A survey of optimization methods for solving the inverse shortest path routing problem

Richard Sandberg

LiTH - MAT - EX - - 2010 / 26 - - SE
A survey of optimization methods for solving the inverse shortest path routing problem

Applied Mathematics, Linköpings Universitet

Richard Sandberg

LiTH - MAT - EX - 2010 / 26 - SE

Examensarbete: 30 hp

Level: D

Supervisor: Mikael Call,
Applied Mathematics, Linköpings Universitet

Examiner: Kaj Holmberg,
Applied Mathematics, Linköpings Universitet

Linköping: Augusti 2010
A survey of optimization methods for solving the inverse shortest path routing problem

Richard Sandberg

The routing of traffic in IP networks is often done with a set of weights that determine which way the traffic will go (shortest path routing). The problem here is to determine if there exists a set of weights for a desired routing pattern. This thesis will investigate the performance of a number of different models and solvers for solving this type of problem which is usually called the inverse shortest path routing (ISPR) problem. The models tested are the same as described in [1]. The different solvers used are mainly the linear CPLEX solvers but also a few multi commodity network solvers.

The tests showed that there is a big performance difference between the models and solvers and that the cycle bases model solved with the CPLEX dualopt solver was the fastest overall.

ISPR, WFP, SPR, OSPF, MMCF, Graph, Network, Optimization, Routing
Abstract

The routing of traffic in IP networks is often done with a set of weights that determine which way the traffic will go (shortest path routing). The problem here is to determine if there exists a set of weights for a desired routing pattern. This thesis will investigate the performance of a number of different models and solvers for solving this type of problem which is usually called the inverse shortest path routing (ISPR) problem. The models tested are the same as described in [1]. The different solvers used are mainly the linear CPLEX solvers but also a few multi commodity network solvers.

The tests showed that there is a big performance difference between the models and solvers and that the cycle bases model solved with the CPLEX dualopt solver was the fastest overall.

Keywords: ISPR, WFP, SPR, OSPF, MMCF, Graph, Network, Optimization, Routing

Abstract in Swedish: Sammanfattning

Ruttningen av trafik i IP-nätverk sker ofta med hjälp av bågvikter som bestämmer vilken väg trafiken tar (kortastevägruttning). Problemet här är att avgöra ifall det existerar en uppsättning vikter givet ett önskat ruttningsschema. Den här rapporten undersöker prestandan hos ett antal modeller och optimeringsprogram avsedda att lösa denna typ av problem som ofta kallas inversa kortastevägruttningsproblemet.

Undersökningen visar att det existerar en stor skillnad mellan modellerna och optimeringsprogrammen och att modellen baserad på cykelbaser löst med CPLEX dualopt lösaren är snabbast.
Acknowledgements

First of all I would like to thank my examiner Kaj Holmberg for giving me the opportunity to write this thesis at MAI.

Secondly, I would like to give special thanks to my supervisor Mikael Call for all the help given and great patience shown throughout this thesis. He was always in good spirit and was able to explain difficult concepts in a clear way which helped to keep this thesis work both interesting and fun.

My opponent, Anders Olofsson also deserves my thanks for giving very detailed and constructive suggestions for improving the report.

I would also like to give thanks to my wife for all the support and great tolerance shown, especially considering the many late nights I sat in front of a computer.
Nomenclature

Most of the reoccurring abbreviations and symbols are described here.

Symbols

\(A\) \hspace{1cm} \text{Multiset of all arcs in the network, including added distance arcs.}
\(A_l\) \hspace{1cm} \text{Set of arcs that must be in a shortest path to } l.
\(\bar{A}_l\) \hspace{1cm} \text{Set of arcs that must not be in a shortest path to } l.
\(U_l\) \hspace{1cm} \text{All other arcs not in } A_l \text{ nor in } \bar{A}_l.
\(T_l\) \hspace{1cm} \text{Set of arcs in a spanning SP-tree to } l.
\(N_l\) \hspace{1cm} \text{Set of arcs in } A_l \setminus T_l.

Abbreviations

MMCF \hspace{1cm} \text{Multicommodity Minimum Cost Flow}
SPG \hspace{1cm} \text{Shortest Path Graph}
ISPR \hspace{1cm} \text{Inverse Shortest Path Routing}
CPLEX \hspace{1cm} \text{C simPLEX, commercial optimization software from IBM}
IGP \hspace{1cm} \text{Interior Gateway Protocol}
OSPF \hspace{1cm} \text{Open Shortest Path First}
IS-IS \hspace{1cm} \text{Intermediate System to Intermediate System}
RIP \hspace{1cm} \text{Routing Information Protocol}
ECMP \hspace{1cm} \text{Equal Cost Multi Path}
WFP \hspace{1cm} \text{Weight Finding Problem}
## Contents

1 **Introduction** 1
   1.1 Background to Inverse Shortest Path Routing 1
   1.2 Objective of the thesis 2
   1.3 Related work 2
   1.4 Topics covered 2

2 **Graph theory and optimization theory** 5
   2.1 Graph theory 5
   2.1.1 Network Flow 6
   2.2 Description of Shortest Path Graphs 7
   2.3 Cycle bases 8

3 **Mathematical Models and Algorithms** 11
   3.1 P1. The natural formulation 11
   3.2 P2. The Dual formulation of P1 12
   3.3 P3. The Dual formulation of P1 with variable transformation 13
   3.4 P4. The Cycle basis formulation 13
   3.5 P5. The Cycle basis formulation simplified 14
   3.6 Valid cycles 14
   3.6.1 Finding valid cycles 16

4 **Method used in performing the study** 19
   4.1 CPLEX optimization software 19
   4.2 MMCF Solvers 20
   4.3 The instance generator 20

5 **Results of the tests** 23
   5.1 Abbreviations 23
   5.2 Problems encountered 24
   5.3 Preliminary test 24
   5.4 Extensive test 30

6 **Analysis of the results** 39
   6.1 Preliminary test 39
   6.2 Extensive test 40

Sandberg, 2010. xiii
6.3 Valid Cycle search ........................................... 40
6.4 Different problem types .................................... 40

7 Summary and conclusion ...................................... 41
  7.1 Summary .................................................. 41
  7.2 Limitations of this study .................................. 41
  7.3 Future work ............................................... 42

A The SPG file format ........................................... 45
  A.1 Overview .................................................. 45
  A.2 Sample file ................................................ 45

B The main program ............................................. 47
  B.1 General description ....................................... 47
  B.2 Commandline parameters ................................ 47
Chapter 1

Introduction

This chapter describes the ISPR problem and how it applies to IP networks. It also specifies the purpose of this thesis and gives an outline of the chapters to follow.

1.1 Background to Inverse Shortest Path Routing

A common method for routing traffic in an IP network is to assign to each arc in the network a weight and let all traffic take the shortest path from each origin to each destination with respect to these weights, this is referred to as Shortest Path Routing (SPR). The shortest path between two nodes is the same as the path with the minimal sum of weights of the path’s constituent arcs. Routing protocols that use SPR are e.g. OSPF, IS-IS and RIP, a class of protocols in the Interior Gateway Protocol (IGP) family. The weights used in the OSPF protocol must be integral and between 1 and 65535 and all the algorithms we study here will also assume that the weights are greater than or equal to one.

The only way for a network administrator to control the routing of traffic is

Figure 1.1: An overview of the network design process.
by assigning these weights in a manner that yields the desired routing pattern. This is far from trivial to do, and in some cases there is no possible set of weights that will induce the requested routing pattern. The problem of finding a compatible set of weights given a set of shortest paths is called Inverse Shortest Path Routing (ISPR) or sometimes it’s called the Weight Finding Problem (WFP).

The problem of finding a compatible set of weights is only one part in the network design process as seen in Figure 1.1. It is the only part of the process that this thesis will deal with.

One complication is what to do if there are multiple shortest paths between an origin and a destination. OSPF doesn’t specify what to do in this case, but a common assumption made is that the traffic is split evenly, i.e. if at a node there are several outgoing arcs that are in a shortest path to the destination the flow out of this node will be split evenly on these arcs, this is called Equal Cost Multi Path (ECMP). Some models explicitly restrict themselves to the unique shortest path case, all the models in this thesis however allow multiple shortest paths.

### 1.2 Objective of the thesis

The objective of this thesis is to implement and compare the performance of known algorithms that deal with ISPR-problems in order to determine which are useful for further study and/or improvement. For details and a more rigorous treatment of the theory behind the algorithms and mathematical models used, please see [1, 2, 3].

### 1.3 Related work

The most important work for this thesis is [1] which gives a detailed description of the complete network design problem and a theoretical foundation for the mathematical models used for the Inverese Shortest Path Routing problem, the models which are tested in this thesis.

Earlier but related work that touches on different aspects of Inverse Shortest Path Routing are [2, 3].

### 1.4 Topics covered

There are seven chapters and two appendixes. The topics dealt with are:

**Chapter 1: Introduction.** This chapter.

**Chapter 2: Graph theory and optimization theory.** A brief introduction to the theory used in the models examined.
Chapter 3: Mathematical Models and Algorithms. A description of the models implemented and tested.

Chapter 4: Method used in performing the study. Here we describe how the tests were carried out and what software was used.

Chapter 5: Results of the tests. Contains tables and diagrams of the results of the survey.

Chapter 6: Analysis of the results. We explain some of the interesting parts from the results.

Chapter 7: Summary and conclusion. We summarize our findings and also point out some weaknesses of the study.

Appendix A: The SPG file format. We give a detailed description of the file format used for storing the SPGs.

Appendix B: The main program. Here we describe how the main program used for all the tests works and what its features are.
Chapter 1. Introduction
Graph theory and optimization theory

This chapter will give an overview of graph theory and optimization theory, especially as related to shortest-path problems.

2.1 Graph theory

A directed graph (digraph) consists of a set of nodes (vertices) $\mathcal{N} = \{1, 2, \ldots, m\}$ and a set of arcs $\mathcal{A} = \{(i, j), (k, l), \ldots, (s, t)\}$ joining pairs of nodes in $\mathcal{N}$ where arc $(i, j)$ is directed from node $i$ to node $j$.

A path from node $x_0$ to a node $x_n$ is a sequence of nodes $x_0, x_1, \ldots, x_n$ where $\forall i : 0 \leq i \leq n - 1 \exists (x_i, x_{i+1}) \in \mathcal{A} \lor (x_{i+1}, x_i) \in \mathcal{A}$. A path is directed if $\forall i : 0 \leq i \leq n - 1 \exists (x_i, x_{i+1}) \in \mathcal{A}$.

