Les Cahiers du GERAD ISSN: 0711–2440

Schedule-based pushback design with varying geotechnical constraints in the stochastic optimization framework

I. Farmer,
R. Dimitrakopoulos

G–2016–106

November 2016

GERAD HEC Montréal
3000, chemin de la Côte-Sainte-Catherine
Montréal (Québec) Canada H3T 2A7

Tél. : 514 340-6053
Téléc. : 514 340-5665
info@gerad.ca
www.gerad.ca

Schedule-based pushback design with varying geotechnical constraints in the stochastic optimization framework

Iain Farmer [a]
Roussos Dimitrakopoulos [a]

[a] *GERAD & COSMO Stochastic Mine Planning Laboratory, Department of Mining and Materials Engineering, McGill University, FDA Building, 3450 University Street, Montreal, Quebec, H3A 2A7, Canada*

iain.farmer@mail.mcgill.ca
roussos.dimitrakopoulos@mcgill.ca

November 2016

Les Cahiers du GERAD
G–2016–106

Abstract: The traditional mine planning framework employs a flawed approach in the design of ultimate pit limits and phases. Conventional methods arbitrarily confine the mines extraction schedule during the initial stages, detracting from its optimality before it is created. This work aims to provide a method by which a mines phase design is created from an optimal extraction schedule. The schedule-based approach to phase design yields implementable mining phases that mimic the initial raw optimal schedule from which it is based. An attempt to minimize the trade-of between mineability and value is inherent to the approach. We apply the method is a case study using a gold deposit. Results show a phase design that adheres much better to the optimal production schedule.

Keywords: Open pit mine design, pushbacks, phases, stochastic optimization, clustering, slope constraints

1 Introduction

Mine planning is the practice of developing optimal an ultimate pit limit and extraction schedule. Optimality is usually defined in terms of net present value. The input for conventional mine planning is perennially a single estimated- usually krigged- orebody model (Dimitrakopoulos, Farrellly and Godoy, 2001). This model, which does a good job of reproducing global-scale variability but a bad job of reproducing local-scale variability, forms the basis for the misguided desire to obtain the "best" mine plan – defined as the mine plan that optimizes this single input (Journel and Huijbregts, 1978). The resulting plan generated from this deterministic approach is inherently flawed.

A brief look at the traditional, and current best-practice, mine planning procedure will provide the impetus for a new, more optimal approach to mine planning. More specifically the focus of this work will be phase design, where a "phase" or "pushback" refers to a large discretization of the open pit that can be developed independently with its own working face.

The purpose of phase design is to allow the mining engineer to plan equipment movement while respecting spatial and operational constraints. Figure 1 shows an orebody cross-section and the resulting traditional phase design. Traditionally phase design is performed based on assumed block values and engineering experience rather than through an objective theoretical process. Any phase design must be specified at the outset of mining operations thus limiting a project's flexibility. Only minor adjustments to the LOM extraction sequence can be made once a phase design has been pursued, this makes it a very important initial step in mine planning (Stone et al., 2007).

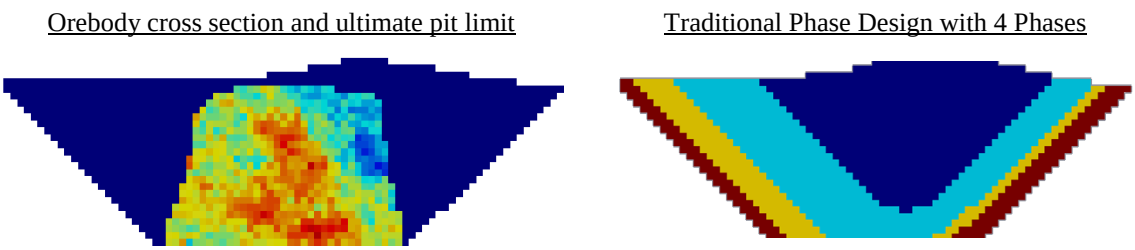

Figure 1: Orebody model with traditional ultimate pit and phase design

Figure 2 illustrates the traditional approach to mine planning which begins by defining ultimate pit limits either using Lerchs-Grossman or maximum flow methods. Through parameterizing either method, nested pit shells are created and then aggregated to form mining phases (Dagdelen, 2001). Once a phase design is created, the next step in the conventional approach would be to create a bench-wise extraction schedule constrained within the phases. This long-term extraction schedule is meant to guide mining within each phase such that value is maximized. As a final step in the traditional framework, other components such as cut-off grades and blending strategies can be optimized.

Figure 2: The traditional approach to long term mine planning

Not only does the traditional approach described above arbitrarily constrain the mine's extraction schedule to a specific pit/phase configuration, but also it: makes a-priori assumptions on block values, largely ignores production and processing constraints, and it often ignores-or does not properly account for-time value of money. A more optimal approach to mine planning would be to base the phase design on an optimized extraction schedule – flipping the traditional approach on its head.

The optimal mine planning framework proposed in Figure 3 would first allow an optimizer to determine the best life of mine (LOM) production schedule (extraction sequence and destination policy etc.) using only an objective function, the orebody model (stochastic simulations), and constraints as inputs. This production schedule can simultaneously integrate: time value of money, capacity constraints, grade blending targets, and destination decisions while optimizing value based on the given objective. With this approach it becomes unnecessary to make a-priori assumptions on block value and risk reduction is integrated organically (Goodfellow and Dimitrakopoulos, 2014).

Further, once the theoretically-optimal long-term production schedule is created, its boundaries will naturally define the ultimate pit limit.

