AN OPTIMIZATION-BASED DECOMPOSITION HEURISTIC
FOR SOLVING COMPLEX UNDERGROUND MINE
SCHEDULING PROBLEMS

by

Dónal O’Sullivan
A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Doctor of Philosophy (Mineral and Energy Economics).

Golden, Colorado
Date ____________________

Signed: ____________________
Dónal O’Sullivan

Signed: ____________________
Dr. Alexandra M. Newman
Thesis Advisor

Golden, Colorado
Date ____________________

Signed: ____________________
Dr. Roderick G. Eggert
Professor and Division Director
Division of Economics & Business
ABSTRACT

Underground mine production scheduling possesses mathematical structure similar to and yields many of the same challenges as general scheduling problems. That is, binary variables represent the time at which various activities are scheduled. Typical objectives seek to minimize costs or some measure of production time, or to maximize net present value; two principal types of constraints exist: (i) resource constraints, which limit the number of activities committed to a time period based on the availability of a given supply and on the amount of that supply required to perform the activity, and (ii) precedence constraints, which dictate the order in which activities must be completed. In our setting, we maximize “discounted metal production” for the remaining life of an underground lead and zinc mine that uses three different underground methods to extract the ore. Resource constraints limit the grade, tonnage, and backfill paste (used for structural stability) in each time period, while precedence constraints enforce the sequence in which extraction (and backfill) is performed in accordance with the underground mining methods used. We tailor existing exact and heuristic approaches to reduce model size, and develop an optimization-based decomposition heuristic; both of these methods transform a computationally intractable problem to one for which we obtain solutions in seconds, or, at most, hours for problem instances based on data sets from the Lisheen mine near Thurles, Ireland. Our solution adds value to the Lisheen mining operation by: (i) shifting metal production forward in the schedule; (ii) reducing waste mining and backfilling delays; (iii) avoiding expensive mill-halting drops in ore production; and (iv) enabling smoother workforce management. Our modeling approach could be applied to other mines, especially to operations with flat lying deposits that practice retreat, i.e., room-and-pillar, mining, such as coal mines, and to mines that are approaching the end of their operational life.
TABLE OF CONTENTS

ABSTRACT ................................................................. iii
LIST OF FIGURES .......................................................... vii
LIST OF TABLES ........................................................... xiii
ACKNOWLEDGMENTS ......................................................... xiv

CHAPTER 1 INTRODUCTION ............................................... 1
  1.1 Background .......................................................... 2
  1.2 Mine Production Schedule Optimization .......................... 5
  1.3 Literature Review ..................................................... 6
  1.4 Original Contribution ................................................. 7
  1.5 Dissertation Organization ........................................... 9

CHAPTER 2 TRACTABILITY OF INTEGER PROGRAMS FOR OPEN PIT
AND UNDERGROUND MINE PRODUCTION SCHEDULING
OPTIMIZATION .......................................................... 11
  2.1 Integer Programming Review ..................................... 11
  2.2 Integer Program Mine Design and Scheduling Models .......... 12
  2.3 Tractability of the \( UPIT \) and \( CPIT \) Problems ................ 15
  2.4 Mine Structure and Production Schedule Optimization .......... 16
    2.4.1 Precedence Constraint Structure .............................. 16
    2.4.2 Problem Decomposition ...................................... 18
  2.5 Ore Block Characteristics and Aggregation Heuristics ........... 20
    2.5.1 Homogeneous Block Size .................................... 20
2.5.2 Aggregation of Ore Blocks ........................................ 22

2.6 Conclusion ......................................................... 23

CHAPTER 3 AN OPTIMIZATION-BASED DECOMPOSITION HEURISTIC METHOD FOR SOLVING COMPLEX UNDERGROUND MINE SCHEDULING PROBLEMS ........................................ 24

3.1 Production at the Lisheen Mine ..................................... 25

3.2 Precedence Framework and Model .............................. 28

3.2.1 Mathematical Formulation .................................... 29

3.3 Preprocessing and Other Exact Techniques to Improve Tractability .... 34

3.3.1 Early Start Algorithm for Activities in Panels .......... 36

3.3.2 Tightened Precedence Constraints ......................... 39

3.4 Heuristic Solution Methods ..................................... 40

3.4.1 Early Start Determination for Haulage Pillars .......... 40

3.4.2 Aggregation .................................................... 41

3.4.3 Optimization-Based Decomposition Heuristic .......... 42

3.5 Computational Results .......................................... 47

3.6 Conclusions ...................................................... 51

CHAPTER 4 APPLICATION OF AN OPTIMIZATION-BASED HEURISTIC TO SCHEDULE EXTRACTION AND BACKFILL ACTIVITIES FOR A COMPLEX UNDERGROUND MINE ........................................ 54

4.1 Production Scheduling at Lisheen Mine ...................... 60

4.2 Integer-Programming Approach ................................. 65

4.3 Implementation .................................................. 72

4.4 Results .......................................................... 74

4.5 Scenario Analysis ............................................... 77
4.6 Insights ................................................................. 80
4.7 Conclusions .......................................................... 82
4.8 APPENDIX A - Model Formulation .............................. 83
CHAPTER 5 CONCLUSIONS ............................................. 88
5.1 Research Contributions .............................................. 88
5.2 Suggested Further Research ........................................ 89
REFERENCES CITED ...................................................... 91
Figure 1.1 Surface mining (left) can be used when ore is located close to the earth’s surface. Overburden (i.e., waste) must be removed before extraction can begin. Haul roads wind up from the bottom of the pit to the surface. The small footprint of the facilities at the Lisheen underground mine in Ireland (right) provides little evidence with which to judge the size of the mine. Below ground, trucks haul ore throughout miles of tunnels. Crushed ore is conveyed to the surface for refinement at the mill. Tailings (i.e., waste produced by the mill) are disposed of in a tailings pond. ................................................................. 3

Figure 1.2 There are five stages in the life cycle of a mine. Some stages overlap others, for example, production can begin once access to the ore body has been developed, while development and exploration continue in other parts of the mine. Mining ventures may wish to begin production as early as possible to generate a positive cash flow that can be used to service debt. ................................................................. 5

Figure 2.1 The “plus sign” convention for open pit block sequencing is shown on the right. Blocks 2–6 must be mined prior to mining block 1. ............ 16

Figure 2.2 We show a conceptual illustration of precedence rules at the Lisheen underground mine. In Box (a), we render a panel of ore blocks adjacent to a haulage route comprised of ore containing haulage pillars. Within the panel, strict precedence rules, similar to constraints (2.1) above, dictate that Block A must be taken before Block B and Block B before Block C, i.e., it is impossible to take Block C without taking both Block A and Block B in advance. Precedence rules of this type will have underlying network structure, i.e., by representing the blocks as nodes, we define arcs between pairs of nodes to represent the precedence rules. Extraction of pillars follows a sequence that retreats toward the exit. However, extraction of a pillar is optional; its extraction does not require or facilitate the extraction of any other block. In addition, as Box (b) illustrates, once a haulage pillar (Pillar 2) is extracted, the roof caves-in, blocking access to pillars (Pillar 1) and blocks (A, B, and C) upstream of the extracted pillar. Consequently, this precedence rule governs an “or” decision that cannot be incorporated into a network model, e.g., we can extract Pillar 1, or Pillar 2, or both. .............. 18
Figure 2.3 We illustrate the effect of complex precedence rules on tractability. For the Lisheen mine, if we exclude haulage pillars from the scheduling problem, we can solve a 52-week instance of the model in a matter of seconds. As we introduce haulage pillars, complexity and solution time increase (in an approximately exponential progression) until problem instances are no longer tractable.

Figure 2.4 At weekly fidelity, the total scheduling gap between Block A, Block B, and Block C is much smaller than at bi-weekly fidelity. These gaps unnecessarily extend the start times for extraction of dependent blocks. While Block C can start during week 3 with weekly fidelity, at bi-weekly fidelity, the same block must wait until period 2 (week 4) before it can start.

Figure 3.1 On the left, we illustrate a representative block model. The “plus sign” convention for open pit block sequencing is shown on the right. Blocks 2–6 must be mined prior to mining block 1.

Figure 3.2 The Lisheen mine employs three underground mining methods.

Figure 3.3 Cut-off grade selection determines ore block shape and size. Box (a) shows part of the ore body with a nonuniform distribution of metal. A cut-off grade of 7% results in a large area classified as ore (see Box (b)) that can partitioned into homogeneous blocks and extracted with a single mining method, e.g., drift-and-fill. At a cut-off grade of 9%, the material classified as ore diminishes (Box (c)). Drift-and-fill mining would now excavate too much waste rock so a different mining approach is used. Box (d) shows material classified as ore at a cut-off grade of 12%. With most of the area now defined as waste, a targeted mining method is suitable.

Figure 3.4 The arrows show the haulage routes through an ore body. We highlight a mining panel adjacent to a haulage route. The pillars that support the haulage route are shown as P1 - P5. The arrows indicate the direction of the mine exit. The haulage route collapses west of any pillar that is removed. Hence, the extraction precedence for these pillars would require P1 be taken first and P5 last. Removal of pillar P3, P4, or P5 would result in the destruction of the haulage path in that area, eliminating a means to access the panel.
Figure 3.5  For this panel, we show in Box (a) that Panelblocks A-F must be mined sequentially; this is also true for L-O. In addition, any panelblock from A-F can be mined concurrently with any panelblock from L-O. However, Panelblocks X-Z must wait until all other panelblocks have been extracted and backfilled (Box (b)). To represent this, we define two ordered sets here: \{A, B, C, D, E, F, BF1, X, Y, Z\} and \{L, M, N, O, BF2, X, Y, Z\}. Both of these sets are linked to Pillar 1, and once that pillar is extracted, the haulage route above it will no longer be accessible.

Figure 3.6  We illustrate the precedence rules that are included in our formulation. In Box (a), constraints (3.5) ensure that Panelblock C can only be extracted if Panelblocks A and B are extracted in advance. Constraints (3.7) ensure that haulage pillars furthest from the mine exit are taken before those that are closer. Box (b) shows that Pillar 2 has been extracted and the roof allowed to cave in; consequently, Pillar 1 can no longer be accessed. Also, the collapsed roof now blocks access to the panel, illustrating the need for constraints (3.8).

Figure 3.7  The nodes 1, 2, 3, and 4 represent activities that must be completed in order to finish the project, i.e., the activity represented by node 4. The directed arcs between the nodes represent the precedence rules, and costs on the arcs represent the duration of the activity associated with the node from which the arc originates. There are three paths to node 4: S-2-4, S-1-3-4, and S-2-3-4, with total durations 15, 5, and 17, respectively. The early start time for node 4 is defined by its longest path: \(\text{max}\{15, 5, 17\} = 17\). Therefore, the project cannot be finished before 17+7=24.

Figure 3.8  When an activity has more than one precedent, it will be a member of more than one set. We illustrate this with a network formed by two ordered precedence sets that both contain the same activity. Set 1 = \{A, B, C, D\} and Set 2 = \{H, I, C, J\}. If we consider only the duration of the activities in Set 1, we should assign an early start time of 14 to activity D. However, we must also account for the fact that C is dependent on activities H and I from Set 2 and, thus, cannot start until period 9. Consequently, this later start time for activity C postpones the start time for activity D to period 16.

Figure 3.9  We illustrate the complicating nature of haulage pillars. If we exclude the haulage pillars, we can solve a 52-week instance of our model in a matter of seconds. As we introduce haulage pillars, the solution time increases exponentially until the instance becomes intractable.
Figure 3.10 There are four possible outcomes when solving instances of our model. Each outcome is associated with either procedure \( Q \) or procedure \( H \). If we can produce an integer feasible solution, then the best theoretical outcome is to solve the monolith to optimality; otherwise, using a heuristic can produce a good solution. If we cannot solve the monolith for an integer-feasible solution, the best we can do is to produce a bound from the LP relaxation solution; if we cannot do that, it indicates that the problem is too large to be read into the solver.

Figure 4.1 Room-and-pillar mining is a self-supporting method, in which pillars of ore are left in place to support the rock that encompasses the ore body, or hanging wall, while the remaining ore is extracted. When only pillars remain, a retreat mining method is employed in which pillars are removed in a sequence that starts with the furthest pillar from the mine exit, allowing the host rock to cave in as the extraction proceeds towards the exit.

Figure 4.2 With a long-hole stoping approach, a sub-level drift is developed below the ore to be excavated. The ore is then drilled and blasted and falls to the sublevel where Load Haul Dump vehicles scoop and load it into trucks for transport to the crusher.

Figure 4.3 The drift-and-fill technique extracts a corridor of ore, known as a drift, from the ore body. The remaining void is then filled using a cement-based mixture to provide structural support before an adjoining drift is cut through the ore.

Figure 4.4 Cut-off grade selection determines ore block shape and size. Box (a) shows part of the ore body with a nonuniform distribution of metal. A cut-off grade of 7% results in a large area classified as ore (see Box (b)) that can partitioned into homogeneous blocks and extracted with a single mining method, e.g., drift-and-fill. At a cut-off grade of 9%, the material classified as ore diminishes (Box (c)). Drift-and-fill mining would now excavate too much waste rock so a different mining approach is used. Box (d) shows material classified as ore at a cut-off grade of 12%. With most of the area now defined as waste, a targeted mining method is suitable.

Figure 4.5 A Gantt chart tool is used for manual scheduling at Lisheen. Extraction and backfilling activities are shown as horizontal bars. The thin lines between the bars represent the precedence relationships between the activities. Planners arrange these activities on the timeline in an effort to meet production targets.
Figure 4.6  This example shows a mining panel adjacent to a haulage route. The pillars that support the haulage route are shown as P1 - P5. The arrows indicate the direction of the mine exit. The haulage path collapses west of any pillar that is removed. Hence, the extraction precedence for these pillars would require P1 be taken first and P5 last. Removal of pillar P3, P4, or P5 would result in the destruction of the haulage path in that area, eliminating access to the panel.

Figure 4.7  The dashed lines show critical haulage routes through a mining zone. These haulage routes are bordered by major panels containing blocks that are candidates for extraction. The pillars that support these haulage routes often consist of valuable ore blocks. However, once removed, we lose access to the area east of the extracted pillar.

Figure 4.8  In this histogram of block size at Lisheen, smaller blocks correspond to a small access drift or a cross-cut used to facilitate extraction of a larger block, such as a stope. The smallest blocks can be mined in a day or less. The largest require months to extract.

Figure 4.9  We illustrate our optimization-based decomposition heuristic with an simple example in which we solve the problem twice at weekly fidelity. Solve 1 shows the solution for a twelve-week horizon during which only high-grade panel and pillar activities are considered for scheduling. The times at which these blocks are extracted constitute the schedule at this point. The following solution, resulting from Solve 2, shows that blocks that were scheduled within Solve 1 are now required to remain scheduled at the equivalent week when solving this problem, but now low-grade panel blocks and pillars are also included.

Figure 4.10  The integer program brings metal forward in the production schedule.

Figure 4.11  The integer-programming solution has more consistent month-to-month production than the manually generated schedule. In addition, production in the integer program schedule remains above 80,000 tonnes, allowing the mill to operate continually. By contrast, the manual schedule suffers from low production starting in week 48, which would result in temporary mill shutdown and labor problems.
Figure 4.12 We compare integer-programming solutions for varying time horizons against the integer-programming solution for the current mine life. Increasing the life of the mine by another three months is a viable option for Lisheen. Similarity in the production tonnage in each week during the first year is a consequence of the low degrees of freedom in the precedence rules. Later, when haulage pillars become candidates for extraction, weekly metal production differs more across scenarios, reflecting the more flexible precedence constraints on these pillars in our model.

Figure 4.13 The integer-programming schedule identifies four high-grade haulage pillars as blocks that would never be mined given the current mine design. The pillars form part of an existing main haulage route (solid arrows) connecting the crusher with a large ore zone to the west (not shown). Under the original plan, mining these pillars would only be possible once all activity in the western zone had finished. However, the integer-programming results show that more profitable higher-grade pillars along the same haulage route would always be extracted in preference to these four pillars. Realizing this, Lisheen added a bypass (dashed arrows) to incorporate these pillars into the schedule.
LIST OF TABLES

Table 1.1 A broad comparison of open pit and underground mining approaches. . . . . . 4

Table 3.1 Performance comparison of the heuristic procedure, (H), with the monolith procedure, (Q), on the basis of solution times and quality. For each data set, we also show the problem size and number of subproblems in (H). Note: The † symbol indicates that the monolith did not provide a bound. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . ..
ACKNOWLEDGMENTS

While completing my Ph.D. degree has been the most challenging activity of my life, it has been a great privilege to spend several years in the Division of Economics and Business at the Colorado School of Mines.

A most sincere thank you to my advisor, Dr. Alexandra Newman, for the academic guidance, the unwavering and selfless support, the patience, and the understanding that she has shown me over the course of my studies, particularly over the past two years. I could not have had a better advisor and mentor for my academic studies. I thank the other members of my thesis committee, Dr. Rod Eggert, Dr. Daniel Kaffine, and Dr. Mark Kuchta, for their guidance, time, and support over the course of my research.

In addition, I thank Tom Bailey and Noeleen Fox of the Lisheen mine for their time and commitment, Brian Hall of AMC Consultants for his comments and advice, and Dr. Tim Kaiser, Director of Research and High Performance Computing at the Colorado School of Mines, for his resources. For her warmth and encouragement, I owe a special thank you to Kathleen Martin of the Division of Economics and Business.

Finally, my deepest gratitude I owe to my wife, Ali, and daughter, Rita. Ali, I could not have completed this endeavor without your loving support, patience, and sacrifice. Rita, your smiles, kisses, and hugs have been a constant source of relief and joy. I love you both!
The profitable extraction of a mineral asset from the earth’s crust is a complex and difficult task that depends on careful decision making. The question of how a mineral should be mined is answered with consideration of the geometry of the mineral deposit and the concentration and distribution of the mineral within. A block model partitions the deposit into discrete areas of valuable ore blocks and unprofitable waste blocks, enabling engineers to select suitable mining methods and to design the infrastructure required to access and support the mining of ore. Correspondingly, a production schedule determines the sequence in which blocks should be extracted over the operational life of the mine.

At a minimum, the production schedule outlines the quantity, location, and timing of ore extraction. The quality of the schedule is dependent on the quantity of alternative scheduling scenarios that planners examine. While mine planning software can enable them to examine a greater number of schedules, the combinational nature of the problem precludes examination of the entire set of alternatives. Consequently, mine production plans can be far from optimal. Optimization has made significant progress in this area over the past thirty years (Newman et al., 2010); however, the difficulty associated with producing optimal production schedules continues to challenge researchers, particularly for underground mining operations.