![Figure 2.1: A graph with a path from node 1 to 4 and a directed path from node 4 to 2.](image)

A path that starts and ends in the same node is called a cycle, and a directed path that starts and ends in the same node is called a directed cycle. If there
is a path between every pair of nodes in the network we say that the graph is connected. If there is a directed path between every pair of nodes in the network we say that the graph is strongly connected. The graph in Figure 2.1 is connected but not strongly connected. A connected component of a graph $G$ is a maximal connected subgraph of $G$. Similarly, a strongly connected component of $G$ is a maximal strongly connected subgraph of $G$. Finally, a connected graph that contains no cycles is called a tree.

![Graph and connected components](image.png)

**Figure 2.2:** A graph to the left and its strongly connected components to the right.

For simplicity when we refer to a path in a graph we mean a directed path unless otherwise stated. We will however maintain the distinction between a cycle and a directed cycle as defined above. In addition we will in the following describe a path simply by the set of arcs that makes up the path.

A network is a graph where each arc is associated with a weight (cost), in our case the weight is a real number greater than or equal to one. In a network the length of a path is simply the sum of the weights of each arc that is included in the path.

### 2.1.1 Network Flow

Many problems can be modeled by studying the flow in a network. Each arc in the network is now associated with a flow variable saying how many units of flow that flows on the arc as well as the cost of sending one unit of flow on the arc. Each arc may also have a capacity constraint limiting how many units that can flow on the arc. Each node will also have a balance constraint specifying how many units that should flow to or from that node in total. Usually the objective is to minimize the total cost of the flow sent in the network and then it’s called a Minimum Cost Flow (MCF) problem. By extension we can have several commodities that will be sent in the network, each arc now having a flow variable for each commodity and each node having a specified supply/demand for each commodity. This is called a Multicommodity Minimum Cost Flow (MMCF) problem.
Using MCF to solve the shortest path problem

Finding the shortest path between a pair of nodes in a network is a classic problem that can be solved in a time proportional to \(|N|^2\) with Dijkstra’s algorithm. The problem can also be formulated as a minimum cost flow problem with the start node having a supply of 1 and the end node having a demand of 1, all other nodes just being transit nodes with the same flow into as the flow out of the node. The total cost of sending 1 unit from s to t will be the sum of the cost for each arc used which is the same as the length of the path taken. The mathematical model takes the form:

\[
\begin{align*}
\min & \sum_{(i,j) \in A} w_{ij} x_{ij} \\
\text{s.t.} & \quad \sum_{j : (j,i) \in A} x_{ij} - \sum_{j : (i,j) \in A} x_{ji} = \begin{cases}
-1, & i = s \\
1, & i = t \\
0, & \text{otherwise}
\end{cases}, \quad i \in N \\
x_{ij} & \geq 0 \quad (i,j) \in A
\end{align*}
\] (2.1)

An interesting aspect of Model (2.1) is that it is integral i.e. the optimal solution will have all the flow variables equal to 0 or 1, and the arcs with flow 1 will form a path from s to t, see [4, 5] for details. Also, this model is a linear program which means that the duality theory applies and we can take a look at how the dual problem looks.

\[
\begin{align*}
\max & \quad \pi_t - \pi_s \\
\text{s.t.} & \quad \pi_j - \pi_i \leq w_{ij}, \quad (i,j) \in A
\end{align*}
\] (2.2)

The variable \(\pi_i\) corresponds to the balance condition for node \(i\) in the primal and is called the node potential of node \(i\). Any feasible solution has

\[\hat{w}_{ij} \equiv w_{ij} + \pi_i - \pi_j \geq 0\]

where \(\hat{w}_{ij}\) is called the reduced cost of arc \((i,j)\). The reduced cost will be 0 for an arc in a shortest path and \(\geq 1\) for an arc not in a shortest path. These equations will be used as a starting point when solving ISPR problems using SP-graphs later on, with the added complexity that we will have multiple destinations/commodities that must be solved simultaneously.

2.2 Description of Shortest Path Graphs

An SP-graph is a subgraph of a network and is a compact way of describing the desired routing pattern in the network, i.e. which arcs should be shortest-path arcs and which should not. Starting from a network \(G = (N, A)\) with nodes \(N\) and arcs \(A\) an SP-graph contains all the nodes in \(N\) and a subset of the arcs in \(A\) plus a set of distance arcs. One of the nodes is designated as the destination node \(l\) and each arc \((i,j)\) in the SP-graph is labeled in exactly one of the following ways:
Chapter 2. Graph theory and optimization theory

A distance arc is a fictitious arc added from every node \( i \) that has no out arc labeled \( A_l \), from \( i \) to \( l \). This arc represents some shortest path to \( l \) and it makes sure that a shortest-path is specified from every node to the destination which is necessary for the mathematical models that follow. This also means that \( A_l \cup D_l \) will be a spanning in-graph to the root node \( l \). Since each arc in \( D_l \) once added is treated the same as an arc in \( A_l \) we may as well let all \( D_l \) arcs be labeled \( A_l \) instead. We will use this simplification from now on. Of note is that after adding the distance arcs to the arcset \( A_l \), it becomes a multiset, i.e. there may be two identical arcs between the same pair of nodes.

Here follows a simple example of an SP-graph. We start out with the network to the left, we then mark the destination for this SP-graph node 2 and label the arcs that must be shortest-path arcs, arc \((4, 3)\), and which that must not \((4, 1)\) with respect to this destination. This is shown in the middle figure. Finally we add the distance arcs so that there is a shortest-path arc leaving every node (except the destination node). This is an example of a complete SP-graph that can be used in the mathematical models that follow. Later on we will see examples of multiple SP-graphs drawn in the same figure.

![Figure 2.3: The underlying network to the left. In the middle a shortest path graph with destination node 2 (thick), \( A_2 = \{(4, 3)\} \) (thick), \( \bar{A}_2 = \{(4, 1)\} \) (dashed),the rest of the arcs \( \in \ U_2 \). To the right, the shortest path graph completed with distance arcs \( D_2 = \{(1, 2), (3, 2)\} \) (thick).]

2.3 Cycle bases

A couple of the models we will use employ the use of cycle bases so a short explanation of the theory behind them is in order.

Let \( G = (N, A) \) be a strongly connected graph with \( m \) arcs and \( n \) nodes, and let \( C \) be an oriented circuit (simple cycle) in the graph. Assign the value 1 to each arc in the circuit that is directed forwards and -1 to each arc directed backwards and 0 to all other arcs in the graph. By reversing the direction of all the arcs with value -1 we obtain a simple directed cycle. To each circuit in the graph we can create a vector with the dimension \( m \) with coordinate values assigned as
Table 2.1: Showing the first column of the matrix to the left and the complete matrix to the right

<table>
<thead>
<tr>
<th>12</th>
<th>14</th>
<th>21</th>
<th>23</th>
<th>31</th>
<th>41</th>
</tr>
</thead>
<tbody>
<tr>
<td>12</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>31</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>41</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>43</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>-1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

described above. The set of all such vectors span the cycle space of the graph G. The dimension of the cycle space is \( m - n + 1 \). One can create a cycle basis for the graph by first creating a spanning tree \( T \) and then adding one arc \((s, t)\) to the tree and thereby inducing a unique circuit and one basis vector \( \gamma_{st} \). Each arc \((s, t)\) will yield a unique basis vector \( \gamma_{st} \).

We will use the SP-graph in Figure 2.3 as an example of how one can create a cycle basis. First we create the tree, which consists of arcs in \( A_l \), by doing a breadth first search from the destination node. We will denote the arcs in \( D_l \) by putting a \( \sim \) above it. In this case the tree will consist of the arcs \( T_l = \{ (4, 3), \sim (1, 2), (3, 2) \} \). Now we take one arc not in the tree, e.g. the arc \((1, 2)\) and add it to the tree. This induces the cycle consisting of the arcs \((1, 2)\) and \( \sim (1, 2) \). The arc not in the tree \((1, 2)\) will be used forwards and the arc in the tree \((1, 2)\) will be used backwards. Thus, there will be a 1 in the row corresponding to the arc \((1, 2)\) and a -1 in the row corresponding to the arc \( \sim (1, 2) \).

Doing the same for the remaining arcs not in the tree will create the complete matrix where each column represents one of the basis vectors.

![Figure 2.4: The SP-graph to the left. The spanning tree, to the right.](image-url)
Figure 2.5: The graphs show the tree with one arc (not in the tree) added. Each one represents one simple cycle and one basis vector or one column in Table 2.1.
Chapter 3

Mathematical Models and Algorithms

This chapter will briefly describe the mathematical models we have implemented and tested. They all solve the same problem, i.e. they tell us if there exists compatible weights for the SP-graphs given see 3.1. They are all linear programs but the structure of the models differ, which makes it interesting to compare how they behave in practice when we use different solvers. The models tested are:

- P1, the natural formulation of the ISPR problem, described in (3.1).
- P2, the dual formulation of P1, described in (3.2).
- P3, the dual formulation of P1 with variable transformation, described in (3.3).
- P4, cycle basis formulation, described in (3.4).
- P5, cycle basis formulation simplified, described in (3.5).

In addition, valid cycles and the algorithm for finding them are described in (3.6).

3.1 P1. The natural formulation

Let \( G = (N, A) \) be a directed graph and \( A_l \) a family of SP-graphs for the set of destinations \( L \subseteq N \) where each SP-graph describes the shortest paths to its destinations. As described earlier we made the simplification that all \( D_l \) arcs were added to \( A_l \). This means that \( \forall l \in L, A_l \) will be a spanning in-graph to \( l \). In addition this also means that some of the arcs in \( A_l \) may not correspond
to an actual arc in the network. For this reason the following model cannot in
general say if the weights when found, are compatible with the SP-graphs, see
[1]. For the special case that \( A_l \) is a spanning in-graph to \( l \) and \( D_l = \emptyset \ \forall l \in L \)
the weights when found will in fact be compatible with the SP-graphs.

