICT For Networked Businesses

Strategic Objectives for “ICT for Networked Businesses” as stated by the Sixth Framework programme are:

- To develop software solutions adaptable to the needs of local/regional Small and Medium Companies (SME's), supporting organizational networking and process integration as well as improving adaptability and responsiveness to rapidly changing market demands and customer requirements.
- To develop distributed and collaborative ambient intelligence-based network oriented systems for efficient, effective and secure product and service creation and delivery. The aim is to explore how ambient intelligence technologies and the vision of duality of existence, in the real world and in cyberspace, can result in innovative products, services and business environments.

Focus: Extended Products And Services

Research in this area will investigate what recent progress in ambient intelligence technologies (e.g., agent based systems, knowledge management, smart wireless tagging, and ubiquitous computing) can mean for new products, services and the business environment. The work can cover decentralized architectures of intelligent communicating objects or processes allowing new approaches to collaboration, planning, scheduling, material management, auctioning, tendering, invoicing, workflow management, knowledge management or other business processes. Underlying issues such as interoperability, flexible, secure and robust infrastructures, information and knowledge sharing, modeling and simulation, and organizational change should be given due consideration.

3.2 Problem Definition

In the face of globalization and emerging economies such as China and India, European companies - including SME's - face the challenge to stay competitive in the global market. They have to implement strategies to improve productivity and reduce cost, rearrange the value chain and exploit opportunities offered by global high performance communication infrastructures.

The Phoenix project addresses cost savings by pooling of resources between independent (SME) companies through multi-agent, bargaining-based decision methods. Cost savings are realized by complex plan-optimization across organizational boundaries, while the pooling decision and the benefit distribution are done by bargaining. As this approach is applied each planning cycle, this will lead to substantial cost reduction and productivity increase through improved resource utilization. We estimate the impact of 20% and more of total cost.

Each agent has an intractable¹ optimization problem (production scheduling, vehicle routing, network design etc.). There is a potential for savings from pooling the individual problems, leading to an intractable joint problem. The decision method is subject to a

¹ A problem is considered tractable if it can be solved within reasonable computing time. In complexity theory computing time that does not grow faster than a polynomial with problem size (n) is considered reasonable. Problems with polynomial time complexity – $O(n^k)$ – belong to the complexity class P . The class of combinatorial problems to which most scheduling and routing problems belong, however, have exponential or worse time complexity and are computationally intractable – complexity classes NP and NP -complete.



response time constraint, so that each agent must rationalize its deliberation and bargaining process in order to optimize its expected gain.

Applications with these characteristics are wide-spread, including transportation (pooling of fleets), manufacturing (joint production plans with pooling of resources), supply-chain management, and telecommunications.

To reach our overall goals, we shall develop a powerful set of concepts and architectures for building agent-based bargaining environments, excellently fitting the ambient intelligence vision. To this end, further development is needed in the following technology areas:

- Meta-heuristics. These are modern problem solving strategies based on local search, stochastic search, and evolutionary algorithms that have proven effective in producing high quality solutions to intractable optimization problems. We shall further develop fast and powerful meta-heuristics with “anytime” characteristics for the targeted planning problems. They will at any time return the best solution reached, along with an indication of the solution quality and a prediction of solution quality to be obtained if given additional computation time.
- Knowledge based concepts. Appropriate deliberation and bargaining models based on the negotiation context including ultimatum games, bargaining with limited information and Nash/Bayesian equilibria.
- Multi-agent technology in which the autonomous behavior of each agent is determined by the above models and communication is enabled through domain specific ontologies and an appropriate Agent Communication Language concept (CCL). Agent platform will be FIPA compliant and based on the JADE suite or any suitable agent platform.

3.3 Relevance

Phoenix complies with this objective by developing systems that add value to collaborative ambient-intelligence based organizational networking. It provides the basis and necessary toolset for innovative business environments and services.

Innovative Business Environment

Business, applying the systems and toolsets provided by Phoenix, implement new business models to improve productivity and reduce cost by pooling resources with competitors in the value chain. The models applied within Phoenix ensures, that each planning cycle the decision making about cooperation or competing is based on objective data on the cost/gains related to the pooling/non-pooling options and a negotiation result on the profit distribution in case pooling is the profitable one. This cooperation between the business parties is not symmetric (i.e. you always cooperate or you don't), you only cooperate when it is profitable for both parties resulting in maximum benefit not only on the company level but also on the industry level.

As the companies remain independent – often competing – parties, this business environment is particularly suitable for a virtual market place, where companies can publicize their wish for pooling and start negotiating with one or more interested parties and come to agreement with one – or even more – parties based on the highest mutual benefit. In that situation the agent platform must impose the appropriate bargaining environment rules to ensure a “level playing ground” for participating businesses.



Innovative Services

The Phoenix technology provides innovative business opportunities for the European IT service industry.

- Consultancy organizations can offer services in applying advanced planning technologies to individual business and in the re-engineering of the relevant business processes within and across the value-chain.
- Service providers can devise new service offerings in providing the regulatory, technical and computational services infrastructures – i.e. agent platforms – for the innovative business environments just discussed.

Focus

The Phoenix project addresses decentralized architectures and intelligent communicating objects: the agent paradigm. The project provides the means for new approaches in collaboration between independent businesses in order to optimize planning and scheduling in logistics and manufacturing and thus increase value creation. Complex optimization in combination with the (non) cooperative¹ bargaining delivered by an autonomous multi-agent delivery vehicle is characteristic for a wide range of applications in technical infrastructures, business, and society.

The Phoenix solution specifically addresses business problems with the following characteristics:

- the company operates in a value chain or network, where each company has a complex planning and optimization problem e.g. in manufacturing or logistics;
- the companies have “general-purpose” production resources – “job-shop” type of process technology or fleet of vehicles – so there is potential for pooling resources;
- the companies may have different cost structures, so there is also room for optimization of the order lists, giving additional room for cost saving;
- there is a strong response time constraint;
- there is a potential for increased value creation. Utility can be expressed in cost saving, service degree improvement (delivery time), more business, and so on;
- partners are independent entities, so for each planning instance there is a decision problem whether pooling is profitable (i.e. has a higher value than in case each company would solve its problem individually) and if so, how the surplus value is shared between the partners.

3.4 Potential Impact

Strategic Impact

The Phoenix project contributes in reinforcing competitiveness of European industry. Globalization and emerging economies such as China and India, challenge European companies - including SME's - to stay competitive. Companies have to implement strategies to improve productivity and reduce cost, rearrange the value chain and exploit opportunities offered by global high performance communication infrastructures.

Around the millennium, many companies have implemented new enterprise-wide ERP solutions, solving their problems in running their day-to-day business. Companies now

¹ Cooperative bargaining exists in a 2 parties situation, when parties agree on a particular common strategy in pursuing each one's individual goal. In bargaining settings with more than 2 parties, cooperative bargaining refers to the possibility of coalitions between bargainers.



must differentiate by seeking innovative opportunities. These opportunities most often lie in:

- optimizing mission critical processes in manufacturing and logistics;
- designing and implementing smarter ways of competing/cooperating with parties in the value chain;
- finding better ways of delivering value to their customers.

The information technology needed to meet those needs must typically provide more intelligent and autonomous solutions, while utilizing the capabilities of high bandwidth networks.

Especially small and medium-sized enterprises (SME's) often lack the knowledge and expertise to identify and evaluate business opportunities specifically enabled by these technologies. Implementing strategic solutions often require knowledge about advanced business and decision models and complex software engineering.

SME's would substantially benefit in terms of competitive advantage, cost leadership and/or customer value if they could accelerate the adoption of smart technologies relevant to their business. The technologies developed in the Phoenix project in combination with an adequate service package would enable them to achieve that goal.

The extent of business value is of course depending on the concrete business situation. However in situations where no previous effort has been done to optimize business processes, which is almost always the case with inter-company networks where the companies are competing independent entities, estimated savings are 20% and above of total cost. This level would constitute a substantial profit contribution on the company level, and a sizable productivity increase on the industry-vertical level.

Innovation-related Activities

The innovation related activities concern:

- the development of the appropriate mathematical models for optimization and application of knowledge engineering and game theory on bargaining situations. The innovation is strongly linked to business cases in industry, ensuring practical, usable results with a huge market potential;
- the translation of these concepts into a software architecture that ensures applicability to a wide variety of applications;
- implementation in reusable software classes and components ensuring extendibility, flexibility and fast (re)configuration of solutions;
- the development of new business concepts, products, and services that utilize the technological advances from Phoenix.

Exploitation Plans

The developed technologies will be embedded in an overall service concept, which would typically have the following elements:

- Management consulting to identify opportunities resulting from business objectives and technological capabilities and to demonstrate and substantiate possible directions and assessing potential business impact of re-engineered business processes and re-arranged value-chain.
- Business value appraisal. Business case evaluation based on selected opportunity against strategic fit, competitive advantage and Risk Adjusted Return on Investment.



- Proof of concept. Rapid application development of critical technologies based on component libraries and solution templates, in order to prove viability of chosen solution.
- Solution delivery. Manage the solution delivery, system integration and change management processes.

3.5 Research Challenges

Meta-heuristics

The cost-saving objective is realized through optimization of complex planning situations. Resource pooling depends on decisions made by the agents. Each agent will compare its locally optimized plan with solutions of the pooled problem and decide whether pooling is profitable after bargaining. Time for deliberation and bargaining will be limited. In technical systems response requirements might be sub-second, in organizations it might be minutes.

The real-world optimization problems we face are particularly difficult due to their discrete nature, their structure, their size, and the strong response requirements. For example, to consider pooling of vehicles for serving transportation orders, each agent will have to solve an instance of the so-called Vehicle Routing Problem (VRP). The VRP is computationally intractable. Even for idealized VRPs, state-of-the-art exact methods cannot consistently solve problem instances with more than some 50 orders within a few minutes, a small number for real-life applications. We find similar, intractable optimization problems in manufacturing and network design.

We shall further develop and utilize any-time approximation optimizers based on meta-heuristics that are capable of handling advanced planning problems in logistics and manufacturing in a bargaining environment. The following requirements are central:

- Fast convergence with high global optimum probability;
- Any-time return of best solution found with estimated quality (approximation ratio);
- Solution quality prediction as a probability distribution per incrementally allocated time unit;
- Self-adaptive parameter control to obtain better predictable asymptotic behavior over disparate classes of problems or problem instances.

Given the above requirements, we shall investigate modern meta-heuristics and design a robust optimization method that performs well over all relevant problem instances. Prime candidates are Iterated Local Search, Variable Neighborhood Search, Guided Local Search, and Evolutionary Algorithms.

Research objectives

We shall focus on two classes of problems at the core of the selected Business Cases: Rich models of the VRP within transportation logistics, and within manufacturing a Multi-Facility Multi-Resource Scheduling Problem. Sub goals are:

- mathematical and conceptual models;
- problem analysis;
- design of self-adaptive optimization algorithm;
- empirical investigations on test suite;
- solution quality measures, including predictions.



Technology objectives

- Generalized architectures and object model.
- Application Program Interfaces (API).
- Component models.
- Demo implementation based on selected business cases.
- Performance assessment.

Bargaining

The bargaining element is essential in order to reach the best business solution and agreement on the distribution of the benefits.

Bargaining will be performed for each party by a rational software agent. The bargaining model consists of two different – though interrelated – parts: deliberation and bargaining.

Deliberation

Each agent has to decide how much resources it wants to spend on which problem, which offer to make and which to accept. These decisions are made, based on the results of its computations up to this point and its expected gain from different future computations. In this proposal we will develop a *normative* deliberation control model. This model incorporates the various options agents have in controlling their computation and bargaining process.

Bargaining

Bargaining is done in situations where

- agents have the possibility to conclude a mutually profitable agreement;
- there is a conflict of interest;
- agents have autonomy with respect to offering and acceptance.

In this project bargaining is done to decide whether to combine resources and solve the joint problem or to solve each problem individually. If the value of the joint solution is higher than the value if the sum of each individual solution, then there is a basis for agreement on a joint approach. In this case bargaining is about the division of the surplus.

Research objectives

We shall focus on two classes of problems in complex multi-agent decision making, which is at the core of our Business cases:

1. Agents Design: the agents decision making process itself;
2. Mechanism Design¹: rules of the bargaining environment to ensure the collective good is reached when agents maximize their own utility, colloquially referred to as “level playing field”.

Sub goals are:

- mathematical and conceptual models and algorithm for Agent Design;
- deliberation strategy profiles and game contexts;
- bargaining strategy profiles and game contexts;

¹ Comparable with the role of government. See [5] Russell and Norvig: “Artificial Intelligence – a Modern Approach” 2nd edition par 17.7 page 640. The interested reader might also want to have a look at the influential article “[The Tragedy of the Commons](#)” by Garret Hardin (published in Science, 1968).



- mechanism designs for relevant game contexts (ultimatum game, known or unknown proposer, known or unknown deadline, pure, mixed and dominant strategies, etc.);
- Nash and Bayesian equilibria and belief system models;
- empirical investigations on test suite.

Technology objectives

- Generalized architectures and object model.
- Application Program Interfaces (API).
- Component models.
- Demo implementation based on the business cases.
- Performance assessment.

Multi-agent Platform

The multi-agent platform is the processing environment that will deliver the solution to the end-users. The optimizing and bargaining concepts will be implemented in an agent model. The agents are autonomous software entities that:

- provide computational services to create optimized plans according to selected approximation models (where available response time determines the selection);
- provide deliberation strategies based on various bargaining contexts;
- may bargain on behalf of their principal.

In short, an agent is a computational process that implements the autonomous deliberation, bargaining, and communicating functionality of an application. The agent operates within a platform that provides the necessary services during the agent's life cycle and enforces the environmental rules imposed on the bargaining process.

An Agent Platform provides the physical infrastructure in which agents are deployed. We will use the JADE¹ as agent platform because its FIPA² compliance. According to the FIPA Agent Standard an Agent Platform provides Directory Facilitator, Agent Management and Messaging services.

Research objectives

We shall focus on the design of a communication language for agents to interact effectively within the core domains of our Business Cases. Sub goals are:

- analysis and design of a suitable ontology (vocabulary);
- analysis and design of interaction patterns;
- analysis and design of a communication language (syntax and semantics);
- analysis and design of sentence construction and parsing rules.

Technology objectives

- Design rules for Agent based application architecture.
- Design rules and engineering guidelines for JADE based Agent Platform.
- Demo implementation of our Business Cases.

1 Acronym of "Java Agent Development framework". See <http://jade.tilab.com/>

2 Acronym of "Foundation for Intelligent Physical Agents". See <http://www.fipa.org/>



3.6 Innovation Over State-of-the-art

Research in the areas of heuristics and bargaining is ongoing. The project will:

- enhance state-of-the-art in meta-heuristics for complex planning problems in a multi-agent deliberation and bargaining setting, in particular:
 - faster, and more instance robust meta-heuristics for routing and scheduling;
 - methods to determine associated strong bounds on solution quality;
 - methods to predict quality improvement as function of additional computing time.
- further develop strategy, deliberation, and bargaining models based on knowledge and game theoretical concepts in different non-cooperative settings;
- improve multi-agent software technology specifically aimed at plan deliberation and bargaining;
- design domain specific Agent Communication Language for optimization and bargaining in complex planning situations.

An important innovative aspect of the project is the combination and integration of these technologies into a comprehensive framework for complex and autonomous problem solving, including a toolbox of reusable and adaptable software building blocks, enabling the assembly and configuration of real solutions to industrial applications.

3.7 Methodology

Method	Goals
Literature study	Define state-of-the-art.
Overall project planning	Define overall project structure, scope, work packages and dependencies, and deliverables. Positioning proof of concept.
Proof of concept	Develop a prototype suitable to provide a better understanding of critical technologies in order to aid decision making and reduce risks.
Vehicle Routing Problem analysis	Research state-of-the-art in VRP solving, describe VRP problem definition.
Object-oriented design	Realization of prototype
Experimentation	Deliver proof of concept.



4 Literature

In the course of the project a great number of papers and literature has been researched. Apart from literature about the basic project idea, most of the researched literature falls in one of the following categories:

- optimization, especially anytime algorithms;
- bargaining with focus on deliberation, decision and game models;
- agent based computing, standards and platforms.

The most important books and papers are discussed below.

4.1 Foundation

When thinking about Artificial Intelligence (AI) the book of *Stuart Russel and Peter Norvig [1]* is a natural choice for getting an overview of the state-of-the-art and getting acquainted with particular methods and techniques.

One of the issues in AI that is dominantly present is computational complexity: optimization, reasoning about deliberation, decision making often constitute very large solution sets and are computationally intractable. One of the directions advocated by Russell is Bounded Optimality. In his paper "*Provably Bounded-Optimal Agents*" [2] he states :

“... Since its inception, artificial intelligence has relied upon a theoretical foundation centered around perfect rationality as the desired property of intelligent systems. We argue, as others have done, that this foundation is inadequate because it imposes fundamentally unsatisfiable requirements. As a result, there has arisen a wide gap between theory and practice in AI, hindering progress in the field. We propose instead a property called “bounded rationality”. Roughly speaking, an agent is bounded-optimal if its program is a solution to the constrained optimization problem presented by its architecture and task environment. ... “.

This concept of bounded rationality lies at the core of the seminal paper for this project "*Bargaining with Limited Computation – Deliberation Equilibrium*" by *Tuomas Sandholm et al* from *Carnegie-Mellon University, Pittsburgh PA, USA*[3].

Abstract. We develop a normative theory of interaction—negotiation in particular—among self-interested computationally limited agents where computational actions are game theoretically treated as part of an agent’s strategy. We focus on a 2-agent setting where each agent has an intractable individual problem, and there is a potential gain from pooling the problems, giving rise to an intractable joint problem. At any time, an agent can compute to improve its solution to its own problem, its opponent’s problem, or the joint problem. At a deadline the agents then decide whether to implement the joint solution, and if so, how to divide its value (or cost). We present a fully normative model for controlling anytime algorithms where each agent has statistical performance profiles which are optimally conditioned on the problem instance as well as on the path of results of the algorithm run so far. Using this model, we introduce a solution concept, which we call *deliberation equilibrium*. It is the perfect Bayesian equilibrium of the game where deliberation actions are part of each agent’s strategy. The equilibria differ based on whether the performance profiles are deterministic or stochastic, whether the deadline is known or not, and whether the proposer is known in advance or not. We present algorithms for finding the equilibria. Finally, we show that there exist instances of the deliberation–bargaining problem where no pure strategy equilibria exist and also instances where the unique equilibrium outcome is not Pareto efficient.

Especially appealing in the paper of Sandholm et al. is the comprehensive and interrelated set of optimization, deliberation and bargaining technologies linked to a widely recognizable practical case. Although the deliberation and bargaining models presented in the paper are far from complete, they form promising topics for further research. Also the inclusion of response time limitations – here presented as processing time being the prime deliberation parameter, but easily extended to a more general idea of system reaction time



– in the model is quite appealing.

The first crucial prerequisite is the availability of an any-time approximation algorithm also providing a estimation of the solution quality reached at a particular point in time, but is also capable of estimating quality improvements given additional processing time as input for deliberation control.

In a related article to [3] the author published the paper "*Using Performance Profile Trees to Improve Deliberation Control*" [4]. In that paper an experimental study is presented of the accuracy of making stopping decisions for anytime algorithms on optimization problems. The authors claim, that performance tree profiles are feasible in practice and lead to significantly better deliberation control decisions.

4.2 Optimization

As mentioned in the previous paragraph an optimization technique is necessary that has the following properties:

- anytime i.e. the algorithm that can be stopped "anytime" during the optimization process and is capable of delivering the best result reached so far;
- must be capable of assessing the quality of the solution it presents;
- must be able to make an estimate of solution quality to be reached given additional computing time.
- must be able to create and maintain a Performance profile tree [3],[4] for that purpose.

While focusing on the case presented in [3] the Vehicle Routing Problem (VRP) (combination of the Traveling Salesman Problem and the Bin-packing problem, both in NP-hard) was the problem of choice. Various methods have been studied in the literature, the most important books were: "*Complexity and Approximation: Combinatorial Optimization Problems and their Approximation Properties*" by Ausiello et al. [7] and "*How to solve it: Modern Heuristics*" by Michalewicz et al. [6]. Traditional methods as Local Search, Greedy algorithms, Branch and Bound, Divide and Conquer and Evolutionary approach have been reviewed. The Evolutionary approach was chosen for the following reasons:

- the concept of generations seems natural for anytime algorithms as after each generation the best (= fittest) result is know;
- computing resources is easily conceived as computation steps each comprising a predefined number of generations;
- methods for monitoring asymptotic behavior are fairly easy to be implemented using the generation concept, thus providing means for solution quality estimation and prediction.

But most important, the following papers showed that the performance of the genetic algorithms for the VRP are good with the chromosome – solution transformation and variation operators the authors designed and used in their study. These ideas provide a promising starting point for the design and implementation of variation operators in the VRP solution.

"GVR: A new Genetic Representation for the Vehicle Routing Problem" by Pereira et al. [9]

Abstract. In this paper we analyze a new evolutionary approach to the vehicle routing problem. We present Genetic Vehicle Representation (GVR), a two-level representational scheme designed to deal in an effective way with all the information that candidate solutions must encode. Experimental results show that this method is both effective and robust, allowing the discovery of new best solutions for some well-known benchmarks.



"Crossover and Diversity: A Study about GVR" by Tavares et al. [10]

Abstract. Genetic Vehicle Representation (GVR) is a two level representational scheme, designed to deal in an effective way with all the information needed by candidate solutions, for the Vehicle Routing Problem (VRP). In this paper, we present an analysis on the influence of two crossover operators in the algorithm performance. A first study on diversity is also presented, regarding the issues of diversity measurement and possible relations to the algorithm performance. Results show that for GVR one type of crossover is more suited for solving VRP instances, and both operators may not avoid the loss of diversity. Nevertheless, solutions discovered by GVR are competitive and are the best ones found by an evolutionary algorithm.

Monitoring performance features for meta-control of an algorithm is a problem area in itself. The paper of Hansen and Zilberstein "Monitoring and Control of anytime algorithms: A dynamic programming approach" [8] provided valuable ideas for designing the meta-control solution the VRP algorithm.

Abstract Anytime algorithms offer a tradeoff between solution quality and computation time that has proved useful in solving time-critical problems such as planning and scheduling, belief network evaluation, and information gathering. To exploit this tradeoff, a system must be able to decide when to stop deliberation and act on the currently available solution. This paper analyzes the characteristics of existing techniques for meta-level control of anytime algorithms and develops a new framework for monitoring and control. The new framework handles effectively the uncertainty associated with the algorithm's performance profile, the uncertainty associated with the domain of operation, and the cost of monitoring progress. The result is an efficient non-myopic solution to the meta-level control problem for anytime algorithms.

In addition a number of benchmarks are available and well researched. For more information see link: [VRPWeb](#)

4.3 Bargaining

The literature study with regard to deliberation and bargaining has been guided by obtaining a basic understanding of deliberation control and decision theoretic models. The standard work "Game Theory: Analysis of Conflict" by Meyerson [11] provides a good insight into the basic models, Nash and Bayesian Equilibria, pure and mixed strategies and the various forms of games and bargaining. In "Reasoning about Uncertainty" by Halpern [12] probability is explored for belief systems, decision rules and multi-agent systems.

In order to get a feel of the state of the art regarding the tractability of reasoning the paper "Tractable Reasoning via Approximation" by Schaerf e.a [13] though quite formal in its content, gave some insight in this area.

Abstract. Problems in logic are well known to be hard to solve in the worst case. Two different strategies for dealing with this aspect are known from literature: language restriction and theory approximation.

In this paper we are concerned with the second strategy. Our main goal is to define a semantically well-founded logic for approximate reasoning, which is justifiable from the intuitive point of view, and to provide fast algorithms for dealing with it even when using expressive languages. We also want our logic to be useful to perform approximate reasoning in different contexts. We define a method for the approximation of decision reasoning problems based on multi-valued logics. Our work expands and generalizes in several directions ideas presented by other researchers. The major features of our technique are: 1) approximate answers give semantically clear information about the problem at hand; 2) approximate answers are easier to compute than answers to the original problem; 3) approximate answers can be improved and eventually they converge to the right answer; 4) both sound approximations and complete ones are described.

