Seifi, Cinna; Schulze, Marco; Zimmermann, Jürgen

Solution procedures for block selection and sequencing in flat-bedded potash underground mines

OR Spectrum

Provided in Cooperation with:
Springer Nature

Suggested Citation: Seifi, Cinna; Schulze, Marco; Zimmermann, Jürgen (2021) : Solution procedures for block selection and sequencing in flat-bedded potash underground mines, OR Spectrum, ISSN 1436-6304, Springer, Berlin, Heidelberg, Vol. 43, Iss. 2, pp. 409-440, https://doi.org/10.1007/s00291-021-00618-z

OR Spectrum (2021) 43:409–440
https://doi.org/10.1007/s00291-021-00618-z

REGULAR ARTICLE

Solution procedures for block selection and sequencing in flat-bedded potash underground mines

Cinna Seifi[1] · **Marco Schulze**[2] · **Jürgen Zimmermann**[1]

Received: 28 June 2019 / Accepted: 25 January 2021 / Published online: 6 March 2021

Abstract

Phosphates, and especially potash, play an essential role in the increase in crop yields. Potash is mined in Germany in underground mines using a conventional drill-and-blast technique. The most commercially valuable mineral contained in potash is the potassium chloride that is separated from the potash in aboveground processing plants. The processing plants perform economically best if the amount of potassium contained in the output is equal to a specific value, the so-called optimal operating point. Therefore, quality-oriented extraction plays a decisive role in reducing processing costs. In this paper, we mathematically formulate a block selection and sequencing problem with a quality-oriented objective function that aims at an even extraction of potash regarding the potassium content. We, thereby, have to observe some precedence relations, maximum and minimum limits of the output, and a quality tolerance range within a given planning horizon. We model the problem as a mixed-integer nonlinear program which is then linearized. We show that our problem is $\mathcal{NP}$-hard in the strong sense with the result that a MILP-solver cannot find feasible solutions for the most challenging problem instances at hand. Accordingly, we develop a problem-specific constructive heuristic that finds feasible solutions for each of our test instances. A comprehensive experimental performance analysis shows that a sophisticated combination of the proposed heuristic with the mathematical program improves the feasible solutions achieved by the heuristic, on average, by 92.5%.

Keywords Underground mining · Block selection and sequencing · Quality objective · Mixed-integer linear programming · Priority rule-based procedure

✉ Cinna Seifi
cinna.seifi@tu-clausthal.de

[1] Institute of Management and Economics, Operations Research Group, Clausthal University of Technology, 38678 Clausthal-Zellerfeld, Germany

[2] K+S Aktiengesellschaft, 34131 Kassel, Germany

1 Introduction

This paper considers one of the biggest German potash mines. In Germany, the potash ores are generally found in deep deposits. Hence, potash mines are typically underground mines with an area-wide expansion of the deposit, a so-called flat-bedded deposit. According to Musingwini (2016), optimization in underground mine planning is not as well developed and widely applied as in open pit mine planning, although the logic of the planning is the same. O'Sullivan et al. (2015) state by comparing the common mathematical formulations for both open-pit and underground mining that scheduling in underground mines is more complex than the scheduling in the open-pit mines based on the complex structure of precedence constraints, the characteristics of the operations and activities, and the irregularity of the size and shape of the blocks. Musingwini indicates that the main reason for the complexity difference between open-pit and underground mine planning is that the direction of mining in open-pit mines is essentially down and outward to the pit limits. However, in underground mines, there are numerous permutations of the direction of mining depending on the mining method chosen.

The mostly applied extraction method for flat-bedded deposits of limited thickness is a drill-and-blast technique using the room-and-pillar mining method. By the use of the room-and-pillar mining method, the material is extracted across a horizontal plane, and pillars, arranged in regular patterns, are left for support purposes. Thus, a grid-like structure is formed, as demonstrated in Fig. 1 (Hamrin 2001; Schulze et al. 2016).