Each arc \((i, j) \in A\) should be assigned a weight \( w_{ij} \). Each node \( i \in N \) should be
assigned a node potential for each destination \( l \in L \) written \( \pi^l_i \). If this system
is infeasible, no weights compatible with the SP-graphs exist. The converse is
not true, see [1].

\[
\begin{align*}
\min & \sum_{(i, j) \in A} w_{ij} \\
\text{subject to} & \quad w_{ij} + \pi^l_i - \pi^l_j = 0 \quad (i, j) \in A_l, \ l \in L \\
& \quad w_{ij} + \pi^l_i - \pi^l_j \geq 1 \quad (i, j) \in A_l, \ l \in L \\
& \quad w_{ij} + \pi^l_i - \pi^l_j \geq 0 \quad (i, j) \in U_l, \ l \in L \\
& \quad w_{ij} \geq 1 \quad (i, j) \in A.
\end{align*}
\]

(3.1)

3.2 P2. The Dual formulation of P1

We are mostly interested in if the system is feasible or not. Therefore we can
change the objective function to zero. After doing this and applying the trans-
formation \( w := w - 1 \) we take the dual of the Model (3.1) to get the following
system.

\[
\begin{align*}
\min & \sum_{(i, j) \in A} \sum_{l \in L} \theta^l_{ij} \\
\text{subject to} & \quad \sum_{j: (i, j) \in A} \theta^l_{ij} - \sum_{j: (j, i) \in A} \theta^l_{ji} = 0 \quad i \in N, \ l \in L \\
& \quad \sum_{l \in L} \theta^l_{ij} \leq 0 \quad (i, j) \in A \\
& \quad \theta^l_{ij} \geq 0 \quad (i, j) \in A \setminus A_l
\end{align*}
\]

(3.2)

If and only if Model (3.2) has an objective of zero (3.1) is feasible.

Since we’re only interested in finding any objective value less than zero we also
limit the variables as follows:

\[
\begin{align*}
-1 \leq \theta^l_{ij} & \leq 1 \quad (i, j) \in A_l \\
0 \leq \theta^l_{ij} & \leq 1 \quad (i, j) \in A \setminus A_l
\end{align*}
\]

This will create a lower bound for the objective, which otherwise would be
unbounded.
3.3 P3. The Dual formulation of P1 with variable transformation

The Model (3.2) is very similar to a MMCF problem but we need all the variables to be greater than or equal to zero so we apply the transformation:

\[
\begin{aligned}
\theta^i_{ij} &= x^i_{ij} - 1 \quad (i, j) \in A_l \\
\theta^i_{ij} &= x^i_{ij} \quad (i, j) \in A \setminus A_l
\end{aligned}
\]

and get

\[
\begin{align*}
\min \quad & \sum_{(i,j) \in A \setminus \bar{A}_l} \sum_{l \in L} x^i_{ij} \\
\sum_{j : (j,i) \in A} x^i_{ij} - \sum_{j : (i,j) \in A} x^i_{ji} &= b^i_l \quad i \in N, l \in L \\
\sum_{l \in L} x^i_{ij} &\leq u_{ij} \quad (i,j) \in A \\
0 &\leq x^i_{ij} \leq 2 \quad (i,j) \in A_l \\
0 &\leq x^i_{ij} \leq 1 \quad (i,j) \in A \setminus A_l
\end{align*}
\]

(3.3)

where

\[
\begin{align*}
b^i_l &= \sum_{j : (i,j) \in A_l} - \sum_{j : (j,i) \in A_l} i \in N, l \in L \\
u_{ij} &= \sum_{l \in L} |(i,j) \in A_l| \quad (i,j) \in A
\end{align*}
\]

The new objective function increases by \( s = \sum_{l \in L} |(i,j) \in A_l| \).

Model (3.3) has an objective of \( s \) if and only if (3.1) is feasible.

3.4 P4. The Cycle basis formulation

Another formulation involves constructing a cycle basis from the SP-graphs, see 2.3. For a complete derivation of the model refer to [1].

For each destination \( l \in L \), create the SP-tree \( T_l \). For each tree \( T_l \), create the matrix \( \Gamma_l \) consisting of the columns \( \gamma_{l,st} \), one column for each \( (s,t) \in A \setminus T_l \).

This yields the following model:

\[
\begin{align*}
\min \quad & \sum_{(s,t) \in A \setminus T_l} \sum_{l \in L} c^l_{st} x^l_{st} \\
\sum_{(s,t) \in A \setminus T_l} \sum_{l \in L} \gamma^l_{l,st} x^l_{st} &\leq 0 \quad (i,j) \in A \\
x^l_{st} &\geq 0 \quad (s,t) \in A \setminus A_l, l \in L
\end{align*}
\]

(3.4)

where

\[
c^l_{st} = \sum_{(i,j) \in A \setminus \bar{A}_l} \gamma^l_{l,st}
\]
The Model (3.4) has an objective of zero if and only if (3.1) is feasible.

### 3.5 P5. The Cycle basis formulation simplified

A theoretical property of Model (3.4) is that the capacity constraint is in some sense binding and it is therefore possible to replace them with an equality constraint, see Chapter 6 in [1]. After this has been done it is possible to simplify the model slightly by eliminating rows in the constraint matrix corresponding to arcs from one of the SP graphs $l' \in L$. This follows from the fact that these rows are linearly dependent.

Change the inequality constraint in 3.4 to an equality constraint. Remove all rows for the arcs $N_{l'} = A_{l'} \setminus T_{l'}$ and $T_{l'}$, and we are left with rows for the arcs in $\bar{A}_{l'} \cup U_{l'}$.

When choosing an SP graph $l'$ it makes sense to take the one with the most arcs, i.e. with max $|N_{l'}|$.

\[
\begin{align*}
\max \sum_{(s,t) \in A \setminus T_l} \sum_{l \in L} c^l_{st} x^l_{st} \\
\sum_{(s,t) \in A \setminus T_l} \sum_{l \in L} \gamma^l_{lst} x^l_{st} = 0 & \quad (i, j) \in \bar{A}_{l'} \cup U_{l'} \\
x^l_{st} \geq 0 & \quad (s,t) \in A \setminus A_{l'}, \; l \in L
\end{align*}
\] (3.5)

where

\[c^l_{st} = \sum_{(i,j) \in A \setminus A_{l'}} \gamma^l_{lst}\]

This model has an objective of zero if and only if (3.1) is feasible.

### 3.6 Valid cycles

When solving the ISPR problem we are more interested in if there exists some set of weights, rather than finding a particular set of weights. One way to find necessary conditions for the existence of a set of weights is to look for conflicts between pairs of SP-graphs. One such conflict which will be described in the following is called a valid cycle. If a valid cycle exists there cannot exist a set of weights compatible with the SP-graph routing pattern. In general, the valid cycle search will be a lot faster than solving the complete problem, the tradeoff is of course that knowing that there is no valid cycle does not tell us if there is another conflict involving more than two SP-graphs. In any event, depending on how much faster it is to search for valid cycles it may still be a worthwhile first step before bringing in the heavy artillery.

In the beginning we talked about what a cycle was, disregarding the direction of the arcs, a path from a node back to the same node is a cycle, and further,
taking the direction of the arcs into account, a path from a node back to the same node is a directed cycle. In our case we will look at the combination of two SP-graphs at once to see if their shortest-path arcs, when combined, form any cycles. In addition, the cycles must be such that, by reversing the direction of one of the SP-graph’s shortest-path arcs the cycle will become a directed cycle, this is called a feasible cycle. Figure 3.1 shows two SP-graphs in the same figure with 4 nodes and 2 arcs each, the solid arcs belonging to $A_3$ and the dotted arcs to $A_4$ with the destination 3 and 4 respectively.

![Figure 3.1: Two SP-graphs forming a feasible cycle that does not cause a conflict.](image)

If we reverse the arcs of $A_4$ we obtain a directed cycle $\{(1, 2), (2, 3), (3, 1)\}$. Similarly we can reverse the arcs in $A_3$ to obtain the directed cycle $\{(1, 3), (3, 2), (2, 1)\}$. This is the first step in finding a valid cycle, but to have a conflict some additional properties must be present. In Figure 3.1 we can find weights making the path $\{(1, 2), (2, 3)\}$ the same length as the path $\{(1, 3)\}$ thereby satisfying the requirement that the solid arcs and the dotted arcs are on shortest paths. If the solid SP-graph however has $\bar{A}_3 = \{(1, 3)\}$ i.e. $(1, 3)$ must not be on a shortest path to node 3 there is no way to satisfy the requirements and we have a conflict that we call a valid cycle. Similarly if the dotted SP-graph has either $(1, 2)$ or $(2, 3)$ in $\bar{A}_4$ we have a valid cycle. This type of conflict is called a saturated valid cycle in [1].

![Figure 3.2: A valid cycle. The arc ending with a dot represents a non-shortest-path arc.](image)

A feasible cycle may also be valid if there is a “double-arc” in the cycle as
shown in Figure 3.3. Here we also have the desired directed cycle when reversing the arcs of one of the SP-graphs but no non-shortest-path arcs. A difference to Figure 3.1 is that here we have two opposing arcs between nodes 2 and 3. This will lead to a conflict. The two paths from node 1 to 2 must be shortest, but also the two paths between 1 and 3. Now, thinking about the weight of each arc in each path we get the following:

\[
\begin{align*}
    w_{12} + w_{23} &= w_{13} \\
    w_{12} &= w_{13} + w_{32} \Rightarrow w_{32} &= -w_{23}
\end{align*}
\]  

(3.6)

The weights however, must be greater than or equal to one so no valid set of weights are compatible with this routing pattern. This type of conflict is called a non-saturated valid cycle in [1].

### 3.6.1 Finding valid cycles

Here follows a description of the algorithm used for finding valid cycles. Given two SP-graphs \(l'\) and \(l''\), find a valid cycle, if one exists.

1. Form the graph \(\bar{G}\) in the following way.
   - \(\forall (i, j) \in A_{\bar{G}}\) add \((i, j)\) to \(\bar{G}\).
   - \(\forall (i, j) \in A_{\bar{G}}^r\) add \((j, i)\) to \(\bar{G}\).
2. Find all strongly connected components \(C_k\) of \(G\).
3. \(\forall C_k : |N| \geq 3\) do the following
   - Find an arc \((i, j) \in C_k\) such that
     \[\begin{align*}
     (i, j) \in A_{\bar{G}} \land (i, j) \in A_{\bar{G}}^r \lor (j, i) \in A_{\bar{G}} \land (j, i) \in A_{\bar{G}}^r
     \end{align*}\]
     if found return Valid cycle exists.
   - Find two arcs \((i, j) \in C_k\), if found return Valid cycle exists.
4. return No valid cycle exists.
3.6. Valid cycles

Figure 3.4: Flowchart of the algorithm for finding valid cycles.
Chapter 4

Method used in performing the study

The main purpose of this thesis has been to implement and test the performance of the models described in chapter 3. This chapter is an overview of how the tests were carried out and the different solvers used. All tests were performed on the following system:

- **OS**: Solaris 10 8/07 s10s_u4wos.12b SPARC
- **System Configuration**: Sun Microsystems sun4v SPARC Enterprise T1000
- **System clock frequency**: 200 MHz
- **Memory size**: 8184 Megabytes
- **CPLEX version**: v10.0

See Appendix B for a description of the program used for performing the tests.