The method we propose is flexible enough to be applied to a wide range of reasoning problems. In our research we considered approximation of several decidable problems with different worst-case complexity, involving both propositional and first-order languages. In particular we defined approximation techniques for: propositional logic, fragments of first order logic (concept description languages) and modal logic. In our research we also addressed the issue of representing the knowledge of the reasoner with limited resources and how to use such knowledge for approximate reasoning purposes.



4.4 Agent Based Computing

Literature research in this area has been guided by search for de facto standards and the choice of a compliant agent-platform package.

The standardization body for agents is Foundation for Intelligent Physical Agents (FIPA) <http://www.fipa.org/>. The foundation has published numerous documents, the most relevant ones, used in the overall architecture, are mentioned here.

FIPA Abstract Architecture Specification [15]

This document is organized into the following sections and a series of annexes.

- Introduction.
- The Scope and methodology section explains the background of this work, its purpose, and the methodology that has been followed. It describes the role of this work in the overall FIPA work program and discusses both the current status of the work and way in which the document is expected to evolve.
- The Themes of the FIPA Abstract Architecture section that explains the style and the themes of the FIPA Abstract Architecture specification.
- The Architectural overview presents an overview of the architecture with some examples. It is intended to provide the appropriate context for understanding the subsequent sections.
- The Architectural Elements section comprises the FIPA Abstract Architecture components.
- The Agent and Agent Information Model defines UML pattern relationships between Architectural Elements.

The annexes include:

- Goals of Service Model.
- Goals of Message Transport Service Abstractions.
- Goals of Directory Service Abstractions.
- Goals for Security and Identity Abstractions.

FIPA Agent management Specification [16].

This document contains specifications for agent management including agent management services, agent management ontology and agent platform message transport. This document is primarily concerned with defining open standard interfaces for accessing agent management services. The internal design and implementation of intelligent agents and agent management infrastructure is not mandated by FIPA and is outside the scope of this specification. The document provides a series of examples to illustrate the agent management functions defined.

FIPA Network Management and Provisioning Specification [17].

Across the world, numerous telecommunications service providers combine service elements from different network providers in order to provide a single service to end customers. The ultimate goal of all parties involved is to find the best solutions available in terms of quality of service and cost. The increasing demand for on-line customer configurable services and on-line provisioning of services requires systems and networks that are capable of co-operating on different levels and that transcend conventional business and national boundaries.

The dynamic Virtual Public Network (VPN) service is a telecommunications service provided to users that want to set up a multimedia connection with several other users. The provisioning of a dynamic VPN service is an example of how service providers and network providers will have to co-operate in order to provide this to the end-customer.

Traditional network management frameworks (for example, TMN or SNMP-based solutions) are based upon fixed management functionality and fixed interaction interfaces that cannot easily satisfy the flexibility and complexity that the dynamic multimedia VPN service demands. Agent technology is



promising in this domain since it facilitates automatic negotiation of service contracts and subsequent configuration of those services, thus enhancing the provisioning process for the users and administrators of dynamic multimedia VPN services.

FIPA agents, which can interact using ACL, have significant advantages in this context. In summary FIPA agents can:

- Support effective negotiations that will be complex,
- Support dynamic service and service condition configuration via knowledge exchange,
- Reduce the dependency on the network reliability and availability by encapsulating the negotiation functionalities in ACL messages,
- Provide friendly and enhanced customer support via agency and,
- Support personalization of the service resource configuration and utilization using more detailed knowledge about users and providers and their preferences.

FIPA Personal Travel Assistance Specification [18]. An interesting case is being presented in the paper to illustrate the design of an application architecture. The ideas in the document have been used to describe the overall architecture (Part 2 – Proof of Concept).

This document extends the FIPA standard by providing an application specification for the travel industry. This specification provides:

- An overview of the current industry in regard to agents,
- A reference architecture for a multi-agent system in this industry,
- Examples of the agent management details such as domains and naming,
- Examples of agent communication details such as content ontologies and communication protocols and,
- Examples of agent/software integration such as for accessing databases and mobile users.

This specification is not complete, but the included examples help to illustrate the use of FIPA standard and thereby quicken the development and deployment of real systems. Some points of this architecture have been selected as normative in order to begin interoperability tests of field trials. These requirements are noted throughout the specification as they arise.

In summary, this specification serves three purposes:

- To continue testing the FIPA technical specifications. The context of a real application serves to determine the strengths and weaknesses of the specifications,
- To demonstrate the real business value and requirement of a standard specification for such a large, distributed, multi-vendor application, and,
- To define initial application architecture, object design and use case analysis for actual development of field trials.

The choice for the JADE platform is based on FIPA conformance, its rich functionality and its Java based platform independence.

"JADE: Java Agent Development Framework - A White Paper" by Bellifante et al. [14] (<http://jade.tilab.com/papers/2003/WhitePaperJADEEXP.pdf>) gives a fair overview of the design, implementation of the platform. The package also provides comprehensive Programming and User Guides. The package is actively developed further.



This page is intentionally left blank.



5 Architecture

5.1 Bargaining Model

The main objective of the project is to provide a means for decision making about pooling of resources and profit distribution. For each user, the problem is delegated to an agent, who after proper instruction will execute the negotiation process with an opposing agent, being the representative of the competitor.

The outcome of that process is a decision to cooperate or not and if so, how the revenues will be shared between the firms. During this process the agents will exchange information like proposals, bids, acceptance/denial messages and so on. This is a dynamic situation: the agents message to its opponent is depending not only on the message received, but also on the deliberation process, bargaining strategy and - of course - the current state-of-affairs. The mechanism generating the stimuli and responses is formulated in a bargaining model. Each agent operates according its own instance of this model, comprising its own strategies and belief system.

The Bargaining Model can be described as the function BMODEL:

BMODEL: $\Gamma \rightarrow \Gamma$

BMODEL_A and BMODEL_B are instances of the BMODEL for Agents A and B respectively:

BMODEL = $\{\{capVRP_{\alpha}, capVRP_{\beta}\}, \mathfrak{R}, \Gamma, \Omega\}$ where $\mathfrak{R}, \Gamma, \Omega$ are finite sets and

- $\{capVRP_{\alpha}, capVRP_{\beta}\}$ are instances of the Capacitated Vehicle Routing Problem;
- \mathfrak{R} is a set of properties describing the bargaining environment;
- Γ is an ontology suitably defined for the domain. The ontology is defined as a set of frames: $\Gamma = \{\text{frame-name, description, \{parameter, description, presence, type, \{reserved values\}\}}\}$. An ontology also contains the "empty" frame;
- Ω is a set of properties, describing the *state-of-affairs* and *beliefs*.

A number of the entities of the model – ontology, agent communication language, state-of-affair and belief system – are identified as research topics in chapter 3.5: Research Challenges. The CapVRP problem and the relevant subset of environment properties are identified and described in Part 2 paragraph 8.4 "CapVRP Problem Definition" and is the first part of the prototype to perform a proof-of-concept.

5.2 Functional Model

The figure 5.2.1 gives an overall, high-level view of the functions within the bargaining model introduced previously.

On the one hand a calculation function is recognized with its control functions, on the other hand a strategy function is identified, which controls both the deliberation as well as the bargaining processes of the agent.

In order for the agent to execute its deliberation strategy, information about options and its effect on resource utilization is needed. The interaction between the strategy and calculation functions is managed by the Deliberation Control function.

On top the Bargaining Control function coordinates all agent's activities and the interaction with the opposing agent via the platform's communication facilities and the agent communication language (ACL).

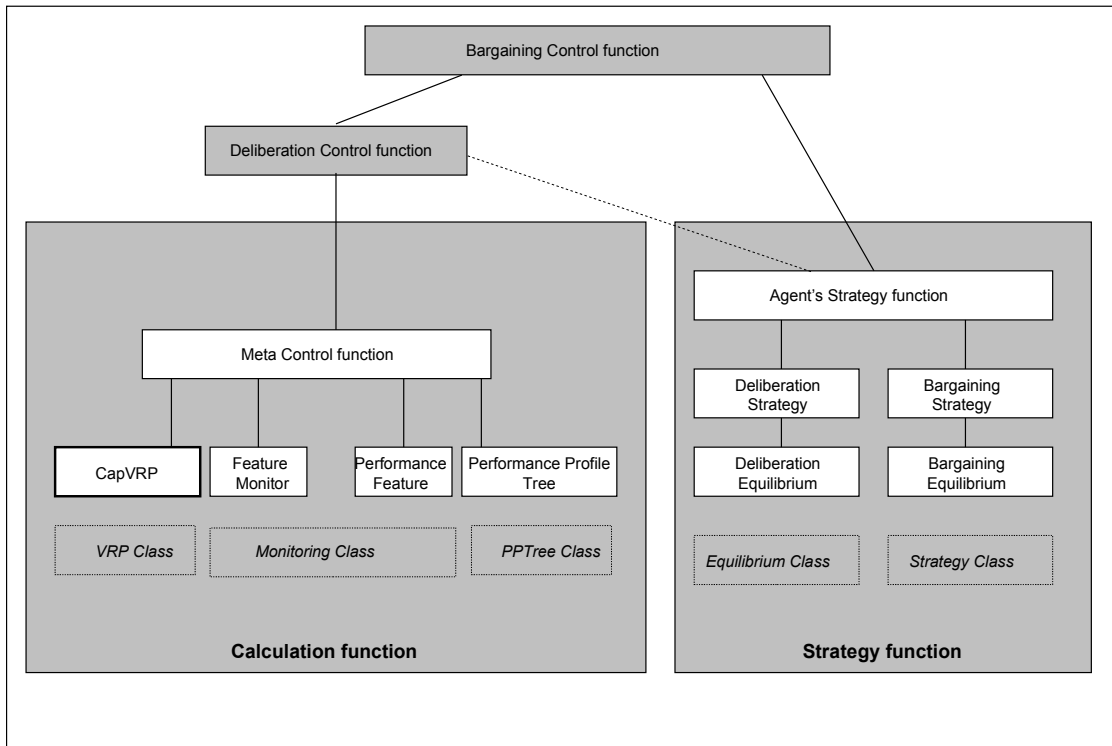


Figure 5.2.1 Functional Model

5.3 Service Platform Architecture

As indicated earlier, the functionality will be realized by means of rational agents. Agents typically live within agent platforms, that provide services like [14][15][16][17]:

- **Agent life cycle services.** These services initiates an agent and manages its states such as active, waiting, suspended etc. during its life cycle. At the end of its life cycle the agent is destroyed.
- **Agent communication services.** Agents must communicate with each other. They do that via an Agent Communication Language (ACL) based on a domain specific ontology. The platform offers Message Transfer Services as the communication vehicle.
- **Directory services.** Agents must be able to identify other agents – also on other platforms – that can provide solutions to problems they have. For example a travel-agency agent might want information on travel options. It might decide to consult airline agents and request proposals. Finding these agents is possible via directory services. Or conversely, an agent might wish to advertise is services and can do so by registering with the directory service.

In figure 5.3.1 three distinct type platforms are distinguished: Company Platform, Bargaining Platform and Computational Services Platform. The reason for this distinction is, that the physical platforms may be located in different places and/or managed by different service providers. For instance the Company platform may be part of the company's own IT infrastructure, whereas the Computational Services may be offered by specialized providers, delivering these services to a variety of clients.

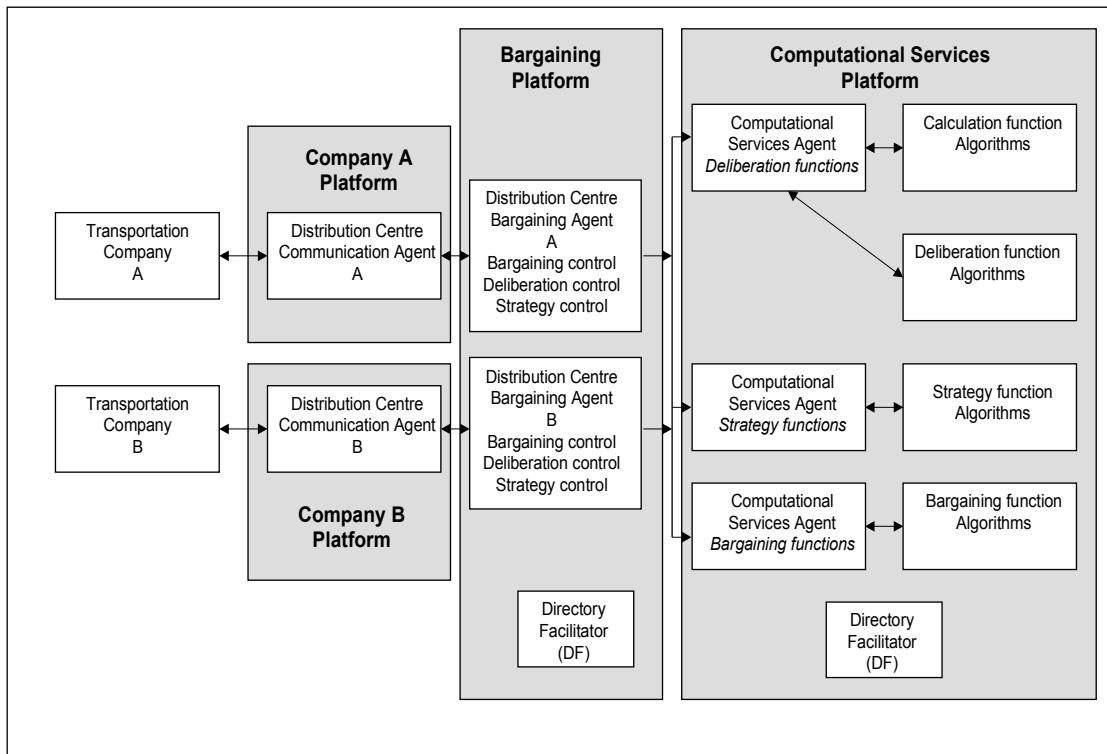


Figure 5.3.1 – Service Platform Architecture

Also the agents have different roles. In the above figure the agent roles are – not surprisingly – implementations of the functional model:

- **Distribution Center Communication Agent.** This agent represents the interests of the company and is responsible for the communication between the company and its distribution center. This agent typically is charged with “Boundary Control” tasks and may expose a Graphical User Interface (GUI) for interaction with the user.
- **Distribution Center Bargaining Agent.** This agent is responsible for the deliberation and bargaining process based on the instructions received from the Communication Agent. The agent uses computational services from Computational Services Agents for deliberation and bargaining. The agent reports the results of the bargaining process back to the Communication Agent.
- **Computational Services Agent.** These agents provide a variety of computational services. They also feature a number of choices in algorithms and environment settings, that requesting agents can use, thus customizing their services to their particular strategies. Although opposing agents may use the same algorithm such as the CapVRP with the same instances, the computational results may be quite different, depending of the strategic choices the have made.



5.4 Platform Implementation

As argued in the previous paragraphs, and the open nature of agent platforms, adherence to standards is essential. The current standardization body for agent systems is FIPA. The JADE platform is an implementation of these standards, offering the basic services required.

FIPA and JADE use a particular definition language to define the various entities. For illustration and orientation purposes only, some examples of definitions are presented below.

Distribution Centre Bargaining Agent

This agent is responsible for the deliberation and bargaining process.

```
(DCBA-agent-description
  :name
    (agent-identifier
      name: Bargainer1@bargain.novum-it.nl
      :addresses (sequence iiop://bargain.novum-it.nl/acc))
  :type fipa-dcba
  :services (set
    (service-description
      :properties (set property
        :name Profile
        :value TransportPlan))
      :ontology (set FIPA-DCBA)
    (service-description
      :properties (set property
        :name ...
        :value ...))
      :ontology (set User)))
  :protocol (set FIPA-Request FIPA-Contract-Net)
  :ontology (set User FIPA-DCBA)
)
```

Company Agent Platform

```
(ap-description
  :name company1.com
  :dynamic false
  :mobility false
  :properties (set
    (property
      :name Change-Environment
      :value Administrator)
    (property
      :name Delegation-Allowed
      :value (set (DCCA.User) (DCCA.Administrator)))
    (property
      :name Grant-Services
      :value Within-Platform)
    (property
      :name Acces-DF
      :value Within-Platform))
  :transport-profile ...
)
```



Table 5.4.1 – Example agent directory

Directory facilitator	Registered Agents
Company platform df@company1.com	DCCA@company1.com bargainer@bargain.novum-it.nl
Bargaining platform df@bargain.novum-it.nl	Bargainer1@bargain.novum-it.nl Bargainer2@bargain.novum-it.nl ... Bargainer-n@bargain.novum-it.nl Comp-services@comp-services.novum-it.nl Computations@who-ever.com
Computational services df@comp-services.novum-it.nl	Transport-scheduler1@comp-services.novum-it.nl Transport-scheduler2@comp-services.novum-it.nl Deliberator1@comp-services.novum-it.nl Deliberator2@comp-services.novum-it.nl etc...



This page is intentionally left blank.



6 Implementation

6.1 Project Structure

The project is structured according the standard problem solving cycle, consisting of the following phases:

- Project initiation.
- Requirements analysis.
- Problem analysis and solution design.
- Solution development.
- Verification and delivery.

The work has been divided into work packages clustering together coherent tasks and activities. The work package setup follow in broad lines this problem solving structure.

Table 6.1.1 – Work packages

#	Phase	Work Package(s)	Description
1	Project Initiation	WP1 – Consortium Management WP2 – Collaborative Environment	The consortium agreement and agreement with the Commission has been concluded. The program management structure is being put in place. The project Lifecycle Model has been decided. The project office and project working space equipment and network is installed. The supporting Hardware and software infrastructures are installed and made operational. The common procedures for management, reporting, use of infrastructure and knowledge management etc. as defined in the Configuration Management Document are being communicated and installed. The Phoenix Intranet is populated with the necessary starting information. A kick-off meeting is held including team building activities.
2	Requirements Analysis	WP3 – Business Cases	The business environments in Manufacturing and Logistics are being analyzed and opportunities for profitable deployment of the Phoenix technologies identified. Valuation of opportunities lead to selection of optimization targets. Selected problem areas are described in solution requirements and profit targets.
3	Problem Analysis and Solution Design	WP4 – Heuristics WP5 – Bargaining WP6 – Solution Architecture WP7 – Agent Communication	Based on the business case analysis further analysis, abstraction and model building is performed for the optimization, bargaining and agent behavior. Various solution models are prototyped and evaluated. The overall solution architecture is set up. The communication language between agents is determined.



#	Phase	Work Package(s)	Description
4	Solution Development	WP8 – Agent Platform WP9 – Software Engineering	Based on the solution design the implementation in agent, agent platform and software components is analyzed and designed. The platform will be installed and configured. Software components are build and tested. Application configurators and standard software is installed. Software verification is being performed.
5	Verification and Delivery	WP10 – Demonstration WP11 – Exploitation and Dissemination	The solutions for the business cases as defined in phase two are now implemented with the software build. Case input has been generated and extensive testing and simulation is being performed. Results are being checked against requirements and targets, deviations analyzed and repaired. All results are documented including analysis, best practices, further development plans and exploitation guidelines. Exploitation and dissemination plans developed and executed. Project conclusion.

The overall management structure and work package allocation is depicted in the following figure 6.1.1 – Project management structure.

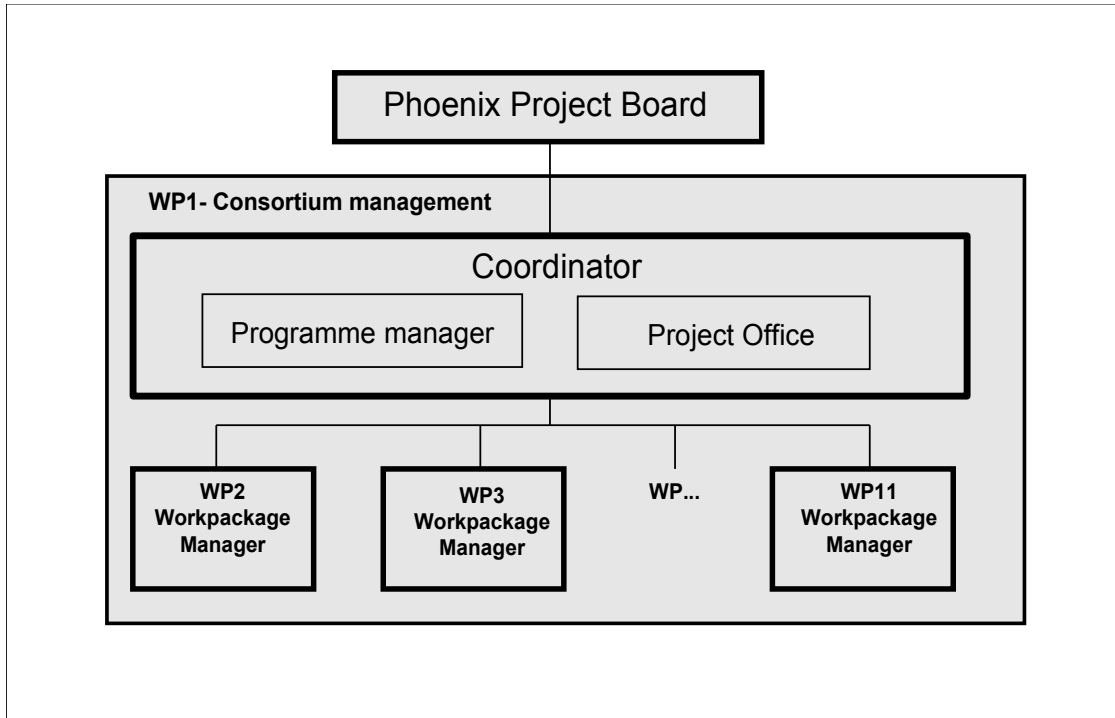


Figure 6.1.1 – Project management structure



The structure described here is presented as a sequence of phased activities. This is done for clarity only. In reality however, due to the R&D nature of the project and the (computational) complexity of the optimization and bargaining models, and algorithms, much of the work will be done in short prototyping cycles. This means that the phases three (analysis, design) and four (software development) and five (verification, WP10 Demonstration) will run in parallel for a considerable amount of time, with numerous iterations between those activities.

6.2 Risk And Risk Mitigation

Research and development projects like Phoenix are inherently risky. The major risk components and the measures to minimize these risks are listed below.

Table 6.2.1 – Risks and risk mitigation

Risk	Qual.	Mitigation
Project complexity Research and development project with many new technologies, executing in a distributed international heterogeneous and virtual environment.	High	Mitigating measures <ul style="list-style-type: none"> • Work breakdown in work packages of limited size • Project office with meeting room, project room and communication facilities • Well set up ICT infrastructure based on accepted Configuration Management Standards to support collaboration in virtual teams • Adequate travel budget to allow for coordination meetings and prolonged teamwork.
Client factors User SME's might have limited expertise and/or resources. Priorities might get compromised by daily work pressure.	High	Mitigating measures <ul style="list-style-type: none"> • The User SMEs are bound to the project through consortium members • Conduct management seminars to inform management on potential of the technologies and the progress being made.
Research and technological factors Research and Technological problems might prove harder or even impossible to solve, leading to strategy changes along the way.	High	Mitigating measures <ul style="list-style-type: none"> • Employing key people on Director/Professor level on the project • Involving high level expertise with an outside-in view through the Advisory Boards structure • Facilities for teamwork and problem solving
Software engineering Software development might take longer than expected.	High	Mitigating measures <ul style="list-style-type: none"> • Usage of proven and open standards • Making use of existing solutions for technological non-critical parts in the project • Scope adaptation for non-functional/non-critical requirements



6.3 Work Packages

Work-package No	Workpackage title	Person-months ¹	Objective
1	Consortium Management Package	14	Management of the overall program in order to ensure: <ul style="list-style-type: none">• coordination of the technical activities;• overall legal, financial, ethical, contractual and administrative issues are addressed;• knowledge management;• scientific and societal issues resolved;• ensure quality management;• ensure proper communication with the Commission.
2	Collaborative Environment Package	6	Set up and maintain the operational working environment for the consortium to enable functioning as a virtual team. The environment includes provisions for project management, content management, work process management and intra team communication.
3	Business Case Package	30	Describes the business impact of applying the developed technologies in two concrete business situations. The business cases concern typical real-life complex optimization problems in logistics and manufacturing where pooling of resources may be mutually beneficial for both parties. For each pooling opportunity, decision making and profit sharing is based on non-cooperative bargaining. The cases will serve amongst others as linking the ambient intelligence technologies to value creation in the area logistics and manufacturing.
4	Heuristics Package	17	High-performance, self-adaptive optimization algorithms specifically designed for solving targeted, computationally complex planning problems in a multi-agent, bargaining setting. The algorithms will be designed for high speed and self-adaptability, to meet the high demands on plan quality versus response time, and robustness towards a wide variety of instances. Sub-goals are generic, rich mathematical and conceptual model; model analysis and algorithmic design; solution quality measures and predicted improvement behavior; suite of industrial test instances; Experimental results.
5	Bargaining Package	23	High-performance, self-adaptive strategy, deliberation and bargaining algorithms specifically designed for solving

¹ The total number of person-months allocated to each work package.



