Underground Mine Plan Optimisation

David Whittle

ORCID 0000-0002-9279-5896

Submitted in total fulfilment of the requirements of the degree of

Doctor of Philosophy

School of Engineering, Department of Mechanical Engineering
THE UNIVERSITY OF MELBOURNE

October 2019
Abstract

This thesis addresses several topics relating to the planning of underground mines, with a focus on underlying mathematical models.

Some mineral resources are mined by a combination of open-pit and underground mining methods and a decision must be made as to which methods to apply to different parts of the resource. This is called the transition problem, to which Chapter 3 is dedicated. My contribution, is a graph theory-based optimisation model that solves the transition problem efficiently for large data sets with various geometric constraints.

The remainder of this thesis focuses on the optimisation of underground mine plans. My major contribution concerns a sub-problem that is framed as a Prize collecting Euclidean Steiner tree problem. This is a generalisation of the Euclidean Steiner tree problem. A problem instance is a set of points in the plane, each with a point weight. Of interest are networks on some subset of these points. Networks can include additional vertices called Steiner points if their inclusion yields a shorter network. The value of the network is calculated as the sum of the point weights in a selected subset, less the sum of the lengths of the edges in the network connecting these points. The question is: What selection of points and connected network has the highest value?

There is a great deal of literature on the Euclidean Steiner tree problem and efficient solutions are available. In contrast, there are no solutions to the prize collecting generalisation, only an approximation scheme. I have developed an algorithmic framework for the problem (Chapter 6). Included are efficient methods to determine a subset of points that must be in every solution (ruled in) and a subset of points that cannot be in any solution (ruled out). Also included are methods to generate and concatenate full Steiner trees. My generation and concatenation approaches are elaborations on existing equiva-
lent functions for the simpler Euclidean Steiner tree problem.

Two of the ruling out methods require new universal geometric constants. For one of these, I have been able to prove an infimum. This is a strong result. The proof for this infimum is long, and a chapter is dedicated to it (Chapter 7: A universal constant for replacement argument A). For the other universal constant, I have a proof for a lower bound, and a conjecture for an infimum. This second universal constant also has its own chapter (Chapter 8: A universal constant for replacement argument B).

Finally, I have also developed two new decompositions of the underground mine planning problem (Chapters 4 and 5). These decompositions are at an early stage and I plan to apply myself to their further development in the future.
Declaration

This is to certify that

1. the thesis comprises only my original work towards the PhD,
2. due acknowledgement has been made in the text to all other material used,
3. the thesis is less than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

______________________________
David Whittle, October 2019
This page intentionally left blank.
Preface

This thesis captures the results of my four years of research. Some of the results have been published and two papers are in preparation. My published papers concern the transition problem which is the topic covered by Chapter 3 of this thesis:


The two papers that are in preparation for submission to a mathematics journal concern the prize collecting Euclidean Steiner tree problem. The first is a shortened version of Chapter 6 of this thesis. The second is a shortened version of Chapters 7 and 8 of this thesis:

- D. Whittle, M. Brazil, P.A. Grossman, H. Rubinstein and D. A. Thomas, “Solving the prize collecting Euclidean Steiner tree problem,”

- D. Whittle, M. Brazil, P.A. Grossman, H. Rubinstein and D. A. Thomas, “Two universal constants for use in solving the prize collecting Euclidean Steiner tree problem,”

I am the grateful recipient of the 2015 Gilbert Rigg Scholarship, which has supported my research for four years. I am also grateful to have been awarded the 2016 John Collier Scholarship to support my research-related travel.
This page intentionally left blank.
Acknowledgements

WITH around 25 years of minerals industry experience, I came to this PhD program with some industry problems in mind. I needed guidance on how to formulate these problems in mathematical terms, and on how to develop and prove the mathematical solutions. Mining is a global industry and the field of mathematics is comprised of a vast range of specialities, serviced by a global academic community. To my great good fortune, it turns out that four academics who are internationally recognised for their work on Steiner trees, reside in my home town of Melbourne: Professor Doreen Thomas, Professor Hyam Rubinstein, Associate Professor Marcus Brazil and Doctor Peter Grossman. In addition, they have achieved considerable success in applying their mathematical know-how in the mining domain. I am extremely grateful and honoured to have had these distinguished academics as my supervisors. I thank them for their consistent generosity in providing advice, support and encouragement.

I’d also like to thank the “Labsters”: colleagues in my lab, from Australia, Chile, China, India, Iran and Sri Lanka. At times I felt like an uncle to a United Nations of talented people. I’m grateful for the mutual support and encouragement we’ve provided to each other. It has been most enjoyable learning about many of the Labsters’ culture and interests.

Finally, my PhD journey would never have begun, nor completed, without the unwavering love and support of my wife of 26 years, Julie Garner and our three sons Adam, Liam and Max. I thank them all from the deepest place in my heart.
This page intentionally left blank.
Contents

1 Introduction 1

2 Underground mine plan optimisation 3
  2.1 Underground mining 3
  2.2 Decomposition 5
  2.3 Mining method selection 7
  2.4 Stope polyhedron optimisation 9
  2.5 Cave polyhedron optimisation 12
  2.6 Shaft, decline, ore pass and level optimisation 14
  2.7 New decompositions 18

3 Transition from open pit to underground mining 19
  3.1 Introduction 19
  3.2 New optimisation method 24
    3.2.1 Pit optimisation model 24
    3.2.2 New model 28
  3.3 Reduction of an ODOC problem to an MGC problem 34
  3.4 Inclusion of non-trivial strongly connected subgraphs 36
    3.4.1 Introduction 36
    3.4.2 SCSs and their properties in closures 37
    3.4.3 Strongly connected components 37
    3.4.4 Functions 38
    3.4.5 Procedures 39
    3.4.6 Computational complexity 42
  3.5 Experimental results 43
  3.6 Discussion 46
  3.7 Conclusions 47

4 Underground mine with shaft access 49
  4.1 Introduction to the decomposition 49
  4.2 Potential stope or cave polyhedron optimisation 50
    4.2.1 Existing methods 50
    4.2.2 A proposed new method 51
  4.3 Level development and level block selection 54
  4.4 Shaft depth optimisation 57
4.5 Discussion and conclusions ........................................... 57

5 Underground mine with decline access .......................... 59
  5.1 Introduction to the decomposition ............................... 59
  5.2 Level development and level block selection ................ 60
  5.3 Decline optimisation ........................................... 60
  5.4 Depth optimisation ............................................. 61
  5.5 Discussion and conclusions ..................................... 61

6 Prize collecting Euclidean Steiner trees ..................... 63
  6.1 Introduction ................................................... 63
  6.2 Minimum spanning trees and minimum Steiner trees ........ 66
    6.2.1 Melzak-Hwang Algorithm ................................ 68
  6.3 Prize collecting Steiner trees in graphs ..................... 72
  6.4 Prize collecting Euclidean Steiner tree problem overview and preliminaries 72
  6.5 Properties of maximum PCESTs ................................ 74
  6.6 Naïve approach to finding a solution to the PCEST problem 77
  6.7 Reducing the number of possible terminals ................... 77
  6.8 Ruling in ..................................................... 78
    6.8.1 Points close together .................................... 78
    6.8.2 The merging and ruling in algorithms for the rooted PCEST problem 80
  6.9 Introduction to ruling out .................................... 84
  6.10 Pre-screening points to be subject to ruling out tests ...... 84
  6.11 Ruling out using an MST ...................................... 85
    6.11.1 Algorithm for ruling out using an MST .................. 86
  6.12 Ruling out using a convex hull ................................ 91
    6.12.1 PCEST Steiner Hulls ...................................... 91
    6.12.2 The ruling out method .................................... 92
    6.12.3 Algorithm for ruling out using a convex hull ........... 95
    6.12.4 Supporting Line 2 ........................................ 97
  6.13 Ruling out based on local connections ....................... 98
    6.13.1 Steiner tree $S_1$ ........................................ 102
    6.13.2 Disk $D$ and Rubin points ................................ 103
    6.13.3 Steiner Tree $S_2$ and $S$ ................................ 105
    6.13.4 Chord length definitions ................................ 109
    6.13.5 Trees $T$, $U$ and $V$ .................................... 112
    6.13.6 Replacement argument preliminaries ...................... 113
  6.14 Replacement argument A ...................................... 114
    6.14.1 Algorithm to apply replacement argument A ............. 116
    6.14.2 Experimental results for Replacement Argument A ....... 118
  6.15 Replacement argument B ...................................... 119
    6.15.1 A method to calculate an upper bound $U_e$ for a given $N$ and $n_q$ 122
    6.16 Joint application of replacement arguments A and B ....... 123
    6.16.1 Illustration of the application of replacement arguments A and B 123
  6.17 Full Steiner tree generation for the PCEST problem ........ 125
This page intentionally left blank.
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1</td>
<td>Results for five cases to test the transition problem solution</td>
<td>45</td>
</tr>
<tr>
<td>6.1</td>
<td>Ruling in random point generation settings</td>
<td>83</td>
</tr>
<tr>
<td>6.2</td>
<td>Ruling in trials</td>
<td>83</td>
</tr>
<tr>
<td>6.3</td>
<td>Small test model details for ruling out using an MST</td>
<td>90</td>
</tr>
<tr>
<td>6.4</td>
<td>Larger test model general settings for ruling out using an MST</td>
<td>90</td>
</tr>
<tr>
<td>6.5</td>
<td>Larger test model results for ruling out using an MST</td>
<td>90</td>
</tr>
<tr>
<td>6.6</td>
<td>Replacement arguments for ruling out on local connections</td>
<td>114</td>
</tr>
<tr>
<td>6.7</td>
<td>Test model details for replacement argument A</td>
<td>118</td>
</tr>
<tr>
<td>6.8</td>
<td>Test model results for replacement argument A</td>
<td>118</td>
</tr>
<tr>
<td>6.9</td>
<td>Predicates defined for conditions of interest (Lune and disk properties)</td>
<td>128</td>
</tr>
<tr>
<td>6.10</td>
<td>All combinations of predicates (Lune and disk properties)</td>
<td>129</td>
</tr>
<tr>
<td>6.11</td>
<td>Claims (Lune and disk properties)</td>
<td>130</td>
</tr>
<tr>
<td>6.12</td>
<td>Applications of Theorems 6.17.5 to 6.17.8 (Lune and disk properties)</td>
<td>131</td>
</tr>
<tr>
<td>6.13</td>
<td>TRLs for contributions in this chapter and supporting Chapters 7 and 8</td>
<td>136</td>
</tr>
<tr>
<td>7.1</td>
<td>Hierarchy of cases for Lemma 7.4.4</td>
<td>159</td>
</tr>
<tr>
<td>7.2</td>
<td>L or R insertions into each edge of the RLLL group-1 simplest topology</td>
<td>186</td>
</tr>
<tr>
<td>7.3</td>
<td>L or R insertions into each edge of the RLLL group-2 simplest topology</td>
<td>195</td>
</tr>
<tr>
<td>8.1</td>
<td>Some previously defined terms relevant to this chapter</td>
<td>198</td>
</tr>
<tr>
<td>8.2</td>
<td>Derivatives for expansions and contractions.</td>
<td>220</td>
</tr>
<tr>
<td>A.1</td>
<td>Symbols used in Chapter 3</td>
<td>231</td>
</tr>
<tr>
<td>A.2</td>
<td>Abbreviations used in Chapters 6 to 8</td>
<td>232</td>
</tr>
<tr>
<td>A.3</td>
<td>Symbols $D$ to $R^B$ used in Chapters 6 to 8</td>
<td>233</td>
</tr>
<tr>
<td>A.4</td>
<td>Symbols $S$ to $W_N$ used in Chapters 6 to 8</td>
<td>234</td>
</tr>
</tbody>
</table>
This page intentionally left blank.
List of Figures

2.1 A shaft, ore pass and surrounding infrastructure .................. 16

3.1 Arcs representing block dependencies and a closure .................. 28

3.2 A simple model for open pit optimisation with open pit values ... 28

3.3 Example of \(X^p\) and \(X^v\) sets of vertices and \(\gamma\) arcs 30

3.4 Depiction of various NSCs to model the crown pillar .............. 32

3.5 Blocks: optimal pit; crown pillar; available for underground mining 33

3.6 Example MGC model for the ODOC problem ......................... 36

3.7 Procedure A and B .................................................. 40

4.1 Illustration of a block model ........................................ 52

4.2 The use of surfaces to model foot walls and hanging walls .... 53

4.3 A level, shaft location and level blocks ................................ 55

4.4 A minimum Steiner tree on the shaft and level blocks ............. 56

6.1 Melzak-Hwang merging steps ........................................ 71

6.2 Melzak-Hwang reconstruction steps .................................. 71

6.3 Two different maximum PCESTs on the same set of points ....... 75

6.4 Two maximum rooted PCESTs with different terminal sets ...... 76

6.5 Two maximum un-rooted PCESTs with different numbers of terminals 76

6.6 Two maximum rooted PCESTs with different terminal sets ....... 76

6.7 MST on points in the test model ................................... 89

6.8 Artefacts relevant to Theorem 6.12.4 ................................ 93

6.9 Artefacts relevant to Algorithm 4 and supporting line 2 .......... 95

6.10 Sub-tree \(S'\) of \(S\) .................................................. 100

6.11 Alternate tree \(S''\) .................................................... 102

6.12 The Simpson line \(e_1e_2\) for alternate tree \(S''\) ................. 103

6.13 Example of an MSTT \(S^1\) showing the disk \(D\) centred on \(n_q\) . 104

6.14 \(S^2\) with Rubin points .............................................. 105

6.15 Example of a bridge .................................................. 107

6.16 Example of a non-bridge .............................................. 108

6.17 One Steiner point in the path ........................................ 109

6.18 Two Steiner points in the path .................................... 110

6.19 Three Steiner points in the path ................................... 110

6.20 Four Steiner points in the path .................................... 111

6.21 MSTT \(S\) on \(R \cup \{n_q\}\) and MST \(T\) on \(R\) ......................... 112
Chapter 1
Introduction

In this introduction, I first give a brief overview of the mining industry, and the role of mathematical optimisation in improving asset values in the industry. I then describe the two opportunities to further apply mathematical optimisation, which I have pursued in my research. Finally I provide an overview of the structure of this dissertation.

Mining is a global industry present in most countries and on every continent other than Antarctica. It is a substantial industry, producing around 17 billion tonnes of mineral fuels, metals and industrial minerals annually (Reichl, Schatz & Zsak [1]). The top 40 mining companies have combined market capitalization of around USD900b (PWC [2]).

The underlying assets of all mining businesses are mineral resource rights, mines to exploit these resources and downstream facilities to process and transport the products from the mines. Mathematical optimisation has long been used to increase the value of these underlying assets, and with great success: New optimisation methods applied to one or several aspects of mine planning and scheduling commonly add double-digit percentage improvements in asset values. Never the less, there are still many opportunities to develop new mathematical optimisation methods for the industry, and my research has focused on two such opportunities.

The vast majority of minerals are extracted by methods falling into the categories of open pit (an open excavation from the surface) and underground (a network of tunnels and/or shafts giving access to the minerals underground). The application of optimisation to open pit mine design is quite mature, a situation discussed in the introduction to Chapter 3. The application of optimisation to underground mine design is less mature, representing an opportunity: to devise methods that optimise the layout of underground
mines. Chapter 2 [Underground mine plan optimisation] provides a brief introduction to underground mining, followed by a review of the literature on the use of optimisation in the planning of underground mines.

Some minerals are extracted by a combination of open pit and underground mining methods and a choice must be made as to where to end one method and to commence the other. This is often referred to as the transition problem and I have developed a new model and method to solve this problem. This is my first contribution and is presented in Chapter 3 [Transition from open pit to underground mining].

My second contribution relates exclusively to underground mine plan optimisation and consists of two decomposition models:

- Chapter 4 [Underground mine with shaft access]
- Chapter 5 [Underground mine with decline access]

Each of these decompositions include optimisation steps called level development and polyhedron selection. These optimisation steps can be transformed into Prize collecting Euclidean Steiner tree (PCEST) problems. PCEST problems have received very little attention in the literature. The most mathematically interesting contribution that I make in this thesis is the development of an algorithmic framework to solve two variants of the PCEST problem (rooted and non-rooted) in the following chapters:

- Chapter 6 [Prize collecting Euclidean Steiner trees]
- Chapter 7 [A universal constant for replacement argument A]
- Chapter 8 [A universal constant for replacement argument B]

I conclude with some guidance on reading order. Whilst the most straightforward approach is to read the thesis from start to finish, the following comments may be of assistance to the reader who prefer to take a different path:

- Chapters 2, 4 and 5 will make most sense if read in order.
- Chapter 3 is self-contained.
- Chapters 6, 7 and 8 should be read in order, but are otherwise collectively self-contained, relying on the earlier chapters only for industrial context.
Chapter 2
Underground mine plan optimisation

This chapter provides a brief introduction to underground mining, followed by a review of the literature on the use of optimisation in the planning of underground mines. Each contribution to the literature generally deals with one or a few elements of the underground mine planning problem. Some contributions connect a few elements together; suggest a decomposition of the main problem; or suggest prioritisation of the various elemental solutions. It is for this reason that the literature review is organised into different optimisation elements and groups of elements, and that decomposition is an underlying theme. The last section in this chapter provides a roadmap to three other chapters in this thesis, which include two new decompositions of the underground mine plan optimisation problem, together with solutions to the elements in each of the decompositions.

2.1 Underground mining

For the benefit of readers unfamiliar with underground mining, general information about stoping and caving underground mining methods is provided. These are not the only classes of underground mining methods available to miners, but they are the two that are of interest in my research, specifically in this chapter, and in Chapters 4 and 5. Readers interested in more details about underground mining can refer to Hamrin [3].

In a stoping mine, the ore (defined on page 5) is blasted with explosives, and removed by mechanical means from constructed drawpoints (places where ore can be loaded and removed). The voids that result from ore removal may be left open, or they may be backfilled with waste and cement in order to avoid weakening of the surrounding rock. The drawpoints in a stoping mine are often distributed in a number of horizontal or near-horizontal levels with the distance separating these levels depending on the specifics of the orebody and the variant of the stoping method selected. In the literature, the term
**Underground mine plan optimisation**

*stope* is used in a variety of ways. It can mean the overall polyhedron encompassing a contiguous set of ore. It can also mean that part of the overall polyhedron that exists between defined levels, or some subsidiary elements thereof.

In a *caving mine*, gravity is relied upon to collapse the rock down to drawpoints where it is extracted. Since the ground collapses in (sometimes all the way up to the Earth’s surface), there should not be any voids left over. Depending on the variant of caving method selected, there may be drawpoints on one or several levels. For example, in the *block caving* method, drawpoints are all on one level, whereas with *sublevel caving*, drawpoints are distributed across many regularly spaced levels.

In either stoping or caving mines, drawpoints are accessed by networks of shafts and tunnels. They can be roughly divided into categories of vertical and horizontal access, although some special purpose tunnels belong in both categories.

A *shaft* provides vertical access to each level. Winches are used to haul people, matériel and rock up and down the shaft in cages and skips. A *decline* (also called a *ramp*) is another method for providing vertical access. It is a steeply inclined underground road leading from a *Portal* (entrance to the mine) to the workings below. A decline is traversed by wheeled vehicles. These vehicles provide transportation for people, matériel and rock. A decline not only provides vertical access, but it also provides flexibility when it comes to accessing widely dispersed parts of an orebody in the horizontal plane.

A *level* is a horizontal or nearly horizontal network of tunnels connecting drawpoints to a shaft or a decline. The network of tunnels provide access to the orebody for drilling, blasting and extracting the ore.

The three-dimensional shapes that are targeted for mining are called stopes in a stoping mine and caving blocks in a caving mine, however to avoid confusion with regular blocks discussed elsewhere, the terms *stope polyhedron* and *caving polyhedrons* will be used. A collection of blocks in a regular block model can be used to represent a stope polyhedron or a caving polyhedron.
I conclude this section with a selection of definitions for terms used in chapters 2 to 5. The definitions are principally sourced from Hamrin [3]:

- **cross-cut**: A horizontal or near-horizontal tunnel connecting a shaft or decline to access points to the orebody.
- **cut-off**: A grade for rock, above which one thing is done and below which another thing is done.
- **dilution**: \( \frac{T(W)}{T(M)} \), where \( T(W) \) and \( T(M) \) are tonnes of waste and rock mined respectively.
- **drift**: A horizontal opening near an orebody, generally parallel to the strike.
- **grade**: The relative quantity or the percentage of an element in rock. It may be expressed as a percentage or as a ratio such as \( \text{grams/tonne} \) for gold.
- **head grade**: The grade of rock as it is extracted from the mine.
- **ore**: Rock that can be profitably extracted from the earth. The profit derives from the inherent value of the contained product (gold, copper, diamonds etc.), less the cost of extracting the product from the earth and delivering it to the market.
- **ore pass**: A vertical or near-vertical shaft down which ore is dropped from one or more levels. The ore is collected at the bottom of the ore pass for delivery to the surface via a shaft or decline.
- **pillar**: A column or section of rock in between mined areas (to provide structural integrity to mined areas and infrastructure).
- **strike**: Main horizontal course or direction of a mineral deposit.
- **sublevel**: System of horizontal underground workings between levels used for access to stoping areas where required for ore production.
- **waste**: Rock that does not meet the definition of ore.

### 2.2 Decomposition

A long-term strategic plan for an underground mine is comprised of many elements. These elements include, but are not limited to: mining method selection; the polyhedral designs for all potential stopes and/or caves; the selection of stopes and/or caves; design of tunnels and shafts to provide for access to and the extraction of ore; and a long-term schedule for construction of the mine and ore extraction.

Not surprisingly, for any given orebody, there is an almost infinite combination of solutions for the aforementioned elements. There is no known procedure to globally optimise everything simultaneously. Accordingly, decomposition and prioritisation are necessary in solving the problem:
• Decomposition – In some cases, it is possible to decompose a problem in such a way as to allow a procedure to give a globally optimal solution. In the underground mine planning problem, decomposition does not normally allow for such guarantees. Instead, iteration is required to achieve a solution in which each element of the decomposed problem is solved, such that the combination of solutions is internally consistent. Decomposition can be by element, or by category. An example of elemental decomposition is to separate the optimisation of mining polyhedrons from the optimisation of the design of tunnels and shafts. An example of category decomposition is to separate the optimisation of mining polyhedrons into the optimisation of caving polyhedrons or the optimisation of stope polyhedrons. The optimisation methods used for caves and stopes can be quite different, and so for an orebody that might be amendable to either, both methods should be modelled and the best one chosen.

• Prioritisation – An order can be applied to the solving of elemental sub-problems. In 1981 Trotter & Goddard [4] succinctly stated the approach in the context of optimising sublevel caving layouts:

If the least sensitive parameters [elements] can be identified and then set to values commonly used in practice, optimisation can be performed using only the more sensitive parameters.

In their case, they were able to combine the optimisation of four sensitive elements together. In other cases, the prioritisation might require the treatment of elements separately, starting with the most sensitive. It is then usual to re-examine the less sensitive elements, checking for fitness and consistency.

A generic decomposition of the underground mine plan optimisation procedure by element (and category) is as follows:

1. mining method selection
2. mining polyhedrons (stopes; caves)
3. haulage network (shaft; decline; ore pass, level)
4. ventilation and other ancillary network
5. schedule
In this thesis the focus is on the first, second and third elements in the above decomposition.

2.3 Mining method selection

Much of the recent literature on underground mining method selection references the Nicholas technique, described in a series of papers in the 1980s. In [5] Nicholas identified the classes of information concerning the orebody that need to be considered in selecting a mining method, and for each class, he discretized a selection of measures. For example, in the class of geometry of deposit, he trichotomized depth below surface as shallow (< 150 m); intermediate (150 m – 600 m) and deep (> 600 m). He then associated various mining methods with these measures. For example, he associated four mining methods, including sublevel caving and block caving, with massive orebodies where the ore is weak. The Nicholas technique then involves selecting a few of the indicated mining methods for detailed engineering and economic analysis. In [6] Nicholas presents a multi-criteria decision-making model which accumulates scores for ten mining methods on four discretized measures, where individual suitability scores have been determined. As for [5], the top scoring methods are then subject to more detailed analysis.

Later literature on mining method selection focuses on refining the measures and suitability scores (for example, see the UBC method in Miller-Tait, Panalkis & Poulin [7]), and in bringing more sophistication to the decision-making process by reconciliation of expert opinion (for example see Fu et. al. [8]). For a wider review of mining method selection refer to Kant et. al. [9].

The common feature of all the techniques is to first of all eliminate technically infeasible mining methods for a given orebody, and then to determine which of the remaining methods can be most profitably applied. In terms of decomposition of the underground mine optimisation problem, these authors use method selection as a first step, with at most a few variants to be analysed in more detail through other elements of the decomposition.

However, it is observed that for a given orebody, the application of different mining
methods will lead to different outcomes in a number of areas that are easy to model in other elements of a decomposition as follows:

- **Development costs**: The spacing between levels is a major contributor to development costs, since each level must provide access to, and drawpoints for the full width of the mined orebody. For a given orebody, the smaller the spacing between levels, the more levels there are, and the greater their cumulative costs. Level spacing varies significantly by mining method. For example, in the block caving method, drawpoints are all on one level, hundreds of metres below the surface. In contrast, spacing between levels for cut and fill stoping can be around 25 metres (Bullock [10]).

- **Recovery**: The degree of mining selectivity afforded by a mining method is particularly influential to recovery (the proportion of ore intended to be mined that is actually mined). For example, stoping methods involve drilling and blasting, and the miner can select exactly what is drilled and blasted. This gives greater control over what is recovered from drawpoints than in a caving mine. A further important contributor to the degree of selectivity available is the size of the mining envelopes, be they in stopes or caves. Smaller mining envelopes convey more selectivity and consequently, better recovery.

- **Dilution**: Selectivity also affects dilution (defined on page 5). Mining and processing costs, which represent the majority of total costs for a mine, are roughly proportional to the volume of rock mined. Dilution can be a major issue, for example, Pakalnis, Poulin & Hadjigeorgiou [11] found that a large proportion of open stope operations in Canada experienced dilution in excess of 20%; Trotter & Goddard [4] state that dilution can be up to 40% in sublevel caving mines.

- **Mining costs**: Highly selective mining methods are associated with higher unit costs of production, whereas bulk mining methods such as block caving and sublevel caving are associated with lower unit costs of production.

- **Production rate**: Larger orebodies generally warrant higher production rates (the details of this claim are beyond the scope of this thesis to cover), and some mining methods are more amenable to high production rates than others. For example,
block caving is amenable to very high production rates, whereas some highly selective stoping methods are not. The production rate can also place a lower limit on the number of open faces required, given that production is limited by the size and number of pieces of equipment that can be deployed in a limited area. Consequently, the production rate can provide a lower limit on the number of levels and therefore, the level separation (Zambo [12]).

The relationship between method selection and the aforementioned factors suggests the possibility of supporting method selection with a greater use of other elements of the decomposition. In simple terms, this would mean letting a wider range of potential methods pass the initial method selection, to be further assessed in other elements of the decomposition on economic grounds, ultimately better informing the final method selection decision.

2.4 Stope polyhedron optimisation

In 1995 Alford [13] developed the first three-dimensional stope optimisation procedure dubbed *floating stope*. The procedure, implemented in commercial mine planning software Datamine, employs a heuristic to identify high-value stope envelopes that comply with various requirements, including a minimum stope polyhedron, a cut-off grade that differentiates ore from waste and a minimum head grade. The floating stope method is however a heuristic and cannot guarantee optimal results. In particular, there is potential for the value of ore to be double counted in the valuation of a stope polyhedron. For example, two different polyhedrons may share a valuable block and each may be economic, but their union is not necessarily economic. When double counting occurs, both polyhedrons are selected.

In 1999 Thomas & Earl [14] reported on development of an alternative to Alford’s floating stope method, that deals with the double counting issue, also making it possible to constrain stope generation with user-defined levels. Furthermore, the method simultaneously delivers a mining sequence that the authors claim maximises net present value (NPV). However, the method uses a simple heuristic to generate sequences that,
while likely to deliver relatively high NPV, cannot guarantee an optimal result. In 2004 Ataee-pour [15] also developed a heuristic for stope envelope optimisation dubbed maximum value neighbourhood. Ataee-pour’s method does not suffer from the double counting issue.

In 2009 Alford & Hall developed a stope shape annealing process [16] which can be summarised as follows:

1. generation of a high-quality stope seed
2. rapid annealing
3. aggregation of stope shapes to identify groups of stope shapes that satisfy mineability criteria

The method is capable of accepting a variety of user-defined constraints, including the imposition of fixed separations of levels and pillar widths. The development was supported by AMIRA (Australian Mineral Industries Research Association; refer to sections on project P884 in [17], [18], [19] and [20]) and was later extended under another AMIRA project (refer to sections on P1037 in [20], [21], [22], [23], [24] and [25]) to handle more complex stope shapes. This method is the most widely deployed method, being available on several mining software platforms.

In 2014 Sandanayake, Topal & Asad [26] implemented a new heuristic solution to the stope layout problem. A square template of user-defined dimension is used to represent a minimum stope component. Using that template, Sandanayake’s method generates possible stopes; that is, any fit of the template across blocks in the block model such that the sum of the values of the blocks is positive. The next step combines possible stopes into possible solutions, of which the possible solution with the highest value is chosen. If the possibilities are enumerated, this is an intractable problem. However, the authors use heuristic short-cuts to converge on a solution quickly. Their paper reports on two case trials that generate better value than the maximum value neighbourhood approach [15].

The aforementioned methods are mostly generic in their application, defining large polyhedrons which are later subdivided into levels depending on the selected mining method. In terms of decomposition, these methods can be applied before very much is settled in terms of method selection. Thomas & Earl [14] and Alford & Hall [16] allow
for other constraints and data to be included such as level separation, and so can be used as a tool to inform the method selection decision. In terms of decomposition it seems the intention is to identify potential stopes, somewhat qualified for stope selection.

A novel approach was developed by Bai, Marcotte & Simon in 2013 [27] using a network flow method. Rather than a regular block model, as used by all other authors, Bai uses a block model with cylindrical coordinate system centred on a vertical raise (the central access point for a variant of stoping called sublevel stoping). A corresponding digraph models mining dependencies with directed edges. The problem can then be treated as a network flow problem. Bai reported superior results compared to the floating stope method in all four of his test cases. In Bai, Marcotte & Simon’s original method, the position of the raise must be given, but in a subsequent paper [28] they added a genetic algorithm to determine the location of the raise, taking into account the cost of a drift (defined on page 5) to access the draw-points and the impact on the stope shape. In 2014 Bai, Marcotte & Simon [29] extended the approach to deal with multiple raises. Bai’s method is specific to a category of stoping called sublevel stoping. This, therefore, is an example of category decomposition, since the method applies to only one of many possible stoping methods.

In 2017 Erdogan et.al. [30] compared some commercially available systems with two research systems. The existing approaches include a Datamine implementation of Alford & Hall stope shape annealing process [16] called “Datamine MSO” and an “MVN Algorithm” attributed to commercial vendor Minesite. The research systems included a heuristic by Sens & Topal [31] and the aforementioned heuristic by Sandanayake, Topal & Asad [26]. In a single case study, Erdogan et.al. reported that the four systems generated stopes with widely varying values, with Datamine MSO delivering the highest value.

In 2018 Nhleko, Tholana & Neingo [32] published a survey, finding none of the ten systems they studied (other than a very old branch and bound method that only operates in 1-dimension) provide optimal solutions.

Subsequent to the aforementioned survey, Nikbin et. al. [33] developed an exact solution to the stope polyhedron optimisation problem in the form of an integer programming model, which was able to solve simple cases. For more complex cases, the authors
presented a new greedy algorithm that they claim produces better results than floating stope [13] and maximum value neighbourhood [15].

The aforementioned methods all use a single deterministic model as input. In 2007 Grieco & Dimitrakopoulos [34] reported their probabilistic mixed integer programming model to optimise stope envelopes for an open stope mine, considering grade uncertainty and predefined degree of acceptable risk. Multiple stope designs are created and then evaluated against a set of stochastic orebody models. The design with the highest expected value (the anticipated value taking into account the probabilities of events and/or uncertainty of information) can then be chosen. Grieco & Dimitrakopoulos reported on the successful application of their approach in [35]. Villalba Matamoros & Kumral [36] also developed a method that incorporated grade uncertainty, in this case using a genetic algorithm. The genetic algorithm is used to improve on a seed solution in order to maximise the expected value for multiple equiprobable stochastically generated realisations of the orebody model.

In summary, there exists a wide range of exact and heuristic approaches to solve the stope envelope optimisation problem, mainly focused on qualified generation of potential stopes, rather than final stope selection. The available exact solutions are restricted to specific mining methods, and/or simple cases. The heuristics can handle larger and more complex models and constraints, but do not guarantee optimality. Of all the deterministic approaches that are commercially available, methods developed by Alford & Hall ([16], [25]) are likely to be the most widely used in the mining industry. A few authors have sought to incorporate grade uncertainty into the optimisation of stope envelopes, but these methods remain in the research domain.

### 2.5 Cave polyhedron optimisation

Some of the methods described in Section 2.4 can also be applied to cave polyhedron optimisation. However other approaches have emerged from a more specific examination of the peculiarities of cave mining, commencing in the 1970s, with Laubscher ([37], [38]) who developed a set of design rules. Key to these design rules is Laubscher’s mining
2.5 Cave polyhedron optimisation

rock mass rating (MRMR, [39], [40]). MRMR is designed for use in mine design and is an adaptation of an earlier rock mass rating (RMR). The rating is calculated on the basis of geotechnical data derived from drill cores, structural interpretation and in an operating mine, observations from previously mined out areas. It is calibrated with empirical data from other existing mines. On the basis of the MRMR for a given orebody, it is possible to estimate the critical hydraulic radius (required to determine the minimum caving footprint) and cave propagation behaviour. It is this cave propagation behaviour that governs the formation of a caved area in response to the establishment of drawpoints and subsequent drawdown. Mine planners seek a drawdown strategy and associated cave polyhedron that will maximise profits.

Laubscher also developed a mixing model, upon which Diering, Richter & Villa [41] developed an algorithm for the purposes of finding an optimal footprint for a block cave. This was implemented in the PCBC software marketed by Dassault Systèmes. This same software is used for estimation of the economic ore that can be extracted from a draw cone, or a cave polyhedron (Diering [42]). The determination of a cave polyhedron, the footprint of the cave and the drawdown strategy are therefore interrelated and typically achieved in an iterative manner.

Trotter & Goddard [4] detailed a decomposition of the sublevel cave layout problem. Their focus was not so much on the definition of the cave polyhedron, but on important parameters associated with the application of sublevel caving to a particular orebody. Implicit in their approach was the a priori determination of the cave polyhedron, by geological control or some cut-off driven approach. Although this section is focused on the cave polyhedron, their method is never the less worth exploring, as it represents a cohesive approach to dealing with a complex design task of the sort that is of interest here. They define mathematical relationships between various design parameters based on knowledge of material flow, and then through a process of enumeration seek to optimise four key design parameters. A key metric they use, attributable to Just, Free &
Bishop [43] is extraction efficiency \((E)\):

\[
E = R \left(1 - \frac{D}{100}\right)
\]  

(2.1)

where \(R\) denotes recovery and \(D\) denotes dilution. They consider development costs to be paramount over operating costs and use cost and \(E\) to find optimal settings for four key geometric parameters:

1. extraction heading height (the height of the extraction cavities, dictated by equipment size)
2. pillar width
3. sublevel interval (the vertical distance between sublevels)
4. ring burden (the thickness of the slice taken with each blast)

Their method relies on having established relationships between extraction efficiency and various geometric parameters, including the aforementioned four. These relationships are based on empirical data from other mines, and knowledge of material flow.

Diering & Breed have contributed to finding solutions to the sublevel cave layout problem [44] applying many of the same concepts as Diering applies to block caving.

From a decomposition perspective, the methods described in this section are obviously separated by category: each method focuses in either caving, or sublevel caving mining methods.

## 2.6 Shaft, decline, ore pass and level optimisation

The optimisation of shaft location is driven by economics (minimising the cost of access) and constrained by numerous engineering considerations. Engineering considerations include the need for standoffs (separation from the shaft and areas that will be mined) and the need to avoid unstable rock. In 1968 Zambo [12] presented methods to optimise the location of a shaft, the capacity of a mine and the separation between levels in a multi-level mine. His tools were typically basic orebody information including layout and densities, empirical data for similar mines and regression analysis. The methods were intended to be applied sequentially. In Zambo’s treatment of level separation, he
observed the requirements of planned production rates, cost minimisation and mining loss due to pillars, but was silent on the effect of level separation on mining dilution.

In 1972 Humphreys & Leonard [45] addressed the shaft location problem. They presented a manual method for optimising the plan coordinates of a shaft, using operations research principles. The method takes into account combined capital and operating costs of transporting ore from various predefined locations.

In 1985 Lizotte & Elbrond [46] presented various techniques to locate facilities (like a shaft) underground and also to design efficient underground networks. Their methods include limited use of computers in procedures:

- **Shaft location** – They applied an iterative procedure: *hyperboloid approximation*.
- **Level layout** – They illustrate the use of minimum Steiner trees using a heuristic approach developed by Liebenthal & Mutmansky [47] to seek a minimum Steiner tree. A minimum Steiner tree solves the level layout problem, if the shaft and stope locations are known and when the requisite haulage capacities of tunnels is ignored (that is, the Steiner tree is un-capacitated).
- **Combined level layout and shaft location** - they develop the *multifacility hyperboloid approximation* procedure, which uses the un-capacitated Steiner tree as a starting point. The multiple facilities in their case are the shaft, and the Steiner points.

In 2010 Gligoric, Beljic & Simeunovic [48] published a *Fuzzy Technique for Order of Preference by Similarity to Ideal Solution* (fuzzy TOPSIS) to locate the shaft. Their case involved a single level which connected multiple sets of drawpoints for stopes. Their solution to connecting the stopes in the level was to use a minimum Steiner tree. In this case, the drawpoint locations are given and the determination of the minimum Steiner tree is straightforward. What remains is to locate the shaft in the horizontal plane to connect to the Steiner tree. TOPSIS is an approach to find a positive ideal solution to a multi-criteria decision and the fuzzy variant of TOPSIS allows it to deal with vagueness of human thought, especially qualitative assessment of value for a given criterion.

Volz, Brazil & Thomas [49] published a new method for shaft location optimisation, with a case study application for the Callie Underground Mine in Northern Territory, Australia. Their problem was more complex than the case addressed by Gligoric. It
involved the determination of the optimal position of an ore pass to which ore from all stopes on several levels will be delivered, the position of the shaft, taking into account the tunnel required to facilitate transportation of ore from the bottom of the ore pass to the shaft, and subject to geotechnical constraints. So for Volz et. al., the focus was the ore pass. To find its ideal coordinates, they reduced it to a Fermat-Weber problem (see figure 2.1), that is, to find a point such that the sum of distances from point \( x \) (the ore pass) to a set of given points \( \{p, p_1, \ldots p_5\} \) is minimised. In this case the shaft location is constrained by a standoff requirement from the ore pass and stopes.
2.6 Shaft, decline, ore pass and level optimisation

In 2009 Brazil et. al. [50] described a software program *Planar underground network optimiser* (PUNO) that solves the level layout problem, treating it as a capacitated Steiner tree. A problem instance includes the 3D coordinates for the various stopes, location of a given cross-cut (defined on page 5), production tonnages and a cost model. It uses the theory of weighted Steiner trees to give a minimum cost network for possible layouts. PUNO is designed to work in conjunction with Brazil et. al.’s underground optimisation software *Decline optimisation tool* (DOT) [50]. A problem instance for DOT includes the 3D coordinates for the surface portal, and groups of nodes, where each node is an optional connection point for the decline to a cross-cut for a level. A problem instance also includes navigation constraints, boundaries of no-go regions and a cost model. DOT and PUNO are used in the commercial environment, being available through mining software vendors Maptek and RPMGlobal. In terms of breadth of commercial implementation, DOT and PUNO are thought to be second only to software provided by Alford Mining Systems with similar functions, which was developed in part under AMIRA project P1037 in [20], [21], [22], [23], [24] and [25]). However, no details on the Alford system are available in the academic literature.

In 2010, Li [51] presented a method to optimize the layout of a level and the selection of stopes simultaneously. The starting point is some network connecting all potential stopes. The solution is a selection of stopes and a subset of the original network, such that the value of the selected stopes, less the cost of their connecting network is maximised. Li modelled the problem as a *prize collecting Steiner tree in graphs* problem. He proposed a solution for a specialized version of this problem, the *k-cardinality prize collecting Steiner tree in graphs* problem. By solving this special version n times, where n is the number of potential stopes, the general problem can be solved.

In summary, shaft, decline and level development are generally treated as discrete problems, each being a minimum cost network in two or three dimensions. Prioritisation is employed to provide an order in which each optimisation task ought to be applied for best effect. Early methods were manual or purely analytical in approach, but with the increased availability of faster computers and highly-developed software, mathematical optimisation, including the use of un-capacitated and capacitated Steiner trees has come
to the fore.

2.7 New decompositions

In Chapter 4 a new decomposition model is given for a case in which main access is achieved with the use of a shaft. The case for this new decomposition is given in Chapter 4 with reference to this chapter. The new decomposition employs a prize collecting Euclidean Steiner tree (PCEST) for the first time. In order to make use of a PCEST, it has been necessary to develop a new algorithmic framework to solve this problem, a challenge that is addressed in Chapters 6, 7 and 8.

A second decomposition model is given in Chapter 5 for the case in which main access is achieved with the use of a decline. This decomposition is formulated as a derivative of the shaft model given in Chapter 4.
Chapter 3
Transition from open pit to underground mining

This chapter describes a new model and method to solve the transition problem. The material in the chapter is extracted from two peer-reviewed papers, the first of which was published in the proceedings of the Seventh International Conference on Mass Mining [52] and the second was published in the European Journal of Operations Research [53].

3.1 Introduction

The application of mathematical optimisation to open pit mine outlines is widespread, beginning in the 1980s with commercial implementations of a graph algorithm developed by Lerchs & Grossmann [54]. This algorithm, commonly referred to as the LG Algorithm, provides an exact method to optimise the outline of an open pit mine, in order to maximise its un-discounted cash value. The LG Algorithm is in wide use in the mining industry, being available from many of the largest mine planning software vendors (Dassault Systèmes; Maptek; Hexagon Mining). The problem can also be framed as a maximum flow problem (Picard [55]). Maximum flow methods in industrial use include the push-relabel algorithm (Goldberg & Tarjan [56], used by Mincom and Minemax); and Hochbaum’s pseudoflow [57] (used by Dassault, Deswik and Muir & Associates Computer Consultants). Some of these maximum flow implementations provide significantly better performance than LG implementations. For example Dray [58] found that Minemax Planner is significantly faster than Dassault Systèmes LG Algorithm implementation, particularly for large block models (e.g. models with more than 5 million blocks). Importantly, Dray also found that the two optimisers yielded the same results.
These pit optimisers decide the outline of the pit in order to maximise un-discounted cash values and they can do so for very large and detailed models (hundreds of thousands or millions of blocks). The computing times vary from seconds to a few hours, depending on the number of blocks and the complexity of the constraints applied to the model. However, miners would rather work with discounted cash-flows and maximise net present value (NPV). Accordingly, pit optimisers are almost always used in a structured process that pursues high NPV solutions for a wide range of planning decisions. The process involves running various optimisers multiple times with different data inputs (For example see Whittle [59] and Hustrulid, Kuchta & Martin [60]). With long planning processes (for example the multi-year Grasberg case), and with efficient pit optimisers available, pit optimisation is generally not on the critical path. However fast optimisation still offers the advantage of allowing a wider range of alternatives to be tested and for larger, more detailed models to be used.

Various authors have tackled the issue of the combined optimisation of open pit and underground mines and some have focused in particular on the transition problem (optimising the economic decision as to where to stop the open pit and where to start the underground mine). Nilsson [61] gave a good account of the factors that need to be considered when planning a transition from open pit to underground mining, including geotechnical interactions, economic considerations and operational challenges. He recognised that the optimal pit design changes if deeper parts of the orebody can be mined by an underground mine. Nilsson’s paper pre-dates the wide availability of pit optimisation software and he did not give a detailed account of the calculations leading to a change in the pit shape.

J Whittle [62] incorporated a method into pit optimisation software that takes into account the value that ore has if mined by an underground method. Consider a case in which some blocks can be mined by either the open pit or underground methods. For any such block, the value used for pit optimisation should be the difference between its open pit value and its underground value. The assumption underpinning this is that for a block that can be mined by either method, if it is not mined by the open pit method, it will be mined by the underground method. Camus [63] independently described an
approach that will generate equivalent results. This will henceforth be referred to as the opportunity cost approach to solving the transition problem (“opportunity cost approach” for short), following the terminology used in the field of economics (For example see McTaggart, Findlay and Parkin [64]).

**Definition 3.1.1 (Opportunity cost).** Let \( v^c \in \mathbb{R} \) and \( v^d \in \mathbb{R}_{\geq 0} \) be the net values for mutually exclusive alternatives \( c \) and \( d \). If no other alternative to \( c \) has a higher value than \( d \), then the opportunity cost for \( c \) is \( v^d \).

In the case at hand, the mutually exclusive alternatives with respect to any given block are to mine it by open pit method (alternative \( c \) in Definition 3.1.1), or by underground method (alternative \( d \)). If a block is mined by open pit method, its open pit value \( (v^c) \) is gained, but the value that would have been gained if it had instead been mined by underground method \( (v^d - \text{the opportunity cost for } d) \) is lost. When the opportunity cost approach is applied the underground value is subtracted from the open pit value for each block, before doing pit optimisation. When applying this approach, as opposed to optimising a pit without regard for the opportunity cost, the optimised open pit is almost always smaller. Also, the value of the open pit mine is lower, but the sum of the values of the open pit and underground mines is maximised. The reason for this is given in Section 3.2.2. There is more than one way to generate a smaller pit using pit optimisation software; for example, it is common to use a technique called pit parameterization to generate a family of pits by flexing the commodity price (for example see Whittle [59]). However, the pit created using the opportunity cost approach may not match the shape and tonnage of any of the pits created using parameterization techniques due to the different ways in which the block values are calculated.

Chen, Gu & Li [65] described a method similar to the opportunity cost approach. They did not use exact optimisation, but they did include consideration of a crown pillar, which the earlier authors had not. A crown pillar is a body of rock left in place above the shallowest part of an underground mine to ensure stability in the surrounding rock. The need for stability is driven by the land use above the underground mine, which in some cases is an open pit mine. The crown pillar also acts to reduce or avoid the ingress of water to the underground mine and ensure the stability of the cavity below. When crown
pillars are used, their design must take into account the geotechnical characteristics of
the native rock and the planned sizes and shapes of the underground stopes (Carter [66]).
Note that the use of crown pillars is not universal, since collapse of the ground above the
underground mine is sometimes a desired or acceptable outcome.

Pit optimisation can be incorporated into a wider workflow with other optimisation
tools to support a wide range of planning decisions as previously mentioned, and this is
the case also when dealing with the transition problem. Finch & Elkington [67] advocate
for automated scenario analysis in which a number of candidate transition depths are
evaluated with schedule optimisation software. Roberts et. al. [68] provide a case study
using a number of different in-house and commercially available optimisation tools. The
case study involves an existing underground mine and contemplates an open pit expansion
on ore that would otherwise be mined in a future underground expansion. These
authors were able to compare a number of different transition scenarios, each with optimi-
sed schedules.