4.1 CPLEX optimization software

All the models were solved using CPLEX, a commercial optimization software package currently owned by IBM. CPLEX is known as being one of the best and fastest implementations of optimization algorithms around, with bindings and interfaces for many different programming languages. In this thesis we chose to implement all the software in C++ using the CPLEX callable library for interfacing with the CPLEX optimizers. This was a natural choice, C++ is a very mature language today with high quality compilers available for most architectures. The same can be said even more strongly about C, but C++ has the added benefit of having a large standard library simplifying many programming tasks, especially involving dynamic memory management.

CPLEX has several solvers that may be used and different settings that may be applied to the solvers for controlling its behaviour, see [8]. The different
solvers we will look at here are the linear solvers:

1. primopt - Uses the primal simplex algorithm.
2. dualopt - Uses the dual simplex algorithm.
3. baropt - Uses the barrier algorithm.
4. hybbaropt - First applies the barrier algorithm followed by an automatic crossover to a basic solution if barrier determines that the problem is both primal and dual feasible.
5. siftopt - Looks at a subset of the columns of a problem and uses either the barrier or simplex optimizer to solve this reduced model.

4.2 MMCF Solvers

The reason for transforming Model (3.2) into Model (3.3) was to obtain a standard MMCF formulation of the problem. Once we have this formulation we can use any specialized MMCF solver available. The hope being that a solver taking advantage of the special network structure of the problem will be faster than the CPLEX solvers (which may not take advantage of this). The different MMCF solvers looked at are:

1. MMCFB - Solver based on dualizing the mutual capacity "complicat- ing" constraints and solving the resulting Lagrangian Dual with either a Bundle-type algorithm or a SubGradient algorithm, see [7].
2. PPRN - Solver based on a primal partitioning method, see [6].
3. IPM - Solver based on a specialized interior-point algorithm, see [6].

4.3 The instance generator

A crucial part in testing the performance of the different models is to have a lot of varied input data to feed the models. For this reason we created an instance generator that takes a set of parameters and generates a random SPG file (see Appendix A) within the boundaries set by the parameters. In this way we can produce plenty of data quickly and also have the data be varied enough to be able to cover a large spectrum of possible networks and SPGs.

How it works

The instance generator will first create a graph $G = (N, A)$ where $|N|$ and $|A|$ is selected by the user. The set of arcs $A$ are randomly selected and $\forall (i, j) \in A$ $(i, j)$ is given a random weight between 1 and $|N|$. The minimum path length
4.3. The instance generator

is calculated between every pair of nodes using the weights generated. Having
this information enables us to determine, for each arc, if it is on a shortest-path
to a particular node.

The destinations are set in sequence, the first destination will be node 0 (the
nodes are numbered from 0 to \(|N| - 1\)), the second node 1 and so on. For each
destination \(l\) we go through each arc in \(A\), checking if it's on a shortest path
to this destination \(l\) or not and setting it to shortest \((A_l)\), non-shortest \((\bar{A}_l)\) or
don't-care \((U_l)\) depending on the parameters described below.

Note the distinction between an arc being on a shortest path with respect to
the generated weights and the arc being labeled as a shortest-path arc in the
resulting SP-graph.

A point worth mentioning, if an SP-graph contains a directed cycle (with re-
spect to \(A_l\)), the routing pattern is trivially infeasible. If the generator detects
that such a cycle has been created it will remove one of the arcs that make up
the cycle (from \(A_l\)).

The parameters

The set of parameters specified by the user to control the instance generator
are:

- **\(N\)**  Specifies the number of nodes in the network.
- **\(P_A\)**  Probability that an arc is present in the network.
- **\(K\)**     Specifies the number of destinations, between 1 and \(N\).
- **\(ss\)**    Specifies the probability that a shortest-path arc (w.r.t. weights) will be a
               shortest-path arc (in the SPG).
- **\(sn\)**    Specifies the probability that a shortest path arc will be a non-shortest path
               arc.
- **\(ns\)**    Specifies the probability that a non-shortest path arc will be shortest.
- **\(nn\)**    Specifies the probability that a non-shortest path arc will be non-shortest.

The parameter \(P_A\) specifies the probability of an arc being present but to make
the network more realistic we also decide that if an arc is present its reverse arc
is also present. So for instance if we have \(N = 10\) and \(P_A = 0.4\) the expected
number of arcs will be

\[
10 \cdot 9 \cdot (1 - (1 - 0.4)(1 - 0.4)) = 57.6
\]

Which comes from the fact that an arc and its reversal neither being present
has the probability \((1 - P_A)(1 - P_A)\) and the total number of possible arcs
are \(N(N - 1)\).

The parameters \(ss\), \(sn\), \(ns\) and \(nn\) are checked in order. This means that for
instance if \(ss = 0.5\) and \(sn = 0.5\) it will first see if the arc is a shortest path arc
(w.r.t weights). If so it will set it to a shortest-path arc (w.r.t. SPG) for the current
destination with probability 0.5. Only if it didn’t get set to a shortest-path arc
will it set it to a non-shortest path arc with probability 0.5. So in actuality the
chance of a shortest-path arc ending up as a shortest path arc for the destination
is 0.5 and the chance of it ending up as a non-shortest path arc is $(1 - 0.5) \cdot 0.5 =
0.25$ in this case, or $P((i, j) \in A_I) = ss$ and $P((i, j) \in A_{I}) = (1 - ss)sn$. If the arc
didn’t get set as either shortest or non-shortest it will be a don’t-care arc. The
same principle applies for the parameters ns and nn.
Chapter 5

Results of the tests

The results shown here were produced from a large set of different SPG-files generated with the instance generator described earlier. Each Model previously described was solved with each of the different solvers. First a smaller test was done where each sample consists of ten SPG-files, the result showing the average of these ten. This test will decide which models/solvers that will qualify for more extensive testing. In the extensive tests the results are averaged over 100 SPG-files.

5.1 Abbreviations

We will first give a brief explanation of the abbreviations used.

N Number of nodes.

M Number of arcs (average over all instances).

K Number of destinations.

ss Parameter used for generating the files, see section 4.3.

sn Parameter used for generating the files, see section 4.3.

ns Parameter used for generating the files, see section 4.3.

nn Parameter used for generating the files, see section 4.3.

Inf Number of instances in the sample that were infeasible as defined in Model (3.1).

VC Number of instances in the sample where a valid cycle was found.

p Time of CPLEX primopt solver.
5.2 Problems encountered

The multicommodity solver IPM did not manage to solve any of the problems given to it. After a few iterations some of the results showed NaN “Not a Number” and it never recovered from this state. The MMCFB subgradient solver did not manage to solve the problems correctly, many times reporting an objective value of Inf incorrectly. For this reason these two solvers were discarded from further testing.

5.3 Preliminary test

We start out by running a series of smaller tests for a few different network sizes ranging from 10 nodes and 2 destinations up to 20 nodes and 20 destinations. Each row in the data tables represents an average of 10 similar files.
Table 5.1: Definition of instances

<table>
<thead>
<tr>
<th>Name</th>
<th>N</th>
<th>M</th>
<th>K</th>
<th>ss</th>
<th>sn</th>
<th>ns</th>
<th>nn</th>
<th>Inf</th>
<th>VC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>88</td>
<td>2</td>
<td>0.800</td>
<td>0.001</td>
<td>0.100</td>
<td>0.100</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
<td>79</td>
<td>10</td>
<td>0.800</td>
<td>0.001</td>
<td>0.055</td>
<td>0.055</td>
<td>9</td>
<td>9</td>
</tr>
<tr>
<td>C</td>
<td>20</td>
<td>365</td>
<td>2</td>
<td>0.800</td>
<td>0.001</td>
<td>0.010</td>
<td>0.010</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>D</td>
<td>20</td>
<td>290</td>
<td>4</td>
<td>0.800</td>
<td>0.001</td>
<td>0.010</td>
<td>0.010</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>E</td>
<td>20</td>
<td>249</td>
<td>8</td>
<td>0.900</td>
<td>0.001</td>
<td>0.002</td>
<td>0.010</td>
<td>8</td>
<td>4</td>
</tr>
<tr>
<td>F</td>
<td>20</td>
<td>193</td>
<td>20</td>
<td>0.900</td>
<td>0.000</td>
<td>0.001</td>
<td>0.005</td>
<td>8</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 5.2: Comparison of solvers used for P1, average of 10. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>p</th>
<th>d</th>
<th>b</th>
<th>h</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.025</td>
<td>0.022</td>
<td>0.029</td>
<td>0.033</td>
<td>0.058</td>
</tr>
<tr>
<td>B</td>
<td>0.084</td>
<td>0.075</td>
<td>0.527</td>
<td>0.557</td>
<td>2.326</td>
</tr>
<tr>
<td>C</td>
<td>0.058</td>
<td>0.050</td>
<td>0.062</td>
<td>0.069</td>
<td>0.223</td>
</tr>
<tr>
<td>D</td>
<td>0.077</td>
<td>0.076</td>
<td>1.104</td>
<td>1.133</td>
<td>2.364</td>
</tr>
<tr>
<td>E</td>
<td>0.173</td>
<td>0.169</td>
<td>1.044</td>
<td>1.168</td>
<td>3.264</td>
</tr>
<tr>
<td>F</td>
<td>1.074</td>
<td>0.743</td>
<td>8.059</td>
<td>8.735</td>
<td>10.991</td>
</tr>
</tbody>
</table>

Table 5.3: Comparison of solvers used for P2, average of 10. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>p</th>
<th>d</th>
<th>b</th>
<th>h</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.027</td>
<td>0.024</td>
<td>0.029</td>
<td>0.037</td>
<td>0.117</td>
</tr>
<tr>
<td>B</td>
<td>0.240</td>
<td>0.090</td>
<td>0.271</td>
<td>0.290</td>
<td>1.344</td>
</tr>
<tr>
<td>C</td>
<td>0.054</td>
<td>0.052</td>
<td>0.064</td>
<td>0.070</td>
<td>0.482</td>
</tr>
<tr>
<td>D</td>
<td>0.098</td>
<td>0.088</td>
<td>0.142</td>
<td>0.157</td>
<td>1.058</td>
</tr>
<tr>
<td>E</td>
<td>0.497</td>
<td>0.192</td>
<td>0.448</td>
<td>0.478</td>
<td>3.063</td>
</tr>
<tr>
<td>F</td>
<td>7.880</td>
<td>1.293</td>
<td>4.915</td>
<td>5.010</td>
<td>12.790</td>
</tr>
</tbody>
</table>

Figure 5.1: Comparison of solvers for method P1 at different N and K.
Figure 5.2: Comparison of solvers for method P2 at different N and K.