Work-package No	Workpackage title	Person-months	Objective
			targeted, computationally complex bargaining problems in a multi-agent, bargaining setting. The algorithms will be designed for high speed and self-adaptability, to meet the high demands on plan quality versus response time, and robustness towards a wide variety of instances. Sub-goals: generic, rich mathematical and conceptual model; model analysis and algorithmic design; suite of industrial test instances; experimental results
6	Solution Architecture Package	5	Definition the overall solution architecture in terms of processes, interfaces, components, platforms and standards. The architecture serves as blueprint for software development and solution delivery.
7	Agent Communication Package	14	Provides a detailed analysis of the agent communication requirements and adoption/design of ontologies, grammar and semantics of communication acts. The objective is to ensure, that agents are able to communicate in a standardized way with other agent in their problem domain.
8	Agent Platform Package	4	Provides an operational agent platform set-up, based on the architectural requirements and JADE software suite.
9	Software Engineering Package	139	Produce the software component suites according the appropriate Software Engineering Lifecycle methodology
10	Demonstration Package	40	Provide proof-of concept based on the targets set in the business cases. Accumulate experience and best practice information to be applied at exploitation and dissemination.
11	Exploitation and Dissemination Package	8	Ensure and facilitate optimal dissemination and exploitation of the result of the project. Investigate and facilitate strategic alliance with international IT service industry.
	TOTAL	300	



6.4 Consortium

Nr	Name	Description	Background	Role ¹
1	Novum	Novum Information Technology BV Leiden The Netherlands http://www.novum-it.com	<p>Novum Information Technology BV was founded in 1990. Novum is a company initially specializing in strategic IT consultancy and programme/project management. During the past 5 years Novum has specialized in unlocking knowledge technologies for business and accordingly restated its mission. Novum focuses on designing and producing excellent technical solutions.</p> <p>Novum started the Phoenix project in order to build and deliver a proof-of-concept and competence in the area of non-cooperative and computationally bounded bargaining. As Phoenix's underlying technologies and decision model has such great potential we decided to build an international consortium to develop these technologies and submit the project as a European Specific Targeted Research Project (STREP).</p>	Coordinator Vendor
2	TU Delft	Delft University of Technology Delft The Netherlands	<p>Delft University of Technology (TU Delft) is an entrepreneurial university at the forefront of technological developments; advancing the state of technology further on behalf of society through fundamental and applied research and educational programmes. With approximately 13,000 students and 2,100 scientists (including 200 professors) TU Delft is the largest and most comprehensive university of engineering sciences in the Netherlands.</p> <p>TU Delft - faculty of Electrical Engineering, Mathematics and Computer Science - will contribute scientific research and development in Mathematical modelling and Metaheuristics, Knowledge systems and Software engineering.</p>	University

-
- 1 Roles:
- Coordinator: Overall programme management of the consortium.
 - Vendor: Will bring developed technologies to market. Cooperation in consortium is pre-competitive. Will have a major contribution in software engineering and development.
 - Research: Contributes in research and development of (mathematical) models and proofing. May also contribute to software engineering and development.
 - User: Represents industries as potential user of the developed technologies. Contributes typically in Business Case development, requirements and modeling definition and evaluation. Will also play a major role in Demonstration setup and evaluation.



Nr	Name	Description	Background	Role
3	Tekever	Tekever Lisbon Portugal	Tekever is a European information technology SME created in January, 2001 and based in Lisbon, Portugal, with establishments in Silicon Valley (USA), Beijing (China) and São Paulo (Brazil). The main activity is software development for embedded systems, Internet, wireless networks (GSM, GPRS, UMTS) and graphical user interfaces (from simple GUI to 3D visualization). Tekever Staff is composed mainly by Engineers with wide experience in several kinds of software and hardware development projects. This experience was acquired in projects ranging from integration of hardware and complex real-time systems to multiple Artificial Intelligent agents development. Currently the total staff is around 60 persons, in which 95% has graduate or post-graduate degrees, including software, electronic, chemical, physics and aerospace Engineers.	Vendor
4	Polimatica	Polimatica Srl Rivoli Italy	<p>Polimatica S.r.l. was founded in 1982 and today has about 85 employees with an annual revenue of about 4.2 Millions of Euro.</p> <p>The activity of Polimatica covers currently the following for Phoenix relevant areas:</p> <ul style="list-style-type: none"> • Supply Chain Execution (Factory Information Systems and Real-Time Microprocessor based systems), • Supply Chain Management (Logistics and Distribution, Production Planning, Demand Planning, etc.), • E-Business (E-Commerce, E-Acquirement, Customer Relationship Management, etc.). <p>In 1993-1998 Polimatica designed a Plant Supervision and Control (PSC) system currently installed on 2 big plants of the Italian aerospace manufacturer Company Alenia Aeronautica S.p.a. In 1999-2002 Polimatica developed a Framework for building Business-to-Business and Business-to-Consumer E-Commerce Applications.</p>	Vendor User ¹

¹ Polimatica will bring in an End User partner in Logistics. In the initial phase Polimatica will assume the role as User as well.



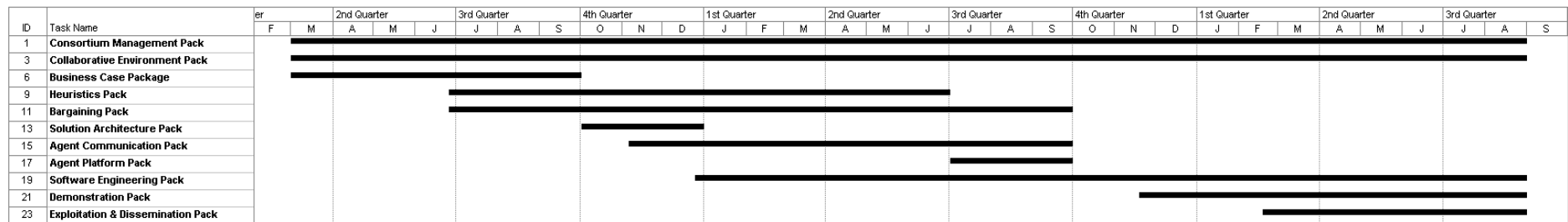
Nr	Name	Description	Background	Role
5	Sintef	Sintef ICT Oslo Norway	<p>SINTEF is a large Norwegian multi-disciplinary contract research institute with some 1750 employees. We put together cross-disciplinary teams of research scientists to meet versatile demands from clients in the best way possible.</p> <p>The Department of Applied Mathematics specializes in making solutions for real-life, computationally intractable discrete optimization problems using its expertise in meta-heuristics, integer programming, constraint programming, and software engineering. In most projects, the main deliverable is software in the form of an optimization kernel. The main market areas are transportation, manufacturing, supply-chain management, forestry, the health sector, and finance.</p>	Research
6	Fraunhofer	Fraunhofer IFF Inst. for Factory Operation and Automation Magdeburg Germany	<p>The Fraunhofer IFF is an autonomous research institute of the Fraunhofer Society. The Fraunhofer IFF employs about 100 scientists from various disciplines working on areas as factory and product planning, production logistics, quality management, maintenance and special application of factory operation.</p>	Research User ¹

¹ Fraunhofer IFF will bring in an End User partner in Manufacturing. In the initial phase Fraunhofer IFF will assume the role as User as well.



6.5 Work Planning

Gantt Chart



Pert Diagram

The Pert diagram is not a suitable representation for this type of project, because of the concurrent and cyclic nature of the work and the interactions between analysis, design and software creation and demonstration. The Pert representation would falsely emphasize the notion the waterfall method would be applicable here, a method which is totally inadequate for this type of research project.

The evaluation of more robust Lifecycle methods for R&D type of projects is part of WP2 and will be decided by the Project Coordination Committee at Project Initiation. Possible candidates are: Boehm's Spiral Model, Saw/Shark Tooth Model, Unified Process, Issue-Based Lifecycle Model, etc.



6.6 Detailed Work Description

Work Package List

Work-pack No ¹	Work package title	Lead contractor ²	Person-months ³	Start month ⁴	End month ⁵
1	Consortium Management Package	1	14	0	29
2	Collaborative Environment Package	1	6	0	29
3	Business Case Package	6	30	0	6
4	Heuristics Package	5	17	4	15
5	Bargaining Package	2	23	4	18
6	Solution Architecture Package	1	5	7	10
7	Agent Communication Package	3	14	8	18
8	Agent Platform Package	4	4	16	18
9	Software Engineering Package	1	139	9	29
10	Demonstration Package	6	40	20	29
11	Exploitation & Dissemination	1	8	23	29
TOTAL			300		

1 Work package number: WP 1 – WP n.

2 Number of the contractor leading the work in this work package.

3 The total number of person-months allocated to each work package.

4 Relative start date for the work in the specific work packages, month 0 marking the start of the project, and all other start dates being relative to this start date.

5 Relative end date, month 0 marking the start of the project, and all ends dates being relative to this start date.



Deliverables List

Deliverable No ¹	Deliverable title	Delivery date ²	Nature ³	Dissemination level ⁴
D1.1	Progress reports comprising Periodic WP-progress reports, Various reports and Final report	3, 6, 9, ..., 29	R	RE
D2.1	Configuration Management Document, Lifecycle Model and Operational Environment	2	R + D	CO
D2.2	Continuous services to Consortium and Project	3-29	D	RE
D3.1	Business Case Manufacturing Analysis Document	7	R	RE
D3.2	Business Case Logistics Analysis Document	7	R	RE
D4.1	Paper on the mathematical and algorithmic description of the model including motivated choice and statistical analysis	15	R	PU
D4.2	Test bench implementation of optimization algorithms	15	R	CO
D5.1	Paper on the mathematical and algorithmic description of the game models including motivated choice	18	R	PU
D5.2	Test bench implementation of bargaining algorithms	18	R	CO
D6.1	Architecture document	10	R	RE
D7.1	ACL specification document	18	R	PU
D7.2	Test bench implementation of constructor and parser	18	R	CO

1 Deliverable numbers in order of delivery dates: D1 – Dn

2 Month in which the deliverables will be available. Month 0 marking the start of the project, and all delivery dates being relative to this start date.

3 Please indicate the nature of the deliverable using one of the following codes:

- R = Report
- P = Prototype
- D = Demonstrator
- O = Other

4 Please indicate the dissemination level using one of the following codes:

- PU = Public
- PP = Restricted to other programme participants (including the Commission Services).
- RE = Restricted to a group specified by the consortium (including the Commission Services).
- CO = Confidential, only for members of the consortium (including the Commission Services).



Deliverable No	Deliverable title	Delivery date	Nature	Dissemination level
D8.1	Report of Design and Engineering rules for Agent Platforms	18	R	CO
D8.2	Report of Agent platform configuration and setup	18	R	CO
D9.1	Software component libraries and Documentation	29	O	CO
D10.1	Evaluation reports	29	R	RE
D10.2	Best Practices Guidelines	29	R	CO
D11.1	Dissemination collateral	29	O	PU
D11.2	Exploitation and alliance collateral	28	O	CO

6.7 Project Organization

This Plan for management and control is part of the negotiations between the Parties and must lead to the formal Consortium Agreement, which needs to be in place before the EC Contract can be awarded.

The Project Team will use the draft "Consortium Agreement For Specific Targeted Research Projects in the FP6" as a guideline.

Organization Structure

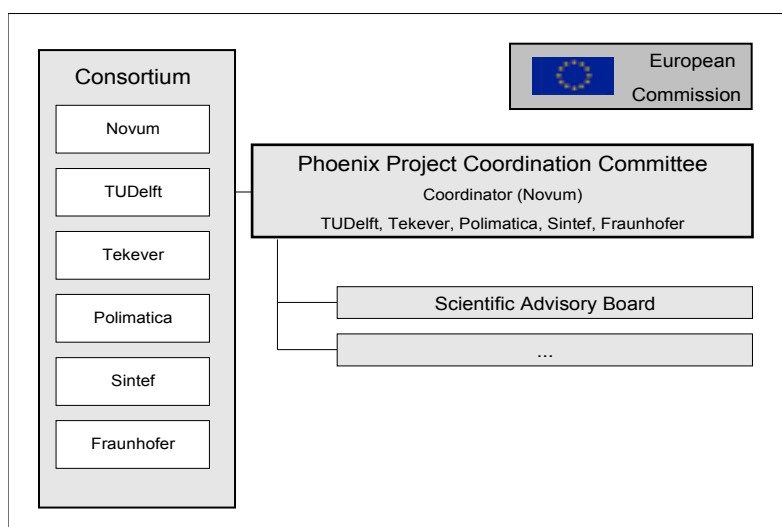


Figure 6.7.1 – Project structure



General Structure

The initial organization structure of the Consortium shall comprise the following:

- Project Coordination Committee is the supervisory body for the project execution and decision-making body in all relevant project matters.
- Panels can be established by the Project Coordination Committee to deal with specific issues or problems. The following panels are being considered: Scientific Panel, Software Engineering Panel, Disseminating Panel, Demo Panel and Financial Panel.
- Project Coordinator is the intermediary to the Commission is authorized to execute the project management, shall report and be accountable to the Project Coordination Committee.

Management Structure

The Project Coordination Committee

Each Party agrees to nominate a representative to the Project Coordination Committee with due authorization to discuss, negotiate and agree decisions or provide recommendations made by the organs within the frame of their responsibilities.

Responsibilities

The Project Coordination Committee shall coordinate the Project.

The Project Coordination Committee assumes overall responsibility for liaison between the Parties in relation to the Project, for analyzing and approving the results, for proper administration of the Project and for implementation of the provisions contained in the Consortium Agreement.

The Project Coordination Committee shall be responsible for:

- supporting the Coordinator in fulfilling obligations towards the Commission;
- ensuring that all work meets functional requirements;
- providing Project management in relation to the activities of the Panels on technical, financial and/or exploitation/ dissemination issues, as applicable;
- agreeing on press releases and joint publications by the Parties with regard to the Project;
- agreeing on procedures and policies for Dissemination of Knowledge from the Project which is not to be used by the Parties;
- checking the progress of the works;
- coordinating the research teams;
- advising and directing the Parties on the developments necessary for the Project.

Decision Structure

All decisions of the Project Coordination Committee as the principal body of the Consortium are legally binding for all Parties.

The Project Coordination Committee decides in cases of:

- coordination, preparation and final approval of reports (technical, financial, etc.) prior to the submission to the Commission;
- all budget-related matters;
- definition, allocation of tasks and changes in work sharing;



- the structure and restructuring of the Project;
- the alteration of the Consortium Agreement;
- the premature completion/ termination of the Project;
- the exclusion of Parties and the acceptance of new parties.

Any decision requiring a vote at a Project Coordination Committee meeting must be identified as such on the pre-meeting agenda, unless there is an unanimous agreement to vote on a decision at that meeting and three-quarters ($\frac{3}{4}$) of the members of the Project Coordination Committee are present or duly represented by proxy.

Advisory Boards

The Project Coordination Committee shall have the right to set up Advisory Boards to advise and support the Project Coordination Committee in the proper management and coordination of the Project.

Typical roles of the Advisory Board are:

- advise on expert issues;
- provide opinions on strategy and road mapping;
- perform audits and other Quality Assurance activities;

Advisory Boards may have experts from outside the Consortium.

The Coordinator

The Coordinator is the single point of contact between the Commission and the Consortium. In this function the Coordinator shall sign the Contract with the Commission after authorization by all Parties who have signed the Contract forms (Forms A and B) and the Consortium Agreement.

Responsibilities of the Coordinator

Pursuant to the Contract, the Coordinator is responsible for the following tasks and functions:

- overall management of the Project with the support of a Project Team, if necessary;
- chairing the Project Coordination Committee;
- preparation of the meetings and decisions of the Project Coordination Committee;
- timely collection and, with the support of the Project Coordination Committee, preparation of statements, including financial audit certificates, from the Parties for transmission to the Commission;
- ensure prompt delivery of all hardware, software and data identified as deliverable items in the Contract or requested by the Commission for reviews and audits, including the results of the financial audits prepared by independent auditors.

The Coordinator shall not be entitled to act or to make legally binding declarations on behalf of any other Party without written authorization of that Party.

Establishment of the Work Plan

The Project Coordinator and the Project Coordination Committee shall be responsible for the development, extrapolation and harmonization of the Work Plan and shall propose specific procedures in decision-making relating these issues.

The Project Coordinator has to inform the Project Coordination Committee about any



changed proposals for activities and any changed budget allocation to be confirmed and approved.

Meetings

The Project Coordination Committee shall convene with the representatives of the Parties and the Coordinator's representative as chairperson.

The Project Coordination Committee shall meet at least quarterly in principle at the request of its chairperson or at the request of a quarter ($\frac{1}{4}$) of the Parties.

Extraordinary meetings may be called at any other time at the request of its chairperson or at the request of a quarter ($\frac{1}{4}$) of the Parties.

Kick-off meeting

The first meeting of the Project Coordination Committee (Kick-off Meeting of the Project) will take place at the latest seven (7) days after the start of the Project. The structure of the Project must be confirmed by the Project Coordination Committee.

Costs – Payments

General Principles

Each Party shall bear its own costs incurred in connection with the performance of the Contract and this Consortium Agreement, carrying out of the Project work and implementation of the Project.

The financial contribution of the Commission will be distributed according to the Contract and the decisions of the Project Coordination Committee.

Financial planning and reporting data

The Parties shall deliver all relevant financial data including but not limited to the application of the budget use and received payments needed for financial planning, its execution and accountability towards the Project and towards the Commission, based upon their financial system as provided in the Contract and the Consortium Agreement.

Each Party shall be solely liable for its financial data. No other Party, including the Coordinator or their representatives acting within the scope of this Consortium Agreement may change these data without express written permission of the Party concerned.

Liabilities

Liabilities and Indemnification between the Parties shall be worked out during contract negotiations and formulated in the final Consortium Agreement.

Intellectual Property Rights

Confidential information provided by any of the partners may be used freely within the scope and for the duration of the project. All information provided to the Project Team (or part thereof) is assumed 'Commercial in Confidence' and the provider loses no rights through its disclosure to the Project Team.

Where it is felt necessary formally to establish Prior Art, a formal summary shall be prepared by the Partner concerned and circulated to all the Project Leaders and will be filed by the Project Coordinator for the duration of the project as amendment to the Consortium Agreement.

While IPR made during the course of the project will be the property of those Partners developing it, a formal summary shall be prepared by the Partner(s) and will be filed with the Coordinator for agreement and to make sure that any specification work carried out with a standardization objective (e.g. API, protocol definition, application profile) will be made available as an open specification.



Apart from knowledge the Phoenix project produces software components and procedures. These entities are not patentable in Europe.

Intellectual Property protection of the Phoenix deliverables is effectuated through:

- copyrights of the software and procedures;
- the complex subject matter and deep expert knowledge needed to effectively exploit this software;
- the 2 year head-start in building and deploying a comprehensive set of software components including the experiences and best practices derived from the demonstration projects.

We will however investigate the patentability of the combination of technologies into a comprehensive toolset because of its newness and uniqueness. We will also investigate the patentability in the USA.

During the Consortium Agreement negotiation phase, the Parties will work out the details of the Intellectual Property Rights issues like:

- ownership of knowledge;
- access-rights in general;
- access-rights to pre-existing know-how;
- access-rights to software;
- publication of results;
- use and dissemination after termination of the contract.

Management Of Knowledge

Coordinator ensures that:

- reports are sent to all Project Leaders;
- regular meetings are held to exchange knowledge and experiences;
- knowledge and expertise is sought both internally as externally to keep abreast of relevant new developments;
- competitive information and benchmarks will be identified and made available.

All documents, reports, minutes of meeting, specifications, source code and so on will be stored and made available to all project members via the Phoenix Intranet.



PART 2

PROOF OF CONCEPT



This page is intentionally left blank.



7 Proof Of Concept

7.1 Approach

The Phoenix project as outlined before, is a major endeavor. Although the potential benefits are substantial, the expected effort is substantial as well and the realization presents considerable risks.

The risks we are facing can be broadly classified in the following categories:

- Technology risks. As previously outlined, the technologies involved address theoretical models in the domains of complex optimizations, game theory and bargaining, and agent rational behavior. In addition these models have to be integrated into one comprehensive solution and build into a set of configurable software components. So a considerable amount of effort must be directed to research of the suitability of these models, the analysis of the behaviour patterns in relation to the target application domains and the computability or complexity issues in algorithm design.
- Organizational risks. The realization of the project requires the involvement of a wide variety of skill and disciplines. The consortium partners come from different organizational and cultural backgrounds. The effective cooperation and management of such an undertaking, especially with substantial research elements, is a complex affair in itself.
- Commercial risks. Will the technology - as envisioned and when properly functioning - have the business and societal impact as expected.

Before we can even start to convince partners, financiers and potential users that it is worthwhile to accept the organisational and commercial risks, we must be quite certain that the technology risks are under control i.e. we must be convinced that the concepts work and that identified technological issues can be resolved.

In order to accomplish a better view on the technology, we have chosen for a prototype approach. This approach is particularly useful to reach the following objectives:

- to explore the various models and implementations, ascertain the suitability if these models by analyzing test results - if possible against known benchmarks - and compose a technology road map;
- get a realistic view on the feasibility and issues of the project and determine solutions or mitigating measures;
- present a "proof-of-concept" to partners, financiers and clients about the viability of the concept and potential effects on the application domain;
- present a realistic demonstration model and impact analysis to convince interested parties.

7.2 Prototype

In the context of this thesis it is not feasible to realize a complete functional prototype of all three main technology areas. Therefore we started to work on the Heuristics parts, as this is a prerequisite for the bargaining part and agent platform. The following approach has been chosen:



- describe a representative – but not overly complex – business case, that serves as a virtual client for system development and demonstration model. The business case in the logistic service business has been selected;
- to provide a proper context for software engineering, we design an outline of the functional model – comprising the major functional components – and overall architecture to serve as a road map for the prototype development process;
- in line with the nature of the business case and the rationale discussed in the next paragraph 7.4 – Prototype Rationale, optimization of the “Capacitated Vehicle Routing Problem” (CapVRP) has been selected to solve first;
- design and build the CapVRP optimizer;
- analyze the capVRP’s converging behavior in order to build the probability performance model, using suitable feature monitors.

Each of these steps are further elaborated in the following chapters.

7.3 Business Case

A proof-of-concept requires a realistic representation of the concept, against which the results of the prototypes can be evaluated. For this purpose a business case has been defined to serve as a virtual client. The business case represents all major characteristics of the business problem we discussed in part 1: Introduction. For convenience these characteristics are listed here again:

- the company operates in a value chain or network, where each company has a complex planning and optimization problem e.g. in manufacturing or logistics;
- the companies have “general-purpose” production resources – “job-shop” type of process technology or fleet of vehicles – so there is potential for pooling resources;
- the companies may have different cost structures, so there is also room for optimization of the order lists, giving additional room for cost saving;
- there is a strong response time constraint;
- there is a potential for increased value creation. Utility can be expressed in cost saving, service degree improvement (delivery time), more business, and so on.

Partners are independent entities, so for each planning instance there is a decision problem whether pooling is profitable (i.e. has a higher value than in case each company would solve its problem individually) and if so, how the surplus value is shared between the partners.

Logistic Service

Transportation companies Jansen BV and Pietersen BV are medium sized transportation services operating throughout the Netherlands. Both companies own their fleet of delivery vehicles operating from a dispatch center in Rotterdam. There are many transportation companies like Jansen and Pietersen, working from their own dispatch center and competing in overlapping service areas. As the delivery service can be regarded as a commodity, Jansen and Pietersen compete on price and delivery time. For services like Jansen and Pietersen, price and delivery time are both determined by total transportation mileage.

Thus minimizing total mileage will result in lowest cost and best service time: resulting in a better competitive position in the market.