The mine production scheduling problem requires mine planners to select and schedule ore blocks for extraction in a sequence that maximizes or minimizes a specific goal, e.g., deviations from planned production targets. We use binary variables to represent the time at which a block is scheduled, while the objective could minimize costs, minimize makespan, or maximize profits. Two principal types of constraints exist: (i) resource constraints, which limit the number of activities committed to a time period based on the availability of a given supply and on the amount of that supply required to perform the activity, and (ii)
precedence constraints, which dictate the order in which activities must be completed.

The quality of the production schedule, or the degree to which the schedule satisfies the objectives of management, is a key driver of profitability for a mining venture. In addition, financial standards and practices for mine valuation (JORC, 2012) mandate that only ore that is scheduled for production can be considered as part of a mine’s reserves. Consequently, the production schedule directly affects the market valuation of the mining operation.

The goal of our research is to develop a solution methodology to produce optimal production schedules for a complex underground lead and zinc mine at Lisheen in Ireland. Correspondingly, we consider the research question: Can we develop a general solution methodology to produce optimal production schedules for underground mining operations? Consequently, in this research, we: (i) explore the aspects of underground mining that complicate schedule optimization; (ii) develop an optimization-based heuristic and show that it produces near optimal schedules for a number of mine datasets; and (iii) implement our solution at the Lisheen mine.

1.1 Background

Mining has always been fundamental to human development. The oldest known mine dates back 43,000 years to the Paleolithic age. Prehistoric man mined flint to produce the tools and weapons required to survive and prosper in the Stone Age. The Bronze and Iron Ages that followed were defined by the production of pure metals that resulted from the discovery of smelting. Ancient miners used fire and water to break the host rock, and by 2000 BC were already reaching depths of 250 meters below ground. Mining continued in this way for centuries until the start of the industrial revolution when modern mining, which continues to provide the minerals that are essential for economic growth, began with the invention of dynamite (Gregory, 1980). Construction of infrastructure, manufacturing of high-tech consumer goods, agricultural productivity, and generation of energy all depend on the extraction of large amounts of mineral resources.
These days, metallic, or hard rock, mining is separated into two principal categories: (i) surface mining, and (ii) underground mining (Figure 1.1). We compare the two mining approaches in Table 1.1. When the ore is located close to the earth’s surface, the most productive and economically efficient mining method is open pit, a form of surface mining. With this method, notional three-dimensional blocks of material are extracted from an ore body, regardless of whether they meet the cut-off grade, i.e., the percentage mineral content at which the material is deemed profitable, declared ore, and processed. Material below the cut-off grade is termed waste and discarded. Consequently, a significant factor in the viability of the open pit approach is the ratio of extracted ore to extracted waste. In the United States, surface mining accounts for approximately 85% of all mining activity (Hartman, 2007).

Despite its relatively lower infrastructure cost, surface mining becomes cost prohibitive when the ratio of extracted waste to ore becomes too high. With targeted extraction methods that minimize costly waste production, underground miners seek to extract only blocks of material that will be processed into mineral concentrate. Data from the United States in
1997 indicated that waste at open pit mines accounted for 73% of total material extracted, whereas underground mining extracted only 7% waste rock (Hartman, 2007). Underground mining is also suitable for mineral deposits located in environmentally sensitive areas, where high reclamation costs would be associated with an open pit operation. However, while they minimize waste production and the environmental footprint, these underground operations are also more complex, have higher extraction costs, and are far more dangerous than surface mines.

Table 1.1: A broad comparison of open pit and underground mining approaches.

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Open Pit</th>
<th>Underground</th>
</tr>
</thead>
<tbody>
<tr>
<td>% of U.S. Mining Activity</td>
<td>85%</td>
<td>15%</td>
</tr>
<tr>
<td>Complexity</td>
<td>Less complex</td>
<td>More complex</td>
</tr>
<tr>
<td>Waste Mining</td>
<td>Very high</td>
<td>Low</td>
</tr>
<tr>
<td>Stockpiling Ore</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Environmental Disruption</td>
<td>Large footprint</td>
<td>Small footprint</td>
</tr>
<tr>
<td>Safety</td>
<td>Relatively Safe</td>
<td>Dangerous</td>
</tr>
<tr>
<td>Reclamation Costs</td>
<td>High, if reclamation required</td>
<td>Relatively Low</td>
</tr>
</tbody>
</table>

There are five stages in a typical mining project: prospecting, exploration, development, exploitation, and reclamation. Geologists prospect by searching for valuable minerals. During the exploration stage, mining engineers drill boreholes to sample and define the extent and richness of a mineral deposit. A mineral deposit that they judge to be economic is termed an ore body and becomes a candidate for development, and the quantity of mineral that can be economically extracted defines the mining reserve. In the third stage, a developer designs and constructs the initial infrastructure required to access the ore body to commence extracting and processing ore. Major decisions at this stage involve extremely high capital costs, are irreversible, and can greatly affect long-term profitability. Exploitation, the longest stage in the life of a mine, includes the production, refinement, and shipping of ore. The operating strategy during this period is strongly influenced by external market factors and mining reserve updates resulting from extraction activity and continued exploration. Conse-
quentely, mines frequently review their operating strategy, e.g., change extraction methods, adjust production targets, and update production schedules, with a view to maximizing profits. The final stage is reclamation, in which the mine is decommissioned and, if it is required by regulatory agencies, remediated. Exhausted mines have been converted to lakes, underground storage facilities, waste disposal sites, and wildlife areas. While there is a sequential order to these stages, some can overlap others (Figure 1.2).

![Diagram](image)

Figure 1.2: There are five stages in the life cycle of a mine. Some stages overlap others, for example, production can begin once access to the ore body has been developed, while development and exploration continue in other parts of the mine. Mining ventures may wish to begin production as early as possible to generate a positive cash flow that can be used to service debt. (Kuchta, 2008)

1.2 Mine Production Schedule Optimization

The quality of the production schedule used in the exploitation stage of mining depends, in part, both on the accuracy of the model (and solution procedure), and on the quantity of alternative scheduling scenarios that planners examine; while mine planning software enables the examination of a greater number of schedules, the complex and mine-specific nature of the operations and the combinational nature of the production scheduling problem in many cases preclude both the generation of near-optimal schedules and the quick examination of the entire set of alternatives, especially for schedules with a large number or activities and/or a long time horizon with fine fidelity. Consequently, mine production plans can be far from
optimal. While better models are being developed, solution techniques are being refined, and hardware and software continue to improve (Newman et al., 2010), production scheduling problems, particularly for underground mining operations, continue to challenge researchers.

1.3 Literature Review

Applications of operations research to mining have concentrated on strategic and tactical planning decisions at the development and exploration stages in the life cycle of a mine (Newman et al., 2010; Topuz & Duan, 1989). Strategic planning is concerned with long-term decisions that affect the value of the mining venture, such as: (i) determining the most economical pit size for a surface mine, or the ultimate pit; (ii) locating the processing plant (i.e., the mill) and other facilities; (iii) selecting machines and equipment; (iv) designing infrastructure, e.g., the number and placement of ramps, roads, and shafts; and (v) scheduling production in the long term. Over shorter horizons, tactical planning considers the decisions required to realize the objectives of the strategic plan. Examples include: (i) routing trucks; (ii) blending ore at the mill; and (iii) scheduling production in the short term.

Williams et al. (1973) first demonstrate the potential role for optimization in underground mine production planning. While their linear programming approach suits certain strategic problems that involve grade control or cost reduction, the method cannot incorporate binary decisions required to enforce the logic to schedule production at an operational level. On the other hand, Chanda (1990) combines mixed integer programming (MIP) and simulation to produce a schedule for six consecutive work shifts for a copper mine in Zambia. Winkler (1996) also employs integer variables, not only to capture operational relationships, but also to model costs as piecewise linear functions. She highlights the advantages of applying mixed integer programming to underground mine scheduling and illustrates the exponential complexity associated with this approach when producing a schedule for a multi-period time horizon. Solving a single-period MIP model at a time, she relies on simulation to produce a multi-period schedule for a German coal mine. Trout (1995) presents a more generalized MIP approach for underground mine production scheduling. Introducing variable restrictions to
reduce the size of his model, he produces a schedule that maximizes the net present value of a copper mine for a 17-period time horizon. Carlyle & Eaves (2001) apply mixed integer programming to schedule production at an underground platinum and palladium mine. They maximize discounted ore revenue by solving their model for a number of mine expansion scenarios.

Sarin & West-Hansen (2005) develop a Benders’ decomposition technique to produce an exact solution for their mixed-integer programming model. Application of their model to a coal mining case study results in a profit-maximizing 100-week schedule. Newman & Kuchta (2007) optimize long-term production at an underground iron ore mine in Sweden. With an objective to minimize deviations from contracted production quantities, they use an aggregation heuristic to solve a mixed-integer programming model for a 60-month horizon. This model forms the basis for Martinez & Newman (2011), who schedule both long-term and short-term production for the same iron ore mine. The authors produce solutions for a 48-month horizon using a decomposition heuristic.

1.4 Original Contribution

While both strategic and tactical planning models can be used to optimize production, the objectives of the models and the results they produce can be quite different. A strategic solution indicates a course of action, with approximate ore tonnage targets at a quarterly or annual level of detail, to maximize a measure of value over the life of the mine, i.e., a net present value (NPV). The underlying assumption is that the production targets are feasible and implementable in the short term, in which case planners can solve a tactical production scheduling model for a detailed plan. However, for complex mines, when strategic models consider only an aggregate level of detail, the production plan may prove infeasible and impractical in reality. Therefore, the more complex the mining operation is, the more tactical-level detail a strategic model must consider.

The Lisheen mine in Ireland motivates our production scheduling work. In this case, the mine was having difficulty satisfying production targets and faced the possibility of early
closure. The complex operation required a strategic production scheduling model that could consider both the implications of extraction decisions for production in future time periods and, at the same time, satisfy tactical constraints. The strategic component necessitated a two-year time horizon, corresponding to the remaining life of the mine. In addition, the complexity of the mine mandated that an implementable schedule would need to be high-fidelity, with decisions made at weekly time intervals. A feasible production plan: (i) satisfies the resource limitations at the mine, i.e., maximum possible production levels; (ii) maintains a blend of ore that meets the mill’s requirements; and (iii) follows a sequence of mining that maintains infrastructural stability and ensures the areas to be mined are accessible to extraction equipment.

Similar to the work we cite in the literature review, we formulate, solve, and implement an integer-programming model for production scheduling in an underground mining operation. However, although there are similarities between almost all underground mine production scheduling models, e.g., blocks, time periods, sequencing constraints and resource restrictions, each mine operates in a specific manner. For example, objectives tend to differ between mines; an appropriate objective for a mine containing precious metal may be to maximize net present value, while one for a mine containing base metal may correspond to minimizing deviations from long-term contracts. Some mines may stockpile to exploit blending and its associated quality of output while others may regard such a policy as a nuisance, requiring rehandling and its costs. Specific to our application are: (i) a production scheduling model that uses only binary variables, which, because of blending requirements, results in indivisible blocks; (ii) a discounted objective involving metal (but no explicit economic parameters); (iii) a non-homogeneous (spatially, temporally, and regarding their metal content) set of ore blocks; (iv) a mine that uses three different underground mining methods, which complicates the associated rules governing extraction, and, where applicable, backfill; and (v) a mine that is approaching closure.
Our contributions consist of the following: (i) we develop a paradigm in which to model complex scheduling operations, such as (but not limited to) those at Lisheen; (ii) we improve the tractability of our problem by using exact and heuristic methods to reduce the problem size; and (iii) we develop an optimization-based decomposition heuristic that can take intractable instances of the model and generate schedules within seconds (for shorter time horizons) or hours (for long, post-life-of-mine schedules).

1.5 Dissertation Organization

This body of this dissertation is comprised of three papers; each one focuses on a different aspect of our research.

Chapter 2 addresses the discrepancies between state-of-the-art optimization software for open pit mines and the absence of such software for underground mining operations. After a brief review of integer programming, we compare the open pit scheduling problem with the underground equivalent and show how the greater complexity of underground mines results in more difficult mathematical structure and, consequently, a lack of commercial software for underground mine scheduling.

Chapter 3 provides a detailed look at tackling a complex underground mine scheduling problem. We develop a conceptual framework for the mining operation and formulate the scheduling problem. We exploit the mathematical structure of the problem to improve its tractability and apply and develop, respectively, relevant exact and heuristic preprocessing techniques to reduce the problem size. Finally, we develop an optimization-based decomposition heuristic solution method and demonstrate its effectiveness by solving a number of problem instances for real mine data sets.

Chapter 4 focuses on the challenges of applied operations research and the process of implementation. We describe the application of our research to the Lisheen mine in Ireland. We detail the traditional manual scheduling approach and develop a mathematical model of the mining operation that adheres to the same constraints as those faced by the mine planner. We employ our optimization-based decomposition heuristic to produce high-quality solutions.
for the mine, and conduct scenario analysis to examine alternative mine closure dates for management.

In Chapter 5, we present our conclusions, summarize our research contribution, and identify possible extensions to this research.
CHAPTER 2
TRACTABILITY OF INTEGER PROGRAMS FOR OPEN PIT AND UNDERGROUND MINE PRODUCTION SCHEDULING OPTIMIZATION

When Lerchs & Grossmann (1965) demonstrated that an apparently intractable open pit mine integer-programming (IP) problem could be easily solved if formulated as a network, they illuminated a path for mine optimization research to follow. Their technique for solving the ultimate pit limit problem, \( \text{UPIT} \), i.e., maximizing the undiscounted economic boundary of a mineral deposit subject to precedence constraints, exploits the network structure that underlies the precedence rules for ore block extraction in many open pit mines. Academic research over the past three decades has focused on using the Lerchs-Grossmann algorithm as a core mechanism for solving more general open pit mine scheduling problems. This knowledge base has transferred to industry and, at present, a number of software firms (GEOVIA, 2012; Maptek, 2012; MineMax, 2012b) offer production scheduling optimization solutions for open pit mining operations.

Because open pit mines comprise the vast majority of mining operations in the United States (Hartman, 2007), the commercial focus on optimizing their production schedules is understandable. However, while companies strive to produce next-generation tools for open pit mines, e.g., optimization of integrated mine design and production problems, scheduling software for underground mines remains limited to supporting manual production planning, a complex and difficult task that can take months to complete.

2.1 Integer Programming Review

Integer programming has been the predominant approach used to model mine scheduling problems. An integer program is a mathematical model comprised of an objective function, integer variables, and constraints. The objective function seeks to minimize or maximize a goal, e.g., maximize the sum of discounted future cash flows. Binary variables are used
to assign blocks to time periods, i.e., $y_{bt} = 1$ if block $b$ is scheduled for extraction at time $t$, otherwise $y_{bt} = 0$. Continuous variables can also be included in mixed integer programs, i.e., we could define variables, $x_{bt}$, to represent the amount of material extracted from block $b$ at time $t$. Finally, the constraints ensure that resource limitations are not exceeded and precedences between blocks are adhered to. Integer programming is particularly suited to scheduling problems because binary variables can be used to enforce the logic required by the precedence constraints.

Optimal solutions to integer programs can be identified with the branch-and-bound (B&B) algorithm. A process of intelligent enumeration, this algorithm systematically examines a tree of potential solutions, using upper and lower bounds to eliminate dominated solutions. Commercial solvers, e.g., CPLEX (IBM, 2011), implement variations of this algorithm in combination with other advanced optimization techniques, i.e., heuristic methods and cutting planes. In theory, as the size of an integer program grows, the time required for the branch-and-bound algorithm to solve the problem increases exponentially. For this reason, despite phenomenal increases in computing power, large and complex mine scheduling models continue to challenge researchers.

2.2 Integer Program Mine Design and Scheduling Models

Before examining the complexity of mine optimization problems, we first review two general mathematical formulations for mining optimization as they appear in Espinoza et al. (2013). While there are many formulations for mining optimization, we focus on the ultimate pit limit problem, ($UPIT$), and on the constrained pit limit problem, ($CPIT$); both of these form the basis for more complex models.

The mathematical formulations follow:

- **Indices and sets:**
  - $t \in \mathcal{T}$: set of time periods $t$ in the horizon.
  - $b \in \mathcal{B}$: set of blocks $b$. 

- $b' \in \hat{B}_b$: set of blocks $b'$ that are predecessor blocks for block $b$.

- $r \in \mathcal{R}$: set of operational resource types $r$.

**Parameters:**

- $p_b (\hat{p}_t)$: profit obtained from extracting (and processing) block $b$ (at time period $t$) ($\$$).

- $q_{br}$: the amount of operational resource $r$ used to extract and, if applicable, process, block $b$ (tons).

- $R_{rt}$: minimum availability of operational resource $r$ in time period $t$ (tons).

- $\overline{R}_{rt}$: maximum availability of operational resource $r$ in time period $t$ (tons).

**Variables:**

- $\hat{x}_b = 1$ if block $b$ is in the final pit design, 0 otherwise.

- $x_{bt}$: 1 if we extract block $b$ in time period $t$, 0 otherwise.

$$(UPIT) \quad \max \sum_{b \in \mathcal{B}} p_b \hat{x}_b$$

subject to $\hat{x}_b \leq \hat{x}_{b'} \quad \forall b \in \mathcal{B}, b' \in \hat{B}_b$ \hspace{1cm} (2.1)

$\hat{x}_b \in \{0, 1\} \quad \forall b \in \mathcal{B}$ \hspace{1cm} (2.2)

The $(UPIT)$ problem is the open pit design problem that the Lerchs-Grossmann algorithm solves. It maximizes profit by determining the pit size which contains the most economic selection of blocks. Each block $b$ has a value, $p_b$, and is associated with the binary variable, $\hat{x}_b$, that assumes a value of 1 if $\hat{x}_b$ is chosen for extraction, and 0 otherwise. Precedence constraints (2.1) ensure that any block, $b \in \mathcal{B}$, can only be extracted once all of its predecessors, $b' \in \hat{B}_b$, have been extracted.