Table 5.4: Comparison of solvers used for P3, average of 10. A ‘-’ indicates no data available: the solver did not complete in time or terminated without a result. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>p</th>
<th>d</th>
<th>PP</th>
<th>b</th>
<th>h</th>
<th>s</th>
<th>MP</th>
<th>MD</th>
<th>MA</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.027</td>
<td>0.023</td>
<td>0.024</td>
<td>0.031</td>
<td>0.034</td>
<td>0.114</td>
<td>0.020</td>
<td>0.025</td>
<td>0.020</td>
</tr>
<tr>
<td>B</td>
<td>0.181</td>
<td>0.088</td>
<td>0.207</td>
<td>0.258</td>
<td>0.276</td>
<td>1.121</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>C</td>
<td>0.053</td>
<td>0.054</td>
<td>0.456</td>
<td>0.063</td>
<td>0.070</td>
<td>0.363</td>
<td>-</td>
<td>0.038</td>
<td>-</td>
</tr>
<tr>
<td>D</td>
<td>0.101</td>
<td>0.089</td>
<td>-</td>
<td>0.135</td>
<td>0.150</td>
<td>1.129</td>
<td>0.122</td>
<td>0.118</td>
<td>0.122</td>
</tr>
<tr>
<td>E</td>
<td>0.433</td>
<td>0.192</td>
<td>-</td>
<td>0.419</td>
<td>0.447</td>
<td>3.002</td>
<td>-</td>
<td>0.399</td>
<td>-</td>
</tr>
<tr>
<td>F</td>
<td>7.779</td>
<td>1.209</td>
<td>-</td>
<td>4.316</td>
<td>4.412</td>
<td>12.889</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Figure 5.3: Comparison of solvers for method P3 at different N and K. No bar means no data.
Figure 5.4: Comparison of solvers for method P3 at different N and K. No bar means no data.

Table 5.5: Comparison of solvers used for P4, average of 10. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>p</th>
<th>d</th>
<th>b</th>
<th>h</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.024</td>
<td>0.023</td>
<td>0.026</td>
<td>0.031</td>
<td>0.062</td>
</tr>
<tr>
<td>B</td>
<td>0.112</td>
<td>0.065</td>
<td>0.154</td>
<td>0.165</td>
<td>0.441</td>
</tr>
<tr>
<td>C</td>
<td>0.052</td>
<td>0.048</td>
<td>0.054</td>
<td>0.062</td>
<td>0.330</td>
</tr>
<tr>
<td>D</td>
<td>0.077</td>
<td>0.072</td>
<td>0.113</td>
<td>0.125</td>
<td>0.865</td>
</tr>
<tr>
<td>E</td>
<td>0.168</td>
<td>0.129</td>
<td>0.247</td>
<td>0.268</td>
<td>1.297</td>
</tr>
<tr>
<td>F</td>
<td>1.126</td>
<td>0.402</td>
<td>1.031</td>
<td>1.083</td>
<td>3.631</td>
</tr>
</tbody>
</table>

Figure 5.5: Comparison of solvers for method P4 at different N and K.
Table 5.6: Comparison of solvers used for P5, average of 10. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>p</th>
<th>d</th>
<th>b</th>
<th>h</th>
<th>s</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.020</td>
<td>0.021</td>
<td>0.019</td>
<td>0.023</td>
<td>0.029</td>
</tr>
<tr>
<td>B</td>
<td>0.114</td>
<td>0.104</td>
<td>0.140</td>
<td>0.152</td>
<td>0.553</td>
</tr>
<tr>
<td>C</td>
<td>0.044</td>
<td>0.045</td>
<td>0.041</td>
<td>0.043</td>
<td>0.072</td>
</tr>
<tr>
<td>D</td>
<td>0.065</td>
<td>0.065</td>
<td>0.079</td>
<td>0.087</td>
<td>0.366</td>
</tr>
<tr>
<td>E</td>
<td>0.115</td>
<td>0.131</td>
<td>0.168</td>
<td>0.186</td>
<td>0.966</td>
</tr>
<tr>
<td>F</td>
<td>0.579</td>
<td>0.736</td>
<td>0.673</td>
<td>0.715</td>
<td>2.824</td>
</tr>
</tbody>
</table>

Figure 5.6: Comparison of solvers for method P5 at different N and K. Time in seconds.

Table 5.7: Side by side Comparison of the best solvers from previous tests, average of 10.

<table>
<thead>
<tr>
<th>Name</th>
<th>P1/d</th>
<th>P2/d</th>
<th>P3/d</th>
<th>P4/d</th>
<th>P5/p</th>
<th>P5/b</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.022</td>
<td>0.024</td>
<td>0.023</td>
<td>0.023</td>
<td>0.020</td>
<td>0.019</td>
</tr>
<tr>
<td>B</td>
<td>0.075</td>
<td>0.090</td>
<td>0.088</td>
<td>0.065</td>
<td>0.114</td>
<td>0.140</td>
</tr>
<tr>
<td>C</td>
<td>0.050</td>
<td>0.052</td>
<td>0.054</td>
<td>0.048</td>
<td>0.044</td>
<td>0.041</td>
</tr>
<tr>
<td>D</td>
<td>0.076</td>
<td>0.088</td>
<td>0.089</td>
<td>0.072</td>
<td>0.065</td>
<td>0.079</td>
</tr>
<tr>
<td>E</td>
<td>0.169</td>
<td>0.192</td>
<td>0.192</td>
<td>0.129</td>
<td>0.115</td>
<td>0.168</td>
</tr>
<tr>
<td>F</td>
<td>0.743</td>
<td>1.293</td>
<td>1.209</td>
<td>0.402</td>
<td>0.579</td>
<td>0.673</td>
</tr>
</tbody>
</table>
Figure 5.7: Comparison of the best methods/solvers at different N and K. Time in seconds.
5.4 Extensive test

From the preliminary tests we conclude that the Models/solvers that qualify for more tests are

- P1/d
- P2/d
- P3/d
- P4/d
- P5/p
- P5/b

Table 5.8: Definition of instances

<table>
<thead>
<tr>
<th>Name</th>
<th>N</th>
<th>M</th>
<th>K</th>
<th>ss</th>
<th>sn</th>
<th>ns</th>
<th>nn</th>
<th>Inf</th>
<th>VC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>75</td>
<td>2</td>
<td>0.800</td>
<td>0.060</td>
<td>0.060</td>
<td>0.010</td>
<td>57</td>
<td>49</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
<td>76</td>
<td>5</td>
<td>0.700</td>
<td>0.010</td>
<td>0.030</td>
<td>0.006</td>
<td>81</td>
<td>73</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
<td>76</td>
<td>10</td>
<td>0.500</td>
<td>0.010</td>
<td>0.010</td>
<td>0.002</td>
<td>80</td>
<td>63</td>
</tr>
<tr>
<td>D</td>
<td>20</td>
<td>287</td>
<td>2</td>
<td>0.600</td>
<td>0.050</td>
<td>0.040</td>
<td>0.005</td>
<td>81</td>
<td>74</td>
</tr>
<tr>
<td>E</td>
<td>20</td>
<td>196</td>
<td>10</td>
<td>0.500</td>
<td>0.001</td>
<td>0.004</td>
<td>0.001</td>
<td>61</td>
<td>46</td>
</tr>
<tr>
<td>F</td>
<td>20</td>
<td>192</td>
<td>20</td>
<td>0.500</td>
<td>0.001</td>
<td>0.004</td>
<td>0.001</td>
<td>95</td>
<td>73</td>
</tr>
<tr>
<td>G</td>
<td>30</td>
<td>732</td>
<td>2</td>
<td>0.500</td>
<td>0.020</td>
<td>0.020</td>
<td>0.010</td>
<td>71</td>
<td>59</td>
</tr>
<tr>
<td>H</td>
<td>30</td>
<td>446</td>
<td>15</td>
<td>0.300</td>
<td>0.002</td>
<td>0.002</td>
<td>0.001</td>
<td>51</td>
<td>30</td>
</tr>
<tr>
<td>I</td>
<td>30</td>
<td>445</td>
<td>30</td>
<td>0.300</td>
<td>0.001</td>
<td>0.001</td>
<td>0.001</td>
<td>58</td>
<td>26</td>
</tr>
</tbody>
</table>