As these companies operate in a commodity market, competitors are looking for cost reduction as well. There must be opportunities for cost reduction if one could choose to cooperate with another transport firm in an overlapping service area. The potential for saving comes from the fact that particular deliveries could better be made with less driving by the other firm due to adjacency and vice-versa. Consequently in some instances it would be profitable to pool deliveries and fleet with a competitor, while in other instances it would not.

The decision whether Jansen and Pietersen would actually coordinate their deliveries is determined by the cost associated with the different solutions. They must negotiate about taking care of their own deliveries or pooling in order to reduce cost. They also must bargain about how they will split the cost and benefits of the joint solution if they decide to carry it out. Before they can decide to compete or cooperate, however, they must have solutions (i.e. possible routes) for the three problems: each individual route and the joint one.

In determining the routes the companies have to honor typical constraints like:

- each vehicle has a maximum load weight constraint. These may differ between vehicles;
- each vehicle has a maximum load volume constraint. These may differ between vehicles;
- each vehicle has a maximum route length, prescribed by law;
- each delivery has to be included in the route of some vehicle;
- since service time is of essence, the time available for negotiating is limited.

The solution will be implemented through a multi-agent system in which each agent represents a transportation firm, performs the deliberation and proposes a decision to its 'principal'. In order to determine the cost saving by pooling instead of each agent operating individually, the agents need to compute solutions for both agent's individual problem as well as the joint problem. The problem is to determine the shortest total distance the vehicles must drive to perform the scheduled deliveries. This problem is known as the Capacitated Vehicle Routing Problem (CapVRP) a variation of the general Vehicle Routing Problem (VRP), a problem that can not be solved deterministically for non-trivial instances. To solve the CapVRP optimization an any-time approximation algorithm will be used, that will provide the best solution within the allocated time along with an estimate of the solution quality and quality improvement per additional time unit. Since the agents only have limited time, with this algorithm they can make choices on what problem to spend calculation time in order to determine their bargaining strategy.

If the calculated value of the pooling solution is higher than the sum of the values of the individual solutions, then there is a *potential* gain from agreeing to pool. Actual agreement will be reached only when agents also agree on the division of the surplus. The non-cooperative¹ bargaining process is modeled according to concepts from game theory using a Nash or Bayesian equilibrium.

7.4 Rationale

Delivering the proof-of-concept consists of evaluating the prototype against a realistic set of planning instances based on the business case. The proof is delivered when each partner has increased his profit over a certain period of time by accepting the prototypes decision

¹ A *non-cooperative* bargaining situation exists when both – rational – agents pursue to maximize their own individual utility. *Cooperative* bargaining on the other hand exists in a two agent situation, when agents agree on a common strategy in pursuing their individual goals. In a more than two agent setting cooperative bargaining refers to the possibility of coalitions between bargainers.



about pooling and profit sharing, as opposed to continue solving each planning instance individually.

Each partner's profit is a function of:

1. the efficiency gains by using advanced optimizations;
2. the efficiency gains by plan and resource pooling as opposed to the each one individually;
3. the bargaining results about sharing, in case pooling is profitable.

Proof of the first two points can be delivered by building a prototype for the CapVRP problem alone, even without the anytime features needed for deliberation control and bargaining. Obviously, the proof of the 3rd point is only relevant if proof for points 1 and 2 has been delivered.

Consequently realization of the CapVRP optimizer is essential in delivering a crucial part of the proof-of-concept.



8 Prototype Architecture

8.1 Model

In this chapter the bargaining model is being developed in functional terms. The resulting functional decomposition will serve as the basis for the solution architecture and implementation in an multi-agent model.

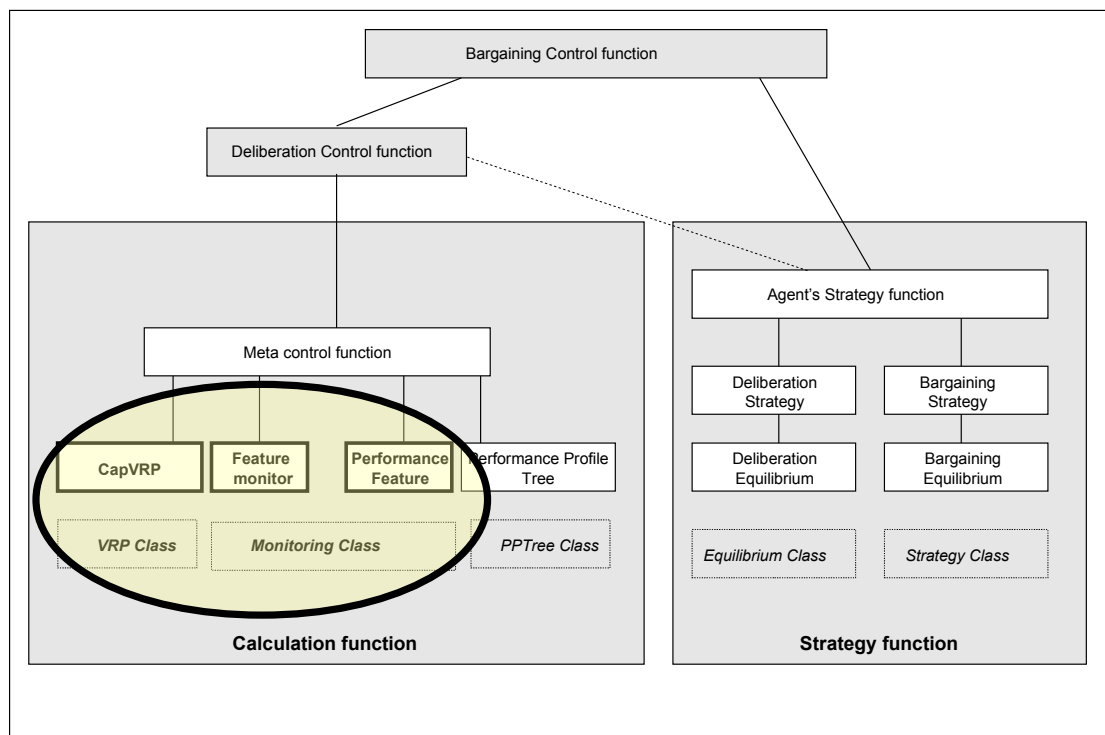


Figure 8.1.1 – Prototype positioned in overall architecture

As discussed in Chapter 7 Proof of Concept, the prototype focuses on an implementation of an optimizer for the Vehicle Routing Problem.

8.2 Vehicle Routing Heuristics

The *Vehicle Routing Problem* (VRP) is a complex combinatorial optimization problem, in essence a combination of the *Traveling Salesman Problem* (TSP) and the *Bin Packing Problem* (BPP) both in NP-hard time complexity class. There exist many variants to the VRP such as Single and Multiple Depots, VRP with Time Windows and the Capacitated VRP. Capacitated VRP (CapVRP) is a VRP in which a fixed fleet of delivery vehicles of uniform capacity must service known customer demands for a single commodity from a common depot at minimum transit cost. CapVRP is like VRP with the additional constraint that every vehicle must have uniform capacity of a single commodity. Figure 8.2.1 illustrates a Single Depot VRP.

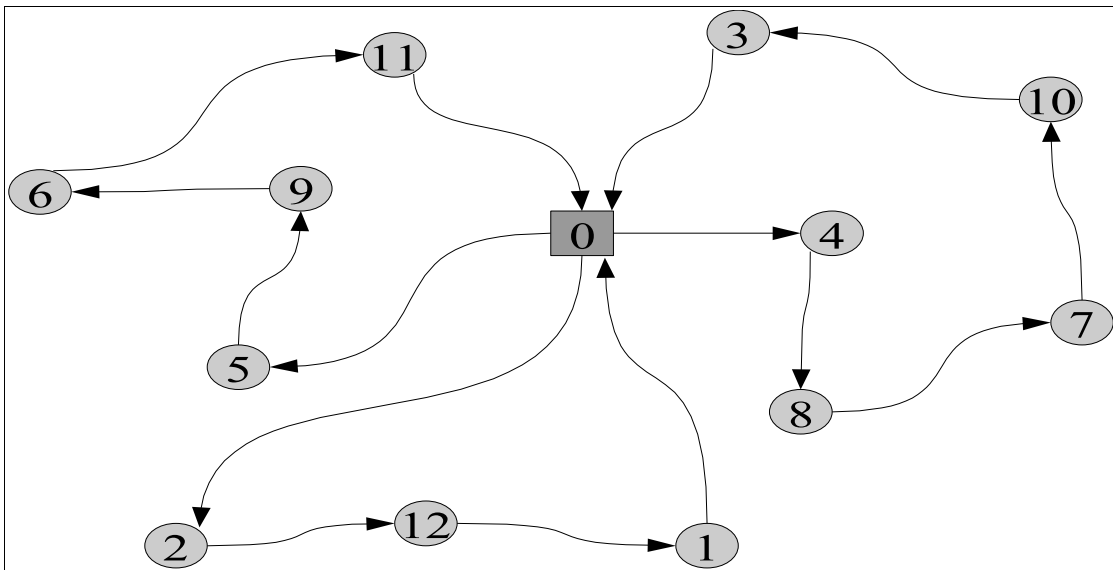


Figure 8.2.1 – Single Depot Vehicle Routing Problem

Solution techniques for the VRP exist in the following categories:

1. Exact approaches. Algorithms of this category computes *every possible solution* in the solution space to find the best one. An example is the *Branch and Bound* algorithm, but there are many others. As VRP is in NP-hard these methods are not suitable for non-trivial instances.
2. Classic Heuristics. These heuristic methods perform a *relatively limited exploration* of the solution space. They typically produce good quality solutions in limited computing time. *Constructive methods* gradually build a feasible solution based on some utility function, but don't perform improvement phases. *Phased methods* however first perform a clustering of vertices in feasible routes and then execute a route construction phase with feedback loops. Examples of the phased methods are: Fisher & Jaikumar and Taillard.
3. Meta-heuristics. The emphasis in this class of algorithms is on a *deep exploration of the most promising regions* of the solution space. The quality of the solutions are often much higher than with classic heuristics. Examples are: Ant Colony, Simulated Annealing, Tabu Search and Genetic Algorithms.

As discussed in paragraph 4.2 – Optimization, the optimizer must possess the following additional properties:

- anytime i.e. the algorithm that can be stopped “anytime” during the optimization process and is capable of delivering the best result reached so far;
- must be capable of assessing the quality of the solution it presents;
- must be able to make an estimate of solution quality to be reached given additional computing time;
- must be able to create and maintain a Performance profile tree [3],[4] for that purpose.



The Evolutionary approach was chosen for the following reasons:

- the concept of generations seems natural for anytime algorithms as after each generation the best (= fittest) result is known;
- computing resources is easily conceived as computation steps, each comprising a predefined number of generations;
- methods for monitoring asymptotic behavior are fairly easy to be implemented using the generation concept. As best solutions are preserved across generations the solution value's growth is monotonic, thus providing means for solution quality estimation and prediction.

Genetic algorithm

Genetic algorithms follow the mechanisms of natural genetics. It maintains a population of individuals (*genotypes* or *chromosomes*) by producing offspring through the application of genetic operators like mutation and crossover, resulting in a new generation. By encoding problem specific features into the genotype, the population is transferred into a – subset of the – solution space.

Genetic algorithms are a class of probabilistic algorithms, where the population stands for the set of potential solutions and in each generation the “fittest” solutions reproduce and the “unfittest” die. Mutations arbitrarily alter one or more *genes* (features of the chromosome) of a selected individual, whereas crossovers exchange genes between different chromosomes. The intuition behind these operators are to exchange information between chromosomes and introduce variability in the population in order to preserve a certain degree of genetic diversity, needed for a continued evolution towards more “fitter” individuals. As fitness is a function of solution quality, the most fittest individual in the population represents the “best” solution. A genetic algorithm for a particular problem typically has the following components[19]:

- a genetic representation for potential solutions of the problem;
- a way to create an initial population of potential solutions;
- an evaluation function that plays a role in the environment, rating solutions in terms of their “fitness”;
- genetic operators that alter the composition of offspring;
- values for various parameters needed for the genetic algorithm like population size, probabilities of applying genetic operators, and so on.

The following paragraphs define the algorithm and monitoring methods. The monitoring methods are needed in order to estimate the solution quality reached at arbitrary points in time. These estimations will be used to build probability models to predict solution quality given extra time. For this purpose, the algorithm should have a calibration mode.

Determining solution quality during computation is needed in order to rationally allocate computation resources to computationally bounded agents in a non-cooperative bargaining setting. CapVRP is NP-hard. Consequently any agent's rationality is bounded by computational complexity. Two evaluation methods will be evaluated: increment ratio and sampling (Monte Carlo method). These methods are correlated with the approximation ratio as an objective measure of the solution quality.

The approximation ratio will be calculated after the algorithm detects no more change in both best and worst solution utility for a predefined number of generations. By also including the worst solution utility, the utility range is known, thus providing a sound basis for calculation of the approximation ratio. In order to generate the data for analysis the CapVRP must be run on a set of instances. Per time step – comprising a predefined number of generations – current utility, increment and sample counts must be determined



and recorded. After determining the utility range, the approximation ratio – solution quality – per time step is calculated retroactively. As the genetic implementation of CapVRP is non-deterministic, each problem instance in the set will be run a number of times in order to analyze the algorithm's stochastic behavior (convergence, global/local maximum and average solution).

In addition benchmarks will be used to verify the algorithm's performance against published instances' best solutions.

8.3 Multi-Depot CapVRP

Thus far we have discussed the single-depot CapVRP problem to be solved for an agent when optimizing its own schedule. In order to make proper deliberations, the agent must also optimize the joint problem. For that purpose two distinct instances of the CapVRP problem must be merged, typically resulting in a multi-depot CapVRP (MCapVRP) problem. As we intend to use the same mechanism, we have to devise a worst-case polynomial time reduction:

$$CapVRP \leq_p MCapVRP$$

For this purpose the following steps have to be performed to prove the validity of such a reduction:

- Devise reduction function $T : MCapVRP \rightarrow CapVRP$;
- Prove $T(\text{yes-instance } MCapVRP) \Rightarrow \text{yes-instance } CapVRP \wedge$;
 $T^{-1}(\text{yes-instance } CapVRP) \Rightarrow \text{yes-instance } MCapVRP$
- Prove $T \in P$.

This problem is eligible for further research and has not been addressed in this thesis. For pragmatic reasons we will consider the joint problem as a single depot VRP for the proof of concept.

8.4 CapVRP Problem Definition

In the following paragraphs the formal description of the Capacitated Vehicle Routing Problem (capVRP), including its genetic – chromosome – representation and applicable genetic operators over sets of chromosomes, are developed.

Instance

A graph $G = (V, E)$, a depot $d \in V$, distance w_e for all $e \in E$, $V = V \setminus \{d\}$, a set of resources $C = \{ \langle \text{vehicle}_0, \text{capacity}_0 \rangle, \dots, \langle \text{vehicle}_n, \text{capacity}_n \rangle \}$, a set of customer demands $O = \{ \langle v_0, \text{volume}_0 \rangle, \dots, \langle v_m, \text{volume}_m \rangle \}$, where $v \in V$ and $\text{volume} \in C$

Solution

A partition of O as a set of tours $T = \{ \langle O'_0, c_0 \rangle, \dots, \langle O'_k, c_k \rangle \} \mid k \leq |C|$, where $O'_i \subseteq O$ and $\sum_{i=0}^k O'_{k,i}(\text{volume}) \leq c_k(\text{capacity})$ and $\bigcap_{i=0}^k O'_i(v) = O(v)$

A *tour* is a *Hamiltonian cycle*, a path through a graph that starts and end at the same vertex and visits every other vertex exactly once.



Solution Space

The solution space S is the set of all possible solutions of the CapVRP instance:

$$S = \{T_0, \dots, T_q\}$$

The lower-bound VRP solution space size is where one vehicle can serve all customers. In that case VRP is *mapping reducible* to TSP: $VRP \leq_m TSP$, where the *reduction function* f is the *identity* function. The cardinality $|S|$ of TSP solution space is $|O|!/2$ for symmetric instances. Consequently for VRP the cardinality of the solution space $|S| \geq |O|!/2$, thus intractable for non-trivial instances.

Measure

$$\text{Utility } u = (-1)^* \sum_{i=0}^{|T|-1} \sum_{j=0}^{|O'_i|-1} w_e(v_j, v_{j+1}) \text{ where } v_j, v_{j+1} \in O'_i \cup \{d\}$$

The utility as defined here is also used as fitness function for the genetic algorithm. We will use both terms interchangeably.

Target

$$\max u \quad \text{Utility range} = [\min u, \max u]$$

8.5 Genetic Representation

Chromosome

The genetic representation of a solution of a CapVRP instance is represented in a chromosome "Genetic VRP Representation" (GVR). We will use a simplified form of T as chromosome, clearly subject to the same constraints.

$$GVR = \{O'_0, \dots, O'_k\}$$

Population

Ordered set of CapVRP candidate solutions represented as genetic chromosomes (GVR). Ordering is dependent on the chosen selection strategy for the parent population.

Algorithm:

```

input instance of CapVRP, timestep t = {gen0, ..., genn}, |R| = m
begin
  generate population R
  while ( ¬(max uupper and min ulower reached) )
    for (n generations)
      if ( ¬ max uupper reached )
        generate Rn-1 → Rn
        if ( ubest > umax ) umax = ubest endif
        monitor incrementFeature
        monitor sampleFeature
        determine max uupper reached
      endif
    if ( ¬ min ulower reached )

```



```
generate  $R_{n-1} \rightarrow R_n$ 
  if (  $u_{worst} < u_{min}$  )  $u_{min} = u_{worst}$ ; endif
  determine min  $u_{lower}$  reached
endif
endfor
write  $\langle ir, \langle count_{low}, count_{high} \rangle, u_{max}, u_{min} \rangle$ 
endwhile
output solution
end
```

Generate Resultset

Randomly generate m candidate solutions GVR_m and ascertain $GVR_m \in S$ according the following algorithm.

```
input CapVRP instance,  $m$ 
 $m = 0$ 
initialize  $tour(m)$ 
while (not ( $O(v)=\{\}$ )) do
  extract random  $v$  from  $O(v)$ 
  if (weight ( $tour(m) + v$ )  $\leq$  capacity vehicle( $m$ ))
    append  $v$  to  $tour(m)$ 
  else
     $m++$ 
    initialize new  $tour(m)$ 
    add  $v$  to  $tour(m)$ 
  endif
endwhile
output { $tour(0), \dots, tour(m)$ }
end
```

Populate resultset in order of $u(GVR)$.

```
input { $tour(0), \dots, tour(m)$ }, depot  $d$ 
for ( $i=0, m, i++$ ) do
  concatenate  $p + tour(i)$ 
  calculate utility( $i$ ) =  $(-1) * \text{length}(p + tour(i))$ 
endfor
output { $tour(0), \dots, tour(m)$ } ordered by utility()
end
```

Selection Operators

These operators act on the individual solution in the current population. Selection is the main filter by which the genetic algorithm determines the composition of the next generation. It is very important that the population maintains diversity over the generations in order to avoid getting trapped in a local optimum. Generally two selection strategies can be identified: deterministic and stochastic. Given the same population the deterministic strategy will always select the same individuals – generally *greedy* i.e. with the highest fitness scores – whereas the stochastic strategy operates based on some probability mass function.

The parent selection operator determines which individual will be selected as parent for the next generation. The replacement operator determines, which individuals of the current population and offspring will survive into the next population.



Parent selection operator

As we are running the CapVRP under real-time constraint – anytime algorithm – we will implement a choice of selections operators: *deterministic*, *mixed stochastic*, and *single tournament*.

For the *deterministic* selection strategy we will implement a *greedy* approach: selection of a number of the best solutions in the population to serve as parents. This strategy converges fast and will most likely produce usable solutions in a short time. The chance, however, that the algorithm converges to a local sub-optimum is greater than when using some *stochastic* strategy [6] pages 179-185.

From the ordered population R we select a subset of P best members from the top of R , to serve as parents to generate $|P|$ descendants.

As the greedy approach may effectively reduce diversity, the *mixed selection operator* will select a fraction of the parent set members stochastically from the population in addition to the greedy part. The number of stochastically chosen members can be set by a parameter. From the ordered population R we select a subset of $G = P - S$ best members from the top of R , and S members stochastically from $R - G$, to serve as parents to generate $|P|$ descendants.

The *single tournament* operator, however, takes a total different approach. It shuffles the population randomly and then divides it in small groups (tournaments). The two most fit members in each tournament are chosen to be parents and produce offspring through mutual cross-over and mutations. The offspring take the place of the two least fit in the tournament. A new generation has evolved when all tournaments have produced offspring. The two advantages are that fitness can not decline and that diversity in the population is maintained.

Replacement operator

For reasons mentioned above, we will apply an *elitist* strategy in order to ensure the best solution(s) in the population will survive. Consequently the fitness of the best solution never deteriorates – thus monotonic growth – and convergence is generally faster than when no elitism is used [6] page 181.

The replacement related to the tournament operator is also elitist, however the two least fit per group are removed as described above.

The chosen variation procedure will add *new* offspring of the parents in P to the population and discard $|P|$ lesser fit solutions.

Variation Operators

We consider two types of genetic operators: crossover and mutation. They must deal with two levels in the representation: 1) change the delivery order and 2) change the allocation of demands to vehicles. The last one can not only switch customers from one tour to another, but also modify the number of vehicles belonging to a solution, i.e. add or remove tours. The genetic representation, crossover and mutation operators used here are based on the work of Pereira et al [1, 2].

Crossover operator

This operator does not effect a mutual exchange of genetic material between both parents, but one individual receives a fragment of genetic material – a sub-route within a tour – from another parent and inserts it in one of its own tours. The donor parent remains unchanged. The geographical proximity of the first customer location in the fragment is used to determine the insert point within the receiving parent. By following this procedure, duplicate customers may be introduced. In order to obtain a legal offspring, duplicates



must be removed and possible excess capacity must be corrected by splitting the particular tour in two.

```
for each individual parent  $I_1$  in  $P$  do
  if (  $\neg$ tournament )
    select randomly another parent  $I_2$  from  $P$ 
  else
    select other parent  $I_2$  in the tournament
  endif
  select randomly from the genetic material of  $I_2$  a subroute  $SR=\{v_1, v_2, \dots, v_n\}$ 
  select customer  $v \notin SR$  geographically closest to  $v_1$ 
  insert  $SR$  into  $I_1$  such that  $v_1$  is placed directly after  $v$ 
  if ( excess weight ) split route endif
  remove from  $I_1$  all  $v \notin SR$  duplicated by the insertion, obtaining descendant  $I_d$ 
endfor
```

The descendant obtained from a crossover operation can be subject to mutation operation.

Mutation operator

Four operators are identified, based on proposals usually applied to order-based representations:

- Swap: selects two customers and swap them. The customers are members of the same tour.
- Inversion: selects a sub-route in a tour and reverses the visiting order of the customers.
- Insertion: select randomly a customer from a random tour in the solution and moves it to a random place in a different randomly selected tour. If receiving tour exceeds capacity by the insertion, the tour is split. Instead of random insertion a new tour is created with probability $1/(2V)$ where V is the number of vehicles in the current solution.
- Displacement: selects randomly a sub-route of a tour and inserts it in another tour. The insertion can be performed in the same or other tour. Like the insertion operator, this operator can also create a new tour with the sub-route; the heuristic for the probability is the same.

Swap and inversion does not change the number of tours, insertion and displacement do.

Parameters

Genetic algorithms generally present more parameters to control than other heuristics. Representation, variation operators, selection operators, population size and so on all require choices that will – sometimes seriously – impact the performance of the algorithm for particular problem instances. Parameter tuning is a very important issue¹.

For the purpose of this study, the settings and probabilities applied by Pereira et al in

¹ Subject for further research. Monitoring solution features as predictor of solution quality is an important technique in anytime algorithms. An accurate determination of the solution quality is an important prerequisite for determining negotiation strategies.

Monitoring solution features, especially the ones we will research here, might also provide useful runtime information on the performance of the algorithm on the current problem instance. In a further study we might go deeper into the issue whether we could use these features also for self-adaptive parameter control, possibly by encoding the parameter structures into the genetic material. This might result in a more predictable asymptotic behavior of the algorithm over classes of problems, thus resulting in more accurate estimation of solution quality.



their study [9] [10] will be used as starting points:

- number of generations: 50.000;
- deterministic, mixed or tournament strategy for parent selection. For the mixed strategy a fraction of 0,8 of parent set will be selected greedily and 0,2 stochastically;
- elitist replacement strategy;
- population size: 200; parent set P : 5;
- crossover rate: 0,75;
- mutation rates: swap: 0,05, inversion 0,15, insertion 0,05, displacement 0,2.