The $(CPIT)$ model builds on the ultimate pit limit problem with the introduction of a time dimension. Consequently, variables and parameters now have an additional index,
t, to track time, and resource constraints can be incorporated into the model. A solution to the \((CPIT)\) problem is a production schedule with a profit maximizing extraction sequence that satisfies resource constraints, e.g., production capacity, and precedence rules. Constraints (2.3) enforce precedence rules. The additional constraints (2.4) restrict a block to be extracted at most once, and resource constraints (2.5) ensure that total resource usage does not exceed its availability.

\[
\text{(CPIT)} \quad \max \sum_{b \in \mathcal{B}} \sum_{t \in \mathcal{T}} \hat{p}_{bt} x_{bt}
\]

subject to
\[
\sum_{\tau \leq t} x_{b\tau} \leq \sum_{\tau \leq t} x_{b'\tau} \quad \forall b \in \mathcal{B}, b' \in \hat{\mathcal{B}}_b, t \in \mathcal{T} \quad (2.3)
\]
\[
\sum_{t \in \mathcal{T}} x_{bt} \leq 1 \quad \forall b \in \mathcal{B} \quad (2.4)
\]
\[
\underline{R}_{rt} \leq \sum_{b \in \mathcal{B}} q_{br} x_{bt} \leq \overline{R}_{rt} \quad \forall t \in \mathcal{T}, r \in \mathcal{R} \quad (2.5)
\]
\[
x_{bt} \in \{0, 1\} \quad \forall b \in \mathcal{B}, t \in \mathcal{T} \quad (2.6)
\]

The tractability of these models depends on: (i) the size of the problem; and (ii) the structure of the precedence constraints. The size of an integer programming problem is primarily a function of the number of variables, the number of constraints, and the density of those constraints, i.e., the number of variables in the constraint. For the \((UPIT)\) problem, the number of variables matches the number of blocks in the problem. The \((CPIT)\) problem contains considerably more variables because we define a variable for every block and start-time combination. And, in addition to the precedence constraints, the \((CPIT)\) formulation has \(|\mathcal{R}| \times |\mathcal{T}|\) sets of resource constraints that limit production. \((CPIT)\) problems will generally contain fewer resource constraints than precedence constraints because the number of precedence constraints is a function of the number of blocks and time periods considered by the model, \(|\mathcal{B}| \times |\mathcal{T}|\).
2.3 Tractability of the \((UPIT)\) and \((CPIT)\) Problems

Many real-world scheduling problems that exhibit simple and repeatable patterns are solved with specialized techniques that take advantage of their structure, i.e., the underlying network in the \((UPIT)\) problem, for which network algorithms can expedite solutions. Ahuja \textit{et al.} (1993) show that by considering each block as a node and representing the precedence relationships between blocks with directed arcs, \((UPIT)\) can be modeled as a maximum weight closure problem and solved with a polynomial-time algorithm relatively quickly in comparison to solving the original problem with branch and bound.

For the \((UPIT)\) example, we can examine the density of the \(A\) matrix by rearranging constraints (2.1) so that all variables are on the left-hand side:

\[
\hat{x}_b - \hat{x}_{b'} \leq 0 \quad \forall b \in B, \forall b' \in \hat{B}_b
\]

The \(A\) matrix of coefficients for this problem follow in matrix format:

\[
\begin{bmatrix}
1 & -1 & 0 & 0 & 0 & \ldots & 0 \\
0 & 1 & -1 & 0 & 0 & \ldots & 0 \\
0 & 0 & 1 & -1 & 0 & \ldots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & 0 & 0 & 0 & \ldots & -1
\end{bmatrix}
\]

The performance of commercial optimizers is significantly affected by the density of a problem’s \(A\) matrix, i.e., the ratio of nonzero to zero values. These solvers reduce problem size by storing only nonzero matrix coefficients in memory. However, when the \(A\) matrix averages more than ten nonzeros per column, it is considered dense. Consequently, with a large number of nonzero values with which the solver must compute, solution time will slow considerably (Klotz & Newman, 2013a). Including too many resource constraints quickly results in a dense \(A\) matrix. Even though precedence constraints can far outnumber resource constraints, the impact of a large number of precedences may not be as drastic if the con-
straints are sparse and, more significantly, if a network structure underlies the precedence constraints.

2.4 Mine Structure and Production Schedule Optimization

The characteristics of a mining operation define the mathematical structure of its scheduling optimization problem; this structure, in turn, determines the tractability of the problem. In this subsection, we examine the relationship between precedence constraints and problem tractability, and discuss the suitability of decomposition solution techniques to both open pit and underground mining operations.

2.4.1 Precedence Constraint Structure

Perhaps the most significant difference between open pit and underground mine scheduling problems is in the structure the underlies the precedence rules that govern the sequence of extraction between blocks of ore. For open pit mines which employ a repeatable precedence rule, such as the plus sign convention (Figure 2.1), this underlying structure forms a network that can be exploited by the Lerchs-Grossmann algorithm when: (i) solving the (UPIT) problem; or (ii) solving the (CPIT) problem in combination with a heuristic or exact method, e.g., a Lagrangian relaxation procedure (Dagdelen & Johnson, 1986).

Figure 2.1: The “plus sign” convention for open pit block sequencing is shown on the right. Blocks 2–6 must be mined prior to mining block 1.

Underground mine precedence structure can differ greatly from one mine to the next. For the most part, the method of extraction used in an area of an underground mine dictates
the order of mining in that area. In addition, underground mines often use a combination of mining methods to extract ore, which makes it difficult to develop a general mine-wide precedence rule similar to the plus sign convention. Finally, while open pit precedence rules only concern the order of mining blocks, at underground operations, precedence rules can relate extraction activities to non-extraction activities, such as ventilation requirements (Brickey, 2013), structural support, or safety protocols, e.g., a legal requirement to maintain at least two paths of egress from a working area. Consequently, even when underground mines have a single, uniform mining method with a repeatable precedence pattern, e.g., sublevel caving at the Kiruna Mine in Sweden (Newman & Kuchta, 2007), other precedence rules may preclude the underlying network structure that commercial solvers could exploit to produce timely solutions. In addition, complex precedence logic (Martinez & Newman, 2011) will result in precedence constraints with more variables, and this will produce a dense A matrix, slowing computation.

In Figure 2.2, we show another example of a mine with complex precedences, the Lisheen mine in Ireland. In this case, a network structure likely underlies the precedence rules between all blocks with the exception of those precedence constraints that govern the haulage pillars, i.e., blocks that are left in place to support the haulage routes (O’Sullivan & Newman, n.d.). The majority of the precedence constraints in the model are similar to constraints (2.1), which we have shown form an underlying network. However, while extracting a haulage pillar is optional, and not required to facilitate extraction of other blocks, it may prevent extraction in other areas of the mine. This type of precedence constraint incorporates an “or” decision and cannot be incorporated into a network model. Consequently, precedence rules governing haulage pillars do not have underlying network structure. As a result, the tractability of the Lisheen scheduling problem is largely determined by the number of haulage pillars included in the model (Figure 2.3). However, while the inclusion of a single pillar will preclude the existence of a pure network, anecdotal evidence (Klotz, 2013) based on the performance of the CPLEX solver suggests that if more than 85% of the constraints in a model form a
network, the solver may be able to exploit that structure for performance improvements.

Figure 2.2: We show a conceptual illustration of precedence rules at the Lisheen underground mine. In Box (a), we render a panel of ore blocks adjacent to a haulage route comprised of ore containing haulage pillars. Within the panel, strict precedence rules, similar to constraints (2.1) above, dictate that Block A must be taken before Block B and Block B before Block C, i.e., it is impossible to take Block C without extracting both Block A and Block B in advance. Precedence rules of this type will have underlying network structure, i.e., by representing the blocks as nodes, we define arcs between pairs of nodes to represent the precedence rules. Extraction of pillars follows a sequence that retreats toward the exit. However, extraction of a pillar is optional; its extraction does not require or facilitate the extraction of any other block. In addition, as Box (b) illustrates, once a haulage pillar (Pillar 2) is extracted, the roof caves-in, blocking access to pillars (Pillar 1) and blocks (A, B, and C) upstream of the extracted pillar. Consequently, this precedence rule governs an “or” decision that cannot be incorporated into a network model, e.g., we can extract Pillar 1, or Pillar 2, or both.

2.4.2 Problem Decomposition

Optimal open pit mine schedules can be produced by decomposing the problem into a series of smaller and more tractable subproblems. One general approach employs the Lerchs-Grossmann algorithm to solve a series of (UPIT) problems which outline nested pits within the ultimate pit; these nested pits are subsequently solved as a series of scheduling subproblems to produce an overall production schedule for the mine. Because the (UPIT)
Figure 2.3: We illustrate the effect of complex precedence rules on tractability. For the Lisheen mine, if we exclude haulage pillars from the scheduling problem, we can solve a 52-week instance of the model in a matter of seconds. As we introduce haulage pillars, complexity and solution time increase (in an approximately exponential progression) until problem instances are no longer tractable.

problem is specific to open pit mines, there is no equivalent decomposition technique that can be applied to underground mines.

A fundamental difference between open pit and underground mining is the treatment of waste material. At open pit mines, no distinction is made between blocks of ore and blocks of waste during extraction because waste blocks must be extracted to access the ore blocks below them. Subsequently, waste blocks are discarded and ore blocks are sent either directly to the mill for refinement, to a leaching heap, or to a stockpile from which the mill can draw a combination of blocks to satisfy grade control blending requirements. From an analytical perspective, the open pit mining process from ore to product can be separated into three decisions: (i) which blocks to extract at time \( t \); (ii) whether to send a block to the mill, leaching heap, stockpile, or dump; and (iii) what ore blocks to select from the stockpiles at time \( t \) to satisfy grade control. While there is a degree of interdependency between these decisions, there may be sufficient separation to model and optimize each one independently.
and then employ a heuristic to provide a solution for the monolith. Consequently, for some mining operations, each model need only consider the subset of operational constraints that is relevant to the particular decision, which reduces the density of the $A$ matrix and improves tractability. Such an approach is commonly used by commercial optimizers, e.g., GEOVIA’s open pit blending optimization module solves linear programming (LP) blending problems within each iteration of their life-of-mine scheduling algorithm.

By contrast, underground miners seek to extract only the blocks of ore that they decide a priori will be refined. Few underground mines stockpile ore; consequently, underground mines that blend must coordinate the extraction of blocks so that the flow of ore directly satisfies the blending requirements at the mill. Consequently, although we have only one decision to consider, it is difficult to apply the types of decomposition heuristics that work well for open pit mine scheduling problems.

2.5 Ore Block Characteristics and Aggregation Heuristics

In addition to the factors mentioned in Section 2.4, the definition of ore block shapes can significantly influence the tractability of mine scheduling problems, particularly, with respect to the application of heuristic techniques that rely on block aggregation to produce solutions.

2.5.1 Homogeneous Block Size

Characterization of blocks in an open pit mine as identical cuboids allows researchers to formulate more tractable open pit scheduling problems. When research into mining optimization was still in its infancy, Johnson (1969) recognized the importance of ore block homogeneity, stating that “the block concept is one of the keys to the improvements in mine planning to-date.” The regularity of block size and shape facilitates: (i) the definition of repeatable precedence rules; (ii) aggregation of ore blocks to reduce problem size (Subsection 2.5.2); and (iii) the selection of a suitable time fidelity for the problem. However, in underground mining, the volume and dimensions of the ore blocks can vary greatly. This
heterogeneity can be a consequence of: (i) the technical design which is based on the host rock strength and the distribution of mineral concentration; (ii) fissures in the rock; or (iii) unpredictable results of blasting.

When block size varies considerably, it can be difficult to choose a time interval for the schedule, which must be small enough so that activities of shorter duration do not unnecessarily push the schedule forward (Figure 2.4). However, because we must account for finer fidelity by defining variables for each activity and start-time combination, we must also be mindful to select a time fidelity that is large enough to produce a tractable model. An obvious way to cope with heterogeneously sized blocks is to aggregate them into clumps of similar sizes. However, this technique is precluded for underground mines that blend ore, and also for operations with complex precedence rules that relate extraction activities to non-extraction activities.

Figure 2.4: At weekly fidelity, the total scheduling gap between Block A, Block B, and Block C is much smaller than at bi-weekly fidelity. These gaps unnecessarily extend the start times for extraction of dependent blocks. While Block C can start during week 3 with weekly fidelity, at bi-weekly fidelity, the same block must wait until period 2 (week 4) before it can start.
2.5.2 Aggregation of Ore Blocks

When commercial software produces optimized tactical schedules for large open pit problems with hundreds of thousands of blocks, the degree to which this schedule is actually “tactical” is questionable. Although an optimal solution may be produced with an exact optimization approach, i.e., branch and bound, these commercial packages reduce the size of the problem by aggregating blocks to reduce the number of variables in the model. For instance, 100,000 blocks to be scheduled over 100 time periods would result in the definition of ten million variables. By aggregating blocks into clumps of 100 blocks each, the model need only consider 1,000 clumps \( \times \) 100 time periods, or 100,000 variables. While this approach can produce a good schedule quickly, it may not be feasible to implement if the aggregation is too high-level, ignoring critical tactical operations. In their recent paper, Bienstock & Zuckerberg (2010) emphasize the drawback to aggregation:

Mine scheduling software packages have recently emerged that aggregate blocks in order to yield a mixed integer program of tractable size. The required degree of aggregation can, however, be enormous; this can severely compromise the validity and the usefulness of the solution.

An aggregated solution that a mine can disaggregate, even if it is in a heuristic manner, is better than no solution at all. In addition, for cases in which the solution is not feasible at a disaggregated level, these solutions may suffice for long-term planning purposes. The more homogeneous the ore blocks in a mine, the more applicable aggregation solution heuristics will be. Consequently, while ore blocks in open pit mines are not perfectly homogeneous, e.g., mineral content will vary, their uniform dimensions and the repeatable precedence structure that results from that, enables aggregation heuristics to work well as a result. By contrast, for underground mines, the irregular ore block shapes and resulting complex precedence rules, in addition to blending requirements at some mines, commonly preclude block aggregation as a viable approach to reduce problem size and improve tractability, even for long-term planning.
2.6 Conclusion

While schedule optimization software for open pit mines has advanced considerably over the past two decades, underground mine planners must remain content with software-assisted manual scheduling. Improvements in open pit mine scheduling software can largely be attributed to software companies’ incorporation of heuristic methods into their products. Because underground mining operations are more complicated than surface mining operations, many open pit heuristic methods are not obviously and directly applicable. Consequently, underground mining operations must rely on tailored models to generate optimized production schedules.

Recently, Bienstock & Zuckerberg (2010) presented a novel algorithm for scheduling large-scale precedence-constrained open pit mines. Their method solves the linear programming relaxation of the integer programming problem, i.e., the integer variables are relaxed to be continuous. However, while their model produces results for large problem instances significantly faster than standard techniques, the problems they consider are relatively simple in structure, with few resource constraints and no blending requirements. Despite this simplicity, their approach, like the Lerchs-Grossmann algorithm of the 1960s, highlights a new research direction. Already, other optimization practitioners are building on Bienstock and Zukerberg’s work by developing rounding algorithms to produce integer solutions from the LP relaxations (Chicoisne et al., 2012). While, researchers have developed many techniques to improve the tractability of the open pit mine scheduling problem over recent decades, few have made the transition to underground mining operations. However, future research will likely uncover techniques that can be translated to underground mines. For example, the Bienstock and Zukerberg method may prove applicable to difficult underground scheduling problems and significantly improve their tractability, particularly if a rounding heuristic is developed to produce integer solutions.
CHAPTER 3
AN OPTIMIZATION-BASED DECOMPOSITION HEURISTIC METHOD FOR SOLVING COMPLEX UNDERGROUND MINE SCHEDULING PROBLEMS

When planning the development of an open pit mine, engineers generate a computerized block model that represents the ore body as a three dimensional array of blocks (Figure 3.1). For a given set of economic parameters, each block possesses a value based on its volume and metal content. In open pit mining, ore blocks can be stored in graded stockpiles from which mill operators draw to produce a consistent grade of feed for the mill. The order in which blocks must be extracted is dictated by geometric precedence constraints, as Figure 3.1 depicts.

Figure 3.1: On the left, we illustrate a representative block model. The “plus sign” convention for open pit block sequencing is shown on the right. Blocks 2–6 must be mined prior to mining block 1. (Cullenbine et al., 2011)

For an underground mine, block models are used, in conjunction with economic information, to identify areas of ore that appear to be profitable to extract. However, unlike open pit mining, underground mining considers the in situ ore as inventory, employing targeted extraction approaches. Because the costs of extraction are significantly higher than those
associated with surface mining, waste blocks are only extracted in order to access the ore.

Figure 3.2: The Lisheen mine employs three underground mining methods. (Hamrin, 2001)

Figure 3.2 depicts the three different mining methods used at the Lisheen lead and zinc mine in south central Ireland (O’Sullivan & Newman, n.d.). Where the host rock is strong, and the ore body is not steeply angled, room-and-pillar mining is preferred (a). Areas of the mine where the ore is thick and the host rock is strong are suited to the large-scale and economically efficient long-hole stoping method (b). Finally, where the mine has poor host rock strength, drift-and-fill mining is practiced (c), i.e., voids must be backfilled for structural stability. Cut-off grade selection determines ore block shape and size, which, in turn, can define the mining method and precedence structure (Figure 3.3).

3.1 Production at the Lisheen Mine

At a depth of 170 meters below the surface, the ore bodies at Lisheen are shallow enough to be accessed by a decline, rather than by a shaft. Their predominantly flat lying orientation allows Load Haul Dump trucks to transport blasted ore blocks along a network of haulage
Figure 3.3: Cut-off grade selection determines ore block shape and size. Box (a) shows part of the ore body with a nonuniform distribution of metal. A cut-off grade of 7% results in a large area classified as ore (see Box (b)) that can partitioned into homogeneous blocks and extracted with a single mining method, e.g., drift-and-fill. At a cut-off grade of 9%, the material classified as ore diminishes (Box (c)). Drift-and-fill mining would now excavate too much waste rock so a different mining approach is used. Box (d) shows material classified as ore at a cut-off grade of 12%. With most of the area now defined as waste, a targeted mining method is suitable. (O’Sullivan & Newman, n.d.)
routes throughout the mine. These haulage routes, supported by large pillars of ore, connect working areas, or panels, to a crushing machine that fragments ore before conveyance to the mill for refinement into zinc concentrate. Each panel contains a number of blocks of ore and can only be accessed while the haulage route that connects it to the crusher remains intact (Figure 3.4).

Figure 3.4: The arrows show the haulage routes through an ore body. We highlight a mining panel adjacent to a haulage route. The pillars that support the haulage route are shown as P1 - P5. The arrows indicate the direction of the mine exit. The haulage route collapses west of any pillar that is removed. Hence, the extraction precedence for these pillars would require P1 be taken first and P5 last. Removal of pillar P3, P4, or P5 would result in the destruction of the haulage path in that area, eliminating a means to access the panel. (O’Sullivan & Newman, n.d.)