Table 5.9: Data for P1/d, average of 100. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>T-avg</th>
<th>Min</th>
<th>Max</th>
<th>Min5</th>
<th>Max95</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.024</td>
<td>0.015</td>
<td>0.028</td>
<td>0.018</td>
<td>0.027</td>
</tr>
<tr>
<td>B</td>
<td>0.038</td>
<td>0.024</td>
<td>0.050</td>
<td>0.029</td>
<td>0.045</td>
</tr>
<tr>
<td>C</td>
<td>0.085</td>
<td>0.042</td>
<td>0.140</td>
<td>0.061</td>
<td>0.121</td>
</tr>
<tr>
<td>D</td>
<td>0.045</td>
<td>0.029</td>
<td>0.072</td>
<td>0.035</td>
<td>0.056</td>
</tr>
<tr>
<td>E</td>
<td>0.382</td>
<td>0.144</td>
<td>1.690</td>
<td>0.167</td>
<td>0.878</td>
</tr>
<tr>
<td>F</td>
<td>4.144</td>
<td>0.624</td>
<td>18.998</td>
<td>1.361</td>
<td>10.931</td>
</tr>
<tr>
<td>G</td>
<td>0.104</td>
<td>0.062</td>
<td>0.396</td>
<td>0.082</td>
<td>0.158</td>
</tr>
<tr>
<td>H</td>
<td>17.961</td>
<td>0.694</td>
<td>101.906</td>
<td>3.314</td>
<td>57.146</td>
</tr>
<tr>
<td>I</td>
<td>139.515</td>
<td>2.398</td>
<td>811.294</td>
<td>45.799</td>
<td>580.909</td>
</tr>
</tbody>
</table>
Table 5.10: Data for P2/d, average of 100. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>T-avg</th>
<th>Min</th>
<th>Max</th>
<th>Min5</th>
<th>Max95</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.024</td>
<td>0.017</td>
<td>0.029</td>
<td>0.019</td>
<td>0.028</td>
</tr>
<tr>
<td>B</td>
<td>0.043</td>
<td>0.034</td>
<td>0.052</td>
<td>0.036</td>
<td>0.050</td>
</tr>
<tr>
<td>C</td>
<td>0.105</td>
<td>0.078</td>
<td>0.167</td>
<td>0.088</td>
<td>0.134</td>
</tr>
<tr>
<td>D</td>
<td>0.049</td>
<td>0.040</td>
<td>0.057</td>
<td>0.043</td>
<td>0.053</td>
</tr>
<tr>
<td>E</td>
<td>0.386</td>
<td>0.260</td>
<td>0.624</td>
<td>0.289</td>
<td>0.502</td>
</tr>
<tr>
<td>F</td>
<td>2.998</td>
<td>1.763</td>
<td>4.312</td>
<td>2.200</td>
<td>4.028</td>
</tr>
<tr>
<td>G</td>
<td>0.094</td>
<td>0.078</td>
<td>0.111</td>
<td>0.087</td>
<td>0.103</td>
</tr>
<tr>
<td>H</td>
<td>3.061</td>
<td>0.890</td>
<td>5.560</td>
<td>1.719</td>
<td>4.713</td>
</tr>
<tr>
<td>I</td>
<td>34.464</td>
<td>15.891</td>
<td>57.465</td>
<td>20.627</td>
<td>48.137</td>
</tr>
</tbody>
</table>

Table 5.11: Data for P3/d, average of 100. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>T-avg</th>
<th>Min</th>
<th>Max</th>
<th>Min5</th>
<th>Max95</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.023</td>
<td>0.018</td>
<td>0.029</td>
<td>0.020</td>
<td>0.026</td>
</tr>
<tr>
<td>B</td>
<td>0.043</td>
<td>0.038</td>
<td>0.053</td>
<td>0.039</td>
<td>0.049</td>
</tr>
<tr>
<td>C</td>
<td>0.105</td>
<td>0.078</td>
<td>0.149</td>
<td>0.087</td>
<td>0.134</td>
</tr>
<tr>
<td>D</td>
<td>0.048</td>
<td>0.040</td>
<td>0.056</td>
<td>0.042</td>
<td>0.053</td>
</tr>
<tr>
<td>E</td>
<td>0.383</td>
<td>0.266</td>
<td>0.643</td>
<td>0.290</td>
<td>0.501</td>
</tr>
<tr>
<td>F</td>
<td>3.001</td>
<td>1.915</td>
<td>4.416</td>
<td>2.143</td>
<td>3.836</td>
</tr>
<tr>
<td>G</td>
<td>0.094</td>
<td>0.080</td>
<td>0.111</td>
<td>0.090</td>
<td>0.102</td>
</tr>
<tr>
<td>H</td>
<td>2.997</td>
<td>1.194</td>
<td>4.814</td>
<td>1.900</td>
<td>4.406</td>
</tr>
<tr>
<td>I</td>
<td>33.544</td>
<td>15.691</td>
<td>51.106</td>
<td>22.170</td>
<td>46.865</td>
</tr>
</tbody>
</table>

Table 5.12: Data for P4/d, average of 100. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>T-avg</th>
<th>Min</th>
<th>Max</th>
<th>Min5</th>
<th>Max95</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.021</td>
<td>0.017</td>
<td>0.027</td>
<td>0.018</td>
<td>0.024</td>
</tr>
<tr>
<td>B</td>
<td>0.034</td>
<td>0.028</td>
<td>0.041</td>
<td>0.031</td>
<td>0.038</td>
</tr>
<tr>
<td>C</td>
<td>0.075</td>
<td>0.060</td>
<td>0.099</td>
<td>0.064</td>
<td>0.089</td>
</tr>
<tr>
<td>D</td>
<td>0.044</td>
<td>0.038</td>
<td>0.051</td>
<td>0.039</td>
<td>0.049</td>
</tr>
<tr>
<td>E</td>
<td>0.217</td>
<td>0.150</td>
<td>0.299</td>
<td>0.170</td>
<td>0.281</td>
</tr>
<tr>
<td>F</td>
<td>1.652</td>
<td>0.953</td>
<td>2.509</td>
<td>1.125</td>
<td>2.236</td>
</tr>
<tr>
<td>G</td>
<td>0.090</td>
<td>0.078</td>
<td>0.170</td>
<td>0.084</td>
<td>0.099</td>
</tr>
<tr>
<td>H</td>
<td>2.053</td>
<td>0.901</td>
<td>3.777</td>
<td>1.159</td>
<td>3.258</td>
</tr>
</tbody>
</table>
Table 5.13: Data for P5/p, average of 100. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>T-avg</th>
<th>Min</th>
<th>Max</th>
<th>Min5</th>
<th>Max95</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.020</td>
<td>0.015</td>
<td>0.024</td>
<td>0.017</td>
<td>0.023</td>
</tr>
<tr>
<td>B</td>
<td>0.031</td>
<td>0.027</td>
<td>0.039</td>
<td>0.028</td>
<td>0.038</td>
</tr>
<tr>
<td>C</td>
<td>0.073</td>
<td>0.055</td>
<td>0.110</td>
<td>0.062</td>
<td>0.088</td>
</tr>
<tr>
<td>D</td>
<td>0.038</td>
<td>0.033</td>
<td>0.044</td>
<td>0.034</td>
<td>0.042</td>
</tr>
<tr>
<td>E</td>
<td>0.220</td>
<td>0.129</td>
<td>0.344</td>
<td>0.148</td>
<td>0.311</td>
</tr>
<tr>
<td>F</td>
<td>3.541</td>
<td>1.686</td>
<td>5.885</td>
<td>2.050</td>
<td>5.553</td>
</tr>
<tr>
<td>G</td>
<td>0.078</td>
<td>0.070</td>
<td>0.089</td>
<td>0.071</td>
<td>0.082</td>
</tr>
<tr>
<td>H</td>
<td>3.415</td>
<td>2.124</td>
<td>5.246</td>
<td>2.613</td>
<td>4.364</td>
</tr>
</tbody>
</table>

Table 5.14: Data for P5/b, average of 100. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>T-avg</th>
<th>Min</th>
<th>Max</th>
<th>Min5</th>
<th>Max95</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.020</td>
<td>0.013</td>
<td>0.024</td>
<td>0.015</td>
<td>0.022</td>
</tr>
<tr>
<td>B</td>
<td>0.048</td>
<td>0.033</td>
<td>0.056</td>
<td>0.039</td>
<td>0.054</td>
</tr>
<tr>
<td>C</td>
<td>0.155</td>
<td>0.121</td>
<td>0.192</td>
<td>0.130</td>
<td>0.181</td>
</tr>
<tr>
<td>D</td>
<td>0.036</td>
<td>0.029</td>
<td>0.044</td>
<td>0.030</td>
<td>0.043</td>
</tr>
<tr>
<td>E</td>
<td>0.429</td>
<td>0.309</td>
<td>0.602</td>
<td>0.354</td>
<td>0.520</td>
</tr>
<tr>
<td>F</td>
<td>2.807</td>
<td>1.931</td>
<td>3.995</td>
<td>2.234</td>
<td>3.464</td>
</tr>
<tr>
<td>G</td>
<td>0.072</td>
<td>0.064</td>
<td>0.095</td>
<td>0.066</td>
<td>0.078</td>
</tr>
<tr>
<td>H</td>
<td>4.051</td>
<td>3.052</td>
<td>6.575</td>
<td>3.366</td>
<td>4.824</td>
</tr>
<tr>
<td>I</td>
<td>28.147</td>
<td>22.216</td>
<td>41.344</td>
<td>23.390</td>
<td>34.588</td>
</tr>
</tbody>
</table>

Table 5.15: Side by side Comparison of the best solvers from previous tests, average of 100. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>P1/d</th>
<th>P2/d</th>
<th>P3/d</th>
<th>P4/d</th>
<th>P5/p</th>
<th>P5/b</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.024</td>
<td>0.024</td>
<td>0.023</td>
<td>0.021</td>
<td>0.020</td>
<td>0.020</td>
</tr>
<tr>
<td>B</td>
<td>0.038</td>
<td>0.043</td>
<td>0.043</td>
<td>0.034</td>
<td>0.031</td>
<td>0.048</td>
</tr>
<tr>
<td>C</td>
<td>0.085</td>
<td>0.105</td>
<td>0.105</td>
<td>0.075</td>
<td>0.073</td>
<td>0.155</td>
</tr>
<tr>
<td>D</td>
<td>0.045</td>
<td>0.049</td>
<td>0.048</td>
<td>0.044</td>
<td>0.038</td>
<td>0.036</td>
</tr>
<tr>
<td>E</td>
<td>0.382</td>
<td>0.386</td>
<td>0.383</td>
<td>0.217</td>
<td>0.220</td>
<td>0.429</td>
</tr>
<tr>
<td>F</td>
<td>4.144</td>
<td>2.998</td>
<td>3.001</td>
<td>1.652</td>
<td>3.541</td>
<td>2.807</td>
</tr>
<tr>
<td>G</td>
<td>0.104</td>
<td>0.094</td>
<td>0.094</td>
<td>0.090</td>
<td>0.078</td>
<td>0.072</td>
</tr>
<tr>
<td>H</td>
<td>17.961</td>
<td>3.061</td>
<td>2.997</td>
<td>2.053</td>
<td>3.415</td>
<td>4.051</td>
</tr>
<tr>
<td>I</td>
<td>139.515</td>
<td>34.464</td>
<td>33.544</td>
<td>26.660</td>
<td>53.569</td>
<td>28.147</td>
</tr>
</tbody>
</table>
Figure 5.8: Comparison of methods/solvers at different N and K.

Figure 5.9: Comparison of methods/solvers at different N and K.
Table 5.16: Side by side comparison for 40 nodes, average of 10. Time in seconds.