By employing a conventional drill-and-blast technique, the mining activities are conducted at the salt faces (cf. Fig. 1). At each salt face, the following discrete steps (mining operations) must be processed in chronological order (K+S 2013):

1. scaling the mine roof and sidewalls,
2. removing the scaled material,
3. bolting the roof with anchors,
4. drilling large diameter boreholes,
5. removing drilling dust,

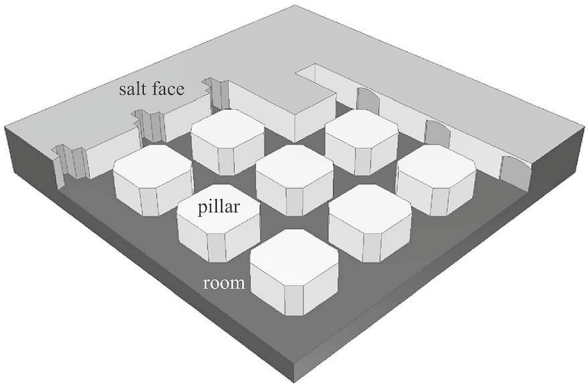

Fig. 1 Grid structure caused by the room-and-pillar mining method. Reprinted from Schulze et al. (2016)

6. drilling blast holes,
7. filling blast holes with explosive substances,
8. blasting,
9. transporting broken material to a feeder breaker.

During mining operation 4 (see the enumerated list above), large drill jumbos drill three adjacent horizontal boreholes with a diameter of 0.28 m and a length of 7 m at each salt face. The large boreholes act in particular as a direction guideline for blasting. After the detonation (mining operation 8), chambers, so-called *rooms*, are created in the direction of the mining activity (cf. Fig. 1). The height and the width of the exploded area are then scaled based on a given plan. Thus, after each detonation, a three-dimensional *block* of potash is removed, and a new room of the same size arises. During mining operation 9, the raw material is delivered using loaders from each salt face to a feeder breaker, where the lumps are broken. After completing mining operations 1 to 9, the position of the current salt face is shifted by the respective block's length, exposing a new salt face, which can be operated on directly afterward.

The onward transportation of crude materials from feeder breakers takes place using a conveyor belt system to a bunker near the shaft. The excavated crushed potash is transported from the bunker through the shaft to the surface. The potash ores are rich in potassium chloride, sodium chloride, and different associated minerals. Potassium chloride is a valuable salt that is mainly used as a fertilizer. Furthermore, it is an integral additive in the chemical, medical as well as human and animal food-processing industry (Chesworth 2008; USGS 2011; Schulze et al. 2016). After transporting the crude salt to the surface, potassium chloride is separated from the extracted potash by flotation, recrystallization, or electrostatic separation in aboveground processing plants. For each technical device, there is an *optimal operating point* at which the device has the best performance. This point can be determined based on various factors. The processing plants above ground can be most cost-effectively utilized if the amount of potassium contained in the extracted material is equal to a specific value. The equivalent content of potassium oxide is often reported to indicate the percentage of potassium by weight in the potassium chloride, where 100% potassium chloride is precisely equal to 63.17% potassium oxide (cf. Heinz and von der Osten 1982, p. 147). As mentioned, at a salt face, a block of potash can be unearthed. For each block, the amount of potash is measured according to the dimensions of the excavation. Moreover, the potassium contained in each block of potash *in percent* is determined based on geological investigations. The percentage of potassium contained in the extracted potash from a block is defined as the *quality value* of the corresponding block. Accordingly, the quality value of a block multiplied by the amount of potash obtained from that block determines the amount of potassium contained. In general, the blocks are different regarding the amount and the quality value of the potash contained. Moreover, not all the available salt faces and the corresponding blocks can be mined within a given planning horizon. Therefore, the quality of the amount of potash extracted within different time intervals

strongly depends on the selected blocks and the sequence of the extraction. Vast fluctuations in the quality value result in non-homogeneous output that leads to high costs for aboveground processing. Because quality fluctuations occur frequently owing to the way in which the potash is extracted, quality-oriented mining of blocks plays a decisive role in reducing the processing costs.