Some authors have applied mixed integer programming methods to the problem,
though they are often faced with tractability issues. Chung, Topal & Ghosh [69] for-
mulated an integer programming model to optimise simple open pit and underground
mines with a simple crown pillar. The crown pillar was modelled as a flat exclusion zone
with a specified thickness across the full width of the block model. This approach is good
for simple cases in which an underground mine begins beneath the deepest part of a pit,
but will not cater to a case in which a pit might extend deeper in an area where under-
ground mining is not viable. Chung, Topal & Ghosh’s model maximises un-discounted
cash-flows. In a case study using a model with 83,000 blocks, the run time was 34.5
hours. In 2012 Bakhtavar, Shahriar & Mirhassani [70] applied integer programming in
the consideration of transition depth, taking into account the crown pillar. They com-
pared a range of transition depths to determine the depth that gave the highest NPV,
although only applied to a two-dimensional model. Newman, Yano & Rubio [71] for-
mulated the transition as a large monolithic longest-path network problem. MacNeil &
Dimitrakopoulos [72] formulated the open pit to underground transition as a stochas-
tic optimisation problem. In both cases, the authors incorporated the consideration of a
crown pillar, scheduling decisions and mining rate decisions and in order to make the problem tractable, they represented the mines in a set of strata: conceptually horizontal slices of the model. The result is an assignment of strata to open pit, crown pillar and underground mines respectively. King, Goycoolea & Newman [73] optimise the open pit and underground mining schedules, the placement of the crown pillar and also the placement of sill pillars (material left in-situ in a particular type of underground mine to allow for a change in mining direction). The open pit model used in this case is not a regular block model, but a set of open pit scheduling polygons produced in part by preprocessing a block model with a pit optimiser. This approach has the advantage of reducing a problem with a large number of blocks, to a problem with just hundreds of open pit scheduling polygons (336 in their case study). This reduction in numbers dramatically improves tractability. However, since the preprocessing steps use regular pit parameterization rather than an opportunity cost approach, the shapes of the open pit scheduling polygons may be inappropriate. Like Chung, Topal & Ghosh [69], King, Goycoolea & Newman [73] model the crown pillar as a single model-wide exclusion zone.

In summary, there are three general approaches:

1. Use an opportunity cost approach with a pit optimiser.
2. Use a purpose-built integer programming optimisation tool.
3. Use workflows that rely on a combination of optimisation tools and judgement to make a wide range of planning decisions.

None of the approaches can handle the full range of complexities confronting mine planners. The opportunity cost approach can use a very capable and mature pit optimiser that can handle large and detailed models, but cannot model the crown pillar or optimise for NPV directly. The purpose-built optimisation tools can optimise to maximise NPV and can model the crown pillar, but the block model and mining models must be simplified in order to make the problem tractable. The workflow-based approaches promise to cover the gaps, but would be better served by more capable mathematical optimisation models.

In Section (3.2), some of these deficiencies are overcome with a new optimisation model and method based on the opportunity cost approach, that provides for flexible
modelling of the physical and economic characteristics of the crown pillar.

3.2 New optimisation method

First, the normal model used for pit optimisation with the LG Algorithm or a maximum flow method is restated, and then the new model is described.

For the convenience of readers of this chapter, a glossary of more frequently used symbols is included in Table A.1 of Appendix A.

3.2.1 Pit optimisation model

Two models are described below: A regular block model (a model in which each block record in a regular three-dimensional grid carries estimates of rock type and mineral grades) is used to represent the material in the ground; an equivalent digraph (a graph with directed edges) is used to frame the optimisation problem.

Given price and cost information, it is possible to assign dollar values to each of the blocks in the regular block model. The open pit value for block \(i\) is \(m_i = r_i - c_i\) where \(r_i\) is the revenue and \(c_i\) is the extraction cost. In keeping with the norms of block value calculations for pit optimisation, the cost \(c_i\) includes only the cost to mine and remove the rock in the block, to dump and rehabilitate the rock that is classified as waste, and to process the rock that is classified as ore (For further details see [60] or [59]). In the problem at hand, assume that the open pit value is not time dependent. A block’s open pit value will be achieved if and only if it is included in the open pit outline. In an open pit mine, in order to mine any block, the blocks above it must also be mined. Moreover, if the walls of the excavation are too steep, they will be unstable and collapse, so a safe maximum pit slope must be maintained. Safe maximum pit slopes typically range from 40 to 55 degrees (measured from the horizontal), depending on the strength of the rock. The open pit optimisation problem involves finding the pit outline that encloses a subset of blocks such that the sum of the values of the blocks in the subset is maximised, whilst obeying pit slope constraints.

In tackling this optimisation problem, it is convenient to use a digraph. A formal
3.2 New optimisation method

Definition of a digraph is given in Definition 3.2.1, but for readers unfamiliar with the domain, it may suffice to think of a digraph as being comprised of a set of points, and a set of arrows (each with a tail and a head) connecting pairs of these points. In the sequel we will refer to these points and arrows formally as vertices and arcs respectively. These two terms are also defined in Definition 3.2.1. Blocks in the block model are represented by vertices, and the maximum safe pit slope constraints are modelled with arcs. Let the set $X$ of vertices $x_i$, the set $A$ of arcs $a_k = (x_i, x_j)$ and the set $M_X$ of vertex weights $m_i$ define a digraph $G = (X, A, M_X)$. In this digraph:

- Each vertex $x_i \in X$ corresponds to block $i$ in the block model, and has a weight $m_i \in M_X, m_i \in \mathbb{R}$. The weight represents the open pit value of the corresponding block $i$ in the block model.
- Each arc $a_k = (x_i, x_j), a_k \in A$ represents a mining dependency, specifically relating to the need to uncover a block and to maintain maximum safe pit slopes. If mining the block represented by the vertex $x_i$, is dependent on the block represented by $x_j$ being mined, then the arc $a_k = (x_i, x_j)$ will be included in $A$.

Definition 3.2.1 (Graph, digraph, vertex, edge, arc (directed edge), head, tail, vertex weight). A graph $G_X$ is an object consisting of two sets, a vertex set $X$ and an edge set $E_X$. Each element $x$ of $X$ is called a vertex. Each element $e$ of $E_X$ is called an edge, and is a two-element subset of $X$. A digraph is a graph consisting of a vertex set and an arc set $A_X$. Each element $a$ of $A_X$ is called an arc (also known as a directed edge), and is a two-element permutation of $X$. One element of such a permutation is called the tail of the arc and the other element is called the head of the arc. A vertex weight $m$ is a real number corresponding to a vertex $x$. The set of vertex weights is denoted $M_X$.

Definition 3.2.2 (subgraph). A graph $G_Y$ is a subgraph of graph $G_X$ if $Y \subseteq X$ and $E_Y \subseteq E_X$. The vertex weights $m_i \in M_Y, i : x_i \in Y$, are induced by the weights in $G$.

Definition 3.2.3 (Closure). A closure $G_Y$ of $G_X$ is a digraph $(Y, A_Y, M_Y)$, defined by a set of vertices $Y \subseteq X$ such that $A_Y$ includes all the arcs in $A$ that have tails in vertices of $Y$. Vertices at the heads of all arcs in $A_Y$ must also be included in $Y$. The vertex weights $m_i \in M_Y, i : x_i \in Y$, are induced by the weights in $G$. 
Note that the possibility that a closure $G_Y = \emptyset$ is not excluded. Figure 3.1 shows an example of a two-dimensional model with arcs (left) and an example of a closure (right). This is a simplistic two-dimensional model for an open pit. With square blocks, only three arcs per block are required to model 45-degree slopes. To model slopes accurately in a typical three-dimensional model, thirty or more arcs per block may be needed, including arcs that extend up more than one level.

The aforementioned pit optimisation problem can now be restated in Graph Theory terms as a *maximum graph closure problem*.

**Definition 3.2.4 (Maximum graph closure (MGC)).** A *maximum graph closure* is a closure $G_Y$ of a digraph $G$ that maximises $M_Y$:

$$M_Y = \sum_{i : x_i \in Y} m_i$$  \hspace{1cm} (3.1)

and such that for each closure $G_X \subset G_Y$, $M_X < M_Y$.

Note that this definition for MGC differs from some earlier papers (for example see Picard [55]) by the inclusion of the condition for each closure $G_X \subset G_Y$, $M_X < M_Y$. This condition is consistent with the operation of the LG Algorithm, which guarantees to find a maximum graph closure with the smallest number of vertices and this is a desirable feature in pit design. Suppose there were two solutions to the problem, both with the same value but one with more blocks than the other. A miner would much rather see the solution with fewer blocks, implying less work, less time, and less environmental disturbance.

Theorem 3.2.1 formalises a proof given by J Whittle in the early 1990s and included in unpublished training material for pit optimisation software.

**Theorem 3.2.1.** The maximum graph closure (MGC) is unique.

**Proof.** Suppose a graph $G$ has two distinct non-empty MGCs $G_Y$ and $G_Z$ implying $M_Y = M_Z > 0$, and consider these cases:
3.2 New optimisation method

1. One MGC is a proper subset of the other: It follows from Definition 3.2.4 that an MGC cannot be a proper subset of another MGC leading to a contradiction.

2. The two MGCs do not share any vertices: If \( G_Y \cap G_Z = \emptyset \), then \( G_Y \cup G_Z \) is a closure that has a value \( M_Y + M_Z \). Since \( M_Y = M_Z > 0 \), \( G_Y \cup G_Z \) is a closure with a higher value than either \( G_Y \) or \( G_Z \) leading to a contradiction (that is, neither \( G_Y \) nor \( G_Z \) are MGCs).

3. The two MGCs share one or more vertices and neither is a proper subset of the other: Let \( M_{Y \setminus Z}, M_{Z \setminus Y} \) and \( M_{Y \cap Z} \) be the sums of the weights of vertices in \( G_Y \setminus G_Z, G_Z \setminus G_Y \) and \( G_Y \cap G_Z \) respectively. \( M_Y \) and \( M_Z \) are given by Equations 3.2 and 3.3

\[
M_Y = M_{Y \setminus Z} + M_{Y \cap Z}
\]  
(3.2)

\[
M_Z = M_{Z \setminus Y} + M_{Y \cap Z}
\]  
(3.3)

If \( G_Y \) and \( G_Z \) are closures, then \( G_Y \cup G_Z \) and \( G_Y \cap G_Z \) are closures and the value of \( G_Y \cup G_Z \) is given by Equation 3.4

\[
M_{Y \cup Z} = M_{Y \setminus Z} + M_{Y \cap Z} + M_{Z \setminus Y}
\]  
(3.4)

Since \( G_Y \) and \( G_Z \) are both MGCs, \( M_Y = M_Z \) and given Equations 3.2 and 3.3, \( M_{Y/Z} = M_{Z \setminus Y} \). Hence \( M_{Y \cup Z} = 2M_{Y \setminus Z} + M_{Y \cap Z} \).

Since \( G_Y \cap Z \) is a proper subset of \( G_Y \), \( M_{Y/Z} \neq 0 \) (by Case 1). Consider the remaining alternatives:

- If \( M_{Y \setminus Z} > 0 \) then \( G_{Y \cup Z} \) is a closure with a higher value (a contradiction).
- If \( M_{Y \setminus Z} < 0 \) then \( G_{Y \cap Z} \) is a closure with a higher value (a contradiction).

\[ \square \]

Definition 3.2.5 (Maximum graph closure problem). This is the problem of finding an MGC for a given digraph.

Figure 3.2 illustrates an optimal pit design. In this simple two-dimensional example, pit slopes are 45 degrees and there are just a few blocks. In practice, models are
28 Transition from open pit to underground mining

Figure 3.1: A two-dimensional model with arcs representing block dependencies (left) and an example closure (right).

Figure 3.2: A simple model for open pit optimisation with open pit values, $m_i$. The blocks in the optimal pit are shown in green.

three-dimensional; have tens of thousands to millions of blocks; with thirty or more arcs per block to represent complex pit slope requirements. Commercial pit optimisers are efficient enough to solve these large problems in practical time-frames.

3.2.2 New model

A new model is defined as follows. $G = (X, A, M_X)$ is a digraph representing the open pit optimisation problem with underground option and allowance for a well-formed crown pillar. Note that the concepts that follow are illustrated in figures showing trivial numbers of blocks. This does not imply that the methods only work for small models and an industrial-size application is demonstrated in Section 3.5.
3.2 New optimisation method

Vertices and weights

Vertices are divided into two sets corresponding to two copies of the set of blocks:

- \( x_i^p \in X^p \) are vertices corresponding to blocks \( i \) in the block model with weights set to the open pit value of the block: \( m_i^p = r_i^p - c_i^p \) (as in regular pit optimisation model).
- \( x_j^u \in X^u \) are vertices corresponding to blocks \( j \) in the block model with weights set to the negative underground mining value of the block: \( m_j^u = -(r_j^u - c_j^u) \). This is a negative value since the effect of including this vertex in a closure \( G_Y \) is that it is no longer available for underground mining, and so its underground mining value is lost. The important assumption here is that if this vertex is not included in a closure \( G_Y \), then the block it represents will be included in an underground mine and its underground mining value will be realised. In micro-economic terms, the vertex weight is an opportunity cost (See Definition 3.1.1) with respect to the opportunity to mine a corresponding block in \( X^p \) represented by a vertex \( x_i^p \) and connected to this vertex via an arc \( a_k^\gamma = (x_i^p, x_j^u) \) (see below).

A trivial example is shown in Figure 3.3. The underground mining values for blocks are always positive. They represent the value of including a block in the underground mine. Consequently the corresponding vertex weights are always negative. Note also that only values for blocks representing material in stopes and caving polygons are needed, so there may be fewer vertices defined for underground than for open pit. In the trivial example shown in Figure 3.3 all vertices in \( X^u \) have been assigned the same negative underground mining value of \(-2\), but they could have been individually assigned any negative real values.

The vertices in \( X^p \) and \( X^u \) correspond to the same blocks in the block model’s three dimensional framework, but are distinct and separate vertices in the digraph with \( X = X^p \cup X^u \).

Arcs

The arcs \( A \) in \( G \) are of three different types, \( \beta, \gamma \) and \( \delta \), where \( A = \beta \cup \gamma \cup \delta \).
30 Transition from open pit to underground mining

![Diagram of Xp and Xu sets of vertices and γ arcs.](image)

Figure 3.3: Example of $X^p$ and $X^u$ sets of vertices (See Section 3.2.2) and $\gamma$ arcs (See Section 3.2.2). Only a selection of the arcs are shown.

\[ \beta \] is the set of arcs $a_{k}^\beta = (x^p_i, x^p_j)$, $a_{k}^\beta \in \beta$ with tails and heads both in $X^p$. Each arc represents a mining dependency to maintain maximum safe pit slopes (as in the normal pit optimisation model).

\[ \gamma \] is the set of arcs $a_{k}^\gamma = (x^p_i, x^u_j)$, $a_{k}^\gamma \in \gamma$ with tails in $X^p$ and heads in $X^u$. These arcs bring the opportunity cost into the closure. Each arc connects vertex $x^u_j$ to vertex $x^p_i$, for every $x^u_j$ that has a weight defined (recall only underground values for blocks in stopes or caving polygons are needed, and only the vertices corresponding to these will have weights defined). Observe that block $x^u_j$ has the same physical location as block $x^p_i$. The block $x^u_j$ is under the block $x^p_i$, with the vertical separation being equal to the minimum required thickness of the crown pillar as illustrated in Figure 3.3. The purpose of these arcs is as follows: if a block in $X^p$ is included in a closure, then the opportunity to mine another block (some levels down - with the number of levels being equivalent to the minimum required thickness of the crown pillar) by underground method will be lost.

Before describing the $\delta$ arcs, some definitions are presented.

**Definition 3.2.6 (Strongly connected subgraph (SCS)).** A strongly connected subgraph (SCS)
3.2 New optimisation method

is a subgraph of a digraph in which, for every ordered pair of vertices \((x_i, x_j)\), there is a directed path from \(x_i\) to \(x_j\) and from \(x_j\) to \(x_i\).

Note that the path in Definition 3.2.6 need not be between distinct pairs and that a vertex is considered to have a path to itself.

Definition 3.2.7 (Trivial SCS). A trivial SCS is an SCS containing one vertex.

Definition 3.2.8 (Non-trivial SCS (NSCS)). A non-trivial SCS is an SCS containing more than one vertex.

\(\delta\) is the set of arcs \(a_{\delta}^k = (x_u^i, x_u^j), a_{\delta}^k \in \delta, x_u^i \neq x_u^j\), with tails and heads both in \(X^u\) forming NSCSs (and observe that neither \(\beta\) nor \(\gamma\) arcs give rise to NSCS). Each NSCS represents the required shape for the ceiling of the underground mine (the base of the crown pillar) at a given depth as illustrated in Figure 3.4. In the example on the left, eight NSCSs prescribe a flat ceiling, with each NSCS corresponding to a different level. In the example on the right, the NSCSs corresponding to the three highest levels prescribe a dome shape (shown here in 2D) and at lower levels, a flat ceiling is prescribed. Each NSCS can be constructed by using arcs to form one or more overlapping directed cycles. Note that the \(\delta\) arcs combined with the \(\gamma\) arcs also influence the shape of the bottom of the open pit mine, but do not fix the shape. There is often an operational requirement to have a minimum flat area at the bottom of a pit, and these NSCSs might contribute to the achievement of this requirement in some cases. However, full consideration of this requirement is beyond the scope of this model.

Problem definition

The problem is to find a closure \(G_Y\) of \(G\) that maximises \(M_Y\) (i.e. the maximum closure of this digraph):

\[
M_Y = \sum_{i: x_i^p \in Y} m_i^p + \sum_{j: x_j^u \in Y} m_j^u
\]  

(3.5)

Derivation is now used to show how maximisation of \(M_Y\) is equivalent to maximisation of the total value of the open pit and underground mines. The value of the open pit
Let \( M_T \) denote the sum of these values.

\[
M_T = \sum_{i: x^p_i \in Y} m^p_i - \sum_{k: x^u_k \notin Y} m^u_k \tag{3.6}
\]

The total value of material potentially available for underground mining is a constant \( -\sum_{i: x^u_i \in X^u} m^u_i \). Let \( \kappa = -\sum_{i: x^u_i \in X^u} m^u_i \). Some of the vertices are in the MGC \( (x^u_j \in Y) \) and the rest are not \( (x^u_j \notin Y) \) and Equations 3.7 and 3.8 follow from this observation.

\[
\kappa = -\sum_{j: x^u_j \in Y} m^u_j - \sum_{k: x^u_k \notin Y} m^u_k \tag{3.7}
\]

\[
\Rightarrow \sum_{k: x^u_k \notin Y} m^u_k = -\kappa - \sum_{j: x^u_j \in Y} m^u_j \tag{3.8}
\]

Substitute from (3.8) into (3.6).

\[
M_T = \sum_{i: x^p_i \in Y} m^p_i - (-\kappa - \sum_{j: x^u_j \in Y} m^u_j) = \sum_{i: x^p_i \in Y} m^p_i + \sum_{j: x^u_j \in Y} m^u_j + \kappa \tag{3.9}
\]

Substitute from (3.2) into (3.9).

Figure 3.4: The figures depict various NSCSs to model the ceiling of the underground mine (the base of the crown pillar).
### 3.2 New optimisation method

#### Figure 3.5: Blocks in the optimal pit

- \( x^p_i \in Y \) represent blocks in the optimal open pit.
- \( x^u_i \in Y \) represent blocks not available for underground mining (including blocks in the crown pillar).
- \( x^u_i \in X^u \cap Y : x^p_i \notin Y \) represent blocks in the crown pillar.
- \( x^u_i \notin Y \) represent blocks available for underground mining.

\[
M_T = M_Y + \kappa \quad (3.10)
\]

Since \( \kappa \) is a constant we conclude that maximising \( M_Y \) also maximises \( M_T \).

Figure 3.5 provides an illustration of the interpretation once the MGC is found. The details are as follows:

- \( x^p_i \in Y \) are shown on the left in green.
- \( x^u_i \in Y \) are shown in red on the right.
- \( x^u_i \notin Y \) are shown in blue on the right.
- The green and red blocks on the right are not available for underground mining (\( x^u_i \in Y \)).
3.3 Reduction of an opportunities with dependencies and opportunity costs (ODOC) Problem to an MGC Problem

An optimisation problem involving multiple opportunities with dependencies is constructed, and with opportunity costs (ODOC) (Definition 3.3.1), which is a generalised form of the New Model (Section 3.2.2) not including the \( \delta \) arcs, which are discussed later in Section 3.4. It is then shown that this problem can be reduced to an MGC problem (Definition 3.2.5).

Recall Definition 3.1.1 for Opportunity Cost and recall that in the case at hand, the mutually exclusive alternatives with respect to any given block are to mine it by open pit method, or by underground method. If it is mined by open pit method, its open pit value is gained, but the value that would have been gained if it had been mined by underground method is lost.

Let \( C \) be a set of opportunities. For each opportunity \( c_i \in C \), let \( v^{c_i} \in \mathbb{R} \) be the value and \( v^{d_i} \in \mathbb{R}_{\geq 0} \) be the opportunity cost. If opportunity \( c_i \) is selected, then the value \( v^{c_i} \) will be gained but the value \( v^{d_i} \) will be lost. Let \( E \) be a set of dependencies and let \( e_j = (c_i, c_k) \in E \) represent a dependency between \( c_i \) and \( c_k \) such that if \( c_i \) is selected, then \( c_k \) must be selected.

In the mining problem at hand, an opportunity corresponds to a decision to include a given block in the open pit, and the opportunity cost corresponds to the resultant inability to include some block or blocks in the underground mine.

**Definition 3.3.1 (ODOC problem).** Find the set \( C' \subseteq C \) such that \( \sum_{i \in C'} (v^{c_i} - v^{d_i}) \) is maximised, subject to: if \( e_j = (c_i, c_k) \in E \) and if \( c_i \in C' \) then \( c_k \in C' \).

An example digraph model for an ODOC problem is shown in Figure 3.6. In this example, opportunities \( c_1, c_2, c_3 \) and their values \( v^{c_1}, v^{c_2}, v^{c_3} \) are represented by vertices \( x^{c_1}, x^{c_2}, x^{c_3} \) and weights \( m^{c_1}, m^{c_2}, m^{c_3} \) respectively. Dependencies \((c_1, c_2), (c_1, c_3)\) and \((c_2, c_3)\) are represented by arcs \((x^{c_1}, x^{c_2}), (x^{c_1}, x^{c_3})\) and \((x^{c_2}, x^{c_3})\). The opportunity costs
are represented by combinations of arcs and weighted vertices. For example arc \((x^{c_1}, x^{d_1})\) and vertex \(x^{d_1}\) with weight \(m^{d_1} = -v^{d_1}\) represent the opportunity cost \(v^{d_1}\) for \(c_1\).

**Theorem 3.3.1.** For any set of opportunities with values in \(\mathbb{R}\), the ODOC Problem reduces to an MGC problem.

**Proof.** First a transformation of the objects and values in the ODOC problem to vertices, vertex weights and arcs in a digraph is described. Then it is shown that with this transformation, values accrue to the objective functions for ODOC and MGC in exactly the same way. Since both ODOC and MGC aim to maximise an objective function, we conclude that for any given problem instance, a solution to the MGC problem will be identical to a solution to the ODOC problem.

Let \(G\) be a digraph \(G = (X, A, M_X)\) with weights for each vertex \(x_i \in X\) given by \(m_i \in M_X\). Let \(X = X^c \cup X^d\), \(M_X = M^c \cup M^d\) and let the ODOC problem be represented in the digraph as follows:

- For each \(c_i \in C\) in the ODOC problem, there exists a vertex \(x^{c_i} \in X^c\) with weight \(m^{c_i} = v^{c_i}\), and a vertex \(x^{d_i} \in X^d\) with weight \(m^{d_i} = -v^{d_i}\). Note that since \(v^{d_i} \geq 0\), it follows that \(m^{d_i} \leq 0\).
- For each \(i\) there exists an arc \((x^{c_i}, x^{d_i})\).
- For each \(e_j = (c_i, c_k) \in E\) in the ODOC problem there exists an arc \((x^{c_i}, x^{c_k})\).

Given the above representation, in order to show that the ODOC problem reduces to the MGC problem, it suffices to show that for a set \(C' \subseteq C\) solving the ODOC problem, there exists a set \(G_Y \subseteq G\) solving the MGC problem such that \(\sum_{i : c_i \in C'} (v^{c_i} - v^{d_i}) = \sum_{i : x_i \in Y} m_i\).

Recall that \(X = X^c \cup X^d\) and \(M = M^c \cup M^d\). It follows that \(\sum_{i : c_i \in C'} m_i = \sum_{i : x^{c_i} \in Y} m^{c_i} + \sum_{i : x^{d_i} \in Y} m^{d_i}\). It is clear that the arcs \((x^{c_i}, x^{c_k})\) exactly model the dependencies \((c_i, c_k) \in E\) in the ODOC problem and so the value \(v^{c_i}\) accruing to \(\sum_{i : c_i \in C'} (v^{c_i} - v^{d_i})\) is equivalent to the value \(m^{c_i}\) accruing to \(\sum_{i : x^{c_i} \in Y} m^{c_i}\). It will be shown that vertices \(x^{c_i}\) and \(x^{d_i}\) must either be both in the MGC or both not in the MGC. It then follows that the value \(-v^{d_i}\) accruing to \(\sum_{i : c_i \in C'} (v^{c_i} - v^{d_i})\) is equivalent to \(m^{d_i}\) accruing to \(\sum_{i : x^{d_i} \in Y} m^{d_i}\).

Any arc with its tail in a closure must also have its head in the closure. Hence, since \((x^{c_i}, x^{d_i}) \in A\), it follows that \(x^{c_i}\) can only be in the MGC if \(x^{d_i}\) is in the MGC. Vertex
$x^d_i$ could be in a closure without $x^e_i$ but since $m^d_i \leq 0$, it cannot be in the MGC unless $m^e_i > -m^d_i$ in which case $x^e_i$ is also in the MGC.

\square

3.4 Inclusion of non-trivial strongly connected subgraphs

3.4.1 Introduction

Recall Definition 3.2.8 for non-trivial SCS (NSCS). In section 3.5 it is demonstrated with a realistic data set that a specific application of the LG Algorithm behaves well with respect to NSCSs, created by the $\delta$ arcs which are used to model well-formed crown pillars. Although this one demonstration is encouraging, it is not sufficient to give comfort that all solutions to the MGC problem will operate properly in all cases involving NSCSs.

Lerchs & Grossmann [54] and other authors applying maximum graph closure models in mine optimisation only contemplated the pit optimisation problem, not the transition problem or crown pillars and only described the use of arcs to model pit slope limits. Such arcs do not form NSCSs and the authors did not consider whether their method would be affected by NSCSs. Caccetta & Giannini [74], who restated the LG Algorithm in
more formal terms, made no mention of NSCSs either. Furthermore, as discussed in the Introduction, other methods based on maximum flow algorithms have been developed and there is no way to guarantee that all implementations of LG and maximum flow will deal with NSCSs correctly. To avoid this risk, a procedure is developed that collapses all NSCSs to representative vertices prior to solving the MGC problem, then expands representative vertices in the MGC to their original vertices. It will be shown that a digraph with NSCSs is equivalent to a digraph containing the aforementioned representative vertices. Furthermore, supposing some method for solving the MGC problem correctly deals with NSCSs, it will be shown that the procedure will generate identical results, with the added advantage of reducing the size of the MGC problem instance, and so promising to reduce computing time.

Consider the digraph $G = (X, A, M_X)$ and a closure $G_Y = (Y, A_Y, M_Y)$ such that the sum of the weights of the vertices in the closure $M_Y$ is maximised and such that for each closure $G_X$, where $G_X \subset G_Y, M_X < M_Y$ (i.e. the MGC as per Definition 3.2.5).

### 3.4.2 SCSs and their properties in closures

**Lemma 3.4.1.** Members of an SCS are either all in or all out of a closure.

*Proof.* Consider some vertex that is a member of an SCS. As a property of an SCS, the vertex must have a path to every other vertex in the SCS and every other vertex in the SCS must have a path to the vertex. It follows from the basic properties of a closure (Definition 3.2.3) that if one member of an SCS is in a closure then all members of the SCS must be in the same closure.

### 3.4.3 Strongly connected components

**Definition 3.4.1** (Strongly connected component (SCC)). A strongly connected component (SCC) is an SCS that is not contained in any other SCS.

The following observations are made:
A trivial SCS (Definition 3.2.7) is an SCC provided it is not contained in any other SCS.

Every vertex in a digraph must be in exactly one SCC.

Let $S_X$ be the set of SCCs in a digraph $G = (X, A, M_X)$ such that $S_X$ contains all the elements of $X$. Let $S_i = (X^i, A^i, A^i_R, A^i_T, M^i_X) \in S_X$ denote the $i^{th}$ of $r$ SCCs in $G$ where:

- $x_j \in X^i$ is a vertex member of the SCC.
- $A^i$ is the set of arcs connecting vertices in $X^i$ to each other.
- $A^i_R$ is the set of arcs with tails outside $X^i$ and heads in $X^i$.
- $A^i_T$ is the set of arcs with heads not in $X^i$ and tails in $X^i$.
- $m_j \in M^i$ is the weight of vertex $x_j \in X^i$.

It follows from the properties of an SCS that each vertex at the tail of an arc in $A^i_R$ can reach all vertices in $X^i$ and that all vertices at the heads of arcs in $A^i_T$ can be reached from all vertices in $X^i$. From Definition 3.2.6 (SCS) and Definition 3.4.1 (SCC), it is immediately apparent that there is an equivalence relation between any two vertices $x_j \in X^i$ and $x_k \in X^i$ (i.e. the relation is reflexive, symmetric and transitive) and that $S_X$ forms a partition of $X$ (i.e. $X = \bigcup_{i \in \{1,2,\ldots,r\}} X^i$, where $\bigcup$ signifies a disjoint union).

### 3.4.4 Functions

A function to collapse all SCCs in a digraph and the inverse of this function, to expand all SCCs in a digraph are described. Observe that in collapsing and expanding all SCCs, these function collapse and expand all NSCSs. MGC is also re-framed as a function using the same notation.

Define $\Gamma(X, A, M_X) = (X^*, A^*, M_X^*, S_X)$ [Collapse SCCs] as follows:

1. Identify all $r$ SCCs $S_i = (X^i, A^i, A^i_R, A^i_T, M^i_X) \in S_X$ and denote the set of sets $S_X$.
2. Let $X^* = X$, $A^* = A$ and $M_X^* = \emptyset$.
3. For $i \in \{1, 2, \ldots, r\}$:
   
   (a) In $X^*$: replace all of the vertices in $X^* \cap X^i$ with a single vertex $x^i_0$.
   (b) In $M_X^*$: add $m^i_0 = \sum_{m_j \in M^i} m_j$. 

(c) In $A^*$:
   i. Remove arcs $A^* \cap A^i$.
   ii. Modify arcs in $A^i_R$ such that vertex $x_{i0}$ is at the head of each arc.
   iii. Modify arcs in $A^i_T$ such that vertex $x_{i0}$ is at the tail of each arc.
   iv. Replace each set of parallel arcs with a single arc.

Define $\Gamma^{-1}(X^*, A^*, M^*, S_X) = (X, A, M_X)$ [Expand SCCs] as follows:
1. Let $X = \emptyset$, $A = \emptyset$ and $M_X = \emptyset$.
2. For $i \in \{1, 2, ..., r\}, i : S_i \in S_X$:
   (a) In $X$: If $x_{i0} \in X^*$ add vertices $X^i$.
   (b) In $M_X$: If $x_{i0} \in X^*$ add weights $M^i_{X^i}$.
   (c) In $A$: Add any arcs in $A^i \cup A^i_R \cup A^i_S$ incident to any vertex in $X^i$.

Define $\Pi(X, A, M_X) = (Y, A_Y, M_Y)$ as finding the MGC $G_Y = (Y, A_Y, M_Y)$ of $G = (X, A, M_X)$.

Define $\Pi^*(X, A, M_X, S_X) = (Y, A_Y, M_Y, S_Y)$, differing from $\Pi$ only in the addition of $S_X$ and $S_Y$ to the domain and co-domain respectively, where $S_i \in S_Y, i : X_i \subseteq Y$.

Observations:
- $\Gamma^{-1}(\Gamma(X, A, M_X)) = (X, A, M_X)$.
- Functions $\Gamma$ and $\Gamma^{-1}$ map closures to closures (by the basic properties of a closure (Definition 3.2.3) and Lemma 3.4.1) and preserve the value of any closure.
- Since weights $m_j$ for vertices in $X$ each accrue to exactly one $m_{i0}$ in $\Gamma$ it follows that $\sum_{j : x_j \in X} m_j = \sum_{i : x_{i0} \in X^*} m_{i0}$.

3.4.5 Procedures

Define Procedure $A$ as $\Pi$ with notation as shown in Section 3.4.4.

Define Procedure $B$ as a composition of three functions $\Gamma^{-1}(\Pi^*(\Gamma(X, A, M_X))) = (X', A', M_{X'})$, and with notation as follows:
The equivalence of Procedures A and B will be shown, as illustrated in Figure 3.7.

Observe that $Y^*$ and $X'$ are the vertex outputs of $\Pi^*$ and $\Gamma^{-1}$ respectively in Procedure B.

**Lemma 3.4.2.** The vertex output of $\Pi^*$ will contain the representative vertex of an SCC, if and only if the vertex output of $\Gamma^{-1}$ contains the vertices of the SCC. That is: $x_i^0 \in Y^* \iff X^i \subseteq X'$.

**Proof.** The proof follows from the function $\Gamma^{-1}$, in which $x_i^0$ is replaced with the vertices $X^i$.

Denote the elements of the domains for $\Pi$ in Procedure A and $\Pi^*$ in Procedure B as $G$ and $G^*$ respectively. That is, $G = (X, A, M_X)$ and $G^* = (X^*, A^*, M_X^*, S_X^*)$.

**Lemma 3.4.3.** The MGC from $\Pi$ in Procedure A will contain the vertices of an SCC if and only if the MGC from $\Pi^*$ in Procedure B contains the representative vertex of the SCC. That is, $X^i \subseteq Y \iff x_i^0 \in Y^*$.

**Proof.** Through the operation of $\Gamma$, collectively the vertices in $X^i \in X$ have the same weight and the same dependencies as $x_i^0 \in X^*$. The effect of including $X^i$ in a closure
of \(G\) is identical to the effect of including \(x^i_0\) in a closure of \(G^*\). It follows that for any closure of \(G\) containing some selection of \(X^i\)'s there is a closure of \(G^*\) containing the representative vertices \(x^i_0\) for the same selection, such that the sum of the weights of the vertices in the first closure is equal to the sum of the weights of the vertices in the second closure.

Observe that \(Y\) and \(Y^*\) are the vertices of the MGCs of \(G\) and \(G^*\) respectively and by Theorem 3.2.1 each is unique.

Consider these two cases, one of which must be true if Lemma 3.4.3 is false.

Case (1). Suppose \(X^i \subseteq Y\) and \(x^i_0 \notin Y^*\): Then there is an MGC on \(G^*\) that does not contain \(x^i_0\) and a closure on \(G^*\) that does contain \(x^i_0\), but which must be an MGC because it is induced by the MGC on \(G\) that contains \(X^i\).

Case (2). Suppose \(X^i \nsubseteq Y\) and \(x^i_0 \in Y^*\): Then there is an MGC on \(G^*\) that contains \(x^i_0\) and a closure on \(G^*\) that does not contain \(x^i_0\), but which must be an MGC because it is induced by the MGC on \(G\) that does not contain \(X^i\).

Both of the above cases lead to contradictions of Theorem 3.2.1.

**Theorem 3.4.4.** A maximum graph closure (MGC) induces an MGC in which SCCs have been collapsed to representative vertices. Furthermore, when the representative vertices in the MGC are expanded to their original vertices, this set of vertices equals the set of vertices in the first-mentioned MGC. That is, \(\Gamma^{-1}(\Pi^*(\Gamma(X,A,M_X))) = \Pi(X,A,M_X)\).

**Proof.** It is sufficient to show that the vertices in \(Y\) (the vertex output of Procedure A) are the same as the vertices in \(X'\) (the vertex output of \(\Gamma^{-1}\) in Procedure B). In other words, \(x^i_j \in Y \iff x^i_j \in X'\), which follows immediately as a consequence of Lemmas 3.4.2 and 3.4.3.
The implication of Theorem 3.4.4 is that Procedure B can be used to avoid the risk of NSCSs being dealt with incorrectly when solving the MGC problem.

### 3.4.6 Computational complexity

Procedure A (Π) solves the MGC problem. Caccetta & Giannini [74] state that MGC can be solved in $O(n^2 \log n)$ time, where $n$ is the number of vertices. For Procedure B:

1. **Γ**: Recall that NSCSs are created with the δ arcs in the new optimisation model (Section 3.2.2), so it is known in advance which NSCS each vertex belongs to. Accordingly, Γ can be completed in $O(n)$ time.

2. **Π**: This can be solved in the same time as Π in Procedure A, however the input size $m$ may be smaller. That is, it can be completed in $O(m^2 \log m)$, where $m \leq n$ and $n - m$ is the reduction in the number of vertices as a result of Γ.

3. **Γ⁻¹**: Following similar reasoning to that given above, Γ⁻¹ can be completed in $O(n)$ time.

Overall the computational complexity of Procedure B is $O(n + m^2 \log m)$, which compares favourably with Procedure A:

- If $n < m^2 \log m$, then Procedure B has the same or less complexity than Procedure A. That is, $O(n + m^2 \log m) = O(m^2 \log m) \leq O(n^2 \log n)$.
- If $n \geq m^2 \log m$ then Procedure B is less complex than Procedure A. That is, $O(n + m^2 \log m) = O(n) < O(n^2 \log n)$.

Supposing that there is a method to solve MGC that deals properly with NSCSs and it is used in both procedures:

- If there are no NSCSs in the digraph $G$, Procedure B will take more processing time than Procedure A. This is because Π in Procedure B will take the same time as Procedure A, and Γ and Γ⁻¹ in Procedure B will take additional time.
As the number of NSCSs increases, the time taken by Procedure B will decrease relative to the time taken by Procedure A. For a large number of NSCSs, it is expected that Procedure B will take considerably less time than Procedure A.

3.5 Experimental results

In Section 3.2.2 the new method is illustrated with small, simple two-dimensional examples. In this section, the new method is demonstrated to work with a large, complex, three-dimensional model.

The starting point to obtain a suitable model set, is a well-known block model Marvin, constructed in the 1990s by Australian geologist Norm Hanson. It represents a 20 mT copper and gold deposit, with sulphide and oxide ores. The model was initially designed to test open pit optimisation software and is still used for that purpose (Espinoza et. al [75]). A much deeper model than the original Marvin was needed, so the lowest bench was duplicated to make 17 lower benches. An underground version of the model was then developed, as well as various sets of arcs. The full set of models and arcs:

- An open pit model (Xp) with 124,440 blocks. This is the vertically extended Marvin model.
- An underground model (Xu) with 124,440 blocks. As above, but each block is assigned a negative underground value representing the opportunity cost.
- A set of 6,733,168 β arcs to model pit slopes in Xp.
- For a zero thickness crown pillar: A set of 13,534 γ arcs to model the dependencies between blocks in Xp and Xu.
- For a 120 metre thick crown pillar: A second set of 13,534 γ arcs to model the dependencies between blocks in Xp and Xu.
- For a well-formed crown pillar: A set of 12,075 δ arcs to form NSCSs in Xu. Note that the pit optimiser used (see below) behaved well with respect to NSCSs so it was not necessary to run a function to collapse them (see Section 3.4.4).

Five cases were developed and optimised using a widely used commercial pit optimisation system. The system was an implementation of the LG Algorithm and an example
of the several highly efficient solutions to the MGC problem that are available (refer to section 3.1). The results of those five cases are summarised below, and then a description of a sixth case is provided:

- **Case 1 (Normal pit optimisation):** Using $X^p$ blocks and $\beta$ arcs.
- **Case 2 (Opportunity cost):** The software’s inbuilt facilities were used to effect the opportunity cost approach (the method described by J Whittle [62] and Camus [63]) with $X^p$ blocks and $\beta$ arcs. As expected the open pit was smaller than for case 1 due to the effect of the opportunity cost, but the combined value of the open pit and underground mine was greater than the value for case 1. It was observed that there were some isolated underground mine blocks beside the open pit. They contributed to the value calculation, but in practice these isolated blocks would not be economic to mine.
- **Case 3 (Opportunity cost with new model):** $X^p$ and $X^u$ blocks and $\beta$ and $\gamma$ arcs were used for a zero thickness crown pillar. The result was identical to that for case 2 as expected.
- **Case 4 (120 metre thick crown pillar):** As for case 3, but with $\gamma$ arcs modelling a 120 metre thick crown pillar. There was a reduction in value compared to case 3. This result was expected, since further constraints had been added to the optimisation.
- **Case 5 (Well-formed 120 metre thick crown pillar):** As for case 4, but with the addition of $\delta$ arcs to model well-formed crown pillars. The value in this case was lower than for case 4, since a new constraint had been added. The pit depth increased, which was unexpected, but turned out to be correct for this data set.

The results for these cases are shown in Table 3.1.

The processing time for the new method to model a well-formed crown pillar (case 5) was 14 seconds, compared to 8 seconds for case 1. The extra 6 seconds is immaterial. Recall from the Introduction section (3.1), these optimisation runs are conducted many times, but in the context of a strategic planning process that may take months or years. Recall also from the Introduction section that Chung et al. [69] solved a similar (but not identical) problem to case 5 using a purpose-built integer programming model. Their 83,000 block model took 34.5 hours to process. This relatively long time is likely because
3.5 Experimental results

Table 3.1: The results for the five cases

<table>
<thead>
<tr>
<th>Cases (processing time)</th>
<th>Arcs</th>
<th>Mines</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Marvin with vertical extension</td>
<td>$3,298,908$</td>
<td>Pit ore 350 mT</td>
<td>Open pit $1,483m</td>
</tr>
<tr>
<td>(8 sec)</td>
<td>$0$</td>
<td>Pit depth 630 m</td>
<td>Underground N/A</td>
</tr>
<tr>
<td>&amp; $0$</td>
<td>Underground levels N/A</td>
<td>Total $1,483m$</td>
<td></td>
</tr>
<tr>
<td>2: Standard opportunity cost</td>
<td>$3,298,908$</td>
<td>Pit ore 350 mT</td>
<td>Open pit $1,230m</td>
</tr>
<tr>
<td>(8 sec)</td>
<td>$0$</td>
<td>Pit depth 510 m</td>
<td>Underground $830m</td>
</tr>
<tr>
<td>&amp; $0$</td>
<td>Underground levels N/A</td>
<td>Total $2,060m$</td>
<td></td>
</tr>
<tr>
<td>3: First modification - alternative opportunity cost</td>
<td>$6,773,168$</td>
<td>Pit ore 257 mT</td>
<td>Open pit $1,230m</td>
</tr>
<tr>
<td>(11 sec)</td>
<td>$13,524$</td>
<td>Pit depth 510 m</td>
<td>Underground $830m</td>
</tr>
<tr>
<td>&amp; $0$</td>
<td>Underground levels 16</td>
<td>Total $2,060m$</td>
<td></td>
</tr>
<tr>
<td>4: Second modification - simple crown pillar</td>
<td>$6,773,168$</td>
<td>Pit ore 257 mT</td>
<td>Open pit $1,230m</td>
</tr>
<tr>
<td>(11 sec)</td>
<td>$13,524$</td>
<td>Pit depth 540 m</td>
<td>Underground $483m</td>
</tr>
<tr>
<td>&amp; $0$</td>
<td>Underground levels 11-19</td>
<td>Total $1,879m$</td>
<td></td>
</tr>
<tr>
<td>5: Third modification - well-formed crown pillar</td>
<td>$6,773,168$</td>
<td>Pit ore 307 mT</td>
<td>Open pit $1,402m</td>
</tr>
<tr>
<td>(14 sec)</td>
<td>$13,524$</td>
<td>Pit depth 450 m</td>
<td>Underground $452m</td>
</tr>
<tr>
<td>&amp; $12,075$</td>
<td>Underground levels 14</td>
<td>Total $1,858m$</td>
<td></td>
</tr>
</tbody>
</table>

their approach does not allow them to use the available highly efficient solutions to the MGC problem.

A sixth case is now described to provide a reasonable benchmark against which to compare the dollar value of the plan produced by the new method (case 5). Case 6 is based on case 3 and on a hypothesis as to how a mine engineer might manually adjust it to add a well-formed crown pillar. A mine engineer would also remove the aforementioned isolated underground blocks, since they are not economic to mine.

Removal of the isolated underground blocks is straight forward. As to the addition of the crown pillar, there are several options, including the removal of blocks from the underground mine, or the open pit mine, or both. The removal of blocks from the underground mine was chosen for several reasons. Firstly, removal of blocks from a mine reduces its value, and on the assumption that the open pit precedes the underground mine, removal of blocks from the latter is preferred. Secondly, this approach is the only one explicitly described in the literature [70]. Finally, it is the simplest and most easily reproduced method. The result was a value of $1,617m$ for case 6, as compared to $1,858m$ for case 5. This means that the new method (case 5) provides a $241m$ or 15% increase in value compared to the benchmark (case 6). This improvement in value is an example of the efficacy of the new method. The general case for the efficacy is given in the previous sections which prove the mathematical integrity of the approach.
3.6 Discussion

In the Introduction and New Model Sections (3.1 and 3.2.2), it is stated that an underground mine plan is required as input (specifically underground mining values for blocks) and that the details of this plan are not matters of concern. However, when the transition depth is optimised by running my new method, blocks are assigned to the open pit mine that may have been included in the initial underground mine plan. This reduction in blocks in the underground mine may or may not materially change key assumptions. For example, suppose the initial underground mine plan was made on the assumption that there would be 5 million tonnes of ore and then the transition optimisation reassigns 0.5 mT of this to the open pit. Then a mine engineer may determine that the unit costs of mining (including development) used in the initial underground mine plan still hold, the result can be accepted. Now suppose the transition optimisation cuts the underground mine down to 2mT of the deepest ore. This is a substantially different underground mine, with more development required to reach the first ore, and much less ore across which to spread development costs. In this case, a mine engineer may determine that unit costs for the underground mine must be increased and a new underground plan developed.

In the Introduction to this chapter, the work of King, Goycoolea & Newman [73] was described. They optimise the open pit and underground mining schedules, the placement of the crown pillar and also the placement of sill pillars (recall a sill pillar is material left in-situ in a particular type of underground mine to allow for a change in mining direction). Their method cannot handle very large models but they overcome that by producing a small number of open pit scheduling blocks. My new method, described in this chapter, can handle very large models, but does not optimise directly for NPV. There is scope for the combination of my new method and King et al.’s. Firstly, use my new method as part of the generation of the open pit scheduling blocks. This could be done by running the my new method several times with different open pit prices to generate a family of pits. Then, use that family of pits to produce open pit scheduling blocks, with optimisation methods well-known in the mining industry. These open pit scheduling blocks would have shapes that respond to the opportunity costs. Secondly, modify the
3.7 Conclusions

King et al. method slightly to only require a crown pillar above the underground mine. Using this combination, it is believed that mine planners could access the full benefits of both of these very different approaches.