<table>
<thead>
<tr>
<th>N</th>
<th>M</th>
<th>K</th>
<th>ss</th>
<th>sn</th>
<th>ns</th>
<th>nn</th>
<th>Inf</th>
<th>VC</th>
<th>P3/d</th>
<th>P4/d</th>
<th>P5/b</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>980</td>
<td>2</td>
<td>0.800</td>
<td>0.020</td>
<td>0.003</td>
<td>0.005</td>
<td>2</td>
<td>2</td>
<td>0.115</td>
<td>0.114</td>
<td>0.092</td>
</tr>
<tr>
<td>40</td>
<td>1004</td>
<td>20</td>
<td>0.400</td>
<td>0.008</td>
<td>0.000</td>
<td>0.001</td>
<td>8</td>
<td>6</td>
<td>16.123</td>
<td>9.482</td>
<td>16.340</td>
</tr>
<tr>
<td>40</td>
<td>790</td>
<td>40</td>
<td>0.400</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>8</td>
<td>2</td>
<td>222.365</td>
<td>131.005</td>
<td>127.157</td>
</tr>
</tbody>
</table>
5.4. Extensive test

Table 5.17: Definition of instances

<table>
<thead>
<tr>
<th>Name</th>
<th>N</th>
<th>M</th>
<th>K</th>
<th>ss</th>
<th>sn</th>
<th>ns</th>
<th>nn</th>
<th>Inf</th>
<th>VC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>75</td>
<td>10</td>
<td>0.500</td>
<td>0.010</td>
<td>0.010</td>
<td>0.002</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>20</td>
<td>198</td>
<td>10</td>
<td>0.500</td>
<td>0.001</td>
<td>0.004</td>
<td>0.001</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>20</td>
<td>189</td>
<td>20</td>
<td>0.500</td>
<td>0.001</td>
<td>0.004</td>
<td>0.001</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>30</td>
<td>738</td>
<td>2</td>
<td>0.500</td>
<td>0.020</td>
<td>0.020</td>
<td>0.010</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>30</td>
<td>447</td>
<td>15</td>
<td>0.300</td>
<td>0.002</td>
<td>0.002</td>
<td>0.001</td>
<td>10</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>30</td>
<td>454</td>
<td>30</td>
<td>0.300</td>
<td>0.001</td>
<td>0.001</td>
<td>0.001</td>
<td>10</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5.18: Filtered for infeasible problems with no VC, average of 10. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>P1/d</th>
<th>P2/d</th>
<th>P3/d</th>
<th>P4/d</th>
<th>P5/p</th>
<th>P5/b</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.095</td>
<td>0.107</td>
<td>0.108</td>
<td>0.076</td>
<td>0.071</td>
<td>0.153</td>
</tr>
<tr>
<td>B</td>
<td>0.703</td>
<td>0.423</td>
<td>0.423</td>
<td>0.232</td>
<td>0.235</td>
<td>0.421</td>
</tr>
<tr>
<td>C</td>
<td>9.695</td>
<td>3.141</td>
<td>3.062</td>
<td>1.653</td>
<td>3.463</td>
<td>2.910</td>
</tr>
<tr>
<td>D</td>
<td>0.103</td>
<td>0.095</td>
<td>0.095</td>
<td>0.090</td>
<td>0.078</td>
<td>0.075</td>
</tr>
<tr>
<td>E</td>
<td>34.007</td>
<td>3.091</td>
<td>3.298</td>
<td>2.213</td>
<td>3.574</td>
<td>4.452</td>
</tr>
</tbody>
</table>

Table 5.19: Definition of instances

<table>
<thead>
<tr>
<th>Name</th>
<th>N</th>
<th>M</th>
<th>K</th>
<th>ss</th>
<th>sn</th>
<th>ns</th>
<th>nn</th>
<th>Inf</th>
<th>VC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>77</td>
<td>2</td>
<td>0.800</td>
<td>0.060</td>
<td>0.060</td>
<td>0.010</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
<td>77</td>
<td>5</td>
<td>0.700</td>
<td>0.010</td>
<td>0.030</td>
<td>0.006</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
<td>76</td>
<td>10</td>
<td>0.500</td>
<td>0.010</td>
<td>0.010</td>
<td>0.002</td>
<td>10</td>
<td>8</td>
</tr>
<tr>
<td>D</td>
<td>20</td>
<td>289</td>
<td>2</td>
<td>0.600</td>
<td>0.050</td>
<td>0.040</td>
<td>0.005</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>E</td>
<td>20</td>
<td>191</td>
<td>10</td>
<td>0.500</td>
<td>0.001</td>
<td>0.004</td>
<td>0.001</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>F</td>
<td>20</td>
<td>185</td>
<td>20</td>
<td>0.500</td>
<td>0.001</td>
<td>0.004</td>
<td>0.001</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>G</td>
<td>30</td>
<td>734</td>
<td>2</td>
<td>0.500</td>
<td>0.020</td>
<td>0.020</td>
<td>0.010</td>
<td>10</td>
<td>9</td>
</tr>
<tr>
<td>H</td>
<td>30</td>
<td>454</td>
<td>15</td>
<td>0.300</td>
<td>0.002</td>
<td>0.002</td>
<td>0.001</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>I</td>
<td>30</td>
<td>440</td>
<td>30</td>
<td>0.300</td>
<td>0.001</td>
<td>0.001</td>
<td>0.001</td>
<td>10</td>
<td>4</td>
</tr>
</tbody>
</table>
Table 5.20: Filtered for infeasible problems, average of 10. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>P1/d</th>
<th>P2/d</th>
<th>P3/d</th>
<th>P4/d</th>
<th>P5/p</th>
<th>P5/b</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.023</td>
<td>0.024</td>
<td>0.022</td>
<td>0.021</td>
<td>0.021</td>
<td>0.021</td>
</tr>
<tr>
<td>B</td>
<td>0.038</td>
<td>0.043</td>
<td>0.043</td>
<td>0.035</td>
<td>0.033</td>
<td>0.048</td>
</tr>
<tr>
<td>C</td>
<td>0.094</td>
<td>0.100</td>
<td>0.098</td>
<td>0.077</td>
<td>0.075</td>
<td>0.161</td>
</tr>
<tr>
<td>D</td>
<td>0.042</td>
<td>0.048</td>
<td>0.049</td>
<td>0.044</td>
<td>0.038</td>
<td>0.036</td>
</tr>
<tr>
<td>E</td>
<td>0.391</td>
<td>0.375</td>
<td>0.374</td>
<td>0.216</td>
<td>0.215</td>
<td>0.395</td>
</tr>
<tr>
<td>F</td>
<td>4.264</td>
<td>3.362</td>
<td>3.280</td>
<td>1.876</td>
<td>3.720</td>
<td>2.725</td>
</tr>
<tr>
<td>G</td>
<td>0.112</td>
<td>0.094</td>
<td>0.094</td>
<td>0.089</td>
<td>0.076</td>
<td>0.074</td>
</tr>
<tr>
<td>I</td>
<td>250.116</td>
<td>37.707</td>
<td>38.850</td>
<td>31.471</td>
<td>44.452</td>
<td>30.107</td>
</tr>
</tbody>
</table>

Table 5.21: Definition of instances

<table>
<thead>
<tr>
<th>Name</th>
<th>N</th>
<th>M</th>
<th>K</th>
<th>ss</th>
<th>sn</th>
<th>ns</th>
<th>nn</th>
<th>Inf</th>
<th>VC</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>10</td>
<td>75</td>
<td>2</td>
<td>0.800</td>
<td>0.060</td>
<td>0.060</td>
<td>0.010</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>B</td>
<td>10</td>
<td>75</td>
<td>5</td>
<td>0.700</td>
<td>0.010</td>
<td>0.030</td>
<td>0.006</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>C</td>
<td>10</td>
<td>77</td>
<td>10</td>
<td>0.500</td>
<td>0.010</td>
<td>0.010</td>
<td>0.002</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>D</td>
<td>20</td>
<td>283</td>
<td>2</td>
<td>0.600</td>
<td>0.050</td>
<td>0.040</td>
<td>0.005</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>E</td>
<td>20</td>
<td>189</td>
<td>10</td>
<td>0.500</td>
<td>0.001</td>
<td>0.004</td>
<td>0.001</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>F</td>
<td>30</td>
<td>731</td>
<td>2</td>
<td>0.500</td>
<td>0.020</td>
<td>0.020</td>
<td>0.010</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>G</td>
<td>30</td>
<td>432</td>
<td>15</td>
<td>0.300</td>
<td>0.002</td>
<td>0.002</td>
<td>0.001</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>H</td>
<td>30</td>
<td>438</td>
<td>30</td>
<td>0.300</td>
<td>0.001</td>
<td>0.001</td>
<td>0.001</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 5.22: Filtered for feasible problems, average of 10. Time in seconds.

<table>
<thead>
<tr>
<th>Name</th>
<th>P1/d</th>
<th>P2/d</th>
<th>P3/d</th>
<th>P4/d</th>
<th>P5/p</th>
<th>P5/b</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>0.023</td>
<td>0.024</td>
<td>0.022</td>
<td>0.022</td>
<td>0.021</td>
<td>0.021</td>
</tr>
<tr>
<td>B</td>
<td>0.040</td>
<td>0.045</td>
<td>0.043</td>
<td>0.033</td>
<td>0.030</td>
<td>0.048</td>
</tr>
<tr>
<td>C</td>
<td>0.095</td>
<td>0.114</td>
<td>0.110</td>
<td>0.075</td>
<td>0.072</td>
<td>0.160</td>
</tr>
<tr>
<td>D</td>
<td>0.042</td>
<td>0.047</td>
<td>0.046</td>
<td>0.042</td>
<td>0.038</td>
<td>0.034</td>
</tr>
<tr>
<td>E</td>
<td>0.286</td>
<td>0.387</td>
<td>0.375</td>
<td>0.204</td>
<td>0.210</td>
<td>0.422</td>
</tr>
<tr>
<td>F</td>
<td>0.085</td>
<td>0.092</td>
<td>0.093</td>
<td>0.086</td>
<td>0.077</td>
<td>0.070</td>
</tr>
<tr>
<td>G</td>
<td>10.860</td>
<td>3.135</td>
<td>2.989</td>
<td>2.014</td>
<td>3.376</td>
<td>4.001</td>
</tr>
<tr>
<td>H</td>
<td>63.203</td>
<td>29.194</td>
<td>29.630</td>
<td>24.755</td>
<td>35.047</td>
<td>26.022</td>
</tr>
</tbody>
</table>

Table 5.23: Time of valid cycle search, average of 10. Time in seconds.