Lisheen’s objective is to determine a schedule for the extraction of ore that maximizes the quantity of metal produced by the mill over the remaining mine life. A feasible schedule: (i) satisfies the resource limitations, i.e., maximum production rates, at the mine; (ii) ensures that concurrently mined ore blocks and haulage pillars produce a blend of ore that satisfies mill requirements, (iii) guarantees that the paste fill required to create structurally stable
voids doesn’t exceed a maximum volume per time period, and (iv) enforces precedence rules. Lisheen, like most underground mines, does not stockpile ore. Consequently, grade control for the milling operation is more challenging than for an open pit mine that can draw from a stockpile.

3.2 Precedence Framework and Model

Mine planners at Lisheen partition the ore bodies into 88 distinct panels, clustering adjacent blocks of ore on the basis of a common mining method, or shared access to haulage routes. For each panel, they define a logical sequence for extraction and backfilling activity. Considering haulage pillar extraction to be independent of that of any panel, we model the structure of the mine as panels of ore that are accessed by a network of critical haulage routes supported by haulage pillars. Consequently, we define three distinct types of activities in our model: (i) panel block extraction; (ii) pillar extraction; and (iii) panel void backfilling. We differentiate between the two types of extraction activity because: (i) panel extraction activities must be proceeded and/or followed by other activities, whereas pillar activities must not; and (ii) execution of a panel activity has consequences only for activities that are adjacent to it, whereas extraction of a pillar prevents the execution of any activity “upstream” of that pillar. To simplify our presentation, we refer to the target of a panel extraction activity as a panelblock and the target of a pillar extraction activity as a pillar. We use the term block in a general way to refer to any extraction activity.

We capture the precedence relationships between panel activities and haulage pillar activities as follows:

1. We group panelblock extraction and void backfilling activities into ordered sets in which the order reflects the precedence rules, e.g., if the order of mining is $A$ before $B$ before $C$, we define the set \{A,B,C\}.

2. Any panel may have more than one ordered set associated with it because, for many panels, multiple activities can be scheduled concurrently.
3. Any panel activity can be a member of multiple ordered sets if that activity requires more than one precedent to finish before the activity can begin (Figure 3.5).

4. We place haulage pillar activities into ordered sets in which the order reflects the precedence rules. Each ordered set represents a section of a critical haulage route. The first member of the set corresponds to the pillar furthest from the mine exit.

5. A pillar activity may belong to more than one ordered set. When critical haulage routes intersect, their corresponding ordered sets will contain the same pillar activities downstream from the point intersection.

6. Each panel ordered set is linked to exactly one pillar. Extraction of this pillar or any other pillar that follows in that pillar’s ordered set destroys access to the panel.

### 3.2.1 Mathematical Formulation

We present the integer-programming formulation for the scheduling problem at weekly fidelity. The decision variables dictate whether or not we begin an activity during a specified time period. The objective discounts metal in an attempt to pull it forward in the schedule. In addition to the precedence constraints, the model enforces resource constraints.

Sets:

- \( \mathcal{A} \): set of all activities \( a, a' \)
- \( \mathcal{E} \): set of all extraction activities
- \( \mathcal{B} \): set of all backfilling activities
- \( \mathcal{F}_a \): set of extraction activities that necessitate backfill activity \( a \)
- \( \mathcal{T} \): set of all time periods \( t, t', \hat{t} \)
- \( \hat{T}_a \): restricted set of time periods in which activity \( a \) can start
Figure 3.5: For this panel, we show in Box (a) that Panelblocks A-F must be mined sequentially; this is also true for L-O. In addition, any panelblock from A-F can be mined concurrently with any panelblock from L-O. However, Panelblocks X-Z must wait until all other panelblocks have been extracted and backfilled (Box (b)). To represent this, we define two ordered sets here: \( \{A, B, C, D, E, F, BF1, X, Y, Z\} \) and \( \{L, M, N, O, BF2, X, Y, Z\} \). Both of these sets are linked to Pillar 1, and once that pillar is extracted, the haulage route above it will no longer be accessible.
• $\mathcal{I}$: groups of ordered sets $i$

• $\mathcal{P}_i$: ordered set $i$ of panel activities, where $(\cdot)_i$ is the $i^{th}$ member of the precedence set $\mathcal{P}_i$

• $\mathcal{H}_i$: ordered set $i$ of haulage pillar extraction activities, where $(\cdot)_i$ is the $i^{th}$ member of the precedence set $\mathcal{H}_i$

• $\hat{\mathcal{P}}_a$: set of haulage pillar activities that must not precede activity $a$, where $(\cdot)_i$ is the $i^{th}$ member of the precedence set $\mathcal{P}_a$

**Parameters:**

• $v_{at}$: volume of ore obtained in time period $\tilde{t}$ given we started extraction activity $a$ at time $t$ (tonnes)

• $g_a$: average percentage grade of the ore produced from extraction activity $a$

• $\bar{e}$: maximum allowable tonnage of ore excavated in a week (tonnes)

• $\bar{g}, \underline{g}$: maximum/minimum allowable metal produced by the mill in a week (tonnes)

• $d_a$: duration of activity $a$ (weeks)

• $p_{at}$: paste applied in time period $\tilde{t}$ given we started backfilling activity $a$ at time $t$ (cubic meters)

• $\bar{p}$: available paste for backfilling in each week (cubic meters)

• $r$: discount rate used to encourage production to shift toward the start of the time horizon

**Binary Decision Variable:**

$$y_{at} = \begin{cases} 
1 & \text{if activity } a \text{ starts during time period } t \\
0 & \text{otherwise} 
\end{cases}$$
Formulation:

\[
(Q) \quad \max \sum_{a \in E} \sum_{t \in \hat{T}_a} \sum_{\tilde{t} \in T : \tilde{t} \leq t + d_a - 1} g_a v_{at\tilde{t}} y_{at\tilde{t}} (1 + r)^{-\tilde{t}}
\]

subject to

\[
\sum_{t \in \hat{T}_a} y_{at} \leq 1 \quad \forall a \in A \quad (3.1)
\]

\[
\sum_{a \in E} \sum_{t \in \hat{T}_a : t \geq \tilde{t} - d_a + 1} v_{at\tilde{t}} y_{at\tilde{t}} \leq \bar{v} \quad \forall \tilde{t} \in T \quad (3.2)
\]

\[
g \leq \sum_{a \in E} \sum_{t \in \hat{T}_a : t \geq \tilde{t} - d_a + 1} g_a v_{at\tilde{t}} y_{at\tilde{t}} \leq \bar{g} \quad \forall \tilde{t} \in T \quad (3.3)
\]

\[
\sum_{a \in E} \sum_{t \in \hat{T}_a : t \geq \tilde{t} - d_a + 1} p_{at\tilde{t}} y_{at\tilde{t}} \leq \bar{p} \quad \forall \tilde{t} \in T \quad (3.4)
\]

\[
y_{at} \leq \sum_{t' \in \hat{T}_{a'} : t' \leq t - d_{a'}} y_{a't'} \quad \forall a, a' \in \hat{P}_i, (\tilde{a})_i \geq (\tilde{a}')_i, t \in \hat{T}_a \quad (3.5)
\]

\[
\sum_{t \in \hat{T}_a} y_{at} = \sum_{t' \in \hat{T}_{a'}} y_{a't'} \quad \forall a \in B, a' \in \hat{F}_a, t \in \hat{T}_a \quad (3.6)
\]

\[
y_{at} \leq \sum_{t' \in \hat{T}_{a'} : t' \leq t - d_{a'}} y_{a't'} + (1 - \sum_{t' \in \hat{T}_{a'}} y_{a't'}) \quad \forall a, a' \in \hat{P}_i, (\tilde{a})_i \geq (\tilde{a}')_i, t \in \hat{T}_a \quad (3.7)
\]

\[
y_{at} \leq 1 - \sum_{t' \in \hat{T}_{a'} : t' \leq t + d_a - 1} y_{a't'} \quad \forall a \in \hat{P}_i, a' \in \hat{P}_i, t \in \hat{T}_a \quad (3.8)
\]

\[
y_{at} \text{ binary} \quad \forall a \in A, t \in T \quad (3.9)
\]

Objective Function

We maximize the discounted quantity of extracted metal over the horizon. (Discounting used as an operations research technique can also improve the run-time performance of mathematical programs (Klotz & Newman, 2013b).) The timing of these extraction activities need only correspond to valid periods based on precedence and resources (see Subsections 3.3.1 and 3.4.1).

We use a parameter, double subscripted by time, \(v_{at\tilde{t}}\), to ensure that an extraction activity beginning in period \(t\) continues at a predefined rate (with predefined consequences)
through subsequent periods until completion. Because our model is deterministic, we assume fixed activity rates; consequently, equal quantities of ore are extracted in each period of the activity’s duration. By restricting the time at which an activity can produce ore to the interval \([t, t+d_a-1]\), we can effectively ensure that we account for the full tonnage of a block.

We categorize our constraints as follows:

**Packing Constraint**

Constraints (3.1) prevent partial extraction or backfilling of activities by ensuring that an activity can only be scheduled to begin in at most one time period.

**Knapsack Constraints**

These constraints are implemented to ensure the demand for resources in any time period does not exceed the availability of those resources. Constraints (3.2) restrict extraction in any time period to be no more than the production capacity for that time period. Production includes extraction of ore blocks and waste blocks and, in practice, is limited by factors such as labor and equipment availability. Constraints (3.3) require the model to maintain grade control at the mill. To operate efficiently, the metal output from the mill must remain within certain bounds. The mill is designed to accommodate maximum production levels from the mine. However, if too much high-grade ore enters the mill, the efficiency of the operation will suffer; consequently, the upper limit on metal production from the mill can be a tighter constraint on extraction activity than the production constraint.

Constraints (3.4) ensure that the paste being used for backfill activities does not exceed its availability. Note that here we also use a parameter, double-scripted by time \((p_{ati})\), to account for the quantity of paste applied in \(\tilde{t}\) given the backfill activity began in period \(t\). We require that for activity \(a\), paste can only be applied within the interval \([t, t+d_a-1]\).
Sequencing Constraints

Constraints (3.5), (3.6), (3.7), and (3.8) enforce the logic of the precedence rules between activities (Figure 3.6). Constraints (3.5) and (3.6) sequence activities in panels. Constraints (3.5) ensure that a panel activity \(a\) cannot begin until all of its predecessor panel activities, \(a' \in \bar{P}_a\), have been completed. Constraints (3.6) ensure that backfilling activity is carried out if it is required. Without this constraint, a backfill activity that: (i) falls at the end of a precedence set, or (ii) is not required as a precedent for any activity, will not be scheduled because backfill activity does not directly contribute to the objective value. However, to maintain structural support, backfilling of specific voids must occur, even if extraction stops in a panel. Extraction of any one of the activities in \(F_a\) forces the inclusion of the corresponding backfill activity in the solution.

Constraints (3.7) ensure that the order of mining along a critical haulage route be maintained, regardless of whether or not all predecessors of a pillar, \(a' \in \hat{P}_a\), have been extracted. Sequencing constraints (3.8) ensure that a panel activity \(a\) cannot occur once a haulage pillar \(a' \in \hat{P}_a\) on which the activity depends is extracted.

3.3 Preprocessing and Other Exact Techniques to Improve Tractability

Instances of \((Q)\) are NP-hard. They are at least as hard to solve as a category of very difficult nondeterministic polynomial time-complete problems (NP-Complete), which, in theory, require exponential time to produce a solution. By rearranging constraints (3.8), i.e., moving the right-hand-side variable term to the left-hand side, we identify sets of set packing constraints: \(\sum_{v \in \bar{T}_{a'}} y_{a'v} + y_{at} \leq 1\). While some set packing problems have a special structure that permits them to be solved easily as network problems, e.g., cyclic staffing (Bartholdi et al., 1980), set packing problems that lack special structure are categorized as NP-complete. Consequently, because a subset of our problem is NP-complete, we can say that the entire problem is at least as hard, or NP-hard (Gary & Johnson, 1979).

On a practical level, the model is complicated by the following factors: (i) the precedence constraints are irregular, following a variety of different logical structures, (ii) the block size is
Figure 3.6: We illustrate the precedence rules that are included in our formulation. In Box (a), constraints (3.5) ensure that Panelblock C can only be extracted if Panelblocks A and B are extracted in advance. Constraints (3.7) ensure that haulage pillars furthest from the mine exit are taken before those that are closer. Box (b) shows that Pillar 2 has been extracted and the roof allowed to cave in; consequently, Pillar 1 can no longer be accessed. Also, the collapsed roof now blocks access to the panel, illustrating the need for constraints (3.8).
extremely heterogeneous, with corresponding extraction times of a few days to many months, and (iii) the ore grade is highly variable, resulting in the employment of three different mining methods. These factors create difficulties both in the nature of the constraints, and in the number of variables per instance, owing both to the number of time periods necessary to capture the required resolution and the inability to aggregate spatially.

Given its difficult mathematical structure, our high-fidelity model quickly becomes intractable as we increase the number of weeks in the time horizon. Consequently, we consider a variety of solution techniques to improve the tractability of our problem. In this subsection, we mention only exact techniques, i.e., those that do not compromise optimality of the solution. Because these are not enough to ensure that our model instances are solvable, we also resort to heuristic (or inexact) techniques, which we detail in the following subsection.

### 3.3.1 Early Start Algorithm for Activities in Panels

An early start algorithm can be used to determine the earliest time period, $t_*$, in which an activity could be begin in any feasible solution. If $y_{at}$ is a binary variable that assumes a value of 1 if block $a$ is extracted at time $t$, then we can use the early start time to restrict $t$ to $t_* \leq t \leq T$. By doing this, we eliminate all instances of $y_{at}$ for which $t$ is an element of $\{1, \ldots, t_* - 1\}$. These restrictions do not compromise the quality of the solution, but only serve to eliminate variables which would necessarily assume a value of zero in an optimal, or even any feasible, solution.

For precedence sets that form a directed, acyclic network, early start times can be found with a Critical Path Method (CPM) (Rardin, 1998). This method is commonly applied to large project management scheduling problems in which all activities, $a$, with duration $d_a$, must be completed. The precedence rules between the activities form a network structure, which can be solved as a longest path problem to produce early start times for each activity.

While our example, shown in Figure 3.7, includes only activities and their durations, there are many variants on the procedure that incorporate resource constraints into the
Figure 3.7: The nodes 1, 2, 3, and 4 represent activities that must be completed in order to finish the project, i.e., the activity represented by node 4. The directed arcs between the nodes represent the precedence rules, and costs on the arcs represent the duration of the activity associated with the node from which the arc originates. There are three paths to node 4: S-2-4, S-1-3-4, and S-2-3-4, with total durations 15, 5, and 17, respectively. The early start time for node 4 is defined by its longest path: \(\text{max}\{15, 5, 17\} = 17\). Therefore, the project cannot be finished before 17+7=24.


For panel activities, we use a variant of an existing procedure for establishing early start times for machine placement activities in a sublevel caving mine (Martinez & Newman, 2011). Our variant considers blocks already scheduled to start, the prescribed sequence in which blocks must be extracted, and mining rates. We assume that the order of mining in a panel is fixed, and we represent this with an ordered set of activities in which no activity can start until its preceding activity is completed. If an activity has more than one precedent, it will be a member of more than one set (see Figure 3.8). The following algorithm applies only to panel activities because of the underlying network structure between their precedences:

Notation

- \(\mathcal{A}\): set of all activities
- \(\mathcal{P}_i\): an ordered set \(i\) of panel activities, \(i = 1, \ldots, n\)
- \(d_a\): duration of activity \(a\)
- \(ES_a\): earliest start time of activity \(a\)
Early Start Algorithm

DESCRIPTION: An algorithm to assign the earliest possible start time that a block may be extracted or a void backfilled.

INPUT: \( A, P_i, d_a \)

OUTPUT: \( ES_a, \forall a \in \bar{P}_i, i = 1, \ldots, n \)

\{
/* Initialize the early start values*/
for (all \( a \in A \)) \{ \( ES_a \leftarrow 1 \) \}
/* Set \( ES_a \) equal to the total duration of all predecessors in same set*/
for \( (i = 1, \ldots, n) \) \{
    for (all \( a \in \bar{P}_i \)) \{
        \[ ES_a \leftarrow \sum_{a' \in \bar{P}_i: (a')_i \geq (a)_i} d_{a'} \]
    \}
/* Loop to update \( ES_a \) from predecessor durations in other sets*/
set changes_made = true
while changes_made \( = \) true \{
    set changes_made \( = \) false
    for \( (i = 1, \ldots, n) \) \{
        for (all \( a \in \bar{P}_i \)) \{
            if \((ES_a < ES_{a'} + d_{a'})\), where \( a' \) immediately precedes \( a \) \{
                \( ES_a \leftarrow ES_{a'} + d_{a'} \)
                set changes_made \( = \) true
            \}
        \}
    \}
\}
\}
Figure 3.8: When an activity has more than one precedent, it will be a member of more than one set. We illustrate this with a network formed by two ordered precedence sets that both contain the same activity. Set 1 = \{A, B, C, D\} and Set 2 = \{H, I, C, J\}. If we consider only the duration of the activities in Set 1, we should assign an early start time of 14 to activity D. However, we must also account for the fact that C is dependent on activities H and I from Set 2 and, thus, cannot start until period 9. Consequently, this later start time for activity C postpones the start time for activity D to period 16.

3.3.2 Tightened Precedence Constraints

When solving integer programs with the branch-and-bound algorithm, performance can be significantly impacted by the quality of the linear programming (LP) relaxation solution that provides a starting point for the branch-and-bound tree. An LP solution with an objective function value that is close to that of the optimal integer solution contains few fractional values, and, of those fractional values, ones that are closer to integers. This implies a smaller branch-and-bound tree for the solver to examine.

For instances of (Q), linear programming (LP) relaxation solutions produce weak bounds that can be up to 15% greater than the objective function value of the optimal integer solution, because, without binary restrictions on the variables, the precedence rules cannot be enforced correctly. Consequently, an LP solution to (Q) schedules haulage pillars for concurrent extraction even though they are members of the same critical haulage route. The resulting solution to (Q) schedules significantly more haulage pillars than is physically possible to extract.
Precedence constraints (3.5) prevent activity $a$ from being scheduled at time $t$ if $a$’s direct predecessor $a'$ is not completed by $t$. This implies that $a$ is also prevented from being scheduled in all periods prior to $t$. As a consequence, we can formulate stronger constraints by summing $y_{ai}$ over all $i \leq t$, as shown in (3.10) below. The additional variables included on the left-hand side produce a stronger constraint that is valid, i.e., no integer solutions are excluded, and can reduce fractionalization in the LP solution, resulting in a stronger initial LP bound (Newman, 2009, 2010):

$$\sum_{i \in \mathcal{I}, i \leq t} y_{ai} \leq \sum_{t' \in \hat{T}_a: t' \leq t - d_{a'}} y_{a't'} \quad \forall a, a' \in \mathcal{P}_i, (\bar{a}_i) \geq (\bar{a'}_i), \ t \in \hat{T}_a. \quad (3.10)$$

Implementing tightened constraints produces models with a denser constraint matrix, so while we begin the branch-and-bound procedure with a better LP bound, progression through the tree is slowed by an increase in computation time at each node. For our problem, the computational cost of implementing tightened precedence constraints outweighs the benefit from having stronger initial bounds. Consequently, we revert to using the original form of the precedence constraints.