The new model and method in this chapter can be applied with the unconventional application of commercially available software. I have demonstrated it with a commercially realistic orebody model. With reference to Appendix B: Technology readiness levels, I believe the model and method to be at technology readiness level (TRL) 5. The application is relatively complicated, but is within reach of advanced mine planners with some time to apply to the problem. As such, TRL 6 is within reach. More widespread application and achievement of TRL 7+ will require some software development from one of the commercial mining software providers.

I conclude this section with some comments on the commercial relevance of this work. Mining companies can improve the value of their combined open pit and underground mines by solving the transition problem. Whilst there are methods in the literature to solve the transition problem directly for maximum NPV, they generally suffer from an inability to use commercial-scale models and they treat the crown pillar in a very simplistic way. Of all the methods for solving the transition problem, only the cash-maximising opportunity cost approach is currently widely available to mine planners through regular pit optimisation software. The new method in this chapter allows mine planners to extend the opportunity cost model to include detailed geometric and economic consideration of crown pillars with specific shapes. In the test case a 15% increase in cash value was achieved.

3.7 Conclusions

In this chapter the mathematical integrity of a new method to solve the transition problem is established. The new method includes three modifications to the normal model used for pit optimisation (solving for the maximum graph closure problem). The modifications allow the algorithm to take account of the underground mining value of a block; to take account of the requirement for a crown pillar with a specified thickness; and to
impose a shape to the crown pillar above the underground mine.

The modifications include:

- The use of an additional set of vertices \( X^u \), with each additional vertex representing the opportunity cost of mining a block by open pit method (the opportunity cost being the lost opportunity to mine a block by underground method).
- The use of an additional set of \( \gamma \) arcs to connect vertices representing open pit values to vertices representing the aforementioned opportunity costs.
- The use of a further set of \( \delta \) arcs to create NSCSs, each of which defines a required shape for the crown pillar above the underground mine at a given level.

Theorem 3.2.1 proves that the general form of the new model (not including the NSCSs) can be reduced to an MGC problem. Theorem 3.4.4 proves that NSCSs can be removed from the MGC problem, by collapsing each SCC down to a single representative vertex, reducing the size of the input to the MGC problem and obviating the need to prove that any given approach to solving the MGC problem can correctly deal with NSCSs.

There is good potential for this new method described in this chapter to make positive contributions to the value of a mine plan. In a test case, a 15% increase in cash value was achieved.
Chapter 4

Underground mine with shaft access

This chapter describes a new decomposition and method to optimise the layout of an underground mine, where the main access is via shaft.

4.1 Introduction to the decomposition

Before describing the decomposition some terms are defined: In the interests of brevity reference to a polyhedron means a stope or cave polyhedron. Recall from Chapter 2 a drawpoint is a place where ore can be loaded and removed. A level block is a block that is on a user-defined level and is a potential location for a drawpoint. A prize collecting Euclidean Steiner tree (PCEST) is an efficient geometric network on an optimally selected set of terminals. A rooted PCEST is a PCEST for which there is one mandatory terminal. These last two terms are more formally defined in Chapter 6.

The decomposition consists of executing the following three optimisation steps in order:

1. Potential polyhedron optimisation: The identification of potential polyhedrons between each pair of user-defined levels is performed without regard to the costs associated with gaining access to drawpoints on the levels. Value from the polyhedrons accumulate to level blocks, which are either drawpoints or proxies for drawpoints (depending on which optimisation method is applied).

2. Level development and level block selection: Optimisation of these elements is performed for each defined level. The optimisation can be treated as a rooted PCEST problem.

3. Shaft depth optimisation: Given values accumulated to each level as a result of the
previous optimisation step, it is a trivial matter to determined how deep the shaft should go.

Details of all of the above optimisation steps are provided in the sections that follow.

### 4.2 Potential stope or cave polyhedron optimisation

In this section, I briefly comment on the applicability of existing methods for the optimisation of potential stope or cave polyhedrons in the context of the new decomposition. In addition I describe a new method. The conclusion is that either by adapting existing methods, or implementing a new method, this optimisation component of the decomposition is relatively easily achieved: It does not represent a barrier to the implementation of the decomposition.

#### 4.2.1 Existing methods

Of the methods discussed in Section 2.4 “Stope polyhedron optimisation” it appears the following methods could be used to perform potential stope or cave polyhedron optimisation in the context of the proposed decomposition:

- Alford & Hall’s stope shape annealing process [16].
- Thomas & Earl’s “StopeSizor” [14].

The main attributes that make these methods suitable is their general applicability to a wide variety of mining methods, and their ability to take user-defined level separation as an input to the optimisation. Both methods are discussed in more detail in Section 2.4. Both methods yield polyhedrons as sets of blocks representing the full extent of a stope or cave between levels. Both methods claim, but do not provide details for, a range of constraints modelling capabilities, such as the ability to accept minimum and maximum sizes for, and separation distances between polyhedrons.

If one of the above methods is used in the decomposition, the underground values of each block included in a polyhedron and their respective drawpoints are of particular interest. The underground value for block $i$ is $m_i = r_i - c_i$ where $r_i$ is the revenue and $c_i$ is the extraction cost. As in the open pit case (Section 3.2.1), assume that the underground
value is not time dependent. A block’s underground value will be achieved if and only if it is included in a polyhedron, or if it mined as a level block. The drawpoints occur on a level, and the value of the rock drawn through a drawpoint can be accumulated and attributed to the drawpoint. It is this information that is required for the next step in the decomposition.

A disadvantage of trying to apply these existing methods in the decomposition is that the details of their mathematics and implementation have not been published. Accordingly, their behaviour in the context of the decomposition would be difficult to predict in detail. Furthermore, neither method is an exact optimisation. Alford and Hall employ simulated annealing. Thomas and Earl do not describe their method in detail, but claim only “close to optimal” results.

4.2.2 A proposed new method

As an alternative to the methods outlined in Section 4.2.1, it is proposed that a graph theoretical model is employed. As in Section 3.2.1 Pit optimisation model, a regular block model is to be used to represent the material in the ground and an equivalent digraph is to be used to frame the optimisation problem. Underground values can be assigned to each block in the block model. In the underground mine, in order to mine any block, the blocks below it must also be mined, all the way down to a level block. These mining dependencies can be modelled with directed edges (arcs) in the digraph representation of the block model as illustrated in Figure 4.1.

Recall the definition of a Closure (Definition 3.2.3). Recall also that a closure of a graph need not be contiguous. The question is, what closure of the graph maximises the combined value of the selected points?

Stated another way: What is the maximum graph closure?

A formal definition for a Maximum graph closure is given in Definition 3.2.4. Whilst the context for that definition is an open pit model / pit optimisation, it is also applicable to this case.

The model described thus far is very simple, with only downward pointing arcs. The model can be made more useful (and more complicated) with the introduction of surfaces
to model foot walls\(^1\) and hanging walls\(^2\). An illustration of their intended effect on the generation of arcs in the model is given in Figure 4.2.

Some other dependencies can be modelled using arcs:

- The use of horizontal arcs adjacent to foot walls to force a minimum width to the polyhedron.
- The removal of arcs from the blocks under a level to model a crown pillar\(^3\).

Methods involving modifications to the block model prior to the optimisation can be applied to simulate different draw-down dynamics:

- We can assume that material mixing varies as a function of proximity to drawpoints, hanging wall and foot wall. Accordingly it is possible to precondition a model by mixing the contents of blocks with the contents of neighbouring blocks using a function based on these distances.

\(^1\)A foot wall, simply defined, is a geologically constrained surface below which rock will generally not flow to a drawpoint.

\(^2\)A hanging wall, simply defined, is a geologically defined surface, such that rock above the surface will generally not fall.

\(^3\)Recall from Chapter 3 a crown pillar is a body of rock left in place. In this case, it at the shallowest part of a polyhedron to ensure stability for the drawpoints and tunnels in the level above.
Mining dilution with waste and mining loss can occur to blocks near a hanging wall or foot wall. Accordingly a preconditioning function for dilution and loss can be applied.

In contrast to the existing methods (Section 4.2.1) polyhedrons generated by this proposed method are more elemental, each comprising a tree of blocks, with exactly one level block each. A set of contiguous level blocks and their respective trees of blocks, represents a set of drawpoints and an associated polyhedron. There is not necessarily a one-to-one relationship between drawpoint and level block.

In comparison to the application of the maximum graph closure model to the open pit optimisation problem (see Section 3.2.1), this application is simple, producing a small number of blocks, and a small number of arcs per block. A smaller number of blocks is required since only blocks between levels are used in this underground model, whereas in the open pit model all blocks in the mining area are required. Between 1 and 2 arcs per block will be used in the underground model compared to between 15 and 50 arcs per block for the open pit model. There are several methods available to solve maximum graph closure problems. For example in 2017, Bai, Turczynski, Baxter, Place, Sinclair-Ross & Ready report on the implementation of the Hochbaum algorithm “pseudoflow”
It can solve a problem with 0.5 million blocks and 7 million arcs in 16 seconds. It can solve a problem with 21 million blocks and 348 million arcs in under 13 minutes. The larger of these two is a very large model in open pit terms.

Advantages of this proposed method compared to the two methods discussed in Section 4.2.1 include:

- The detailed mathematics for several exact solutions to the maximum graph closure problem are readily available.
- It is envisaged that the method will be relatively simple to implement and therefore obviates the need to access and integrate proprietary software.
- Availability of more elemental polyhedrons, may be an advantage in the decomposition, as it leaves more of the selection of polyhedrons to the next step in the decomposition, which optimises the selection of polyhedrons as well as the network that connects them to the shaft.

Disadvantages of the proposed method include:

- The constraints modelling methods devised so far are modest (vertical arcs, with modifications for foot walls and hanging walls).
- Some software development is required (albeit relatively straight forward).

The only comparable method found in the literature to the one proposed here is a method developed by Bai, Marcotte & Simon [27]. Their model uses concentric cylindrical blocks to approximate shapes relevant to a particular mining method: sub-level stoping. The optimisation model described here operates on a more conventional regular rectangular block model and is intended for application to a wide variety of mining methods.

### 4.3 Level development and level block selection

A level is a horizontal or near-horizontal surface for which the following information is assumed:

- A position for the shaft (specified in advance of the optimisation).
- A set of level blocks and their accumulated values. These are available from the
previous step in the decomposition: Potential stope or cave polyhedron optimisation (Section 4.2).

• Cost per unit of distance for development of tunnels.

Refer to Figure 4.3 for an illustration of a level.

![Figure 4.3: Plan view of a level, with shaft location and some level blocks with (unspecified) positive value highlighted.](image)

The question is, which selection of level blocks can be made, such that, when they are connected to the shaft by an efficient network, gives the highest net value?

Net value is defined as the accumulated values for the selected level blocks, less the cost of the network to connect them to the shaft. The cost of the network is the length of the tunnels multiplied by the cost per unit of distance for development. This problem is equivalent to a rooted PCEST problem. The PCEST problem (inclusive of the rooted variant) is a topic to which three chapters of this thesis are devoted:

• Chapter 6: Prize collecting Euclidean Steiner trees
• Chapter 7: A universal constant for replacement argument A
• Chapter 8: A universal constant for replacement argument B

The result of such an optimisation is illustrated in Figure 4.4.

The net value resulting from the optimisation represents the value of breaking out of
Figure 4.4: An efficient network (a minimum Steiner tree in this example) on the shaft and a selection of the positive value level blocks.

In the literature, the optimisation approach that is most similar to the one proposed here is given by Li [51]. Recall from Section 2.6, Li models the level development and potential stope selection problem as a prize collecting Steiner tree in graphs problem and he proposes a solution for a specialized version of this problem, the \textit{k-cardinality prize collecting Steiner tree in graphs} problem. By solving this special version \( n \) times, where \( n \) is the number of potential stopes, the general problem can be solved. The differences between the method proposed by Li and my approach are that Li requires a network that connects all potential stopes to start with, from which he finds a solution in a subtree of that network. Li does not specify a method to generate the starting network. Suppose the starting point is a minimum length network: In finding a subtree there is no guarantee that a subtree in minimal and it follows that there is no guarantee that the selection of potential stopes is also optimal. In contrast, the method that I have proposed will guarantee a minimum length network connecting the optimally selected subset of potential stopes.
4.4 Shaft depth optimisation

Suppose levels are numbered 1, 2, ..., n starting from the level closest to the surface. Let \( v_1, v_2, ... v_n \) denote the net values of breaking out of the shaft at levels 1, 2, ..., n found in the previous step of the decomposition. Let \( c_1, c_2, ..., c_n \) denote the costs associated with reaching each successive level from the surface. Cost \( c_1 \) includes the cost of building all necessary surface infrastructure, the cost of dropping the shaft to level 1 and the cost to construct the shaft-related infrastructure at level 1. Cost \( c_2 \) is the cost to drop the shaft from level 1 to level 2, and the cost to construct the shaft-related infrastructure at level 2, and so on for costs \( c_3, ..., c_n \). Cost \( c_s \) is the cost to construct a sump\(^4\) for the shaft. A sump will be required if a shaft is dropped, no matter how far it is dropped. Then the problem of deciding the shaft depth is as follows:

Find the depth \( e \) of the shaft (where \( e \) is an integer number of levels) such that the net value of the mine \( v_m \) is maximised:

\[
v_m = \sum_{i=1}^{i=e} (v_i - c_i) - \min (1, e)c_s, 0 \leq e \leq n \quad (4.1)
\]

The number \( n \) of possible levels in a mine would rarely exceed 50. Even if the problem size was many orders of magnitude larger, the problem would be a trivial one to solve with enumeration. \( e = 0 \) is a possible solution, meaning that no depth of shaft has a net positive value.

4.5 Discussion and conclusions

The decomposition is given in three steps, the most complex of which is level development and level block selection. This can be modelled as a PCEST problem to which a further three chapters of this thesis are devoted. The end result of a single run of the three optimisation steps is a complete mine plan for which the value is maximised:

- The optimal depth of the mine in levels.

\(^4\) A sump is a structure at the base of the shaft below the lowest level accessed. It is required for the collection and pumping of ground water and also accommodates components of the lifting gear.
For each level mined, an optimal selection of level blocks (proxies for drawpoints) and an optimal network to connect them to the shaft.

For each level block selected, a value-maximising tree of stope/cave blocks, which, when combined represent the optimal polyhedrons.

Design parameters that are not optimised, and which are to be determined by repeated trials of the optimisation are as follows:

- The shaft location: The shaft location is constrained by factors such as required stand-off from polyhedrons. Since it is not known before an optimisation exactly where the polyhedrons are, the proposed work-flow includes experimenting with the shaft position through repeated optimisation trials until a satisfactory solution is found. A satisfactory solution in this case is one in which the minimum stand-off requirements are met, with minimal impact on the value of the mine.

- Distance between levels / level separation: The level separation is associated with mining method selection (See Section 2.3). Each mining method has associated with it a set of development costs, mining recoveries, mining dilution, mining costs and production rate. The work-flow to effect mining method selection and associated with that, the level separation, costs and other factors, is repeated optimisation trials until the method selection with the highest value for the orebody is found.

This decomposition is unique and appears to have some advantages compared to other decompositions described in Chapter 2. In particular, the decomposition should provide the ability to support method selection effectively, using a single value-driven optimisation process. This chapter provides industrial context for my research into solving the PCEST problem in Chapters 6, 7 and 8. It is beyond the scope of this thesis to fully develop and test the underground optimisation model presented in this chapter. With reference to Appendix B, Technology readiness levels I believe this decomposition to be at TRL 2.
Chapter 5

Underground mine with decline access

This chapter describes a new method to optimise the layout of an underground mine, where the main access is via decline. It is derived from the method to optimise the layout of an underground mine where the main access is via shaft and accordingly, draws heavily on Chapter 4.

5.1 Introduction to the decomposition

The proposed decomposition is similar to the decomposition described in Chapter 4: Underground mine with shaft access with one additional step, and some modifications to existing steps:

1. Potential polyhedron optimisation: As for Chapter 4.
2. Level development and level block selection: As for Chapter 4 except that the problem is modelled as a un-rooted PCEST problem, rather than a rooted PCEST problem.
3. Decline optimisation: This new step optimises the routing of a decline to connect the levels to the surface, so as to minimise the cost of the decline.
4. Depth optimisation: Similar to the shaft depth optimisation step in Chapter 4 modified for the use of a decline in this case.

Details of all of the above optimisation steps (other than Step 1, which is the same as for Chapter 4) are provided in the sections that follow.
5.2 Level development and level block selection

Level development and level block selection for this decomposition are the same as the optimisation described in Section 4.3, with one exception. In place of a predefined shaft position, there is an as-yet undefined connection to a main decline at each level. Accordingly, the problem is to be modelled and solved as a un-rooted PCEST problem, rather than as a rooted PCEST problem.

5.3 Decline optimisation

A useful benchmark “best case” cost for each decline segment can be calculated as \( \frac{cd}{\sin \theta} + e \) where \( c \) is the cost per unit of length of decline development, \( d \) is the vertical distance between levels, \( \theta \) is the maximum angle from horizontal for the decline and \( e \) is any extra costs for the decline segment, such as the cost to break out at a level.

For an optimal decline with minimal cost, that is also subject to a set of practical constraints, it is proposed that an existing optimisation method is applied: The Decline optimisation tool (DOT) [50]. Recall from Chapter 2 A problem instance for DOT includes the 3D coordinates for the surface portal, and groups of nodes, where each node is an optional connection point for the decline to a cross-cut for a level. A problem instance also includes navigation constraints, boundaries of no-go regions and a cost model. The problem instance inputs are partially supplied by the level definitions. Other details will need to be provided as additional inputs to the optimisation. The outputs include the decline design, the cost for the decline section that connects the surface to the first level, and the costs for the decline sections connecting each adjacent pair of levels.

Note in the preceding description, there is a simplifying assumption of there being one decline. Many mines have a network comprising more than one decline. DOT can deal with such cases.
5.4 Depth optimisation

The depth optimisation for this decomposition is the same as in Section 4.4: Shaft depth optimisation, other than the levels being connected by decline sections in this case, rather than by shaft sections.

5.5 Discussion and conclusions

As for Chapter 4, this chapter provides industrial context for my research into solving the PCEST problem in Chapters 6, 7 and 8. It is beyond the scope of this thesis to fully develop and test the underground optimisation model presented in this chapter. With reference to Appendix B: Technology readiness levels, I believe this decomposition to be at TRL 2.
This page intentionally left blank.
Chapter 6
Prize collecting Euclidean Steiner trees

This chapter describes a new algorithmic framework for solving the Prize collecting Euclidean Steiner tree (PCEST) problem.

6.1 Introduction

An instance of a Prize collecting Euclidean Steiner Tree (PCEST) problem is a set of points in the plane, each with a point weight. Of interest are connected networks on some subset of these points. The networks can include additional vertices called Steiner points with zero weights if their inclusion yields a shorter network. The network itself is necessarily a tree. The value of the network is calculated as the sum of the point weights in a selected subset, minus the sum of the lengths of the edges in the network. The question is: What point selection and connected network has the highest value?

My research covers two variants of the PCEST problem:

- The rooted PCEST problem includes one mandatory point in the problem instance. This variant is relevant to Section 4.3 of Chapter 4: Underground mine with shaft access. Recall in section 4.3 we are interested in selecting level blocks, such that, when they are connected to the shaft by an efficient network, the selection, less the cost of the network, has the highest net value. Each level block can be treated as a point in the plane with a point weight in a rooted PCEST problem instance. The shaft location is represented in the PCEST problem as the mandatory point, with no point weight. In section 4.3 we are interested in an efficient network, and a rooted PCEST delivers the most efficient network of all, a minimum Steiner tree.
The rooted PCEST problem is formally defined on page 72

- The un-rooted PCEST problem does not include a mandatory point in the problem instance. This variant is relevant to Section 5.2 of Chapter 5. Underground mine with decline access. Recall from section 4.3 we pose a similar problem to that in the above-mentioned section 5.2. The only difference is that rather than having a fixed shaft location, we wish to connect the network to a decline. Since a mine planner has some flexibility in locating the intersection of a decline with a level, we can consider this problem to be un-rooted. The un-rooted PCEST problem is formally defined on page 72.

The rooted PCEST problem is easier to solve than the un-rooted variant. However, there is a straightforward way to extend a rooted PCEST solution to the un-rooted case, and this is given in Section 6.19: Solving the un-rooted PCEST problem.

In the interests of brevity if the term “PCEST” is used without qualification, it can be taken to mean either rooted PCEST or un-rooted PCEST.

Exact solutions to the PCEST problem are absent from the literature, however Remy & Steger published in 2009 [77]: Approximation schemes for node weighted geometric Steiner tree problems. The paper’s title belies the fact that the range of problems covered by their method includes the prize collecting geometric Steiner tree problem. They allow for node weights (penalties paid for each Steiner point included in the solution) and point weights paid for each point not included in the solution. By setting the penalties for Steiner points to zero, their approximation scheme applies to the PCEST problem. For 2-dimensional problems Remy & Steger were able to devise a polynomial time approximation scheme, as a variant of an approximation scheme devised earlier (1998) by Arora [78]. However, this chapter is on an exact solution to the PCEST problem rather than an approximation scheme, and so no further discussion of approximation schemes is included.

Whilst exact solutions to the PCEST problem are absent from the literature, some attention has been paid to the related node weighted geometric Steiner tree problem. A node weighted geometric Steiner tree differs from a PCEST in that each Steiner point carries a penalty, rather than each potential terminal having a prize. Regrettably, it turns out that finding a solution to the node weighted geometric Steiner tree problem is difficult.
Steiner points in a node weighted Euclidean Steiner tree can be degree-3 or 4\(^1\) whereas a (non-weighted) Euclidean Steiner tree can only have degree-3 Steiner points (Brazil & Zachariasen 2015 [79], Rubinstein, Thomas & Weng 1992 [80]). The possibility of degree-4 Steiner points changes a sub-problem (construction of Steiner trees for a given topology) from one that is known to run in polynomial time in the classic Euclidean Steiner tree problem, to one that is not known to run in polynomial time (Underwood [81]). Although an algorithm has been devised to solve the node weighted Euclidean Steiner tree problem, in practice it is only capable of solving problems with up to ten terminals (Xue [82]). The use of penalties on Steiner points has some appeal in the proposed application at hand (The aforementioned level development problems): A junction in a network of tunnels has some additional costs associated with its construction and any such real costs have a place in the formulation of a comprehensive solution. However this is a second-order issue compared to the inclusion of the value of including points in the solution. Given the difficulty of the node weighted problem and that it is a second order issue, it is not given further consideration in this chapter.

The solution to the PCEST problem in this chapter is build on the foundation of a rich body of literature on the classic Euclidean Steiner tree problem and to some extent, on a solution to the related prize collecting Steiner trees in graphs problem. These topics are reviewed in Sections 6.2 and 6.3 respectively. Much of the rest of this chapter is devoted to reducing the complexity of the problem by classifying as many given points as possible as either ruled in or ruled out of a solution to the PCEST problem. Each time a point is classified as ruled out, or ruled in to a solution, the computational burden is lifted. It is this model preconditioning, supported by two additional Chapters [7] and [8], that are the most challenging aspects of finding a scalable method for solving the PCEST problem.

For the convenience of readers of Chapters 6 to 8, glossaries of more frequently used abbreviations and symbols are included in Tables A.2 to A.4 of Appendix A.

\(^1\)The degree of a point is the number of edges incident to it.
6.2 Minimum spanning trees and minimum Steiner trees

In this chapter and in chapters 7 and 8 the Euclidean plane is the only norm considered. In the sequel, in the interests of brevity, the Euclidean plane is assumed unless otherwise stated.

**SPANNING TREE PROBLEM**

- **Input:** A set of points $N$ lying in the plane
- **Question:** What geometric network $T = (V(T), E(T))$, such that $N = V(T)$ minimises the sum of the lengths of the edges $E(T)$?

A solution to the spanning tree problem is called a *minimum spanning tree* (MST).

**STEINER TREE PROBLEM**

- **Input:** A set of points $N$ lying in the plane
- **Question:** What geometric network $T = (V(T), E(T))$, such that $N \subseteq V(T)$ minimises the sum of the lengths of the edges $E(T)$?

A solution to the Steiner tree problem is called a *minimum Steiner tree* (MST).

Note the important distinction between the above two problem definitions: An MST can include additional vertices, whereas an MST cannot.

There is a rich body of literature on the topic of MSTs. This section briefly covers the concepts that are important to the development of a PCEST solution in the sequel. We recommend to readers interested in a more comprehensive coverage of MSTs and many other types of efficient networks to consult the book *Optimal Interconnection Trees in the Plane* by Brazil & Zachariasen [79].

It is permissible to include additional vertices in the solution to a Steiner tree problem, if doing so yields a shorter connected network. The points provided in the problem instance are known as *terminals* and any additional vertices are known as *Steiner points*.

A problem instance with two terminals $a$ and $b$ has a trivial solution of the line segment $\overline{ab}$. A problem instance with three terminals is known as the *Fermat-Torricelli problem*: Pierre de Fermat is said to have posed the problem in the 1600s and Evangelista
6.2 Minimum spanning trees and minimum Steiner trees

Torricelli gave a geometric solution, also in the 1600s (Krarup [83]). Provided no angle between the three terminals exceeds $\frac{2\pi}{3}$, the solution is a star $T$ with a single Steiner point at the centre with degree-3. The angles between each pair of edges incident to the Steiner point are all $\frac{2\pi}{3}$. This solution is a full Steiner tree (FST): A tree in which all terminals have degree-1 and all Steiner points have degree-3. If a sub-tree of a MStT has only degree-1 terminals, the sub-tree is called a full component. Any MStT can be uniquely decomposed into full components intersecting only at terminals. Thus the aforementioned tree is a single FST, which is necessarily comprised of one full component. If, in the case of three terminals, an angle between a pair of terminals is greater than $\frac{2\pi}{3}$, then the MStT is comprised of two full components, each consisting of a single line segment. It is easy to see that an MStT in this case is the same as an MST.

The underlying graph of an MStT is known as a Steiner topology. A full Steiner topology is a topology for an FST. For any set of terminals there is a super-exponential number of potential full topologies, however, most of these potential topologies cannot be drawn without overlapping or crossing lines and are therefore not minimal. The Melzak-Hwang algorithm can be used to construct a full Steiner tree for a given topology and set of terminals, or decide that no such full Steiner tree exists. More details of the Melzak-Hwang algorithm that are pertinent to this chapter are given in Section 6.2.1. For any given Steiner topology without crossing edges (not necessarily corresponding to an MStT), there exists a tree that has minimal length. Such a tree is called a relatively minimal Steiner tree. A pair of terminals adjacent to a single Steiner point in a Steiner topology, and the two edges between them, is called a cherry.

An MStT is at most as long as an MST, but it can be considerably shorter. The question of how much shorter has been studied, with the Steiner ratio conjecture claiming that the length of an MStT can be no less than $\frac{\sqrt{3}}{2}$ times the length of an MST on the same set of terminals (i.e. The Steiner ratio is greater than or equal to $\frac{\sqrt{3}}{2}$). The Steiner ratio conjecture is widely considered to be true, and has been confirmed for cases with eight or fewer terminals by Kirszenblat [85] in 2014, with earlier authors paving the way for seven or fewer (de Wet [86]) and six or fewer terminals (Rubinstein & Thomas, [87]). For

\footnote{Originally given by Gilbert & Pollack in 1968 [84]. A brief history is given by Brazil & Zachariasen in 2015 [79] pp. 23-24}
cases with more than eight terminals, the lower bound for the Steiner ratio is 0.82416874... (Chung & Graham [88]).

Garey, Graham & Johnson [89] proved that all Euclidean Steiner tree problems are NP-complete. Despite this, there exist efficient solutions for most instances, most notably, GeoSteiner. This is a software package that provides exact solutions to several Steiner tree problems, including the Euclidean Steiner tree problem (See Brazil & Zachariasen [79] and Warme, Winter & Zachariasen [90]). The solution method for solving the PCEST problem in this chapter relies in part on a modification to the generation and concatenation functions in the GeoSteiner package, discussed in Sections 6.17 and 6.18 respectively.

### 6.2.1 Melzak-Hwang Algorithm

This section describes and elaborates on an algorithm that is referenced in this chapter, and in Chapters 7 and 8. In 1961, Melzak [91] proved that for any given set of points $X$ embedded in the plane, there is a finite number of Euclidean constructions that yield all minimum Steiner trees for $X$. As part of his proof he described an algorithm to find the unique relatively minimal full Steiner Tree $S$ for any $X$ and a corresponding full Steiner topology $T$, or deciding that no such Steiner tree exists. Melzak’s algorithm was effective but inefficient. In 1986 Hwang [92] published a method to determine an order of constructions that can be completed in linear time. The resulting algorithm is known as the Melzak-Hwang algorithm. The description of the algorithm below is largely based on a recent explanation in [79], which in turn is based on a comprehensive examination of the long and discontinuous history of developments in the field in Brazil et. al. [93]. The Melzak-Hwang algorithm is comprised of two phases, merging and reconstruction. The interest in this thesis is principally in the merging phase from which we derive three Lemmas [6.2.1],[6.2.2] and [6.2.3]. A brief description of the reconstruction phase is included for completeness.
Merging phase

Consider a set of terminals $X$ and a full Steiner topology $\mathcal{T}$ for $X$. It is well known that in a full Steiner topology, there are $n - 2$ Steiner points. The merging phase comprises $n - 2$ merging steps, the first of which replaces $\mathcal{T}$ with a new topology $\mathcal{T}'$. The two terminals of a cherry and the cherry’s Steiner point in $\mathcal{T}$ are replaced with a pseudo terminal in $\mathcal{T}$. The three edges incident to the cherry’s Steiner point in $\mathcal{T}$ are replaced with a single edge incident to the pseudo terminal, and to the third point incident to the Steiner point in $\mathcal{T}'$. The subsequent steps commence with the topology yielded by the previous step, and proceed in the same way as the first step. After $n - 2$ merging steps, the final topology comprises one original terminal, one pseudo terminal and one edge connecting the two, all embedded in the Euclidean plane.

The position of the pseudo terminal is given by the third point of an equilateral triangle (the equilateral point), in which the other two points of the triangle are each a terminal (or pseudo terminal) in the cherry. Observe that any pair of terminals can be the vertices to two equilateral triangles, only one of which correctly gives the pseudo terminal position for a given $\mathcal{T}$. Melzak’s 1961 algorithm [91] did not determine which was the correct triangle and consequently it was necessary for the algorithm to test $2^{n-2}$ potential cases. Hwang’s main contribution in [92] was a method to order the construction of triangles, such that the correct choice could be made every time. Hwang’s method commences with the arbitrary choice of a terminal, which is never removed by a merging step.

All steps of a merging phase are illustrated in 6.1. In (i), the original topology is shown, with Steiner points (black dots) not embedded in the plane. Illustrations (ii) to (v) show the merging steps, resulting in a single Simpson line connecting an original terminal (the arbitrarily chosen terminal in Hwang’s method) and a pseudo terminal. Denote this tree as $\tilde{S}$.

**Lemma 6.2.1.** For any terminal $t \in V(S)$ it is possible to devise a merging order for the Melzak-Hwang Algorithm that yields a Simpson line $\tilde{S}$ such that $t \in V(\tilde{S})$.

**Proof.** The proof follows immediately from Hwang’s construction technique, which commences with the arbitrary choice of a terminal, and the fact that this terminal is never in
a cherry that is merged.

**Lemma 6.2.2.** \( L_{\tilde{S}} = L_S \) when the Melzak-Hwang Algorithm is used to construct \( \tilde{S} \) from \( S \).

**Proof.** It is sufficient to show that the length of the Simpson line following a merging step is equal to the length of the tree before the merging step. According to Brazil & Zachariasen [79] this was first proven by Simpson in 1750, later by Heinen in 1834 and by several others. An elegant proof known as the rotation proof is reproduced in [79]. The rotational proof was discovered at least twice independently with the first possibly being by Hoffman in 1890.

**Lemma 6.2.3.** A sufficiently small perturbation of the position of some terminal \( t \in V(S), t \in V(\tilde{S}) \) will change the length of \( S \) and \( \tilde{S} \) equally.[3]

**Proof.** If \( e \) is a leaf edge incident with a given terminal \( t \) then there exists a Simpson line from \( t \) to a point \( x \) obtained by performing merging operations on the rest of the tree. If \( t \) is perturbed a small distance but all other terminals are kept fixed, then the point \( x \) on the Simpson line also remains fixed. It follows that the change in length of \( S \) under the perturbation exactly corresponds to the change in length of \( xt \). This is true as long as the movement of \( t \) is sufficiently small that it does not cause \( S \) to become degenerate.

**Reconstruction phase**

Figure 6.2 shows the reconstruction steps for the same example for which merging steps were shown in Figure 6.1. The reconstruction proceeds in the reverse order to merging. In (i) the first merging is shown. The Euclidean position of the Steiner points (shown as a red dot) can be found at the intersection of the Simpson line \( \overline{ae} \) and the minor arc between \( e_2 \) and \( e_3 \) of a circumcircle of \( \triangle e_2e_3e_4 \). In (ii) to (iv) the remaining reconstruction steps are shown. The resulting Full Steiner tree in which all Steiner points are embedded in the plane is shown in (v). Compare this to the starting point in Figure 6.1 (i) in which the Steiner point locations are not known.

---

[3]This Lemma is based on one of two parts of Lemma 2.2 in [94].
Figure 6.1: Illustration of all steps in a merging phase for a topology for a size-6 FST.

Figure 6.2: Illustration of reconstruction steps after the merging steps in Figure 6.1.
6.3 Prize collecting Steiner trees in graphs

The prize collecting Steiner tree in graphs (PCSTG) problem is of interest since it is possible to formulate concatenation for PCEST (See Section 6.18) as a PCSTG problem. Unlike its geometric counterpart, the PCSTG problem has received some attention in the literature. Early work focused on approximation schemes (e.g. [95], [96]). Fast heuristics have application in some settings and a recent example is Sun, Brazil, Thomas & Halgamuge (2019) [97]. However, for the purposes of solving the concatenation sub-problem for PCEST, an exact solution to PCSTG is ideal. In 2005, Ljubic et. al. [98] devised and demonstrated a branch and cut algorithm to solve large PCSTG problem instances to optimality. In 2017 Gamrath et. al. [99] published details of their solver “SCIP-Jack”. The solver is capable of solving a wide variety of Steiner tree problems in graphs, including PCSTG and the rooted variant thereof. Gamrath’s approach is discussed in more detail as a method to effect concatenation for the PCEST problem in Section 6.18.

6.4 Prize collecting Euclidean Steiner tree problem overview and preliminaries

**UN-ROOTED PRIZE COLLECTING EUCLIDEAN STEINER TREE (PCEST) PROBLEM**

*Given:* A finite set of points $N = \{n_i\}$ each with a corresponding weight $w_i \in \mathbb{R}$

*Find:* A connected geometric network $P = (V(P), E(P))$ embedded in the plane and a subset $\hat{N} \subseteq N$ such that $\hat{N} \subseteq V(P)$ and $|P| := \sum_{i:n_i \in \hat{N}} w_i - \sum_{i,j:(v_i,v_j) \in E(P)} |(v_i, v_j)|$ is maximised, and such that for each $P^*$ where $\hat{N}^* \subset \hat{N}$, $|P^*| < |P|$.

The points included in the problem instance are called *possible terminals* and the points included in $\hat{N} \subseteq N$ are called *terminals*.

The last condition for the un-rooted PCEST problem ensures a *parsimonious* network. That is, a maximum value network without unnecessary terminals. The condition is framed to be consistent with the methods developed in this thesis, but it does not guarantee that the solution will have the least possible number of terminals. This is because it
is possible to have two maximal solutions with different number of terminals, where one set of terminals is not a subset of the other. An example of such a case is given in Section 6.5 Figure 6.5.

**Definition 6.4.1** (Maximum un-rooted PCEST). A maximum un-rooted PCEST is a network embedded in the plane that solves the un-rooted PCEST problem.

The un-rooted PCEST problem is germane to Chapter 5. A variant, the rooted PCEST problem is germane to Chapter 4. A rooted PCEST has a single mandatory terminal. Having a single mandatory terminal suits the application (in which the mandatory terminal corresponds to the known location of the shaft), and it also assists in formulating a solution, as will become apparent in the sequel. Let \( n_0 \) denote the mandatory terminal.

When the level development problem in Chapter 4 or Chapter 5 is treated as a PCEST problem, points correspond to level blocks, and edges correspond to tunnels connecting the selected level blocks to the shaft. The model is normalised by calculating the weights \( w_i \) for each block \( n_i \) as the net value of the corresponding level block, divided by the cost per unit of distance for development of tunnels.

**ROOTED PCEST PROBLEM**

**Given:** A finite set of points \( N = \{n_i\} \) each with a corresponding weight \( w_i \in \mathbb{R} \) including a mandatory terminal \( n_0 \in N \).

**Find:** A connected geometric network \( P = (V(P), E(P)) \) and a subset \( \hat{N} \subseteq N \) such that \( \hat{N} \subseteq V(P) \), \( n_0 \in \hat{N} \) and \( |P| := \sum_{i : n_i \in \hat{N}} w_i - \sum_{i,j : (v_i, v_j) \in E(P)} |(v_i, v_j)| \) is maximised, and such that for each \( P^* \) where \( \hat{N}^* \subset \hat{N} \), \( |P^*| < |P| \).

As for the un-rooted PCEST problem, the last condition in the rooted PCEST problem ensures a parsimonious solution.

**Definition 6.4.2** (Maximum rooted PCEST). A maximum rooted PCEST is a network embedded in the plane that solves the rooted PCEST problem.

In the sequel a maximum rooted or un-rooted PCEST will be denoted by \( P \).

When finding a maximum rooted PCEST, it is not necessary to specify the weight
of the mandatory terminal, since no matter what the value, the networks that solve the maximum rooted PCEST problem remain the same.

We observe that a solution to the un-rooted PCEST problem can be found by solving the rooted PCEST problem \( m = \| N \| \) times, where each time a different point is assigned as the mandatory terminal, and the resultant network with the highest value is chosen (a more efficient approach is given in Section 6.19).

A solution to the rooted PCEST problem for \( N \) can be found by assigning a sufficiently high weight to the point in \( N \) that is to be the mandatory terminal and solving the un-rooted PCEST problem for \( N \).

In both the un-rooted PCEST problem and the rooted PCEST problem, \( V(P) \) may include a set of Steiner points \( S(P) \), if their inclusion contributes to the maximisation of \( |P| \). Accordingly, \( V(P) = \hat{N} \cup S \). Note that Steiner points have no weight associated with them.

The classic (Euclidean) Steiner tree problem (page 66) is a special case of a PCEST problem in which all the elements of \( W_N \) are set to sufficiently high values, such that \( \hat{N} = N \). Thus, a PCEST problem is a generalisation of the classic (Euclidean) Steiner tree problem.

### 6.5 Properties of maximum PCESTs

For a given set of points \( N \) there may be more than one maximum PCEST. Figure 6.3 provides an example of two different PCESTs on the same set of points, and in this case, both have the same terminal set. Figures 6.4 and 6.5 provide examples in which two different maximum un-rooted PCESTs have entirely different terminal sets. There cannot be two maximum rooted PCESTs with entirely different terminal sets, since all must include the mandatory terminal \( n_0 \), but the terminal sets for two different PCESTs can still differ and an example is shown in Figure 6.6. This example also works for the un-rooted PCEST problem by replacing \( n_0 \) with a non-mandatory terminal with a high weight (say, 10 in this example).

For multiple PCESTs to be possible, the value \( |P| \) of one solution must be identical
Figure 6.3: The two trees are maximum PCESTs on points \( N = \{n_1, n_2, n_3, n_4, n_5\} \). Both have the same terminal set \( \hat{N} = \{n_1, n_2, n_3, n_4\} \). Points are shown in blue. Blue points in a tree are terminals. Steiner points are shown as smaller black dots.

to the others. In the example shown in Figure 6.3 an arbitrarily small change in the coordinates of \( n_1, n_2, n_3 \) or \( n_4 \) would lead to only one maximum PCEST. In Figure 6.4 or 6.5 an arbitrarily small change in any one of the point weights or coordinates of would leave only one solution. This means that cases of multiple solutions are expected to be very rare for real-world data sets.

**Lemma 6.5.1.** A maximum PCEST \( P \) on \( N \) is an MStT on \( \hat{N} \).

**Proof.** Suppose there exists a maximum PCEST \( P \) with terminals \( \hat{N} \) that is not an MStT on \( \hat{N} \). Recall that Steiner points in a PCEST have no weight. Then the length of \( P \) must be greater than the length of an MStT on \( \hat{N} \) and all the edges and Steiner points can be removed from \( P \) and replaced with the edges and Steiner points from an MStT, thereby increasing the value of \( P \) and leading to a contradiction. 

The value of Lemma 6.5.1 is that it is known that a MStT on some \( \tilde{N} \subseteq N \), is also a maximum PCEST if \( \tilde{N} = \hat{N} \) and hence some of the techniques and algorithms for solving the MstT problem can be applied to the PCEST problems.
Figure 6.4: The two trees are maximum un-rooted PCESTs on \( N = \{n_1, n_2, n_3, n_4, n_5\} \) and have different terminal sets. Note that \(|n_2n_3| = 3\).

Figure 6.5: The two trees are maximum PCESTs on \( N = \{n_1, n_2, n_3, n_4, n_5, n_6, n_7\} \), but with different numbers of terminals in each solution.

Figure 6.6: The two trees are maximum PCESTs on \( N = \{n_0, n_1, n_2, n_3\} \), but with different terminal sets.
6.6 Naïve approach to finding a solution to the PCEST problem

Some subset of points in \( N \) is included as terminals in the solution to the PCEST problem. Accordingly, all of the members of \( N \) can be thought of as possible terminals (a formal definition is given in 6.7.1). Let \( N' \) denote the \( 2^m \) set of subsets of \( N \setminus \{n_0\} \) (that is, the power set of \( N \setminus \{n_0\} \)). Let \( N'_k \) be the \( k^{th} \) element of \( N' \). The naïve approach to finding a solution to the PCEST problem is:

1. Generate \( N' \).
2. For \( k = \{1, 2, ..., 2^m\} \):
   (a) Find an MST \( S \) for \( N'_k \cup \{n_0\} \)
   (b) Calculate the PCEST value \(|S^*|\) for \( S \).
3. Find all cases of \( N'_k \cup \{n_0\} \) with the highest \(|S^*|\). Choose from these cases, one with the fewest terminals. The MST \( S \) for this terminal set corresponds to a maximum PCEST.

Finding an MST in step 2(a) is an NP-Hard problem \cite{94} that must be performed an exponential number of times. Even if GeoSteiner is used (a very efficient program for solving the MST problem), processing times quickly explode. For example, consider a PCEST problem with 30 points. The GeoSteiner documentation \cite{100} reports a 0.02 second computing time for a sample 20 terminal Euclidean Steiner tree. Suppose this is (optimistically) taken to be the mean time to conduct step 2(a) when \( m=30 \). Then the total processing time for step 2(a) would be 0.02 \( \times 2^{30} \) \( \approx \) 21 million seconds, or around eight months.

6.7 Reducing the number of possible terminals

In Section 6.6 it was shown that GeoSteiner could be run \( 2^m \) times in order to find a solution to the PCEST problem, but also that this is an intractable endeavour for large problem sets. The central issue is that the set of points \( N \) contains the terminals of a maximum PCEST, but with the exception of a mandatory terminal (if applicable) it is uncertain as to which points these are. Accordingly, all the points in \( N \) other than the
mandatory terminal, commence as possible terminals unless they are either ruled in or ruled out, as discussed below.

In order to ease the computation burden, it is useful to attempt to identify in advance some points that can be ignored and some points which are in every maximum PCEST. Recall that \( N \) denotes a set of points \( n_i \in N \) lying in the plane. The aim is to partition \( N \) into three subsets: \( N = N_1 \uplus N_O \uplus N_P \) where these subsets are defined below.

**Definition 6.7.1** (Possible terminal). A possible terminal is a point \( n_i \in N_P \subseteq N \) that has neither been ruled in nor ruled out.

**Definition 6.7.2** (Ruled in as a terminal). A point \( n_i \in N_I \subseteq N \) is said to be a ruled in as a terminal (“ruled in” for short) if it has been proven to be in every maximum PCEST.

Ruling in is discussed in Section 6.8.

**Definition 6.7.3** (Ruled out as a terminal). A point \( n_i \in N_O \subseteq N \) is said to be ruled out as a terminal (“ruled out” for short) if it has been proven to be not in any maximum PCEST.

Ruling out is discussed in Section 6.9.

### 6.8 Ruling in

In this section, an efficient method for ruling in terminals for a rooted PCEST is described and trialled. Some of the lemmas apply to both rooted PCESTs and un-rooted PCESTs, in which case the unqualified term PCEST is used. When a point is ruled in as a terminal, the number of possible terminals reduces by one, thereby halving the cardinality of the power set and consequently halving the number of computations needed to solve the rooted PCEST problem.

We commence by observing that by the definition of the rooted PCEST problem the mandatory terminal is ruled in.

#### 6.8.1 Points close together

Points that are sufficiently valuable to pay for their connection to an already ruled-in point, can themselves be ruled in.
Lemma 6.8.1. Let $n_i$ be a point that is ruled in to a PCEST and let $n_j$ be a point with weight $w_j$. If $n_j$ can be connected to $n_i$ with an edge $(n_j, n_i)$ such that $w_j > |n_jn_i|$ then $n_j$ can be ruled in.

Proof. Suppose to the contrary $n_j$ is not a member of a maximum PCEST. Then it can be added to the network with an edge $(n_j, n_i)$, increasing the network’s value and giving a contradiction.

It is useful to cluster together points that can form local valuable networks.

Lemma 6.8.2. Consider a ruled in terminal $n_i$ and possible terminals $n_j$ and $n_k$ with weights $w_j$ and $w_k$ respectively. If $\min\{w_j, w_k\} - |n_jn_k| > 0$ and if $w_j + w_k - |n_jn_k| > \min\{|n_jn_i|, |n_kn_i|\}$ then both $n_j$ and $n_k$ can be ruled in.

Proof. Suppose to the contrary the condition of the lemma is met, and that $n_j$ and $n_k$ are not in a PCEST. Then clearly their addition can be achieved with two edges, such that the sum of the edge lengths is less than the sum of the point weights giving a contradiction.

Lemma 6.8.2 provides a method to rule in pairs of terminals when an attempt to rule them in individually using Lemma 6.8.1 would fail. Take for instance a case in which $w_j = 3, |n_in_j| = 4, w_k = 3, |n_in_k| = 4$ and $|n_jn_k| = 1$.

Remark 6.8.1. [clustering possible terminals] A consequence of Lemma 6.8.2 is that when ruling in, a merging step can be effected if two possible terminals $n_j$ and $n_k$ are found to have a distance between them that is less than the smaller of the two point weights ($\min\{w_j, w_k\} - |n_jn_k| > 0$). The merging assigns to a single point a combination of the characteristics of the original two points, which are then discarded. Denote the new point $n_{js}$ with point weight $w_{js} = w_j + w_k - |n_jn_k|$. The distances between $n_{js}$ and $n_i$ (for $i : n_i \in N_I \cup N_P \setminus \{n_j, n_k\}$) is given by $w_{js} = \min\{|n_jn_i|, |n_kn_i|\}$. If point $n_{js}$ is subsequently ruled in by Lemma 6.8.1 then it can be interpreted as the original points $n_j$ and $n_k$ being ruled in. Clearly this process can be iterated, effectively allowing multiple points to be merged.
6.8.2 The merging and ruling in algorithms for the rooted PCEST problem

As part of this research, algorithms to give effect to Lemmas 6.8.1 and 6.8.2 were developed and programmed in ANSI C. Details of the program are included in Appendix C. The implementation differs slightly from the preceding description in that rather than merging two points to a new third point, one point is merged to the other and the former is deactivated. Consider a set of points \( N_I \cup N_P = \{n_0, n_1, ..., n_m\} \) in the plane with corresponding weights \( W_N = \{w_0, w_1, ..., w_m\} \) where \( n_0 \) is the mandatory terminal. These points can be represented in a point and edge weighted complete graph on \( N_I \cup N_P \). Point weights are initially given by \( W_N \). Edge weights \( c_{ij} \) are initially given the edge length \( |n_i n_j| \).