Monitoring Solution Quality

Increment ratio

The monitored feature is the average growth ratio of the utility over the generations comprising a time step t .

$$ir = \left[\sum_{i=0}^n (u_{offspring,i} - u_{offspring,i-1}) \mid u_{offspring,i} < u_{offspring,i-1} \right] * \frac{1}{n}$$

The growth ratio has an analogue to the derivative of a function in mathematics. Based on the presumption that the utilities in the solution space are evenly distributed, the steepness of the ratio is related to the position in the solution space, with zero being (sub) optimum.

Sample ratio

The monitored feature chosen here consists of the utility counts of all offspring per generation. Each offspring is counted as low or high respectively depending on its utility being equal, lower, or higher as the starting utility of the current time step.

$$sr = \langle count_{low}, count_{high} \rangle \text{ where}$$

$$count_{low,i} = count_{low,i-1} + 1 \mid u_{offspring,i} \leq u_0$$

$$count_{high,i} = count_{high,i-1} + 1 \mid u_{offspring,i} > u_0$$

The sample ratio has an analogue to sampling a set of items with two distinct properties. Take a large set of red and white marbles. Drawing a sample of sufficient size from this set, the ratio red to white has a statistical correlation to the distribution of red and white marbles in the total set.

Again under the assumption of equally distributed utility in the solution space, a solution with a utility greater than the current one is labeled white and the ones with utility less are labeled red. As the variation process is a random operation, the ratio white to red measured over a sufficient number of generations must be correlated to the ratio between the number of solutions with better utility and the one with worse utility in the solution space. Hence a measure for solution quality.

In this thesis we will not verify the assumption of an *equidistributed utility* solution space, as the monitoring functions are not part of the proof of concept.



This page is intentionally left blank.



9 CapVRP Implementation

This chapter describes the design and realization of the prototype software.

9.1 Requirements

As argued in Chapter 7 – Proof of Concept, the software development will be restricted to the solution of the CapVRP problem by means a a genetic algorithm, according to the specifications developed in Chapter 8 – Prototype Architecture.

The solution must have the following functional characteristics:

- evolutionary algorithm, based on an implementation of the described genetic – chromosome – representation;
- optimizing and calibrating modes (calibrating mode, however, is out of prototype scope);
- progress logging for convergence analysis;
- performance feature monitoring for solution quality estimation, out of prototype scope;
- performance profile tree maintenance, out of prototype scope.

The prototype developed will be used for an essential first phase in the proof-of-concept, namely to demonstrate the efficiency gains resulting from optimizing before and after pooling.

The envisioned service architecture as presented in paragraph 4.3 – Service Platform Architecture, however, imposes additional non-functional requirements with considerable design implications. These requirements comprise:

- the optimizer must be runnable as a service node within an agent platform. This means that a clear *application program interface (API)* must be designed for communicating instance descriptions, optimizer control options, runtime parameters, and returning optimization results. The API must encapsulate the optimizer complexity in a CapVRPEngine instance, that can be invoked by any authorized agent;
- the engine must be able to handle concurrent service requests from multiple agents, thus having some form of process thread control;
- the optimizer must be able to handle a considerable number of optimizer features and parameters. This implies the optimizer must have the structure of a *framework* in which it must be able for features to be plugged in and invoked at runtime;
- the engine must be scalable in order to service a varying number of requests within strict performance criteria.

The design presented in the following paragraphs has been performed with these requirements in mind and follows the *object-oriented* design paradigm.

9.2 Structural Model

The structural model results from translation of the functional and non-functional



requirements into a class model defining the way data and function (methods) are encapsulated. The following table provides the mapping of the CapVRP problem description and its genetic representation into the CapVRP class model.

Table 9.2.1 – Mapping requirements to classes

Element of Problem Definition or Genetic Representation	Classes
Instance	<p>The CapVRP instance consists of a collection of sets. These sets are modeled in container classes and element (data) classes. The container classes all inherit from the Container class template, implementing all methods to navigate and manipulate its elements efficiently. The element classes all inherit from a BasicNode class template. Both Container and Node templates enforce required interface requirements. The template concept is implemented as C++ abstract classes.</p> <p>Container<T>¹ template from which all container classes inherit BasicNodeContainer<T> specialization of container class for nodes Locations the collection of all location nodes (set V, $v_o = d$), Edges the collection of all edges (set E) Customers the collection of customer demands (set O) Resources the collection of vehicles (set C)</p> <p>BasicNode<T> template for all nodes Location location node ($v \in V$) EuclidianLocation specialization of location with Point(x,y) in euclidean plane Edge edge node ($e \in E$) Customer customer node ($o \in O$) Resource vehicle node ($c \in C$)</p> <p>CapVRPInstanceBuilder class responsible for converting the instance definition (xml) into container and node objects VRPInstance template class for all VRP instances CapVRPInstance class responsible for lifecycle management of containers and objects. Inherits from VRPInstance and is a specialization the Capacitated VRP variant.</p>
Solution	<p>Solution specialization of Chromosome, collection of vehicleRoutes (<i>tours</i>) as valid solution VehicleRoute <i>tour</i></p>
Solution Space	See Population
Measure	<p>UtilityCalculator template class for all utility calculators TotalDistanceCalculator distance calculator, inherits from UtilityCalculator</p>
Target	Not Applicable
Chromosome	Chromosome collection of vehicleRoutes as genetic representation of a solution instance of VRP. Inherits from Container.
Population	<p>Population container of valid solutions eligible for genetic manipulation. Inherits from Container. PopulationCreationOperator template class for all population creation operators SinglDepotSingleCapacityPopulation creates a population with said characteristics. Inherits from PopulationCreationOperator.</p>

¹ The notation *class<T>* stands for the C++ *template* concept. In C++ the template can be instantiated with any *type* resulting in a class specialized for that specific type. For container classes like *vector*, *list*, *map* and so on type may also be *pointers* to objects of that type.



Element of Problem Definition or Genetic Representation	Classes
Selection operators	<p>ParentSelectionOperator template class for all parent selection operators</p> <p>DeterministicSelectionOperator implements the <i>greedy</i> selection strategy. Inherits from ParentSelectionOperator.</p> <p>MixedSelectionOperator implements the mixed greedy and stochastic selection strategy. Inherits from ParentSelectionOperator.</p> <p>TournamentSelectionOperator implements the tournament selection strategy. Inherits from ParentSelectionOperator.</p> <p>ReplacementOperator template class for replacement operators.</p> <p>ElitistReplacementOperator implements the elitist replacement strategy. Inherits from ReplacementOperator.</p>
Variation operators	<p>VariationOperator template class for all variation operators</p> <p>CrossoverOperator implements the <i>crossover</i> mutation. Inherits from VariationOperator.</p> <p>MutationOperator implements the <i>Swap, Inversion, Insertion and Displacement</i> mutations. Inherits from VariationOperator.</p>
Parameters	<p>CapVRPInstanceBuilder class responsible for converting the instance definition (xml) into parameter settings</p> <p>CapVRPPParameter class containing all optimizer parameters.</p>
Solution Quality	<p>Monitor template class for all solution quality monitors.</p> <p>IncrementMonitor implements the <i>increment ratio</i> method. Inherits from Monitor.</p> <p>SampleMonitor implements the <i>sample ratio</i> (Monte Carlo) method. Inherits from Monitor.</p>

In order to create a structure capable of implementing the non-foundational requirements stated above, the class model contains a number of template classes. These classes are defined such that they impose strict compliance to interface requirements, essential for realizing an optimizer *framework* with pluggable extensions.

The instance classes all derive from the BasicNodeContainer template, that derives from the Container template which in turn derives from a language specific implementation of the *doubly-linked list* data structure. As all operator classes – and especially the *variation operators* – operate intensely on these data structures to effect the genetic transformations, *sequence* data structures with optimized behavior for *insertion* and *deletion* of elements are essential. The typical *time complexity* for these operations in a *doubly-linked list* is $O(1)$.

The CapVRPInstance class derives from the VRPInstance template enforcing a strict compliance to the required data structures expected by the Optimizer class. Instantiation of this class can be performed by any process implementing the template and delivering the object to the Optimizer. In the prototype this is done by the CapVRPInstanceBuilder class, but could also be an agent delivering a serialized instance object to the CapVRPEngine.

All operator classes derive from an appropriate template class for the reasons argued above.

Figures 9.2.1 and 9.2.2 show the overall class structure and a detailed view on the container and node classes. The complete API documentation and source code is available on the companion CD-Rom (Appendix-B).

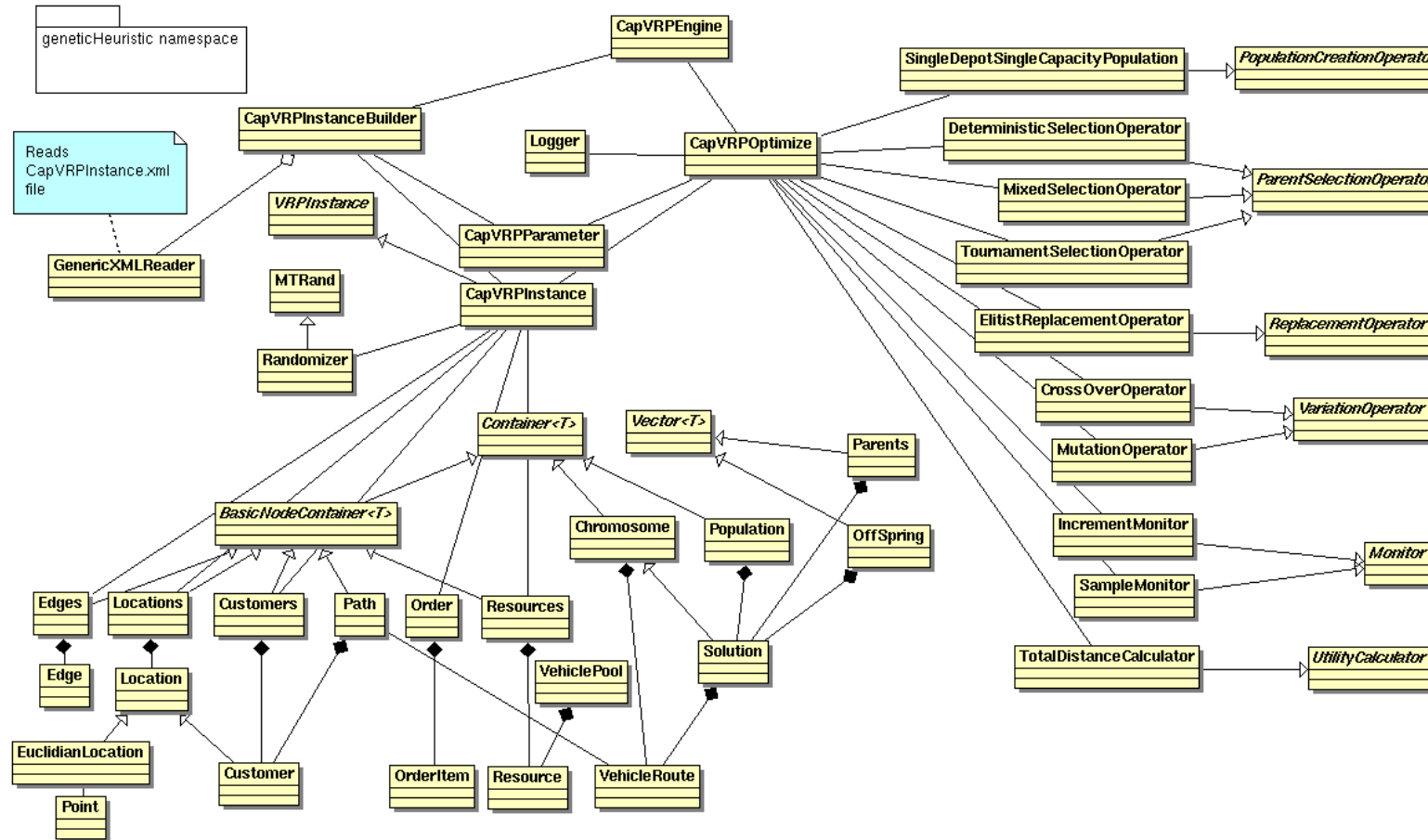


Figure 9.2.1 – CapVRP Class Model

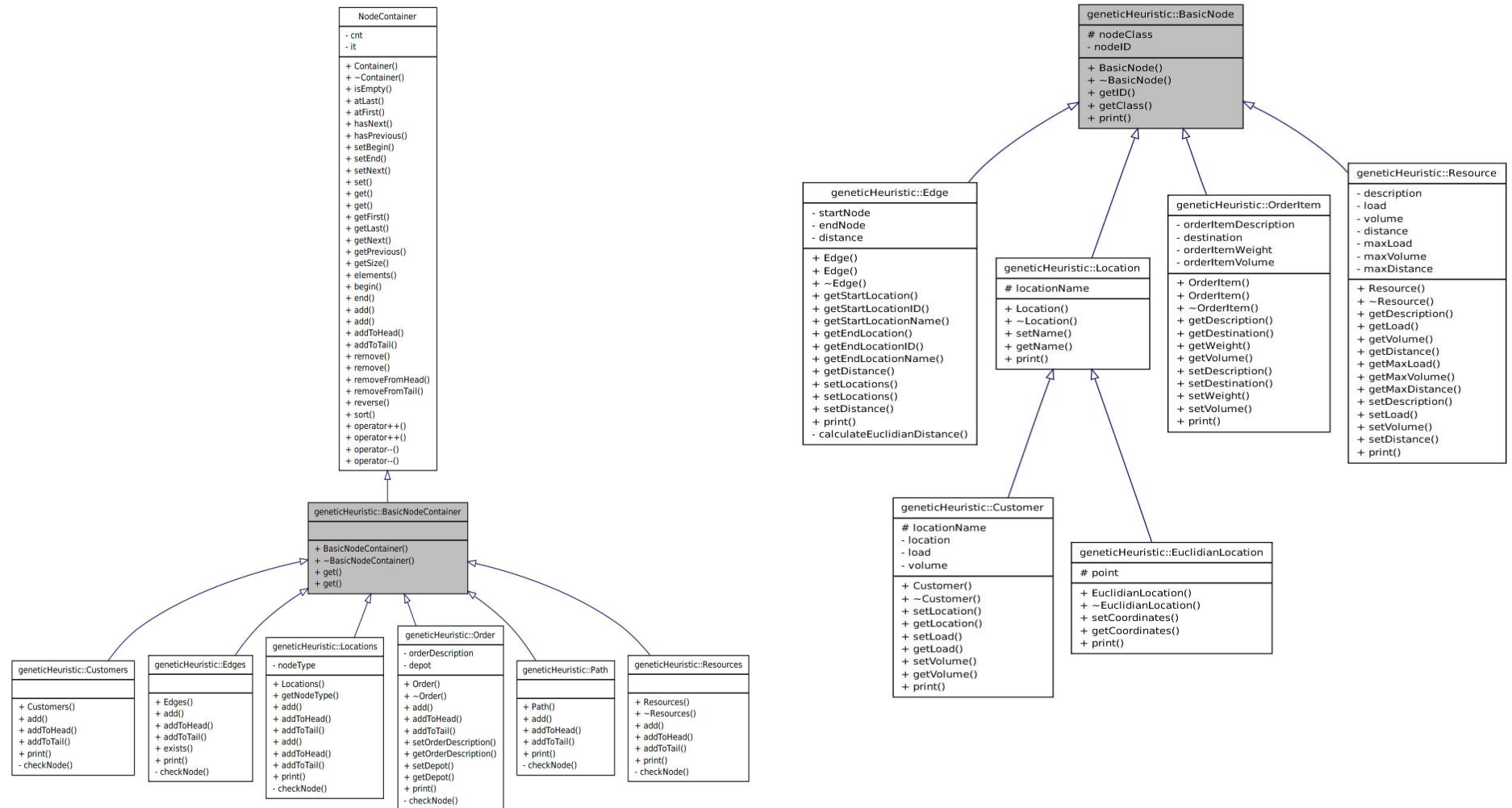


Figure 9.2.2 – Detail class models of NodeContainer and Node Classes



9.3 Behavioral Model

The optimizer functionality is encapsulated in a `capVRPEngine` that exposes a very concise API. The engine object accepts a `capVRPInstance` object, a `capVRPParameter` object, and in case the instance has to be build a `capVRPInstanceBuilder` object. It exposes a `optimize` method, that returns the best solution.

The optimization process is performed in a number of cycles (steps) set by an instance parameter. The optimization starts with the creation of a population, the size of which is set by an instance parameter.

A step consists of selecting a set of parents from the population using a particular parent selection operator, and applying the variations operators on the members of that set to generate offspring. The offspring is replaced into to population according the selected replacement strategy. The steps completes by sorting the population on utility such, that the best solution is at head and the worst at tail.

When the all steps have been processed, the best solution is returned to the calling object.

Figure 9.3.1 shows the overall sequence diagram.

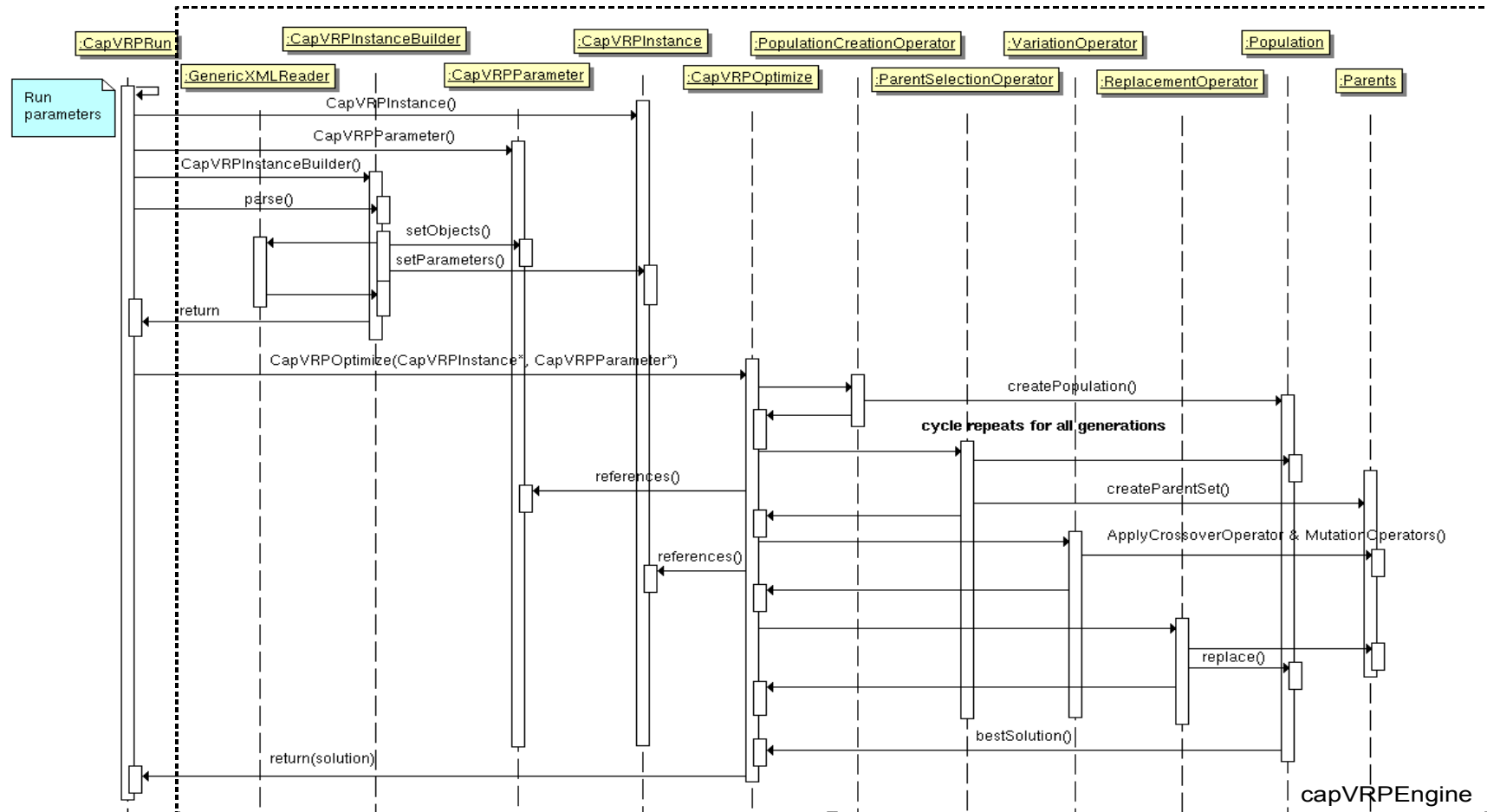


Figure 9.3.1 – CapVRP Sequence Diagram



9.4 Software Architecture

The prototype software has been fully developed in the object-oriented language C++. Reasons for this are:

- portable to many platforms through ISO/IEC standardization;
- object-oriented language with strong typing, virtual functions, abstract classes (comparable with Java Templates), and multiple inheritance;
- standard template library (STL) with powerful type independent classes for handling data structures;
- many special purpose libraries available;
- high performance.

The prototype includes two software libraries:

- Xerces-C++, a validating XML parser written in a portable subset of C++. Xerces-C++ makes it easy to give an application the ability to read and write XML data. A shared library is provided for parsing, generating, manipulating, and validating XML documents. Xerces-C++ implements the XML 1.0 and 1.1 recommendations and many associated standards. Xerces-C++ is an open source project of the [Apache Software Foundation](#);
- [Jasper Bedaux's C++ port](#) of the [Mersenne Twister \(MT\)](#) random number generator. A genetic approach depends heavily on true random selection. A high quality random number generator is therefor essential. The MT generator has a far longer period and far higher order of equidistribution than any other implemented generators. It is proved that the period is $2^{19937}-1$, and 623-dimensional equidistribution property is assured. The Randomizer class inherits from MT.

The software was developed using the superior open source [Netbeans Integrated Development Environment](#), supported by the following open source systems:

- C++ Netbeans plugin;
- GNU/GCC compiler collection for C/C++
- GDB debugger;
- Make build system, including libtool, automake and friends;
- Subversion (SVN) version management system ;

The CapVRPEngine is build as dynamically a loadable module, that can be linked to any executable that has been build with the CapVRP header files. The build sequence and the used configure and make files are printed below. Development and deployment has been performed on a Linux¹ platform, with KDE Desktop Environment.

```
> cd /home/phoenix/phoenix-repos/trunk/app/capVRPEngine
> aclocal
> libtoolize -copy -automake
> aclocal
> autoconf
> autoheader
> automake -add-missing -copy
```

¹The Gentoo distribution has been used because of its configurability and superior version management system.



```
> ./configure  
> make && make install
```

Listing 9.4.1 – build sequence

```
AC_PREREQ(2.61)  
AC_INIT(capVRP, 1.0, phoenix@novum-it.com)  
AC_CONFIG_AUX_DIR(config)  
AM_INIT_AUTOMAKE(capVRPEngine, 1.0)  
AC_CONFIG_SRCDIR([../../com/optimization/capVRP/include/VRPInstance.h])  
AC_PROG_CXX  
AC_PROG_CC  
AC_LIBTOOL_DLOPEN  
AC_PROG_LIBTOOL  
AC_HEADER_STDBOOL  
AC_HEADER_STDC  
AC_C_CONST  
AC_CHECK_FUNCS([pow sqrt])  
AC_CONFIG_FILES([Makefile  
                 README])  
  
AC_OUTPUT
```

Listing 9.4.2 – configure.ac file

```
# Copyright (C) 2007 novum information technology bv <info@novum-it.com>  
#  
# This file is free software; as a special exception the author gives  
# unlimited permission to copy and/or distribute it, with or without  
# modifications, as long as this notice is preserved.  
#  
# This program is distributed in the hope that it will be useful, but  
# WITHOUT ANY WARRANTY, to the extent permitted by law; without even the  
# implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  
  
lib_LTLIBRARIES = libcapVRPEngine.la  
libcapVRPEngine_la_SOURCES =  
  $(TOP_SRCDIR)/phoenixrepos/trunk/com/optimization/capVRP/src/basicnode.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/src/location.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/src/edge.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/src/order.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/src/customer.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/src/resource.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/src/solutiontypes.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/util/xml/genericXMLReader.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/util/random/mtrand.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/src/capVRPInstance.cpp \  
  $(TOP_SRCDIR)/phoenix-  
    repos/trunk/com/optimization/capVRP/src/capVRPInstanceBuilder.cpp \  
  $(TOP_SRCDIR)/phoenix-  
    repos/trunk/com/optimization/capVRP/src/capVRPParameter.cpp \  
  $(TOP_SRCDIR)/phoenix-  
    repos/trunk/com/optimization/capVRP/src/populationCreationOperators.cpp \  
  $(TOP_SRCDIR)/phoenix-  
    repos/trunk/com/optimization/capVRP/src/selectionOperators.cpp \  
  $(TOP_SRCDIR)/phoenix-  
    repos/trunk/com/optimization/capVRP/src/variationOperators.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/src/vehiclePool.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/src/logger.cpp \  
  $(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/src/capVRPOptimize.cpp  
libcapVRPEngine_la_LDFLAGS = -version-info 1:0:0  
  
bin_PROGRAMS = capVRPRun  
capVRPRun_SOURCES = $(TOP_SRCDIR)/phoenix-  
  repos/trunk/com/optimization/capVRP/src/capVRPRun.cpp  
capVRPRun_LDADD =  
  -L$(TOP_SRCDIR)/phoenix-repos/trunk/lib -lcapVRPEngine \  

```



```
-L/usr/lib -lXerces-c
INCLUDES =
-I$(TOP_SRCDIR)/phoenix-repos/trunk/com/optimization/capVRP/include \
-I$(TOP_SRCDIR)/phoenix-repos/trunk/app/capVRPGui/src \
-I$(TOP_SRCDIR)/phoenix-repos/trunk/util \
-I/usr/include
```

Listing 9.4.3 – Makefile.am file

9.5 Deployment

Instance Encoding

For the prototype implementation the instance encoding is described in XML grammar and is read by the GenericXMLReader object and transformed into the CapVRPInstance and CapVRPParameter objects. The following table describes the syntax and semantics. The related XML Schema is shown in Appendix-A1 – CapVRP DTD Schema.