Newman et al. (2010) classify the existing approaches and models in mining companies according to the planning horizon or the hierarchy level into long-term (strategic), medium-term (tactical), and medium- and short-term (tactical-operative) problems. In this paper, we consider a medium-term planning horizon. Within the planning horizon, we want to have a homogeneous output regarding the quality value of the extracted potash. More precisely, the quality value of the extracted material should deviate as little as possible from a given quality target value so that the mineral processing above ground is conducted economically. In this regard, we want to answer two questions: 1. which block should be excavated (i.e., block selection), and 2. if a block is extracted, the time at which it is extracted (i.e., block sequencing). Taken together, we solve a block selection and sequencing problem with a quality-oriented objective function at a tactical planning level. For our problem, precedence relations, maximum and minimum limits of the output, and a quality tolerance range have to be taken into account.

The remainder of the paper is organized as follows: In Sect. 2, the characterization of the problem at hand and a literature review are given. Sect. 3 introduces a mathematical formulation for our problem. Since some constraints are not linear, more decision variables and constraints are introduced to linearize the mathematical program. Subsequently, we show that our problem is $\mathcal{NP}$-hard in the strong sense. Accordingly, in Sect. 4, we devise a constructive heuristic, which is embedded in a multi-start environment and provides good, feasible solutions for the problem through a sophisticated time-saving procedure. In Sect. 5, based on some generated realistic problem instances, we compare the introduced mathematical program and the proposed constructive heuristic. We also show if we solve our problem heuristically and use the feasible solution found as an initial solution for the mathematical program, we obtain much better results, especially for large and challenging problem instances. The paper concludes with a summary of the achieved results and an outlook on future research in Sect. 6.

2 Problem specification and related literature

For a proper operation, from a geographical point of view, underground mines are usually divided into smaller spatial areas, so-called *mining districts*. Accordingly, an underground potash mine has, on average, up to 5 mining districts. The area of a mining district may be several square kilometers. Due to this spatial expansion, several *tipple areas* are constructed for a mining district to divide the area into smaller parts avoiding long transportation routes. In each mining district, depending on its area, 4 to 6 tipple areas are involved. In a tipple area, a feeder breaker is installed, where the lumps of the extracted potash in that tipple area are delivered to and crushed. As mentioned, the mining operations are conducted at a salt face.

Subsequently, the detonation occurs in the mining direction, in which a block of potash is extracted. After mining operation 8 (blasting), a three-dimensional block of potash with the known dimensions is removed, and a room is created. A chain of consecutive blocks that are extracted one after another in a certain mining direction is defined as an *underground location*. Since a tipple area can have an extent of up to a few 100 meters, several underground locations (usually between 5 and 11) are assigned to the single tipple areas. That means the blocks of material extracted from each underground location are transported to the feeder breaker installed in the corresponding tipple area.

Figure 2 illustrates a tipple area with three underground locations (UL 1 to UL 3) that are assigned to a feeder breaker, i.e., the potash unearthed from UL 1 to UL 3 is transported to the illustrated feeder breaker. In UL 1, three blocks are already removed in the mining direction. The associated rooms are designated by squares with solid lines. After mining a block, a new salt face (a potential block) in the mining direction becomes available. Geological sampling and mining investigations determine how many of the consecutive blocks in an underground location can be removed within a considered planning horizon. In Fig. 2, the dashed squares in UL 1 and UL 2 indicate the blocks that can be mined according to the plan for the considered time horizon. On the contrary, no further block can be removed in UL 3