### 3.4 Heuristic Solution Methods

When the exact solution methods mentioned above are not sufficiently effective, heuristic techniques offer an additional approach that can be used to produce good solutions quickly. While it may be difficult or impossible to get a measure of the quality of a heuristic solution, the alternative is often to have no solution at all. Even so, we show in Subsection 3.5 that we can bound the quality of our heuristic solutions.

#### 3.4.1 Early Start Determination for Haulage Pillars

As described in Subsection 3.2, pillar extraction precedence rules have more degrees of freedom than those for the panel activities. Specifically, although a pillar’s extraction will preclude all panel and pillar mining upstream from that pillar’s haulage route, pillars do not have any strict precedence. Consequently, we cannot use an exact network-based algorithm
to calculate start time restrictions for the pillars as we did for the panelblocks; instead, we rely on a heuristic to determine early start times for pillars.

We take advantage of the fact that we can solve smaller instances of our problem to optimality to heuristically determine the early start times for the pillars. This heuristic procedure, $\mathcal{V}$, includes all of the pillars, but only a subset of medium-grade panel activities, which we solve for a shortened horizon. Including only medium-grade panel activities not only reduces the model size, but also makes the pillars more attractive and more likely to appear in a solution. We use the date at which the first pillar along a critical haulage route is scheduled in the solution produced by $\mathcal{V}$ as an early start time for all pillars along that haulage route.

Admittedly, using the same early start time for all pillars along a haulage path does not eliminate as many variables as were we to set the early starts based on the start time of each pillar in the solution to the smaller problem. However, it would be overly restrictive to employ the second approach, because we want to provide the solver with the option to take any pillar early, regardless of its location along the haulage route.

3.4.2 Aggregation

Aggregation is a heuristic variable reduction technique, in which adjacent ore blocks are grouped together into a superblock under the assumption that all of the blocks included in the aggregated superblock are extracted if the superblock is extracted. This approach can significantly reduce the number of variables in a model and, consequently, reduce problem size and computational time. While it can work well for strategic production planning over a time horizon of decades or more, the benefits of block aggregation are countered by a reduction in scheduling flexibility which: (i) may preclude potential solutions in which the solver might cherry pick individual higher grade blocks, rather than the superblock itself, and (ii) reduces the resolution at which extraction and blending decisions can be made because the superblock must be mined continuously from start to completion. In addition, block aggregation may not be possible if there are complicated precedence rules, particularly
between blocks in the superblock and their precedences, which are not necessarily spatially proximate.

For our model, the high variation of the grade within adjacent blocks, in addition to the complexity of the mining precedences, preclude block aggregation for mining activities. However, with straightforward precedence rules between each other and no blending to consider, backfilling activities are candidates for aggregation. Aggregating backfill activities is the most efficient and cost effective way of backfilling in practice. For a number of panels, the backfilling of voids can be postponed until the mining sequence has been completed, after which the voids are filled together at the same time. In our model, we replace groups of backfills that can be scheduled consecutively \textit{a priori} with single \textit{superbackfill} activities that have a volume equal to the total volume of its group members.

### 3.4.3 Optimization-Based Decomposition Heuristic

Our objective is to find an optimal production schedule for an operating mine at weekly fidelity to capture tactical decision making. We can solve instances of $(Q)$ for 52 weeks, but these results are not useful to the mine in practice because most of the pillar activities have early start times greater than one year. When the horizon is increased to a length that would produce a practical solution, i.e., two years or more, we have great difficulty finding any feasible solution to our problem, let alone an optimal one, simply by solving the monolith using a state-of-the-art commercial optimizer. Eventually, if we want to examine a mine life scenario of three years, instances of $(Q)$ become too large for our hardware to read in.

Initially, we sought to take advantage of the tractability of model instances with shorter horizons by experimenting with variants of a sliding time window heuristic (STWH) (Brown \textit{et al.}, 2001). With this approach, a subproblem window containing an initial segment of the time horizon is solved. The schedule is then fixed for the initial segment, the window slid forward so that it overlaps the time horizon contained in the previous solution (and variable values in the overlapping portion are allowed to vary), and the next subproblem is solved. The heuristic continues in this manner, fixing variables and sliding, until the end of
the horizon is reached. STWH approaches do produce solutions for our problem relatively quickly; however, the quality of the solutions is poor when compared to the actual schedule used at the mine as a consequence of the greediness of the STWH, which, for horizons longer than 52 weeks, schedules pillars too early and prevents panelblocks that depend on those pillars for access from being scheduled.

A decomposition heuristic that works well for our problem is based on sequentially solving a series of subproblems that: (i) contains only a subset of activities determined on the basis of ore grade in each subproblem; and (ii) considers the entire time horizon. We begin by solving for a schedule that considers only the high-grade extraction activities, i.e., those activities above a specified grade for which their inclusion produces a tractable problem. We then fix this schedule, include activities of the next highest grade, and resolve the model. We repeat this process of fixing variables, adding progressively lower grade activities, and resolving the model until we produce a solution that includes all of the extraction and backfill activities.

We now describe the optimization-based decomposition heuristic that we apply to \((Q)\), the monolith integer program. The goal of the heuristic procedure, which we label \((H)\), is to achieve near-optimal solutions more quickly than by solving \((Q)\) directly, i.e., by executing the procedure \((Q)\). (Note: The use of Roman font in parentheses denotes a problem, while a similar use of the script letter denotes a procedure.)

Our procedure for \((H)\) follows:

**Additional Notation**

- \(k \in \mathcal{K}\): a set of subproblems.
- \(\hat{E}_k\): the subset of extraction activities considered in subproblem \(k\).
- \(S_k\): set of fixed activities that have nonzero variable values in the solution to problem \(k\).
• $t_{ak}$: the time period that fixed activity $a$ was scheduled to start in the solution to subproblem $k$.

• $f, \bar{f}$: used to define a time window for fixed activity $a$, $t_{ak} - f \leq t \leq t_{ak} + \bar{f}$.

**Heuristic ($\mathcal{H}$):**

**Step 1** Define the initial subproblem, $(k = 1)$, to include all backfill activities and a subset of high-grade extraction activities $\hat{E}_1$.

**Step 2** Solve the initial subproblem for the entire time horizon, $\mathcal{T}$.

**Step 3** Define a set $\mathcal{S}_1$ to include activities that have a nonzero variable value in the solution to the initial subproblem.

**Step 4** For subproblems $k \in \mathcal{K}: k \neq 1$,

**Step 4a** Define subproblem $k$ to include all activities considered by the previous subproblem, $k - 1$, in addition to the extraction activities of the next lowest grade.

**Step 4b** Add a constraint requiring that each nonzero variable, $y_{at}$, in $\mathcal{S}_{k-1}$ assume a value of 1 within a restricted time window of the horizon, $t_{a,k-1} - f \leq t \leq t_{a,k-1} + \bar{f}$.

**Step 4c** Solve subproblem $k$ for the entire time horizon.

Each subproblem $k$ considers all backfill activities and a subset of extraction activities, $\hat{E}_k$. We amend our model shown in Subsection 3.2.1 to reflect this change. The objective function for each subproblem reflects this change:

$$(\hat{Q}_k) \max \sum_{a \in \hat{E}} \sum_{t \in F_a} \sum_{i \in T: i \leq t} g_av_{ati}y_{ati}(1 + r)^{-i}$$

Constraints (3.1), (3.2), and (3.3) now apply to the subset $\hat{E}_k$, rather than to the entire set $\mathcal{E}$, and constraints (3.6) now consider only the subset of mining activities, $\mathcal{F}_a \cap \hat{E}_k$. In addition, we modify the precedence constraints to consider only the extraction activities.
within a precedence set that are also in $\hat{E}_k$. Consequently, each subproblem, with the exception of the final one, will produce an infeasible schedule because some required predecessors are missing. Therefore, activities that appear to be skipped because they were not included in a subset must be incorporated into the schedule as the heuristic proceeds through the subproblems. To ensure that space is allotted for the missing activities, we change precedence constraints (3.5) so that the time at which a panel activity can begin is based on the difference between the early start times for panel activities $a$ and $a'$, $ES_{a'} - ES_a$, and not on the duration of the preceding activity, $d_{a'}$.

$$y_{at} \leq \sum_{t' \in \hat{T}_a : t' \leq t - ES_a - ES_a} y_{a't'} \quad \forall a, a' \in \mathcal{P}_i \cap \hat{E}_k \cup \mathcal{B}, (\bar{a})_i \geq (\bar{a'})_i, t \in \hat{T}_a$$

Because the early start times for panel activities are calculated by considering all panel activities, our algorithm, based on the critical path method, ensures that the difference between early start times for any two activities in an ordered set is sufficient to complete all activities between them in the ordered set. We do not make similar changes to constraints (3.7) and (3.8) because they relate pillars to each other and to panels. Exact early start times for pillar activities cannot be calculated using a network-based algorithm, and pillars’ extraction is never required to facilitate further mining. Consequently, we are not required to maintain space for pillar activities in any subproblem to produce a feasible schedule.

In Step 1 of the heuristic $H$, the initial subproblem considers all backfill activities, but only the extraction activities associated with the highest grade of ore. Solving the initial subproblem, Step 2, produces a schedule that forms the backbone for the finished schedule. We require all activities that appear in the schedule at this point to remain in the schedule as the heuristic progresses. Subsequent subproblems merely fit lower grade extraction and related backfill activities around the activities that are already fixed in the schedule. Consequently, we try to obtain as good an initial schedule as possible by including as many activities as possible in the corresponding subproblem. By doing this, we may
attain a higher quality heuristic solution because the subproblem more closely resembles the monolith. We create a set of “fixed” activities, $S_1$, in Step 3, that contains all activities that were scheduled in the initial subproblem solution.

**Step 4** outlines the procedure for the remaining subproblems. In Step 4a, we identify the activities to include in the current subproblem $k$, which include: (i) all backfilling activities; (ii) all previously considered extraction activities; and (iii) a set of lower grade activities that were not previously considered for scheduling. The set $\hat{E}_k$ is comprised of (ii) and (iii). In Step 4b, we ensure that nonzero variables from the previous solution also have nonzero values in the current subproblem by adding the following constraint to the model:

$$\sum_{t_{a,k-1} - \bar{f} \leq t \leq t_{a,k-1} + \bar{f}} y_{at} = 1 \quad \forall \ a \in S_{k-1}$$ (3.11)

In this constraint, the set $S_{k-1}$ consists of the fixed activities from the solution to the previous subproblem, $k-1$. We constrain these activities to appear in the current subproblem solution close to the time at which the activity was scheduled in subproblem $k-1$. We restrict the start times for these fixed activities by creating a time window around $t_{a,k-1}$, the time at which the fixed activity was scheduled in the solution to subproblem $k-1$. We control the size of the window by choosing parameter values, $\underline{f}$ and $\bar{f}$. A larger window size may produce higher quality solutions because it presents the solver with more time periods in which to schedule “fixed” activities over the horizon. However, larger windows generate more variables to be considered by the solver, reducing the tractability of the problem. We solve the current subproblem in Step 4c and repeat Step 4 while additional subproblems remain. The final subproblem considers the entire set of activities and produces a feasible production schedule.

We may encounter an infeasible subproblem if the inclusion of new activities, combined with those already fixed, would require more of a resource than is available, violating one or more of constraints (3.2), (3.3), and (3.4). Widening the windows around the “fixed” variables can induce a feasible subproblem in this case. Otherwise, a strategy of solving
the initial subproblem subject to tighter resource bounds provides flexibility that allows subsequent subproblems to satisfy the resource constraints.

The tractability of each subproblem in the heuristic procedure, \( \mathcal{H} \), like the monolith procedure \( \mathcal{Q} \), is highly correlated with: (i) the number of activities in the problem; (ii) the ratio of pillar activities to other panel activities; and (iii) the number of intervals in the time horizon. Tractable instances of \( \mathcal{Q} \), which we solve for short time horizons with procedure \( \mathcal{Q} \), provide us with an indication of the appropriate size for the initial subproblem of the heuristic. Problem size is largely determined by the number of variables, \(|A| \times |T|\). Consequently, to calculate the number of activities that will produce an initial subproblem of the same size as a tractable \( \mathcal{Q} \), we simply divide the number of variables in \( \mathcal{Q} \) by the number of intervals desired in the heuristic time horizon.

As mentioned previously, the number of pillars included in an instance of the problem significantly influences tractability. Consequently, we try to ensure that a manageable number of pillars are included in each subproblem. We can estimate the number of pillars to add to the data by solving a series of instances of \( \mathcal{Q} \) in which each instance considers more pillars than the previous one. Figure 3.9 shows the results of one such trial, and indicates that the problem becomes difficult to solve when more than 120 pillars are included.

We attempt to solve the initial subproblem for the estimated number of activities and pillars. If a subproblem solves slowly, we reduce the number of pillar and/or panel activities. Conversely, an instantaneous solution indicates that we should add more activities to the subproblem. As we proceed through subproblems, i.e. as \( k \) increases, we consider an increasing number of activities. However, the size of the problem remains manageable because, for “fixed” activities, we only define variables for the number of time periods in their window.

### 3.5 Computational Results

In this subsection, we evaluate the performance of our heuristic \( \mathcal{H} \), described above in Subsection 3.4, in comparison to the monolith solution procedure \( \mathcal{Q} \). We use three real data sets that correspond to different stages in the life of the Lisheen mine. The design of
Figure 3.9: We illustrate the complicating nature of haulage pillars. If we exclude the haulage pillars, we can solve a 52-week instance of our model in a matter of seconds. As we introduce haulage pillars, the solution time increases exponentially until the instance becomes intractable.
the mine was significantly altered between each of these stages, resulting in changes to block shape definitions and, consequently, new extraction approaches, precedences, and rates of extraction. We consider three time horizons for each data set, resulting in a total of nine data sets with which to compare the performance of \((H)\) to \((Q)\). The range of possible outcomes for the problem \((Q)\) is represented in Figure 3.10. We model our integer program using the AMPL programming language, (Fourer et al., 2003) and solve it with the CPLEX solver, Version 12.1 (IBM, 2011) on a machine with two Dual Intel Xeon X5570 Quad Core processors and 24 GB of RAM. For all instances of \((Q)\), we set a time limit of 24 hours.

![Solution Outcomes Diagram]

Figure 3.10: There are four possible outcomes when solving instances of our model. Each outcome is associated with either procedure \((Q)\) or procedure \((H)\). If we can produce an integer feasible solution, then the best theoretical outcome is to solve the monolith to optimality; otherwise, using a heuristic can produce a good solution. If we cannot solve the monolith for an integer-feasible solution, the best we can do is to produce a bound from the LP relaxation solution; if we cannot do that, it indicates that the problem is too large to be read into the solver.

We summarize our computational results in Table 3.1. For each data set, we first list the problem size in terms of the numbers of variables and constraints in \((Q)\) after implementation of the variable reduction techniques mentioned in Subsections 3.3.1 and 3.4.1. Consequently, \((Q)\) and \((H)\) solve identically sized problems. Column 4 compares the quality of the solutions
expressed as a ratio of the objective function value produced by \((H)\) relative to that produced by \((Q)\). Values close to 1 indicate that the heuristic produces a near-optimal solution, and, specifically, a value less than 1 indicates that solving \((Q)\) directly yields a better objective, while a value greater than 1 indicates that the heuristic found a better solution than the monolith due to a gap resulting after the time limit. A value of infinity indicates that, after the time limit, the monolith had found no integer solution. In lieu of obtaining a feasible integer solution, a best bound provides a measure of heuristic performance. Column 5 displays the ratio of the best bound value from \((Q)\) to the best integer solution values from \((H)\). Values close to 1 indicate this solution is close to its theoretical bound. The total solution time for \((H)\) and \((Q)\) are compared in columns 6 and 7. The final column lists the number of subproblems that we solved in an instance of the heuristic.

We attain optimal integer solutions by solving \((Q)\) directly for all three 52-week data sets. Although \((Q)\) produces a better objective value than \((H)\) for two of the three data sets, the objective values resulting from instances solved with \((H)\) are within 1% of the optimal solutions produced by \((Q)\). For all three data sets, \((H)\) produces solutions in under four minutes. By contrast, at best, procedure \((Q)\) requires 13 minutes to solve and, at worst, requires five hours to produce a solution.

For the 104-week horizon, \((Q)\) can no longer identify an optimal solution within the time limit of 24 hours and can only produce an integer feasible solution for data set C2. Solving \((H)\) now requires significantly more time than solving problem instances for the shorter horizon (just over two hours, on average, versus 38 minutes). For these data sets, we use the best bound to compute the quality of the heuristic solutions. For data sets A2 and B2, the best bound corresponds to the tightened root node value, i.e., the objective value of the linear program (LP) relaxation after CPLEX has added a number of cuts. For data set A2, our heuristic produces a solution within 8.8% of a theoretical best possible objective value. We implemented an instance of \((Q)\) with the tightened precedence constraints mentioned in Subsection 3.3.2. However, while the initial bound provided by the LP relaxation was tighter
than before, the solver progressed slowly and did not generate enough cuts at the root node to produce a better bound than the original formulation did after generating cuts.

With a solution within 3.7% of its best bound, the heuristic performs well for data set B2. Although this best bound is also a result of cuts generated at the root node, the cuts appear to be more effective than they were on the preceding data set. We measure the quality of the heuristic solution for data set C2 by comparing it against the best feasible solution (Column 4) and also against the best bound (Column 5) produced by \( Q \). In this case, because the solver outputs a solution for \( Q \) at 24 hours, a gap of 5.49% remains between the best feasible integer solution and the best bound. Consequently, column 4 shows that \( H \) produces an objective value that is better than that provided by the monolith procedure, \( Q \), by 5.4%. Again, the best bound from \( Q \) provides a better measure with which to assess the quality of the heuristic solution. In this instance, the heuristic produces a solution within 0.1% of the best bound.

We cannot bound solutions produced with \( H \) for the 156-week horizon because the model size for \( Q \) exceeds 30 Gigabytes, which is beyond the capability of our hardware. We do solve all three instances of \( Q \) by applying heuristic, \( H \), although the problem size now requires more subproblems to be solved than for shorter horizons and the solve times increase to 40 hours in the worst case.