One of the main challenges with schedules generated by stochastic optimization techniques - such as SIPs and metaheuristics - is that they are not necessarily implementable (or "smooth"). Stochastic schedules often seek to mine patchy portions of the mine within the same period. While this is technically possible, it is almost always unrealistic since it often involves prohibitive cycle times and equipment movement costs. For this reason, attempts have been made to account for spatial smoothness or connectivity during the schedule optimization stage. Ramazan and Dimitrakopoulos (2004) and Bendorf and Dimitrakopoulos (2012) both employ SIP formulations that use smoothness "windows" to penalize the objective function when adjacent blocks are mined in different periods. While this approach works well with small orebody models (~5,000 blocks), the formulation requires the addition of many additional constraints which makes the approach prohibitive for models of a more realistic size. Further, these methods do not contemplate the additional step of creating the mine's optimal phase design after a schedule has been created.

From the schedule and within the pit limits a practical (or implementable) phase design must be generated in order to plan and control equipment movement. Once a phase design is established, a mineable production schedule can be adapted. The point here is that the optimal production schedule should drive the pit/phase design, and not the other way around as with the traditional mine planning approach. Regardless of the impractical nature of the raw schedule, it remains a good basis for further design considerations (like phases) because it results in the highest theoretical value of our objective function. Thus, we would like to be able to create a phase design that inherits the major features from the initial schedule. Notably, the initial raw schedule can be the output from any one of the number of optimizers such as: the stochastic optimization of a single mine (Ramazan and Dimitrakopoulos, 2004), a global mining complex optimization (Goodfellow and Dimitrakopoulos, 2014), or the optimization of mineral supply chains (Zhang and Dimitrakopoulos, 2014).

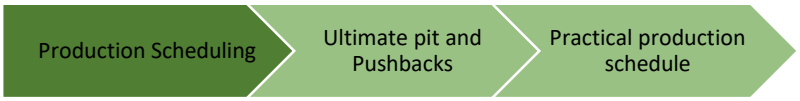

Figure 3: The optimal mine planning framework

According to best knowledge to date there exists only one known method that has been applied to create a phase design based on a pre-conceived schedule. Stone et al. (2007) propose a mine planning software called Blasor, the concept of which is to use an optimal extraction sequence to design ultimate pit limits and mining phases. Blasor performs a number of aggregation (and disaggregation) steps, both during the scheduling stage and during the phase design stage, which detract from the program's optimality. Below we propose our own method of schedule-based phase design.

2 A proposed method of schedule-based phase design

Taking the assumption that the raw schedule generated by the stochastic optimizer is the best basis for a phase design, the questions becomes: how to design phases that best mimic the LOM production schedule? The first step in the proposed process is to establish our building blocks – the constituent elements that will combine to form our phases. Since the goal of the phase design is to mimic the raw schedule as best as possible, what better source for the building blocks then the raw schedule itself?

At the left of Figure 4 is a cross-section from a raw LOM extraction sequence with the different colours representing different periods of extraction. The collection of blocks that form each 4D-contiguous (spatially-connected with the same period/colour) part of the raw schedule will constitute a component of the phase design. In Section 3.1 we discuss the method by which these components (herein called "shapes") are aggregated to build phases. The logic is that by building mineable phases out of the raw schedule's features, its major characteristics can be largely preserved, thus preserving as much of its optimality as possible.

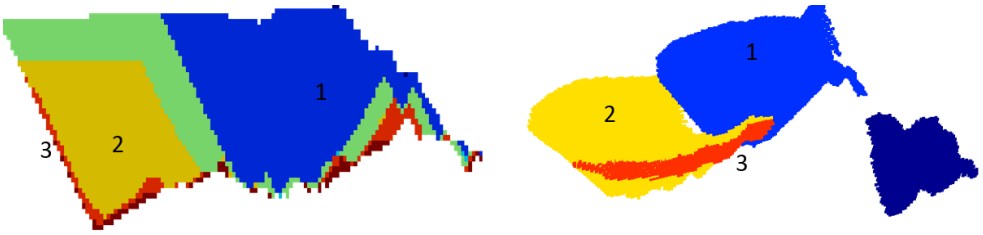

Figure 4: Cross-section (left) of a mine's schedule and some resulting 4D shapes (right)

The shapes shown at the right of Figure 4 above are created by amalgamating the orebody model's scheduled blocks using a Breadth First Search algorithm (BFS) whereby each block represents a vertex, and arcs (connectivity) are established between adjoining blocks belonging to the same period.

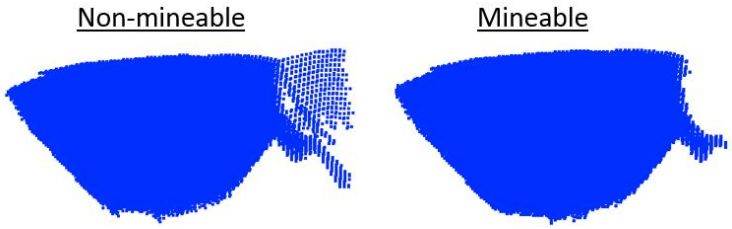

Figure 5: Separating non-mineable and mineable shapes

Once all shapes are built, they are disaggregated to form mineable and non-mineable groups (as shown in Figure 5 above) where mineability is determined by specifying a minimum number of connected blocks in each spatial dimension. This "minimum mining width" specification depends on the size of machinery and geotechnical considerations for the particular operation in question.