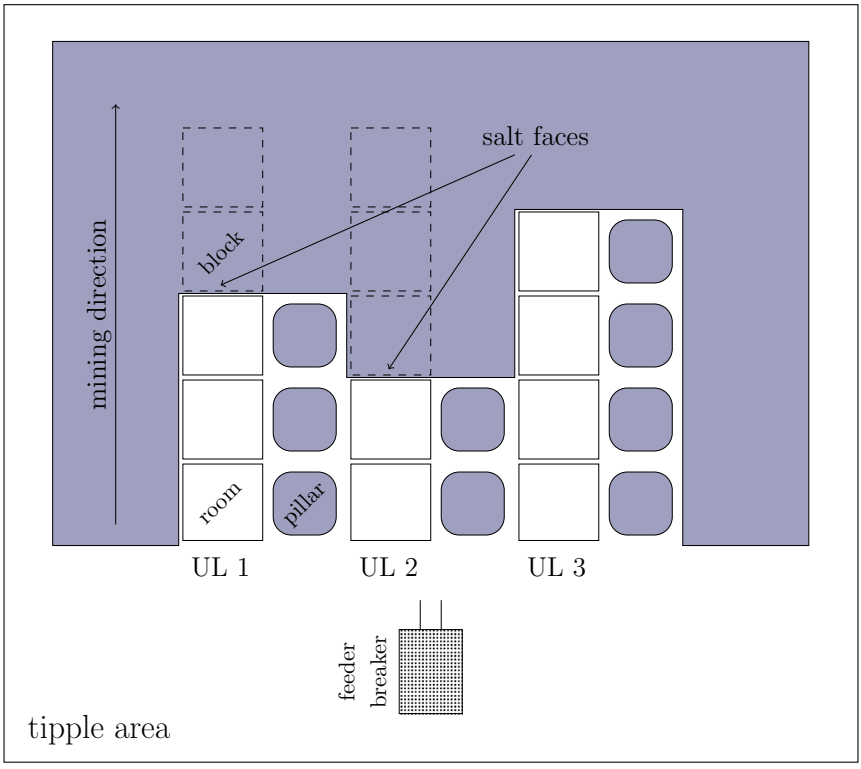

Fig. 2 A tipple area with associated underground locations. Adapted from Clausen (2013), p. 23

Table 1 Definitions of the mining terminology introduced

Term	Definition
Block	A cube of material with known dimensions removed after a detonation
Feeder breaker	A crushing machine in a tipple area where the lumps extracted from the underground locations assigned to this tipple area are delivered to
Mining district	The largest unit of an underground mine that comprises some smaller units (see the definition of a tipple area)
Pillars	Parts of underground mines that are not extracted to support the roof from collapsing
Room	A space of known dimensions created after a detonation
Salt face	A place at which the mining operations are conducted, i.e., it is the front side of the block that is extracted
Tipple area	The largest unit of a mining district that is characterized by the assigned underground locations and a feeder breaker
Underground location	A chain of consecutive blocks that can be removed in the mining direction

within the planning horizon. For better clarity, we summarize in Table 1 the aforementioned mining terms.

The excavation of a block can only then be started if the previous blocks in the same underground location have been fully excavated. A block has, at most, one direct predecessor and, at most, one direct successor block in the corresponding underground location. The time needed to remove a block is the sum of the processing times of the required mining operations 1 to 9. Each mining operation must be processed by one machine and by one skilled worker. According to the speed of the assigned machine and the skill level of the assigned worker, different processing times are needed to accomplish a mining operation. Machines and workers are scheduled at the beginning of each work shift at an operational planning level (see, e.g., Schulze and Zimmermann 2017; Seifi et al. 2019, and Seifi et al. 2020). Since we deal with a block sequencing problem at a tactical planning level, the processing times required for the extraction of blocks must be estimated. In doing so, the dimensions or shapes of blocks are crucial factors. Moreover, the current status of a block at the beginning of the planning horizon affects the needed processing time. For example, a block for which mining operations 3–9 must be carried out has a shorter processing time than a block for which all mining operations 1–9 must still be processed.

In every tipple area, the extracted material is initially carried out via the loaders from underground locations to the assigned feeder breaker. The conveyance of the extracted potash from the particular tipple areas of each mining district takes place through a conveyor belt system to a central bunker system close to the shaft, from where the material ultimately reaches the surface. The capacities of the conveyor system, the bunker, and the processing plants above ground are limited. Hence, in each work shift, an upper limit of the output for each tipple area and each mining district must be observed. On the other hand, the primary task of mining companies is the extraction of raw minerals. Accordingly, a lower

limit for the total output over the planning horizon must be considered. Moreover, a minimum output in each work shift and every mining district must be satisfied.