Table 9.5.1 – Instance encoding syntax and grammar

Token	Semantic	Application
CAPVRP		
CAPVRPInstance		
instancIdent		
instanceID	Name of the instance	required
setID	Name of the set of instances	required
nodes		
coordinates	Options NONE EUCLIDIAN	required
node+		
cityID	Unique identification of the location	required
cityName	Name of location	required
X	x-coordinate	required if
Y	y-coordinate	EUCLIDIAN
edges		
method	Options SPECIFY GENERATE	required
distance	Options SPECIFY EUCLIDIAN	required
triangleInequality	true false	required
symmetric	true false	required
edge?		required if method SPECIFY
edgeID	Unique identification	required
departureID	Departure nodeID	required
destinationID	Destination nodeID	required
distance	Road distance	required
order		
orderDescription	Description of order	required
depotLocation	Depot nodeID	required
orderItem+		
orderLine	Unique identification	required



Token	Semantic	Application
lineDescr	Description	required
destination	Destination nodeID	required
weight	Freight weight	required
volume	Freight volume	required
resources		
vehicle+		
vehicleID	Unique vehicle identification	required
vehicleDescr	Description	required
maxWeight	Maximum charge	required
maxVolume	Maximum volume	required
maxDistance	Maximum trip length	required
CAPVRParameters		
generation		
populationCreation	Indicates selected population creation strategy SINGLEDEPOT_SINGLECAPACITY MULTI (not implemented)	required
populationSize	Size of the population (9...)	required
generationsPerStep	Number per time step (9...)	required
maxGenerations	Maximum generations (9...)	required
utility	Utility measure TOTALDISTANCE OTHER (not implemented)	required
operatorClass	Class of operator for further extension CAPVRP DEFAULT (not implemented)	required
selectionOperator		
parentSelectionStrategy	Select strategy GREEDY STOCHASTIC (not implemented) MIXED TOURNAMENT	required
parentSetSize	Number parents for breeding (9...)	required
fractionStochasticInMixed	Only for MIXED strategy (99.9...)	optional (0,100)
replacementStrategy	Choice of ELITIST DEFAULT (not implemented)	required
variationOperator		
crossoverRate	Probability to apply (99.9...)	required (0,100)
mutationSwapRate	Probability to apply (99.9...)	required (0,100)
mutationInversionRate	Probability to apply (99.9...)	required (0,100)
mutationInsertionRate	Probability to apply (99.9...)	required (0,100)
mutationDisplacementsRate	Probability to apply (99.9...)	required (0,100)
monitors		
monitor+		
type	Choice of INCREMENT_MONITOR SAMPLE_MONITOR	required
Legend:		
token+	multiplicity = 1 – n	
token?	multiplicity = 0 – n	
token	multiplicity = 1	
(9...)	integer number	
(99.9...)	decimal number	



Runtime Invocation

The following console print shows an optimizer run executing 1000 steps, and solution print. The time needed for 520.000 generations was approximately 75 seconds. Result utility of 1880 was 6.6% from benchmark of 1764¹.

```
novum@ferdinand /home/phoenix/phoenix-repos/trunk/bin $ ./capVRPRun
/home/phoenix/phoenix-repos/trunk/app/capVRPGui/data/xml/capVRP_A-n80-k10-
tourn_default.xml 1000

capVRPRun started
capVRPInstance constructed
capVRPParameter constructed
capVRPInstanceBuilder constructed
genericXMLReader constructed
XML-parser started with file: /home/phoenix/phoenix-
repos/trunk/app/capVRPGui/data/xml/capVRP_A-n80-k10-tourn_default.xml
XML-parser start document
capVRPInstanceBuilder start
XML-parser end document
Start Edge generation
End Edge generation
Start Customer building
End Customer building
capVRPInstanceBuilder end
XML-parser completed succesfully
randomizer created with seed = 3696
Start Operator building
SingleDepotSingleCapacity population creator constructed
Vehicle pool created for population type = 202
Tournament parent selection operator constructed
Elitist replacement operator constructed
Mutation operator constructed
Crossover operator constructed
End Operator building
Optimizer created
logger created and openend with filename: A-n80-k10-tourn_default.log
start optimize with number of steps = 1000
logger closed
optimized - utility = 1880
Solution - utility = 1880.0454 - depot = n1
vRoute - vehicle = VF-01-01 : load = 0.0000 volume = 100.0000 path = Path object
Customer ID = 60035 n37 location = n37 load = 0.00 volume = 12.00
Customer ID = 60075 n77 location = n77 load = 0.00 volume = 14.00
Customer ID = 60071 n73 location = n73 load = 0.00 volume = 2.00
Customer ID = 60053 n55 location = n55 load = 0.00 volume = 2.00
Customer ID = 60008 n10 location = n10 load = 0.00 volume = 23.00
Customer ID = 60054 n56 location = n56 load = 0.00 volume = 14.00
Customer ID = 60032 n34 location = n34 load = 0.00 volume = 1.00
Customer ID = 60014 n16 location = n16 load = 0.00 volume = 2.00
Customer ID = 60040 n42 location = n42 load = 0.00 volume = 13.00
Customer ID = 60045 n47 location = n47 load = 0.00 volume = 11.00
Customer ID = 60063 n65 location = n65 load = 0.00 volume = 6.00

vRoute - vehicle = VF-01-01 : load = 0.0000 volume = 100.0000 path = Path object
Customer ID = 60073 n75 location = n75 load = 0.00 volume = 19.00
Customer ID = 60028 n30 location = n30 load = 0.00 volume = 10.00
Customer ID = 60004 n6 location = n6 load = 0.00 volume = 11.00
Customer ID = 60058 n60 location = n60 load = 0.00 volume = 22.00
Customer ID = 60026 n28 location = n28 load = 0.00 volume = 4.00
Customer ID = 60059 n61 location = n61 load = 0.00 volume = 13.00
Customer ID = 60038 n40 location = n40 load = 0.00 volume = 21.00

vRoute - vehicle = VF-01-01 : load = 0.0000 volume = 97.0000 path = Path object
```

¹See chapter 10 – Experimentation for information on the used benchmarks.



```
Customer ID = 60072 n74 location = n74 load = 0.00 volume = 12.00
Customer ID = 60057 n59 location = n59 load = 0.00 volume = 7.00
Customer ID = 60031 n33 location = n33 load = 0.00 volume = 9.00
Customer ID = 60003 n5 location = n5 load = 0.00 volume = 5.00
Customer ID = 60021 n23 location = n23 load = 0.00 volume = 26.00
Customer ID = 60044 n46 location = n46 load = 0.00 volume = 23.00
Customer ID = 60049 n51 location = n51 load = 0.00 volume = 10.00
Customer ID = 60069 n71 location = n71 load = 0.00 volume = 5.00

vRoute - vehicle = VF-01-01 : load = 0.0000 volume = 99.0000 path = Path object
Customer ID = 60005 n7 location = n7 load = 0.00 volume = 23.00
Customer ID = 60007 n9 location = n9 load = 0.00 volume = 9.00
Customer ID = 60036 n38 location = n38 load = 0.00 volume = 14.00
Customer ID = 60001 n3 location = n3 load = 0.00 volume = 22.00
Customer ID = 60023 n25 location = n25 load = 0.00 volume = 7.00
Customer ID = 60000 n2 location = n2 load = 0.00 volume = 24.00

vRoute - vehicle = VF-01-01 : load = 0.0000 volume = 98.0000 path = Path object
Customer ID = 60016 n18 location = n18 load = 0.00 volume = 20.00
Customer ID = 60018 n20 location = n20 load = 0.00 volume = 12.00
Customer ID = 60025 n27 location = n27 load = 0.00 volume = 4.00
Customer ID = 60034 n36 location = n36 load = 0.00 volume = 2.00
Customer ID = 60064 n66 location = n66 load = 0.00 volume = 2.00
Customer ID = 60068 n70 location = n70 load = 0.00 volume = 9.00
Customer ID = 60055 n57 location = n57 load = 0.00 volume = 7.00
Customer ID = 60046 n48 location = n48 load = 0.00 volume = 2.00
Customer ID = 60024 n26 location = n26 load = 0.00 volume = 12.00
Customer ID = 60002 n4 location = n4 load = 0.00 volume = 23.00
Customer ID = 60076 n78 location = n78 load = 0.00 volume = 2.00
Customer ID = 60050 n52 location = n52 load = 0.00 volume = 3.00

vRoute - vehicle = VF-01-01 : load = 0.0000 volume = 95.0000 path = Path object
Customer ID = 60029 n31 location = n31 load = 0.00 volume = 9.00
Customer ID = 60077 n79 location = n79 load = 0.00 volume = 2.00
Customer ID = 60067 n69 location = n69 load = 0.00 volume = 9.00
Customer ID = 60042 n44 location = n44 load = 0.00 volume = 3.00
Customer ID = 60015 n17 location = n17 load = 0.00 volume = 6.00
Customer ID = 60060 n62 location = n62 load = 0.00 volume = 22.00
Customer ID = 60056 n58 location = n58 load = 0.00 volume = 21.00
Customer ID = 60074 n76 location = n76 load = 0.00 volume = 6.00
Customer ID = 60019 n21 location = n21 load = 0.00 volume = 15.00
Customer ID = 60030 n32 location = n32 load = 0.00 volume = 2.00

vRoute - vehicle = VF-01-01 : load = 0.0000 volume = 96.0000 path = Path object
Customer ID = 60010 n12 location = n12 load = 0.00 volume = 14.00
Customer ID = 60033 n35 location = n35 load = 0.00 volume = 2.00
Customer ID = 60078 n80 location = n80 load = 0.00 volume = 24.00
Customer ID = 60017 n19 location = n19 load = 0.00 volume = 26.00
Customer ID = 60047 n49 location = n49 load = 0.00 volume = 7.00
Customer ID = 60013 n15 location = n15 load = 0.00 volume = 2.00
Customer ID = 60070 n72 location = n72 load = 0.00 volume = 12.00
Customer ID = 60009 n11 location = n11 load = 0.00 volume = 9.00

vRoute - vehicle = VF-01-01 : load = 0.0000 volume = 88.0000 path = Path object
Customer ID = 60048 n50 location = n50 load = 0.00 volume = 13.00
Customer ID = 60037 n39 location = n39 load = 0.00 volume = 23.00
Customer ID = 60066 n68 location = n68 load = 0.00 volume = 5.00
Customer ID = 60065 n67 location = n67 load = 0.00 volume = 11.00
Customer ID = 60052 n54 location = n54 load = 0.00 volume = 13.00
Customer ID = 60041 n43 location = n43 load = 0.00 volume = 23.00

vRoute - vehicle = VF-01-01 : load = 0.0000 volume = 100.0000 path = Path object
Customer ID = 60039 n41 location = n41 load = 0.00 volume = 13.00
Customer ID = 60020 n22 location = n22 load = 0.00 volume = 13.00
Customer ID = 60027 n29 location = n29 load = 0.00 volume = 20.00
Customer ID = 60051 n53 location = n53 load = 0.00 volume = 6.00
Customer ID = 60062 n64 location = n64 load = 0.00 volume = 22.00
Customer ID = 60006 n8 location = n8 load = 0.00 volume = 26.00
```




10 Experimentation

10.1 Introduction

With reference to paragraph 7.4 – Rationale, the proof of concept is about providing a convincing argument that:

- usage of meta heuristics in general and in this case an genetic implementation, produces sufficient *quality* and *robust* results in order to be a base for pooling decisions and,
- pooling itself is potentially *profitable* and to what extend.

The experimentation described in the next paragraphs has been performed in order to substantiate that argument.

An extensive collection of test have been performed with a non-trivial instance of a well-known benchmark: Augerat Set A instance A80k10. This benchmark and others are presented on [VRP Web](#), a website dedicated to the Vehicle Routing Problem and published by the Networking and Emerging Optimization (NEO) Research Institute of the University of Malaga, Spain. Along the published instances *best known results* are provided that serve as benchmark for the particular instance. Pereira et al. [9] published experiments using a great number of instances of Augerat A and B sets – including the A80k10 instance – and Christofides and Eilon set E.

Using identical optimization parameters to our settings, Pereira concludes that the results show consistent approximation behavior across all sets in terms of average optimum reached, best solution, and distance to best known solution. Given this conclusion we will concentrate on the set A80k10, which is shown in Appendix-A – Augerat A80k10 and experiment with variations, that have not been covered that research.

The instance defines 80 points in the Euclidean plane as destinations linked one on one (*bijection*) with a customer demand. The first location is the depot with demand zero. The total demand of 942 has to be satisfied by at least 10 identical vehicles with capacity 100. Utility is measured in total distance traveled to satisfy demand. The benchmark of best known utility for the A80k10 instance is 1764.

In paragraph 10.3 – Profitability, we will in addition to A80k10 also use the smaller instance A32k5 to get an impression about the variability of efficiency gains through pooling.

10.2 Quality And Robustness

First we will experiment with a mixed parent selection strategy and vary with population size and the mix greedy versus stochastic parent selection. Table 10.2.1 shows the results of 50 runs each with 100 steps resulting in 50.000 generations per run.

The default settings are: population size = 50, mixed selection strategy with 20% random selected parents, elitist replacement strategy, crossover rate 75%, swap rate 5%, inversion rate 15%, insertion rate 5% and displacement rate 20%.

Variations with crossover, swat, inversion, insertion and displacement settings have been extensively performed by Pereira, so we will not repeat that here.

From this experiment it seems that the optimization is most sensitive for the parent set



size, which seems to be best around 20. Increasing the stochastic contribution in the parent set seems to have deteriorating effect on the solution quality, probably by introducing too many sub-quality parents resulting in imbalanced diversity. Although the crossover operation is very important in improving offspring, variations with the rate within reasonable bounds seem not to have a great effect on overall solutions quality.

Table 10.2.1 – Mixed strategy variations

Variation				Best solution	Worst solution	Average solution	Average as distance from benchmark
Parent set	mix	Xover rate					
Mixed strategy with default parameter settings over 50 runs	5	0.2	0.75	1910.85	2265.65	2081.61	18.0%
	5	0.4	0.75	1959.78	2267.62	2103.78	19.3%
	10	0.2	0.75	1899.08	2213.13	2059.65	16.8%
	20	0.2	0.75	1890.04	2169.48	2013.48	14.2%
	20	0.4	0.75	1897.07	2196.80	2048.80	16.1%
	20	0.2	0.8	1881.96	2196.65	2021.53	14.6%

It is, however, clear that the reachable solution quality is somewhat disappointing with at best 14% off benchmark in the mixed strategy.

Consequently it is interesting to see whether other parent selection strategies – deterministic (greedy) or tournament – yield better results. In table 10.2.2 the results of the various strategies with the default instance are presented.

Table 10.2.2 – Parent selection strategy comparison

Parent selection strategy with default parameter settings	Best solution	Worst solution	Average solution	Average as distance from benchmark
Deterministic or Greedy	1878.88	2289.70	2082.36	18.0%
Mixed	1910.85	2265.65	2081.61	18.0%
Tournament	1874.16	2148.04	1985.08	12.5%

From above experiment it is clear that the tournament strategy has a substantially better average solution quality, although there is still potential for improvement. The Pereira study mentions a best average of around 1810, that is about 3% distance from benchmark.

Beside average solution quality, asymptotic behavior is another important consideration. Fast convergence to a reasonable solution quality is a major design consideration in the proof of concept.

In figure 10.2.1 we analyze the three selection strategies in terms of asymptotic



behavior. Beside the fact that the tournament strategy yields better overall solution quality, its asymptotic behavior is much more pronounced than the other greedy and mixed methods. A possible explanation would be that the tournament method better exploits the diversity in the population, a pre-condition for having a better probability to escape local optima, and consequently consistently produces better solutions. An overall conclusion consistent with other research, is that the genetic algorithm shows initially a very fast convergence followed by an almost steady state. In the experiments this appears to be happening after approximately 25 steps equivalent to 10.000 generations or more.

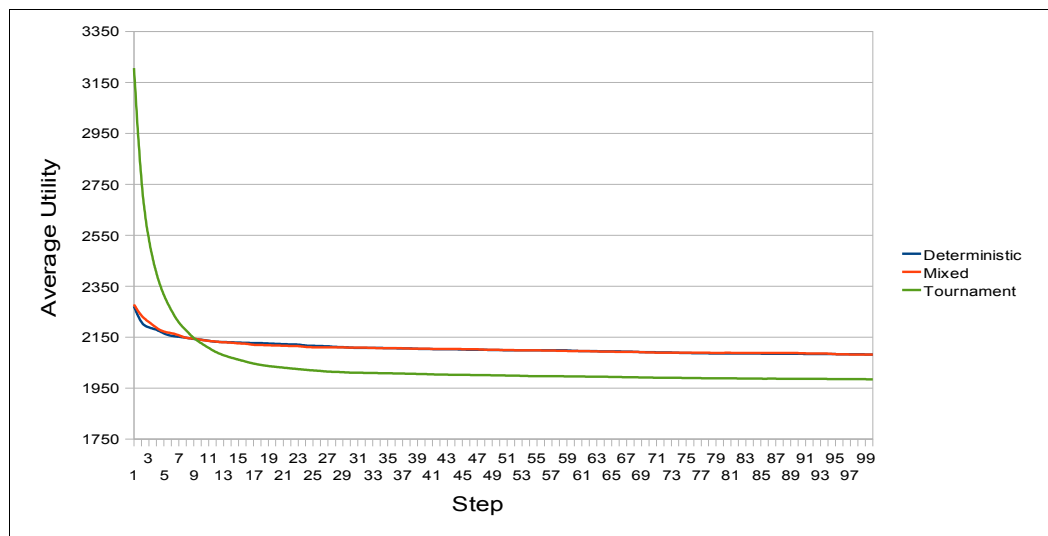


Figure 10.2.1 – Asymptotic behavior by parent selection strategy

Following the previous observation we now focus on the behavior of the tournament method in more detail to find out how convergence is related to solution quality. The following figures show convergence graphs in absolute utility (10.2.2) and relative utility as percentage of distance to own end point (10.2.3), for best, average and worst solutions in a 50 run sequence.

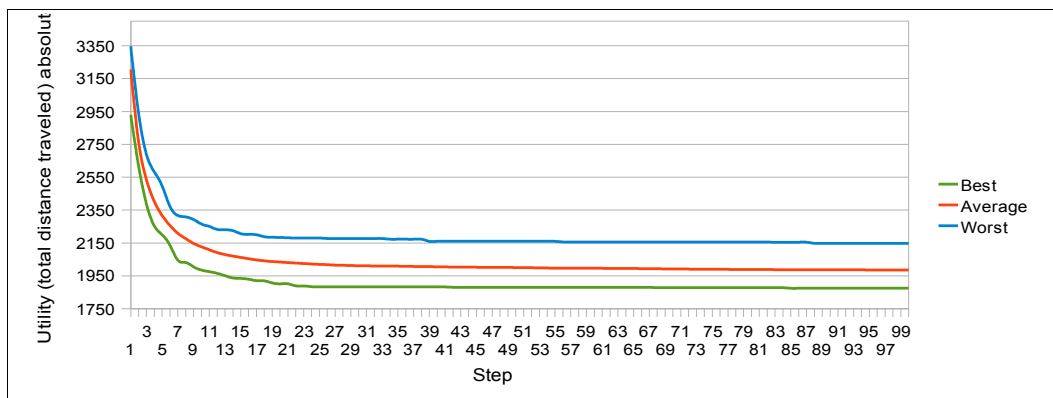


Figure 10.2.2 – Asymptotic behavior in absolute values with tournament method

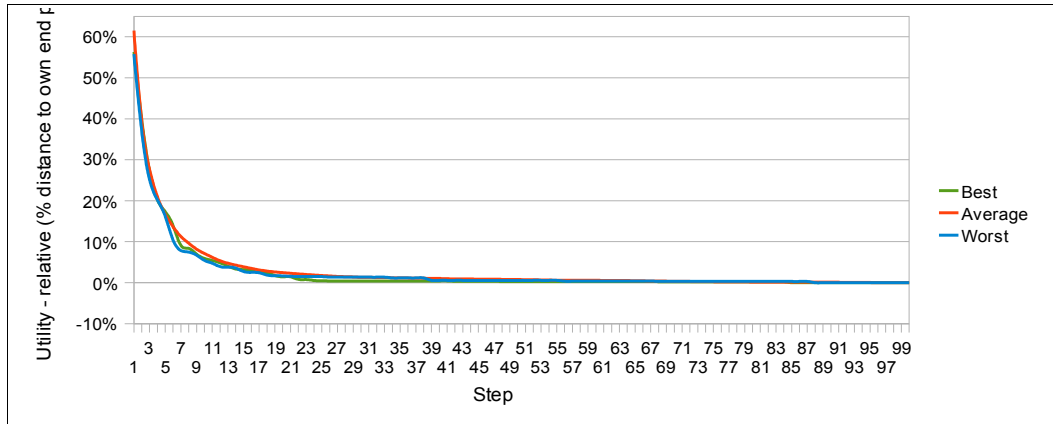


Figure 10.2.3 – Asymptotic behavior relative to end point

Figure 10.2.3 shows that convergence is invariant to solution quality and figure 10.2.2 suggests that the optimal solution is depending in the initial quality. This might indicate that population genetic diversity is not well balanced as to escape local optima. The next table 10.2.3 shows experimentation results with a changed population creation operator that better preserves the random diversity into the initial population. As the the genetic diversity in subsequent populations is strongly dependent on the crossover operator we vary with the crossover probabilities.

Table 10.2.3 – Results of various crossover rates over more genetic diverse population

Variation		Best solution	Worst solution	Average solution	Average as distance from benchmark
Higher diverse population Population size 200	Xover rate				
Parent set = 5	0.95	1836.79	2113.50	1980.80	12.3%
	0.85	1890.71	2148.96	1998.63	13.3%
	0.75	1896.44	2137.63	2002.15	13.5%
	0.65	1880.32	2139.63	1994.98	13.1%
	0.55	1897.08	2183.33	2022.33	14.6%
Parent set = 4	0.75	1851.41	2139.05	1973.16	11.9%

The results show, however, that increasing diversity within constant population size is not improving average solution quality. There is a weak indication, that increasing crossover rate might improve average solution quality, although not significantly.

The robustness (stability) of the algorithm can be assessed by comparing average solution quality when increasing the number of runs. Table 10.2.4 shows a stable behavior of the algorithm when increasing number of runs twenty fold.