Pseudo code for merging is given in Algorithm 1 with the following additional variables:

- \( f \) is the merge flag. If a merging has occurred in a loop, then \( f = 1 \), otherwise \( f = 0 \).
- \( m_{ij} \) is a merge variable where \( i, j : n_i, n_j \in N_I \cup N_P, i \neq j \). If point \( n_j \) is merged to point \( n_i \), then \( m_{ij} = 1 \), otherwise \( m_{ij} = 0 \).
- \( t_i \) is a temporary weight for each corresponding point \( n_i \in N_I \cup N_P \).
- \( i, j \) and \( k \) are counters.
- \( a_i \) is the active flag for each \( i : n_i \in N \). If point \( n_i \) has been deactivated then \( a_i = 0 \), otherwise \( a_i = 1 \).

Pseudo code for ruling in is given in Algorithm 2 with the following additional variables:

- \( f \) is the ruling flag. If a ruling in has occurred in a loop, then \( f = 1 \), otherwise \( f = 0 \).
- \( r_i \) is a ruling in variable \( i : n_i \in N_P \). If point \( i \) is ruled in then \( r_i = 1 \), otherwise \( r_i = 0 \).

A number of trials of the algorithm were conducted using randomly generated data sets. In these data sets, points are uniformly distributed inside defined spatial limits and point weights are uniformly distributed between defined upper and lower limits. Exam-

\[^4\text{A standard for the C programming language defined by the American National Standards Institute.}\]
Algorithm 1: Merging

Input
For each $i : n_i \in N$:
- $x_i, y_i$ coordinates from which $c_{ij} = |n_i n_j|$ can be calculated.
- $w_i$ point weight.
- $cat_i$ point category. 0 = ruled out, 1 = possible, 2 = ruled in.

Algorithm
// Initialize active variable $a_i$, temp. weights $t_i$, merging variables $m_{ji}$ and distance variables $c_{ij}$
for $i : n_i \in N \land (cat_i = 1 \lor cat_i = 2)$ do
  $a_i = 1$
  $t_i = w_i$
  for $j : n_j \in N \land (cat_j = 1 \lor cat_j = 2) \land j \neq i$ do
    $m_{ji} = 0$
    $c_{ij} = |n_i n_j|$
  // Merging test loop
  $f = 1$
  while $f = 1$ do
    $f = 0$
    for $i : n_i \in N \land (cat_i = 1 \lor cat_i = 2)$ do
      if $a_i = 1$ then
        for $j : n_j \in N \land (cat_j = 1 \lor cat_j = 2) \land j > i$ do
          if $((cat_i = 2) \land (t_j - c_{ij} > 0) \land (a_j = 1))$
            $\lor (((cat_i \neq 2) \land (t_j - c_{ij} > 0) \land (t_i - c_{ij} > 0) \land (a_j = 1))$ then
            $t_i = t_i + t_j - c_{ij}$
            $a_j = 0$
            $m_{ji} = 1$
            $f = 1$
          for $k : n_k \in N \land k \neq i \land k \neq j$ do
            $c_{ik} = \min (c_{ik}, c_{jk})$
  Output
For each point $n_j$ merged to point $n_i$, the merging variable $m_{ji} = 1$.
For each active point $n_i$, the final (temp.) weight $t_i$.
For each pair of active points $n_i$ and $n_j, j > i$ the final distance variables $e_{ij}$.
Algorithm 2: Ruling in

Input
For each $i : n_i \in N$:
\[ \text{cat}_i \] point category. 0 = ruled out, 1 = possible, 2 = ruled in.
For each ordered pair of points $(n_i, n_j) : n_i, n_j \in N, j \neq i$:
\[ m_{ji} \] the merging variable (generated in Algorithm 1)

Algorithm
// Initialize ruling variables
for $i : n_i \in N$ do
\[ r_i = 0; \]
// Ruling test loop
\[ f = 1; \]
while $f = 1$ do
\[ f = 0; \]
for $i : n_i \in N \land \text{cat}_i = 2$ do
\[ \text{for } j : n_j \in N \land j > i \land a_i = 1 \text{ do} \]
\[ \text{if } \text{cat}_j = 1 \land m_{ji} = 1 \text{ then} \]
\[ r_j = 1; \]
\[ \text{cat}_j = 2; \]
\[ f = 1; \]
Output
For each point $n_j$ ruled in, $r_j = 1$ and $\text{cat}_j$ is changed from 1 to 2.
Table 6.1: General settings for random point generation

<table>
<thead>
<tr>
<th>Variable</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of points to generate</td>
<td>10000</td>
</tr>
<tr>
<td>Spatial limits for points (i)</td>
<td>(0 \leq \text{point}_x[i] \leq 1000) (0 \leq \text{point}_y[i] \leq 1000)</td>
</tr>
<tr>
<td>Lower weight limit for points (i)</td>
<td>(0.1 \leq \text{point}_{value}[i] )</td>
</tr>
<tr>
<td>Coordinates for mandatory terminal (0)</td>
<td>(\text{point}_x[0] = 500) (\text{point}_y[0] = 500)</td>
</tr>
<tr>
<td>Computer details</td>
<td>Intel Core i7-2600 CPU @ 3.40GHz Ubuntu 16.04 LTS operating system</td>
</tr>
</tbody>
</table>

Table 6.2: Ruling in trials

<table>
<thead>
<tr>
<th>Trial</th>
<th>Upper weight limit</th>
<th>Number of points ruled in</th>
<th>Processing time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>&lt; 10</td>
</tr>
<tr>
<td>2</td>
<td>5</td>
<td>0</td>
<td>&lt; 16</td>
</tr>
<tr>
<td>3</td>
<td>10</td>
<td>9</td>
<td>&lt; 22</td>
</tr>
<tr>
<td>4</td>
<td>25</td>
<td>13</td>
<td>&lt; 8</td>
</tr>
<tr>
<td>5</td>
<td>50</td>
<td>577</td>
<td>&lt; 6</td>
</tr>
<tr>
<td>6</td>
<td>100</td>
<td>7920</td>
<td>&lt; 6</td>
</tr>
<tr>
<td>7</td>
<td>250</td>
<td>9415</td>
<td>&lt; 6</td>
</tr>
<tr>
<td>8</td>
<td>500</td>
<td>9863</td>
<td>&lt; 5</td>
</tr>
</tbody>
</table>

The results are shown in Tables 6.1 and 6.2. As can be seen, a large data set comprising 10,000 points is processed in less than 22 seconds (and this time can probably be significantly reduced by improved implementation of the algorithms). Smaller data sets take considerably less time. For example, when the same set of trials was run for 1000 points, the longest processing time was less than two seconds. The trial results shown in tables 6.1 and 6.2 and the results from other trials are consistent with the following hypothesised drivers of increases in the number of points ruled in:

- An increase in the number of possible terminals in a given space
- An increase in point weights (in the randomly generated cases, this translates to higher lower and upper limits on point weight)
6.9 Introduction to ruling out

When a point is ruled out, the number of possible terminals reduces by one, thereby halving the cardinality of the power set and consequently halving the number of computations needed to solve the PCEST problem using a naïve approach (Section 6.6).

In the sections below, a number of tests are presented, which, if successful, will definitively rule out a point as a terminal in a PCEST. The methods are presented in their intended order of implementation.

Let \( n_q \) denote some point in \( N_P \) with weight \( w_q \) that is to be subject to a ruling out test.

6.10 Pre-screening points to be subject to ruling out tests

If the clustering and ruling in algorithms have been applied (Section 6.8.2), then there may be one or more non-trivial clusters (clusters with two or more elements) of possible terminals, which were not ruled in. For non-trivial clusters with two or more elements, none of the elements can be successfully ruled out by the ruling out methods developed as part of this research. This is because each of the methods rely on \( w_q < d \), where \( d \) is the distance between the point \( n_q \) being tested and its nearest neighbour in \( N_I \cup N_P \). Accordingly, only possible terminals that are in trivial clusters need be subject to a ruling out test.

**Theorem 6.10.1.** If \( w_q \leq 0 \), then \( n_q \) can be ruled out.

**Proof.** Suppose a point \( n_q \) with weight \( w_q \leq 0 \) is a member of a maximum PCEST and consider these three cases:

*Case (1). \( n_q \) has one incident edge:* Then \( n_q \) can be removed and its incident edge, without decreasing the value of the PCEST giving a contradiction.

*Case (2). \( n_q \) has two incident edges:* Let \( v_j \) and \( v_k \) denote the vertices (Terminals or Steiner points) adjacent to \( n_q \). Observe that \( |v_jv_k| \leq |n_qv_k| + |n_qv_j| \) by the triangle inequality. Then \( n_q \) and its incident edges can be replaced with the single edge \((v_j, v_k)\) without reducing the value of the PCEST giving a contradiction.
Case (3). \( n_q \) has three incident edges: Let \( v_j, v_k \) and \( v_l \) denote the vertices (Terminals or Steiner points) adjacent to \( n_q \). Then \( n_q \) can be replaced with a Steiner point without decreasing the value of the PCEST, giving a contradiction.

6.11 Ruling out using an MST

Let \( T \) denote an MST on \( N_I \cup N_P \) with length denoted \( L_T \). For any degree-1 terminal in \( T \), \( n_q \in N_P \), let \( w_q \) denote the weight of \( n_q \) and let \( e_q \) denote the edge incident to \( n_q \) in \( T \) with length \( |e_q| \). Let \( \rho \) denote the Steiner ratio, which we recall is the smallest possible ratio between the length of an MStT and the length of the corresponding MST for any set of terminals.

Theorem 6.11.1. If \( n_q \in N_P \) is degree-1 in \( T \) and:

\[
 w_q \leq |e_q| - (1 - \rho)L_T \tag{6.1}
\]

then \( n_q \) can be ruled out as a terminal in a PCEST.

Proof. Let \( S \) denote a Steiner tree on \( N_I \cup N_P^* \), such that \( N_P^* \subseteq N_P \). Denote the length of \( S \) as \( L_S \). For some \( N_I \cup N_P^* \), \( S \) is a maximum PCEST.

Assume Inequality 6.1 is true and consider these cases:

Case (1). \( \{n_q\} = N_P \): It is easy to see that the terminals of a maximum PCEST are either \( N_I \) or \( N_I \cup \{n_q\} \). Suppose \( n_q \) is a terminal of \( S \). Then it follows from the definition of the Steiner ratio that:

\[
 \rho L_T \leq L_S \leq L_T \tag{6.2}
\]

Let \( T' \) and \( S' \) denote an MST and MStT on \( N_I \). Then:

\[
 \rho L_{T'} \leq L_{S'} \leq L_{T'} \tag{6.3}
\]

Since \( n_q \) is degree-1 in \( T \):

\[
 |e_q| = L_T - L_{T'} \tag{6.4}
\]
Substitute for $|e_q|$ from Equation 6.4 into Inequality 6.1:

$$w_q \leq \rho L_T - L_T'$$

(6.5)

Substitute for $\rho L_T$ from Inequality 6.2 and for $L_T'$ from Inequality 6.3 into Inequality 6.5:

$$w_q \leq L_S - L_S'$$

(6.6)

which implies that the Steiner tree for the PCEST is not $S$, but rather $S'$, and the lemma follows for this case.

Case (2). $\{n_q\} \subset N_P$: In this case, there are one or more possible terminals other than $n_q$ in $N_P$, some subset of which is an $N_p^*$ corresponding to a PCEST. It suffices to prove that if Inequality 6.1 holds, then the inequality only becomes stronger if any possible terminal other than $n_q$ is removed from $N_P$. Observe the effect on the right hand side of Inequality 6.1 of such a removal:

- $|e_q|$ remains the same or becomes larger.
- $L_T$ remains the same or becomes smaller, and since $\rho < 1$, the term $-(1 - \rho)L_T$ remains the same or becomes less negative.
- the number of terminals in the tree becomes strictly smaller.

Recall from Section 6.2 a lower bound for the Steiner ratio is 0.82416874, but for eight or fewer terminals there is a tighter bound: $\frac{\sqrt{3}}{2}$. Accordingly, as a general rule, $(1 - \rho)$ can be replaced in Inequality 6.1 with 0.17583126. However, if there are eight or fewer elements in $N_I \cup N_P$, $(1 - \frac{\sqrt{3}}{2})$ can be used, making Inequality 6.1 stronger.

6.11.1 Algorithm for ruling out using an MST

As part of this research, an algorithm to give effect to Theorem 6.11.1 was developed and programmed in ANSI C. Details of the program are included in Appendix C. The implementation included a modified version of Prim’s algorithm for computing MSTs,
though any of the MST algorithms would have been suitable.

**Remark 6.11.1.** If a degree-1 vertex and its incident edge are removed from an MST, the tree that results is an MST on the remaining vertices. This is because any sub-tree of an MST must itself be an MST, otherwise the sub-tree can be replaced with an MST, thereby shortening it, and contradicting the fact that the original MST was minimal. This can be used to advantage in an algorithm applying Theorem 6.11.1, since it means that only one MST needs to be generated by the classical way (i.e. Prim’s algorithm in this case). All subsequent MSTs required in the algorithm are obtained simply by removing degree-1 vertices and their incident edges in pairs.

In the interests of brevity in describing the algorithm (given in Algorithm 3), and since there are many well-known methods to generate MSTs, details of MST generation are omitted. Instead assume an MST has been generated on \( N \setminus N_O \) generating the following data:

- Denote the MST as \( T \) and its length as \( L_T \).
- For \( i : n_i \in N \setminus N_O \):
  - Let \( d_i \) denote the degree of point \( n_i \) in \( T \).
  - If \( d_i = 1 \): let \( g_i \) denote the index of the point adjacent to \( n_i \) and let \( e_i \) denote the length of the edge incident to \( n_i \).

Recall \( w_i \) denotes the weight of point \( n_i \)

The application of the algorithm using the C program can be illustrated with a small test model, illustrated in Figure 6.7 and with all point data included in Table 6.3.

Observe that points 0, 5 and 6 are degree-1 in the MST and that points 5 and 6 are possible terminals. The result was as follows:

1. Point 5 is tested for ruling out and the ruling out fails.
2. Point 6 is tested for ruling out and the ruling out succeeds.
3. Point 5 is tested for ruling out and the ruling out succeeds.
4. After ruling out points 5 and 6, points 3 and 4 are degree-1.
5. Point 3 is tested for ruling out and the ruling out fails.
6. Point 4 is tested for ruling out and the ruling out fails.
7. No further ruling out is possible.
Algorithm 3: Rule out using an MST

Input
\(\rho\)

For each \(i : n_i \in N \land (\text{cat}_i = 1 \lor \text{cat}_i = 2)\):

- \(w_i\) the weight of \(n_i\)
- \(\text{cat}_i\) the category of \(n_i\)

Variables generated by an MST function for all \(n_i \in N \land (\text{cat}_i = 1 \lor \text{cat}_i = 2)\):

- \(L_T\) the sum of the lengths of the edges in the MST

For \(i : n_i \in N \land (\text{cat}_i = 1 \lor \text{cat}_i = 2)\):

- \(d_i\) degree of point \(n_i\) in the MST
- \(g_i\) the index of the point adjacent to \(n_i\) if \(d_i = 1\)
- \(e_i\) the length of the longest edge incident to \(n_i\)

Algorithm

// Initialize the ruling flag for \(i : n_i \in N_P\) do
\(r_i = 0;\)

// Ruling test loop
\(f = 1;\)

while \(f = 1\) do

  for \(i : n_i \in N_P\) do

    if \((d_i = 1) \land (w_i < e_i - (1 - \rho)L_T)\) then

      // Ruling out

      \(r_i = 1;\)
      \(\text{cat}_i = 0;\)
      \(L_T = L_T - e_i;\)
      \(d_i = 0;\)
      \(d_{g_i} = d_{g_i} - 1;\)
      \(f = 1;\)

  end for

end while

Output

for each point \(n_i\) ruled out, \(r_i = 1\) and \(\text{cat}_i\) is changed from 1 to 0.
Further tests using the ANSI C program were conducted and details of two such tests are now discussed. The first test involved a randomly generated set of 100 points and the second test involved a modified version of this same set. The modification was to move one of the points a long distance away from the rest of the points (The point with index 99 was moved to coordinates $(5649, 5449)$). Tables 6.4 and 6.5 show details and the results. In the test using the unmodified randomly generated file, no points were ruled out. Any other result would have been surprising given the low probability of the random generation process yielding a point set containing one or more points with the requisite attributes. In the modified point set, the point with index 99 was ruled out, as expected.

Due to inefficiencies in the prototype implementation of the program, it was not possible to process more than about 100 points. It is very likely that this limit would be lifted with more time spent improving the efficiency of the implementation.

Generally speaking, in order for this ruling out method to be successful, the point set must include a closely grouped set of points, and one or a few points that are relatively far away from the group.

Figure 6.7: MST on points in the test model.
Table 6.3: Small test model details for ruling out using an MST

<table>
<thead>
<tr>
<th>i</th>
<th>( w_i )</th>
<th>( x_i )</th>
<th>( y_i )</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0</td>
<td>1.0</td>
<td>1.0</td>
<td>mandatory</td>
</tr>
<tr>
<td>1</td>
<td>1.1</td>
<td>2.0</td>
<td>1.0</td>
<td>possible</td>
</tr>
<tr>
<td>2</td>
<td>1.1</td>
<td>2.0</td>
<td>2.0</td>
<td>possible</td>
</tr>
<tr>
<td>3</td>
<td>1.1</td>
<td>3.0</td>
<td>2.0</td>
<td>possible</td>
</tr>
<tr>
<td>4</td>
<td>1.1</td>
<td>2.0</td>
<td>3.0</td>
<td>possible</td>
</tr>
<tr>
<td>5</td>
<td>1.1</td>
<td>1.0</td>
<td>5.0</td>
<td>possible</td>
</tr>
<tr>
<td>6</td>
<td>3.5</td>
<td>8.0</td>
<td>5.0</td>
<td>possible</td>
</tr>
</tbody>
</table>

Table 6.4: Larger test model general settings for ruling out using an MST

<table>
<thead>
<tr>
<th>Variable</th>
<th>Setting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of points to generate</td>
<td>100</td>
</tr>
<tr>
<td>Spatial limits for points ( i )</td>
<td>( 0 \leq \text{point_x}[i] \leq 1000 )</td>
</tr>
<tr>
<td></td>
<td>( 0 \leq \text{point_y}[i] \leq 1000 )</td>
</tr>
<tr>
<td>Weight limits for points ( i )</td>
<td>( 0.1 \leq \text{point_value} \leq 250 )</td>
</tr>
<tr>
<td>Coordinates for mandatory terminal 0</td>
<td>( \text{point_x}[0] = 500 )</td>
</tr>
<tr>
<td></td>
<td>( \text{point_y}[0] = 500 )</td>
</tr>
<tr>
<td>Computer details</td>
<td>Intel Core i7-6600U CPU @ 2.60GHz - 2.70GHz</td>
</tr>
<tr>
<td></td>
<td>Windows 7 Enterprise operating system</td>
</tr>
</tbody>
</table>

Table 6.5: Larger test model results for ruling out using an MST

<table>
<thead>
<tr>
<th>Trial</th>
<th>Details</th>
<th>Number of points ruled out</th>
<th>Processing time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Unmodified data set</td>
<td>0</td>
<td>&lt; 1</td>
</tr>
<tr>
<td>2</td>
<td>Point 99 moved</td>
<td>1</td>
<td>&lt; 1</td>
</tr>
</tbody>
</table>
6.12 Ruling out using a convex hull

The aim in this section is to find conditions that allow ruling out of points that are on the outer boundary of the set of all ruled in and possible terminals. Unlike the MST ruling out test in Section 6.11, this test can be successful even if the point set does not include one or a few points that are relatively far away from the rest of the points, all of which are closely grouped. The section commences with an introduction to the useful concept of PCEST Steiner Hulls in subsection 6.12.1, followed by the ruling out method, and an algorithm for the method in subsections 6.12.2 and 6.12.3 respectively. Finally, section 6.12.4 includes the detailed derivations of equations required by the algorithm.

6.12.1 PCEST Steiner hulls

A Steiner hull for a set of terminals $N$ is a region that is known to contain $N$ and all Steiner points of any MSTTs for $N$. The value of a Steiner hull is that it can be used to confine the area in which Steiner points need to be computed. Accordingly, a smaller Steiner hull is better. Steiner hulls are defined in the literature (Brazil & Zachariasen [79] page 26) for MSTTs. The application of Steiner hulls to maximum PCESTs will now be discussed and for clarity, the terms MSTT Steiner Hulls and PCEST Steiner Hulls will be used. Three MSTT Steiner Hulls to consider are as follows:

1. The convex hull on terminals: The convex hull on the terminals of the MSTT is known to be an MSTT Steiner hull [84]. It contains all Steiner points and edges of an MSTT.

2. A polygon containing all terminals such that the vertices of the polygon are terminals and such that no external angle is less than $\frac{2\pi}{3}$: This is known to be an MSTT Steiner hull (A minimal area polygonal Steiner hull). This polygon is a subregion of the convex hull on terminals and it contains all Steiner points and edges of an MSTT. Winter [101] devised an efficient method for producing such a Steiner hull from a convex hull.

3. The complement of the union of all wedges satisfying the wedge property, which is

---

*An open wedge having angle $\frac{2\pi}{3}$ and containing no terminals [84]*
known to be a Steiner hull ([84], [79], page 26). This contains all Steiner points, but it does not contain all edges of an MStT. It is a subregion of the aforementioned convex polygon.

Any of the aforementioned MStT Steiner hulls can be used, but the smaller two are more useful, since they reduce the areas in which Steiner points must be computed.

Now consider some set of points $N = N_I \cup N_P$. The terminals of a maximum PCEST contain $N_I$ and some $\tilde{N} \subseteq N_P$.

**Lemma 6.12.1.** The convex hull on $N_I \cup N_P$ is a PCEST Steiner hull that contains the Steiner points and edges of all PCESTs.

**Proof.** Since the convex hull of any subset of $N_I \cup N_P$ lies in the convex hull of $N_I \cup N_P$ the result immediately follows. \hfill \Box

**Lemma 6.12.2.** The minimal area polygonal Steiner hull on $N_I \cup N_P$ is a PCEST Steiner hull that contains the Steiner points and edges of all PCESTs.

**Lemma 6.12.3.** The complement of the union of all wedges satisfying the wedge property for $N_I \cup N_P$ is a PCEST Steiner hull that contains the Steiner points, but not necessarily all the edges of all PCESTs.

The proofs for Lemmas 6.12.2 and 6.12.3 are similar to the proof for Lemma 6.12.1

### 6.12.2 The ruling out method

Let $\Gamma$ denote a disk centred on $n_q$ with radius $w_q$. For any point $z$ on the boundary of $\Gamma$, and let $W$ denote an open wedge (a translation of a convex cone) with vertex $z$ and angle $\frac{2\pi}{3}$, such that the boundary rays of $W$ each make an angle of $\frac{2\pi}{3}$ with $n_qz$. Refer to 6.8 for illustrations of $n_q$, $z$, $\Gamma$ and $W$.

**Theorem 6.12.4.** If there exists a point $z$ on the boundary of $\Gamma$ such that $W$ contains $N_I \cup N_P \setminus \{n_q\}$ then $n_q$ can be ruled out as a terminal in a maximum PCEST.

**Proof.** Suppose that $W$ contains $N_I \cup N_P \setminus \{n_q\}$ and that contrary to the theorem, $n_q$ is a member of a maximum PCEST $P$. Observe that any Steiner point outside $W$ must
Figure 6.8: Artefacts relevant to Theorem 6.12.4.
connect to at least two vertices outside \( W \). Since there is only one possible terminal \( n_q \) outside \( W \) (and no ruled in terminals outside \( W \)), then a Steiner point outside \( W \) must connect to another Steiner point outside \( W \), which must connect to another Steiner point outside \( W \), and so on. We can conclude that if \( W \) contains \( N_I \cup N_P \setminus \{n_q\} \) then all Steiner points are inside \( W \).

The point \( n_q \) cannot be degree-3 in \( P \) since that requires it to connect to vertices with edges that make \( \frac{2\pi}{3} \) angles to each other, an impossibility when all other terminals in \( P \) are in \( W \). Similarly, \( n_q \) cannot be degree-2 in \( P \), as this would require incident edges making an angle \( \geq \frac{2\pi}{3} \). The remaining possibility is that \( n_q \) has degree-1. If so, its incident edge must have length greater than \( w_q \) since it must connect to a vertex inside \( W \). Then \( n_q \) and its incident edge can be removed from \( P \), increasing the value of the tree and contradicting the maximality of \( P \).

Observe that a possible terminal \( n_q \) that is capable of being ruled out by Theorem 6.12.4 has these characteristics:

- \( n_q \) must be a vertex on the convex hull for \( N_I \cup N_P \) which contains the Steiner points and edges of all PCESTs.
- The internal angle of the convex hull for \( N_I \cup N_P \) at vertex \( n_q \) must be less than \( \frac{2\pi}{3} \).

These observations can be used to implement an efficient test for Theorem 6.12.4. Refer to Figure 6.9 with artefacts and labels of interest in this case:

- Possible terminal \( n_q \) is depicted with the simplified label \( q \).
- The convex hull or \( N_I \cup N_P \), for which it is a vertex, is shown in purple (only a part of the convex hull is shown).
- Point \( v \) is the convex hull vertex immediately anticlockwise from \( q \). Vertex \( v \) corresponds to some ruled in or possible terminal other than \( q \).
- Point \( z \) is uniquely defined as the third point in \( \triangle qvz \) such that \( \angle qzv = \frac{2\pi}{3} \) and \( |qz| = w_q \).
- The supporting Line 1 passes through \( v \) and \( z \).

---

7As a basic property of an MStT, each pair of incident edges for a terminal must meet at an angle of \( \frac{2\pi}{3} \) or greater (Theorem 1.2 in [79]).

8For details refer to Section 6.12.1

9In this thesis the convention with respect to angles given in the form \( \angle abc \) is that they can be positive or negative and are measured in an anti-clockwise direction.
6.12 Ruling out using a convex hull

Figure 6.9: Artefacts relevant to Algorithm 4 and to the calculation of an equation for supporting Line 2 (Subsection 6.12.4).

- The supporting Line 2 passes through point $z$ and is rotated $\frac{\pi}{3}$ anticlockwise from $qz$.

The equation for supporting Line 2 is given in subsection 6.12.4 below.

6.12.3 Algorithm for ruling out using a convex hull

It is easy to see that satisfying the requirements in Theorem 6.12.4 is equivalent to the following pair of conditions being true:

- $q$ is a hull vertex, with an internal angle $< \frac{2\pi}{3}$, and;
- $q$ falls on the opposite side of Line 2 to all members of $N_I \cup N_P \setminus q$.

This leads to a simple ruling out test shown in Algorithm 4. A requirement for the algorithm is the generation of a convex hull for $N_I \cup N_P$. An efficient bespoke method was implemented in the aforementioned C program, the details of which are unimportant since there exist several methods to generate convex hulls (for example, the Jarvis march [102] and Graham’s scan [103]). For this reason, details of convex hull generation are omitted from the description of the algorithm.
Algorithm 4: Ruling out using a convex hull

Input
For each $i : n_i \in N \wedge (\text{cat}_i = 1 \lor \text{cat}_i = 2)$:
- $x_i, y_i$ coordinates for $n_i$
- $w_i$ the weight of $n_i$
- $\text{cat}_i$ the category of $n_i$

Algorithm
// Ruling test loop
$f = 1$;
while $f = 1$ do
    $f = 0$;
    Calculate a convex hull for all ruled in and possible terminals and identify the hull vertices;
    for each hull vertex do
        Calculate the internal angle for the hull vertex; (see Remark 6.12.1)
        if the internal angle is $< \frac{2\pi}{3}$ and the hull vertex is a possible terminal then
            Calculate Line 2 for the hull vertex;
            Check if the hull vertex is on the opposite side of Line 2 to all other ruled in and possible terminals;
            if the hull vertex is on the opposite side of supporting Line 2 then
                Recategorise the hull vertex as ruled out;
                $f = 1$;
        end
    end
end

Output
Points that are ruled out now have the category ruled out, whereas previously their category was possible.
Remark 6.12.1. Once a hull vertex has been ruled out during an iteration of Algorithm\[4\] then in the next iteration, it is only necessary to recalculate internal angles for vertices of the convex hull that are new, or were adjacent to the vertex that has now been ruled out.

6.12.4 Supporting Line 2

This subsection presents an equation for supporting Line 2. In the interests of clarity, it is presented as a set of related equations, along with the known information upon which the calculations are based. Refer to Figure\[6.9\] for an illustration of the important artefacts:

- $q$, $v$, and $z$ are defined as in the previous section.
- Let $(x_q, y_q)$ and $(x_v, y_v)$ be the coordinates of $q$ and $v$ respectively.
- Let $b$ denote a point on a horizontal line passing through $v$ and to the right of $v$.

The series of related equations follows.

An equation for $|qv|$ in terms of $x_q, y_q, z_v$ and $y_v$ (all inputs):

$$|qv| = \sqrt{(x_q - x_v)^2 + (y_q - y_v)^2} \quad (6.7)$$

An equation for $|vz|$, in terms of $|qz|$ (an input) and $|qv|$ (Equation\[6.7\]) follows from the law of cosines:

$$|vz| = \frac{-|qz| + \sqrt{4|qv|^2 - 3|qz|^2}}{2} \quad (6.8)$$

An equation for $\angle vqz$ in terms of $|qz|$ (an input), $|qv|$ and $|vz|$ (Equations\[6.7\] and \[6.8\]) follows from the application of the law of sines:

$$\angle vqz = \arcsin \left( \frac{\sqrt{3}|vz|}{2|qv|} \right) \quad (6.9)$$

An equation for $\angle bvq$ in terms of $|qv|$ (Equation\[6.7\]), $y_q$ and $y_v$ (inputs):

$$\angle bvq = \arcsin \left( \frac{y_q - y_v}{|qv|} \right) \quad (6.10)$$

An equation for $\angle zvq$ in terms of $\angle vqz$ (Equation\[6.9\]) and $\angle qzv$ (an input):
\[ \angle zvq = \pi - \frac{2\pi}{3} - \angle vqz = \frac{\pi}{3} - \angle vqz \quad (6.11) \]

An equation for \( \angle bvz \) in terms of \( \angle bvq \) (Equation 6.10) and \( \angle zvq \) (Equation 6.11)

\[ \angle bvz = \angle bvq - \angle zvq \quad (6.12) \]

An equation for \( x_z \) in terms of \( x_v \) (an input), \( |vz| \) (Equation 6.8), \( \angle bvq \) (Equation 6.10) and \( \angle bvz \) (Equation 6.12)

\[ x_z = x_v + |vz| \cos(\angle bvq - \angle bvz) \quad (6.13) \]

An equation for \( y_z \) in terms of \( y_v \) (an input), \( |vz| \) (Equation 6.8), \( \angle bvq \) (Equation 6.10) and \( \angle bvz \) (Equation 6.12):

\[ y_z = y_v + |vz| \sin(\angle bvq - \angle bvz) \quad (6.14) \]

Finally, an equation for supporting Line 2 in terms of \( \angle bvz \) (Equation 6.12), \( x_z \) (Equation 6.13) and \( y_z \) (Equation 6.14):

\[ y = \tan \left( \angle bvz - \frac{\pi}{3} \right) (x - x_z)x + y_z \quad (6.15) \]

### 6.13 Ruling out based on local connections

Sections 6.13 to 6.15 focus on some point \( n_q \in N_P \) with weight \( w_q \) and the question as to whether it can be ruled out as a terminal in a PCEST, based on what is known about \( n_q \)'s possible local connections and some easily determinable facts about all the ruled in and possible terminals.

To commence, a couple of important properties of MStTs are established. The properties are placed here because of their general application and will be referred to later in the thesis.

Let \( S \) denote an MStT with vertices \( V(S) = T(S) \cup St(S) \) where \( T(S) \) are the terminals
of \( S \) and \( St(S) \) are the Steiner points of \( S \). Consider a connected sub-tree \( S' \) of \( S \) such that the vertices of \( S' \) are \( V(S') = T'(S) \cup S'(S) \cup X(S) \cup X^*(S) \) where:

- \( T'(S) \subset T(S) \) are terminals in \( S' \) that are also terminals in \( S \).
- \( St'(S) \subseteq St(S) \) are Steiner points in \( S' \).
- \( X(S) \) is a set of points on the interior of edges of \( S \) that are leaves of \( S' \).
- \( X^*(S) \subseteq St(S) \) are degree-2 terminals in \( S' \). An element \( v_1 \in X^*(S) \) arises if a terminal, a Steiner point \( v_1 \) and an edge between the two are in \( S \), but the edge and the terminal are not in \( S' \).

**Lemma 6.13.1.** If \( S \) is an MStT on \( T(S) \), then every sub-tree \( S' \) of \( S \) is an MStT on some terminal set \( T'(S) \cup X(S) \cup X^*(S) \).

**Proof.** Suppose to the contrary there exists an MStT \( S'' \) spanning \( T'(S) \cup X(S) \cup X^*(S) \) such that \( L_{S''} < L_{S'} \). Then the vertices and edges of \( S' \) in \( S \) can be replaced with the vertices and edges of \( S'' \) giving a shorter tree and contradicting the minimality of \( S \). \( \square \)

The next lemma concerns any MStT \( S \) that contains at least one pair of adjacent Steiner points. Consider a sub-tree \( S' \) of such a Steiner tree \( S \) where \( V(S') = \{x_1, s_1, s_2, v_2\} \) as illustrated in Figure 6.10 such that:

- \( s_1 \) and \( s_2 \) are Steiner points that are adjacent in \( S \).
- \( v_1 \) is a vertex (either Terminal or Steiner point) that is adjacent to \( s_1 \).
- \( v_2 \) is a vertex (either Terminal or Steiner point) that is adjacent to \( s_2 \) such that \( v_2 \) is on the same side of \( s_1s_2 \) as \( v_2 \) and \( |s_2v_2| \leq |s_1v_1| \).
- \( x_1 \) is a point on edge \( (v_1, s_1) \) of \( S \) such that \( |x_1s_1| = |v_2s_2| \)

**Remark 6.13.1.** A sub-tree such as \( S' \) described above can be found in any Steiner tree with a pair of adjacent Steiner points.

**Lemma 6.13.2.** \( |s_1s_2| \geq \frac{\sqrt{3} - 1}{2} |s_2v_2| \)

**Proof.** The length of \( S' \) is:

\[
L_{S'} = |s_1s_2| + 2|s_2v_2| \tag{6.16}
\]
Figure 6.10: Sub-tree $S'$ of $S$. The edges of $S'$ are shown as solid lines.

Now consider an alternate tree $S''$ where $V(S'') = \{x_1, s_1, s_2, v_2, s_3, s_4\}$ as shown in Figure 6.11 in green.

The length of $S''$ is:

$$L_{S''} = \sqrt{3}|s_1s_2| + \sqrt{3}|s_2v_2|$$  \hspace{1cm} (6.17)

The result in Equation 6.17 can be understood easily with the construction of a Simpson line $e_1e_2$ as illustrated in Figure 6.12 with the derivation for the length of the Simpson line given in Equation 6.18. Refer to Section 6.2.1 for more details about Simpson lines.

$$|e_1e_2| = (|e_1a|) + (|ab|) + (|be_2|)$$
6.13 Ruling out based on local connections

\[ = \left( |s_1 s_2| \cos \frac{\pi}{6} \right) + \left( |s_2 v_2| \cos \frac{\pi}{6} \right) + \left( |s_1 s_2| + 2|s_2 v_2| \sin \frac{\pi}{6} \right) \]

\[ = \sqrt{3}|s_1 s_2| + \sqrt{3}|s_2 v_2| \quad (6.18) \]

If \( S \) is an MStT, then by Lemma 6.13.1 so is \( S' \) and this implies:

\[ L_{S''} \geq L_{S'} \quad (6.19) \]

Substituting for \( L_{S''} \) and \( L_{S'} \) from Equations 6.17 and 6.16 respectively into inequality 6.19:

\[ \sqrt{3}|s_1 s_2| + \sqrt{3}|s_2 v_2| \geq |s_1 s_2| + 2|s_2 v_2| \implies |s_1 s_2| \geq \frac{2 - \sqrt{3}}{\sqrt{3} - 1}|s_2 v_2| \quad (6.20) \]

Simplify Inequality 6.20 (multiply numerator and denominator by the conjugate of the denominator):

\[ |s_1 s_2| \geq \frac{\sqrt{3} - 1}{2}|s_2 v_2| > 0.366|s_2 v_2| \quad (6.21) \]

In the subsections below some networks embedded in the plane are described and constructed in sequence. Each network is an MStT (denoted with \( S \)), some other tree (\( T \)) or a forest (\( F \)). The sequence of construction is usually given by superscripts, so \( S^1 \) is followed by \( S^2 \), which is followed by \( T^3 \) etc. In order to decide whether some point \( n_q \in N_P \) can be ruled out as a terminal in a PCEST, without calculating MStTs or PCESTs, we investigate the possible local connections of \( n_q \). Accordingly, some networks within the boundary of a disk \( \mathbb{D} \) centred on \( n_q \) (defined in detail in Subsection 6.13.2) are also described and these are denoted with \( S \) or \( U \) for an MStT, \( T \) for some tree and \( F \) for a forest (all defined in detail below).

Throughout the rest of this chapter, let \( P = (V(P), E(P)) \) denote a PCEST on a set \( N \) of points, where \( T(P) \subseteq V(P) \) denotes the terminals of \( P \) and \( T(P) \subseteq N \) such that the PCEST value is maximised. \( P \) is a PCEST on \( N \).

Remark 6.13.2. A PCEST \( P \) on \( N \) is an MStT on \( T(P) \).
6.13.1 Steiner tree $S^1$

**Definition 6.13.1** ($S^1, L_{S^1}$). A tree $S^1 = (V(S^1), E(S^1))$ is an MStT where $V(S^1) = T(S^1) \cup St(S^1)$ for some $T(S^1) \subseteq N$. Let $L_{S^1}$ denote the sum of the lengths of the edges in $S^1$.

It follows from Remark 6.13.2 that for some $T(S^1) \subset N$, $S^1$ is a PCEST on $N$.

We wish to determine whether $n_q \in T(S^1)$ is consistent with $S^1$ being a PCEST. For example: if $n_q$ can be removed from $S^1$ and if some edges are replaced; and if the resulting network is a tree with length at least $w_q$ (i.e. the weight of point $n_q$) shorter than $S^1$, then $S^1$ is not a PCEST. It would then follow that $n_q$ can be ruled out as a terminal in a PCEST on $N$. 

Figure 6.11: Alternate tree $S''$ shown in green.
6.13 Ruling out based on local connections

6.13.2 Disk $D$ and Rubin points

Let $D$ denote a disk centred on $n_q$ with a radius $r$, where $r + \mu$ is equal to the distance between $n_q$ and its nearest neighbour in $N_I \cup N_P$ and $\mu$ is an infinitesimal. The only terminal inside $D$ is $n_q$. If $n_q \in T(S^1)$, then there must be at least one edge passing through the boundary of $D$ in $S^1$. There can be Steiner points inside $D$. An example is shown in Figure 6.13. In this example, there are four edges passing through the boundary of $D$ including one connecting $n_q$ to its nearest neighbour through a Steiner point. For convenience in the sequel, assume that $r = 1$ and all other distances are measured on this scale.
Figure 6.13: Example of an MStT $S^1$ showing the disk $\mathbb{D}$ centred on $n_q$. Note that the separation between the nearest neighbour to $n_q$ and disk $\mathbb{D}$ is exaggerated to highlight that the point is outside of $\mathbb{D}$.

Recall FST abbreviates for full Steiner tree (see page 67).

Remark 6.13.3. Since $n_q$ is the only terminal of $S^1$ inside $\mathbb{D}$, there can be no complete FST of $S^1$ inside $\mathbb{D}$.

Let $R^* = S^1 \cap \delta(\mathbb{D})$ (where $S^1$ is treated as an embedded network in the plane). $R^*$ is a finite set of points. Observe that each element of $R^*$ is a point on an edge of $S^1$. Let $R \subseteq R^*$ be a set of points $r_i$, such that each corresponding edge of $S^1$ has exactly one end point inside $\mathbb{D}$. The elements of $R$ will be referred to as Rubin points in the sequel\textsuperscript{10}. For convenience assume that some Rubin point is labelled $r_1$ and all other Rubin points are labelled $r_2, r_3, \ldots r_m$ according to their anti-clockwise order on the boundary of $\mathbb{D}$.

\textsuperscript{10}Named in honour of Professor Hyam Rubinstein, who suggested this approach.
Accordingly, \( m \) denotes the cardinality of \( R \).

### 6.13.3 Steiner Tree \( S^2 \) and \( S \)

**Definition 6.13.2** \((S^2, L_{S^2})\). A tree \( S^2 \) results from inserting Rubin points \( R \) into some \( S^1 \) as new terminals. Let \( L_{S^2} \) denote the sum of the lengths of the edges in \( S^2 \).

Refer to Figure 6.14 for an example of an \( S^2 \).

**Remark 6.13.4.** \( S^2 \) is an MST and \( L_{S^2} = L_{S^1} \).

**Remark 6.13.5.** By the construction of \( S^2 \), every Rubin point in \( S^2 \) has degree-2.
Definition 6.13.3 ($S, L_S$). $S$ is defined as the connected component of $S^2 \cap D$ containing $n_q$ the Rubin points, all the Steiner points inside $D$, and all the edges connecting these vertices inside $D$ (Figure 6.21). The sum of the lengths of all the edges of $S$ is denoted $L_S$.

Remark 6.13.6. $S$ is a Steiner tree on $R \cup \{n_q\}$.

Remark 6.13.7. By the construction of $S$, all Rubin points in $S$ have degree-1.

Lemma 6.13.3. There can be no full Steiner sub-trees in $S$ in which the only terminals are Rubin points.

Proof. Suppose to the contrary that there is a full Steiner sub-tree in $S$ in which the only terminals are Rubin points. Denote this set of Rubin points $R^*$. Note that $n_q$ is also a terminal in $S$ and accordingly there must be a path in $S$ from $n_q$ to a terminal in $R^*$ contradicting Remark 6.13.7.

Corollary 6.13.4. It follows from Lemma 6.13.3 that every full Steiner sub-tree in $S$ has $n_q$ as one of its terminals. It further follows that the number of full Steiner sub-trees in $S$ is equal to the degrees of $n_q$ in $S$.

Definition 6.13.4 (Bridge in $S$). A bridge in $S$ is a convex path (a path in which every turn is anticlockwise or every turn is clockwise) between two Rubin points in the same FST of $S$ that are adjacent on the boundary of $D$ (Figure 6.15).

Remark 6.13.8. A cherry on two Rubin points in $S$ is a bridge in $S$.

Remark 6.13.9. A path that is not convex includes both clockwise and anti-clockwise turns.

Definition 6.13.5 (Non-bridge in $S$). A non-bridge in $S$ is a path between two Rubin points that are adjacent on the boundary of $D$ such that the path contains both clockwise and anti-clockwise turns, or such that the two Rubin points are contained in different FSTs of $S$ (Figure 6.16).

Lemma 6.13.5. In the case that $n_q$ has one incident edge in $S$, there is exactly one non-Bridge in $S$.

\[11\] Defined on page 67
**Figure 6.15:** Example of a bridge between \( r_i \) and \( r_j \). In this example, there are three Steiner points in the path.

**Proof.** Refer to Figure 6.16. The tree \( S \) partitions the interior of \( D \) into regions, each of which has exactly one arc of the boundary of \( D \) as part of its boundary (If a region has no arcs on its boundary then \( S \) contains a cycle, while if a region has more than one arc on its boundary then \( S \) is disconnected). Each region has Rubin points at the end of the region's arc. It is easy to see that for every region other than the one containing \( n_q \), the unique path through \( S \) between the two Rubin points is convex, comprising two or more straight line segments with interior angles of \( \frac{2\pi}{3} \). The exception is the region containing the point \( n_q \). The point \( n_q \) has degree-1 in \( S \), and \( S \) is full, \( n_q \) is adjacent to a Steiner point that is in the unique path in \( S \) between the two Rubin points for the region. This path has an interior angle of \( \frac{4\pi}{3} \), and accordingly, the region is not convex (and therefore the path is not a bridge). Since \( n_q \) is unique, so is the non-bridge. \( \square \)

**Remark 6.13.10.** No vertex is farther than 1 from \( n_q \) and no edge in \( S \) incident to \( n_q \) can be longer than 1 since any such edge would have an end outside of \( D \).

**Lemma 6.13.6.** There can be no edge in \( S \) with length greater than 1.
Prize collecting Euclidean Steiner trees

Figure 6.16: The path from \( r_1 \) to \( r_4 \) is a non-bridge in \( S \). In the case that \( n_q \) has degree-1 in \( S \) there is only one non-bridge in \( S \).

Proof. Suppose \( S \) has an edge with length greater than 1 and suppose this edge is removed breaking \( S \) into two components, one of which must include \( n_q \). Since no vertex can be farther than 1.0 from \( n_q \), the components can be reconnected with an edge of length at most 1.0, giving a tree with length shorter than \( S \) contradicting the minimality of \( S \). 

Lemma 6.13.7. The distance between Rubin points corresponding to the ends of a bridge in \( S \) is less than \( \sqrt{3} \).

Proof. Let \( r_1 \) and \( r_2 \) be the two end points of a bridge in \( S \), and suppose \( |r_1r_2| \geq \sqrt{3} \). There are four cases to consider:

Case (1). One Steiner point in the path: Since \( n_q \) has degree-1, \( s \) cannot coincide with \( n_q \), and hence \( \max(|r_1s|,|r_2s|) > 1 \) contradicting Lemma 6.13.6 (see Figure 6.17).

Case (2). Two Steiner points \( \{s_1,s_2\} \) in the path: See Figure 6.18. The vertex \( n_q \) must be on the opposite side of the edge \( (s_1,s_2) \in E(S) \) to vertices \( r_1 \) and \( r_2 \), otherwise this is not a bridge. Observe that \( (s_1,s_2) \) the edge achieves minimum length if it is parallel to a line through \( r_1 \) and \( r_2 \). Hence \( |r_1r_2| \geq \sqrt{3} \) implies that \( |s_1s_2| > \frac{2}{\sqrt{3}} \) (by basic geometry).
contradicting Lemma 6.13.6

Case (3). Three Steiner points \( \{s_1, s_2, s_3\} \) in the path: Since \( r_1 s_1 \parallel r_2 s_2 \), then \( \max(|s_1 s_2|, |s_3 s_2|) > 1 \) contradicting Lemma 6.13.6 (See Figure 6.19).