In our problem, we consider a planning horizon (e.g., a month) that is a union of some smaller sub-intervals (e.g., weeks). Within those sub-intervals, the quality value of the entire extracted potash must be within a given quality tolerance range. The assumptions regarding the length of the planning horizon and the corresponding sub-intervals can be customized according to the current situation of the mine at hand. Assume that a set of blocks is excavated within a specific time interval. The quality value of the entire extracted material within that time interval is the weighted average of the quality values of the removed blocks. Due to our objective, the quality value of the extracted material should deviate as little as possible from a given quality target value. The quality target value in percent typically represents the optimal operating point of the processing plant above ground. We speak of a negative deviation if the weighted average of the quality of the extracted material is less than the quality target value. Analogously, there is a positive deviation if the weighted average of the quality of the extracted material is greater than the quality target value. For formulating the objective function, we first determine the absolute deviation's value of the weighted average of the output quality from the predetermined quality target value in each considered sub-interval in the planning horizon. The value of the weighted average of the output quality and, thus, the values of negative and positive deviations are determined based on the amount of material extracted within the considered time interval. Since the amount of material removed is not known a priori, determining the deviations from the given target value requires some nonlinear constraints in the mathematical formulation that must be linearized. The aim is then to minimize the average of the calculated deviations over the number of sub-intervals considered in the planning horizon.

Altogether, we minimize a quality-oriented objective function observing the following groups of constraints:

Precedence relations between the blocks in an underground location;
Minimum limit of the output over the entire planning horizon, in each work shift, and for every mining district;
Maximum limit of the output for each tipple area and every mining district within every single work shift; and
Quality tolerance range over each certain sub-interval in the planning horizon.

Newman et al. (2010), Kozan and Liu (2011), as well as Blom et al. (2019) published survey articles on the application of operations research methods in the field of mining. Lately, Leite et al. (2020) gave a review on state-of-the-art applications of operational research techniques to mining problems taking the mentioned surveys into account. Leite et al. (2020) consider (1) layout and design problems, (2) production and scheduling problems, and (3) operational equipment allocation problems at strategic, tactical, and operational planning levels, respectively; consequently, they review the published articles in both open-pit and underground mines.

For more convenience, in Table 2, we list the most significant works published in the previous decade that introduce a mathematical formulation for a block

Table 2 Literature on block selection and sequencing problems

	Constraints			OF
	Quantity	Quality	Precedence	
Bley et al. (2010)	•	–	•	M
Nehring et al. (2010)	•	•	•	M
Martinez and Newman (2011)	•	–	•	Q
Askari-Nasab et al. (2011)	•	•	•	M
Chicoisne et al. (2012)	•	–	•	M
Nehring et al. (2012)	•	•	•	M
Ramazan and Dimitrakopoulos (2013)	•	•	•	M
Clausen (2013)	•	•	•	Q
Espinoza et al. (2013)	•	•	•	M
Smith and Wicks (2014)	•	•	•	Q
Lambert and Newman (2014)	•	–	•	M
O'Sullivan and Newman (2015)	•	•	•	Q
Montiel and Dimitrakopoulos (2015)	•	•	•	M
Lamghari and Dimitrakopoulos (2016)	•	–	•	M
Mousavi et al. (2016)	•	•	•	M
Jélvez et al. (2016)	•	–	•	M
Blom et al. (2016)	•	•	•	Other
Liu and Kozan (2016)	•	–	•	M
Vossen et al. (2016)	•	–	•	M
King et al. (2017)	•	–	•	M
Samavati et al. (2017)	•	–	•	M
Azzamouri et al. (2018)	•	•	•	Q
Reus et al. (2018)	•	–	•	M
Samavati et al. (2018)	•	–	•	M
Mousavi and Sellers (2019)	•	•	•	M
Mai et al. (2019)	•	•	•	M
Elsayed et al. (2020)	•	–	•	M
Jélvez et al. (2020)	•	•	•	M
Campeau and Gamache (2020)	–	–	•	M
Rivera Letelier et al. (2020)	•	•	•	M

•, constraints for the group are considered; –, constraints for the group are not considered

scheduling problem in the field of mining. Under columns "Constraints," we observe the three types of constraints we deal with in the problem at hand (*Quality*, *Quantity*, and *Precedence*). The "OF" column specifies whether the objective function is quantity- ("Q") or monetary-related ("M").