<table>
<thead>
<tr>
<th>N</th>
<th>M</th>
<th>K</th>
<th>ss</th>
<th>sn</th>
<th>ns</th>
<th>nn</th>
<th>Inf</th>
<th>VC</th>
<th>T-avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>40</td>
<td>1002</td>
<td>2</td>
<td>0.800</td>
<td>0.02</td>
<td>0.003</td>
<td>0.005</td>
<td>3</td>
<td>2</td>
<td>0.052</td>
</tr>
<tr>
<td>40</td>
<td>1012</td>
<td>20</td>
<td>0.400</td>
<td>0.000</td>
<td>0.000</td>
<td>0.001</td>
<td>4</td>
<td>2</td>
<td>0.501</td>
</tr>
<tr>
<td>40</td>
<td>791</td>
<td>40</td>
<td>0.400</td>
<td>0.000</td>
<td>0.000</td>
<td>0.000</td>
<td>9</td>
<td>1</td>
<td>1.063</td>
</tr>
</tbody>
</table>
Figure 5.10: Comparison of problem types for different methods/solvers at N-K=30-30.
Analysis of the results

In general it is not possible to say which model is the best regardless of solver used. The models perform very differently depending on the solver. For example, dualopt works best with model P4 and primopt works best with model P5. The solver that is most often the best is dualopt.

6.1 Preliminary test

- P1. The CPLEX dualopt solver was consistently the fastest.

- P2. Again the CPLEX dualopt solver was the best.

- P3. This one is a little more interesting since we also have the MMCF solvers included. Unfortunately the MMCF solvers tested did not perform very well. PPRN works for small networks only and even then is slower than CPLEX dualopt. Out of the MMCFB solvers the MD seems the fastest and most stable one. It stands up well against the CPLEX solvers for smaller networks, however it doesn’t perform well at all for larger networks. In general the CPLEX dualopt solver performed the best.

- P4. Like the previous ones CPLEX dualopt performed the best overall.

- P5. All the solvers are very close, except siftopt who stands out as a lot slower than the others. Overall CPLEX primopt performed the best. CPLEX baropt was a close second on the 20-20 network.

From this preliminary test the methods/solvers P1/d, P2/d, P3/d, P4/d, P5/p and P5/b were chosen for more extensive testing.
6.2 Extensive test

P4/d is fastest in most tests, but all the methods/solvers are very close in general, except for larger networks where P5/p is a tad slower and P1/d is much slower than the others. P2/d and P3/d are very close in all tests which isn’t so surprising since P2 and P3 are essentially the same method where in P3 some of the variables are shifted by one.

Looking at the table with 40 nodes we can see that P4/d and P5/b are fastest for the large $K = 40$ case, and that they are very close.

Overall the P4/d method/solver seems to be the fastest of the ones from these tests.

6.3 Valid Cycle search

Comparing the time taken to look for valid cycles (see Table 5.23) with the time taken to solve the problem (see Table 5.16) for the same size network we can see that the valid cycle search is more than 100 times faster in the largest case. The valid cycle search is therefor a good candidate for finding conflicts before solving the complete problem.

6.4 Different problem types

Looking at the difference between feasible and infeasible problems one may conclude that feasible problems are faster to solve than infeasible ones. Whether infeasible problems without valid cycles are slower to solve than infeasible problems in general is harder to say from this data though it seems to be the case for at least some of the methods/solvers tested.
Chapter 7

Summary and conclusion

7.1 Summary

In this thesis we have looked at a number of different methods and solvers that can be used in solving the ISPR problem. We have run tests on a varied range of routing patterns (SPG-files) in order to determine how the different combinations of methods and solvers behave in different scenarios.

When it comes to the solvers tested the results indicated that the linear CPLEX solvers primopt, dualopt and baropt outperform the specialized MMCF solvers and that the CPLEX solver siftopt performed very badly in almost all cases. The best method/solver combination overall was P4/d i.e. the first cycle basis formulation Model (3.4) solved with the CPLEX dualopt solver. Interestingly the P5/b combination was slightly faster on the largest network tested N-K = 40-40.

7.2 Limitations of this study

As with any kind of benchmark data the absolute time measurements reported are highly dependent on the hardware and software used and has little relevance on another testsystem. The tests carried out here were done with a particular CPLEX version running on a particular hardware/software platform and it does not necessarily hold that, the relative difference between the methods/solvers shown here, stays the same when running the tests on a completely different platform. Ideally one would like to include more platforms to test on to see what impact that could have on the results.

It is possible that including more data in the study may show a different result, but since the combinations of methods and solvers are quite large it becomes impractical to test much more data within a reasonable time.
There may be other optimization methods and algorithms available that are not included here which could prove to be faster at solving these types of problems.

7.3 Future work

As mentioned in 7.2 it could be interesting to run tests on other platforms and also to test some other general solvers besides CPLEX in order to see if the same results hold in another setting.

It could also be interesting to analyze why the different models tested here behave the way they do. Especially it would be interesting to investigate why the model P5 is fastest to solve using CPLEX baropt as opposed to CPLEX dualopt for the other models.
Bibliography


http://www-eio.upc.es/~jcastro/index.html

http://sorsa.unica.it/index.html

http://publib.boulder.ibm.com/infocenter/cosinfoc/v12r2/index.jsp
The SPG file format

The file format used for storing SP-graphs is described here.

A.1 Overview

The file is processed line by line. On every line each token is separated by whitespace. The % character is used for comments. Everything from the % character to the end of the line is discarded. Any empty lines (line with no tokens) are discarded.

The first line gives the size of the problem by three integers N, M and K where N is the number of nodes in the network, M is the number of arcs in the network and K is the number of SP-graphs. Then follows the M arcs in the network. One arc per line, the startnode of the arc first and the endnode second. The nodes are numbered from 0 to N-1 and consequently the arcs are also zero based. After the arc list follows the K SP-graphs, where each SP-graph follows the following format. The first line of each SP-graph starts with SPG k where k specifies the destination of this SP-graph. After that follows M decimal numbers. One number per line, each number specifying the property of this arc in the current SP-graph. The number 1.0 means the arc is a shortest-path arc, 0.0 means the arc is a non-shortest-path arc and 0.5 means the arc is unspecified or a don’t-care arc.

A.2 Sample file

% #nodes #arcs #SPGs
6 6 2
% Arcs
0 1
0 4
1 2
3 2
3 4
4 5

% SPG_1: dest
SPG 5
1 % 0 1 - shortest
0.5 % 0 4 - don’t care
1 % 1 2 - shortest
0.5 % 3 2 - don’t care
1 % 3 4 - shortest
1 % 4 5 - shortest

% SPG_2: dest
SPG 4
0.5 % 0 1 - don’t care
1 % 0 4 - shortest
0.5 % 1 2 - don’t care
1 % 3 2 - shortest
0 % 3 4 - non-shortest
0.5 % 4 5 - don’t care

Figure A.1: The corresponding graph of the two SPGs. Shortest-path arcs of the first (solid) and the second (dotted) SPG and the non-shortest-path arc ending with a dot.
The main program

The main program used to do all the tests is described here.

**B.1 General description**

The main program is called ispr and is programmed in C++ using the gcc g++ compiler. It is meant to run on a Unix system but can probably without much of a rewrite be made to run on other operating systems. It is a command-line program that takes a set of parameters as command line options to control the behaviour of the program. One of the parameters supplied is one or more SPG-files that are to be processed.

The general structure of the program is that it reads an SPG-file into memory, it then creates the necessary data structures used for passing the problem on to the different solvers, which are always one of the CPLEX solvers. The other MMCF solvers used are completely separate programs downloaded from their respective sites see [7, 6]. After the solver has solved the problem the result of the solver is printed out along with the time it took to solve the problem. In order to use the CPLEX software one has to first obtain a licence. There are however academic licenses one can get as a teacher or a graduate student.

**B.2 Commandline parameters**

Usage: [-g infile] [-m 1 2 3 4 5 6 7] [-b] [-d] [-p 1 2 3 4 5] inputfiles

- `g` parameterfile
- `b` for benchmark
- `d` for dual presolver
Argument to \(-m\) one of:
1: ISPR cplex
2: ISPR dual cplex
3: ISPR dual cplex with transformation
4: Cyclebase cplex
5: Cyclebase cplex simplified matrix
6: Cyclebase lagrange
7: Valid cycle search

Argument to \(-p\) one of:
1: primopt
2: dualopt
3: baropt
4: hybbaropt
5: siftopt

The \(-g\) option is used to invoke the instance generator to create random SPG-files. It reads its settings from a parameter file with the following format:

```
% parameter file to generate instances of spg files
% Number of instances to create
100
% nodes spgs arc probability (1.0 = complete graph)
20 4 0.8
% pSP_SP pSP_NonSP pSP_DontCare pNonSP_SP pNonSP_NonSP pNonSP_DontCare
0.8 0.02 0.01 0.003 0.005 0.2
```

As in the SPG-file format a % is used for comments. The number of instances to create is the number of files it will produce. The rest of the parameters are described earlier in this document. The DontCare parameters are not used by the program but has to be present in this file since the program reads them in and expects them to be there. They can be set to anything however.

The command

```
./ispr -m 1 -p 3 myspgfile.spg
```

will run the method PI using the CPLEX baropt solver on the SPG-file myspgfile.spg.

The code for the program is available at [http://www.mai.liu.se/~mical/ISPR_Computations/](http://www.mai.liu.se/~mical/ISPR_Computations/)
Copyright

The publishers will keep this document online on the Internet - or its possible replacement - for a period of 25 years from the date of publication barring exceptional circumstances.

This document is free from copyright of any kind and can be used by anyone as they like.

For additional information about the Linköping University Electronic Press and its procedures for publication and for assurance of document integrity, please refer to its WWW home page: http://www.ep.liu.se/

Upphovsrätt

Detta dokument hålls tillgängligt på Internet - eller dess framtida ersättare - under 25 år från publiceringsdatum under förutsättning att inga extraordinära omständigheter uppstår.

Detta dokument är ej skyddat av upphovsrätt och kan användas av vem som helst på vad sätt de finner lämpligt.

För ytterligare information om Linköping University Electronic Press se förlagets hemsida http://www.ep.liu.se/

© 2010, Richard Sandberg