Case (4). Four or more Steiner points in the path: Refer to Figure 6.20 for artefacts relevant to this case. Since \( |r_1 r_2| \geq \sqrt{3} \), the tangents of \( \mathbb{D} \) through \( r_1 \) and \( r_2 \) meet at an angle less than or equal to \( \frac{2\pi}{3} \). Observe \( s_1 r_1 || s_3 s_4 \) and \( s_1 r_1 || s_3 s_4 \). Now consider rays extending through \( s_1 r_1 \) and \( s_4 r_2 \) and observe that they meet at an angle of \( \frac{2\pi}{3} \). We conclude that \((r_1, s_1)\) or \((r_2, s_4)\) is outside the boundary of \( \mathbb{D} \), contradicting the definition of \( S \). 

6.13.4 Chord length definitions

Throughout this thesis, the term chord is used exclusively in relation to the straight line segment between Rubin points that are adjacent on the boundary of \( \mathbb{D} \).

Definition 6.13.6 (longest chord). A longest chord is a chord between adjacent Rubin points on the boundary of \( \mathbb{D} \) such that no other chord between adjacent Rubin points on the boundary of \( \mathbb{D} \) is longer.
Figure 6.18: Two Steiner points in the path (Lemma 6.13.7).

Figure 6.19: Three Steiner points in the path (Lemma 6.13.7).
Figure 6.20: Four Steiner points in the path (Lemma 6.13.7).
Definition 6.13.7 (shorter chord). A shorter chord is a chord between adjacent Rubin points on the boundary of $\mathbb{D}$ that is shorter than a longest chord.

6.13.5 Trees $T$, $U$ and $V$

Definition 6.13.8 ($T$, $L_T$). $T$ is a tree as follows: Its vertices are the Rubin points in $S^2$. Its edges are all the chords, other than a longest chord (see Figure 6.21). Denote the length of $T$ as $L_T$.

![Figure 6.21: MStT $S$ (black) on $R \cup \{n_q\}$ and MST $T$ on $R$ (red) (Constructed from the example $S^2$ in Figure 6.13).](image)

Lemma 6.13.8. $T$ is an MST for $R$.

Proof. Let $T_0$ be an MST for $R$ and suppose that there is no $T$ as defined above such that
$T = T_0$. Note that since $T_0$ is a tree with $m$ vertices, it must have exactly $m - 1$ edges. Then one of the following cases must apply to $T_0$.

**Case (1).** $T_0$ contains $m - 1$ edges corresponding to all chords other than a shorter chord: Then an edge in $T_0$ corresponding to a longest chord can be replaced with an edge corresponding to the shorter chord mentioned in the case condition to give a shorter tree, contradicting the minimality of $T_0$.

**Case (2).** $T_0$ contains $m - 1$ edges, at least one of which is not a chord\footnote{Recall a reference to a chord in this thesis always means a chord between Rubin points that are adjacent on the boundary of $D$.} Let $r_i$ and $r_k$ denote the end points of an edge of $T_0$ that is not a chord. Then there must exist a Rubin point $r_j$ on the open minor arc of $\delta(D)$ between $r_i$ and $r_k$ that is an end point of an edge in $T_0$ whose other end $q_0$ does not lie on that minor arc. The point $q_0$ must be (a) $r_i$; (b) $r_k$; or (c) some other Rubin point outside the aforementioned minor arc. We can immediately dismiss sub case c, since this implies crossed edges in $T_0$. In sub case a, $(r_j, r_k)$ cannot be an edge in $T_0$ (otherwise $T_0$ contains a cycle). Then by the triangle inequality, the edge $(r_i, r_k)$ can be replaced in $T_0$ by edge $(r_j, r_k)$ reducing the length of the tree and contradicting the minimality of $T_0$. A similar argument applies for sub case b. □

**Definition 6.13.9** (Gap in $T$). The **gap in $T$** is the chord that is not an edge of $T$.

**Definition 6.13.10** ($U$, $L_U$). $U$ is an MStT with terminals consisting of all the Rubin points in $R$. Denote the length of $U$ as $L_U$.

**Remark 6.13.11.** $L_U \leq L_T$.

**Remark 6.13.12.** $L_U < L_S$.

**Definition 6.13.11** ($V$, $L_V$). $V$ is a spanning tree that connects $R$, but is not necessarily minimal in length. Denote the length of $V$ as $L_V$.

**Remark 6.13.13.** $L_T \leq L_V$.

### 6.13.6 Replacement argument preliminaries

As a basic property of a Steiner tree, any vertex must have degree-1, 2 or 3, meaning it is incident to 1, 2 or 3 edges respectively. In the following sections, two replacement
arguments are developed, applicable to different degrees of \( n_q \) in \( S \) (Table 6.6). If the 
degrees of \( n_q \) in \( S \) cannot be predetermined for a given \( n_q \), it can only be ruled out if the 
conditions for both replacement arguments are met.

Table 6.6: Replacement arguments for ruling out on local connections

<table>
<thead>
<tr>
<th>Degrees for ( n_q ) in ( S )</th>
<th>Replacement argument</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A</td>
</tr>
<tr>
<td>2 or 3</td>
<td>B</td>
</tr>
</tbody>
</table>

### 6.14 Replacement argument A

This replacement argument applies for the case in which \( n_q \) has degree-1 in \( S \), if the 
Steiner tree \( S_1 \) from which it is derived is a candidate PCEST. It relies on a finding that 
for all \( R \), in the case that \( n_q \) has degree-1 in \( S \), the following holds:

\[
\inf(L_S - L_U) = \inf(L_S - L_T) = 2 - \sqrt{3}.
\]  

(6.22)

The claim is given formally in Theorem 7.5.1 and since the proof is lengthy, it is given 
in a separate chapter (Chapter 7: A universal constant for replacement argument A).

Replacement Argument A only applies to the degree-1 case, since it relies on \( \inf(L_S - L_U) \) having a strictly positive value. This is not true in cases where \( n_q \) has degree-2 or 3. 
Consider the following examples:

- Two evenly spaced Rubin points (Figure 6.22) giving \( L_S - L_U = 0 \) with \( n_q \) having 
degree-2.
- Three evenly spaced Rubin points (Figure 6.23) giving \( L_S - L_U = 0 \) with \( n_q \) having 
degree-3.

These cases in which \( n_q \) has degree-2 or 3, are examined in Section 6.15 Replacement 
argument B.

Let \( \inf(L_S - L_U) \) denote the greatest lower bound for \( L_S - L_U \) in the case that \( n_q \) has 
degree-1 in \( S \).
Figure 6.22: Two evenly spaced Rubin points (S (black) and U (red)). Note that U is shown as a curve only to distinguish it from S.

Figure 6.23: Three evenly spaced Rubin points (S (black) and U (red)). Note that the edges of U are shown as curves only to distinguish them from the edges of S.
Theorem 6.14.1 (Replacement argument A). In the case that \( n_q \) has degree-1 in \( S \), if \( w_q \leq \inf(L_S - L_U) \) then \( n_q \) can be ruled out as a terminal in a PCEST.

Proof. If \( w_q \leq \inf(L_S - L_U) \) then \( w_q - L_S \leq -L_U \). Replacing \( S \) with \( U \) does not decrease the value of the weighted tree, hence \( n_q \) can be ruled out. \( \square \)

Since Theorem 7.5.1 proves \( \inf(L_S - L_U) = \inf(L_S - L_T) = 2 - \sqrt{3} \), Theorem 6.14.1 can be restated as:

Corollary 6.14.2. If \( n_q \) has degree-1 in \( S \) and if \( w_q \leq (2 - \sqrt{3})r \) where \( r \) is the distance between \( n_q \) and its nearest neighbour in \( N_I \cup N_P \), then \( n_q \) can be ruled out as a terminal in a PCEST.

6.14.1 Algorithm to apply replacement argument A

Suppose it is possible to determine that a given possible terminal must have degree-1 if it is a member of a PCEST. Then replacement argument A can be used as a ruling out test.

The algorithm below introduces a test to find possible terminals that cannot be degree-2 or 3 in any MSt on \( N_I \cup \tilde{N}, \tilde{N} \subseteq N_P \) thus meeting the aforementioned requirement. The test involves the use of a minimal area polygonal PCEST Steiner hull\(^{13}\) (in the interests of brevity it will be referred to as the Steiner hull henceforth in this section). The Steiner hull contains all edges of all PCESTs (Lemma 6.12.2). It is a basic property of an MStT that the angle between any two edges meeting at a vertex is at least \( \frac{2\pi}{3} \). It follows that if the internal angle of the Steiner hull at a particular Steiner hull vertex is less than \( \frac{2\pi}{3} \), then that vertex cannot be degree-2 or 3 in any PCEST. Any such vertex that is also a possible terminal, is a candidate for a ruling out test with replacement argument A. This leads to a simple ruling out test shown in Algorithm 5. A requirement for the algorithm is the generation of the Steiner hull.

Other variables used in this algorithm:

- \( d_i \) denotes the distance between \( v_i \) and its nearest neighbour in \( N_I \cup N_P \).
- \( w_i \) denotes the weight of \( v_i \).
- \( f \) is a ruling flag, where \( f = 1 \) indicates a ruling has occurred in the loop.
- \( r_i \) is a ruling variable, where, \( r_i = 1 \) indicates that vertex \( v_i \) has been ruled out.

\(^{13}\)This term is defined in Section 6.12.1
Algorithm 5: Rule out points on a PCEST Steiner hull using replacement argument A

Inputs
For each $i : n_i \in N \land (cat_i = 1 \lor cat_i = 2)$:
- $x_i, y_i$ coordinates for $n_i$
- $w_i$ the weight of $n_i$
- $cat_i$ the category of $n_i$

Functions
- $d(i)$ returns $|n_i n_j|$ where $n_j$ is the nearest neighbour in $N_I \cup N_P$ to $n_i$.
- $a(i)$ returns the internal hull angle of $n_i$ (if it is a hull vertex).

Algorithm
$f = 1$
while $f = 1$ do
  $f = 0$
  Generate a minimal area polygonal Steiner hull on $N_I \cup N_P$ returning $v(i) = 1$
  for all points that are hull vertices.
  forall $i : v(i) = 1$ do
    if $cat_i = 1 \land a(i) < \frac{2\pi}{3} \land w_i \leq (2 - \sqrt{3})d(i)$ then
      / / Ruling out
      $r_i = 1$
      $f = 1$
      $cat_i = 0$
      (hence, $N_P = N_P \setminus \{n_i\}$)
  end
end

Outputs
$r_i = 1$ for each point $n_i$ ruled out.
6.14.2 Experimental results for Replacement Argument A

A version of the algorithm was programmed and tested in ANSI C (See Appendix C for more details). The version programmed used a convex hull rather than a minimal area polygonal Steiner hull to expedite the testing. A convex hull function is simpler than, and a precursor to, a minimal area polygonal Steiner function. It is however valid to use a convex hull since it too contains all the edges of all PCESTs (Lemma 6.12.1).

Table 6.7: Test model details for replacement argument A

<table>
<thead>
<tr>
<th>Model</th>
<th>A</th>
<th>B</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of points</td>
<td>10000</td>
<td>25</td>
</tr>
<tr>
<td>Lowest point weight</td>
<td>0.100</td>
<td>5.535</td>
</tr>
<tr>
<td>Highest point weight</td>
<td>1.000</td>
<td>86.570</td>
</tr>
<tr>
<td>Minimum X and Y coordinates</td>
<td>0.305, 0.027</td>
<td>141.364, 143.637</td>
</tr>
<tr>
<td>Maximum X and Y coordinates</td>
<td>10000.000, 99.997</td>
<td>1389.933, 639.430</td>
</tr>
<tr>
<td>X and Y coordinates for mandatory terminal</td>
<td>500, 50</td>
<td>569.667212, 385.043945</td>
</tr>
</tbody>
</table>

The results of two trials are shown in Tables 6.7 and 6.8. The longest processing time was less than a second. Observe that a relatively small number of possible terminals were ruled out, even in the case with 10000 points. The main reason is that the number of convex hull vertices that can satisfy the internal angle test is small. Even for large sets of points (e.g. 10000), the number of vertices that are eligible for testing with replacement argument A in any given pass will typically range from 0 - 4. The result would likely be improved with the use of a minimal area polygonal Steiner hull since it is not convex, and hence can have more vertices with small internal angles than a corresponding convex hull.

Table 6.8: Test model results for replacement argument A

<table>
<thead>
<tr>
<th>Model</th>
<th>Pass 1 rulings</th>
<th>Pass 2 rulings</th>
<th>Pass 3 rulings</th>
<th>Total rulings</th>
<th>Processing time (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>4</td>
<td></td>
<td>4</td>
<td>4</td>
<td>&lt; 1</td>
</tr>
<tr>
<td>B</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>5</td>
<td>&lt; 1</td>
</tr>
</tbody>
</table>
6.15 Replacement argument B

This replacement argument applies when \( n_q \) can have degree-2 or 3 in \( S \), if it is in a PCEST. Before presenting the main theorem, it is necessary to provide a number of definitions.

Definition 6.15.1 \((F, L_F)\). \( F \) is a forest on \( R \) with edges corresponding to all the chords other than two longest chords. \( L_F \) denotes the sum of the lengths of the edges of \( F \).

An example of an \( F \) forest is shown in Figure 6.24.

Figure 6.24: An example of an \( F \) forest (red).

Definition 6.15.2 \((F^3)\). \( F^3 \) is the forest that results from replacing \( S \) with \( F \) in \( S^2 \).

An example of an \( F^3 \) forest is shown in Figure 6.25. Observe:

- \( F \) and \( F^3 \) each have exactly two components.
- \( L_{F^3} = L_{S^2} - (L_S - L_F) \)

Definition 6.15.3 \((e, U_e)\). \( e \) is the length of a shortest edge between terminals that connects the two components of a forest \( F^3 \). For a given \((N, n_q)\), \( U_e \) is an upper bound for \( e \) for any \( F^3 \).
A method to calculate a $Ue$ for an $N$ is given in Section 6.15.1.

**Definition 6.15.4** $(T^4, L_{T^4})$. $T^4$ is the tree resulting from the addition of an edge that connects the two components of $F^3$ with length $e$ to $F^3$. $L_{T^4}$ denotes the sum of the lengths of all the edges in $T^4$.

Observe:
- $T^4$ is a tree that spans all the terminals that $S^2$ spans other than $n_q$.
- $L_{T^4} = L_{F^3} + e \leq L_{F^3} + Ue$

Replacement argument B relies on the claim that for any $R$, in the case that $n_q$ has degree-
2 or 3 in $\mathcal{S}$, then $L_\mathcal{S} - L_F \geq 2(2 - \sqrt{3}) \approx 0.536$. In the sequel this will be referred to as a lower bound $\mathcal{L}(L_\mathcal{S} - L_F) = 2(2 - \sqrt{3})$. The claim is proven in Theorem 8.3.1 in Chapter 8.

A universal constant for replacement argument $B$. In the same chapter, a much stronger conjecture is given, for the infimum $\inf(L_\mathcal{S} - L_F) = \sqrt{2}(\sqrt{3} - 1) \approx 1.035$ (Conjecture 8.4.1). The completion of a proof for this conjecture would considerably improve the efficacy of replacement argument $B$.

**Theorem 6.15.1** (Replacement argument $B$). In the case that $n_q$ has degree-2 or 3 in $\mathcal{S}$ and if:

$$w_q \leq \mathcal{L}(L_\mathcal{S} - L_F) - Ue$$

then $n_q$ can be ruled out as a terminal in a PCEST.

**Proof.** Suppose contrary to the Theorem, that Inequality 6.23 holds and $n_q$ is a member of a maximum PCEST. Such a PCEST meets the definition of an $S^1$ (Definition 6.13.1). Now consider a set of modifications that changes the $S^1$ into a $T^4$ (Definition 6.15.4) and the corresponding changes in tree length:

1. Insert Rubin points into the $S^1$ resulting in an $S^2$ (Definition 6.13.2). Observe the $S^2$ connects the same set of terminals as $S^1$ and:

$$L_{S^2} = L_{S^1}$$

2. Replace $\mathcal{S}$ with $\mathcal{F}$ in $S^2$ resulting in an $F^3$ forest (Definition 6.15.2). Observe $F^3$ has two components and:

$$L_{F^3} = L_{S^2} - (L_\mathcal{S} - L_\mathcal{F})$$

3. Add an edge with length $e$ to the $F^3$ forest, connecting its two components, resulting in a $T^4$ tree. Observe:

$$L_{T^4} \leq L_{F^3} + Ue$$

Now substitute for $L_{S^2}$ and $L_{F^3}$ from Equations 6.24 and 6.25 into Inequality 6.26:

$$L_{T^4} \leq L_{S^1} - (L_\mathcal{S} - L_\mathcal{F}) + Ue$$
Observe $T^4$ spans the same terminals as its corresponding $S^1$ spans other than $n_q$. If $S^1$ is a maximum PCEST, then:

$$w_q > L_{S^1} - L_{T^4} \quad (6.28)$$

Now substitute for $L_{T^4}$ from Inequality 6.27 into Inequality 6.28:

$$w_q > (L_S - L_F) - U_e \quad (6.29)$$

Inequalities 6.23 and 6.29 cannot simultaneously be true, disproving the contrary supposition.

### 6.15.1 A method to calculate an upper bound $U_e$ for a given $N$ and $n_q$

An edge with length $e$ is used to connect the two components of $F^3$ to give $T^4$. However for any given $N$ the terminals of $F^3$ are not known other than $T(F^3) = N_I \cup N'_P$ where $N'_P \subseteq N_P \setminus \{n_q\}$. Accordingly, an upper bound $U_e \geq e$ is sought.

The objective is to find a subset of $N_I \cup N'_P \setminus \{n_q\}$ such that all $N_I$ are included and such that the length of the longest edge in an MST on the subset is maximised. This length can be taken as $U_e$.

A naïve approach to finding a $U_e$ is to generate MSTs for every $N_I \cup N'_P$ where $N'_P \subset N_P \setminus \{n_q\}$ and to choose the longest edge among all of these. However, this requires the computation of $2^{(n-1)}$ MSTs, where $n$ is the number of possible terminals. Furthermore, $U_e$ calculated with this approach may be large and lead to inefficient ruling out.

Improvements to the naïve approach flow from the following observations:

- Consider an MST $MST_1$ and the length $e_1$ of its longest edge:
  - An MST $MST_2$ with longest edge length $e_2$ can be obtained by removing a degree-1 vertex and its incident edge from $MST_1$ (recall Remark 6.11.1). In this case $e_2 \leq e_1$.
  - An MST $MST_2$ with longest edge length $e_2$ can be obtained by removing a vertex with degree-2 or higher from $MST_1$, then using an algorithm such as Prim’s algorithm to generate a new MST. In this case $e_2 \geq e_1$. 
Now consider an MST on $N_1 \cup N_P \setminus \{n_q\}$ and suppose some of the possible terminals are degree-1 in the MST. When considering MSTs on $N_1 \cup N_P$, and with the objective of finding an upper bound $Ue$, only these degree-1 possible terminals need to be considered for inclusion in $N'_P$. Furthermore, if there are no degree-1 possible terminals in the aforementioned MST, then it suffices to consider just an MST on ruled in terminals for the determination of $Ue$.

- Recall from Section 6.8.1 that points can be clustered in such a way that if one member of the cluster is ruled in then all members of the cluster are ruled in. This implies that if any members of a cluster of possible terminals $N_P \setminus \{n_q\}$ are ruled out, they must all be ruled out. Let $C_P$ denote the set of clusters of possible terminals not including $n_q$. The implication of this observations in terms of improving the naïve approach to finding an upper bound $Ue$ is: Rather than generating an MST for every $N'_P \cup N_I$, it suffices to generate an MST for every $C'_P \cup N_I$, where $C'_P \subset C_P$. Combining this with the first observation: it suffices to consider for inclusion in $C'_P$ only clusters of possible terminals that include at least one degree-1 vertex in the MST on $N_1 \cup N_P \setminus \{n_q\}$.

### 6.16 Joint application of replacement arguments A and B

If, for a given possible terminal $n_q$, it cannot be predetermined if it would be degree-1, 2 or 3 in a PCEST, then it can be tested for ruling out by applying both replacement arguments A and B.

The test is as follows:

- If $w_q < 2 - \sqrt{3}$ (Replacement argument A), and;
- if $w_q \leq L(L_S - L_F) - Ue$ (Replacement argument B),

then $n_q$ can be ruled out as a terminal in a PCEST.

### 6.16.1 Illustration of the application of replacement arguments A and B

Refer to Figure 6.26 for an example of a case in which the combined application of replacement arguments A and B is successful in ruling out $n_q$. 
Observe $n_q$’s nearest neighbour is $n_1$ and all distances and the node weight $w_q$ are normalised, so that $|n_qn_1| = 1$. This example has been constructed in such a way that the ruling out methods other than replacement arguments A and B (Theorems 6.10.1, 6.11.1 and 6.12.4) are all incapable of ruling out $n_q$.

**Application of replacement argument A**

With the application of Theorem 7.5.1 and Corollary 6.14.2 it is easy to see that $n_q$ cannot be degree-1 in a PCEST since $w_q = 0.1 \leq 2 - \sqrt{3}$.

**Application of replacement argument B**

Attention turns first to the calculation of $U_e$. In Figure 6.26 the green points are ruled in terminals. The blue terminals on the right are possible terminals in a cluster $C_1$. The blue
6.17 Full Steiner tree generation for the PCEST problem

Terminals on the left are possible terminals that are not in clusters. Consider first an MST on \( N_I \cup N_P \setminus \{n_q\} \). Observe that for the possible terminals on the left, three have degree > 1 and can be ignored. It follows that \( Ue \) is the length of the longest edge in MSTs on the \( N_I \) in union with each element of the power set of \( \{C_1, \{n_2\}, \{n_3\}, \{n_4\}\} \). In this case it is easy to see that the longest edge in an MST is obtained for MSTs on \( N_I \cup \{n_3\} \) and also on \( N_I \cup C_1 \cup \{n_3\} \). It follows that \( Ue = 0.42 \).

Recall Inequality 6.23 \( (w_q \leq L(L_S - L_F) - Ue) \). Due to Theorem 8.3.1 in Chapter 8 we can substitute \( 2(2 - \sqrt{3}) \) for \( L(L_S - L_F) \) in Inequality 6.23. Also substitute 0.42 for \( Ue \) for this example giving:

\[
w_q \leq 0.11 < 2(2 - \sqrt{3}) - 0.42
\]

(6.30)

Since \( w_q = 0.1 \) Inequality 6.30 is true and it follows that \( n_q \) can be ruled out as a degree-2 or 3 terminal in a PCEST.

Finally, if Conjecture 8.4.1 is true, then a much stronger inequality applies in this case:

\[
 w_q \leq 0.61 < \sqrt{2}(\sqrt{3} - 1) - 0.42
\]

(6.31)

6.17 Full Steiner tree generation for the PCEST problem

This section describes a method to efficiently generate full Steiner trees (FSTs - see page 67) for the PCEST problem. The method is a modification to an existing method implemented for the Euclidean Steiner Tree (EST) problem in a software package called GeoSteiner [100].

In solving the EST problem, The Generation function in GeoSteiner efficiently enumerates full Steiner trees (FSTs) such that it is guaranteed that at least one MST can be constructed from a union of some of the FSTs. Efficient enumeration relies on the use of various pruning rules that can be used to discard FSTs or classes of FSTs that are not feasible as candidate components of an MST. The pruning rules used in GeoSteiner Generation in solving the EST problem rely on having a fixed set of terminals. The addition or subtraction of a terminal from this set will change the set of feasible FSTs, both adding
new FSTs and removing others. In solving the PCEST problem, the efficient enumeration of FSTs should proceed such that it is guaranteed that at least one maximum PCEST can be constructed from a union of some of the FSTs. A particular challenge is that there is not a fixed set of terminals, instead there is a set of ruled in terminals ($N_I$) and a set of possible terminals ($N_P$).

A naïve approach to generating a sufficient set of FSTs for PCEST is to run the original GeoSteiner Generation for $N_I \cup \tilde{N}$ for all $\tilde{N} \subseteq N_P$ and then to eliminate duplicate FSTs. However, this would be computationally burdensome since there are $2^n$ combinations for $\tilde{N}$ where $n$ equals the number of points in $N_P$.

PCEST Steiner hulls were previously described in Subsection 6.12.1. These are useful in PCEST Generation because they confine the area in which Steiner points need to be computed. PCEST Steiner hulls are derived from an established concept, MStT Steiner hulls. In subsections 6.17.1 and 6.17.2, two other mechanisms useful in PCEST Generation are derived from MStT counterparts.

### 6.17.1 PCEST bottleneck Steiner distance bound

With respect to the MStTs, the Bottleneck Steiner Distance (BSD) bound provides a useful way to eliminate some FSTs from consideration. The BSD as defined for MStTs cannot be used directly for the maximum PCEST, but modified forms are presented here.

Let $Z_T(n_i, n_j)$ denote the unique path between $n_i$ and $n_j$ in some MST $T$ for $N$.

**Definition 6.17.1 (MStT Bottleneck Steiner Distance (MStT BSD)).** The bottleneck Steiner distance BSD($n_i, n_j$) for two terminals $n_i$ and $n_j$ in $T$, is equal to the length of the longest edge $Z_T(n_i, n_j)$. (Brazil & Zachariasen [79] p.29)

**Lemma 6.17.1 (MStT BSD bound).** Given two terminals $n_i, n_j \in N$, let $Z_T(t_i, t_j)$ be the path between two terminals in some MStT $T$ for $N$. For any edge $e \in Z_T(n_i, n_j)$, $|e| \leq$ BSD($n_i, n_j$).

**Proof.** The proof is given in Brazil & Zachariasen [79] p.29.

We now consider the use of the BSD for solving the PCEST problem. During the Generation phase of GeoSteiner, when it would be useful to make use of the BSD, some
of the terminals in the solutions are known \((N_I)\), but it is not known which of the points in \(N_P\) are in the solution. Accordingly, an MST on the terminals in the solution for the purposes of calculating BSDs cannot be made. However, for any points \(n_i, n_j \in N_I\) it is sufficient to calculate a PCEST BSD on the basis of an MST on \(N_I\).

Consider the set of terminals \(\hat{N}\) for a PCEST, its corresponding MST \(\hat{T}\) and the bottleneck Steiner distance \(BSD(n_i, n_j)\) for \(n_i, n_j \in \hat{N}\).

**Lemma 6.17.2.** For \(n_i, n_j \in \hat{N}\), the addition of a point to \(\hat{N}\) will not increase \(BSD(n_i, n_j)\).

**Proof.** Let \(\hat{N}' := \hat{N} \cup \{n_k\}\). Denote the corresponding MST and bottleneck Steiner distances as \(\hat{T}'\) and \(BSD'(n_i, n_j)\) respectively. It will be proven that \(BSD(n_i, n_j) \geq BSD'(n_i, n_j)\). Suppose to the contrary that \(BSD(n_i, n_j) < BSD'(n_i, n_j)\). Then the longest edge \(e'\) in the path \(P_{\hat{T}'}(n_i, n_j)\) must be longer than the longest edge \(e\) in the path \(P_{\hat{T}}(n_i, n_j)\). If \(e'\) is removed from \(P_{\hat{T}'}(n_i, n_j)\), then two component networks remain, one of which contains \(n_i\) and the other containing \(n_j\). Denote these two components \(A\) and \(B\). Observe that \(\hat{T}\) must include an edge \(f\) that connects a member of \(A\) to a member of \(B\) and that \(|f| < |e'|\). It follows that if \(e'\) is replaced with \(f\) then a spanning tree on \(N'\) will have a shorter length, contradicting that \(\hat{T}'\) is an MST.

**Definition 6.17.2 (PCEST Bottleneck Steiner Distance (PBSD)).** The **PCEST bottleneck Steiner distance** \(PBSD(n_i, n_j)\) for points \(n_i, n_j \in N_I \cup N_P\), is equal to the length of the longest edge in \(Z_{\hat{T}}(n_i, n_j)\) for some MST of \(N_I \cup \{n_i, n_j\}\).

**Lemma 6.17.3 (PBSD bound).** Given two points \(n_i, n_j \in N_I \cup N_P\), let \(Z_P(n_i, n_j)\) be a path in some PCEST \(P\) for \(N\). For any edge \(e \in Z_P(n_i, n_j)\), \(|e| \leq PBSD(n_i, n_j)\).

### 6.17.2 PCEST lune and disk properties

The **Lune property** was established by Gilbert & Pollak \[84\] as a condition that must be satisfied by MSTTs for a given terminal set.

**Definition 6.17.3 (Lune).** Given a line segment \(\overline{bc}\), the **lune** \(L(b, c)\) is the intersection of open disks \(\Gamma(b)\) and \(\Gamma(b)\) centred at the points \(b\) and \(c\) respectively, each with radius \(|bc|\).
Table 6.9: Predicates defined for conditions of interest (Lune and disk properties).

<table>
<thead>
<tr>
<th>Predicate</th>
<th>Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Point $a$ is inside the lune $L(b, c)$</td>
</tr>
<tr>
<td>B</td>
<td>Point $b$ is inside the disk $\Gamma(a)$</td>
</tr>
<tr>
<td>C</td>
<td>Point $c$ is inside the disk $\Gamma(a)$</td>
</tr>
<tr>
<td>E</td>
<td>Edge $(b, c)$ passes through or into the disk $\Gamma(a)$</td>
</tr>
<tr>
<td>$A'$</td>
<td>$a$ is a vertex in a maximum PCEST</td>
</tr>
<tr>
<td>$B'$</td>
<td>$b$ is a vertex in maximum PCEST</td>
</tr>
<tr>
<td>$C'$</td>
<td>$c$ is a vertex in maximum PCEST</td>
</tr>
<tr>
<td>$E'$</td>
<td>$(b, c)$ is an edge in maximum PCEST</td>
</tr>
</tbody>
</table>

Lemma 6.17.4 (The Lune property). If $bc$ is an edge or a part of an edge of a MStT $S$, then the lune $L(b, c)$ does not contain any points of $S$.

Proof. The proof is given by Gilbert & Pollak [84].

The Lune property is utilized in a new lune and disk set of properties for solving the PCEST problem. Let $a$, $b$, and $c$ denote some points and let edge $(b, c)$ be a candidate edge for solving the PCEST problem. Let $L(b, c)$ denote the lune on $b$ and $c$ and let $\Gamma(a)$ denote a disk with its centre on $a$ and with radius equal to the weight $w_a$ of $a$. Without loss of generality, let the distance between $a$ and $b$ be less than or equal to the distance between $a$ and $c$. Point $a$ may be outside or inside the lune. Points $b$, $c$ may be outside or inside the disk. Edge $(b, c)$ may be outside the disk, or it may pass through or into the disk. $a$, $b$, and $c$ may be independently in or out of a maximum PCEST and edge $(b, c)$ may be in a maximum PCEST if both $a$ and $b$ are also in. Table 6.9 assigns predicate variables to these conditions. All combinations of these conditions are shown in Table 6.10. Observe that $B \lor C \rightarrow E$, meaning that some combinations of predicates (for example $B \land C \land \neg E$) are infeasible. Furthermore, some combinations are infeasible because the distance between $a$ and $b$ must be less than or equal to the distance between $a$ and $c$, as previously stated. The feasible combinations are illustrated in Figure 6.27.

The cases can be grouped on the basis of common clauses and one such grouping is shown in Table 6.11. Observe that each of the eight cases is assigned to exactly one group, and that no case could be assigned to any other group. The conclusion from this is that the groups are mutually exclusive and collectively exhaustive. Table 6.11 gives claims for
Table 6.10: All combinations of predicates (Lune and disk properties). The 8 feasible cases are illustrated in Figure 6.27.

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>E</th>
<th>Case</th>
</tr>
</thead>
<tbody>
<tr>
<td>¬A</td>
<td>¬B</td>
<td>¬C</td>
<td>¬E</td>
<td>1</td>
</tr>
<tr>
<td>¬A</td>
<td>¬B</td>
<td>¬C</td>
<td>E</td>
<td>2</td>
</tr>
<tr>
<td>¬A</td>
<td>¬B</td>
<td>C</td>
<td>¬E</td>
<td>Infeasible because B ∨ C ↔ E</td>
</tr>
<tr>
<td>¬A</td>
<td>¬B</td>
<td>C</td>
<td>E</td>
<td>Infeasible because</td>
</tr>
<tr>
<td>¬A</td>
<td>B</td>
<td>¬C</td>
<td>¬E</td>
<td>Infeasible because B ∨ C ↔ E</td>
</tr>
<tr>
<td>¬A</td>
<td>B</td>
<td>¬C</td>
<td>E</td>
<td>3</td>
</tr>
<tr>
<td>¬A</td>
<td>B</td>
<td>C</td>
<td>¬E</td>
<td>Infeasible because B ∨ C ↔ E</td>
</tr>
<tr>
<td>¬A</td>
<td>B</td>
<td>C</td>
<td>E</td>
<td>4</td>
</tr>
<tr>
<td>A</td>
<td>¬B</td>
<td>¬C</td>
<td>¬E</td>
<td>5</td>
</tr>
<tr>
<td>A</td>
<td>¬B</td>
<td>¬C</td>
<td>E</td>
<td>6</td>
</tr>
<tr>
<td>A</td>
<td>¬B</td>
<td>C</td>
<td>¬E</td>
<td>Infeasible because B ∨ C ↔ E</td>
</tr>
<tr>
<td>A</td>
<td>¬B</td>
<td>C</td>
<td>E</td>
<td>Infeasible because</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>¬C</td>
<td>¬E</td>
<td>Infeasible because B ∨ C ↔ E</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>¬C</td>
<td>E</td>
<td>7</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
<td>¬E</td>
<td>Infeasible because B ∨ C ↔ E</td>
</tr>
<tr>
<td>A</td>
<td>B</td>
<td>C</td>
<td>E</td>
<td>8</td>
</tr>
</tbody>
</table>

Figure 6.27: All feasible cases (Lune and disk properties).
Table 6.11: Claims (Lune and disk properties).

<table>
<thead>
<tr>
<th>Cases</th>
<th>Common Clauses</th>
<th>Claims</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,2</td>
<td>¬A ∧ ¬B ∧ ¬C</td>
<td>(No claim)</td>
</tr>
<tr>
<td>3</td>
<td>¬A ∧ B ∧ ¬C ∧ E</td>
<td>B → A</td>
</tr>
<tr>
<td>4</td>
<td>¬A ∧ B ∧ C ∧ E</td>
<td>B ∨ C → A'</td>
</tr>
<tr>
<td>5</td>
<td>A ∧ ¬B ∧ ¬C ∧ ¬E</td>
<td>A' → ¬E'; E' → ¬A'</td>
</tr>
<tr>
<td>6,7,8</td>
<td>A ∧ E</td>
<td>¬E'</td>
</tr>
</tbody>
</table>

each group, which are elaborated in Theorems 6.17.5 to 6.17.8.

**Theorem 6.17.5.** If point \( b \) is inside the disk \( \Gamma(a) \) (Case 3) and if \( b \) is in a maximum PCEST, then so is \( a \).

**Proof.** Suppose to the contrary \( a \) is not a member of a maximum PCEST. Then it can be connected to \( b \) with an edge \((a, b)\) which has a length less than the weight \( w_a \) of \( a \), increasing the value of the maximum PCEST and giving a contradiction. \( \square \)

**Theorem 6.17.6.** If points \( b \) and \( c \) are both inside the disk \( \Gamma(a) \) (Case 4), and if either \( b \) or \( c \) are in a maximum PCEST, then so is point \( a \).

**Theorem 6.17.7.** If point \( a \) is inside the lune \( \mathbb{L}(b, c) \) and \( b \) and \( c \) are outside the disk \( \Gamma(a) \) (Case 5), and if point \( a \) is in a maximum PCEST, then edge \((b, c)\) cannot be in the maximum PCEST. Similarly in Case 5, if the edge \((b, c)\) is in the maximum PCEST, then \( a \) cannot be.

**Proof.** The proof is essentially the same as that given by Gilbert & Pollak [84] for Lemma 6.17.4. \( \square \)

**Theorem 6.17.8.** If \( a \) is inside the lune \( \mathbb{L}(b, c) \) and if edge \((b, c)\) intersects the interior of disk \( \Gamma(a) \) (Cases 6,7,8), then \((b, c)\) is not an edge in a maximum PCEST.

**Proof.** Suppose to the contrary that \((b, c)\) is an edge in a maximum PCEST. Then a Steiner point \( s \) can be inserted anywhere on \((b, c)\) inside \( \Gamma(a) \) and a new edge \( \overline{ta} \) added, thereby increasing the value of the maximum PCEST and giving a contradiction. \( \square \)

The applications of Theorems 6.17.5 to 6.17.8 in Generation and Concatenation (Refer to Section 6.18) are shown in Table 6.12.
6.18 Concatenation for PCEST

The standard integer programming formulation for GeoSteiner Concatenation (MStT) is first described and then a revised formulation that seeks to solve the concatenation problem in the case of a PCEST is presented in \[6.18.2\] It is beyond the scope of this research to propose a detailed solution for this revised formulation. However, in the subsection \[6.18.3\] a transformation from the concatenation problem to a prize collecting Steiner tree in graphs (PCSTG) problem is given, and it is noted that there exists a solution to this latter problem.

6.18.1 Standard GeoSteiner concatenation formulation for MStT

This is a summary of the description of GeoSteiner concatenation for Steiner problems provided by Brazil & Zachariasen \[79\]. It is included for the convenience of the reader as an introduction to the next subsection, in which a revised concatenation formulation for the PCEST problem is proposed.

The approach is to treat each full component (generated in the generation stage) as a hyper edge on the set of terminals in a hypergraph, and to find the MST of the hypergraph.

Consider a hypergraph \( G = (V, E) \), with vertices \( v_i \in V \) and hyper edges \( e_j \subseteq V \) and \( e_j \in E \). Note that \( e_j \) is a subset of vertices, but lower case is used since edges in graphs are by convention denoted in lower case. A tree is denoted \( T = (V_T, E_T) \) such that \( V_T = \bigcup_{e_j \in E_T} e_j \). A tree is a spanning tree if \( V_T = V \). Let \( c_j \) denote the weight of hyper edge \( e_j \) which is equal to the length of the Steiner tree in the full component represented by \( e_j \). The decision variable \( x_j = 1 \) if \( e_j \in E_T \), otherwise \( x_j = 0 \). Let \( |e_j| \) and \(|V| \) denote

<table>
<thead>
<tr>
<th>Cases</th>
<th>Edge ((b,c)) in Generation</th>
<th>Edge ((b,c)) in a Concatenation trial</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,2</td>
<td>accepted</td>
<td>regardless of whether point ( a ) is in the trial</td>
</tr>
<tr>
<td>3,4</td>
<td>accepted</td>
<td>only if point ( a ) is in the trial</td>
</tr>
<tr>
<td>( 5 \land a \in N_I )</td>
<td>discarded</td>
<td>n/a</td>
</tr>
<tr>
<td>( 5 \land a \in N_P )</td>
<td>accepted</td>
<td>only if point ( a ) is not in the trial</td>
</tr>
<tr>
<td>( 6,7,8 )</td>
<td>discarded</td>
<td>n/a</td>
</tr>
</tbody>
</table>

---

Table 6.12: Applications of Theorems \[6.17.5\] to \[6.17.8\] (Lune and disk properties)
the cardinality of $e_j$ and $V$ respectively. The spanning tree formulation is as follows:

\[
\text{minimise } \sum_{j : e_j \in E} c_j x_j \tag{6.32}
\]

Subject to:

\[
\sum_{j : e_j \in E} (|e_j| - 1)x_j = |V| - 1 \tag{6.33}
\]

Let $W_k$ be some subset of vertices in $V$.

\[
\sum_{e_j \in E, e_j \cap W_k \neq \emptyset} (|e_j \cap W_k| - 1)x_j \leq |W_k| - 1, \; \emptyset \neq W_k \subset V \tag{6.34}
\]

Constraint 6.33 ensures that $T$ has the correct number of edges for a spanning tree on $V$. Constraint 6.34 ensures that $T$ does not contain any cycles.

A solution to this problem is available in the GeoSteiner software, based in the work of Warme [104].

### 6.18.2 Revised concatenation IP formulation for the PCEST problem

In this revised formulation, the full components are treated as hyper edges in a hypergraph as before. The revision involves adding decision variables for the inclusion of the terminals and weights for the terminals. The notation is the same as is used in Section 6.18.1 with the following additions: Let $w_i$ denote the weight of vertex $v_i$. Let the decision variable $y_i = 1$ if $v_i \in V_T$, otherwise $y_i = 0$. The formulation to find the point-weighted spanning tree is as follows:

\[
\text{maximise } \sum_{i : v_i \in V} w_i y_i - \sum_{j : e_j \in E} c_j x_j \tag{6.35}
\]

Subject to:

\[
\sum_{j : e_j \in E} (|e_j| - 1)x_j - \sum_{i : v_i \in V} y_i = -1 \tag{6.36}
\]
6.18 Concatenation for PCEST

\[ \sum_{j : e_j \in E, e_j \cap W_k \neq \emptyset} \left( |e_j \cap W_k| - 1 \right)x_j \leq |W_k| - 1, \ \emptyset \neq W_k \subset V \quad (6.37) \]

\[ \sum_{i : v_i \in e_j} y_i - x_j \geq 0, \ j : e_j \in E \quad (6.38) \]

Constraints 6.36 and 6.37 have the same functions as 6.33 and 6.34 respectively, modified to accommodate the selection of vertices in the optimisation. Constraint 6.38 ensures the spanning tree spans the selected vertices. That is, if \( y_i = 1 \), then there must exist an edge \( e_j \) such that \( v_i \in e_j \).

6.18.3 Transforming PCEST concatenation into a PCSTG problem

In this subsection a transformation from the PCEST problem into a PCSTG problem is given. The transformation allows the application of an existing efficient software solution. Some background to the PCSTG problem is included in Section 6.3. We focus here on Gamrath’s method: In 2017 Gamrath et. al. [99] published details of their solver “SCIP-Jack”. The solver is capable of solving a wide variety of Steiner tree problems in graphs, including PCSTG and the rooted variant thereof. The general approach used by Gamrath et. al. is to transform a PCSTG problem into an equivalent Steiner arborescence problem and to solve that problem. A PCSTG problem instance is an undirected graph \( G = (V, E) \) with edge weights \( c : E \to \mathbb{Q}_{\geq 0} \) and point weights \( p : V \to \mathbb{Q}_{\geq 0} \), with the latter representing penalties for not including a point in the solution. The problem is thus framed as a minimisation problem: to find a tree \( S = (V_S, E_S) \) such that:

\[ \text{minimise} \quad P(S) := \sum_{e \in E_S} c_e + \sum_{v \in V \setminus V_S} p_v \quad (6.39) \]

SCIP-Jack is reported to solve problems with 25,000 edges, 1,000 points (of which 500 have assigned penalties) in less than 0.4 seconds.

The transformation from PCEST concatenation to PCSTG is as follows for each full component:

- Each possible terminal \( n_i \in N_P \) is represented as a point \( v \in V \) with penalty \( p_v = \)]
• Each ruled in terminal (including the mandatory terminal) is represented as a point \( v \in V \) with penalty \( p_v \) set sufficiently high to ensure its inclusion in the solution.
• Each Steiner point \( s_i \) is represented as a point \( v \in V \) with penalty \( p_v = 0 \).
• Each edge \((v_i, v_j)\) is represented by an edge \( e \in E \) with weight \( c_e = |v_i v_j| \).

6.19 Solving the un-rooted PCEST problem

Recall Section 6.8 Ruling in applies to the rooted PCEST problem, whereas the various ruling out techniques, FST generation and concatenation described in Sections 6.9 to 6.18 apply to either rooted or the un-rooted PCEST problems. Ruling in is therefore the limiting factor in terms of the applicability of the full suite of techniques to the un-rooted problem. To overcome this shortcoming, it suffices to find the smallest possible set of points, one or more of which must be a member of a un-rooted PCEST. Then, the full set of techniques can be applied once each for the candidate points as the mandatory terminal and the result with the highest value is a un-rooted PCEST. There must be such a non-empty set since at worst it consists of all points with positive weights. The case of all points having zero or negative weights can be ignored since this would mean there are no possible terminals and the un-rooted PCEST is empty. A more satisfactory result is obtained if one or more points can be ruled in as terminals in a un-rooted PCEST. Then it suffices to apply the full set of techniques once with one of the identified points as the mandatory terminal. The result is a un-rooted PCEST.

Let \( N \) denote a set of possible terminals none of which is identified as a mandatory terminal. Let \( c_i \) denote the \( i \)th cluster resulting from the application of the merging algorithm (see Algorithm 1 on page 81) to \( N \). Let \( C \) denote the set of clusters. Now the ruling in algorithm (see Algorithm 2 on page 82) is utilised in a new way.

For each cluster \( c_j \in C \):

1. Select any one point in cluster \( c_j \). For the purposes only of the next step, change the selected point’s identification to \( n_0 \) (i.e. the mandatory terminal).
2. Run the ruling in algorithm (see Algorithm 2 on page 82). Denote the set of clus-
ters ruled in as $U_i$ (including the cluster containing $n_0$. Observe $U_i \subseteq C$. Note that each $U_i$ represented points ruled in to a rooted PCEST, with $n_0$ as the mandatory terminal.

Let $U$ denote the set of all $U_i$s.

Now the results of the aforementioned ruling in runs are analysed to see if a ruled in terminal for a un-rooted PCEST can be identified, or alternatively, find the smallest possible set of points, one or more of which must be a member of a un-rooted PCEST.

Let $Q_k$ denote a largest possible subset of $U$ such that there is at least one cluster that is common to all $U_i$ elements of $Q_k$. Let $|Q|$ denote the cardinality of $Q$. If $|Q| = 1$, then the common cluster to $Q_k$ contains points that are all in a un-rooted PCEST. If $|Q| > 1$, then each $Q_k$ contains a cluster $c_{Q_k}$ at least one of which must be in a un-rooted PCEST.

6.20 Discussion

This chapter and supporting Chapters 7 and 8 describe work at various technology readiness levels\footnote{Refer to Appendix B: Technology readiness levels}, summarised in Table 6.13. The development steps that are required to advance this work further are:

1. Prototype ruling out using a convex hull and replacement argument B. This requires a modest amount of work, which can make use of the existing architecture for other ruling prototypes.
2. Further develop the ruling in and ruling out prototype programs to ensure reliability and improve efficiency.
3. Prototype FST generation and concatenation functions. This work is best done inside the architecture of GeoSteiner. GeoSteiner source code is available under a Creative Commons Attribution 4.0 International Public License. The implementation of these functions also requires integration of the SCIP-Jack solver with GeoSteiner. Source code for SCIP is available under an academic license.
6.21 Conclusions

In this chapter, I have defined a new algorithmic framework for solving PCEST problems. The broad strategy is to find a solution to the rooted PCEST problem. Then, if a solution to the un-rooted PCEST problem is required, that can be obtained by judicious application of the rooted PCEST to a small set of cases. A key issue is that points provided in a problem instance for PCEST, are only possible terminals in the solution. The strategy for solving the rooted PCEST problem is to firstly reclassify possible terminals to ruled in or ruled out, subject to various ruling tests. Points that are ruled out are discarded. What remains is a set of ruled in terminals and a set of possible terminals. It is shown in this chapter that the algorithms designed to solve Euclidean Steiner tree problems, can be adapted to accommodate the more voluminous and complex data for the PCEST problem.

The clustering and ruling in algorithms have been been prototyped in ANSI C and has been shown to be efficient at processing large data sets. The clustering algorithm is principally designed to serve the ruling in algorithm, but also serves to pre-screen points to be subject to ruling out tests: Only possible terminals that are not in clusters need be subject to the ruling out tests in this chapter (the others cannot be ruled out by any method described in this chapter.