Regarding our quality-related objective function, we first have to calculate the weighted average of the output quality. Assume that a set of blocks $\mathcal{B}$ is available to be extracted. Let A_b and Q_b be the amount of material (in tonnes) and the

quality value (in percent) of block $b \in \mathcal{B}$, respectively. Moreover, let x_b be the binary decision variable that is 1 if block b is excavated. Then, the quality value of the entire extracted material is the weighted average of the quality values of the removed blocks, denoted by $\overline{q}$. The value of $\overline{q}$ (in %) is calculated as follows:

$$\overline{q} = \frac{\sum_{b \in \mathcal{B}} Q_b \cdot A_b \cdot x_b}{\sum_{b \in \mathcal{B}} A_b \cdot x_b}.$$

In the literature, the average proportion of the most valuable mineral contained in the ore is indicated as "ore grade." Martinez and Newman (2011) formulate a mixed-integer linear program to minimize the deviations from a given target demand for three different ore grades in LKAB's Kiruna iron ore mine. However, the target demand for a particular ore grade is given in tonnes. Accordingly, expressed in our notation, they only consider the linear term $\sum_{b \in \mathcal{B}} Q_b \cdot A_b \cdot x_b$ (the amount of extracted iron in tonnes) to measure the deviations. Thus, we categorized the proposed objective function as quantity-related. Azzamouri et al. (2018) minimize the deviations from the demand for a certain grade of the extracted material, where the objective function and the related constraints are formulated quantity-oriented, too. Analogously, Clausen (2013) determines the amount of potassium oxide contained in the extracted potash in tonnes to minimize the deviations from a given target value in an underground potash mine.

In the literature, the quality-related constraints are mostly known as "grade control constraints" or "grade blending constraints." Those constraints ensure that $\overline{q}$ is within a permitted tolerance range $[Q^-, Q^+]$. Thus, the following inequalities must apply:

$$Q^- \leq \frac{\sum_{b \in \mathcal{B}} Q_b \cdot A_b \cdot x_b}{\sum_{b \in \mathcal{B}} A_b \cdot x_b} \leq Q^+.$$

If we multiply both sides of the above inequalities with the denominator of $\overline{q}$, we obtain linear constraints (see, e.g., Rivera Letelier et al. 2020). Except for Montiel and Dimitrakopoulos (2015) and Blom et al. (2016), all the quality-related constraints observed in the papers from Table 2 are linear grade control constraints. Montiel and Dimitrakopoulos (2015) maximize discounted profits in an open-pit copper mine. They penalize the deviations regarding mining, processing, transportation, and blending targets and consider a penalty cost in the objective function. In the problem they consider, multiple material types are sent to the available processes or to stockpiles where they are blended to meet the quality requirements. The grade of the material handled in a given period corresponds to the grade of the stockpiles. Montiel and Dimitrakopoulos emphasize in their work that the corresponding blending constraint is nonlinear; consequently, the authors propose a risk-based heuristic approach to tackle the problem without suggesting any linear mathematical formulation. Blom et al. (2016) consider a multiple mine, multiple time-period, open-pit production scheduling problem. The authors define the productivity of a mine in terms of the desirable utilization of dig and trucking resources, i.e., dig and trucking resources should be fully utilized. In each period, ore produced at each mine

is transported by rail to a set of ports and blended into products for shipping. The objective function minimizes the deviation between the composition of port products and desired bounds and maximizes the productivity achieved at each mine. The observed objective function is denoted by "Other" in Table 2. The authors propose a nonlinear mathematical program to ensure blending constraints at each port while generating schedules for each mine. To tackle the problem, Blom et al. suggest a decomposition-based algorithm that can find high-quality solutions.

Our literature review suggests that no one, to the best of our knowledge, has introduced a linearization of the nonlinear quality-related constraints taking a quality-oriented objective function into account. It is straightforward to show that the scheduling problems observing upper and lower limits for the operational resources are $\mathcal{NP}$-hard. The computational time required for solving $\mathcal{NP}$-hard problems may be affected by the numerical parameters of the input data. A $\mathcal{NP}$-hard problem **in the strong sense** is still $\mathcal{NP}$-hard even when all of its numerical parameters are bounded by a polynomial in the length of the input. The contributions of this paper are:

- to introduce a **linear** mathematical program for the problem described;
- to show that the problem at hand is $\mathcal{NP}$-hard **in the strong sense**; and
- to introduce a solution approach that provides high-quality solutions for realistically sized problem instances.