Table 10.2.4 – Tournament strategy stability

Variation		Best solution	Worst solution	Average solution	Average as distance from benchmark
Runs					
Tournament strategy with default parameter settings and parent set = 4	50	1843.80	2094.87	1973.25	11.9%
	150	1843.80	2094.87	1969.83	11.7%
	1150	1827.32	2246.33	1976.67	12.1%

The effect of parent set size in the tournament selection operator shown in table 10.2.5 shows a significant increase in average solution quality while reducing the parent set to 4 which is the – technical – minimum for a *single tournament* strategy.

Table 10.2.5 – Tournament strategy parent set variations

Variation		Best solution	Worst solution	Average solution	Average as distance from benchmark
Population size 200					
Parent set					
Tournament strategy with default parameter settings exclusive parent set over 50 runs	5	1874.16	2148.04	1985.08	12.5%
	4	1843.80	2094.87	1973.25	11.9%

Finally the effect of population size with the tournament selection operator shown in table 10.2.6 shows a significant increase in average solution quality, however scales approximately in $O(n)$.

Table 10.2.6 – Tournament strategy population size variations

Variation		Run time (sec)	Best solution	Worst solution	Average solution	Avg. distance benchmark
Parent set size 4						
Population						
Tournament strategy with default parameter settings exclusive population over 50 runs	200	10	1843.80	2094.87	1973.25	11.9%
	500	35	1837.40	2019.49	1932.92	9.6%
	1000	125	1824.35	1894.11	1959.93	7.4%



The distribution of solutions over the utility range is shown in figure 10.2.5.

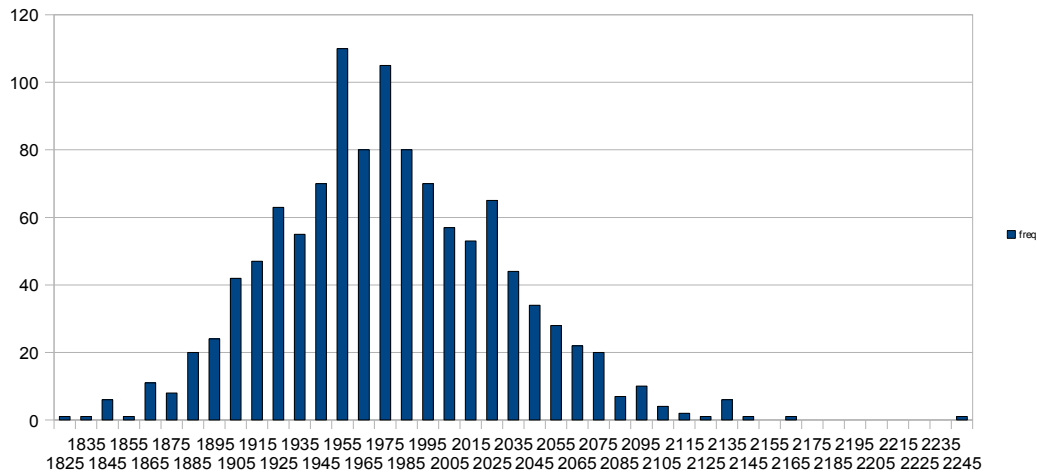


Figure 10.2.5 – Utility distribution over 1150 runs, tournament strategy with parent set 4

This appears to resemble a *normal distribution* with *mean* = 1976.67 and *standard deviation* of 53.41.

The conclusions from our experimentation seem to be consistent with results of the similar experiments performed by Pereira et.all [9] that uses a great number of instances of Augerat A and B sets, including the A80k10 instance.

10.3 Profitability

In this paragraph we will review, based on the experimentation results, whether pooling is potentially beneficial for both parties. We will use the A80k10 instance as we have research its behavior and have a good impression of its asymptotic behavior, robustness and average solution quality. We will use the tournament selection method with a parent set of 4 and further the default optimization settings as mentioned before. In addition to A80k10 we will also use a smaller instance A32k5 to get an impression of the variability of efficiency gains over instances.

In order to simulate the pooling versus non-pooling situation using A80K10 and A32k5, we will regard the A80k10/A32k5 solution as the pooled solution and compare it with two 'extreme' non-pooled situations derived from the A80k10/A32k5 instances as follows:

1. divide the A80k10/A32k5 instances into two instances A and B, with disjunct node sets of equal size. This situation resembles two competing companies, each with own unique customers;
2. divide the A80k10/A32k5 instances into two identical instances C, with half of the original demand per customer.

Real life situation will often be a mix of these extremes. By calculating these two non-pooling situations and comparing with the pooled one gives a pragmatic impression of the profit potential of the pooling option.



Table 10.3.1 – Comparison of non-pooling options versus pooling

Instance A80k10	Best solution	Worst solution	Average solution	Saving by pooling
Sum of solutions disjunct sets A and B	2083	2274	2153	9,10%
Solutions C times 2	2396	2686	2525	28.0%
Pooled	1844	2095	1973	n.a.

Figure 103.1. shows a frequency distribution A80k10 for pooled and non-pooled options

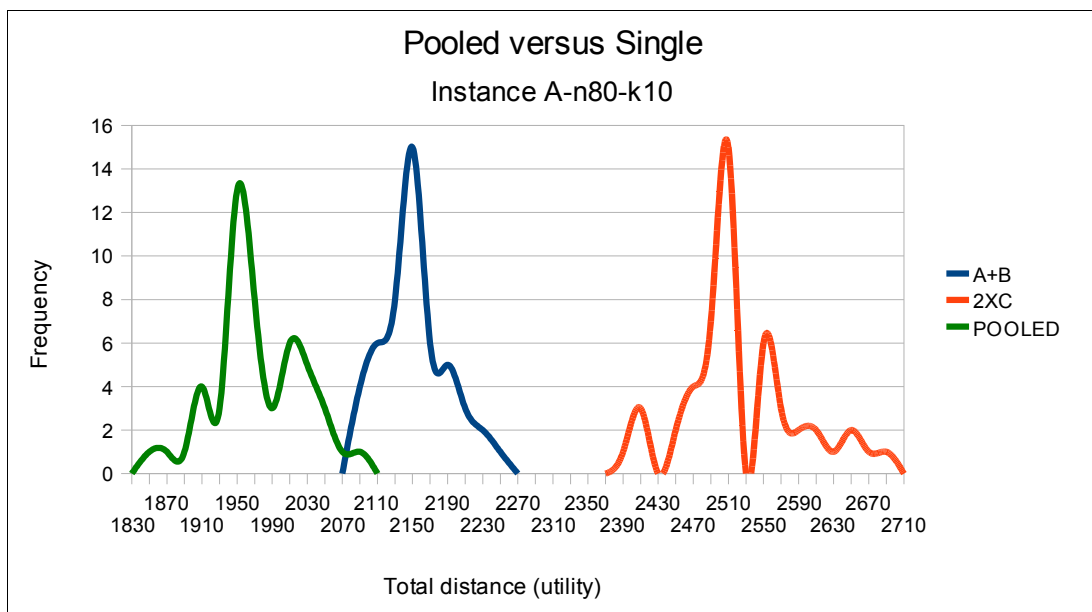


Figure 10.3.1 – Frequency distribution comparison A80k10 pooled versus non-pooled options

In order to see if this effect is consistent across instances we also selected an other instance A32k5 from the Agerat set with 32 destinations. Here the pooling profitability were much more pronounced as shown in table 10.3.2.

Table 10.3.2 – Comparison instance A32k5 of pooling options versus non-pooling

Instance A32k5	Best solution	Worst solution	Average solution	Saving by pooling
Sum of solutions disjunct sets A and B	977	974	969	19.3%
Solutions C times 2	1127	1337	1160	42.7%
Pooled	787	857	812 ¹	n.a.

¹ Average solution quality is 3.7% off benchmark's best solution

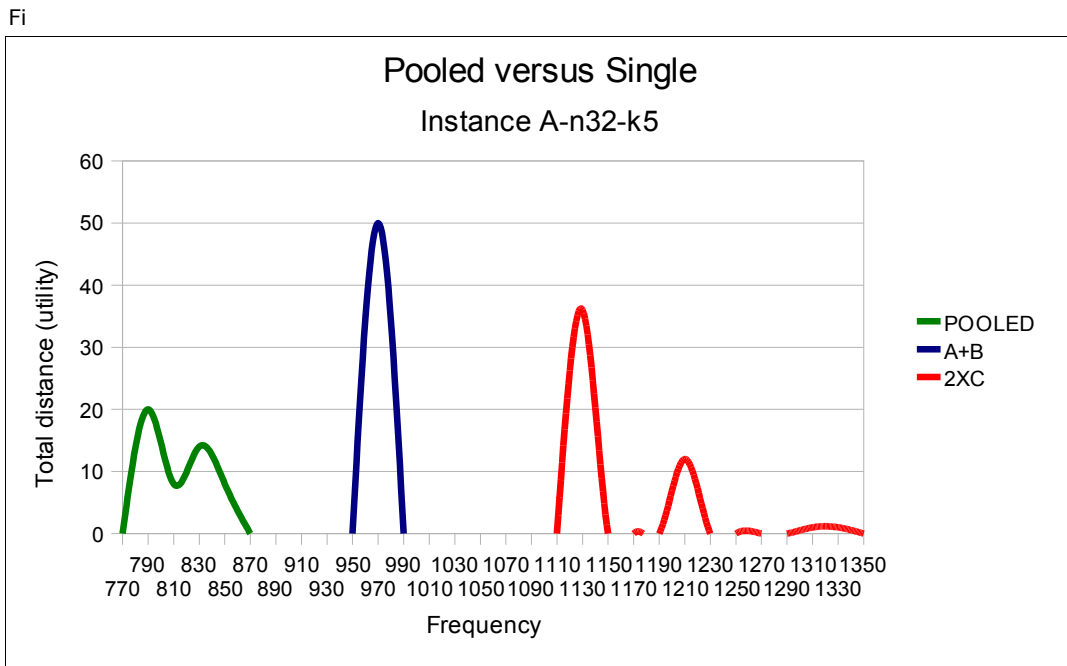


Figure 10.3.1 – Frequency distribution comparison A32k5 pooled versus non-pooled options

The pooled options are on average superior to any of the two non-pooled alternatives. In these instances pooling is always substantially more profitable as the individual solutions together.



11 Conclusions

In this chapter we summarize our conclusions with regard to the goals and sub goals set in paragraph 3.7 – Methodology. Finally we make recommendations for further research.

11.1 Literature Study

The selected literature was strongly focused on the state-of-the-art of the core technologies instrumental in the realization of the Phoenix project proposal. Central to these technologies is the computational complexity of the combinatorial optimization problems, deliberation and decision models, and the equilibria models used in conflict resolution arising in bargaining situations. Often very sound mathematical models for these problems exist, but solving these models for realistic instances is in many cases computationally intractable.

Especially in the area of meta-heuristics there is a strong and fast growing research base in solving many of these NP-hard problems with good solution quality. For the Capacitated Vehicle Routing Problem there exist many well researched and documented meta-heuristics, of which the genetic algorithm is a very interesting one. The chromosome encoding by Pereira et al. [9][10] where the genes are tightly coupled to domain specific structures – in this case a collection of tours – basic for the optimization goal, is a promising approach.

Related to the complexity issue is the rethinking now in evidence of the *perfect rationality* concept central to classic artificial intelligence. An interesting development is the *bounded optimality* concept [2] where problem architecture and task environment constraint an agent's optimum.

The literature study in the area of bargaining has been focused on the underlying game theoretic models and the *rational agent* model. In combination with the state-of-the-art regarding meta-heuristics, realization of the Phoenix project must be a feasible though challenging venture. Many of the core elements are designated as research topics in the project proposal.

11.2 Phoenix Project Proposal

The initial project outline was well received in the European Partner Search facility IDEAL-IST, and attracted in one week 25 expressions of interest, most of whom were of very good quality and well distributed over the various EU countries and SME, research and industry. Based on the response a quality consortium could be formed within a very short period of time.

Although the time of one month to submitting of the project proposal was very limited, the overall rating of EU Evaluating Committee were good to very good, although not good enough to reach the threshold. Especially the relevance of the proposal's objective as stated by the committee "...to help SME's to get more involved in a complex digital environment..." was rated very good.



11.3 Prototype

The prototype fitted very well into the overall architecture as an implementation of a CapVRP optimizer. Given the service oriented architecture, an extended choice of optimizers based on other classes of meta-heuristics such as ant-colony or tabu-search, remain possible provided these optimizers conform the stated functional and non-functional requirements and CapVRPEngine API.

The mathematical CapVRP problem definition and its genetic representation provide a powerful description model for specifying the chromosome structure, the solution space and the various operators on these representations. The model is very well mappable on a class model, containing specifications of data types and methods. The class model proved to be a very sound basis to implement a variety of genetic operators compliant to abstract operator classes (interfaces) that can be instantiated through instance parameters.

The implementation in C++ on a Unix (Linux) platform resulted in a very stable and fast runtime environment.

11.4 Proof Of Concept

Considering the results presented in the previous chapter it is a fair conclusion that the proof on concept has been delivered. This conclusion is motivated referring to the two objectives stated in paragraph 7.4 – Rationale:

- efficiency gains through advanced optimizations and,
- efficiency gains by plan and resource pooling.

Advanced optimization in general and in particular genetic meta-heuristics, have very interesting properties in producing acceptable solution for real-life problems. This class of optimizers as demonstrated in this thesis are capable of solving complex planning situation with acceptable solution quality and within limited run time.

The optimization process shows fast and stable approximation behavior. Within about 25 generation steps the approximation has reached near optimum reachable in the step. This number of approximation steps takes about 2 seconds runtime.

The limited runtime resource consumption make it feasible to perform hundreds of runs and selectively take the best solution. As shown in figure 10.2.5 a high probability can be reached to get a near-optimum solution.

The average solution quality is robust. The average solution stays stable after some 50 runs, even when continuing for more than 1000 runs. The frequency distribution of solutions over an increasing number of runs appears to tend to a normal distribution. In the 1150 runs the mean was 1976 and standard deviation 53.

Although our optimizer's average solution is about 11% off best known solution. Pereira et al. show that an average solution quality is sustainable about 3% off best known. This shows that our genetic representation is very promising and further improvement is feasible.

Comparing pooling versus 2 extreme non-pooling scenarios shows that pooling in all cases is profitable. The best non-pooling solution is equal to the worst pooling solution. The *efficiency loss* of non-pooling lies between 9% and 28% relative to the pooling solution. This result support our intuition of about 20% efficiency gain.



11.5 Recommendations

Further research is needed in order to improve the operator quality in terms of maintaining a balanced diversity in the population, necessary to escape local optima. The research should be directed at an improved domain specific gene representation in the chromosome and exploitation of these characteristics through smarter operators .

The prototype was based on a single objective of the pooling partners being minimization of traveled distance. For profit distribution this results in a *zero-sum game*. In real life, however, the utility function of the partners often have multiple disjunct sets of objectives. This creates opportunities for synergy effects beneficial for both parties. Multi-objective optimization leading to an improved synergy will substantially widen the applicability of the technologies discussed in this thesis.



This page is intentionally left blank.



12 References

- [1] Russel, S., Norvig, P., "Artificial Intelligence – a Modern Approach" 2nd edition, Prentice Hall, 2003.
- [2] Russel, S., Subramanian, D., "Provably Bounded-Optimal Agents " Journal of Artificial Intelligence Research 2 (1995) page 575-609.
- [3] Larson, K., Sandholm, T., "Bargaining with limited computation: deliberation equilibrium" Carnegie Mellon University, Pittsburgh PA USA. 2001.
- [4] Larson, K., Sandholm, T., "Using performance profile trees to improve deliberation control", Computer Science Department, Carnegie Mellon University, Pittsburgh, PA 15213, USA. ©2004 American Association for Artificial Intelligence (www.aaai.org).
- [6] Michalewicz, Z., Fogel, D., "How to solve it: Modern Heuristics" 2nd edition Springer Verlag (2004).
- [7] Ausiello, G., et al., "Complexity and Approximation: Combinatorial Optimization Problems and their Approximation Properties" Springer Verlag 1999.
- [8] Hansen, E., Zilberstein, S., "Monitoring and control of anytime algorithms: a dynamic programming approach" Mississippi State University, Mississippi MS, University of Massachusetts, Amherst MA USA, 2001.
- [9] Pereira, F., Tavares, J., Machado, P., Costa, E., "GVR: a New Genetic Representation for the Vehicle Routing Problem". Centro de Informatica e Systemas de Universidade de Coimbra, Portugal.
- [10] Tavares, J., Pereira, F., Machado, P., Costa, E., "Crossover and Diversity: A Study about GVR". Centro de Informatica e Systemas de Universidade de Coimbra, Portugal.
- [11] Myerson, R., "Game Theory: Analysis of Conflict" 6th printing Harvard University Press, Cambridge MA USA 2004
- [12] Halpern, J., "Reasoning about Uncertainty" The MIT Press, Cambridge MA USA 2003.
- [13] Schaerf, M., Cadoli, M., "Tractable Reasoning via Approximation" Universita di Roma, Rome Italy.
- [14] Bellifemine, F., Caire, G., Poggi, A., Rimassa, G., "JADE: Java Agent Development Framework. A White Paper" Exp – Volume-3 – n.3, September 2003.
<http://jade.tilab.com>
- [15] FIPA "FIPA Abstract Architecture Specification" FIPA Architecture Board – 2004-03-18, www.fipa.org
- [16] FIPA, "FIPA Agent Management Specification", FIPA Architecture Board – 2004-03-18, www.fipa.org
- [17] FIPA, "FIPA Network Management and Provisioning Specification", FIPA Architecture Board – 2001-10-08, www.fipa.org
- [18] FIPA, "FIPA Personal Travel Assistance Specification", FIPA Architecture Board – 2001-10-08, www.fipa.org
- [19] Michalewicz, Z., "Genetic Algorithms + Data Structures = Evolution Programs" 3rd edition. Springer Verlag Berlin Heidelberg (1996).



This page is intentionally left blank.



13 Appendix A

This Appendix contains the XML encoding and instance file used in the experimentation described in chapter 10 – Experimentation.

13.1 CapVRP DTD Schema

```
<?xml version="1.0" encoding="UTF-8" ?>

<!ELEMENT CAPVRP          ( CAPVRPInstance,
                           CAPVRParameters ) >

<!ELEMENT CAPVRPInstance ( instanceIdent,
                           nodes,
                           edges,
                           order,
                           resources ) >

<!ELEMENT nodes           ( node+ ) >
<!ELEMENT edges           ( edge? ) >
<!ELEMENT order           ( orderItem+ ) >
<!ELEMENT resources       ( vehicle+ ) >

<!ELEMENT instanceIdent  (#PCDATA) >
<!ATTLIST instanceIdent instanceID CDATA #REQUIRED
                           setID CDATA #REQUIRED >

<!ATTLIST nodes coordinates ( NONE | EUCLIDIAN ) #REQUIRED >

<!ELEMENT node (#PCDATA) >
<!ATTLIST node cityID CDATA #REQUIRED
               cityName CDATA #REQUIRED
               X CDATA #IMPLIED
```



```
Y CDATA #IMPLIED >

<!ATTLIST edges method ( SPECIFY | GENERATE ) #REQUIRED
distance ( SPECIFY | EUCLIDIAN ) #REQUIRED
triangleInequality ( true | false ) #REQUIRED
symmetric ( true | false ) #REQUIRED >

<!ELEMENT edge (#PCDATA) >
<!ATTLIST edge edgeID CDATA #REQUIRED
departureID CDATA #REQUIRED
destinationID CDATA #REQUIRED
distance CDATA #IMPLIED >

<!ATTLIST order orderDescription CDATA #REQUIRED
depotLocation CDATA #REQUIRED >

<!ELEMENT orderItem (#PCDATA) >
<!ATTLIST orderItem orderLine CDATA #REQUIRED
lineDescr CDATA #REQUIRED
destination CDATA #REQUIRED
weight CDATA #REQUIRED
volume CDATA #REQUIRED >

<!ELEMENT vehicle (#PCDATA) >
<!ATTLIST vehicle vehicleID CDATA #REQUIRED
vehicleDescr CDATA #REQUIRED
maxWeight CDATA #REQUIRED
maxVolume CDATA #REQUIRED
maxDistance CDATA #REQUIRED >

<!ELEMENT CAPVRParameters ( generation,
selectionOperator,
variationOperator,
monitors ) >

<!ELEMENT generation (#PCDATA) >
<!ATTLIST generation populationCreation ( SINGLEDEPOT_SINGLECAPACITY | MULTI ) #REQUIRED
populationSize CDATA #REQUIRED
generationsPerStep CDATA #REQUIRED
maxGenerations CDATA #REQUIRED
```



```
utility ( TOTALDISTANCE | OTHER ) #REQUIRED
operatorClass ( CAPVRP | DEFAULT ) #REQUIRED>

<!ELEMENT selectionOperator (#PCDATA) >
<!ATTLIST selectionOperator parentSelectionStrategy ( GREEDY | STOCHASTIC | MIXED | TOURNAMENT ) #REQUIRED
parentSetSize CDATA #REQUIRED
fractionStochasticInMixed CDATA #IMPLIED
replacementStrategy ( ELITIST | DEFAULT ) #REQUIRED >

<!ELEMENT variationOperator (#PCDATA) >
<!ATTLIST variationOperator crossoverRate CDATA #REQUIRED
mutationSwapRate CDATA #REQUIRED
mutationInversionRate CDATA #REQUIRED
mutationInsertionRate CDATA #REQUIRED
mutationDisplacementRate CDATA #REQUIRED >

<!ELEMENT monitors ( monitor+ ) >
<!ELEMENT monitor (#PCDATA) >
<!ATTLIST monitor type (INCREMENT_MONITOR | SAMPLE_MONITOR) #REQUIRED >
```

13.2 Default Instance File

The instance file used for experimentation based on Augerat A80k10 instance

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE CAPVRP SYSTEM "file:///home/phoenix/phoenix-repos/trunk/app/capVRPGui/data/xml/phoenix.dtd" >
<CAPVRP>
  <CAPVRPInstance>
    <instanceIdent instanceID="Augerat: A-n80-k10 benchmark=1764" setID="A-n80-k10-tourn_default_pset4" />
    <nodes coordinates="EUCLIDIAN" >
      <node cityID="101" cityName="n1" X="92" Y="92" />
      <node cityID="102" cityName="n2" X="88" Y="58" />
      <node cityID="103" cityName="n3" X="70" Y="6" />
    </nodes>
  </CAPVRPInstance>
</CAPVRP>
```



```
<node cityID="104" cityName="n4" X="57" Y="59" />
<node cityID="105" cityName="n5" X="0" Y="98" />
<node cityID="106" cityName="n6" X="61" Y="38" />
<node cityID="107" cityName="n7" X="65" Y="22" />
<node cityID="108" cityName="n8" X="91" Y="52" />
<node cityID="109" cityName="n9" X="59" Y="2" />
<node cityID="110" cityName="n10" X="3" Y="54" />
<node cityID="111" cityName="n11" X="95" Y="38" />
<node cityID="112" cityName="n12" X="80" Y="28" />
<node cityID="113" cityName="n13" X="66" Y="42" />
<node cityID="114" cityName="n14" X="79" Y="74" />
<node cityID="115" cityName="n15" X="99" Y="25" />
<node cityID="116" cityName="n16" X="20" Y="43" />
<node cityID="117" cityName="n17" X="40" Y="3" />
<node cityID="118" cityName="n18" X="50" Y="42" />
<node cityID="119" cityName="n19" X="97" Y="0" />
<node cityID="120" cityName="n20" X="21" Y="19" />
<node cityID="121" cityName="n21" X="36" Y="21" />
<node cityID="122" cityName="n22" X="100" Y="61" />
<node cityID="123" cityName="n23" X="11" Y="85" />
<node cityID="124" cityName="n24" X="69" Y="35" />
<node cityID="125" cityName="n25" X="69" Y="22" />
<node cityID="126" cityName="n26" X="29" Y="35" />
<node cityID="127" cityName="n27" X="14" Y="9" />
<node cityID="128" cityName="n28" X="50" Y="33" />
<node cityID="129" cityName="n29" X="89" Y="17" />
<node cityID="130" cityName="n30" X="57" Y="44" />
<node cityID="131" cityName="n31" X="60" Y="25" />
<node cityID="132" cityName="n32" X="48" Y="42" />
<node cityID="133" cityName="n33" X="17" Y="93" />
<node cityID="134" cityName="n34" X="21" Y="50" />
<node cityID="135" cityName="n35" X="77" Y="18" />
<node cityID="136" cityName="n36" X="2" Y="4" />
<node cityID="137" cityName="n37" X="63" Y="83" />
<node cityID="138" cityName="n38" X="68" Y="6" />
<node cityID="139" cityName="n39" X="41" Y="95" />
<node cityID="140" cityName="n40" X="48" Y="54" />
<node cityID="141" cityName="n41" X="98" Y="73" />
<node cityID="142" cityName="n42" X="26" Y="39" />
<node cityID="143" cityName="n43" X="69" Y="76" />
```