A total of four ruling out methods have been devised with most very easy to apply. Most of the methods have been prototypes in an ANSI C program. The most complex ruling out method is Ruling out on local connections, based on the two replacement argu-
ments A and B. These replacement arguments are relatively simple to apply, but their proofs are long and involved. In fact the proofs spill out into another two chapters of this thesis. Each proof requires the determination of a separate universal constant. In the case of replacement argument A, a very strong universal constant (an infimum) has been obtained. I achieved a weaker result for the universal constant for replacement argument B. There remains a conjecture for the infimum for replacement argument B. It is hoped that further research will be able to confirm this conjecture, which will strengthen the replacement argument.

The remaining topics covered in this chapter concern GeoSteiner, a software package that includes efficient algorithms to solve Euclidean Steiner tree problems. This chapter includes changes to the way in which GeoSteiner’s Generation and Concatenation functions operate. These changes have been theorised but not implemented. The implementation of these changes remains as a future development opportunity.
This page intentionally left blank.
Chapter 7
A universal constant for replacement argument A

The “universal constant” referred to in the title of this chapter is $2 - \sqrt{3}$. More specifically, it is the finding that for all $R$, if $n_q$ has degree-1 in $S$, then $\inf(L_S - L_U) = \inf(L_S - L_T) = 2 - \sqrt{3}$. This is formally given in Theorem 7.5.1 in Section 7.5. The supporting constructs and lemmas for Theorem 7.5.1 account for the rest of this chapter. The strategy is to search for all cases of $R^A$, where $R^A$ is an $R$ for which $L_S - L_U$ is minimal. The lemmas include some interesting findings, such as a requirement that topologies are linear for any $S$ for an $R^A$. The combined effects of the lemmas is to restrict the possibilities for $R^A$ to a single benchmark case.

7.1 Preliminaries

A number of abbreviations and symbols used in this chapter are common to Chapters 6 and 8. Readers may find the listings of commonly used symbols and abbreviations in Tables A.2 to A.4 of Appendix A useful.

Definition 7.1.1 (R-cherry). An R-cherry is a cherry in $S$ on two Rubin points.

Remark 7.1.1 (At least one R-cherry). If there are two or more Rubin points in an FST of an $S$ for $R \cup \{n_q\}$, then there must be at least one R-cherry in the FST and the Rubin points in the R-cherry must be adjacent on the boundary of $\mathbb{D}$. Clearly this is true for an FST that contains two Rubin points. For an FST that contains more than two Rubin points, then the FST has at least two disjoint cherries, at most one of which includes $n_q$.

Definition 7.1.2 ($R^A$). $R^A$ is the smallest set of Rubin points that satisfies the following conditions:
A universal constant for replacement argument A

• \( n_q \) has degree-1 in \( S \), and;
• no additions, removals or perturbations of Rubin points can reduce \( L_S - L_T \) without changing the degrees of \( n_q \) in \( S \).

Remark 7.1.2 (perturbations and derivatives). If for some perturbation of Rubin points in some \( R \), the derivative \( \dot{L}_S - \dot{L}_T \leq 0 \) then it follows that \( R \neq R^A \).

Recall from page 105 that \( m \) denotes the cardinality of \( R \).

In the sequel it will be shown that \( \inf(L_S - L_U) = 2 - \sqrt{3} \) (Theorem 7.5.1). The argument is lengthy and a summary is provided here for the benefit of the reader:

• If \( m = 1 \) it is easy to show that \( L_S - L_U = 1 \) (Lemma 7.2.1).
• If \( m = 2 \) in \( R^A \) then \( \inf(L_S - L_U) = 2 - \sqrt{3} \) (Lemma 7.2.3).
• If \( m > 2 \) then \( R \neq R^A \) for a variety of reasons including \( L_S - L_T > 2 - \sqrt{3} \) which implies \( L_S - L_U > 2 - \sqrt{3} \). (See Section 7.3).

The bound occurs in the limiting case for \( m = 2 \) and we conclude that \( \inf(L_S - L_U) = 2 - \sqrt{3} \).

### 7.2 One or two Rubin points in \( R \)

**Lemma 7.2.1.** In the case that \( n_q \) has degree-1 in \( S \) and if \( R \) has 1 Rubin point, then \( L_S - L_U = 1 \).

*Proof.* Observe there is one edge in \( S \) with length 1 and \( U \) has zero length. (Figure 7.1). \( \square \)

**Corollary 7.2.2.** If \( n_q \) has degree-1 in \( S \) and \( R \) has 1 Rubin point, then \( U = T \).

Lemma 7.2.3 provides a benchmark for \( L_S - L_U \) and in the sequel it will be proven that no other case can lead to a lower value for \( L_S - L_U \).

**Lemma 7.2.3.** If \( m = 2 \) in \( R^A \), then \( \inf(L_S - L_U) = 2 - \sqrt{3} \).

*Proof.* Refer to Figure 7.2. Denote the Rubin points as \( r_1 \) and \( r_2 \); the Steiner point as \( s \) and angle \( \phi = \angle r_1 n_q r_2 \).\(^1\)

Observations:

\(^1\)Recall that in this thesis the convention with respect to angles given in the form \( \angle abc \) is that they can be positive or negative and are measured in an anti-clockwise direction.
7.2 One or two Rubin points in $R$

Figure 7.1: If $S$ has 1 Rubin point, then $L_S - L_U = 1$.

Figure 7.2: $S$ (black) and $U$ (red) in the case that $R$ has two Rubin points.
Figure 7.3: $S$ (black) and $U$ (red) shown with the equilateral point $e$ and various construction points, angles, edges and line segments as described in the text.

- With two Rubin points, $U$ is a straight line segment between the two Rubin points.
- $S$ has one Steiner point $s$, and is degenerate outside domain $(\phi) = (0, \frac{2\pi}{3})$.
- $L_S - L_U$ can be parametrised with just $\phi$.

Figure 7.3 shows the use of an equilateral point $e$ and Simpson line $(n_q, e)$ to calculate the length of the Steiner tree as follows.

Commence with the calculation of $L_S$ from the Simpson line, where $a$ is the midpoint of $r_1r_2$:

$$L_S = |n_qe| = |n_qa| + |ae| \quad (7.1)$$

From basic trigonometry:

$$|n_qa| = \cos \frac{\phi}{2} \quad (7.2)$$

$$|ae| = |r_1e| \cos \frac{\pi}{6} = \frac{\sqrt{3}}{2} |r_1e| \quad (7.3)$$

Given $\triangle r_1r_2e$ is equilateral:
|r_1e| = |r_1r_2| = 2 \sin \frac{\phi}{2} \quad (7.4)

Substitute for |r_1e| from Equation 7.4 into Equation 7.3

|ae| = \sqrt{3} \sin \frac{\phi}{2} \quad (7.5)

Substitute from Equations 7.2 and 7.5 into 7.1 and apply Ptolemy’s identity for the sine sum formula:

\[ L_S = |n_qe| = \cos \frac{\phi}{2} + \sqrt{3} \sin \frac{\phi}{2} = 2 \left( \sin \frac{\pi}{6} \cos \frac{\phi}{2} + \cos \frac{\pi}{6} \sin \frac{\phi}{2} \right) = 2 \sin \left( \frac{\phi}{2} + \frac{\pi}{6} \right) \quad (7.6) \]

The length of \( U \) is given by:

\[ L_U = |r_1r_2| = 2 \sin \frac{\phi}{2} \quad (7.7) \]

The first derivative of \( L_S - L_U \) is used to find critical points. Applying the sine and cosine derivative rules and chain rule to Equations 7.6 and 7.7 gives:

\[ \dot{L}_S - \dot{L}_U = \frac{1}{2} \cos \left( \frac{\pi}{6} + \frac{\phi}{2} \right) - \frac{1}{2} \cos \frac{\phi}{2} \quad (7.8) \]

Interest turns to critical points, but since cosine is a decreasing function for the domain \((0, \pi)\) it follows that \( \frac{1}{2} \cos \left( \frac{\pi}{6} + \frac{\phi}{2} \right) - \frac{1}{2} \cos \frac{\phi}{2} \) is strictly negative for \( \phi \in (0, \pi) \). Accordingly, the minimum and maximum are to be found at the end points of the domain:

\[ \lim_{\phi \to 0} L_S - L_U = 1 \quad (7.9) \]

\[ \lim_{\phi \to 2\pi/3} L_S - L_U = 2 - \sqrt{3} \quad (7.10) \]

The lemma follows from Equation 7.10.
Remark 7.2.1. If \( n_q \) has degree-1 in \( S \) and \( R \) has 2 Rubin points, then \( U = T \) since the MST and minimum Steiner tree (MStT) on two points are the same.

7.3 Three or more Rubin points in \( R \)

This section commences with several remarks and lemmas to establish useful facts about cases in which there are three or more Rubin points in \( R \). Then, a minimum set of conditions required for \( R = R^A \) is established and simplest topologies that are minimally capable of meeting these conditions are defined. Next it is shown that these simplest topologies are incapable of being consistent with \( R^A \). Finally, attention turns to elaborations (one or more insertions of a Steiner point, edge and terminal as defined on page 157) on these simplest topologies and it is proven that no such elaborations are capable of being consistent with \( R^A \).

Remark 7.3.1 (Discarding candidates for \( R^A \)). By Lemma 7.2.3 any cases in which \( L_S - L_U > 2 - \sqrt{3} \) and \( m \leq 2 \) can be discarded as candidates for \( R^A \). In addition, since \( L_U \leq L_T \), any cases in which \( L_S - L_T > 2 - \sqrt{3} \) and \( m > 2 \) can be discarded as candidates for \( R^A \).

The proofs for Lemma 7.3.1 and for several subsequent lemmas in this section utilize a variational approach to geometric proofs developed by Rubinstein & Thomas for their paper on the Steiner ratio conjecture [105] and later refined for their paper on the Steiner ratio conjecture for co-circular points [106]. In [106] the objective is to show that no configuration \( Y \) of co-circular points corresponds to a Steiner ratio \( \rho < \frac{\sqrt{3}}{2} \) by studying the differential \( D\rho \). In the case at hand, co-circular points are also involved but rather than the Steiner ratio, it is the value of \( L_S - L_T \) that is of interest.

Lemma 7.3.1. If there are more than two Rubin points in \( R^A \), then there can be no fewer than two longest chords in \( R^A \).

Proof. Suppose to the contrary that there is exactly one longest chord in \( R^A \) and that \( R^A \) has more than two Rubin points. The gap in \( T \) must correspond to this longest chord. Given our contrary supposition, there must be a shorter chord adjacent to the gap in \( T \).
7.3 Three or more Rubin points in \( R \)

Label the Rubin points corresponding to such a shorter chord and the gap in \( T \) as \( r_1, r_2 \) and \( r_3 \) as illustrated in Figure 7.4. Let the line segment \( e \) be the Simpson line \( \tilde{S} \) (from an application of the Melzak-Hwang Algorithm to \( S \) - see Section 6.2.1). Refer to Figure 7.4 for other elements relevant to this case: Let \( a \) and \( b \) denote points on a tangent to \( D \) at \( r_2 \) that are as close as possible to \( r_1 \) and \( r_3 \) respectively. Let \( \theta_1 = \angle er_2a, \theta_2 = \angle r_1r_2a \) and \( \theta_3 = \angle br_2r_3 \).

Now suppose \( r_2 \) is perturbed on the boundary of \( D \) towards \( r_3 \). Recall from Lemma 6.2.2 that \( |er_2| = L_{\tilde{S}} \) and for any sufficiently small perturbation of \( r_2 \), the equality holds (Lemma 6.2.3). The chord \( r_2r_3 \) is a unique longest chord, and a sufficiently small perturbation will maintain it as a unique longest chord. It follows that the derivative of \( L_{\tilde{S}} - L_T \) with respect to this perturbation using a geometric approach is given by:

\[
\dot{L}_{\tilde{S}} - \dot{L}_T = |\dot{e}r_2| - \dot{L}_T = \cos \theta_1 - \cos \theta_2 \quad (7.11)
\]

Since \( \theta_2 < \theta_1 < \pi \) it follows that \( \dot{L}_{\tilde{S}} - \dot{L}_T < 0 \) giving a contradiction. If however \( R^A \) has
Lemma 7.3.2. If there are more than two Rubin points in \( R^A \), then the length of a longest chord is less than or equal to \( \sqrt{3} \).

Proof. Recall Lemma 6.13.7 (No bridge in \( S \) can span two Rubin points farther than \( \sqrt{3} \) apart). Now consider these cases:

Case (1). The gap in \( T \) corresponds to a bridge in \( S \). Since the gap in \( T \) must correspond to a longest chord, then the length of a longest chord is less than or equal to \( \sqrt{3} \).

Case (2). The gap in \( T \) corresponds to the non-bridge in \( S \). Then by Lemma 7.3.1, there must be another longest chord, and by Lemma 6.13.5, the other longest chord must correspond to a bridge in \( S \) and the \( \sqrt{3} \) length constraint applies.

Lemma 7.3.3. If an \( S \) for some \( R \cup \{n_q\} \) has an R-cherry that is adjacent to two shorter chords, then \( R \neq R^A \).

Proof. Note that implicit in the lemma is that \( m > 3 \). Consider an R-cherry in \( S \) on adjacent Rubin points \( r_2, r_3 \in R \) as depicted in Figure 7.5. The figure also shows a Simpson line \( \tilde{S} \) (from the application of Melzak-Hwang Algorithm to \( S \)) and elements of a choice of \( T \) for \( R \) of importance in this lemma. Note that only the edges of \( T \) that are affected by a perturbation described in the sequel are depicted in the figure. Recall from Lemma 6.2.2 that \( L_{\tilde{S}} = L_S \) and for any sufficiently small perturbation of terminals, the equality holds (Lemma 6.2.3).

Observe that there are edges of \( T \) either side of \( r_2 \) and \( r_3 \), consistent with being shorter chords as required in the lemma (i.e. all shorter chords correspond to edges in \( T \), consistent with the definition of \( T \)). The chord \( r_2r_3 \) could be a shorter chord or a longest chord. However, we can assume that \( r_2r_3 \) is an edge in some \( T \) since even if it does correspond to a longest chord, there must be at least one other longest chord (Lemma 7.3.1) which can correspond to the gap in a \( T \).

Now suppose contrary to the lemma that \( R = R^A \) and consider a perturbation of \( r_2 \) and \( r_3 \) towards each other on the boundary of \( D \) at equal speeds. Refer to Figure 7.5 for...
Figure 7.5: Illustration of the important elements of \( \hat{S} \) (black dotted line) and \( T \) (solid red lines) for Lemma 7.3.3.
elements of interest to this case. Let $b$ and $c$ denote points on a tangent to $D$ at $r_2$ that are as close as possible to $r_1$ and $r_3$ respectively. Let $d$ and $f$ denote points on a tangent to $D$ at $r_3$ that are as close as possible to $r_2$ and $r_4$ respectively. Observe $\angle cr_2r_3 = \angle r_2r_3d$ Let $\theta_1 = \angle r_1r_2b$, $\theta_2 = \angle cr_2r_3 = \angle r_2r_3d$ and $\theta_3 = \angle f r_3r_4$. Let $e_1$ and $e_2$ denote the equilateral points for $\tilde{S}$ (black dotted line in Figure 7.5) and let $\theta_4 = \angle n_qe_2e_1$. The derivative for $L_T$ using a geometric approach is:

$$L_T = \cos \theta_1 - 2 \cos \theta_2 + \cos \theta_3 \quad (7.12)$$

The following equations show the development of the derivative $L_S$ of $L_S$ using a geometric approach. Firstly note that $\triangle r_2e_2r_3$ is equilateral and remains so through the perturbation, so $|r_2r_3| = |r_2e_2| = |r_3e_2|$ and their corresponding derivatives for the perturbation are:

$$|r_2^\prime r_3| = |r_2^\prime e_2| = |r_3^\prime e_2| = -2 \cos \theta_2 \quad (7.13)$$

Let $a$ denote a point at the intersection of $r_2r_3$ and $n_qe_2$.

$$|e_2^\prime a| = |r_3^\prime e_2| \cos \frac{\pi}{6} = \frac{\sqrt{3}}{2} |r_3^\prime e_2| \quad (7.14)$$

Substitute for $|r_3^\prime e_2|$ from 7.13 into 7.14

$$|e_2^\prime a| = -\sqrt{3} \cos \theta_2 \quad (7.15)$$

Observe due to simple geometry that $\angle an_qr_3 = \theta_2$; $|n_qr_3| = 1$ and $|n_qa| = \cos \theta_2$. The derivative for $\cos \theta_2$ is $-\sin \theta_2$ and since perturbation decreases $\theta_2$:

$$|n_q^\prime a| = \sin \theta_2 \quad (7.16)$$

The derivative for $|n_qe_2|$ is equal to the sum of $|e_2^\prime a|$ (Equation 7.14) and $|n_q^\prime a|$ (Equation 7.16) and can be simplified by application of Ptolemy’s identity for the cosine sum formula:
\[ |n_\varphi e_2| = \sin \theta_2 - \sqrt{3} \cos \theta_2 \]
\[ = -2 \left( \cos \frac{\pi}{6} \cos \theta_2 - \sin \frac{\pi}{6} \sin \theta_2 \right) \]
\[ = -2 \cos \left( \theta_2 + \frac{\pi}{6} \right) \] (7.17)

The derivative for \( |e_1 e_2| \) follows:
\[ |e_1 e_2| = |n_\varphi e_2| \cos \theta_4 \] (7.18)

Since \( L_S = |e_1 e_2| \) and substituting for \( |n_\varphi e_2| \) from Equation 7.17 into Equation 7.18:
\[ \dot{L}_S = -2 \cos \left( \theta_2 + \frac{\pi}{6} \right) \cos \theta_4 \] (7.19)

The derivative for \( L_S - L_T \) follows from the application of the derivative sum rule to Equations 7.12 and 7.19:
\[ \dot{L}_S - \dot{L}_T = -2 \cos \left( \theta_2 + \frac{\pi}{6} \right) \cos \theta_4 - \cos \theta_1 + 2 \cos \theta_2 - \cos \theta_3 \]
\[ = 2 \cos \theta_2 - 2 \cos \left( \theta_2 + \frac{\pi}{6} \right) \cos \theta_4 - \cos \theta_1 - \cos \theta_3 \] (7.20)

A minimum for \( L_S - L_T \) must correspond to the condition \( \dot{L}_S - \dot{L}_T = 0 \) for a critical point. If an upper bound for \( \dot{L}_S - \dot{L}_T \) is less than zero over the defined domain, then there are no critical points for the domain. First choose values for \( \theta_1 \) and \( \theta_3 \) that will maximise \( \dot{L}_S - \dot{L}_T \) given that each angle must be less than \( \frac{\pi}{3} \) since \( 0 < |r_1 r_2| < \sqrt{3} \) and \( 0 < |r_3 r_4| < \sqrt{3} \):
\[ \dot{L}_S - \dot{L}_T < 2 \cos \theta_2 - 2 \cos \left( \theta_2 + \frac{\pi}{6} \right) \cos \theta_4 - 1 \] (7.21)

Observe \( |r_2 r_3| \leq \sqrt{3} \implies \theta_2 \leq \frac{\pi}{3} \) and as a consequence:
\[ -2 \cos \left( \theta_2 + \frac{\pi}{6} \right) \leq 0 \] (7.22)

The domain of \( \theta_4 \):

\[ -\frac{\pi}{6} < \theta_4 < \frac{\pi}{6} \implies 0 < \cos \theta_4 < \frac{\sqrt{3}}{2} \] (7.23)

When \( \theta_4 \rightarrow \frac{\pi}{6} \), \( \cos \theta_4 \) is minimised and consequently the term \( -2 \cos \left( \theta_2 + \frac{\pi}{6} \right) \cos \theta_4 \) is maximised. Accordingly, \( \theta_4 = \frac{\pi}{6} \) can be substituted into Equation 7.21 for a new upper bound:

\[
\hat{L}_S - \hat{L}_T < 2 \cos \theta_2 - 2 \cos \left( \theta_2 + \frac{\pi}{6} \right) \cos \frac{\pi}{6} - 1 \\
= 2 \cos \theta_2 - \sqrt{3} \cos \left( \theta_2 + \frac{\pi}{6} \right) - 1 \] (7.24)

Since \( 2 \cos \theta_2 - \sqrt{3} \cos \left( \theta_2 + \frac{\pi}{6} \right) \leq 1 \) for the domain \( 0 < \theta_2 \leq \frac{\pi}{3} \):

\[
\hat{L}_S - \hat{L}_T < 0 \] (7.25)

Inequality 7.25 contradicts that \( R = R_A \) and the lemma follows.

It follows that if \( R = R_A \) and \( m > 2 \) then any R-cherry in an \( S \) for \( R \cup \{n_q\} \) is adjacent to at least one longest chord. Refer to Figure 7.6 where \( r_3 r_4 \) is a longest chord adjacent to an R-cherry on \( \{r_2, r_3\} \). Figure 7.6 also shows artefacts such as angles of interest in the sequel.

**Lemma 7.3.4.** If there exists an \( R = R_A \) where \( m > 2 \) such that \( S \) for \( R \cup \{n_q\} \), has an R-cherry with a longest chord on at least one side, then \( \phi_1 + \phi_2 + \phi_3 > \pi \).

**Proof.** Refer to Figure 7.6 and consider the same perturbation as in Lemma 7.3.3. If \( r_1 r_2 \) was a longest or shorter chord before the perturbation, it remains respectively a longest or shorter chord after the perturbation. The chord \( r_3 r_4 \) is a longest chord before and after the perturbation. The chord \( r_2 r_3 \) is a shorter chord after the perturbation, whether or not it was a longest or shorter chord before the perturbation. We can assume that the gap in
Three or more Rubin points in $R$

Figure 7.6: Illustration of the important elements of $S$ and $T$ for Lemma 7.3.4

$T$ corresponds to $\{r_3, r_4\}$ before and after the perturbation since it is sufficient to prove the lemma with respect to any $S$ for $R \cup \{n_q\}$ and any $T$ for $R$. The derivative with respect to the perturbation is:

$$L_S - L_T = (\sin \theta_2 - \sqrt{3} \cos \theta_2) \cos \theta_4 - \cos \theta_1 + 2 \cos \theta_2$$  \hspace{1cm} (7.26)

Note that this is the same as the derivative given in Lemma 7.3.3 (Equation 7.20), except for the absence of the last term.

Observe $\sin \theta_2 - \sqrt{3} \cos \theta_2 = -2 \cos \left( \theta_2 + \frac{\pi}{6} \right) \leq 0$ for the domain $0 < \theta_2 \leq \frac{\pi}{3}$, corresponding to a maximum chord length of $\sqrt{3}$ given by Lemma 7.3.2. It follows that the first term in Equation 7.26 can be maximised by choosing the highest possible value for $\theta_4$ (thereby minimising $\cos \theta_4$).

Since $\theta_4 < \frac{\pi}{6}$, substitute $\cos \frac{\pi}{6} = \frac{\sqrt{3}}{2}$ into Equation 7.20 giving the following inequality:
\[ \dot{L}_S - \dot{L}_T < \frac{\sqrt{3}}{2} \sin \theta_2 - \frac{3}{2} \cos \theta_2 - \cos \theta_1 + 2 \cos \theta_2 = \frac{\sqrt{3}}{2} \sin \theta_2 + \frac{1}{2} \cos \theta_2 - \cos \theta_1 \]
\[ \implies \dot{L}_S - \dot{L}_T < \sin \left( \theta_2 + \frac{\pi}{6} \right) - \cos \theta_1 \quad (7.27) \]

Since \( \sin a = \cos \left( \frac{\pi}{2} - a \right) \):

\[ \dot{L}_S - \dot{L}_T < \cos \left( \frac{\pi}{3} - \theta_2 \right) - \cos \theta_1 \quad (7.28) \]

For critical points \( \dot{L}_S - \dot{L}_T = 0 \), giving a constraint that must apply if \( R = R^A \):

\[ \cos \theta_1 < \cos \left( \frac{\pi}{3} - \theta_2 \right) \implies \theta_1 + \theta_2 > \frac{\pi}{3} \quad (7.29) \]

Observe that \( \phi_1 = 2\theta_1 \) and \( \phi_2 = 2\theta_2 \). Substituting into Inequality 7.29 gives:

\[ \phi_1 + \phi_2 > \frac{2\pi}{3} \quad (7.30) \]

Since \( r_3r_4 \) is a longest chord, \( \phi_3 > \phi_1 \) and \( \phi_3 > \phi_2 \). It follows that \( \phi_3 > \frac{\pi}{3} \) and it further follows that:

\[ \phi_1 + \phi_2 + \phi_3 > \pi \quad (7.31) \]

\[ \square \]

7.4 Simplest Topologies for \( S \), Insertions and Elaborations

Definition 7.4.1 (Simplest topology for \( S \)). A simplest topology for \( S \) is a topology with the fewest Rubin points that can satisfy the following conditions:

1. \( n_q \) has degree-1 in \( S \).
2. More than two Rubin points.
3. No chord is longer than \( \sqrt{3} \) (Lemma 7.3.2).
4. Two or more longest chords (Lemma 7.3.1).
An implication of the first condition for a simplest topology is there is exactly one non-bridge in $S$, meaning that there is one pair of Rubin points that are adjacent on the boundary of $\mathbb{D}$, between which there is not a convex path (Lemma 6.13.5). This can be thought of as a path with anti-clockwise and clockwise $\frac{2\pi}{3}$ turns in which there is at least one turn in either direction. The difference between the number of anti-clockwise and clockwise turns is referred to as the *net turns*. If there are fewer turns in one direction than the other, then the one direction is called the *minority direction*. Exactly one net turn in the minority direction can be assumed since at least one is required to reach the terminal $n_q$ on the interior of $\mathbb{D}$ and any more than one implies more terminals on the interior of $\mathbb{D}$, which there cannot be given the construction of $S$.

**Lemma 7.4.1.** If there exists an $R$ such that $R = R^A$ and $m > 2$, there are two or more net turns in the path between Rubin points for the non-bridge in $S$ for $R$.

**Proof.** Denote the Rubin points for the non-bridge in $S$ as $r_1$ and $r_m$ and consider these two cases:

*Case (1). No net turns in the path:* If there are no net turns in the path from $r_1$ to $r_m$ then the path contains exactly 2 Steiner points as in (Figure 7.7). This implies that the two edges incident to $r_1$ and $r_m$ are parallel and that $r_1r_m$ is the unique longest chord in $S$ contradicting Lemma 7.3.1.

*Case (2). Exactly one net turn in the path:* Refer to Figure 7.8 for an example for this case. Find the intersection of rays from the pair of Rubin points $\{r_1, r_m\}$ in the non-bridge through their adjacent Steiner points. Observe $\angle r_mar_1 = \frac{2\pi}{3}$. Further observe that $n_q$ must be inside $\triangle r_mar_1$ otherwise $r_1r_m$ is a unique longest chord. That $n_q$ is inside $\triangle r_1ar_m$ implies that $\angle r_mn_qr_1 > \frac{2\pi}{3}$ which further implies $|r_1r_m| > \sqrt{3}$ contradicting Lemma 7.3.2.

**Corollary 7.4.2.** An $R$ such that $m > 2$ with fewer than two net turns in the path between Rubin points for the non-bridge in $S$ for $R$ either has an exclusive longest chord, or a chord with length $\sqrt{3}$.

Recall exactly one net turn in the minority direction can be assumed. Then as a result of Lemma 7.4.1 there must be a minimum of four turns in total. Each turn requires a
Figure 7.7: Zero net turns implies exactly one longest chord.

Figure 7.8: One net turn implies $|r_1r_m| > \sqrt{3}$. 
Steiner point, implying at least four Steiner points, and five Rubin points (six terminals in total including $n_q$ due to a full Steiner tree having exactly $m - 2$ Steiner points ([107], page 7)). An immediate implication of Lemma 7.4.1 and Corollary 7.4.2 is that in order to comply with the third and fourth conditions of the definition of a simplest topology for $S$ (see page 152), a simplest topology for $S$ must have least 5 Rubin points. Furthermore, since a simplest topology must have “the fewest Rubin points that can satisfy [the conditions in the definition]”, it must have exactly 5 Rubin points.

Attention is now turned to identifying all the simplest topologies, which can be parametrized by the directions of turns in the path between the pair of Rubin points for the non-bridge ($\{r_1, r_3\}$, see Figure 7.9). Let $L$ denote a left or anti-clockwise turn and let $R$ denote a Right or Clockwise turn in the path. Considering all possibilities for four turns, there are eight possibilities with a minority turn and a net turning number of two and they can be organised into two groups within which the topologies are symmetrical (Figure 7.9):

- **Group 1**: RLLL, LLRL, RRRL, LRRR
- **Group 2**: LLRL, LRLL, RLRR, RRLR

Observe:
• $n_q$ is in a cherry in each of the group-1 simplest topologies and not in a cherry in any group-2 simplest topology.

• Group-1 simplest topologies have one R-cherry and group-2 simplest topologies have two R-cherries.

Due to symmetry, it will be sufficient in the sequel to prove that $R \neq R^A$ for just one example topology from each of group-1 and group-2.

**Lemma 7.4.3.** Any topology $T$ for a full Steiner tree with $m \geq 3$ terminals can be created by inserting a Steiner point into the interior of an edge of some topology $T'$ for a full Steiner tree with $m - 1$ terminals, and then adding a new edge whose endpoints are that Steiner point and a new terminal.

**Proof.** This proof is based on a process described in the proof of a related lemma (Lemma 1.4: Number of full Steiner topologies) in [79]. It suffices to show that for any $T$ there exists a $T'$ to which the insertions described in the lemma can be made to yield $T$ and that $T'$ fulfills the requirements of a full Steiner topology. These requirements are that $T'$ is a tree, and then adding a new edge whose endpoints are that Steiner point and a new terminal. Also note that if $m > 2$ then no terminal is adjacent to another terminal. Starting with any $T$ suppose the reverse of the insertions described in the lemma are performed:

1. Remove any one terminal and its incident edge. Since $m > 2$, all terminals are adjacent to Steiner points. After the removal of one terminal and its incident edge a degree-2 vertex $v_0$ (i.e. the original Steiner point with one edge removed) remains. Denote the vertices at the other ends of the remaining two edges as $v_1$ and $v_2$.

2. Remove $v_0$ and replace its two incident edges with a single edge $(v_1, v_2)$.

The result ($T'$) is clearly a tree and since the degrees of $v_1$ and $v_2$ are not changed it follows that $T'$ is a Steiner topology.

In the interests of brevity and clarity, the following terms are defined:

**Definition 7.4.2** (Internal edge, external edge). An *internal edge* is an edge in $S$ between two Steiner points. An *external edge* is an edge incident to a Rubin point.
Each edge in S is either an internal edge, an external edge or the edge incident to \( n_q \).

**Definition 7.4.3 (Linear topology).** A **linear topology** is a topology for a full Steiner tree that contains exactly two cherries.

**Example 7.4.1 (Linear topology).** The trees in Figure 7.9 all have linear topologies.

**Definition 7.4.4 (Insertion, L insertion, R insertion, elaboration).** An **insertion** is the procedure described in Lemma 7.4.3 for creating a new Steiner tree with one extra Rubin point. An **L insertion** as an insertion such that the last turn in the path from \( n_q \) to the new Rubin point is anti-clockwise. An **R insertion** is an insertion in which the last turn is clockwise. An **elaboration** is an ordered set of one or more L or R insertions.

It follows from Lemma 7.4.3 that all full Steiner topologies where \( m > 2 \) are elaborations on the full Steiner tree for \( m = 2 \). It is easy to see that any topology that satisfies the three conditions given in the definition of a simplest topology above, is an elaboration on a group-1 or a group-2 simplest topology. Due to symmetry, it will be sufficient in the sequel to prove that \( R \neq R^A \) for all elaborations on just one example simplest topology from each of group-1 and group-2.

### 7.4.1 Internal R-Cherries

**Definition 7.4.5 (Internal R-cherry).** An **internal R-cherry** is an R-cherry that does not share a Rubin point with a non-bridge in S.

**Lemma 7.4.4.** If an S for an \( R \cup \{ n_q \} \) includes an internal R-cherry, then \( R \neq R^A \).

**Proof.** The proof is by contradiction in cases: Assume that an S for an \( R \cup \{ n_q \} \) includes an internal R-cherry and that \( R = R^A \) and show that this gives rise to contradictions in every case.

An internal R-Cherry implies \( m \geq 4 \): two Rubin points for the R-Cherry and one Rubin point either side of the R-Cherry as a minimum. Label this minimum set of Rubin points \( r_1, ..., r_4 \), where \( r_2 \) and \( r_3 \) correspond to the internal R-Cherry. If there is an internal R-cherry in S then there is a sub-tree of S with the structure shown in Figure 7.10 (or a mirror image of this structure) where:
• $v_1 = r_4$ or $v_1$ is a Steiner point on the path between $s_2$ and $r_4$.
• $v_2 = r_1$ or $v_2$ is a Steiner point on the path between $s_3$ and $r_1$.
• $v_3 = n_q$ or $v_3$ is a Steiner point on the path between $s_3$ and $n_q$.

Observe $v_1 s_2 || s_3 v_2$. For convenience, but without loss of generality, the picture is rotated such that both are parallel to the x-axis for the Euclidean plane (Figure 7.10). Let $x(p)$ and $y(p)$ denote the $x$ and $y$ coordinates of vertex with label $p$. For convenience, but without loss of generality let $x(n_q) = y(n_q) = 0$. Observe the path from $r_1$ to $r_2$ and from $r_3$ to $r_4$ must be convex since they are bridges and we conclude:

$$y(r_1) \geq y(v_2) \quad (7.32)$$

$$y(r_4) \geq y(v_1) \quad (7.33)$$

The remainder of this proof involves a number of cases and sub-cases which are listed.
in Table 7.1.

<table>
<thead>
<tr>
<th>Cases and sub-cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case (1). $v_3 = n_q$</td>
</tr>
<tr>
<td>Case (2). $v_3 \neq n_q$ (i.e. $v_3 = s_4$)</td>
</tr>
<tr>
<td>Sub-case (2.1). No Steiner points in the path from $s_4$ to $n_q$</td>
</tr>
<tr>
<td>Sub-case (2.2). Exactly one Steiner point, $s_5$ in path from $s_4$ to $n_q$</td>
</tr>
<tr>
<td>Sub-case (2.2.1). $x(v_1) \leq x(s_5)$</td>
</tr>
<tr>
<td>Sub-case (2.2.2). $x(v_1) &gt; x(s_5)$</td>
</tr>
<tr>
<td>Sub-case (2.3). Exactly two Steiner points, $s_5$ and $s_6$ in path from $s_4$ to $n_q$</td>
</tr>
<tr>
<td>Sub-case (2.3.1). $x(v_1) \leq x(s_6)$</td>
</tr>
<tr>
<td>Sub-case (2.3.2). $x(v_1) &gt; x(s_6)$</td>
</tr>
<tr>
<td>Sub-case (2.4). Three or more Steiner points in the path from $s_4$ to $n_q$</td>
</tr>
</tbody>
</table>

Case (1). $v_3 = n_q$
Recall $\phi_1 + \phi_2 + \phi_3 > \pi$ (Lemma 7.3.4). Clearly this is not true if $v_3 = n_q$ as it implies $y(r_1) > 0, y(r_4) > 0$ and therefore $\phi_1 + \phi_2 + \phi_3 < \pi$.

Case (2). $v_3 \neq n_q$ (i.e. $v_3 = s_4$)
There is either no other or there are some other Steiner point in the path between $s_4$ and $n_q$ to allow for $y(r_1)$ to be less than zero (That is, $y(s_3) < 0$ is required if $y(r_1) < 0$).

Case (2.1). No Steiner points in the path from $s_4$ to $n_q$
An edge incident to $s_4$ on the path to $n_q$ must be parallel to the x-axis implying that if $(n_q, s_4)$ is an edge in $S$, then $y(s_4) = y(n_q) = 0$. However, $y(s_4) = 0$ implies $y(r_1) > 0$ and since $y(r_4) > 0$ this contradicts the requirement that $\phi_1 + \phi_2 + \phi_3 > \pi$.

Case (2.2). Exactly one Steiner point, $s_5$ in path from $s_4$ to $n_q$
Refer to Figure 7.11 for this case. Denote the extra Steiner point $s_5$ and consider sub-cases 2.2.1 and 2.2.2.

Case (2.2.1). $x(v_1) \leq x(s_5)$
Let $a$ denote a point on $(v_1, s_2)$ such that $x(a) = x(s_5)$ (See Figure 7.12). Observe:

$$y(a) = y(v_1)$$

Further observe $|n_qa| > |n_qs_5|$ otherwise $(n_q, s_5)$ can be replaced in $S$ with a new edge.
Figure 7.11: Illustration for sub-case 2.2 (Lemma 7.4.4).

Figure 7.12: Illustration for sub-case 2.2.1 (Lemma 7.4.4).
(n_q, a) giving a shorter tree and contradicting the minimality of S. Accordingly, if S is minimal:

\[ |n_q a| > |n_q s_5| \implies y(a) > -y(s_5) \quad (7.35) \]

Note that there are easily stronger replacement arguments available, but this one is sufficient and leads to a simple proof.

Attention now turns to some geometric constraints. Firstly, since \( \mathcal{v}_1 \mathcal{v}_2 \) is horizontal and S cannot be self-intersecting:

\[ y(v_1) > 0 \implies y(a) > 0 \quad (7.36) \]

In order to comply with Lemma 7.3.4 \((\phi_1 + \phi_2 + \phi_3 > \pi)\), which is a requirement if \( R = R^A \), \( r_1 \) must be further below the x-axis than \( r_4 \) is above it. That is:

\[ y(r_4) < -y(r_1) \quad (7.37) \]

Since \( y(v_1) = y(a) < y(r_4) \) and \( y(s_3) = y(v_2) \leq y(r_1) \) this implies:

\[ y(a) < -y(s_3) \quad (7.38) \]

Observe due to basic geometry:

\[ y(s_5) < y(s_3) \implies -y(s_3) < -y(s_5) \quad (7.39) \]

Substituting for \( y(s_3) \) from Inequality 7.39 into Inequality 7.38 gives a condition that must apply if \( R = R^A \):

\[ y(a) < -y(s_5) \quad (7.40) \]

Observe that the minimality requirement for S in Inequality 7.35 contradicts Inequality 7.40 which must apply if \( R = R^A \).

Case 2.2.2. \( x(v_1) > x(s_5) \)

Refer to Figure 7.13 for this case. The case condition implies that \( v_1 \neq r_4 \). Let \( v_3 \) and \( v_5 \) denote the vertices incident to \( v_1 \) other than \( s_2 \). As for case 2.2.1, Inequality 7.40 applies
Figure 7.13: Illustration for sub-case 2.2.2 (Lemma 7.4.4).

if $R = R^A$. Given Inequality 7.40 and the case condition $x(v_1) > x(s_5)$ it is easy to see that the ray $\overrightarrow{v_1v_5}$ intersects $(n_q, s_5)$. This implies that depending on the length of $(v_1, v_5)$, either $(v_1, v_5)$ intersects $(n_q, s_5)$, or another edge on the path between $v_5$ and a Rubin point intersects $(n_q, s_5)$. Either way, $S$ is self-intersecting which contradicts the minimality of $S$.

Case (2.3). Exactly two Steiner points, $s_5$ and $s_6$ in path from $s_4$ to $n_q$

Denote the extra Steiner points $s_5$ and $s_6$ and consider sub-cases 2.3.1 and 2.3.2.

Case (2.3.1). $x(v_1) \leq x(s_6)$

Refer to Figure 7.14 for this case. The same argument applies as for sub-case 2.2.1 by substituting $s_6$ for $s_5$ in the Inequalities 7.35 (i.e. $y(a) > -y(s_6)$) and 7.40 (i.e. $y(a) < -y(s_6)$): The minimality requirement for $S$ in Inequality 7.35 contradicts Inequality 7.40 which must apply if $R = R^A$.

Case (2.3.2). $x(v_1) > x(s_6)$

Refer to Figure 7.15 for this case. As for case 2.3.1 $s_6$ can be substituted for $s_5$ in the Inequalities 7.40 (i.e. If $R = R^A$ then $y(a) < -y(s_6)$). Observe that $v_1$ cannot be closer to $n_q$ than $s_6$, otherwise $(n_q, s_6)$ can be replaced with a new shorter edge $(n_q, v_1)$ contradicting
the minimality of $S$. This can be represented by an exclusion disk centred on $n_q$ with radius $|n_qs_6|$: Vertex $v_1$ must not be inside this disk (Illustrated in Figure 7.15). It is easy to see in this case that at least one edge in the path between $v_1$ and a Rubin point must cross either edge $(s_4, s_5)$ or $(s_5, s_6)$ contradicting the minimality of $S$.

Case (2.4). Three or more Steiner points in the path from $s_4$ to $n_q$

Refer to Figure 7.16 for this case. Clearly an edge in the path from $s_7$ to a Rubin point must cross either $(s_2, v_1)$ or an edge in the path between $v_1$ and a Rubin point. This contradicts the minimality of $S$. 

\[\square\]
Figure 7.15: Illustration for sub-case 2.3.2 (Lemma 7.4.4).

Figure 7.16: Illustration for sub-case 2.4 (Lemma 7.4.4).
Lemma 7.4.5. If a topology for $S$ contains an internal R-cherry, every elaboration on the topology has an internal R-cherry.

Proof. Refer again to the sub-tree of $S$ with the structure shown in Figure 7.10 for an internal R-cherry. Observe:

- An insertion or series of insertions in the paths from $s_2$ to $r_1$ or $r_4$ results in a new bridge adjacent to the R-cherry on $\{r_2, r_3\}$ replacing the existing bridge.
- An insertion or series of insertions into $(s_2, s_1)$ results in a new bridge adjacent to the R-cherry on $\{r_2, r_3\}$ replacing the existing bridge.
- An insertion or series of insertions into $(s_1, r_2)$ or $(s_1, r_3)$ creates a new R-cherry with bridges either side.
- An insertion or series of insertions in the path from $s_3$ to $n_q$, or an insertion or series of insertions in any any edge of any branch from the the path from $s_3$ to $n_q$ make no difference to the R-cherry on $\{r_2, r_3\}$ or the bridges either side of it.

Remark 7.4.1. An $S$ without an internal R-cherry has a linear topology.

Corollary 7.4.6. If $R = R^A$ then an $S$ for $R$ has a linear topology.

7.4.2 Bridges

Definition 7.4.6 (intermediate bridge). An intermediate bridge is a bridge in $S$ that is not adjacent to the non-bridge in $S$.

Attention is drawn to certain geometric aspects of intermediate bridges in $S$ that can occur when $R = R^A$. From Corollary 7.4.6 it follows that only cases in which $m > 2$ and where the topology is linear need to be considered. That is, all intermediate bridges for $S$ have two Steiner points in the path between adjacent Rubin points. Refer to Figure 7.17 for an illustration of artefacts and their labels that are of interest in Lemma 7.4.7.

- Let $r_j, s_j, s_k$ and $r_k$ denote the vertices for an intermediate bridge of $S$.
- Let $v_0$ denote a vertex adjacent to $s_j$ that is on the path(s) from $s_j$ to $n_q$. 

• Let $v_1$ denote a vertex adjacent to $s_k$ that is either $r_m$ or a Steiner point on the path to $r_m$.
• We can assume that non-bridge in $S$ is the path between $r_1$ and $r_m$.

![Figure 7.17: Artefacts and their labels that are of interest in Lemma 7.4.7.](image)

**Lemma 7.4.7.** $|s_k r_k| \leq |s_j r_j|$

**Proof.** Observe some positional constraints for $s_j$:

• $s_j$ is in the interior of $D$.
• $|s_j s_k| \leq |n_q s_k|$, otherwise $(s_j, s_k)$ can be replaced in $S$ with a new edge $(n_q, s_k)$ giving a shorter tree and contradicting the minimality of $S$. Let $C_1$ denote a circle centred on $s_k$ with radius $|n_q s_k|$. Then $s_j$ is in the interior of, or on the boundary of
7.4 Simplest Topologies for \( \mathbb{S} \), Insertions and Elaborations

Figure 7.18: \( s_j \) is inside \( \mathbb{D} \) and in the interior of, or on the boundary of \( C_1 \) (Lemma 7.4.7).

\( C_1 \) (See Figure 7.18).

Suppose contrary to the lemma that \( |s_k r_k| > |s_j r_j| \). This implies that \( |n_q s_k| < |n_q s_j| \).

This implication can be seen from the following:

Firstly, if \( |s_k r_k| = |s_j r_j| \) and since \( \angle s_j s_k r_k = \angle r_j s_j s_k = \frac{2\pi}{3} \), then \( \{r_j, r_k, s_k, s_j\} \) are the vertices of an isosceles trapezoid (see Figure 7.19). Furthermore, \( r_j r_k \) is the chord of a circle with \( n_q \) at its centre, so \( \angle n_q r_k r_j = \angle r_k r_j n_q \) and due to the symmetry of the isosceles trapezoid, \( \triangle n_q s_k r_k \) and \( \triangle n_q r_j s_j \) are similar. Further observe that \( |n_q r_k| = |n_q r_j| = 1 \) and so \( \triangle n_q s_k r_k \) and \( \triangle n_q r_j s_j \) are congruent and it follows that \( |n_q s_k| = |n_q s_j| \).

Secondly and finally, refer to Figure 7.20 for a graphical definition of \( \theta \) and an illustration of a perturbation of interest: an anti-clockwise perturbation of \( s_k \), starting in any
Figure 7.19: $|s_k r_k| = |s_j r_j| \implies |n_q s_k| = |n_q s_j|$ (Lemma 7.4.7).

position, in a circular arc about $s_j$.

Note that the perturbation does not change $|s_j r_j|$ or $|n_q s_j|$. The derivatives for $|n_q s_k|$ and $|r_k s_k|$ are as follows:

$$|n_q s_k| = -\cos \theta < 0 \quad (7.41)$$

$$|r_k s_k| = \cos \frac{\pi}{6} > 0 \quad (7.42)$$

Let $C_2$ denote a circle centred on $n_q$ with radius $|n_q s_k|$ (see Figure 7.21). Our contrary supposition implies $s_j$ is outside the boundary of $C_2$.

Let $a$ denote a point at the intersection of $C_1$ and $C_2$ that is closest to $s_j$. Since $C_1$ and $C_2$ have the same radius, it follows that $\triangle n_q a s_k$ is equilateral (Figure 7.22) and hence:

$$\angle n_q s_k a = \frac{\pi}{3} \quad (7.43)$$

Since $s_k$ is a Steiner point:
\[ \angle v_1 s_k s_j = \frac{2\pi}{3} \]  
\hspace{1cm} (7.44)

From Equations 7.43 and 7.44 and basic geometry:
\[ \angle v_1 s_k n_q = \frac{\pi}{3} - \angle a_s k s_j \]  
\hspace{1cm} (7.45)

Observe that due to the aforementioned positional constraints for \( s_j \):
\[ \angle a_s k s_j > 0 \]  
\hspace{1cm} (7.46)

It follows from Equation 7.45 and Inequality 7.46
\[ \angle v_1 s_k n_q < \frac{\pi}{3} \]  
\hspace{1cm} (7.47)

Consider these two cases:
- Case (1). \( |v_1 s_k| \geq |n_q s_k| \)
- Case (2). \( |v_1 s_k| < |n_q s_k| \)
Figure 7.21: The contrary supposition implies $s_j$ is outside the boundary of $C_2$ (Lemma 7.4.7).
Figure 7.22: $\angle v_1 s_k n_q < \frac{\pi}{3}$ (Lemma 7.4.7).
Case (1). $|v_1s_k| \geq |n_qs_k|$

Let $b$ denote a point on $(v_1, s_k)$ such that $|bs_k| = |n_qs_k|$ (Figure 7.23).