Algorithm 1: Building mineable and non-mineable shapes from the raw schedule

Initialization

$\chi(u) \coloneqq \{u\}$, the set of blocks that will be formed into shapes consisting of all scheduled blocks in the- orebody model
$\eta(u) \subset \chi(u)$, the set of blocks adjoining u with same period (t_u)
S_i, shape with ID *i* containing the set of blocks $\sigma_i(u) \subset \chi(u)$
S_i^m, mineable shape with ID *i* containing blocks $\sigma_i^m(u) \subset \chi(u)$
S_i^n, non-mineable shape with ID *i* containing blocks $\sigma_i^n(u) \subset \chi(u)$
$(x, y, z, t)_u$, the coordinates and period of the block *u* as determined by the raw schedule
(nx, ny, nz), the minimum number of blocks in the X, Y and Z dimensions that determine mineability

Perform Breadth-First-Search (BFS) to form initial shapes

$blocksForShapes = \chi(u)$

$i = 0$
while $blocksForShapes \neq \emptyset$ **do**
    $Q \neq \emptyset$
    Enqueue $(Q, j \in blocksForShapes)$
    $S_i = S_i \cup j$
    while $Q \neq \emptyset$ **do**
        $b =$ Dequeue (Q)
        $blocksForShapes$.Erase (b)
        for each $k \in \eta(b)$
            if $blocksForShapes$.Find (k) **then**
                'adjoining block can be added to shape.
                Enqueue (Q, k)
                $S_i = S_i \cup k$
            end if
        end for
    end while
    $i = i + 1$
end while

Disaggregate shapes into mineable and non-mineable groups

while $i < numShapes$ **do**
  $blocksInShape = \sigma_i(u)$
  while $blocksInShape \neq \emptyset$ **do**
    $j =$ Dequeue$(blocksInShape)$
    $blocksInShape$.Erase(j)
    if j satisfies minimum mining widths, $S_i^m = S_i^m \cup j$
    else $S_i^n = S_i^n \cup j$
    for each $l \in \eta(j)$
      if l satisfies minimum mining widths, $S_i^m = S_i^m \cup l$
      else $S_i^n = S_i^n \cup l$
    end for
  end while
  $i = i + 1$
end while

Once created each shape can be defined by the following attributes:

- A unique shape ID which allows easy reference of a given shape and efficient memory/runtime
- The set of constituent blocks mapped to the shape ID
- Its period of extraction
- A mineability Boolean
- The set of neighbour shapes that determine valid connections during aggregation into phases
- Its centroid – a rough estimate of the shape's location within the proposed mine

After the 4D schedule-based shapes are generated, the goal is to group them into an implementable phase design; the proposed procedure for this step is outlined in Section 3.1 below. The benefits of using the proposed schedule-based approach for phase design are that: slope constraints and temporal precedence relationships are automatically respected since the shapes are taken from the raw schedule, the problem size is reduced from the block scale to the shape scale, and any combination of shapes will remain theoretically feasible.

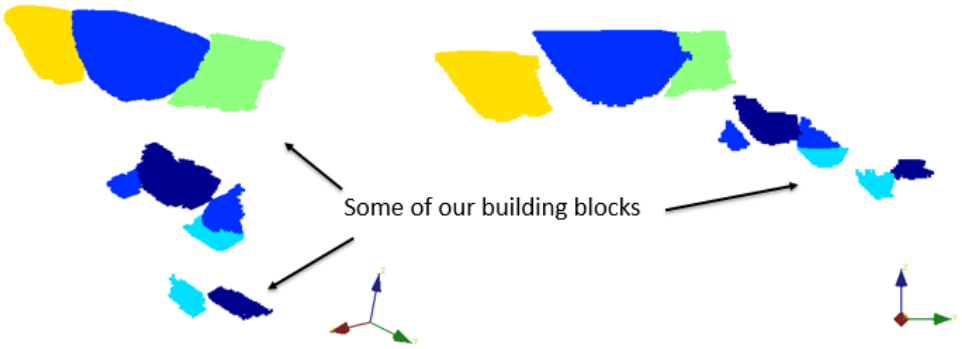

Figure 6: Some mineable shapes that we seek to combine into phases

Since any combination of 4D schedule-based shapes is feasible the next step is to decide the best way to combine the shapes to form a mineable phase design. The objective of this step is to preserve as much of the raw schedule's structure as possible while accounting for mineability. The proposed method comprises three main steps:

1. Aggregating mineable shapes through clustering
2. Using mineable clusters as seeds to grow phases
3. Post processing for minimum mining width

2.1 Aggregating 4D-contiguous shapes

The first step of the proposed method for schedule-based phase design is clustering of the mineable shapes. Clustering is a method whereby elements are assigned cluster memberships based on their similarity to one another. Similarity is defined using a distance function and attribute weightings. In our case, shapes are clustered in order to group similar elements of the raw schedule into starter-phases (or "phase seeds"). Only mineable shapes are clustered to form these phase seeds because they, in theory, are the shapes that can be kept intact during the process of creating an implementable mine plan. Non-mineable shapes are saved and incorporated into phases later (as described in Section 3.2). This grouping process is what contributes to a less fragmented, or smoother, phase design.

Clustering is performed using the k-means++ algorithm which seeks to minimize the average distance between data within the same cluster (Arthur and Vassilvitskii 2007). The method is a more efficient extension of the k-means clustering problem which takes a given integer number of clusters *k* and a set of *n* data points $\chi \in R^d$. Each data point $x \in \chi$ has *d* attributes, each with their own weighting. In order to create clusters that best mimic the initial schedule, a heavy weighting is placed on the attribute corresponding to the period of extraction, thus urging shapes with similar periods of extraction to be grouped into the same seeds. The spatial attributes of each shape (its centroid coordinates) can also carry a weighting, but any spatial weighting should be minimal and will be deposit-dependant.