```
<node cityID="144" cityName="n44" X="40" Y="1" />
<node cityID="145" cityName="n45" X="65" Y="41" />
<node cityID="146" cityName="n46" X="14" Y="86" />
<node cityID="147" cityName="n47" X="32" Y="39" />
<node cityID="148" cityName="n48" X="14" Y="24" />
<node cityID="149" cityName="n49" X="96" Y="5" />
<node cityID="150" cityName="n50" X="82" Y="98" />
<node cityID="151" cityName="n51" X="23" Y="85" />
<node cityID="152" cityName="n52" X="63" Y="69" />
<node cityID="153" cityName="n53" X="87" Y="19" />
<node cityID="154" cityName="n54" X="56" Y="75" />
<node cityID="155" cityName="n55" X="15" Y="63" />
<node cityID="156" cityName="n56" X="10" Y="45" />
<node cityID="157" cityName="n57" X="7" Y="30" />
<node cityID="158" cityName="n58" X="31" Y="11" />
<node cityID="159" cityName="n59" X="36" Y="93" />
<node cityID="160" cityName="n60" X="50" Y="31" />
<node cityID="161" cityName="n61" X="49" Y="52" />
<node cityID="162" cityName="n62" X="39" Y="10" />
<node cityID="163" cityName="n63" X="76" Y="40" />
<node cityID="164" cityName="n64" X="83" Y="34" />
<node cityID="165" cityName="n65" X="33" Y="51" />
<node cityID="166" cityName="n66" X="0" Y="15" />
<node cityID="167" cityName="n67" X="52" Y="82" />
<node cityID="168" cityName="n68" X="52" Y="82" />
<node cityID="169" cityName="n69" X="46" Y="6" />
<node cityID="170" cityName="n70" X="3" Y="26" />
<node cityID="171" cityName="n71" X="46" Y="80" />
<node cityID="172" cityName="n72" X="94" Y="30" />
<node cityID="173" cityName="n73" X="26" Y="76" />
<node cityID="174" cityName="n74" X="75" Y="92" />
<node cityID="175" cityName="n75" X="57" Y="51" />
<node cityID="176" cityName="n76" X="34" Y="21" />
<node cityID="177" cityName="n77" X="28" Y="80" />
<node cityID="178" cityName="n78" X="59" Y="66" />
<node cityID="179" cityName="n79" X="51" Y="16" />
<node cityID="180" cityName="n80" X="87" Y="11" />
</nodes>
<edges method="GENERATE" distance="EUCLIDIAN" triangleInequality="true" symmetric="true" >
  <!-- edges will be automatically generated -->
```



```
</edges>
<order orderDescription="test" depotLocation="101">
<!-- <orderItem orderLine="1001" lineDescr="order 1001" destination="101" weight="0" volume="0" /> -->
<orderItem orderLine="1002" lineDescr="order 1002" destination="102" weight="0" volume="24" />
<orderItem orderLine="1003" lineDescr="order 1003" destination="103" weight="0" volume="22" />
<orderItem orderLine="1004" lineDescr="order 1004" destination="104" weight="0" volume="23" />
<orderItem orderLine="1005" lineDescr="order 1005" destination="105" weight="0" volume="5" />
<orderItem orderLine="1006" lineDescr="order 1006" destination="106" weight="0" volume="11" />
<orderItem orderLine="1007" lineDescr="order 1007" destination="107" weight="0" volume="23" />
<orderItem orderLine="1008" lineDescr="order 1008" destination="108" weight="0" volume="26" />
<orderItem orderLine="1009" lineDescr="order 1009" destination="109" weight="0" volume="9" />
<orderItem orderLine="1010" lineDescr="order 1010" destination="110" weight="0" volume="23" />
<orderItem orderLine="1011" lineDescr="order 1011" destination="111" weight="0" volume="9" />
<orderItem orderLine="1012" lineDescr="order 1012" destination="112" weight="0" volume="14" />
<orderItem orderLine="1013" lineDescr="order 1013" destination="113" weight="0" volume="16" />
<orderItem orderLine="1014" lineDescr="order 1014" destination="114" weight="0" volume="12" />
<orderItem orderLine="1015" lineDescr="order 1015" destination="115" weight="0" volume="2" />
<orderItem orderLine="1016" lineDescr="order 1016" destination="116" weight="0" volume="2" />
<orderItem orderLine="1017" lineDescr="order 1017" destination="117" weight="0" volume="6" />
<orderItem orderLine="1018" lineDescr="order 1018" destination="118" weight="0" volume="20" />
<orderItem orderLine="1019" lineDescr="order 1019" destination="119" weight="0" volume="26" />
<orderItem orderLine="1020" lineDescr="order 1020" destination="120" weight="0" volume="12" />
<orderItem orderLine="1021" lineDescr="order 1021" destination="121" weight="0" volume="15" />
<orderItem orderLine="1022" lineDescr="order 1022" destination="122" weight="0" volume="13" />
<orderItem orderLine="1023" lineDescr="order 1023" destination="123" weight="0" volume="26" />
<orderItem orderLine="1024" lineDescr="order 1024" destination="124" weight="0" volume="17" />
<orderItem orderLine="1025" lineDescr="order 1025" destination="125" weight="0" volume="7" />
<orderItem orderLine="1026" lineDescr="order 1026" destination="126" weight="0" volume="12" />
<orderItem orderLine="1027" lineDescr="order 1027" destination="127" weight="0" volume="4" />
<orderItem orderLine="1028" lineDescr="order 1028" destination="128" weight="0" volume="4" />
<orderItem orderLine="1028" lineDescr="order 1029" destination="129" weight="0" volume="20" />
<orderItem orderLine="1030" lineDescr="order 1030" destination="130" weight="0" volume="10" />
<orderItem orderLine="1031" lineDescr="order 1031" destination="131" weight="0" volume="9" />
<orderItem orderLine="1032" lineDescr="order 1032" destination="132" weight="0" volume="2" />
<orderItem orderLine="1033" lineDescr="order 1033" destination="133" weight="0" volume="9" />
<orderItem orderLine="1034" lineDescr="order 1034" destination="134" weight="0" volume="1" />
<orderItem orderLine="1035" lineDescr="order 1035" destination="135" weight="0" volume="2" />
<orderItem orderLine="1036" lineDescr="order 1036" destination="136" weight="0" volume="2" />
<orderItem orderLine="1037" lineDescr="order 1037" destination="137" weight="0" volume="12" />
<orderItem orderLine="1038" lineDescr="order 1038" destination="138" weight="0" volume="14" />
```



```
<orderItem orderLine="1039" lineDescr="order 1039" destination="139" weight="0" volume="23" />
<orderItem orderLine="1040" lineDescr="order 1040" destination="140" weight="0" volume="21" />
<orderItem orderLine="1041" lineDescr="order 1041" destination="141" weight="0" volume="13" />
<orderItem orderLine="1042" lineDescr="order 1042" destination="142" weight="0" volume="13" />
<orderItem orderLine="1043" lineDescr="order 1043" destination="143" weight="0" volume="23" />
<orderItem orderLine="1044" lineDescr="order 1044" destination="144" weight="0" volume="3" />
<orderItem orderLine="1045" lineDescr="order 1045" destination="145" weight="0" volume="6" />
<orderItem orderLine="1046" lineDescr="order 1046" destination="146" weight="0" volume="23" />
<orderItem orderLine="1047" lineDescr="order 1047" destination="147" weight="0" volume="11" />
<orderItem orderLine="1048" lineDescr="order 1048" destination="148" weight="0" volume="2" />
<orderItem orderLine="1049" lineDescr="order 1049" destination="149" weight="0" volume="7" />
<orderItem orderLine="1050" lineDescr="order 1050" destination="150" weight="0" volume="13" />
<orderItem orderLine="1051" lineDescr="order 1051" destination="151" weight="0" volume="10" />
<orderItem orderLine="1052" lineDescr="order 1052" destination="152" weight="0" volume="3" />
<orderItem orderLine="1053" lineDescr="order 1053" destination="153" weight="0" volume="6" />
<orderItem orderLine="1054" lineDescr="order 1054" destination="154" weight="0" volume="13" />
<orderItem orderLine="1055" lineDescr="order 1055" destination="155" weight="0" volume="2" />
<orderItem orderLine="1056" lineDescr="order 1056" destination="156" weight="0" volume="14" />
<orderItem orderLine="1057" lineDescr="order 1057" destination="157" weight="0" volume="7" />
<orderItem orderLine="1058" lineDescr="order 1058" destination="158" weight="0" volume="21" />
<orderItem orderLine="1059" lineDescr="order 1059" destination="159" weight="0" volume="7" />
<orderItem orderLine="1060" lineDescr="order 1060" destination="160" weight="0" volume="22" />
<orderItem orderLine="1061" lineDescr="order 1061" destination="161" weight="0" volume="13" />
<orderItem orderLine="1062" lineDescr="order 1062" destination="162" weight="0" volume="22" />
<orderItem orderLine="1063" lineDescr="order 1063" destination="163" weight="0" volume="18" />
<orderItem orderLine="1064" lineDescr="order 1064" destination="164" weight="0" volume="22" />
<orderItem orderLine="1065" lineDescr="order 1065" destination="165" weight="0" volume="6" />
<orderItem orderLine="1066" lineDescr="order 1066" destination="166" weight="0" volume="2" />
<orderItem orderLine="1067" lineDescr="order 1067" destination="167" weight="0" volume="11" />
<orderItem orderLine="1068" lineDescr="order 1068" destination="168" weight="0" volume="5" />
<orderItem orderLine="1069" lineDescr="order 1069" destination="169" weight="0" volume="9" />
<orderItem orderLine="1071" lineDescr="order 1071" destination="170" weight="0" volume="9" />
<orderItem orderLine="1071" lineDescr="order 1071" destination="171" weight="0" volume="5" />
<orderItem orderLine="1072" lineDescr="order 1072" destination="172" weight="0" volume="12" />
<orderItem orderLine="1073" lineDescr="order 1073" destination="173" weight="0" volume="2" />
<orderItem orderLine="1074" lineDescr="order 1074" destination="174" weight="0" volume="12" />
<orderItem orderLine="1075" lineDescr="order 1075" destination="175" weight="0" volume="19" />
<orderItem orderLine="1076" lineDescr="order 1076" destination="176" weight="0" volume="6" />
<orderItem orderLine="1077" lineDescr="order 1077" destination="177" weight="0" volume="14" />
<orderItem orderLine="1078" lineDescr="order 1078" destination="178" weight="0" volume="2" />
```



```
<orderItem orderLine="1079" lineDescr="order 1079" destination="179" weight="0" volume="2" />
<orderItem orderLine="1080" lineDescr="order 1080" destination="180" weight="0" volume="24" />
</order>
<resources>
  <vehicle vehicleID="2001" vehicleDescr="VF-01-01" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2002" vehicleDescr="VF-02-02" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2003" vehicleDescr="VF-03-03" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2004" vehicleDescr="VF-04-04" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2005" vehicleDescr="VF-05-05" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2006" vehicleDescr="VF-06-06" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2007" vehicleDescr="VF-07-07" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2008" vehicleDescr="VF-08-08" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2009" vehicleDescr="VF-09-09" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2010" vehicleDescr="VF-10-10" maxWeight="999" maxVolume="100" maxDistance="100000" />
</resources>
</CAPVRPInstance>
<CAPVRPParameters>
  <generation populationCreation="SINGLEDEPOT_SINGLECAPACITY"
    populationSize="200"
    generationsPerStep="13"
    maxGenerations="65000"
    utility="TOTALDISTANCE"
    operatorClass="CAPVRP" />
  <selectionOperator parentSelectionStrategy="TOURNAMENT"
    fractionStochasticInMixed="0.2"
    parentSetSize="4"
    replacementStrategy="ELITIST" />
  <variationOperator crossoverRate="0.75"
    mutationSwapRate="0.05"
    mutationInversionRate="0.15"
    mutationInsertionRate="0.05"
    mutationDisplacementRate="0.2" />
  <monitors>
    <monitor type="INCREMENT_MONITOR" />
    <monitor type="SAMPLE_MONITOR" />
  </monitors>
</CAPVRPParameters>
</CAPVRP>
```



The instance file used for experimentation based on Augerat A32k5 instance

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE CAPVRP SYSTEM "file:///home/phoenix/phoenix-repos/trunk/app/capVRPGui/data/xml/phoenix.dtd" >
<CAPVRP>
  <CAPVRPInstance>
    <instanceIdent instanceID="Augerat e.a." setID="A-n32-k5" />
    <nodes coordinates="EUCLIDIAN" >
      <node cityID="101" cityName="n1" X="82" Y="76" />
      <node cityID="102" cityName="n2" X="96" Y="44" />
      <node cityID="103" cityName="n3" X="50" Y="5" />
      <node cityID="104" cityName="n4" X="49" Y="8" />
      <node cityID="105" cityName="n5" X="13" Y="7" />
      <node cityID="106" cityName="n6" X="29" Y="89" />
      <node cityID="107" cityName="n7" X="58" Y="30" />
      <node cityID="108" cityName="n8" X="84" Y="39" />
      <node cityID="109" cityName="n9" X="14" Y="24" />
      <node cityID="110" cityName="n10" X="2" Y="39" />
      <node cityID="111" cityName="n11" X="3" Y="82" />
      <node cityID="112" cityName="n12" X="5" Y="10" />
      <node cityID="113" cityName="n13" X="98" Y="52" />
      <node cityID="114" cityName="n14" X="84" Y="25" />
      <node cityID="115" cityName="n15" X="61" Y="59" />
      <node cityID="116" cityName="n16" X="1" Y="65" />
      <node cityID="117" cityName="n17" X="88" Y="51" />
      <node cityID="118" cityName="n18" X="91" Y="2" />
      <node cityID="119" cityName="n19" X="19" Y="32" />
      <node cityID="120" cityName="n20" X="93" Y="3" />
      <node cityID="121" cityName="n21" X="50" Y="93" />
      <node cityID="122" cityName="n22" X="98" Y="14" />
      <node cityID="123" cityName="n23" X="5" Y="42" />
      <node cityID="124" cityName="n24" X="42" Y="9" />
      <node cityID="125" cityName="n25" X="61" Y="62" />
      <node cityID="126" cityName="n26" X="9" Y="97" />
      <node cityID="127" cityName="n27" X="80" Y="55" />
      <node cityID="128" cityName="n28" X="57" Y="69" />
      <node cityID="129" cityName="n29" X="23" Y="15" />
      <node cityID="130" cityName="n30" X="20" Y="70" />
    </nodes>
  </CAPVRPInstance>
</CAPVRP>
```



```
<node cityID="131" cityName="n31" X="85" Y="60" />
<node cityID="132" cityName="n32" X="98" Y="5" />
</nodes>
<edges method="GENERATE" distance="EUCLIDIAN" triangleInequality="true" symmetric="true" >
  <!-- edges will be automatically generated -->
</edges>
<order orderDescription="test" depotLocation="101">
  <!--
    <orderItem orderLine="1001" lineDescr="order 1001" destination="101" weight="0" volume="0" /> -->
    <orderItem orderLine="1002" lineDescr="order 1002" destination="102" weight="0" volume="19" />
    <orderItem orderLine="1003" lineDescr="order 1003" destination="103" weight="0" volume="21" />
    <orderItem orderLine="1004" lineDescr="order 1004" destination="104" weight="0" volume="6" />
    <orderItem orderLine="1005" lineDescr="order 1005" destination="105" weight="0" volume="19" />
    <orderItem orderLine="1006" lineDescr="order 1006" destination="106" weight="0" volume="7" />
    <orderItem orderLine="1007" lineDescr="order 1007" destination="107" weight="0" volume="12" />
    <orderItem orderLine="1008" lineDescr="order 1008" destination="108" weight="0" volume="16" />
    <orderItem orderLine="1009" lineDescr="order 1009" destination="109" weight="0" volume="6" />
    <orderItem orderLine="1010" lineDescr="order 1010" destination="110" weight="0" volume="16" />
    <orderItem orderLine="1011" lineDescr="order 1011" destination="111" weight="0" volume="8" />
    <orderItem orderLine="1012" lineDescr="order 1012" destination="112" weight="0" volume="14" />
    <orderItem orderLine="1013" lineDescr="order 1013" destination="113" weight="0" volume="21" />
    <orderItem orderLine="1014" lineDescr="order 1014" destination="114" weight="0" volume="16" />
    <orderItem orderLine="1015" lineDescr="order 1015" destination="115" weight="0" volume="3" />
    <orderItem orderLine="1016" lineDescr="order 1016" destination="116" weight="0" volume="22" />
    <orderItem orderLine="1017" lineDescr="order 1017" destination="117" weight="0" volume="18" />
    <orderItem orderLine="1018" lineDescr="order 1018" destination="118" weight="0" volume="19" />
    <orderItem orderLine="1019" lineDescr="order 1019" destination="119" weight="0" volume="1" />
    <orderItem orderLine="1020" lineDescr="order 1020" destination="120" weight="0" volume="24" />
    <orderItem orderLine="1021" lineDescr="order 1021" destination="121" weight="0" volume="8" />
    <orderItem orderLine="1022" lineDescr="order 1022" destination="122" weight="0" volume="12" />
    <orderItem orderLine="1023" lineDescr="order 1023" destination="123" weight="0" volume="4" />
    <orderItem orderLine="1024" lineDescr="order 1024" destination="124" weight="0" volume="8" />
    <orderItem orderLine="1025" lineDescr="order 1025" destination="125" weight="0" volume="24" />
    <orderItem orderLine="1026" lineDescr="order 1026" destination="126" weight="0" volume="24" />
    <orderItem orderLine="1027" lineDescr="order 1027" destination="127" weight="0" volume="2" />
    <orderItem orderLine="1028" lineDescr="order 1028" destination="128" weight="0" volume="20" />
    <orderItem orderLine="1028" lineDescr="order 1029" destination="129" weight="0" volume="15" />
    <orderItem orderLine="1030" lineDescr="order 1030" destination="130" weight="0" volume="2" />
    <orderItem orderLine="1031" lineDescr="order 1031" destination="131" weight="0" volume="14" />
    <orderItem orderLine="1031" lineDescr="order 1032" destination="132" weight="0" volume="9" />
  </order>
```



```
<resources>
  <vehicle vehicleID="2001" vehicleDescr="VF-01-01" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2002" vehicleDescr="VF-02-02" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2003" vehicleDescr="VF-03-03" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2004" vehicleDescr="VF-04-04" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2005" vehicleDescr="VF-05-05" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2006" vehicleDescr="VF-06-06" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2007" vehicleDescr="VF-07-07" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2008" vehicleDescr="VF-08-08" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2009" vehicleDescr="VF-09-09" maxWeight="999" maxVolume="100" maxDistance="100000" />
  <vehicle vehicleID="2010" vehicleDescr="VF-10-10" maxWeight="999" maxVolume="100" maxDistance="100000" />
</resources>
</CAPVRPInstance>
<CAPVRPParameters>
  <generation populationCreation="SINGLEDEPOT_SINGLECAPACITY"
    populationSize="200"
    generationsPerStep="13"
    maxGenerations="65000"
    utility="TOTALDISTANCE"
    operatorClass="CAPVRP" />
  <selectionOperator parentSelectionStrategy="TOURNAMENT"
    fractionStochasticInMixed="0.2"
    parentSetSize="4"
    replacementStrategy="ELITIST" />
  <variationOperator crossoverRate="0.75"
    mutationSwapRate="0.05"
    mutationInversionRate="0.15"
    mutationInsertionRate="0.05"
    mutationDisplacementRate="0.2" />
  <monitors>
    <monitor type="INCREMENT_MONITOR" />
    <monitor type="SAMPLE_MONITOR" />
  </monitors>
</CAPVRPParameters>
</CAPVRP>
```



13.3 Augerat A80k10 Instance

NAME : A-n80-k10
COMMENT : (Augerat et al, Min no of trucks: 10, Best value: 1764)
TYPE : CVRP
DIMENSION : 80
EDGE_WEIGHT_TYPE : EUC_2D
CAPACITY : 100
NODE_COORD_SECTION

node	X	Y																					
1	92	92	20	21	19	39	41	95	58	31	11	77	28	80	13	16	32	2	51	10	70	9	
2	88	58	21	36	21	40	48	54	59	36	93	78	59	66	14	12	33	9	52	3	71	5	
3	70	6	22	100	61	41	98	73	60	50	31	79	51	16	15	2	34	1	53	6	72	12	
4	57	59	23	11	85	42	26	38	61	49	52	80	87	11	16	2	35	2	54	13	73	2	
5	0	98	24	69	35	43	69	76	62	39	10				17	6	36	2	55	2	74	12	
6	61	38	25	69	22	44	40	1	63	76	40	DEMAND_SECTION	18	20			37	12	56	14	75	19	
7	65	22	26	29	35	45	65	41	64	83	34	node capacity	19	26			38	14	57	7	76	6	
8	91	52	27	14	9	46	14	86	65	33	51	1	0	20	12			39	23	58	21	77	14
9	59	2	28	50	33	47	32	39	66	0	15	2	24	21	15			40	21	59	7	78	2
10	3	54	29	89	17	48	14	24	67	52	82	3	22	22	13			41	13	60	22	79	2
11	95	38	30	57	44	49	96	5	68	52	82	4	23	23	26			42	13	61	13	80	24
12	80	28	31	60	25	50	82	98	69	46	6	5	5	24	17			43	23	62	22		
13	66	42	32	48	42	51	23	85	70	3	26	6	11	25	7			44	3	63	18	DEPOT_SECTION	
14	79	74	33	17	93	52	63	69	71	46	80	7	23	26	12			45	6	64	22	1	
15	99	25	34	21	50	53	87	19	72	94	30	8	26	27	4			46	23	65	6	-1	
16	20	43	35	77	18	54	56	75	73	26	76	9	9	28	4			47	11	66	2	EOF	
17	40	3	36	2	4	55	15	63	74	75	92	10	23	29	20			48	2	67	11		
18	50	42	37	63	83	56	10	45	75	57	51	11	9	30	10			49	7	68	5		
19	97	0	38	68	6	57	7	30	76	34	21	12	14	31	9			50	13	69	9		



13.4 Augerat A32k5 Instance

```

NAME : A-n32-k5
COMMENT : (Augerat et al, Min no of trucks: 5, Optimal value: 784)
TYPE : CVRP
DIMENSION : 32
EDGE_WEIGHT_TYPE : EUC_2D
CAPACITY : 100
NODE_COORD_SECT 6 29 89      15 61 59      24 42 9      DEMAND_SECTION 8 16      17 18      26 24      -1
ION              7 58 30      16 1 65      25 61 62     node capacity 9 6      18 19      27 2      EOF
node X Y         8 84 39      17 88 51     26 9 97      1 0      10 16      19 1      28 20
  1 82 76        9 14 24      18 91 2      27 80 55     2 19      11 8      20 24      29 15
  2 96 44        10 2 39      19 19 32     28 57 69     3 21      12 14      21 8      30 2
  3 50 5         11 3 82      20 93 3      29 23 15     4 6      13 21      22 12      31 14
  4 49 8         12 5 10      21 50 93     30 20 70     5 19      14 16      23 4      32 9
  5 13 7         13 98 52     22 98 14     31 85 60     6 7      15 3      24 8      DEPOT_SECTION
                14 84 25     23 5 42      32 98 5      7 12      16 22      25 24      1

```



This page is intentionally left blank.



14 Appendix-B – Companion CD-Rom

14.1 Complete Software API Documentation Including Source Code

14.2 Logger Files And Analysis Spreadsheets

14.3 Used Instance Files (XML)

The content of these appendices is supplied separately on the Companion CD-Rom.



This page is intentionally left blank



15 Appendix-C – The CapVRP Problem Revisited – A Paper

The paper is supplied separately