### 3.6 Conclusions

Our integer programming model considers the strategic implications of extraction activities while scheduling production at a tactical level. As a result, model instances are large and possess excessive solution times for schedules of requisite length. We expedite solutions by: (i) using exact and heuristic methods to reduce the problem size; and (ii) developing an optimization-based decomposition heuristic.

Comparing performance on nine real data sets from the Lisheen mine, the decomposition heuristic produces good solutions significantly faster than the monolith problem. For the three problem instances that the monolith procedure solves to optimality, the heuristic
Table 3.1: Performance comparison of the heuristic procedure, (\(H\)), with the monolith procedure, (\(Q\)), on the basis of solution times and quality. For each data set, we also show the problem size and number of subproblems in (\(H\)). Note: The † symbol indicates that the monolith did not provide a bound.

<table>
<thead>
<tr>
<th>data set</th>
<th>Binaries</th>
<th>Constraints</th>
<th>Solution Quality</th>
<th>Time (s)</th>
<th>Stages</th>
</tr>
</thead>
<tbody>
<tr>
<td>(52-weeks)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(A1)</td>
<td>12,356</td>
<td>129,643</td>
<td>1.000</td>
<td>793</td>
<td>27</td>
</tr>
<tr>
<td>(B1)</td>
<td>15,708</td>
<td>136,850</td>
<td>.994</td>
<td>15,458</td>
<td>217</td>
</tr>
<tr>
<td>(C1)</td>
<td>15,586</td>
<td>30,677</td>
<td>.997</td>
<td>17,989</td>
<td>183</td>
</tr>
<tr>
<td>(104-weeks)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(A2)</td>
<td>52,653</td>
<td>866,195</td>
<td>∞</td>
<td>86,226</td>
<td>7,551</td>
</tr>
<tr>
<td>(B2)</td>
<td>51,190</td>
<td>754,389</td>
<td>∞</td>
<td>86,400</td>
<td>2,321</td>
</tr>
<tr>
<td>(C2)</td>
<td>47,477</td>
<td>242,115</td>
<td>1.054</td>
<td>86,378</td>
<td>2,650</td>
</tr>
<tr>
<td>(156-weeks)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(A3)</td>
<td>100,989</td>
<td>1,873,350</td>
<td>∞</td>
<td>-</td>
<td>145,127</td>
</tr>
<tr>
<td>(B3)</td>
<td>92,967</td>
<td>1,690,596</td>
<td>∞</td>
<td>-</td>
<td>50,112</td>
</tr>
<tr>
<td>(C3)</td>
<td>89,587</td>
<td>635,182</td>
<td>∞</td>
<td>-</td>
<td>5,951</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>((H)) obj. fn. val.</th>
<th>((Q)) obj. fn. val.</th>
<th>((H)) best bound</th>
<th>((Q)) best bound</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(52-weeks)</td>
<td></td>
<td></td>
<td>((Q)) best bound</td>
<td>((H)) best bound</td>
<td></td>
</tr>
<tr>
<td>(104-weeks)</td>
<td></td>
<td></td>
<td>((Q)) best bound</td>
<td>((H)) best bound</td>
<td></td>
</tr>
<tr>
<td>(156-weeks)</td>
<td></td>
<td></td>
<td>((Q)) best bound</td>
<td>((H)) best bound</td>
<td></td>
</tr>
</tbody>
</table>

\(a\) \(Q\) obj. fn. val.
\(b\) \(Q\) best bound
\(c\) Computer run terminated due to memory limitations after 22 minutes without either an integer feasible solution or a bound.
\(d\) Computer run terminated due to memory limitations after 1,320 seconds without either an integer feasible solution or a bound.
\(e\) Computer run terminated due to memory limitations after 1,440 seconds without either an integer feasible solution or a bound.

produces objective values within 1% of optimality and solves the problems, on average, 98% faster than solving the monolith directly. In three more case, the monolith can only provide a bound to compare against the heuristic solution. In only one of these cases is the gap between the bound and heuristic greater than 3.7%; however, this difference does not provide sufficient evidence to question the quality of the heuristic solution because the gap is also a function of the quality of the LP bound. Most significantly, the heuristic can produce practical solutions for datasets that generate a problem size in monolithic form that exceeds the memory capacity of our hardware.

In an attempt to improve the quality of the LP bound for the monolith, we implement tightened precedence constraints. However, the increase in computational time associated
with these denser constraints reduces tractability, producing unsatisfactory results. A potential topic for future research is to implement a “by” formulation (Caccetta & Hill, 2003) for the problem, in which \( y_{at} \) would assume a value of 1 if activity \( a \) is taken by time \( t \). This type of formulation reduces the density of constraints that contain summations of variables over time. However, there is no guarantee that such an approach would expedite solutions because reducing the density of one type of constraint might increase the computational cost associated with other constraints in our model.
CHAPTER 4
APPLICATION OF AN OPTIMIZATION-BASED HEURISTIC TO SCHEDULE
EXTRACTION AND BACKFILL ACTIVITIES FOR A COMPLEX UNDERGROUND
MINE

During the 1980s, surveys demonstrated that the Lisheen ore body, located in south central Ireland, might be profitable; correspondingly, the execution of a drilling program commenced. The seventh hole, drilled in the spring of 1990, revealed nearly 6.5 meters of an ore body containing almost 15% sphalerite (zinc) and 3% galena (lead). By the mid-1990s, after 550 holes had been drilled, a 22.5 million ton deposit, comprised of two separate flat-lying ore bodies of similarly high-grade zinc and lead content, was defined.

In 1999, the first ore was extracted. Since then, the Lisheen Mine has become one of Europe’s largest zinc producers, and, at the time of this writing, over three million tonnes of ore have been extracted from the mine and shipped. As many as 6,300 tonnes of ore are transported to the surface daily. The mine operates six days a week and employs nearly 400 people. Approximately two years of mine life remain, although continued mineral exploration at the fringes of the mine may extend that time frame.

The two ore bodies at Lisheen are partitioned into 11 zones, which are subsequently divided into 88 panels. Within each panel, the ore body is further discretized into (i) stopes, areas outlined solely for the purpose of extracting ore, (ii) drifts, areas that are mined to develop access to a stope and/or for the extraction of ore, and (iii) pillars, large ore blocks that support the mine infrastructure. For ease of presentation, we refer to any stope, drift, or pillar as a block. At the time of this writing, the Lisheen deposit contains 1,193 mining blocks, i.e., candidates for extraction.

A critical factor in planning production is the determination of a profit-maximizing cut-off grade that classifies a block either as ore or as waste based on the percentage of mineral
content or *grade* of that block. Because Lisheen is a polymetallic deposit with a zinc to lead ratio of 5:1, a combined mineral content or *zinc equivalent grade* is calculated for each block. Ore blocks are candidates for extraction and refinement to mineral concentrate, while waste blocks are only extracted to facilitate extraction of adjacent ore blocks and are disposed of underground.

Geotechnical engineers consider the strength of the *host rock* that encompasses the ore body, in addition to other factors such as the ore body slope and thickness, to determine the best extraction technique to employ in a given area of the mine. With varying host rock strength and blocks that range in thickness from 1 to 30 meters, the Lisheen Mine applies three mining methods to extract ore: room-and-pillar, long-hole stoping, and drift-and-fill. Where the host rock is strong and the ore body is not steeply angled, room-and-pillar mining is preferred (Figure 4.1). Areas of the mine where the ore is particularly thick and the host rock is strong are suited to the large-scale and economically efficient long-hole stoping method (Figure 4.2). Finally, where the mine has poor host rock strength, drift-and-fill mining is practiced (Figure 4.3). Figures 4.1 – 4.3 are representative of the operations at Lisheen, but, in reality, the operations are tailored. Long-hole stoping dominates 70% of the extraction, while drift-and-fill accounts for another 20%, and room-and-pillar constitutes the remainder.

With 12 years of production already completed, the mine has an extensive network of *haulage routes* that are used to transport the ore from the panels to the *crusher*, a machine used to break the pieces of ore into manageable sizes before conveyance to the surface. Once above ground, the ore is transported to the *mill* where it is refined into metal concentrate. The operational plan at Lisheen recommends that the mill be run continually at as close to full capacity as possible. Known as “filling the mill,” the requirement is difficult to satisfy because the quantity and the average quality, or *head-grade*, of the ore that enters the mill during a given time period must be blended within a certain range; otherwise, the process stops. Many mining companies stockpile ore to more precisely control the quality and quantity of the ore that feeds into the mill. At Lisheen, a small surge-pile above ground buffers
Figure 4.1: *Room-and-pillar* mining is a self-supporting method, in which pillars of ore are left in place to support the rock that encompasses the ore body, or *hanging wall*, while the remaining ore is extracted. When only pillars remain, a retreat mining method is employed in which pillars are removed in a sequence that starts with the furthest pillar from the mine exit, allowing the host rock to cave in as the extraction proceeds towards the exit. (Hamrin, 1997)

Figure 4.2: With a *long-hole stoping* approach, a sub-level drift is developed below the ore to be excavated. The ore is then drilled and blasted and falls to the sublevel where Load Haul Dump vehicles scoop and load it into trucks for transport to the crusher. (Hamrin, 1997)
ore production to ensure continual mill operation over holiday weekends or during unplanned production shutdowns. However, this stockpile is not suited to blending operations. Therefore, with limited stockpiling capability, mine planners must carefully select the blocks that they schedule for extraction at any one time so that the quantity and average head-grade of the ore reaching the surface is within the acceptable limits of the mill.

After refinement at the mill, the concentrate is loaded on trucks and shipped, and the waste from the milling process, or tailings, are disposed of. Most are deposited into a tailings pond that will be drained and covered once mining is exhausted. Remaining tailings are used to create a cement paste for filling some of the voids left by extraction, a technique known as backfilling.

Aside from providing a location for the disposal of tailings, backfilling is required in some areas to maintain structural integrity and to allow mining to continue. Prior to backfilling a void, it must first be prepared by sealing it with wooden panels and by installing hoses that run from the underground void to the surface. Backfill paste, mixed in a plant above
ground, is then pumped via the hoses down into the area being filled. A void, depending on its size, may require anywhere from a day to a month to fill. Once filling is complete, an additional 24 days are allowed for the paste cement to set, after which time extraction may begin on the adjacent blocks that require the support from the backfilled areas.

![Diagram showing cut-off grade selection](image)

Figure 4.4: Cut-off grade selection determines ore block shape and size. Box (a) shows part of the ore body with a nonuniform distribution of metal. A cut-off grade of 7% results in a large area classified as ore (see Box (b)) that can partitioned into homogeneous blocks and extracted with a single mining method, e.g., drift-and-fill. At a cut-off grade of 9%, the material classified as ore diminishes (Box (c)). Drift-and-fill mining would now excavate too much waste rock so a different mining approach is used. Box (d) shows material classified as ore at a cut-off grade of 12%. With most of the area now defined as waste, a targeted mining method is suitable.

The quality of the schedule for the production process described above is a significant driver of the profitability of the mining operation. To produce a timely and coordinated production plan, the planner at Lisheen must consider the rate of extraction associated with each block’s mining method, the dependencies between activities including backfilling requirements, and the size and grade of each block so that the blend of ore reaching the
mill is satisfactory. This difficult assignment is further complicated because mining follows a number of narrow veins of high-grade ore located along fault-lines between plates of different rock strength. The interaction of these plates has produced an ore body with an inconsistent distribution of metal. Engineers highlight pockets of high-grade material by their choice of cut-off grade and these pockets form the basis for the creation of block shapes and the selection of extraction methods (Figure 4.4). Hence, the grade of a block can vary greatly between adjacent blocks, and it is not uncommon to find a high-grade ore block located beside a waste block. In addition, unlike open-pit mining in which the ore body is subdivided into blocks of equal dimensions, there is great variation in the size and shape of the blocks. As a result, some blocks can be extracted in a day while others require months for their excavation. Consequently, each panel at Lisheen has such specific mining requirements that it is not possible to develop general mining rules, even at the zone level, that would make sequencing activities straightforward.

Schedule creation is made particularly difficult by the inter-temporal effects on future mining activity associated with each extraction decision. These effects become especially important as mining reserves approach depletion and the variety in grade of the ore that is available to satisfy milling requirements also diminishes. With mine closure on the horizon, critical decisions must be made about what blocks should be taken and what blocks should be left behind.

In this chapter, we show how we use integer programming to determine a near-optimal schedule for the Lisheen underground mine for its remaining years of operation. Arguably, the mine would have benefited greatly from an automated scheduling procedure more than a decade ago. However, we only became involved with the project in the past few years, and that involvement stemmed from a weak connection resulting from a fortuitous meeting in South Africa and the coincidental subsequent relocation of the former General Mine Manager of Lisheen from South Africa to Thurles, Ireland. As we describe in this chapter, legacy certainly limits the extent to which we can impact mining operations with our scheduling
procedures. However, we are able to dictate a sequence in which blocks should be extracted to satisfy the profit-maximizing goals for the remaining life of the mine, especially as those goals pertain to extraction of haulage pillars along a critical retreat path out of the mine, and the ability of the mine to sustain an operationally acceptable level of production for as long as possible.

4.1 Production Scheduling at Lisheen Mine

At the Lisheen Mine, ore production will continue as long as the operation is profitable. However, as at other mines, a time will come when the metal produced from the mine will no longer be able to meet the costs of its extraction and, even though ore blocks remain below ground, the mine will close. This time is called economic exhaustion. The challenge then is to extract the combination of blocks that realizes the most value from the mine before economic exhaustion is reached.

![Figure 4.5: A Gantt chart tool is used for manual scheduling at Lisheen. Extraction and backfilling activities are shown as horizontal bars. The thin lines between the bars represent the precedence relationships between the activities. Planners arrange these activities on the timeline in an effort to meet production targets.](image)
Production schedulers at Lisheen use iGantt (MineMax, 2012a) mine planning software to generate production schedules manually. This difficult and time-consuming task, performed semi-annually, requires the manual arrangement of approximately two thousand extraction and backfilling activities on a Gantt chart (Figure 4.5) such that planned ore production is sufficient to keep the mill running as close to capacity as possible.

To narrow the scope of the task, the planner adopts the following assumptions about the mining operation:

1. *Financial aspects of the operation are ignored.* Operational costs, mineral prices, and costs associated with mine closure are not explicitly considered by planners when generating schedules.

2. *The objective is to maximize metal production over the life of the mine.* Although financial information is not considered explicitly by the mine planners, they seek to produce as much metal upfront in the remaining life of the mine as possible. The rationale behind this decision is that, because Lisheen sells metal on the spot market, there is too much risk associated with scheduling large quantities of metal late in the life of the mine when a drop in the spot price might render that metal uneconomical to mine.

3. *The mining methods (room-and-pillar, long-hole stoping, drift-and-fill) for each mining area are fixed.* A change in the method of extraction for an area may require the planner to redefine the ore block shape, establish a new order of mining, and change the rate of mining for that area. As a result, we also assume the same fixed rates for each mining activity that Lisheen has used to generate its manual schedules.

4. *The mine design and infrastructure are fixed.* The planner will not consider developing new access drifts (passageways) to reach the ore blocks or haulage routes to connect panels with the crusher.
5. *The cut-off grade that defines a block as ore or as waste based on the percentage of mineral content is fixed.* A cut-off grade selected to maximize mine value is chosen during the engineering design of the mine. Although economic conditions might indicate that a different cut-off grade may increase the value of the mine, the change would also require a new engineering plan to be developed (Figure 4.4).

6. *The rates for extraction and backfilling activities are fixed.* Based on the block tonnage, excavation distance, or void volume, the planner can quickly calculate the time required for an activity to be completed. This is a standard assumption in most scheduling models, and changing the rates would be tantamount to changing a strategic decision (e.g., fleet size or work shift length).

7. *Sufficient resources are in place to implement the schedule.* Production at Lisheen is not limited by the availability of equipment or labor.

With these assumptions in mind, the mine planner must determine a start date for each extraction and backfilling activity, given the following constraints:

- *Monthly tonnage targets for mine production cannot be exceeded.* The mine has a production capacity constraint for each month based on the number of working areas that can be active at any one time and the number of working days in that month. Mine production includes extraction of ore that feeds the mill and of necessary waste.

- *Metal output from the mill must be maintained between monthly maximum and minimum levels.* These blending constraints on metal production ensure that the quantity and head-grade of the ore feeding the mill are maintained within operational limits.

- *A monthly limit on cement paste for backfilling must not be surpassed.* Paste availability is dependent on the tailings produced as waste by the mill and the capacity of the backfill plant that produces the paste cement. The production of tailings is itself a
function of the quantity and grade of ore reaching the mill, which means that backfilling may be constrained by the production of ore in a previous period. However, because only a fraction of the voids created by extraction need to be filled at Lisheen, paste production is not limited by the availability of tailings.

- **Sequencing constraints must be observed.** The extraction of a block or backfilling of a void must satisfy any sequencing rules that exist between that activity and any other activity. Often called *precedences*, these relationships may be defined implicitly as a consequence of the combination of mining methods chosen for a panel. Otherwise, precedences are defined explicitly to ensure that the schedule remains feasible.

- **Once started, an activity must proceed continuously until completion.** The engineering design defines blocks under the assumption that the entire block will be extracted. Partial extraction of blocks or partial backfilling of voids would result in structural instability.

Manual scheduling of ore production under these constraints is challenging. To satisfy the Lisheen business objectives, the planner adopts a trial-and-error approach to bring metal forward in the production plan while simultaneously adhering to the ore production limits and grade blending requirements. For large problems, the combinational nature of this sequencing and blending problem would require the planner to enumerate a staggering number of alternative scenarios. Consequently, it is not surprising that the planner can have difficulty producing a schedule that satisfies these constraints. Finally, the planner must be mindful that extraction decisions made today can have consequences for future mining activities. For example, tactical scheduling decisions to satisfy the next week’s production quota may unintentionally prevent access to an area of ore and destroy value.

As Lisheen approaches closure, a particularly difficult decision facing the planner is when, if ever, to extract ore blocks that form part of the mine’s critical infrastructure. Throughout the mine, trucks carry ore from the panels to the crusher along haulage routes that are sup-
ported by large ore blocks called *haulage pillars*. Moreover, these pillars are also candidates for extraction, often containing valuable high-grade ore. However, once a haulage pillar is removed, the ore blocks that require the pillar to remain in place for their extraction can no longer be reached and are termed *sterilized reserves*. Hence, the planner must consider the opportunity cost of sterilizing the ore associated with the extraction of a haulage pillar (Figure 4.6).