Algorithm 2: Aggregating mineable shapes into phase seeds

Initialization

S_i^m, mineable shape with ID *i* containing blocks $\sigma_i^m(u) \subset \chi(u)$

A_i, the attribute values (x, y, z, t) for shape *i*

K, number of phases to use

P_k, phase seed $k \in [0, K]$ with cluster center C_k

Perform k-means++ clustering on mineable shapes

'Randomly select initial cluster center

$C_0 = randomShape()$

do until $k = K$

  for each *i*

    let $minDist = \min\{dist(A_i, C_l)\}$

    randomly select $C_k = A_i$ with probability $\frac{minDist^2}{\sum_i dist(A_i, C_l)^2}$

  end for

end do

'initial cluster centroids have been chosen intelligently

while $membershipChange =$ true

  for each *k*

    for each shape *i*

      $dist[i][k] = dist(A_i, C_l)$

    end for

  end for

  for each shape *i*

    if $dist[i][k]$ is the minimum distance $\forall k \in K$

    then $P_k = P_k \cup S_i^m$

    'adding shapes to cluster with nearest centroid

  end for

  if there are no membership changes

    $membershipChange =$ false

  end if

  for each *k*

  'update cluster centers

    $$C_k = \frac{1}{|P_k|} \sum_{|P_k|} A_i \quad \forall S_i^m \in P_k, k \in K$$

  end for

end while

The result of the above k-means++ clustering is *k* clusters that will be used to seed our phase design. Together the clusters contain all mineable shapes as seen in the example presented in Figure 7 where each cluster is represented by a different colour.

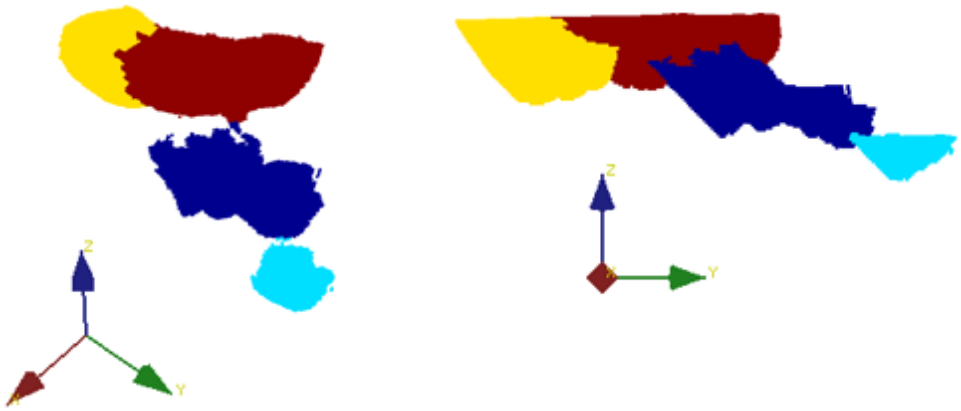

Figure 7: Mineable clusters used to seed our phase design

2.2 Growing phases from robust seeds

Once the phase design has been seeded with the clusters of mineable shapes the remaining, non-mineable shapes, are added. The incorporation of the non-mineable shapes into the phase design is performed by first ordering the seeds based on their constituent shapes' average extraction period and then through applying slope/precedence constraints; adding predecessor shapes to blocks within each ordered seed. When adding predecessors, (moving up through the deposit) it is important to apply the slope constraints beginning with the youngest seed for two reasons: first, it tends to be the nearest seed to the surface and second, it tends to have the highest discounted per-block value. When adding successor shapes, the reverse is true and we begin with the oldest seed.

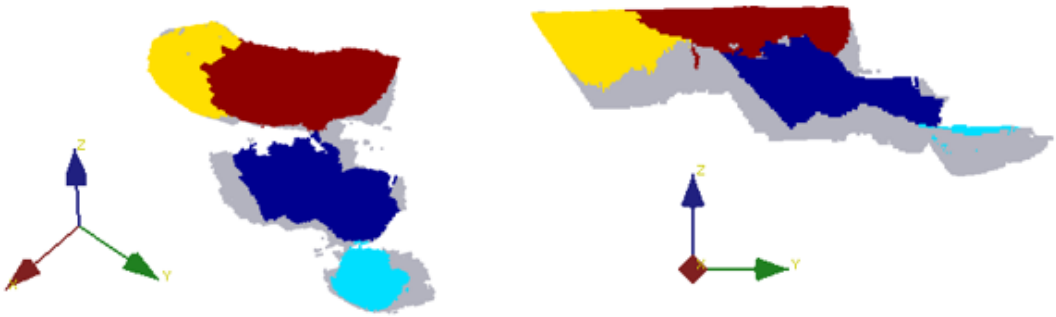

Figure 8: Phase seeds with non-mineable shapes (grey) that are to be added

Algorithm 3 which is used to add the predecessors and successors of each phase seed is presented below. For each seed there exists a list of predecessors, $preds_k$ and a list of successors, $succs_k$. These lists depend on the predecessors and successors for all the individual blocks located within each phase seed which, in turn, depend on the different geotechnical zones each having their accompanying slope constraints. The task of efficiently incorporating varying geotechnical zones is discussed in the appendix.

If a non-mineable shape can be added in its entirety to a phase without violating the slope constraints (i.e. all of its constituent blocks lie in the predecessor or successor list of a single seed), the shape is added to the phase. If it cannot be added as a whole, the non-mineable shape is disaggregated into blocks and the individual blocks are added to phases based on their predecessor/successor relationships.