Then $\triangle n_qs_kb$ is isosceles and given Inequality $7.47$

$$|n_qb| < |s_kb|$$

(7.48)

Consider this line segment replacement:

- Remove line segment $s_kb$. This separates $S$ into two components, one of which contains $b$ and the other that contains $n_q$.
- Insert line segment $n_qb$. This reconnects the two components.
Observe the resulting tree has length $L_S - |s_kb| + |n_qb|$ which, given Inequality 7.48, contradicts the minimality of $S$ in this case.

**Case (2).** $|v_1s_k| < |n_qs_k|

The case conditions imply that $v_1 \neq r_m$. The point $b$ is redefined to be on the ray $s_kv_1$ such that $|s_kb| = |s_kn_q|$. Let $v_2$ denote a vertex that is either $r_m$ or on the path to $r_m$. Let $c$ denote the point at the intersection of $v_1v_2$ and $n_qb$ (Figure 7.24). Clearly if $|v_1v_2| < |v_1c|$ then $S$ is self-intersecting, so we can assume now that $|v_1v_2| \geq |v_1c|$.
Observe by Inequality 7.47 and the fact that $\angle v_2v_1s_k = \frac{2\pi}{3}$ that:

$$|n_qc| < |s_kv_1| \quad (7.49)$$

Consider this line segment replacement:

- Remove line segment $s_kv_1$. This separates $S$ into two components, one of which contains $c$ and the other that contains $n_q$.
- Insert line segment $n_qc$. This reconnects the two components.

Observe the resulting tree has length $L_S - |s_kv_1| + |n_qc|$ which, given Inequality 7.49, contradicts the minimality of $S$ in this case.
7.4 Simplest Topologies for $\mathbb{S}$, Insertions and Elaborations

7.4.3 Group-1 Simplest Topologies for $\mathbb{S}$: Preliminaries

The RLLL group-1 simplest topology for $\mathbb{S}$ is shown in Figure 7.25.

Lemma 7.4.8. For any elaboration on a group-1 simplest topology for $\mathbb{S}$ in which $n_q$ is in a Cherry, if $R = R^A$ the net turning number for the non-bridge $\leq 3$.

Proof. The proof is by contradiction. It will be shown that if there are four or more net turns in the non-bridge of $\mathbb{S}$ then $R \neq R^A$. Without loss of generality we assume an RLLL group-1 simplest topology (Due to symmetry, it is sufficient to prove that $R \neq R^A$ for just one example topology from group-1). Label each Steiner point in the path along the backbone of $\mathbb{S}$ from $n_q$ as $s_1, \ldots, s_4$. Let $r_1$ denote the Rubin point adjacent to $s_1$ and label the four other Rubin points $r_2, r_3, r_4, r_5$ respectively in an anticlockwise order from $r_1$ as shown in Figure 7.25. Observe the non-bridge in $\mathbb{S}$ corresponds to the path between $r_1$
First, suppose to the contrary, there are four net turns in the non-bridge of $S$. Observe that $|n_q s_1| < 2 - \sqrt{3}$ otherwise $n_q$ and its incident edge can be removed from $S$ and the resultant tree will be shorter by more than $2 - \sqrt{3}$, implying that $L_S - L_T > 2 - \sqrt{3}$ (Recall from Remark 7.3.1 that any cases in which $L_S - L_T > 2 - \sqrt{3}$ can be discarded as candidates for $R^A$):

$$0 < |n_q s_1| < 2 - \sqrt{3} < 0.268 \quad (7.50)$$

Note: In Inequality 7.50 and subsequent inequalities, exact and approximate bounds are presented. It is often convenient and sufficient to use the approximate bounds in the sequel.

It follows that $|s_1 r_1|$ must be quite long. Applying basic trigonometry; Inequality 7.50 and the known variables $|n_q r_1| = 1$ and $\angle n_q s_1 r_1 = \frac{2\pi}{3}$:

$$|s_1 r_1| \geq 2 \sqrt{3} \sin \left( \frac{\pi}{3} - \arcsin \left( \frac{\sqrt{3}(2 - \sqrt{3})}{2} \right) \right) > 0.838 \quad (7.51)$$

Refer to Figure 7.26 for an illustration of a topology with a cherry at $n_q$ and a net turning number of 4. Observe that no Rubin point can be closer than $|s_1 r_1|$ to $r_1$, otherwise $(s_1, r_1)$ can be replaced with a shorter edge reducing $L_S$. Inequality 7.52 then follows from Inequality 7.51:

$$|r_7 r_1| \geq |s_1 r_1| > 0.838 \quad (7.52)$$

Observe that $r_7 r_{T}$ is parallel to $n_q r_{T}$. As shown in Figure 7.26, let $a$ denote a point on the tangent to $D$ at $r_7$ (on the opposite side of $n_q r_7$ to $r_1$); $\alpha = \angle a r_7 s_6; \beta = \angle r_1 n_q s_1$, and $\gamma = \angle r_7 n_q r_1$.

Then by simple geometry:

$$\alpha = \beta + \gamma - \frac{\pi}{2} \quad (7.53)$$

Angle $\alpha$ is of interest because if $\alpha > 0$ then $s_6$ and $r_6$ are outside of $D$, which is inconsistent with the construction of $S$.

Apply the law of cosines to calculate $\beta$:

$$\beta = \arccos \left( \frac{|n_q s_1|^2 + |n_q r_1|^2 - |s_1 r_1|^2}{2|n_q s_1||n_q r_1|} \right) \quad (7.54)$$
Figure 7.26: Net turning number cannot be 4 or more (Group-1 simplest topology).
Substituting for $|n_q s_1|$ from 7.50 and $|s_1 r_1|$ from 7.51 into 7.54:

$$\beta > \arccos\left(\frac{0.268^2 + 1 - 0.838^2}{2 \cdot 0.268}\right) > 46.4^\circ$$ (7.55)

Use simple trigonometry to calculate $\gamma$:

$$\gamma = 2 \sin\left(\frac{|r_1 r_7|}{2}\right)$$ (7.56)

Substitute for $|r_1 r_7|$ from Inequality 7.52 into Equation 7.56:

$$\gamma > 2 \arcsin\left(\frac{0.838}{2}\right) > 49.5^\circ$$ (7.57)

Substituting for $\beta$ from Inequality 7.55 and for $\gamma$ from Inequality 7.57 into Equation 7.53:

$$\alpha > 46.4^\circ + 49.5^\circ - 90^\circ = 5.9^\circ$$ (7.58)

Since $\alpha > 0$, $s_6$ and $r_6$ are outside disk $D$ and the conclusion is that 4 net turns is geometrically impossible for a group-1 topology. Clearly any more than 4 net turns also results in vertices outside disk $D$ and the remaining possibility is 3 or fewer net turns. $\square$

### 7.4.4 Group-1 simplest topologies for $S$: $R \neq R^A$

**Lemma 7.4.9.** If an $S$ for some $R$ has a group-1 simplest topology, or a topology that is an elaboration on a group-1 simplest topology, then $R \neq R^A$.

**Proof.** For a group-1 RLLL simplest topology it will be proven that:

1. There is a series of deformations of $S$ to yield a tree on $R$ with length greater than or equal to $L_T$ and such that the deformations monotonically decrease the length of the tree. Furthermore, it is shown that the change in length resulting from the deformations must be at least $2 - \sqrt{3}$ and therefore $L_S - L_T > 2 - \sqrt{3}$.

As a consequence of this and Remark 7.4.1 (An $S$ without an internal R-cherry has a linear topology) any $R$ with a group-1 RLLL simplest topology for $S$ can be discarded as candidates for $R^A$. By symmetry, any $R$ with any group-1 simplest
topology for $S$ can be discarded.

2. It will be shown that the above results hold for elaborations on group-1 simplest topologies.

The deformations described below deform $S$ to some spanning tree $V$ on $R$. It will be proven that $L_S - L_V > 2 - \sqrt{3}$ and it follows that $L_S - L_T > 2 - \sqrt{3}$ (see Remarks 6.13.11 and 6.13.13).

**Deformation 1:** Remove $(n_q, s_1)$ from $S$ and denote the result $S^1$ (Figure 7.27).

$$L_S - L_{S^1} = |n_q s_1|$$ (7.59)

The remaining deformations rely on the claim that $|s_1 r_1| \geq |s_2 r_2| \geq |s_3 r_3| \geq |s_4 r_4|$ if $R = R^4$ and justification for the claim is provided by Lemma 7.4.7.
Deformation 2: Starting with $S_1$, move $s_4$ to $r_4$ and move each of the other Steiner points towards their adjacent Rubin points by a distance of $|s_4 r_4|$. Denote the resulting tree $S_2$ (Figure 7.28).

$$L_{S_1} = |s_1 r_1| + |s_1 s_2| + |s_2 r_2| + |s_2 s_3| + |s_3 r_3| + |s_3 s_4| + |s_4 r_4| + |s_4 r_5| \quad (7.60)$$

$$L_{S_2} = |s'_1 r_1| + |s'_1 s'_2| + |s'_2 r_2| + |s'_2 s'_3| + |s'_3 r'_3| + |s'_3 s'_4| + |s'_4 r_4| + |r_4 r_5| \quad (7.61)$$

Observe three trapeziums with vertices $\{s_1, s'_1, s'_2, s_2\}$, $\{s_2, s'_2, s'_3, s_3\}$ and $\{s_3, s'_3, r_4, s_4\}$, each with internal angles of $\frac{2\pi}{3}$ and $\frac{\pi}{3}$. With these angles, each trapezium has the useful feature that the longest edge is equal in length to the sum of the lengths of its opposite
edge and one adjacent edge. It follows that:

\[ |s_1s_2| + |s_1r_1| = |s'_1r_1| + |s'_1s_2| \] (7.62)

\[ |s_2s_3| + |s_2r_2| = |s'_2r_2| + |s'_2s_3| \] (7.63)

\[ |s_3s_4| + |s_3r_3| = |s'_3r_3| + |s'_3s_4| \] (7.64)

It follows from equations 7.60 through 7.64:

\[ L_{S_1} - L_{S_2} = |s_4r_4| + |s_4r_5| - |r_4r_5| \] (7.65)

A lower bound for \( L_{S_1} - L_{S_2} \) as a multiple of \(|s_4r_4|\) is useful in the sequel. Commence by dividing both sides of Equation 7.65 by \(|s_4r_4|\):

\[ \frac{L_{S_1} - L_{S_2}}{|s_4r_4|} = \frac{|s_4r_4|}{|s_4r_4|} + \frac{|s_4r_5|}{|s_4r_4|} - \frac{|r_4r_5|}{|s_4r_4|} = 1 + \frac{|s_4r_5|}{|s_4r_4|} - \frac{|r_4r_5|}{|s_4r_4|} \] (7.66)

Observe that \(|s_4r_4| < |s_4r_5|\) since the ray \(\overrightarrow{s_4s_3}\) must pass between \(r_4\) and \(n_q\) in the RLLL topology:

\[ 0 < |s_4r_4| < |s_4r_5| \] (7.67)

Refer to Figure 7.29 for some details concerning the points \(s_4, r_4,\) and \(r_5\). Let \(e\) denote the third vertex on an equilateral triangle whose other vertices are \(r_4\) and \(r_5\) such that \(e\) is on the opposite side to \(r_4r_5\) to \(s_4\). Let \(c\) denote the centre of the equilateral triangle. Let \(A\) denote the circular arc between \(r_4\) and \(r_5\) and centred on \(c\). It is a basic property of a Steiner tree that \(s_4\) must be on arc \(A\).

Consider a perturbation of \(s_4\) along arc \(A\) towards \(r_5\). Observe that \(|r_4r_5|\) is unaffected by this perturbation. The effect of the perturbation on Equation 7.66 is as follows:

- The second term \((\frac{|s_4r_5|}{|s_4r_4|})\) is a decreasing function.
- The third term \((\frac{|r_4r_5|}{|s_4r_4|})\) is an increasing function and the rate of increase is less...
than the rate of decrease for the second term because \( |r_4r_5| > |s_4r_5| \), and because \( |s_4r_5| \) decreases with the perturbation but \( |r_4r_5| \) does not change.

It follows that \([7.66]\) is a decreasing function with respect to the perturbation and as such a lower bound can be found when \( |s_4r_4| \) is as close as possible to \( r_5 \). Since \( |s_4r_4| < |s_4r_5| \) (Inequality \([7.67]\)) it follows that a lower bound for \( \frac{L_{S_1} - L_{S_2}}{|s_4r_4|} \) (Equation \([7.66]\)) can be found when \( |s_4r_4| \) approaches \( |s_4r_5| \) from below.

\[
\frac{L_{S_1} - L_{S_2}}{|s_4r_4|} > 2 - \frac{|r_4r_5|}{|s_4r_4|}
\]

(7.68)

If \( |s_4r_4| = |s_4r_5| \) then \( \triangle s_4r_4r_5 \) is isosceles and since \( \angle r_4s_4r_5 = \frac{2\pi}{3} \) it is easy to see that:

\[
|r_4r_5| = 2|s_4r_4| \sin \frac{\pi}{3} = \sqrt{3}|s_4r_4|
\]

(7.69)

Substitute for \( |r_4r_5| \) from Equation \([7.69]\) into Inequality \([7.68]\) and multiply both sides by \( |s_4r_4| \):

\[
\frac{L_{S_1} - L_{S_2}}{|s_4r_4|} > 2 - \sqrt{3} \implies L_{S_1} - L_{S_2} > (2 - \sqrt{3})|s_4r_4|
\]

(7.70)

Deformations 3-5: Deformations 3, 4 and 5 are illustrated in Figures \([7.30]\), \([7.31]\) and \([7.32]\) respectively.

Following similar notation as for Deformation 2, denote the trees resulting from the deformations 3, 4 and 5 as \( S_3 \), \( S_4 \) and \( S_5 \) respectively. Observe:

1. Deformations 2 through 5 differ only in the numbers of internal edges and remaining external edges. In each case the shorter external edge of an R-cherry is collapsed (the R-cherry is replaced with a single edge) and \( n \) internal edges and \( n \) other external edges are shortened and lengthened respectively by calculable amounts for some \( n \geq 0 \). The result in each case is the same, with the change in length being \( (2 - \sqrt{3}) \) times the length of the aforementioned shorter edge. It follows that the derivative calculations for deformations 3 through 5 are essentially the same as for deformation 2. It also follows that all deformations give a monotonic decrease in the length of the tree. In the sequel this type of deformation will
be referred to as a Brazil deformation\(^2\).

2. With this sequence of Brazil deformations:

   (a) Deformation 2 moves \(s_4\) a distance of \(|s_4r_4|\).
   (b) Deformation 3 moves \(s_3\) a distance \(|s_3r_3| - |s_4r_4|\).
   (c) Deformation 4 moves \(s_2\) a distance \(|s_2r_2| - (|s_3r_3| + |s_4r_4|)\).
   (d) Deformation 5 moves \(s_1\) a distance \(|s_1r_1| - (|s_2r_2| + |s_3r_3| + |s_4r_4|)\).

3. \(S^5 = V\) (Figure 7.32).

It follows from the above observations that:

\[
L_{S^5} - L_{S^5} = L_{S^5} - L_V > (2 - \sqrt{3})(|s_4r_4| + |s_3r_3| - |s_4r_4| + |s_2r_2| - |s_3r_3| + |s_1r_1| - |s_2r_2|)
\Rightarrow L_{S^5} - L_V > (2 - \sqrt{3})|s_1r_1| \quad (7.71)
\]

\(^2\)Named in honour of Associate Professor Marcus Brazil, who suggested this approach.
Figure 7.30: Deformation 3 (Group-1 simplest topology).

Figure 7.31: Deformation 4 (Group-1 simplest topology).
Substituting for $L_S$ from \(7.59\) into \(7.71\):

$$L_S - L_V > |n_q s_1| + (2 - \sqrt{3})|s_1 r_1|$$  \(7.72\)

The edge \((n_q, s_1)\) has length domain \(0 < |n_q s_1| < 2 - \sqrt{3}\) and there is an obvious geometric relationship between \(|n_q s_1|\) and \(|s_1 r_1|\): \(|n_q s_1| \sin \frac{\pi}{3} + |s_1 r_1| \sin \frac{\pi}{3} = 1\). A lower bound for $L_S - L_V$ can be found by substituting \(|n_q s_1| = 0\) and \(|s_1 r_1| = 1\) into \(7.72\):

$$L_S - L_V > 2 - \sqrt{3}$$  \(7.73\)

Attention finally turns to elaborations. The effects of L or R insertions into each edge of the RLLL group-1 simplest topology are shown in Table \(7.2\).
Table 7.2: The effects of L or R insertions into each edge of the RLLL group-1 simplest topology

<table>
<thead>
<tr>
<th>Insertion</th>
<th>Effect</th>
<th>Subsequent insertions</th>
</tr>
</thead>
<tbody>
<tr>
<td>L or R insertion into the edge incident to $r_1$.</td>
<td>Equivalent to some insertion in a group-2 simplest topology. See Table 7.3.</td>
<td>Any subsequent insertion is equivalent to some elaboration on a group-2 simplest topology. See Table 7.3.</td>
</tr>
<tr>
<td>L or R insertion into the external edge incident to $r_5$, or R insertion into the edge incident to $n_q$, or R insertion into an internal edge.</td>
<td>Extends the linear topology by one bridge. It is easy to see that the Brazil deformation can be extended to this case to achieve the same result as in Inequality 7.73 and accordingly, $R \neq R^A$.</td>
<td>Any subsequent insertions that extend the linear topology have the same effect and in addition, if the insertion leads to a net turning number for the non-bridge $\geq 4$, must have a terminal outside of $D$ (Lemma 7.4.8).</td>
</tr>
<tr>
<td>L or R insertion into an external edge, other than the external edge incident to $r_5$; or L-insertion into the edge incident to $n_q$; or L-insertion into an internal edge.</td>
<td>Creates an internal R-cherry and accordingly $R \neq R^A$ (Lemma 7.4.4).</td>
<td>Any subsequent elaborations also contain an internal R-cherry (Lemma 7.4.5) and accordingly $R \neq R^A$.</td>
</tr>
</tbody>
</table>

7.4.5 Group-2 Simplest Topologies for $S$: Preliminaries

**Lemma 7.4.10.** For any elaboration on a group-2 simplest topology for $S$, the net turning number for the non-bridge $\leq 5$.

**Proof.** Consider the path from one Rubin point to the other in the non-bridge. Starting from one Rubin point, after six or more net turns on the interior of $D$, the other Rubin point cannot be reached without crossing an edge (Figure 7.33).

Clearly seven or more net turns give the same contradiction. $\square$
Figure 7.33: The two edges incident to the Rubin points for the non-bridge are parallel and cannot both intersect the boundary of $\mathbb{D}$ from its interior.

7.4.6 Group-2 Simplest Topologies for $S$: $R \neq R^A$

Lemma 7.4.11. If an $S$ for some $R$ has a group-2 simplest topology, or a topology that is an elaboration on a group-2 simplest topology, then $R \neq R^A$.

Proof. It will first be proven that $R \neq R^A$ for the LRLL group-2 simplest topology, and due to symmetry, the proof holds for all group-2 topologies. Figure 7.34 shows the LRLL topology with Rubin points labelled in such a way as to enable reuse of equations used for group-1 simplest topologies.

As for group-1 simplest topologies, $S$ is transformed into a spanning tree $V$ using a series of Brazil deformations such that the length of the tree monotonically decreases and such that $L_S - L_V > 2 - \sqrt{3}$.

Deformation 1: Remove $(n_q, s_2)$ from $S$ and denote the result $S^1$.

\[ L_S - L_{S^1} = |n_q s_2| \]  

(7.74)

Point $s_2$ is now replaced with coincident but disconnected points $s_{2a}$ and $s_{2b}$, splitting
Figure 7.34: Group-2 LRLL simplest topology for $S$.

$S^1$ into two components $S^{1a}$ and $S^{1b}$ accordingly, such that:

$$V(S^{1a}) = \{ r_1, r_2, s_1, s_{2a} \}$$ (7.75)

$$V(S^{1b}) = \{ r_3, r_4, r_5, s_3, s_4, s_{2b} \}$$ (7.76)

Observe:

$$L_{S^{1a}} + L_{S^{1b}} = L_{S^1}$$ (7.77)

Let $a$ denote the intersection of $(r_2r_3)$ and a ray $n_q s_{2a}$ as shown in Figure 7.35.

Observe $0 < |r_2r_3| < \sqrt{3}$ (Lemma 7.3.2); it follows that:

$$\frac{1}{2} < |n_q a| = |n_q s_{2a}| + |s_{2a}| < 1$$ (7.78)

**Deformation 2:** This Brazil deformation relies on the claim that $|s_3 r_3| \geq |s_4 r_4|$ if $R = R^A$, which follows from Lemma 7.4.7.
Figure 7.35: Deformation 1 for group-2 simplest topology with \( S^{1a} \) (green), \( S^{1b} \) (purple) and construction point \( a \).
Starting with $S^{1b}$, move $s_4$ to $r_4$ and move $s_3$ and $s_2b$ towards $r_3$ and $a$ respectively by distance $|s_4r_4|$. Denote the new positions for $s_3$ and $s_2b$ as $s_3'$ and $s_2b'$ respectively. Denote the resulting tree as $S^{2b}$ (Figure 7.36).

\[
L_{G^{1b}} - L_{G^{2b}} = |s_4r_5| + |s_4r_4| + |s_3s_4| + |s_3r_3| + |s_2bs_3| - |r_4r_5| - |s_3'r_3| - |s_3's_3'| \quad (7.79)
\]

Observe:

\[
|s_2bs_3| = |s_2b's_3'| \quad (7.80)
\]

\[
|s_3s_4| + |s_3r_3| = |s_3'r_3| + |s_3'r_4| \quad (7.81)
\]

Substitute for $|s_2bs_3|$ and for $|s_3s_4| + |s_3r_3| - |s_3'r_3|$ from Equations 7.80 and 7.81 respectively.
7.4 Simplest Topologies for S, Insertions and Elaborations

Figure 7.37: Deformation 3 with $S^{3b}$ shown in solid purple (Group-2 simplest topology).

\[
L_{S^{1b}} - L_{S^{2b}} = |s_4r_5| + |s_4r_4| - |r_4r_5| \quad (7.82)
\]

As for deformation 2 in the group-1 simplest topology observe $\angle r_4s_4r_5 = \frac{2\pi}{3}$ and $|s_4r_4| < |s_4r_5|$. A lower bound for $L_{S^{1b}} - L_{S^{2b}}$ as a multiple of $|s_4r_4|$ is given when $|s_4r_4| \to |s_4r_5|:

\[
\lim_{|s_4r_4| \to |s_4r_5|} L_{S^{1b}} - L_{S^{2b}} = (2 - \sqrt{3})|s_4r_4| \implies L_{S^{1b}} - L_{S^{2b}} > (2 - \sqrt{3})|s_4r_4| \quad (7.83)
\]

\[
L_{S^{1b}} - L_{S^{2b}} > (2 - \sqrt{3})|s_4r_4| \quad (7.84)
\]

Deformation 3: Starting with $S^{2b}$, move $s_3^\prime$ to $r_3$ and move $s_{2b}^\prime$ towards $a$ by distance $|s_3^\prime r_3|$. Denote the new position for $s_{2b}^\prime$ as $s_{2b}^{\prime\prime}$ and denote the resulting tree as $S^{3b}$ Figure 7.37.

The form is the same as deformation 2, just with one fewer edge deformed and it is
Figure 7.38: Deformation 4 with $S^{4b}$ shown in solid purple (Group-2 simplest topology).

easy to see the result is as follows:

$$L_{S^{2b}} - L_{S^{4b}} > (2 - \sqrt{3})|s'_3r_3|$$  \hspace{1cm} (7.85)

**Deformation 4:** Starting with $S^{3b}$, move $s''_{2b}$ to $a$. Denote the resulting tree as $S^{4b}$ (Figure 7.38).

Observe that this differs from previous deformation in that $\angle as''_{2b}r_3 < \frac{2\pi}{3}$, but that the same lower bound form easily applies:

$$L_{S^{3b}} - L_{S^{4b}} > (2 - \sqrt{3})|s''_{2b}a|$$  \hspace{1cm} (7.86)

Observe for deformations 2 through 4:

$$|s_4r_4| + |s'_3r_3| + |s''_{2b}a| = |s_{2b}a|$$  \hspace{1cm} (7.87)
Deformations 5 and 6:

Let $S^{3a}$ be the tree with vertices $r_1, r_2$ and $a$ and edges $r_1r_2$ and $r_2a$; then by a similar argument:

$$L_{S^{3a}} - L_{S^{4b}} > (2 - \sqrt{3})|s_{2a}a|$$

(7.89)

Recall $\mathcal{V}$ is some spanning tree on $R$ and observe:

$$S^{3a} \cup S^{4b} = \mathcal{V}$$

(7.90)
\[ L_{S^{3a}} + L_{S^{4b}} = L_V \]  
(7.91)

It follows from 7.78 and 7.91:

\[ L_{S^1} - L_V = L_{S^{1a}} + L_{S^{1b}} - L_{S^{3a}} + L_{S^{4b}} \]  
(7.92)

Substitute for \( L_{S^1} - L_{S^{1a}} \) and \( L_{S^1} - L_{S^{3a}} \) from 7.84 and 7.89 respectively into 7.92:

\[ L_{S^1} - L_V > (2 - \sqrt{3})(|s_{2a}| + |s_{2b}|) \]  
(7.93)

Since \( |s_{2a}| = |s_{2b}| = |s_2a| \), Inequality 7.93 can be simplified:

\[ L_{S^1} - L_V > 2(2 - \sqrt{3})|s_{2a}| \]  
(7.94)

Add 7.74 to 7.93:

\[ L_S - L_V > |n_q s_2| + 2(2 - \sqrt{3})|s_{2a}| \]  
(7.95)

Recall from 7.78 that \( \frac{1}{2} < |n_q s_2| + |s_{2a}| < 1 \) and observe \( L_S - L_V \) approaches a minimum when \( |n_q a| \to \frac{1}{2} \) and \( |n_q s_2| \to 0 \) implying \( |s_{2a}| \to \frac{1}{2} \):

\[ L_S - L_T \geq L_S - L_V > 2 - \sqrt{3} \]  
(7.96)

We finally turn our attention to insertions and elaborations on group-2 simplest topologies. The effects of L and R insertions into each edge of the LRLL group-2 simplest topology is shown in Table 7.3.

7.5 Final theorem

The following theorem is the main finding supporting Replacement Argument A.

**Theorem 7.5.1.** In the case that \( n_q \) has degree-1 in \( S \), then \( \inf L_S - L_{\partial} = \inf L_S - L_T = 2 - \sqrt{3} \).
Table 7.3: The effects of $L$ or $R$ insertions into each edge of the RLLL group-2 simplest topology

<table>
<thead>
<tr>
<th>Insertion</th>
<th>Effect</th>
<th>Subsequent insertions</th>
</tr>
</thead>
<tbody>
<tr>
<td>L or R insertion into the edge incident to $n_q$</td>
<td>Creates an internal R-cherry and accordingly $R \neq R^A$ (Lemma 7.4.4)</td>
<td>Any further elaborations also contain an internal R-cherry (Lemma 7.4.5) and accordingly $R \neq R^A$.</td>
</tr>
<tr>
<td>L or R insertion into the external edge incident to $r_1$ or $r_5$</td>
<td>Extends the linear topology by one bridge. It is easy to see that a Brazil deformation can be added to the above-mentioned deformations to achieve the same result as in Inequality 7.96 and accordingly, $R \neq R^A$.</td>
<td>Any subsequent insertions that extend the linear topology have the same effect and in addition, if the insertion leads to a net turning number for the non-bridge $\geq 5$, then the topology must have a terminal outside of $D$ (Lemma 7.4.10). Any subsequent insertions that do not extend the linear topology necessarily create an internal R-cherry (Lemma 7.4.5).</td>
</tr>
<tr>
<td>L or R insertion into an internal edge</td>
<td>Either: (a) Creates an internal R-cherry, or; (b) Extends the linear topology by one bridge.</td>
<td>(a) Any subsequent insertions that extend the linear topology have the same effect and in addition, if the insertion leads to a net turning number for the non-bridge $\geq 5$, must have a terminal outside of $D$ (Lemma 7.4.10). (b) Any further elaborations also contain an internal R-cherry (Lemma 7.4.5).</td>
</tr>
</tbody>
</table>
Proof. The proof accumulates from the following:

- Lemmas 7.2.1, 7.2.3 prove that if $R$ has fewer than two Rubin points, then $L_S - L_U > 2 - \sqrt{3}$. Furthermore, observe $2 - \sqrt{3}$ is the infimum for $L_S - L_U$ when $m \leq 2$.
- The limiting cases for $m \leq 2$ have $U=T$ (Corollaries 7.2.2 and 7.2.1) and it follows that $2 - \sqrt{3}$ is the infimum for $L_S - L_T$ when $m \leq 2$.
- Lemmas 7.4.9 and 7.4.11 prove that when $m \geq 3$, $R \neq R^A$, since there is no case in which $L_S - L_T \leq 2 - \sqrt{3}$.

It follows that $2 - \sqrt{3}$ is the infimum for both $L_S - L_T$ and $L_S - L_U$ for all $m$. \qed
Chapter 8
A universal constant for replacement argument B

REPLACEMENT Argument B (Chapter 6, Section 6.15) relies on the claim that for any $R$, if $n_q$ has degree-2 or 3 in $S$, then $L(L_S - L_F) = 2(2 - \sqrt{3})$. This chapter is devoted to proving that claim and outlining a conjectured much stronger claim that $\inf (L_S - L_F) = \sqrt{2}(\sqrt{3} - 1)$. The argument for $L(L_S - L_F) = 2(2 - \sqrt{3})$ is straightforward, as it reuses Theorem 7.5.1 for $\inf (L_S - L_T)$, which applies to the case that $n_q$ has degree-1 in $S$. The approach is to simply double the earlier result for one FST, to obtain a result for two FSTs. The conjecture for $\inf (L_S - L_F) = \sqrt{2}(\sqrt{3} - 1)$ relies on a proof for all cases involving four or fewer Rubin points, and a number of lemmas which increase the likelihood that no $R$ with more than four Rubin points can improve the result.

8.1 Definitions and notation for this chapter

For convenience a list of some previously defined terms relevant to this chapter are provided in Table 8.1. A number of abbreviations and symbols used in this chapter are common to Chapters 6 and 7. Accordingly, readers may find the listings of commonly used symbols and abbreviations in Tables A.2 to A.4 of Appendix A useful.

The following additional definitions are relevant to this chapter.

Definition 8.1.1 ($R^B_k$). $R^B_k$ is a set of Rubin points that satisfies the following conditions:
- $n_q$ has degree-2 or 3 in $S$, and;
- no perturbations of Rubin points can reduce $L_S - L_F$.

Definition 8.1.2 ($R^B$). $R^B$ is the smallest set of Rubin points that satisfies the following conditions:
Table 8.1: Some previously defined terms relevant to this chapter.

<table>
<thead>
<tr>
<th>Name</th>
<th>Brief description</th>
<th>Reference</th>
</tr>
</thead>
<tbody>
<tr>
<td>bridge in $S$</td>
<td>A convex path between two Rubin points adjacent on the boundary of $\mathbb{D}$ in the same FST of $S$.</td>
<td>Def. 6.13.4</td>
</tr>
<tr>
<td>chord</td>
<td>In this thesis, the term chord is used only in relation to Rubin points that are adjacent on the boundary of $\mathbb{D}$.</td>
<td>Sec. 6.13.4</td>
</tr>
<tr>
<td>longest chord</td>
<td>A chord such that no chord is longer.</td>
<td>Def. 6.13.6</td>
</tr>
<tr>
<td>non-bridge in $S$</td>
<td>A non-convex path between two Rubin points adjacent on the boundary of $\mathbb{D}$.</td>
<td>Def. 6.13.5</td>
</tr>
<tr>
<td>R-cherry</td>
<td>A cherry in $S$ on two Rubin points.</td>
<td>Def. 7.1.1</td>
</tr>
<tr>
<td>shorter chord</td>
<td>Shorter than a longest chord.</td>
<td>Def. 6.13.7</td>
</tr>
</tbody>
</table>

- $n_q$ has degree-2 or 3 in $S$, and;
- no additions, removals or perturbations of Rubin points can reduce $L_S - L_F$.

**Definition 8.1.3** (Shortest chord). A shortest chord is a chord such that no other chord is shorter.

**Definition 8.1.4** ($2^{nd}$ longest chord). A $2^{nd}$ longest chord in $R$ is a chord that is not shorter than any chord other than a longest chord.

**Definition 8.1.5** (Middling chord). A middling chord is shorter than a $2^{nd}$ longest chord, and longer than a shortest chord.

Observe that $2^{nd}$ longest chords, middling chords and shortest chords are all shorter chords.

Some notation and naming conventions:
- Let $c_l$, $c_2$, $c_m$ and $c_s$ denote the lengths of a longest, $2^{nd}$ longest, middling and shortest chords respectively.
- With reference to an FST, size-$n$ indicates that the FST has $n$ terminals (that is, $n_q$ and $n-1$ Rubin points).
8.2 Three FSTs

This section provides an argument for setting aside consideration of cases of \( S \) involving three FSTs. This is convenient when considering cases with five or more Rubin points in \( S \).

**Lemma 8.2.1.** For every \( S \) with three FSTs there exists an \( \hat{S} \) with two FSTs where \( L_S \rightarrow L_{\hat{S}} \).

**Proof.** Recall from Section 6.13.2 that the set of Rubin points is \( R \subseteq R^* = \mathbb{S}^1 \cap \delta(\mathbb{D}) \), such that \( R \subset V(S) \) and \( R = \{r_1, r_2, \ldots, r_m\} \). Let \( n_1 \) denote \( n_q \)’s nearest neighbour in \( T(S^1) \) (see Section 6.13.2) and note \( T(S^1) \subseteq V(S^1) \). Let \( r_1 \) denote the Rubin point adjacent to \( n_1 \) in \( S^2 \) (see Section 6.13.3). Recall from Section 6.13.2 that \( |n_1 r_1| = \mu \), where \( \mu \) is an infinitesimal.

\( \hat{S} \) can be constructed from \( S \) as follows:

1. Select one of the three edges in \( S \) incident to \( n_q, (n_q, v_1) \), such that the \( \overrightarrow{r_1n_1} \) component of \( \overrightarrow{n_qv_1} \) is positive (at least one of the three edges meets this requirement).
2. Let \( \hat{n}_q \) denote a vertex on \( n_qv_1 \) such that \( |n_q \hat{n}_q| < |r_1 n_1| \).
3. Let \( \hat{D} \) denote a disk centred on \( \hat{n}_q \) with the same radius as \( D \).
4. Let \( \hat{R} = \{\hat{r}_1, \ldots, \hat{r}_m\} \), \( \hat{R} \subseteq \hat{R}^* = \mathbb{S}^1 \cap \delta(\hat{D}) \), with details of the construction described in Section 6.13.2.
5. Let \( a \) denote a vertex coincident with \( n_q \).
6. \( \hat{S} = (V(\hat{S}), E(\hat{S})) \) where:
   (a) \( V(\hat{S}) = \{\hat{n}_q, a\} \cup \hat{R} \cup V(S) \setminus \{R, \{n_q\}\} \).
   (b) \( E(\hat{S}) \) can be found by commencing with \( E(S) \) and:
      i. Replace edge \( (n_q, v_1) \) with edges \( (n_q, \hat{n}_q) \) and \( (\hat{n}_q, v_1) \).
      ii. For each edge with \( n_q \) as an end point, replace \( n_q \) with \( a \).
      iii. For \( i = \{1, 2, \ldots, m\} \), for the edge with an end point \( r_i \), replace \( r_i \) with \( \hat{r}_i \).

Refer to Figure 8.1 for example \( S, \hat{S} \) and associated artefacts. Observe \( a \) meets the definition of a Steiner point. Observe that \( \hat{n}_q \) is degree-2 in \( \hat{S} \) and that:

\[
\lim_{|n_q \hat{n}_q| \rightarrow 0} L_S = L_{\hat{S}} \tag{8.1}
\]

and the lemma follows.
Corollary 8.2.2. In seeking a lower bound for $L_S - L_{\mathcal{F}}$ it suffices to consider cases in which $S$ has two FSTs.

8.3 An easy lower bound for $L_S - L_{\mathcal{F}}$

Theorem 8.3.1. In the case that $n_q$ has degree-2 or 3 in $S$, then a lower bound for $L_S - L_{\mathcal{F}}$ is given by $L(L_S - L_{\mathcal{F}}) = 2(2 - \sqrt{3})$.

Proof. From Theorem 7.5.1 In the case that $n_q$ has degree-1 in $S$, then:

$$\inf (L_S - L_T) = 2 - \sqrt{3}$$ (8.2)
8.3 An easy lower bound for $L_S - L_F$

where $T$ is a minimum spanning tree on the Rubin points. Let $Z$ denote a tree comprising all the chords corresponding to bridges in $S$. Then $L_Z \geq L_T$ and it follows that in the degree-1 case, that:

$$L_S - L_Z \geq 2 - \sqrt{3}$$

(8.3)

Now consider each of the FSTs in the case that $n_q$ has degree-2 and denote the two components of $S$ as $S^1$ and $S^2$. Let $Z^1$ and $Z^2$ denote trees comprising all the chords corresponding to bridges in $S^1$ and $S^2$ respectively. Recall $F$ denotes a two component forest on $R$ with minimal length. Observe that $Z^1 \cup Z^2$ comprise a two component forest on $R$, not necessarily minimal, and it follows that:

$$L_{Z^1} + L_{Z^2} \geq L_F$$

(8.4)

It is easy to see that $S^1$ and $Z^1$ in the degree-2 case are equivalent to, but more constrained than $S$ and $Z$ in the degree-1 case, and it follows that:

$$L_{S^1} - L_{Z^1} \geq L_S - L_Z$$

(8.5)

The same argument applies to $S^2$ and $Z^2$:

$$L_{S^2} - L_{Z^2} \geq L_S - L_Z$$

(8.6)

Substitute for $L_S - L_Z$ from Inequality 8.3 into Inequalities 8.5 and 8.6:

$$L_{S^1} - L_{Z^1} \geq 2 - \sqrt{3}$$

(8.7)

$$L_{S^2} - L_{Z^2} \geq 2 - \sqrt{3}$$

(8.8)

Add Inequalities 8.7 and 8.8 together:

$$L_{S^1} + L_{S^2} - L_{Z^1} - L_{Z^2} \geq 2(2 - \sqrt{3})$$

(8.9)
Substitute for $L_{Z_1} + L_{Z_2}$ from Inequality 8.4 into 8.9 and recall that $S_1$ and $S_2$ are the two components of $S$ in the degree-2 case and this gives for the degree-2 case:

$$L_S - L_F \geq 2(2 - \sqrt{3}) \tag{8.10}$$

Inequality 8.10 proves the theorem in the degree-2 case and the degree-3 case can be ignored due to Lemma 8.2.1.

8.4 Conjectured tighter lower bound for $L_S - L_F$

**Conjecture 8.4.1.** For any $R$, if $n_q$ has degree-2 or 3 in $S$, then \( \inf (L_S - L_F) = \sqrt{2}(\sqrt{3} - 1) \).

Conjecture 8.4.1 is predicated on a number of lemmas and observations. Firstly a benchmark is given in Lemma 8.5.1 where $L_S - L_F = \sqrt{2}(\sqrt{3} - 1)$. It resembles the benchmark cases for the Steiner ratio conjecture \[84\] and for the universal constant for replacement argument A (Chapter 7, Lemma 7.2.3), in as much as it comprises degree-3 FSTs (two degree-3 FSTs in this case). It remains to eliminate all other Rs as candidates for $R^B$. This has been achieved in part as follows:

- Cases with size-2 FSTs can be ignored (Lemma 8.6.1).
- Cases with three or fewer Rubin points can be ignored (Lemmas 8.7.1 and 8.7.2).
- Cases with four Rubin points, other than the benchmark case (Lemma 8.5.1) can be ignored (Lemma 8.8.1).
- Cases in which $S$ has three FSTs can be ignored (Lemma 8.2.1).

It follows that if the benchmark given in Lemma 8.5.1 is not $R^B$, then $R^B$ must have five or more Rubin points or stated another way, it must be comprised of two FSTs, at least one of which has size-4 or greater. In relation to the universal constant for replacement argument A, it was proven that $R^A$ comprised a single size-3 FST. For this replacement argument B, the equivalent would be to prove that $R^B$ comprises exactly two size-3 FSTs. This has not been proven, but there are certain similarities between replacement arguments A and B that make this seem likely:

- There is a special significance to long chords in both cases:
8.5 A benchmark for $L_S - L_F$

- Replacement argument B: Lemma 8.9.1, Corollary 8.9.2, Lemma 8.9.3
- Replacement argument A: Lemmas 7.3.1, 7.3.4

- There is a special significance to R-cherries in both cases:
  - Replacement argument B: Lemma 8.9.4.
  - Replacement argument A: Lemma 7.3.3.

And finally, it has been proven that there exists an $R^B$ such that the chord lengths for the non-Bridges in $S$ have equal length (Lemma 8.9.5) and the aforementioned benchmark case is consistent with this requirement.

The completion of a proof for this conjecture would considerably improve the effectiveness of replacement argument B, by replacing $L_S - L_F \geq 2(2 - \sqrt{3}) \approx 0.536$ with

$$\inf (L_S - L_F) = \sqrt{2} (\sqrt{3} - 1) \approx 1.035.$$

8.5 A benchmark for $L_S - L_F$

**Lemma 8.5.1.** If $R$ comprises four equally spaced Rubin points then $L_S - L_F = \sqrt{2} (\sqrt{3} - 1)$.

**Proof.** Four points of a square with side length 1 can be connected with a minimum Steiner tree (MStT) with length $1 + \sqrt{3}$ (Jarnik & Kössler [108]). Refer to Figure 8.2.

![Figure 8.2: Four points of a square with side length 1 can be connected with an MStT with length $1 + \sqrt{3}$.](image)

Four equally spaced points on the boundary of a disk with radius 1 are the four corners of a square with side length $\sqrt{2}$ and it follows that an MStT connecting them has
length $\sqrt{2}(1 + \sqrt{3})$ (see Figure 8.3).

Figure 8.3: $S$ has length $\sqrt{2}(1 + \sqrt{3})$. $F$ comprises any two chords each with length $\sqrt{2}$.

Observe that the centre of the disk $n_q$ is a point on the internal edge of the MStT. It follows that $S$ for $R \cup \{n_q\}$, where $R$ comprises four equally spaced Rubin points, has length $\sqrt{2}(1 + \sqrt{3})$. It is easy to see that $F$ for $R$ comprises any two chords, each of which has length $\sqrt{2}$ and:

$$L_S - L_F = \sqrt{2} (\sqrt{3} - 1)$$

(8.11)

Corollary 8.5.2. Any $R$ for which $L_S - L_F > \sqrt{2} (\sqrt{3} - 1)$ and any $R$ with $m > 4$ for which $L_S - L_F \geq \sqrt{2} (\sqrt{3} - 1)$ can be ignored as candidate for $R^B$.

Remark 8.5.1. Recall from Section 6.13.2 that a Rubin point is a point on an edge of $S^1$ where exactly one of the vertices of the edge is inside $\mathbb{D}$. The benchmark case in Lemma 8.5.1 is peculiar in that it further requires that the vertex of the edge that is not inside $\mathbb{D}$ is coincident with the Rubin point. In other words there are exactly four vertices in $S^1$ that are equidistant from $n_q$ in a regular star configuration (see Figure 8.4). Suppose to
the contrary one of the four edges passing through the boundary of $\mathbb{D}$ has a vertex $t_1$ that is not coincident with its corresponding Rubin point. Then the $S$ in the benchmark case is not minimal (In Figure 8.4 compare the benchmark $S$ shown in black with the shorter tree on $t_1$ and three Rubin points shown in purple).

![Figure 8.4: The benchmark case is only applicable when there are exactly four vertices in $S^1$ that are equidistant from $n_q$ in a regular star configuration. This illustration shows that if $t_1$ is not equidistant, then the benchmark cannot apply.](image)

The value of this observation is that it points to possible improvements in replacement argument B, which take into account more neighbours than just the nearest neighbour to $n_q$. It is beyond the scope of this thesis to develop such a method.

### 8.6 Size-2 FSTs

**Lemma 8.6.1.** If an $S$ for an $R$ includes one or more size-2 FSTs then $R \neq R^B$.

**Proof.** Consider these mutually exclusive and collectively exhaustive cases for the com-
position of $S$:

1. Three size-2 FSTs
2. Two size-2 FSTs
3. Three FSTs, one or two of which are size-2
4. One size-2 FST and one size $> 2$ FST

**Case (1). Three size-2 FSTs**: Observe that since $n_q$ has degree-3, all edges incident to $n_q$ must meet at $\frac{2\pi}{3}$ implying that $R$ must comprise three equally spaced Rubin points for which $L_S - L_F = 3 - \sqrt{3}$. Then by Corollary 8.5.2, $R \neq R^B$ (Figure 8.5).

![Figure 8.5: Three size-2 FSTs (Lemma 8.6.1).](image)

**Case (2). Two size-2 FSTs**: In this case $S$ has two edges, each with length 1, and $F$ has no edges. Accordingly, $L_S - L_F = 2$ and by Corollary 8.5.2, $R \neq R^B$ (Figure 8.6).

**Case (3). Three FSTs, one or two of which are size-2**: Consider a size $> 2$ FST and a size-2 FST. The third FST can be ignored, other than noting that because of its presence, all edges incident to $n_q$ must meet at $\frac{2\pi}{3}$. It will be shown that $R \neq R^B$, or that $S$ is self-intersecting (in which case this topology cannot apply to $R^B$). The end-result of the construction is shown in Figure 8.7 and the details of the construction are as follows: There must be a
Steiner point adjacent to $n_q$ in the size $> 2$ FST. Let $s_1$ denote this Steiner point. Suppose $R = R^B$. Then:

$$|n_q s_1| < \sqrt{2} (\sqrt{3} - 1) - 1 < 0.036$$  \hfill (8.12)

otherwise the edge in the size-2 FSTs and $(n_q, s_1)$ can be removed from $\mathcal{S}$ to give a two-component forest $F$ where $L_F \geq L_{\mathcal{S}}$. This implies $L_{\mathcal{S}} - L_F > \sqrt{2} (\sqrt{3} - 1)$ and by Corollary $8.5.2$ $R \neq R^B$. For now, assume $R = R^B$ and accordingly that Inequality $8.12$ holds. Let $s_2$ denote a Steiner point adjacent to $s_1$ as shown in Figure $8.7$ (It will become apparent that this vertex must be a Steiner point, not a Rubin point). Then by Lemma $6.16$

$$|n_q s_1| \geq \frac{\sqrt{3} - 1}{2} |s_1 s_2|$$  \hfill (8.13)

Substitute for $|n_q s_1|$ from $8.12$ into $8.13$

$$\sqrt{2} (\sqrt{3} - 1) - 1 > \frac{\sqrt{3} - 1}{2} |s_1 s_2| \implies |s_1 s_2| < \frac{2\sqrt{2} (\sqrt{3} - 1) - 2}{\sqrt{3} - 1} < 0.097$$  \hfill (8.14)
Inequalities 8.13 and Inequalities 8.14 show that \( s_2 \) must be a Steiner point, as the point is at least \( 1.0 - 0.036 - 0.097 \) from the boundary of \( \mathbb{D} \) (precise calculations are not required – refer to Figure 8.7).