![Figure 4.6: This example shows a mining panel adjacent to a haulage route. The pillars that support the haulage route are shown as P1 - P5. The arrows indicate the direction of the mine exit. The haulage path collapses west of any pillar that is removed. Hence, the extraction precedence for these pillars would require P1 be taken first and P5 last. Removal of pillar P3, P4, or P5 would result in the destruction of the haulage path in that area, eliminating access to the panel.](image)

Given the challenging nature of the task, manual scheduling can lead to solutions that are far from optimal. Also, the time-intensive process of manual scheduling, which can require several weeks to complete at Lisheen, precludes any possibility of scenario analysis to evaluate the operation with respect to other parameter values, e.g., realization of poorer-than-expected ground quality in a panel that would slow down mining in that area and necessitate an update of the mining rates for that zone. Finally, because the planner cannot examine all of the various permutations of the schedule, there is no way for him to measure the quality of his solution.
By contrast, a mathematical optimization approach offers the ability to enumerate a number of schedules, to select the one that results in the highest objective value, and then show a measure of the solution quality. Moreover, in comparison to the manual approach, an integer-programming solution can be produced relatively quickly to address any changes that may arise due to unexpected events.

4.2 Integer-Programming Approach

Adopting an integer-programming approach, we determine a near-optimal schedule for the Lisheen mine. We cast the problem mathematically, with the objective to recover the maximum amount of discounted metal possible from the mine over its life. We also incorporate the same constraints and assumptions mentioned previously in relation to manual scheduling.

We apply a small discount factor to the objective solely to encourage a solution in which metal production is brought forward in the schedule. Specifically, without this discount factor, a solution with one tonne of ore extracted either in week 1 or in week 2 would be equivalent. The addition of the factor produces a solution with the tonne of ore extracted in the first week. Such discounting can also improve the run-time performance of mathematical programs (Klotz & Newman, 2013b).

To measure value, we assume a constant metal price, which we normalize to 1 (see the formulation in the appendix, Subsection 4.8. Note: This is an older formulation that is more specific to Lisheen than the one presented in Chapter 3). Multiplying the objective through by a constant factor would not change the solution we produce, and would obfuscate our intent, which is to pull the blocks with the highest metal content forward in the schedule as a means to realize the greatest metal sales (subject to operational constraints of the mine) in the shortest amount of time possible. Indeed, lack of an explicit monetary factor is not unheard of in a base metal extraction operation (Newman & Kuchta (2007), Martinez & Newman (2011)).
Although relatively straightforward to describe in mathematical terms, the complexity of the problem makes it difficult to solve in practice. The sequencing rules that govern the relationships between activities are idiosyncratic, complicated, and not clearly defined. In describing a challenge of solving integer programs for mine scheduling, Smith (1998) emphasizes the difficulty associated with complex precedence constraints. These precedences - encoded in iGantt during manual scheduling - are provided as an output from that software. However, the iGantt precedence set provides few degrees of freedom, and solving our integer program with these encoded precedences would simply return the existing iGantt schedule. Inspection of the iGantt relationships using a computer-generated block model in Vulcan (Maptek, 2012) reveals that, although many rules are required to maintain a feasible schedule, some rules merely reflect a subjective choice in the order of mining certain ore blocks. Specifically, the precedence rules relating the extraction of infrastructural haulage pillars to other activities in dependent panels are unnecessarily restrictive. In most cases, the existing mining rule simply reflects the order in which the pillar was previously scheduled, often requiring extraction to wait until all dependent panel activities have finished. As we mentioned previously, the timing of the extraction of these high-grade haulage pillars is a critical aspect of the mine schedule. Hence, we want to allow our model the freedom to choose when, if ever, to extract them. To this end, we reverse the existing logic, that a pillar must wait until the mining of the dependent panel has finished, and instead create a rule that prevents any further mining in the dependent panel once the haulage pillar has been extracted. For each panel, we identify the critical haulage pillar whose extraction would prevent any further activity in that panel. We then define a constraint that enforces this relationship, not just for that one critical haulage pillar, but for all blocks along the haulage route that, if extracted, would sterilize remaining ore in that panel. In this way, we delineate critical haulage routes throughout the mine that characterize the relationships between haulage pillars and mining panels (Figure 4.7).
Figure 4.7: The dashed lines show *critical haulage routes* through a mining zone. These haulage routes are bordered by major panels containing blocks that are candidates for extraction. The pillars that support these haulage routes often consist of valuable ore blocks. However, once removed, we lose access to the area east of the extracted pillar.
Another difficulty in constructing the sequencing constraints arises from activities for which no sequencing rules are defined. Including these orphaned activities in our model without defining rules for their execution would likely produce an integer programming solution that could not be implemented. Without explicit and objective mining rules, considerable effort must be spent examining each precedence relationship between a pair of activities to determine if it leads to a valid constraint or whether it is merely a subjective planning choice. Working with Lisheen, we redefine the precedence set by identifying and removing the subjective rules and by defining new precedents where needed.

A complication with our integer-programming approach also arises from the heterogeneity of the ore blocks. The size of an ore block can vary greatly, which makes it difficult to determine a standard time interval, or fidelity, for the problem. The fidelity of the model defines the discrete times during which an activity can begin. In scheduling, we could assign an activity to start at a certain month, day, or hour. However, use of discrete time intervals results in schedules in which activities that, in practice, would take less than the interval to complete appear to require the full time interval before a dependent activity could begin. For example, a block that would require a day to extract in practice would be scheduled for a month if we were solving at monthly fidelity. Thus, the smaller the time interval, the closer together we can schedule activities. But, this benefit comes at a computational cost, because we must account for finer fidelity by defining variables for each activity and start-time combination.

A solution produced at monthly fidelity may work well for a mine that has homogeneous, large blocks which require more than a month to extract, but as the histogram in Figure 4.8 illustrates, at Lisheen there are many small areas that can be mined in less than a day. As a result, applying a model with monthly fidelity yields delays in the schedule because these small areas would be allotted a month for extraction and would thus unnecessarily extend the extraction times of dependent areas. An often used approach to address such fidelity problems is to aggregate the smaller mining areas into larger clumps of similar size.
However, such an approach reduces the resolution at which blending decisions can be made. And, as Smith (1998) points out, deposits rarely contain grade that is homogeneous enough to ignore grade blending requirements. Such is the case at Lisheen, where the high variation of the grade within adjacent blocks, in addition to the complexity of the mining precedences, preclude block aggregation.

![Histogram of block size at Lisheen](image.png)

Figure 4.8: In this histogram of block size at Lisheen, smaller blocks correspond to a small access drift or a cross-cut used to facilitate extraction of a larger block, such as a stope. The smallest blocks can be mined in a day or less. The largest require months to extract.

To produce an implementable schedule, we require a solution at weekly fidelity for a two-year time horizon, i.e., a horizon that corresponds to the projected life of the mine. The resulting problem contains 2,214 (extraction and backfilling activities) \* 105 (weeks) = 232,470 binary variables. We reduce this number by implementing start date restrictions for each activity; these restrictions do not compromise the quality of the solution, but only serve to eliminate variables which would necessarily assume a value of zero in an optimal, or even any feasible, solution. The manual schedule in iGantt provides a basis from which we develop early start time restrictions for haulage pillar retreats. However, because we
relax precedence rules between panels, using the same approach for panel activities is too restrictive. Consequently, for panel activities, we use a variant of an existing procedure for establishing early start times for machine placement activities in a sublevel caving mine (Martinez & Newman, 2011). Our variant considers blocks already scheduled to start, the prescribed sequence in which blocks must be extracted, and mining rates to determine that certain blocks cannot be extracted before a specific early start time. For example, if the following three conditions hold: (i) block A must be mined to obtain access to block B; (ii) block A has started to be extracted in time period 1; and (iii) block A requires two weeks to extract, then we cannot begin to extract block B until at least the start of the third week. Correspondingly, we remove variables from the formulation that represent block B starting to be mined either in week 1 or in week 2. For sequences involving more than a single predecessor block, we simply add the durations of all predecessor blocks to the start date of the first block in the sequence to determine the start date of a given block; in these cases, all blocks in the sequence must be mined, and at a predetermined rate, which produces an exact result, i.e., a time before which a block at the end of a precedence chain cannot possibly begin to be extracted. This approach reduces the problem to 56,276 variables, but, with over one million constraints, the problem remains intractable. Therefore, not only do we use the exact approach of eliminating variables to reduce the problem size, but we also employ a heuristic on the reduced problem to produce a schedule in a reasonable amount of time (Figure 4.9).

We can solve smaller instances of the (reduced) problem by including fewer activities in the model. We take advantage of this insight to find a solution that includes all of the activities by decomposing the original problem into a series of smaller problems that we solve in stages. With this approach, we begin by separating the extraction activities into sets of progressively lower metal content. In practice, we might use two or three sets for panel activities and three to five sets for pillar activities, but to illustrate the approach here, we consider only two sets: “high-grade” and “low-grade” activities. Considering only the
subset of “high-grade” activities, we solve the model \((P)\), given in the appendix, for those activities at a weekly time fidelity for all weeks in the planning horizon. The solution yields a production schedule for the “high-grade” activities on a weekly basis. We then include the set of “low-grade” activities in the next model run, fixing those “high-grade” activities to the corresponding weekly start dates from the first solve. The net result is that we now obtain a solution which includes some subset of the “low-grade” activities, which are scheduled around the fixed “high-grade” activities. Note that in order to enforce precedence between successive solves, we must insert placeholders for the time required to extract the predecessors absent in the current solve whose successors are present in the current solve.

Figure 4.9: We illustrate our optimization-based decomposition heuristic with an simple example in which we solve the problem twice at weekly fidelity. \textit{Solve 1} shows the solution for a twelve-week horizon during which only high-grade panel and pillar activities are considered for scheduling. The times at which these blocks are extracted constitute the schedule at this point. The following solution, resulting from \textit{Solve 2}, shows that blocks that were scheduled within \textit{Solve 1} are now required to remain scheduled at the equivalent week when solving this problem, but now low-grade panel blocks and pillars are also included.

This approach alone will not produce a near-optimal solution for Lisheen because the complexity of the problem is largely determined by the number of haulage pillars included.
in the model. Haulage pillars are complicating because: (i) they have few, if any, precedence requirements and consequently can be scheduled for extraction at any point during the time horizon; and (ii) the value of adding a haulage pillar to the schedule must be weighed against the total value of the dependent panel blocks that would be sterilized as a result. If we exclude the haulage pillars, we can solve the model for the entire time horizon in a matter of minutes; however, as we introduce haulage pillars, the complexity of the problem quickly increases until it becomes intractable. The key to solving the problem in stages is to find the right balance between the number of pillar activities and the number of other extraction and backfilling activities included in each stage of the heuristic. If too few pillars are included in the initial solve, then that solution contains too much panel mining in the schedule, leaving less opportunity to accommodate remaining pillars during the next solve stage. If too many pillars are included, the problem becomes intractable, or, in the tractable case with fewer panel blocks, the solution contains some pillars that are scheduled far too early, sterilizing dependent panel blocks and preventing them from entering the schedule during the next stage of the heuristic. For a given horizon, we use trial-and-error to determine the number of activities and the ratio of pillar-to-panel activities to include at each stage in the heuristic. The number of stages is dependent on the length of the time horizon, with longer time horizons requiring a more gradual introduction of pillars and, thus, more stages.

4.3 Implementation

Our schedule, as it appears in the output file from the CPLEX solver (IBM, 2011), is no more than a list of scheduled activities and their corresponding start weeks. In this format, the schedule is difficult for Lisheen to validate and impractical to implement. Therefore, we integrate our solution with Lisheen’s iGantt scheduling software to present our schedule in a way that is familiar to Lisheen.

iGantt provides effective schedule visualization and reporting features that are used at Lisheen on a daily basis. The software displays the schedule graphically as a Gantt chart (Figure 4.5), i.e., a timeline of scheduled activities and their interdependencies. In addition,
iGantt can present a 3-D block model animation of the schedule, enabling the planner to quickly identify sequencing problems or other infeasibilities. Management at Lisheen use iGantt’s reporting capability extensively, for example, to produce short-term work schedules, to summarize historical backfill paste usage, or to forecast future resource requirements, e.g., the number of cable bolting bits that will be needed. Thus, implementing our schedule using iGantt has the added benefit of providing continuity in the daily managerial operations at the mine.

We transfer our schedule to iGantt by creating an import file of the data, with rows of activities and columns of activity attributes, e.g., tonnage associated with an extraction activity. To this file, we add a start-date column which we populate with the week during which each activity begins according to our solution. Importing this information into iGantt as a comma-separated-value (csv) file is straightforward, but the resulting schedule that iGantt creates is not correct because the default functionality of the software chooses a start date for each activity that differs from the integer program schedule’s start date. We override these automated start dates by using a customized script that forces iGantt to adopt the integer-programming activity start dates which are given in terms of a start week, whereas iGantt’s time fidelity is given in milliseconds. We convert the time at which an activity starts based on the beginning of the week in which it is scheduled in the integer program to milliseconds from the beginning of the horizon, and use that converted date in iGantt. We do not specify activity end-dates to import into iGantt; instead, we let iGantt set them based on the activity rates. Admittedly, we are not scheduling at the level of fidelity that iGantt allows or that manual schedulers at the mine could use with the iGantt software. However, mine planners can use discretion to shift our dates within an allowable window, and a schedule at the level of milliseconds is impractical at any rate. Shifting activities outside of the allowable window, i.e., manually moving activities in iGantt forward to begin at the end-date of the preceding activity, would likely violate ore production and/or milling constraints. Although our schedule may leave gaps in time, e.g., an activity that requires five days is recorded in
our model as requiring a week, our schedule provides a conservative estimate of the metal producible within the given time horizon, and may be more realistic under the unanticipated conditions that necessarily prevail in an underground mining operation.

Review and validation is possible once our solution is formatted into an iGantt schedule. Using the iGantt visualizer, the planner examines our schedule for feasibility. At this point, our schedule may be infeasible from a practical standpoint because of missing or invalid sequencing constraints. The complexity of the sequencing rules makes it difficult to spot minor conflicts or omissions \textit{a priori}, but these mistakes are highlighted when the schedule is animated in iGantt. With guidance from Lisheen, we correct any illogical or overlooked sequencing constraints in our integer program and continue the process of review and correction until a workable schedule is obtained.

### 4.4 Results

Solving our integer-programming model for a 104-week horizon, we compare our solution with the manually generated schedule on the basis of, among other things, metal output (from the mill), ore production (based on extracted material), and backfill paste usage (Table 4.1). With respect to the objective of maximizing metal production over the life of the mine, the integer program solution shows 0.17\% less metal assigned for recovery than the manual approach. When taken in isolation, this shortfall in total metal output is indicative of a poor result. However, there are a number of other measures to be considered before the quality of the integer-programming solution can be properly assessed.

<table>
<thead>
<tr>
<th></th>
<th>Man Sch.</th>
<th>IP Sch.</th>
<th>IP Gain</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metal Output (Tonnes)</td>
<td>321,154</td>
<td>320,621</td>
<td>-0.17%</td>
<td>No IP improvement</td>
</tr>
<tr>
<td>Ore Production (Tonnes)</td>
<td>2,444,446</td>
<td>2,414,298</td>
<td>-1.23%</td>
<td>IP improvement</td>
</tr>
<tr>
<td>Backfill Paste Used (cu.m)</td>
<td>947,104</td>
<td>810,273</td>
<td>-14.45%</td>
<td>IP improvement</td>
</tr>
<tr>
<td>Waste Mined (Tonnes)</td>
<td>115,790</td>
<td>114,424</td>
<td>-1.18%</td>
<td>IP improvement</td>
</tr>
</tbody>
</table>
Most significantly, the integer program discounts future production so that metal is brought forward in the schedule. This satisfies the management’s desire to produce as much metal upfront in the remaining life of the mine as possible. As mentioned previously, Lisheen management are not comfortable with scheduling large quantities of metal late in the life of the mine because a low future spot price might preclude economic viability of the metal’s extraction. While the integer program may not increase total metal production, it does bring a significant quantity of metal forward in the schedule, increasing metal production by 10.51% in the first year (Figure 4.10).

Following the integer-programming schedule would also reduce costs because the decreased metal yield is accompanied by an even greater reduction in mining activity, with 1.23% less ore production and 1.18% less waste mining than in the manual schedule. In this way, the integer program schedules more efficient extraction than the manual scheduling method. The integer program also schedules significantly fewer backfill activities, corresponding to a 14.45% reduction in paste consumption over the remaining life of the mine. By releasing labor to more profitable extraction activities and by reducing wait time for backfill set-up, pouring, and setting, this reduction in backfilling enables the planner to include valuable extraction activities that would otherwise have been precluded. In addition, while a backfilled void generally requires one month to set, at Lisheen, oxides often interact with the backfill paste, preventing or delaying hardening. Consequently, backfilling can sometimes fail to harden or can require two months or longer to complete, again forcing extraction activities out of the schedule, resulting in a loss of value. Thus, scheduling less backfill activity reduces the potential for delay in the schedule.

When concurrently scheduled, backfill activities can also be responsible for large temporary drops in ore production which, for a number of reasons, are problematic. In particular, because it is highly inefficient for the mill to operate below a minimum ore threshold, low production levels result in mill shutdowns and costly restarts. Planners at Lisheen use trial-and-error to shift production activities in the manual schedule in an effort to boost low
Figure 4.10: The integer program brings metal forward in the production schedule.

production periods. However, with this approach, the planner cannot be sure if the drop in production can be rectified by rescheduling alone. For Lisheen, production below approximately 80,000 tonnes of ore results in mill shutdown, and the manual schedule falls below this level for two months at the start of the second year (Figure 4.11). By contrast, the integer-programming model sets a lower bound for production and, as a result, suffers no mill-halting drops in ore production. Also, with lower production requiring fewer workers, management would be forced to reassign miners to other duties or worse, lay off part of their workforce, a decision that would bring union action against Lisheen.

Not only does the integer-programming schedule avoid dips in production, but it also schedules ore production more consistently than the manual method, with an average month-to-month change in production of 10,132 tonnes compared with 14,796 tonnes for the manual schedule. Thus, the integer-programming solution enables the mine management to create a more steady work-flow than that given by the manual plan.
Figure 4.11: The integer-programming solution has more consistent month-to-month production than the manually generated schedule. In addition, production in the integer program schedule remains above 80,000 tonnes, allowing the mill to operate continually. By contrast, the manual schedule suffers from low production starting in week 48, which would result in temporary mill shutdown and labor problems.

4.5 Scenario Analysis

The integer program produces schedules in fewer than 20 hours. This is significantly shorter than the manual approach, which can require several weeks to complete. Consequently, the integer-programming method can be used for scenario analysis or to expedite the generation of new schedules to account for unforeseen events or changes in engineering design. With the mine approaching the end of production, we examine one important set of scenarios: alternative closure dates.