Algorithm 3: Growing seeds into phases by adding remaining shapes using slope constraints

Initialization

P_k, phase seed containing the set of shapes $\zeta_k(S)$
$preds_k$, predecessors of all shapes in seed k
$succs_k$, successors of all shapes in seed k
$sortedSeeds$, sorted list of phase seeds such that $sortedSeeds[0]$ is the youngest seed
$remainingShapes$, initialized as all non-mineable shapes, S_i^n

Go through sorted phases adding predecessors and successors

for $j = sortedSeeds[0]$ to $j = sortedSeeds[K]$
  for each shape $S_i \in \zeta_j$
    if $preds_k \in remainingShapes$
      then $P_k = P_k \cup preds_k$
      $remainingShapes$.erase$(preds_k)$
    end if
  end for
end for

for $j = sortedSeeds[K]$ to $j = sortedSeeds[0]$
  for each shape $S_i \in \zeta_j$
    if $succs_k \in remainingShapes$
      then $P_k = P_k \cup succs_k$
      $remainingShapes$.erase$(succs_k)$

    end if
  end for
end for

Figure 9 compares the original raw schedule to the phase design generated by the proposed method. Notably, the phase design mimics the schedule well with similar portions of the schedule belonging to the same phase.

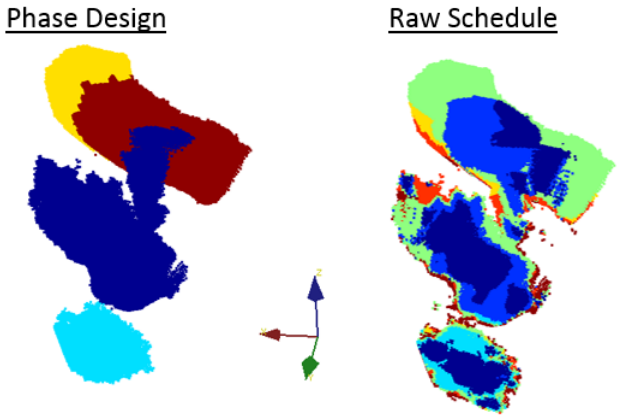

Figure 9: Schedule-based phase design compared with raw stochastic schedule

Due to the pre-defined geometries of what is already a partially-mined operation, the resulting phase design shown in Figure 9 may be fairly obvious and perhaps is not the best example to illustrate the benefits of the proposed method. The benefits of our approach are more recognizable in a second example shown in the case study below.

2.3 Post-processing for minimum mining width

After the seed phases have been grown into complete phases that incorporate all the blocks within the ultimate pit, it is still possible in certain places that the minimum mining dimensions are not respected. For this reason, a brief post-processing step is required which smooths out regions where a phase is too narrow by transferring offending blocks into the earliest adjoining phase (Wharton and Whittle, 1997).

There are two possibilities: either two separate phase boundaries sandwich a third phase boundary, or the space between the pit wall and a phase does not respect the minimum mining width. In the first instance offending blocks within the sandwiched phase are shifted into the earlier of the two surrounding phases, in the second instance the unmineable portions near the pit wall are shifted into their nearest adjoining phase. In this way certain phases get slightly larger after smoothing whereas others get slightly smaller.

2.4 Evaluating and selecting the best design

A first step towards evaluating the quality of the phase design that results from the method is by simple visual inspection. Cross-sections of the schedule and phase design can easily be compared and their similarity can be evaluated. However, this approach merely provides a qualitative measure. To quantify the quality of our schedule-based phase design we would need to generate a mineable extraction schedule confined within the phases and evaluate it using some metric (such as NPV or the objective function from the optimization that generated the initial raw schedule). The phase-constrained mineable schedule would most likely be some form of bench-wise schedule whereby the intersection of phases and benches are sequenced. Although this portion of the study is yet to be completed, the fact that the schedule-based phase design inherently mimics the raw schedule means the trade-off between practicality and value should lead to only a small reduction in value. It is indeed this trade-off that the proposed approach seeks to minimize.

With the assertion that the phase design's quality can be quantified, it only makes sense to see if the method can be iteratively improved upon during the initial optimization procedure. While it is likely that the optimal raw schedule will produce a good phase design and practical schedule, there is no guarantee that it will produce the *best* phase design and practical schedule. For this reason, we recommend perturbing the raw schedule, as well as the parameters used in the above method of phase design, in an attempt to find the combination that can iteratively lead to the optimal practical schedule.

A topic not contemplated is what sort of schedule perturbations would best suit the re-optimization process once an initial schedule is found. This decision is both highly optimizer and deposit-specific.

3 Case study – application of the method to two different mineral projects

We apply the proposed method of schedule-based phase design to two orebodies with different characteristics. The first orebody has been operated for a number of years so it poses some pre-existing spatial restrictions on our phase design, the second has yet to be mined and thus represents a more typical example of the input for a long-term mine plan. Stochastic schedules (assumed to be optimal) for both orebodies have been generated and will form the main input for phase design. Cross-sections of each schedule are shown in Figure 10.