Let \( s_3 \) denote a Steiner point adjacent to \( s_2 \) as shown in Figure 8.7 (It will become apparent that this must be a Steiner point). Then by Lemma 6.16 and Inequality 8.14:

\[
\frac{2\sqrt{2}(\sqrt{3} - 1) - 2}{\sqrt{3} - 1} > \frac{\sqrt{3} - 1}{2} |s_2s_3| \implies |s_2s_3| < \frac{4\sqrt{2}(\sqrt{3} - 1) - 4}{(\sqrt{3} - 1)^2} < 0.264
\] (8.15)

As for \( s_2, s_3 \) must be a Steiner point because it cannot be on the boundary of \( \mathbb{D} \). Let \( x \) denote a vertex (either a Steiner point or Rubin point) adjacent to \( s_3 \) as shown in Figure 8.7. Then \( S \) is clearly self-intersecting.

Figure 8.7: Edges \((n_q,s_1),(s_1,s_2)\) and \((s_2,s_3)\) are drawn at their maximum length. \( S \) must be self-intersecting (Lemma 8.6.1).

Case (4). One size-2 FST and one size> 2 FST: Refer to Figure 8.8 for artefacts relevant
to this case. Let $S_1$ denote the size-2 FST with $T(S_1) = \{n, r_m\}$. Let $S_2$ denote the size $> 2$ FST with $T(S_2) = \{n_q, r_1, \ldots, r_{m-1}\}$. Let $\phi_{1a} = \angle r_m n_q r_1$, $\phi_{1b} = \angle r_{m-1} n_q r_m$ and $\phi_2 = \angle r_1 n_q r_{m-1}$.

![Diagram](image)

Figure 8.8: Artefacts relevant to the case of one size-2 FST and one size $> 2$ FST (Lemma 8.6.1).

Observe $\phi_{1a} \geq \frac{\pi}{3}$ and $\phi_{1b} \geq \frac{\pi}{3}$ otherwise $r_1$ or $r_{m-1}$ are inside the Lune on $(n_q, r_m)$ implying $S$ is not minimal. Observe $S_2$ is an MST such that $T(S_2) = \{n_q\} \cup R \setminus \{r_m\}$ and that $n_q$ is degree-1 in $S_2$. Let $T_2$ denote a minimum spanning tree on $R \setminus \{r_m\}$ and let $U_2$ denote the edges of spanning tree $U$ between $R \setminus \{r_m\}$. It is a trivial fact that $L_{U_2} \leq L_{T_2}$.

From Theorem 7.5.1

$$L_{S_2} - L_{T_2} > 2 - \sqrt{3} \quad (8.16)$$

The edges of $T_2$ can be assumed to correspond to the chords $r_1 r_2, r_2 r_3, \ldots, r_{m-2} r_{m-1}$ because of the following:

\[\text{Recall that in this thesis the convention with respect to angles given in the form } \angle abc \text{ is that they can be positive or negative and are measured in an anti-clockwise direction.}\]
1. The non-bridge in $S_2$ corresponds to $\phi_{1a} + \phi_{1b} \geq \frac{2\pi}{3}$ implying the length of the chord corresponding to the non-bridge $|r_1r_{m-1}| \geq \sqrt{3}$.

2. From Lemma 6.13.7 it is known that no bridge in $S_2$ can span two Rubin points farther than $\sqrt{3}$ apart.

3. It follows from 1 and 2 that $r_1r_{m-1}$ is a longest chord, and therefore there exists a $T_2$ such that $r_1r_{m-1}$ corresponds to the gap in $T_2$.

Now consider $R \cup \{n_q\}$. $F$ is a forest on $R$ such that $F$ has two components and $L_F$ is minimal. Observe that the edges of $T_2$ constitute a forest on $R$ with two components and it follows:

$$L_F \leq L_{T_2} \tag{8.17}$$

Observe:

$$L_{S_2} = L_S - L_{S_1} = L_S - 1 \tag{8.18}$$

Substitute for $L_{T_2}$ and $L_{S_2}$ from 8.17 and 8.18 respectively into 8.16

$$L_S - L_F > 3 - \sqrt{3} \tag{8.19}$$

It follows from 8.19 and from Corollary 8.5.2 that $R \neq R^B$ for this case.

\section*{8.7 Two or three Rubin points}

**Lemma 8.7.1.** $R^B_2 \neq R^B$

**Proof.** There is only one topology that is possible for two Rubin points, that is, two size-2 FSTs and by Lemma 8.6.1, $R \neq R^B$ and the lemma follows.  

**Lemma 8.7.2.** $R^B_3 \neq R^B$

**Proof.** There are only two possible topologies for $S$ shown in Figure 8.9 with three size-2 FSTs on the left and on the right, one size-2 and one size-3 FST.

Since both topologies include size-2 FSTs, by Lemma 8.6.1 $R^B_3 \neq R^B$.  

Corollary 8.7.3. It follows from Lemmas 8.7.1 and 8.7.2 that $R^B$ has no fewer than four Rubin points.

8.8 Four Rubin points

Lemma 8.8.1. If $R^B = R^B$ then $\inf (L_S - L_F) = \sqrt{2}(\sqrt{3} - 1)$.

Proof. The three possible topologies for this case are shown in Figure 8.10.

Figure 8.10: The three possible topologies for $S$ with four Rubin points (Lemma 8.8.1).

Since the first and second topologies each have a size-2 FST, then by Lemma 8.6.1, $R \neq R^B$. Observe the third topology in Figure 8.10 corresponds to the benchmark shown.
in Lemma 8.5.1 with regularly spaced Rubin points and for which $L_S - L_F = \sqrt{2}(\sqrt{3} - 1)$. It suffices now prove that if $S$ for $R^B_4$ comprises two size-3 FSTs, then $R^B_4$ comprises four regularly spaced Rubin points.

Select any $R_4$ for which $S$ has a topology comprising two size-3 FSTs. A deformation will be described, which, when applied no more than three times, transforms $R_4$ into a set of four regularly spaced Rubin points. Furthermore, a proof will be provided that each deformation yields a monotonic decrease in $L_S - L_F$, and the lemma follows.

Observe that the length of a chord between two Rubin points is determined by the relative positions of the Rubin points on the boundary of $D$. Accordingly the deformation is described in terms of the movement of the Rubin points relative to each other, at each end of a chord. A deformation consists of simultaneous expansions and contractions of shortest and longest chords until the stopping condition is met:

**Expansion**: Move the Rubin points corresponding to each shortest chord away from each other such that their relative speed on the boundary of $D$ is $x$.

**Contraction**: Move the Rubin points corresponding to each longest chord towards each other such that their relative speed on the boundary of $D$ is $y$. Speed $y$ is found by multiplying speed $x$ by the number of shortest chords and dividing by the number of longest chords.

**Stopping condition**:

- One or more chords that are neither shortest nor longest change to either shortest or longest, or;
- all chords are shortest or longest and all change to be both shortest and longest.

Observe that the deformation as defined ensures that no chords other than longest and shortest chords change length. Accordingly, when a chord is newly categorised as shortest or longest it is because of a change in the category length, rather than because of a change in the chord’s length. Furthermore, observe that during the deformations the assignment of edges in $F$ need not change.

Deformations are applied repeatedly (but no more than three times) until all chords are both shortest and longest chords (that is, all the same length). Figure 8.11 shows all possible configurations of longest and shortest chords prior to and after deformations,
assuming the maximum number of deformations is required from any starting point to reach \textit{BBBB}. It is also possible for a lesser number of deformations to be required. For example, a deformation could start at \textit{S} \textit{−−} \textit{L} and end at \textit{SSSL} or \textit{BBBB}.

![Diagram of possible starting and stopping conditions for each deformation.](image)

**Figure 8.11:** Possible starting and stopping conditions for each deformation, where "L" denotes a longest chord; "S" denotes a shortest chord; "−" denotes neither longest or shortest and "B" denotes both longest and shortest.

A proof that each deformation gives a monotonic decrease in $L_S - L_F$ follows. Let $b$ denote a bridge in $S$ and $n$ denote a non-bridge in $S$. Let $E$ denote an edge in $F$ and $G$ denote a gap in $F$. Then a chord can be characterised by the four possible combinations of its corresponding $S$ and $F$ artefacts, namely $bE$, $bG$, $nE$ or $nG$. Consider the factors governing the order in which chords are arranged around the boundary of $D$:

- Bridges and non-bridges in $S$ must alternate and there are two of each.
- Edges and gaps in $F$ can appear in any order provided there are exactly two of each.

In the sequel, inequalities for the derivatives for each chord in the deformation will be derived separately, and then the sum the derivatives for the possible configurations of longest and shortest chords will be examined. Let $\dot{X}(\cdot)$ and $\dot{C}(\cdot)$ denote the derivatives for an expansion and a contraction respectively of a chord, where corresponding $S$ and $F$ artefacts are included in the parentheses. Figure 8.12 provides graphic definitions for various angles of interest in the calculation of derivatives for the four combinations of $S$ and $F$ artefacts.

**Expansion/Contraction of a $bE$ chord**
\( \dot{X}(bE) = 2 \cos \theta_4 - 2 \cos \theta_3 \)  
\[ (8.20) \]

The second term in Equation 8.20 is straightforward. The first term can be derived from Equation 7.19 as follows:

First, in order to distinguish between overlapping notation in Equations 7.19 and 8.20, a is appended to each subscript in the former:

\( \left( \sin \theta_{2a} - \sqrt{3} \cos \theta_{2a} \right) \cos \theta_{4a} \)  
\[ (8.21) \]

Both are derivatives for the same perturbation though reversed and the latter is a special case of the former in which \( \theta_{4a} = 0 \). Commence by negating the term 8.21 and substituting \( \theta_{4a} = 0 \):
\[ \sqrt{3} \cos \theta_{2a} - \sin \theta_{2a} \quad (8.22) \]

Observe:

\[ \theta_{2a} = \theta_4 - \frac{\pi}{6} \quad (8.23) \]

Substitute for \( \theta_{2a} \) from 8.23 into 8.22:

\[ \sqrt{3} \cos \left( \theta_4 - \frac{\pi}{6} \right) - \sin \left( \theta_4 - \frac{\pi}{6} \right) \quad (8.24) \]

Substitute with angle sum and difference identities:

\[ \sqrt{3} \left( \frac{\sqrt{3}}{2} \cos \theta_4 \cos \frac{\pi}{6} + \sin \theta_4 \sin \frac{\pi}{6} \right) - \sin \theta_4 \cos \frac{\pi}{6} + \cos \theta_4 \sin \frac{\pi}{6} \quad (8.25) \]

Substitute \( \cos \frac{\pi}{6} = \sqrt{3} \) and \( \sin \frac{\pi}{6} = \frac{1}{2} \) into 8.25:

\[ \sqrt{3} \left( \frac{\sqrt{3}}{2} \cos \theta_4 + \frac{1}{2} \sin \theta_4 \right) - \frac{\sqrt{3}}{2} \sin \theta_4 + \frac{1}{2} \cos \theta_4 \]

\[ = \frac{3}{2} \cos \theta_4 + \frac{\sqrt{3}}{2} \sin \theta_4 - \frac{\sqrt{3}}{2} \sin \theta_4 + \frac{1}{2} \cos \theta_4 = 2 \cos \theta_4 + 1 \quad (8.26) \]

Returning now to Equation 8.20 observe \( \theta_4 > \theta_3 \) and it follows:

\[ \dot{X}(bE) < 0 \quad (8.27) \]

Observe that the derivative for the contraction of a \( bE \) chord is simply the negation of the derivative for the expansion and the same applies to the other chord types.

\[ \dot{C}(bE) = 2 \cos \theta_3 - 2 \cos \theta_4 \implies \dot{C}(bE) > 0 \quad (8.28) \]

Observe:

\[ \theta_4 = \theta_3 + \frac{\pi}{6} \quad (8.29) \]

Further observe that both angles increase with an increase in chord length and that
\( \dot{C}(bE) \) operates on a longer chord than \( \dot{X}(bE) \) implying:

\[
|\dot{X}(bE)| > |\dot{C}(bE)|
\]  

(8.30)

**Expansion/Contraction of a \( bG \) chord**

\[
\dot{X}(bG) = 2 \cos \theta_1
\]  

(8.31)

It follows:

\[
\dot{X}(bG) > 0
\]  

(8.32)

\[
\dot{C}(bG) = -2 \cos \theta_1
\]  

(8.33)

\[
\dot{X}(bG) < 0
\]  

(8.34)

Observe that \( \dot{C}(bG) \) operates on a longer chord than \( \dot{X}(bG) \) implying:

\[
|\dot{X}(bG)| > |\dot{C}(bG)|
\]  

(8.35)

For the remaining chord types, consideration of the change in length of \( S \) is omitted, since an expansion to a chord corresponding to a non-bridge in \( S \) does not directly lead to a change in the length of an FST of \( S \).

**Expansion/Contraction of a \( nE \) chord**

\[
\dot{X}(nE) = -2 \cos \theta_2
\]  

(8.36)

It follows:

\[
\dot{X}(nE) < 0
\]  

(8.37)

\[
\dot{C}(nE) = 2 \cos \theta_2
\]  

(8.38)
\[ \dot{C}(nE) > 0 \] (8.39)

Observe that \( \dot{C}(nE) \) operates on a longer chord than \( \dot{X}(nE) \), implying:

\[ |\dot{X}(nE)| > |\dot{C}(nE)| \] (8.40)

**Expansion/Contraction of an nG chord**

\[ \dot{X}(nG) = \dot{C}(nG) = 0 \] (8.41)

Observe due to basic geometry:

\[ |\dot{X}(nG)| < |\dot{X}(bE)| < |\dot{X}(bG)| < |\dot{X}(nE)| \] (8.42)

\[ |\dot{C}(nG)| < |\dot{C}(bE)| < |\dot{C}(bG)| < |\dot{C}(nE)| \] (8.43)

Now consider the four possible orders of chord types by length. Note that shorter chords are associated with edges (E) in \( F \) and bridges (b) and non-bridges (n) in \( S \) can appear anywhere in this length order:

\[ |bE| \leq |bE| \leq |nG| \leq |nG| \]

\[ |bE| \leq |nE| \leq |bG| \leq |nG| \]

\[ |nE| \leq |nE| \leq |bG| \leq |bG| \]

\[ |nE| \leq |bE| \leq |nG| \leq |bG| \]
In Table 8.2, the derivatives for expansions and contractions for the four possible orders by length are given for the six possible configurations of longest and shortest chords prior to a deformation (that is, the six nodes in Figure 8.11 that have incident tails of arcs). The table also includes justification for the claim that \( \dot{L}_S - \dot{L}_F < 0 \), including reference to the following five notes:

**Note (1).** \(|\dot{X}(bG)| < |\dot{X}(nE)|\) due to Inequality 8.42 and it follows that the sum of derivatives < 0.

**Note (2).** \(|\dot{X}(bE)| > |\dot{C}(bE)|\) due to Inequality 8.30 and it follows that the sum of derivatives < 0.

**Note (3).** Observe:

\[
\dot{L}_S - \dot{L}_F = \dot{X}(bE) + \frac{\dot{C}(nE)}{3} + \frac{\dot{C}(bG)}{3} + \frac{\dot{C}(nG)}{3} \tag{8.44}
\]

Refer to Figure 8.12 for graphical definition of angles relevant to each case and substitute into Equation 8.44 as follows:

- From Equation 8.20 for \( \dot{X}(bE) \)
- From Equation 8.38 for \( \dot{C}(nE) \)
- From Equation 8.33 for \( \dot{C}(bG) \)
- From Equation 8.41 for \( \dot{C}(nG) \)

\[
\dot{L}_S - \dot{L}_F = 2 \cos \theta_4 - 2 \cos \theta_2 + \frac{2 \cos \theta_2}{3} + \frac{-\cos \theta_1}{3} + \frac{0}{3} \tag{8.45}
\]

Note that because of the case conditions:

\[
\theta_3 < \theta_2 \tag{8.46}
\]

\[
\theta_4 < \theta_1 \tag{8.47}
\]

Because the chord lengths associated with \( \theta_1 \) and \( \theta_2 \) are the same:

\[
\theta_1 = \theta_2 + \frac{\pi}{6} \tag{8.48}
\]
Substitute for $\theta_1$ and for $\theta_4$ from 8.29 and 8.48 respectively into 8.45:

$$L_S - L_F = 2 \cos \left( \theta_3 + \frac{\pi}{6} \right) - 2 \cos \theta_3 + \frac{2 \cos \theta_2}{3} - \frac{2 \cos \left( \theta_2 + \frac{\pi}{6} \right)}{3}$$ \hspace{1cm} (8.49)

Observe:

$$\theta_3 < \theta_2$$ \hspace{1cm} (8.50)

Substitute for $\theta_2$ from 8.50 into 8.49:

$$L_S - L_F < 2 \cos \left( \theta_3 + \frac{\pi}{6} \right) - 2 \cos \theta_3 + \frac{2 \cos \theta_3}{3} - \frac{2 \cos \left( \theta_3 + \frac{\pi}{6} \right)}{3}$$

$$\Rightarrow \ L_S - L_F < \frac{4 \cos \left( \theta_3 + \frac{\pi}{6} \right) - 4 \cos \theta_3}{3}$$ \hspace{1cm} (8.51)

Since $\cos \left( \theta_3 + \frac{\pi}{6} \right) < \cos \theta_3$:

$$L_S - L_F < 0$$ \hspace{1cm} (8.52)

**Note (4).** $|\dot{X}(nE)| < |\dot{C}(nE)|$ due to Inequality 8.40 and it follows that the sum of the derivatives is < 0.

**Note (5).** $|\dot{C}(bE)| < |\dot{C}(bG)|$ due to Inequality 8.43 and it follows that the sum of the derivatives is < 0.
Table 8.2: Derivatives for expansions and contractions.

<table>
<thead>
<tr>
<th>Case</th>
<th>Expansion</th>
<th>Contraction</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>S-S-L</td>
<td>$</td>
<td>bE</td>
<td>&lt;</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>&lt;</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>dE</td>
<td>&lt;</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>&lt;</td>
</tr>
</tbody>
</table>

SS-L

<table>
<thead>
<tr>
<th>Case</th>
<th>Expansion</th>
<th>Contraction</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$</td>
<td>bE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>dE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>=</td>
</tr>
</tbody>
</table>

SSS

<table>
<thead>
<tr>
<th>Case</th>
<th>Expansion</th>
<th>Contraction</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$</td>
<td>bE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>=</td>
</tr>
</tbody>
</table>

SSLL

<table>
<thead>
<tr>
<th>Case</th>
<th>Expansion</th>
<th>Contraction</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$</td>
<td>bE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>=</td>
</tr>
</tbody>
</table>

SLLL

<table>
<thead>
<tr>
<th>Case</th>
<th>Expansion</th>
<th>Contraction</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$</td>
<td>bE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>=</td>
</tr>
<tr>
<td></td>
<td>$</td>
<td>nE</td>
<td>=</td>
</tr>
</tbody>
</table>

See note 1

All derivatives ≤ 0
8.9 Five or more Rubin points

Remark 8.9.1. If \( R \) has two or more longest chords then the two gaps in \( F \) correspond to any two longest chords, and; if \( R \) has exactly one longest chord, then the two gaps in \( F \) correspond the longest chord and any 2\textsuperscript{nd} longest chord.

Lemma 8.9.1. If there exists an \( R = R^B \) with exactly one longest chord, then the chords adjacent to the longest chord are 2\textsuperscript{nd} longest chords.

Proof. Implicit in this lemma is that \( m \geq 4 \) since it has already been shown in Corollary 8.7.3 that \( R^B \) has no fewer than four Rubin points. If there exists an \( R = R^B \) with exactly one longest chord, then one gap of every \( F \) for \( R \) corresponds to this longest chord, and the other gap in an \( F \) for \( R \) corresponds to a 2\textsuperscript{nd} longest chord. Denote the length of a 2\textsuperscript{nd} longest chord as \( c_2 \). Suppose that contrary to the lemma that a chord adjacent to the longest chord has length less than \( c_2 \). Then an edge of \( F \) must correspond to this chord. Refer to Figure 8.13.

Let the line segment \( \overline{r_1r_2} \) be the Simpson line \( \overline{S} \) (from the application of the Melzak-Hwang Algorithm to \( S \) - see Section 6.2.1). Recall from Lemma 6.2.2 that \( |er_2| = L_S \) and for any sufficiently small perturbation of terminals, the equality holds (Lemma 6.2.3). Let \( \overline{r_1r_2} \) denote the chord with length less than \( c_2 \), and let \( \overline{r_2r_3} \) denote the longest chord. Now consider a sufficiently small perturbation of \( r_2 \) towards \( r_3 \) on the boundary of \( D \). The derivative for the perturbation is:

\[ \dot{L}_S - \dot{L}_F = \cos \theta_2 - \cos \theta_1 \] \hspace{1cm} (8.53)

Observe that \( \theta_2 > \theta_1 \) and it follows:

\[ \dot{L}_S - \dot{L}_F < 0 \] \hspace{1cm} (8.54)

After a finite number of sufficiently small perturbations, each of which decreases \( \dot{L}_S - \dot{L}_F \) (thereby proving that \( R \) before each perturbation is not equal to \( R^B \)), either:

1. \( |r_1r_2| \) will increase in length to \( c_2 \), or;
2. \( |r_2r_3| \) will decrease in length to \( c_2 \), or;
3. Both 1 and 2 will occur simultaneously.

The first outcome contradicts the contrary supposition that $|r_1r_2| < c_2$. The second outcome gives a perturbed $R'$ in which there is more than one longest chord (since the longest chord now has length $c_2$, as does at least one other chord), so the lemma conditions are not met.

Corollary 8.9.2. $R^B$ has at least two $2^{nd}$ longest chords, or at least two longest chords.

Lemma 8.9.3. The length of a $2^{nd}$ longest chord in $R^B$ is $c_2 \leq \sqrt{3}$.

Proof. Recall Lemma 6.13.7 (No bridge in $S$ can span two Rubin points farther than $\sqrt{3}$ apart). Now consider these three cases:

Case (1). Exactly one longest chord and it corresponds to a bridge in $S$: Then it follows from Lemma 6.13.7 that a longest chord has length less than $\sqrt{3}$ and as a consequence a $2^{nd}$
longest chord is also shorter than $\sqrt{3}$.

**Case (2).** Exactly one longest chord and it corresponds to a non-bridge in $S$: Then the longest chord is adjacent to a bridge and by Lemma 8.9.1 the bridge corresponds to a 2nd longest chord. As a consequence of Lemma 6.13.7 a 2nd longest chord has length less than $\sqrt{3}$.

**Case (3).** Two or more longest chords: Then there must be at least one longest chord or second longest chord corresponding to a bridge in $S$ and since chords that correspond to bridges in $S$ have length less than $\sqrt{3}$ (Lemma 6.13.7) the lemma follows in this case.

**Lemma 8.9.4.** If an $S$ for $R \cup \{n_q\}$ comprises two FSTs, and if an $R$-cherry is adjacent to middling or shortest chords, then $R \neq R^B$.

**Proof.** Implicit in this lemma is that $m \geq 4$ since cases involving size-2 FSTs can be disregarded (Lemma 8.6.1). The remainder of this proof is similar to that for Lemma 7.3.3 but is adapted for $\mathcal{F}$ in this case, rather than $\mathcal{T}$. Consider an $R$-cherry in $S$ on Rubin points denoted $r_2$ and $r_3$ for some $R \cup \{n_q\}$ as depicted in Figure 8.14. The figure also shows elements of an $\tilde{S}$ (from the application of Melzak-Hwang Algorithm to $S$) and an $\mathcal{F}$ for $R$ of importance in this lemma. Note that only the edges of $\mathcal{F}$ that are affected by a perturbation described in the sequel are depicted in the figure. Recall from Lemma 6.2.2 that $L_{\tilde{S}} = L_S$ and for any sufficiently small perturbation of terminals, the equality holds (Lemma 6.2.3).

Observe that there are edges of $\mathcal{F}$ either side of $r_2$ and $r_3$, consistent with being shortest or middling chords (i.e. all shortest and middling chords correspond to edges in $\mathcal{F}$, consistent with the definition of $\mathcal{F}$). The chord $\overline{r_2r_3}$ could be a shortest, middling, 2nd longest or longest chord. There exists an $\mathcal{F}$ in which an edge corresponds to $\overline{r_2r_3}$ unless $R$ has two or fewer longest chords and one of these corresponds to $\overline{r_2r_3}$. In this case, a gap in $\mathcal{F}$ corresponds to $\overline{r_2r_3}$. Suppose contrary to the lemma that $R = R^B$ and $\overline{r_1r_2}$ and $\overline{r_3r_4}$ are shortest or middling chords and consider these two cases:

**Case (1).** An edge in $\mathcal{F}$ corresponds to $\overline{r_2r_3}$: Consider a perturbation of $r_2$ and $r_3$ towards each other on the boundary of $\mathbb{D}$ at equal speeds. Refer to Figure 8.14 for graphic definitions of angles $\theta_1$, $\theta_2$, $\theta_3$ and $\theta_4$. The derivative for $L_\mathcal{F}$ using a geometric approach is:
Figure 8.14: Illustration of the important elements of $\tilde{S}$ and $\tilde{F}$ for Lemma 8.9.4.

\[
\dot{L}_F = \cos \theta_1 - 2 \cos \theta_2 + \cos \theta_3 \quad (8.55)
\]

The derivative $\dot{L}_S$ of $L_S$ is the same as Equation 7.19 in Lemma 7.3.3:

\[
\dot{L}_S = (\sin \theta_2 - \sqrt{3} \cos \theta_2) \cos \theta_4 \quad (8.56)
\]

The derivative for $\dot{L}_S - \dot{L}_F$ follows from Equations 8.55 and 8.56:

\[
\dot{L}_S - \dot{L}_F = (\sin \theta_2 - \sqrt{3} \cos \theta_2) \cos \theta_4 - \cos \theta_1 + 2 \cos \theta_2 - \cos \theta_3 \quad (8.57)
\]

A minimum for $L_S - L_F$ must correspond the condition $\dot{L}_S - \dot{L}_F = 0$ for a critical point. If an upper bound for $L_S - L_F$ is less than zero over the defined domain, then there are no critical points for the domain. First choose values for $\theta_1$ and $\theta_3$ that will maximise $\dot{L}_S - \dot{L}_F$. Recall from Lemma 8.9.3 that a 2nd longest chord has length $c_2 \leq \sqrt{3}$. Since $r_1 r_2$ and $r_3 r_4$ are supposed to be shortest or middling chords, it follows that $0 < |r_1 r_2| \leq \sqrt{3}$.
and $0 < |r_3 r_4| \leq \sqrt{3}$. It further follows that $\theta_1 \leq \frac{\pi}{3}, \theta_3 \leq \frac{\pi}{3}$. Substitute $\cos \frac{\pi}{3} = \frac{1}{2}$ for $\theta_1$ and $\theta_3$ in Equation 8.57:

$$\dot{L}_B - \dot{L}_F \leq (\sin \theta_2 - \sqrt{3} \cos \theta_2) \cos \theta_4 - \frac{1}{2} + 2 \cos \theta_2 - \frac{1}{2} \tag{8.58}$$

Observe that the right-hand side of Inequality 8.58 is identical to the right-hand side of Inequality 7.21 in Lemma 7.3.3 and the same conclusion applies:

$$\dot{L}_S - \dot{L}_F < 0 \tag{8.59}$$

Inequality 8.59 contradicts that $R = R^B$ and the lemma follows for this case.

Case (2). A gap in $F$ corresponds to $r_3 r_4$. The same approach as for Case 1 is followed but with differences in the calculation of the derivative for $L_F$:

$$\dot{L}_F = \cos \theta_1 + \cos \theta_3 \tag{8.60}$$

The calculation of the derivative $\dot{L}_B$ of $L_B$ is same as for Case 1:

$$\dot{L}_B = (\sin \theta_2 - \sqrt{3} \cos \theta_2) \cos \theta_4 \tag{8.61}$$

The derivative for $L_S - L_T$ follows from Equations 8.60 and 8.61:

$$\dot{L}_B - \dot{L}_F = (\sin \theta_2 - \sqrt{3} \cos \theta_2) \cos \theta_4 - \cos \theta_1 - \cos \theta_3 \tag{8.62}$$

As for case 1 an upper bound for $\dot{L}_S - \dot{L}_F$ is sought and the same approach is applied with results in this case as follows:

$$\dot{L}_S - \dot{L}_F < \frac{\sqrt{3}}{2} (\sin \theta_2 - \sqrt{3} \cos \theta_2) - 1 \tag{8.63}$$

Since $\frac{\sqrt{3}}{2} (\sin \theta_2 - \sqrt{3} \cos \theta_2) \leq 0$:

$$\dot{L}_S - \dot{L}_F < 0 \tag{8.64}$$

Inequality 8.64 contradicts that $R = R^B$ and the lemma follows for this case.
Lemma 8.9.5. There exists an $R^B$ for which the chords corresponding to the two non-bridges in $S$ have equal length.

Proof. Let $\phi_1$ and $\phi_3$ denote the angle of the arcs spanned by each of the two FSTs on the boundary of $D$ and let $\phi_2$ and $\phi_4$ denote the angles of the arcs that correspond to the non-Bridges in $S$ as illustrated in Figure 8.15.

Figure 8.15: Angles of interest for the proof in Lemma 8.9.5. Chords shown in dashed red correspond to the non-Bridges in $S$. The FSTs shown in $S$ are examples for illustrative purposes.

Let $l_2$ and $l_4$ denote the lengths of the chords corresponding to $\phi_2$ and $\phi_4$ respectively. Observe that if one of the FSTs is rotated around $n_q$ then:

- the lengths of the chords corresponding to the non-bridges change.
- $L_S$ remains the same.
- $L_F$ changes only if edges in $F$ correspond to non-bridges.

Consider these three cases:
Case (1). There exists an $\mathbb{F}$ such that the two non-bridges in $\mathbb{S}$ correspond to edges in $\mathbb{F}$: For this case, consider this contrary supposition: There exists an $R^B$ for which the chords corresponding to the two non-bridges in $\mathbb{S}$ have unequal length. Without loss of generality, assume $l_2 < l_4$. Now perturb all the Rubin points of one of the FSTs in $\mathbb{S}$ around $n_q$ in a direction that will cause an increase in $l_2$ and a decrease in $l_4$. Observe that this deformation cannot give rise to a gap in $\mathbb{F}$ moving to either of the two non-bridges (such a change would only be a possibility if the longer of the two chords became longer). The derivative $\dot{L}_F$ for $L_F$ is as follows:

$$
\dot{L}_F = 2 \cos \frac{\phi_2}{2} - 2 \cos \frac{\phi_4}{2}
$$  (8.65)

Since $l_2 < l_4$, $\dot{L}_F > 0$. Observe that $\dot{L}_S = 0$ giving:

$$
\dot{L}_S - \dot{L}_F < 0
$$  (8.66)

Inequality 8.66 contradicts that $L_S - L_F$ is minimal for $R^B$ in this case.

Case (2). Conditions for Case 1 are not met and there exists a $\mathbb{F}$ such that one non-bridge in $\mathbb{S}$ corresponds to an edge in $\mathbb{F}$: Consider the same contrary supposition as for Case 1. We can assume that $l_4$ is longer than any edge in $\mathbb{F}$ since if it is not, then there exists an $\mathbb{F}$ for $R^B$ for which case 1 applies. Consider the same perturbation as for case 1. Since it can be assumed that $l_4 > l_2$, angle $\phi_2$ corresponds to an edge in $\mathbb{F}$ and the derivative for $\mathbb{F}$ follows:

$$
\dot{L}_F = 2 \cos \frac{\phi_2}{2} > 0
$$  (8.67)

As for case 1, we conclude that $\dot{L}_S - \dot{L}_F < 0$, contradicting that $L_S - L_F$ is minimal for $R^B$ in this case.

Case (3). Neither case 1 nor case 2 conditions are met: The case conditions imply that no $\mathbb{F}$s for $R^B$ contain edges corresponding to non-bridges. For this case, consider this contrary supposition: There does not exist an $R^B$ for which the chords corresponding to the two non-bridges in $\mathbb{S}$ have equal length. We can assume that $l_4$ and $l_2$ are longer than any edge
in $F$ since if they are not, then there exists a $F$ for which case 1 or 2 applies. Now consider the same perturbation as before. It is easy to see that $\dot{L}_S - \dot{L}_F = 0$ and accordingly the rotations of the FSTs relative to each other is arbitrary, and therefore there exists a rotation for which the chords corresponding to the two non-bridges in $S$ have equal length. In the event that such a rotation leads to an $S_1$ that is not minimal, then this contradicts that the set of Rubin points constitute an $R^B$. \qed
Chapter 9
Conclusion

This thesis addresses several topics relating to the planning of underground mines, with a focus on underlying mathematical models:

- **Determination of the transition from open pit to underground mining:** My new method includes three modifications to the normal model used for pit optimisation (solving for the maximum graph closure problem) to include detailed geometric and economic considerations of crown pillars. In the test case a 15% increase in cash value was achieved. My work on this topic is at technology readiness level\(^1\) (TRL) 5. TRL 6 is within reach. More widespread application and achievement of TRL 7+ will require some software development from one of the commercial mining software providers.

- **Underground mine with shaft access and with decline access:** My new decompositions are unique and appear to have some advantages compared to other decomposition approaches. In particular, my decompositions should provide the ability to support method selection effectively, using a single value-driven optimisation process. These decompositions also provide the industrial context for my research into solving the underlying PCEST problems. It is beyond the scope of my research to fully develop and test these decompositions, which I estimate are at TRL 2. The main bottleneck to taking this work to TRL 3+ is the further development of solutions to PCEST problems (see below).

- **PCEST problems:** PCEST problems are a generalisation of Euclidean Steiner tree problems. Prior to my research, only an approximation scheme existed. My re-

---

\(^1\)Refer to Appendix B: Technology readiness levels.
search is the first to provide an algorithmic framework for an exact solution. The strategy is to first try to reclassify possible terminals to ruled in or ruled out, using various ruling tests. Points that are ruled out are discarded. What remains is a set of ruled in terminals and a set of possible terminals. Then, new functions for FST generation and concatenation must be applied to complete the solution to a PCEST problem. These new functions are defined as adaptations of existing functions designed to solve the related Euclidean Steiner tree problem. Future work on this topic should focus on bringing Replacement argument B, full Steiner tree generation and concatenation to TRL 4. Once that is achieved, it will be possible to empirically test the efficiency of the solution on industrially relevant problems.
### A.1 Transition from open pit to underground mining

Table A.1: Symbols used in Chapter 3

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description (first usage or definition)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\beta, \gamma, \delta$</td>
<td>Sets of arcs $a_j^\beta, a_j^\gamma$ and $a_j^\delta$ respectively (page 29)</td>
</tr>
<tr>
<td>$\kappa$</td>
<td>A constant (page 32)</td>
</tr>
<tr>
<td>$A$</td>
<td>A set of arcs $a_k = (x_i, x_j)$ (page 25)</td>
</tr>
<tr>
<td>$C$</td>
<td>A set of opportunities $c_i$ (page 34)</td>
</tr>
<tr>
<td>$G$</td>
<td>A digraph with vertices $X$, arcs $A$ and vertex weights $M_X$ (page 25)</td>
</tr>
<tr>
<td>$G_Y$</td>
<td>A closure of $G$ with vertices $Y$, edges $A_Y$ and vertex weights $M_Y$ (page 25)</td>
</tr>
<tr>
<td>$M_X$</td>
<td>A set of vertex weights $m_i$, where $m_i$ corresponds to vertex $x_i$ (page 25)</td>
</tr>
<tr>
<td>$M_Y$</td>
<td>The sum of vertex values in a maximum graph closure of $G$ (page 26)</td>
</tr>
<tr>
<td>$M_T$</td>
<td>The sum of vertex values for the open pit and underground mines (page 32)</td>
</tr>
<tr>
<td>$X$</td>
<td>A set of vertices $x_i$ (page 25)</td>
</tr>
<tr>
<td>$X^p, X^u$</td>
<td>A set of vertices corresponding to the open pit and underground block models respectively (page 29)</td>
</tr>
</tbody>
</table>
### A.2 Prize collecting Euclidean Steiner tree

Table A.2: Abbreviations used in Chapters 6 to 8

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description (first usage or definition)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ANSI C</td>
<td>A standard for the C programming language defined by the American National Standards Institute (page 80)</td>
</tr>
<tr>
<td>FST</td>
<td>Full Steiner tree (page 67)</td>
</tr>
<tr>
<td>MST</td>
<td>Minimum spanning tree (page 66)</td>
</tr>
<tr>
<td>MStT</td>
<td>Minimum Steiner tree (page 66)</td>
</tr>
<tr>
<td>PCEST</td>
<td>Prize collecting Euclidean Steiner tree (page 72)</td>
</tr>
<tr>
<td>PCSTG</td>
<td>Prize collecting Steiner tree problem in graphs (page 72)</td>
</tr>
<tr>
<td>Symbol</td>
<td>Description (first usage or definition)</td>
</tr>
<tr>
<td>--------</td>
<td>-----------------------------------------</td>
</tr>
<tr>
<td>$D$</td>
<td>A disk centred on $n_q$ with a radius $r$, where $r + \mu$ is equal to the distance between $n_q$ and its nearest neighbour in $T(S^1)$ and $\mu$ is an infinitesimal distance (page 103)</td>
</tr>
<tr>
<td>$e$</td>
<td>The length of an edge that connects the two components of a forest $F^3$ (page 119)</td>
</tr>
<tr>
<td>$E(X)$</td>
<td>Edges of network $X$ (page 66)</td>
</tr>
<tr>
<td>$F$</td>
<td>A forest (page 101)</td>
</tr>
<tr>
<td>$F$</td>
<td>A forest on $R$ with edges corresponding to all the chords other than two longest chords (page 119)</td>
</tr>
<tr>
<td>$F^3$</td>
<td>The forest that results from replacing $S$ with $F$ in $S^2$ (page 119)</td>
</tr>
<tr>
<td>$L_X$</td>
<td>The length of network $X$ (page 85)</td>
</tr>
<tr>
<td>$m$</td>
<td>The cardinality of a set of Rubin points $R$ (page 105)</td>
</tr>
<tr>
<td>$N$</td>
<td>A set of points lying in the plane (page 66)</td>
</tr>
<tr>
<td>$n_0$</td>
<td>The mandatory terminal in a rooted NWGST problem (page 73)</td>
</tr>
<tr>
<td>$N_I$</td>
<td>The set of possible terminals $N_I \subseteq N$ (page 78)</td>
</tr>
<tr>
<td>$N_O$</td>
<td>The set of possible terminals $N_O \subseteq N$ (page 78)</td>
</tr>
<tr>
<td>$N_P$</td>
<td>The set of possible terminals $N_P \subseteq N$ (page 78)</td>
</tr>
<tr>
<td>$n_q$</td>
<td>A point in $N_P$ with weight $w_q$ that is to be subject to a ruling out test. (page 84)</td>
</tr>
<tr>
<td>$P$</td>
<td>A maximum PCEST (page 72)</td>
</tr>
<tr>
<td>$R$</td>
<td>The set of $r_i$ Rubin points, each of which is a point on an edge of $S^1$ where exactly one of the vertices of the edge is inside $D$ (page 103)</td>
</tr>
<tr>
<td>$R^A$</td>
<td>The smallest set of Rubin points that satisfies the following conditions: (a) $n_q$ has degree-1 in $S$, and (b) no additions, removals or perturbations of Rubin points can reduce $L_S - L_T$ without changing the degrees of $n_q$ in $S$. (page 139)</td>
</tr>
<tr>
<td>$R^B$</td>
<td>The smallest set of Rubin points that satisfies the following conditions: (a) $n_q$ has degree-2 or 3 in $S$, and (b) no additions, removals or perturbations of Rubin points can reduce $L_S - L_E$ (page 197)</td>
</tr>
<tr>
<td>Symbol</td>
<td>Description (first usage or definition)</td>
</tr>
<tr>
<td>--------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>$S$</td>
<td>A minimum Steiner tree (page 77)</td>
</tr>
<tr>
<td></td>
<td>The connected component of $S^2 \cap D$ containing $n_q$ the Rubin points, all the Steiner points inside $D$, and all the edges connecting these vertices inside $D$ (page 105)</td>
</tr>
<tr>
<td>$S^1$</td>
<td>A tree $S^1 = (V(S^1), E(S^1))$ is a minimum Steiner tree where $V(S^1) = T(S^1) \cup S(S^1)$ for $T(S^1) \subseteq N$ (page 102)</td>
</tr>
<tr>
<td>$S^2$</td>
<td>A tree $S^2$ results from inserting Rubin points $R$ into $S^1$ as new terminals (page 105)</td>
</tr>
<tr>
<td>$S(P)$</td>
<td>The set of Steiner points for $P$ (page 74)</td>
</tr>
<tr>
<td>$T$</td>
<td>A tree, defined in detail on each usage (page 66)</td>
</tr>
<tr>
<td>$T^4$</td>
<td>The tree resulting from the addition of an edge with length $e$ to $F^3$ (page 120)</td>
</tr>
<tr>
<td>$T$</td>
<td>A tree as follows: Its vertices are the Rubin points in $S^2$. Its edges are all the chords, other than a longest chord. (page 112)</td>
</tr>
<tr>
<td>$U$</td>
<td>a minimum Steiner tree with terminals consisting of all the Rubin points in $R$ (page 113)</td>
</tr>
<tr>
<td>$Uy$</td>
<td>An upper bound for a $y$. (page 119)</td>
</tr>
<tr>
<td>$\forall$</td>
<td>A spanning tree that connects $R$, not necessarily minimal in length. Denote the length of $\forall$ as $L_\forall$ (page 113)</td>
</tr>
<tr>
<td>$V(X)$</td>
<td>Vertices of a network $X$ (page 66)</td>
</tr>
<tr>
<td>$w_i$</td>
<td>A weight corresponding to point $n_i \in N$ (page 72)</td>
</tr>
<tr>
<td>$w_q$</td>
<td>The weight of point $n_q$ being subject to a ruling out test (page 84)</td>
</tr>
<tr>
<td>$W_N$</td>
<td>A set of weights $w_i$, each corresponding to point $n_i \in N$ (page 72)</td>
</tr>
</tbody>
</table>
Appendix B
Technology readiness levels

During the 1970s, the National Aeronautics and Space Administration (NASA) developed the concept of “technology readiness levels” to enable research, development and operational personnel to categorize and communicate the maturity of research and development in their portfolios (Mankins [109]). The approach has since been adopted by various industries, including some parts of the mining industry, for example the Cooperative Research Centre for Optimising Resource Extraction (CRC ORE) [110]. In order for me to be able to assign levels to my own contributions in this thesis, I have adopted the descriptors used by the European Commission [111].

Technology readiness levels:
1. Basic principles observed
2. Technology concept formulated
3. Experimental proof of concept
4. Technology validated in lab
5. Technology validated in relevant environment
6. Technology demonstrated in relevant environment
7. System prototype demonstration in operational environment
8. System complete and qualified
9. Actual system proven in operational environment
This page intentionally left blank.
Appendix C
Prototype ruling program information

Several of the ruling algorithms in Chapter 6 were prototyped in ANSI C. All were included in a single menu-driven program. Below is an extract from the software documentation, providing an overview of the functions and structure of the program.

****************
* PCEST RULING *
****************
Version 0.91
24 April 2019
Copyright: David Whittle, Melbourne

CONTENTS
=========
- Overview
- Main Menu
- Input File Formats

OVERVIEW
=========

The program name is WDG_PCEST(.exe on PC), which provides menu access to all of the ruling, reporting and utility functions.
On starting the program, the first thing that a user must do is provide a name for a report file (Suggested file name extension: .txt). On exiting the program, the report file will include records of all functions called during the session, including process details and results.

**MAIN MENU**

1. Open a points file
---------------------
Opens a points file and reads data into memory.

2. Generate a random points file and open it
-----------------------------------------------
Generates a set of random points for a user-specified parameters file and overwrites any data currently in memory.

3. Ruling in
------------
Implementation of Algorithm 1: Merging, and Algorithm 2: Ruling in.

4. MST ruling out
-----------------
Implementation of Algorithm 3: Rule out using an MST. It is presently limited to 200 ruled in and possible terminals. Note that this function runs ut_rule_out_neg_weight_points() first.

5. Replacement argument A
This is an implementation of Replacement Argument A, where it is certain that the point being tested can only have degree-1 in a PCEST. This is due to the fact that the only possible terminals tested are convex hull vertices (on the set of ruled in and possible terminals), where the angle of the convex hull at that point is less than 120 degrees. Note that this function runs ut_rule_out_neg_weight_points() first.

6. All ruling functions

Applies ruling functions in this order:
1. Ruling In
2. Replacement argument A
3. MST ruling out

7. Report on points in memory

Writes all point details and summary statistics to the Report file.

8. Save to a points file

Saves the points data in memory to a file with name specified by the user.

9. Utilities

The intention is to provide an easy mechanism to test
sub-functions in development and also to provide access to some lesser-used functions.

Utilities menu:
1. Sort on point_x
2. Sort on point_y
3. Sort on point_cat
4. Sort on point ids
5. Rule out zero and negative weight points
6. Merge two point files
7. Add a point
8. Calculate distances
9. Print distances
10. EXIT UTILITIES
10. EXIT
-------
Note that if point data in memory is not saved with the "Save to a point file" function prior to exit, the data is lost.

INPUT FILE FORMATS
==================
Parameters
---------
Only used by function "Generate a random points file and open it".
Suggested file name extension: .par
This is a text file with all values on one line separated by spaces.
nop lo_x hi_x lo_y hi_y lo_weight hi_weight mand_x mand_y

Where:

top
The number of points including the mandatory terminal (int > 0).

lo_x, hi_x, lo_y, hi_y
The lowest and highest values permitted for the x and y coordinates for the points (double).

lo_weight, hi_weight
The lowest and highest point weights (double).

mand_x, mand_y
Coordinates for a mandatory point (double).

Points
------
Used to store point information and point category.

Suggested file name extension: .pnt

This text file contains a line for each point with values separated by spaces:

point_ind point_weight point_x point_y point_cat

Where:
point_id
Point index (int)

point_weight
Point weight (double)

point_x
Point x coordinate (double)

point_y
Point y coordinate (double)

point_cat
Point category (0 <= int <= 2: 0 = ruled out, 1 = possible, 2 = ruled in)

In the case of the Rooted PCEST problem, the mandatory point is point_id = 0. Its weight will be ignored.
Bibliography


Minerva Access is the Institutional Repository of The University of Melbourne

Author/s: Whittle, David

Title: Underground mine plan optimisation

Date: 2019

Persistent Link: http://hdl.handle.net/11343/228864

File Description: Final thesis file

Terms and Conditions: Copyright in works deposited in Minerva Access is retained by the copyright owner. The work may not be altered without permission from the copyright owner. Readers may only download, print and save electronic copies of whole works for their own personal non-commercial use. Any use that exceeds these limits requires permission from the copyright owner. Attribution is essential when quoting or paraphrasing from these works.