The closure date that defines the end of the mine’s operational life is a best estimate from the manual scheduling process. It is the point in the schedule at which economic exhaustion occurs and the mine can no longer produce enough metal to meet the costs of the operation. The time at which Lisheen reaches economic exhaustion depends on future metal prices and costs, including significant mine closure and rehabilitation expenses. While our model does not explicitly incorporate these financial components, it can be used as a tool to generate feasible production alternatives, thus enabling management to understand how the schedule
might change if the mine life were shortened or lengthened, and to make more informed mine closure decisions. To provide insight, we run scenarios with different time horizons (Figure 4.12).

Table 4.2: A comparison of total metal production scheduled by the integer-programming mine life scenario solutions.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>92 Weeks</th>
<th>104 Weeks</th>
<th>122 Weeks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Short Horizon</td>
<td>280,095</td>
<td>n/a</td>
<td>n/a</td>
</tr>
<tr>
<td>Life-of-Mine Base Case</td>
<td>288,165</td>
<td>320,621</td>
<td>n/a</td>
</tr>
<tr>
<td>Long Horizon</td>
<td>279,532</td>
<td>311,511</td>
<td>342,614</td>
</tr>
</tbody>
</table>

With our previous results for a 104-week mine life providing a base case, we solve our integer program for two mine-life scenarios: (i) an early closure option with a 92-week horizon; and (ii) an extended horizon of 122 weeks. We compare the results of these scenarios to our base case on metal production (Table 4.2).

The early closure scenario is motivated by the observation that after the first 20 weeks of the 104-week integer-programming schedule, ore production and metal output never approach maximum levels again. Consequently, management at Lisheen are interested in examining the possibility that, if there is sufficient production or output slack in the base-case schedule, a solution for a shorter time horizon might produce a similar or greater quantity of metal than the base-case scenario. Specifically, the horizon was shortened to 92 weeks because it corresponds to the end of a calendar year. Simply taking the base-case solution for the first 92 weeks would produce an infeasible schedule because it would omit the backfill activities required before closure. The schedule optimized over the 92-week horizon precludes these infeasibilities, and moves a small quantity of metal forward in the schedule. However, with this schedule, the total metal produced is 8,070 tonnes lower than what is scheduled over the same time frame in the base-case solution. The fact that shortening the horizon does not increase scheduled metal illustrates the limited production flexibility available at Lisheen. With no significant gains using the early closure option, we turn our attention to
The 122-week extended mine life scenario proves to be a more interesting case. While scheduled metal production is 9,109 tonnes lower than the production corresponding to the base case at week 104, the extended schedule includes 21,993 tonnes more metal than the base case by the end of its horizon. However, this gain in metal is accompanied by two additional periods of low metal production. Although these production drops are not low enough to stop the mill, and implementing the schedule would delay mine closure, the results do show that there is a trade-off for management to consider between: (i) the gains from a longer mine life with higher total metal production; and (ii) the early closure risks associated with dips in metal production and potentially low metal spot prices in the future.

Figure 4.12: We compare integer-programming solutions for varying time horizons against the integer-programming solution for the current mine life. Increasing the life of the mine by another three months is a viable option for Lisheen. Similarity in the production tonnage in each week during the first year is a consequence of the low degrees of freedom in the precedence rules. Later, when haulage pillars become candidates for extraction, weekly metal production differs more across scenarios, reflecting the more flexible precedence constraints on these pillars in our model.

While our model can solve a new scenario instance in less than a day, additional time is required to make data changes, prepare the model, and import our solution into iGantt. Parameter values can be altered easily and quickly, but changing precedence relationships between activities, a more involved task, can take a number of days to update with our
approach. For example, a change in the cut-off grade would require engineers to reexamine and, in some cases, redefine block shapes, mining methods, and precedence relationships between activities. Upon completion of this study, we would require up to a week to make the corresponding changes. For a commercial application, this preparation time could be eliminated by writing a software interface to instantly transfer changes made in iGantt to our integer program model. Also, with the addition of a graphical user interface to the optimizer, non-technical users could perform scenario analysis on production scheduling. However, as our academic scope includes limited scenario analyses, developing a commercial interface for our model would be tangential to our study.

4.6 Insights

With the integer-programming results in hand, the planners at Lisheen can see that the current mine design and precedence relationships do not allow for significant metal gains from scheduling changes alone. In particular, the integer-programming schedules highlight the valuable blocks that would always be excluded from an economic production schedule, and no amount of manual rearranging of activities in iGantt would bring these blocks forward into the schedule. This insight has prompted planners to explore alternative methods for capturing these high-value blocks. This can involve a change in the engineering design, such as selecting a different mining method, developing new access, or even blasting a new haulage route through a previously backfilled area (Figure 4.13).

For the planners, the integer-programming solutions also highlight myopic and subjective decisions that were made during manual production scheduling. As an example, the integer program scheduled an area of very low-grade ore that Lisheen had decided was uneconomical to mine. What planners had overlooked was that occasionally production from a number of very high-grade blocks in different panels was scheduled simultaneously, resulting in too high a head-grade for ore feeding the mill. To maintain feasibility, low-grade ore is, at such times, required to balance the head-grade. Additionally, there was a large drop in ore production at the start of the second year in the manual schedule. The integer-programming
Figure 4.13: The integer-programming schedule identifies four high-grade haulage pillars as blocks that would never be mined given the current mine design. The pillars form part of an existing main haulage route (solid arrows) connecting the crusher with a large ore zone to the west (not shown). Under the original plan, mining these pillars would only be possible once all activity in the western zone had finished. However, the integer-programming results show that more profitable higher-grade pillars along the same haulage route would always be extracted in preference to these four pillars. Realizing this, Lisheen added a bypass (dashed arrows) to incorporate these pillars into the schedule.
schedule brought forward a retreat of haulage pillars that were to be extracted late in the manual schedule to cover this deficit. The planner had not considered using these pillars to prevent the production drop because of subjective reliance on a previous mine design. The mine configuration had since included new haulage routes and precedence rules, but the mental heuristic that those pillars could not be taken early remained in the planner’s mind, preventing him from determining a solution for the shortfall in production.

Insights gained from the integer-programming results have prompted Lisheen to take action immediately. In particular, the integer-programming schedules have highlighted a number of panels that need to be brought into production earlier in order to pull dependent blocks into the mining schedule. These areas are currently being mined in accordance with the integer-programming schedule. Lisheen is in the process of changing the engineering design in many areas of the mine. When these designs are complete, the changes will be incorporated into the integer-programming model and a new integer-programming schedule generated. In this way, our work with Lisheen will continue to be an iterative process between mine design and schedule optimization.

4.7 Conclusions

Our integer-programming approach produces near-optimal production schedules for a complex underground mine. Over a 104-week horizon, our schedule contains ore production that is more consistent with managerial goals than the previous manual scheduling method used at the mine. While both the manually produced schedule and the integer programming-produced one appear in iGantt for presentation purposes, our integer-programming schedules add value to the mining operation by: (i) shifting metal production forward in the schedule; (ii) reducing waste mining and backfilling delays; (iii) avoiding expensive mill-halting drops in ore production; and (iv) enabling smoother workforce management. In addition, the integer program provides the mine with a scenario analysis capability which has been instrumental in the management’s decision to revise much of the underground engineering design in an effort to increase the operational life of the mine.
While multiple open-pit optimization software packages are on the market, the idiosyncrasies of underground operations have, so far, made the creation of such a general software package for underground mining too challenging. However, the approach that we outline here can certainly be adapted to other room-and-pillar underground operations, especially as they near the end of their operational lives.

4.8 APPENDIX A - Model Formulation

The primary elements of the model are as follows:

Indices

- $a$: mining or backfilling activity, $a = 1, \ldots, n$
- $t, \hat{t}$: time period, $t = 1, \ldots, T$

Sets

- $\mathcal{A}^M$: set of all extraction activities
- $\mathcal{A}^B$: set of all backfilling activities
- $\mathcal{T}$: set of all time periods
- $\hat{T}_a$: restricted set of time periods in which activity $a$ can start
- $\overline{P}_a$: set of activities that must precede activity $a$
- $\hat{P}_a$: set of activities, each of which must precede activity $a$, if it occurs
- $\hat{P}_a$: set of activities that must not precede activity $a$

Parameters

- $v_{at\hat{t}}$: volume of ore obtained in time period $\hat{t}$ given we started extraction activity $a$ at time $t$ (tonnes)
- $g_a$: average percentage grade of the ore produced from extraction activity $a$
• \( \bar{e}, \underline{e} \): maximum and minimum allowable tonnage of ore excavated in a month, respectively (tonnes)

• \( \bar{g}, \underline{g} \): maximum and minimum allowable metal produced by the mill in a month, respectively (tonnes)

• \( t_{ma} \): number of time periods required for extraction activity \( a \) (months)

• \( t_{ba} \): number of time periods required for backfilling activity \( a \) (months)

• \( p_{ati} \): paste applied in time period \( \bar{t} \) given we started backfilling activity \( a \) at time \( t \) (cubic meters)

• \( \bar{p} \): available paste for backfilling in each month (cubic meters)

• \( r \): discount rate used to decrease the value of the metal produced in future periods

Decision Variable

• \( y_{at} = \begin{cases} 1 & \text{if activity } a \text{ starts during time period } t \\ 0 & \text{otherwise} \end{cases} \)

Objective Function

\[
(P) \quad \max \sum_{a \in A^M} \sum_{t \in T_a} \sum_{i \in T} g_{ai} v_{ati} y_{at} (1 + r)^{-i}
\]

Subject to
\[
\sum_{t \in \hat{T}} y_{at} \leq 1 \quad \forall a \in \mathcal{A}^M \cup \mathcal{A}^B \tag{4.1}
\]
\[
\xi \leq \sum_{a \in \mathcal{A}^M} \sum_{t \in \hat{T}} v_{ati} y_{at} \leq \zeta \quad \forall i \in \mathcal{T} \tag{4.2}
\]
\[
g \leq \sum_{a \in \mathcal{A}^M} \sum_{t \in \hat{T}} g_{ati} y_{at} \leq \gamma \quad \forall i \in \mathcal{T} \tag{4.3}
\]
\[
\sum_{a \in \mathcal{A}^B} \sum_{t \in \hat{T}} p_{ati} y_{at} \leq \rho \quad \forall i \in \mathcal{T} \tag{4.4}
\]
\[
y_{at} \leq \sum_{u \in \hat{T}_{a'}: u \leq t - t^a_{m} - t^a_{b}} y_{a'u} \quad \forall a' \in \hat{P}_a, a \in \mathcal{A}^M \cup \mathcal{A}^B, t \in \hat{T}_a \tag{4.5}
\]
\[
y_{at} \leq \sum_{u \in \hat{T}_{a'}: u \leq t - t^a_{m} - t^a_{b}} y_{a'u} + (1 - \sum_{u \in \hat{T}_{a'}} y_{a'u}) \quad \forall a' \in \hat{P}_a, a \in \mathcal{A}^M \cup \mathcal{A}^B, t \in \hat{T}_a \tag{4.6}
\]
\[
y_{at} \leq 1 - \sum_{u \in \hat{T}_{a'}: u \leq t + t^a_{m} + t^a_{b}} y_{a'u} \quad \forall a' \in \hat{P}_a, a \in \mathcal{A}^M \cup \mathcal{A}^B, t \in \hat{T}_a \tag{4.7}
\]
\[
y_{at} \text{ binary} \quad \forall a \in \mathcal{A}^M \cup \mathcal{A}^B, t \in \mathcal{T} \tag{4.8}
\]

We present the integer-programming formulation for the original problem at monthly fidelity. The decision variables dictate whether or not we begin an activity during a specified time period. In theory, i.e., in iGantt, these activities are scheduled at the beginning of the time period. However, because most activities do not consume an integral number of time periods to complete, in practice, the activity can begin at any point during the time period in which it is scheduled to start, as long as it is finished in an integer number of time periods no larger than the ceiling of the activity’s duration. Our variable definition necessarily discretizes time, and at a fidelity that is arguably coarser than that at which the mine would implement the schedule. However, we require binary variables to enforce the precedence constraints, and the number of these variables is proportional to the product of the number of activities and the number of time periods we consider. A finer time fidelity (approaching a continuum) might more closely approximate the way in which mine operations are conducted, but would render the model so intractable that it would produce no results quickly enough to be practical. Although not mathematically portrayed here, we can reduce
the number of variables (from the product of the number of activities and the number of
time periods we consider) by employing early start times for each activity, as discussed in
the subsection “Integer-Programming Approach.”

We maximize the discounted value of extracted metal over the horizon, where we discount
consistent with the time period in which we extract the metal. Note that we effectively
assume an arbitrary constant metal price (which would be given in dollars or euros per tonne),
which we can omit from the objective without changing the optimal solution. As a result, our
objective is given in tonnes, rather than in a monetary unit. Constraints (4.1) ensure that we
never schedule a mining or backfilling activity more than once. Constraints (4.2) require that
the scheduled production in any time period is no more than the production capacity and no
less than what the mill requires to operate for that time period. Constraints (4.3) force the
model to maintain metal output within the operational limits of the mill. Constraints (4.4)
track the use of paste fill in each time period and ensure that its use does not exceed its
availability. Constraints (4.5), (4.6), and (4.7) are the sequencing constraints that enforce
precedence rules between activities.

The first set of sequencing constraints, (4.5), ensures that an activity \(a\) cannot begin
until all of its predecessor activities, \(a' \in \overline{P}_a\), have been completed. For example, block \(A\)
must be mined to access block \(B\), and block \(B\) must be mined to access block \(C\). Therefore, we
cannot mine block \(C\) without mining block \(A\) and block \(B\) first. The second set of sequencing
constraints, (4.6), ensures that the order of mining in a panel be maintained, regardless of
whether or not all predecessors of a block, \(a' \in \hat{P}_a\), have been extracted. In this example,
block \(A\) and block \(B\) do not both need to be mined if we mine block \(C\). Assume, without loss
of generality, that we only mine block \(A\). Then, we only impose the constraint that this block
must be mined before block \(C\). While all of the blocks’ precedences were initially characterized
according to constraints (4.5), relaxing many to follow (4.6) allows for a feasible, yet better,
solution. Sequencing constraints (4.7) ensure that an activity \(a\) cannot occur once a haulage
pillar, \(a' \in \hat{P}_a\), on which the activity depends is extracted.
We model our integer program using the AMPL programming language, Version 20090327, (Fourer et al., 2003) and solve it with the CPLEX solver, Version 12.3 (IBM, 2011) on a machine with two Dual Intel Xeon X5570 Quad Core processors and 48 GB of RAM.
CHAPTER 5
CONCLUSIONS

The Lisheen mine scheduling problem that motivates our research is deceptively difficult to model and solve. In addition to some of the complexities that we discuss in Chapter 2, working with an operational mine to produce an optimized schedule was made difficult by frequent changes to the data set and major precedence and design updates every six months. Consequently, we spent significant time updating the data set and, in particular, clarifying precedence rules.

Because underground mines are designed based on the characteristics of their mineral deposit, there is a sufficient number of attribute combinations to ensure that no two underground mines are exactly alike. Moreover, with six standard underground mining methods and many variants, each method will necessitate the implementation of different operational policies. Therefore, in consideration of the research question that we stated in the introduction, we conclude that it is difficult to write one general model for underground mine production scheduling. However, our approach could certainly be applied to other mines, regardless of blending requirements, especially to operations with flat lying deposits that practice room-and-pillar retreat mining, particularly coal mines, and to mines that are approaching the end of their operational life.

5.1 Research Contributions

Underground mine scheduling is an area of research that has not been adequately explored. The literature contains only a handful of successful implementations of optimization for underground mine scheduling, and many of these are for relatively small models. Our contributions are:

1. We develop a paradigm in which to model complex scheduling operations, such as those at Lisheen. The framework differentiates critical mining blocks that have remote
inter-temporal effects on other mining areas, i.e., haulage pillars, from all other mining activities. In this way, the complexity of the underground mine scheduling optimization problem can be managed.

2. We develop an optimization-based decomposition heuristic that can take intractable instances of the model and generate schedules within seconds (for shorter time horizons) or hours (for long post-life-of-mine schedules). The key to this technique is the identification of mining activities that are associated with the most complex precedence rules, i.e., those that preclude network structure, and carefully layering them into the schedule as the heuristic progresses through subproblems.

3. We improve the tractability of our problem by using exact and heuristic methods to reduce the problem size, and we solve a larger problem than any documented in the relevant literature. While our use of an early start algorithm for panel activities is not new, our novel heuristic to produce early start times for the haulage pillars adds to the portfolio of operations research techniques for mine scheduling problems.

5.2 Suggested Further Research

- As we mention in Chapter 3, it is possible that the structure of our model may be suited to a tighter “by” formulation that would provide a better bound for the LP relaxation. However, reformulation of the model is not a trivial undertaking and the overall density of the $A$ matrix may not change.

- An alternative approach to a “by” formulation would be to investigate additional cutting planes. Although we developed cuts to produce a tighter LP bound, their usefulness was severely limited because of the heterogeneous pillar sizes. However, discovering a strong cut, based on the pillars, would provide a much tighter LP bound with which to compare the heuristic solution.
• The effect on solution time of introducing pillars into the problem could be further investigated. Also, it would be productive to show to what degree the Lisheen precedence set is actually a network. If the mathematical structure of our precedence set has an underlying special structure, it may be possible to develop a special algorithm to solve similar problems.

• While stockpiling ore at an underground mine is considered by the majority of mining companies as unnecessary, underground mines that blend ore face a more difficult production scheduling problem than they would otherwise. Consequently, the economics of stockpiling for Lisheen could be examined in detail by adapting our model to incorporate the continuous variables required to generate inventory constraints.
REFERENCES CITED


IBM. 2011. *ILOG CPLEX*. Incline Village, NV.


term production scheduling at LKAB’s Kiruna mine. *European Journal of Operational 
Research, 211*(1), 184 – 197.


planning at an underground mine. *European Journal of Operational Research, 176*(2), 
1205–1218.

Research in Mine Planning. *Interfaces, 40*(3), 222–245.

O’Sullivan, D., & Newman, A. Extraction and Backfill Scheduling in a Complex Underground 

Saddle River.

Trans.*, 37*(2), 109–121.

mine modeling software with AMPL/CPLEX. *Internat. J. Surface Mining, Reclamation 

Topuz, E, & Duan, C. 1989. A survey of operations research applications in the mining 

*Pages 395–400 of: 25th APCOM*. Brisbane, Australia: AusIMM.

Sympos. Proc.*