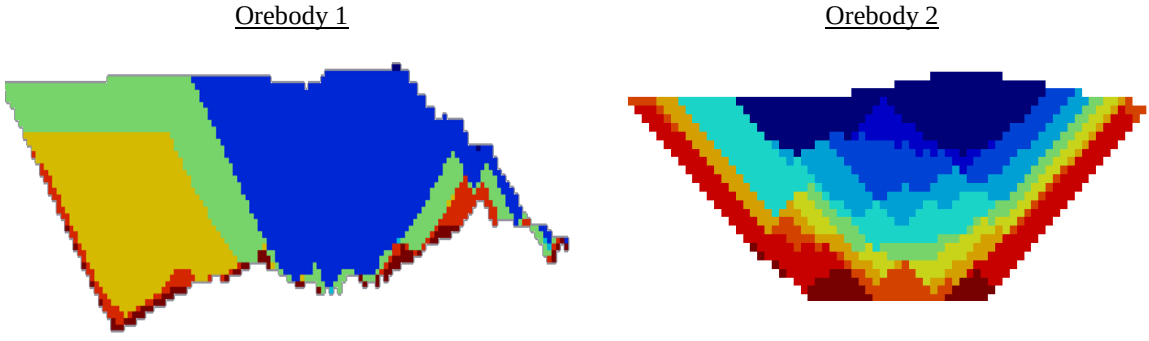

Figure 10: Cross-sections of raw schedules for two different orebodies

Step 1: The first step of the proposed method is to create mineable and non-mineable shapes from our schedules. This is done using a BFS and Algorithm 1. Figure 11 presents some the results for the two orebodies. The figure shows a non-exhaustive set of mineable shapes for each orebody; these shapes are then to be used in the creation of seed phases.

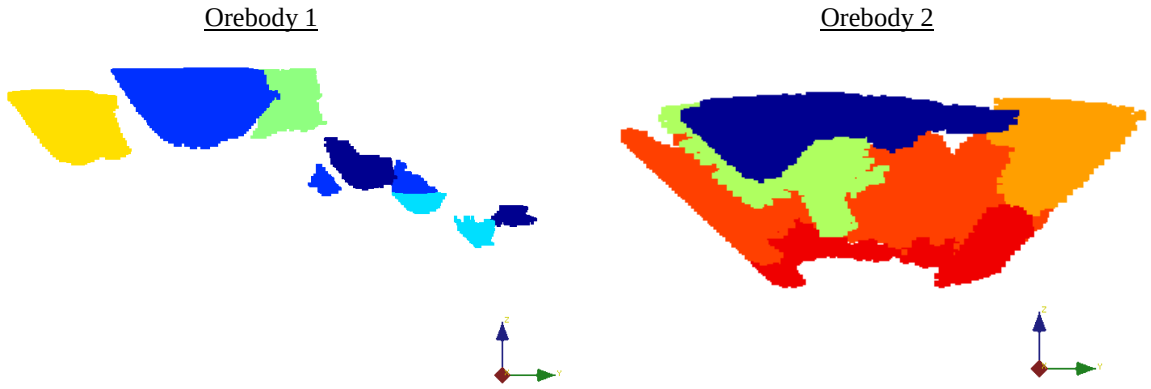

Figure 11: Some mineable shapes for each orebody

Step 2: The next step is to cluster these shapes using k-means++ and Algorithm 2 to form seed phases. The seed phases for the two schedules are shown in Figure 12. The collection of seeds contain all mineable shapes clustered based on the assigned clustering weights.

Figure 12: Initial phase seeds for each deposit

Step 3: Once the phase seeds are generated, they are grown into a final phase design by applying slope constraints, adding all remaining non-mineable shapes to the design. Comparisons between the final designs and the raw schedules for each deposit are shown in Figure 13.