In the next section, we first introduce the original mathematical program in its nonlinear structure and then linearize the corresponding nonlinear constraints. Lastly, we show that our problem is $\mathcal{NP}$-hard in the strong sense.

3 Mathematical model

In this section, we introduce a linear mathematical program for the block selection and sequencing problem described in Sect. 2. From an operational point of view, we consider $\tau_{\max}$ sub-intervals in the given planning horizon. Each sub-interval τ consists of several periods, where each period represents one work shift in the corresponding sub-interval. Each work shift is represented by time interval $(t-1, t]$. Let $T_{\max}$ denote the number of work shifts in the planning horizon under consideration. Thus, a planning horizon of $T_{\max}$ work shifts is a union of time intervals $(t-1, t]$ for $t = 1, 2, \ldots, T_{\max}$ and the point in time 0. Our mathematical formulation is based on the discrete completion times for the extraction of the blocks that are selected and mined in the considered planning horizon. Accordingly, we only focus on the discrete points in time that represent the end times of work shifts in the planning horizon. Note that a completion time at point 0 is not relevant since the point in time 0 is not the end time of any work shift in the planning horizon under consideration. Let $\mathcal{T}$ be the set of positive, discrete points in time in the entire planning horizon. Moreover, let $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_{\tau_{\max}}$ be the disjoint subsets of $\mathcal{T}$, where $\bigcup_{\tau \in \{1,\ldots,\tau_{\max}\}} \mathcal{T}_\tau = \mathcal{T}$. The elements of $\mathcal{T}_\tau$ are the positive, discrete points in time that represent the end times of the work shifts contained in sub-interval τ. In Fig. 3, a planning horizon

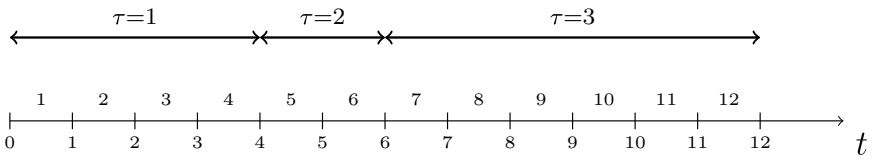

Fig. 3 The planning horizon and the associated sub-intervals

with $T_{\max} = 12$ work shifts and $\tau_{\max} = 3$ sub-intervals is depicted. The whole planning horizon is a union of disjoint time intervals [0, 4], (4, 6], and (6, 12]. Since the point in time 0 is not the end time of any work shift in $\mathcal{T}$, we have $\mathcal{T} = \{1, 2, \ldots, 12\}$, where $\mathcal{T}_1 = \{1, 2, 3, 4\}$, $\mathcal{T}_2 = \{5, 6\}$, and $\mathcal{T}_3 = \{7, 8, \ldots, 12\}$.

Table 3 demonstrates the sets, parameters, and decision variables used in the mathematical program. Note that the auxiliary decision variable $\overline{q}_\tau$ is used for better clarity in the mathematical formulation and is calculated by definition as follows:

$$\overline{q}_\tau = \frac{\sum_{t \in \mathcal{T}_\tau} \sum_{b \in \mathcal{B}} Q_b \cdot A_b \cdot x_{bt}}{\sum_{t \in \mathcal{T}_\tau} \sum_{b \in \mathcal{B}} A_b \cdot x_{bt}} \quad \forall \tau \in \{1, \ldots, \tau_{\max}\}.$$