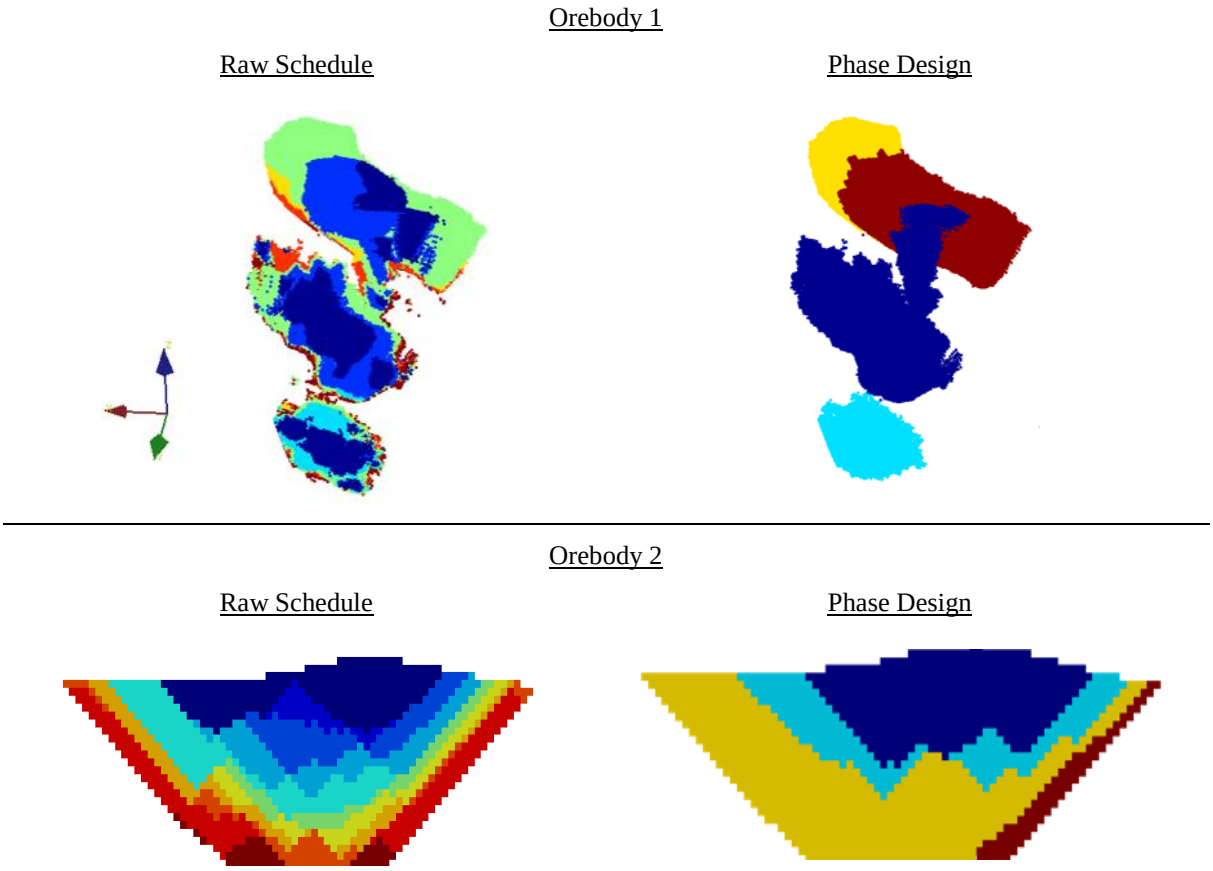

Figure 13: Final schedule-based phase design for each orebody

The above designs mimic the raw schedule well and can be shown to be mineable based on the specified minimum mining widths.

4 Conclusions

Traditional mine planning methods arbitrarily confine the LOM extraction schedule to pre-determined pit limits and phases, limiting the mine's optimality at the outset. A better approach is to generate an optimal schedule that serves as the basis for the physical pit design.

The proposed method for generating a schedule-based phase design seeks to minimize the trade-off between a LOM schedule's optimality and its ability to be implemented. By disaggregating the raw schedule into 4D-contiguous shapes we are able to easily retain the schedule's principal characteristics which can then be combined based on their similarity to generate mineable phases. The method has been shown to achieve a phase design that is different from the one generated using the conventional mine planning framework as seen in Figure 14, and since it is able to retain the raw schedule's features, the schedule-based design does a better job mimicking the optimal raw schedule.

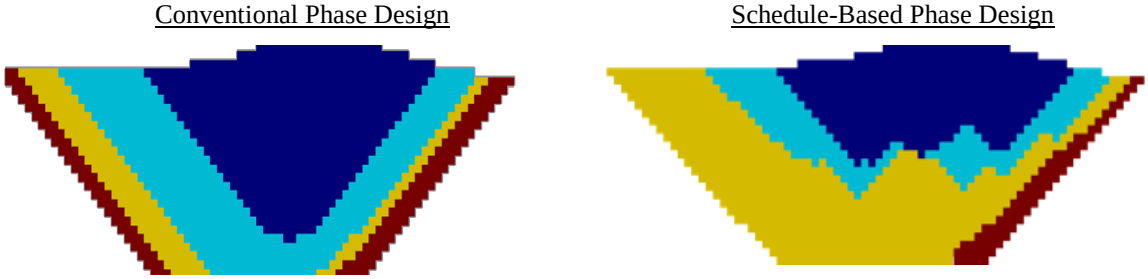

Figure 14: The schedule-based phase design differs from the traditional design

Appendix Efficiently including varying geotechnical zones

In any orebody there may be a number of geotechnical zones with various rock competencies. In more competent rock pit-wall slope angles can be steeper allowing for a lower strip ratio, whereas in poorer-quality rock shallower slope angles are required and strip ratios increase. These varying geotechnical zones determine slope constraints which dictate which overlying blocks (predecessors) must be mined before access can be granted to a target block (the successor).

As noted above, the procedure by which non-mineable shapes are incorporated into the seed phases requires a list of predecessors and successors for each block in the orebody model. When using metaheuristic optimization methods these same lists are also required when generating the raw schedule since a block's predecessors and successors determine how perturbations are allowed to proceed. Given that many perturbations are likely needed to reach a good solution, the most efficient representation of these lists is desired.

The exhaustive list of predecessors for a given block resembles a cone with an elliptical cross-section. The shape of the ellipse is determined by the minimum slope angles in four given directions. Instead of using the full predecessor cone, it is more efficient (in terms of computational time and memory) instead to store a more minimal set of predecessors for each block and algorithmically move up through the deposit while checking, when needed, this smaller set. The minimal predecessor set resembles an enveloping shell as shown in Figure 15. In some places the shell is thicker than others due to the interplay between the predecessor relationship and geotechnical variability. For example, in the simple case where we only consider a block and its overlying block; the predecessor envelope will become thinner as the minimum slope angles of the two blocks become more alike.

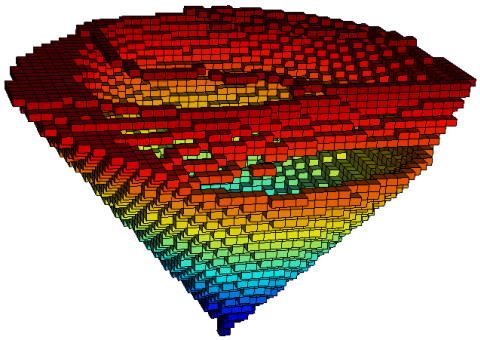

Figure 15: The predecessor envelope for a single block

Once each block's predecessor envelope is generated, it can be stored in a text file and accessed whenever needed. In this way, the predecessor list only has to be generated once and the algorithm for doing so can remain separate from the rest of the optimization/mine planning procedure. The following algorithm illustrates the process by which predecessor envelopes are generated.

Algorithm 4: Generating predecessor envelopes for deposits with varying geotechnical zones

Initialization

A, number of different geotechnical zones with angles α_A

$fullCone[a]$, the full cone of predecessor blocks for a block in zone a

Find full set for every zone

$fullCone[a] = getFullCone(a)$, as in Khalokakaie et al. (2000)

Perform set-subtraction with overlying block to generate minimal set

for each block u (descending z-coordinate)

$F(u) = fullCone[zone(u)]$

for each block $v \in F(u)$

if any $\alpha_a^v < \alpha_a^u$

$fullSet(u) \cup fullCone(v)$

'adding predecessors of predecessors if the block has any wider slope angle

end if

end for

$setSubtraction(u) = fullSet(u) \cap fullSet(u_{k+1})$

'set-subtraction with overlying block

end for

Add back "rings" to complete enveloping cone

for each block u

for each layer z in $fullCone(u)$

$rings(u) = rings(u) \cup extremeBlocks\big(z, fullCone(u)\big)$

end for

$predEnvelope(u) = setSubtraction(u) \cup rings(u)$

end for

Now, in order to take best advantage of the envelope predecessor representation, we must generate an algorithm that will allow us to rigorously check for all possible violations as efficiently as possible. The method for doing so is illustrated in Algorithm 5.

Algorithm 5: Efficiently checking slope violations (R. Goodfellow, 2015)

Identify and repair slope violations

Stack $currentBlock = \{b\}$

'block who's predecessors are being examined

$blocksToChange = \{b\}$

'set of violating blocks

while $currentBlock! = \emptyset$

$u = currentBlock.pop()$

$needsFixing = false$

for each $\{v\} \in preds[u]$

if $period[v] \leq period[u]$: continue

'block *v* is ok

end if

$needsFixing = true$

$blocksToChange = blocksToChange \cup \{v\}$

end for

if $needsFixing =$ true, $currentBlock$.push$\big(u(x, y, z+1)\big)$

'explore block above

end while

References

F.R. Albor Consuegra and R. Dimitrakopoulos (2010) Algorithmic approach to pushback design based on stochastic programming: method, application and comparisons. Mining Technology, 119(2), 88–101

D. Arthur and S. Vassilvitskii (2007) k-means++: The advantage of Careful Seeding. Proceedings of SODA, 1027–1035.

M. Assad and R. Dimitrakopoulos (2013) Implementing a parametric maximum flow algorithm for optimal open pit mine design under uncertain supply and demand. Journal of the Operational Research Society, 64, 185–197.

J. Benndorf and R. Dimitrakopoulos (2012) Stochastic long-term production scheduling of iron ore deposits: integrating joint multi-element geological uncertainty. Journal of Mining Science, 49(1), 68–81.

K. Dagdelen (2001) Open pit optimization - strategies for improving economics of mining projects through mine planning. 17th International Mining Congress and Exhibition of Turkey, 117–121

R. Dimitrakopoulos, C.T. Farrelly and M. Godoy (2002) Moving forward from traditional optimization: grade uncertainty and risk effects in open-pit design. Transactions of the IMM, Section A Mining Industry 111, A82–A89.

R. Goodfellow, and R. Dimitrakopoulos (2014) Stochastic optimisation of mineral value chains-developments and applications for the global optimisation of mining complexes with uncertainty. SMP Symposium, AusIMM, Perth Australia, November 24–26.

R. Goodfellow and R. Dimitrakopoulos (2013) Algorithmic integration of geological uncertainty in pushback designs for complex multiprocess open pit mines. Mining Technology, 122(2), 67–77.

D.S. Hochbaum and A. Chen (2000) Performance analysis and best implementation of old and new algorithms for the open-pit mining problem. Operations Reasearch, 48(6), 894–914.

A.G. Journel and C.J. Huijbregts (1978) Mining Geostatistics. Blackburn Press, 410–443, 494–517.

R. Khalokakaie, P.A. Dowd and R.J. Fowell (2000) Lerchs-Grossman algorithm with variable slope angles. Mining Technology, 109(2), 77–85.

J. Kleinberg and E. Tardos (2006) Algorithm Design, Addison-Wesley, 79–82

C. Meagher, R. Dimitrakopoulos and V. Vidal (2014) A new approach to constrained open pit pushback design using dynamic cut-off-grades. Journal of Mining Science, 50(4), 733–744.

C. Meagher, S.A. Sabour and R. Dimitrakopoulos (2009) Pushback design of open pit mines under geological and market uncertainties. Orebody Modeling and Strategic Mine Planning AusIMM Spectrum Series, 17, 297–304.

S. Ramazan and R. Dimitrakopoulos (2004) Traditional and new MIP models for production scheduling with in-situ grade variability. International Journal of Surface Mining, Reclamation and Environment, 18(2), 85–98.

S. Ramazan and R. Dimitrakopoulos (2004) Uncertainty-based production scheduling in open pit mining. SME Transactions, 316, 106–112.

G. Stone et al. (2007) Blasor – Blended iron ore mine planning optimization at Yandi, Western Australia. Orebody Modeling and Strategic Mine Planning, Spectrum Series, 14, 133–136.

C. Wharton and J. Whittle, (1997) The effect of mining width on NPV. Optimizing with Whittle Conference Proceedings, Perth 1997, 173–177.

R. Zhang and Dimitrakopoulos (2014) Optimizing a mineral suppy chain under uncertainty with long-term sales contracts. SMP Symposium, AusIMM, Perth Australia, November 24–26.