The mathematical program for our problem is formulated as follows:

$$\text{minimize} \qquad \frac{1}{\tau_{\max}} \sum_{\tau \in \{1, \ldots, \tau_{\max}\}} (\delta_\tau^+ + \delta_\tau^-) \tag{1}$$

subject to

$$\sum_{t \in \mathcal{T}} x_{bt} \leq 1 \quad \forall b \in \mathcal{B} \tag{2}$$

$$\sum_{t \in \mathcal{T}} t \cdot x_{bt} = c_b \quad \forall b \in \mathcal{B} \tag{3}$$

$$Z_b \leq c_b + M \cdot (1 - \sum_{t \in \mathcal{T}} x_{bt}) \quad \forall b \in \mathcal{B} \tag{4}$$

$$\sum_{t \in \mathcal{T}} x_{lt} \leq \sum_{t \in \mathcal{T}} x_{bt} \quad \forall (b, l) \in \mathcal{V} \tag{5}$$

$$c_b \leq c_l - Z_l \cdot \left(\sum_{t \in \mathcal{T}} x_{lt} \right) + M \cdot (1 - \sum_{t \in \mathcal{T}} x_{lt}) \quad \forall (b, l) \in \mathcal{V} \tag{6}$$

Table 3 Sets, parameters, and decision variables used in the mathematical program

Sets	
$\mathcal{B}$	Set of blocks in an underground mine
$\mathcal{B}_k$	Set of blocks assigned to tipple area $k \in \mathcal{K}_r$
$\mathcal{B}_r$	Set of blocks assigned to mining district $r \in \mathcal{R}$
$\mathcal{K}_r$	Set of tipple areas in mining district $r \in \mathcal{R}$
$\mathcal{R}$	Set of mining districts in an underground mine
$\mathcal{T}$	Set of positive, discrete points in time in the given planning horizon
$\mathcal{T}_\tau$	Set of positive, discrete points in time contained in sub-interval τ ($\mathcal{T}_\tau \subset \mathcal{T}$)
$\mathcal{V}$	Set of precedence relations between blocks with each element representing an ordered pair of blocks; if $(b, l) \in \mathcal{V} : b, l \in \mathcal{B}$, b must be completed before l can be started
Parameters	
A_b	Amount of potash contained in block b in tonnes
M	A constant number with a sufficiently large value (big-M)
P^-	Minimum output that must be achieved over the planning horizon
P^-_r	Minimum output that must be achieved for mining district r over the planning horizon
P^-_t	Minimum output that must be achieved within time interval $(t-1, t]$
P^+_{kt}	Upper limit of the output for tipple area k within time interval $(t-1, t]$
P^+_{rt}	Upper limit of the output for mining district r within time interval $(t-1, t]$
Q_b	Percentage of potassium contained in the extracted material from block b (quality value of block b)
Q^-	Lower limit of the quality tolerance range in %
Q^+	Upper limit of the quality tolerance range in %
Q_τ	Quality target value within sub-interval τ in %
$T_{\max}$	Number of work shifts considered in the planning horizon
$\tau_{\max}$	Number of sub-intervals considered in the planning horizon
Z_b	Processing time required to extract block b measured in work shifts
Decision variables	
c_b	Positive continuous decision variable; completion time of block b
δ^-_τ	Positive continuous decision variable; negative deviations of the output quality from the quality target value over sub-interval τ
δ^+_τ	Positive continuous decision variable; positive deviations of the output quality from the quality target value over sub-interval τ
x_{bt}	Binary decision variable; 1, if the excavation of block b is completed in time interval $(t-1, t]$, and the material removed is available at point in time t; 0, otherwise
Auxiliary decision variables	
θ^-_{bt}	Positive continuous decision variable that substitutes the product of decision variables x_{bt} and δ^-_τ
θ^+_{bt}	Positive continuous decision variable that substitutes the product of decision variables x_{bt} and δ^+_τ
$\overline{q}_\tau$	Weighted average of the quality value of the extracted blocks during sub-interval τ

$$\sum_{t\in\mathcal{T}}\sum_{b\in\mathcal{B}} A_b \cdot x_{bt} \geq P^- \tag{7}$$

$$\sum_{b\in\mathcal{B}} A_b \cdot x_{bt} \geq P_t^- \quad \forall t \in \mathcal{T} \tag{8}$$

$$\sum_{t\in\mathcal{T}}\sum_{b\in\mathcal{B}_r} A_b \cdot x_{bt} \geq P_r^- \quad \forall r \in \mathcal{R} \tag{9}$$

$$\sum_{b\in\mathcal{B}_r} A_b \cdot x_{bt} \leq P_{rt}^+ \quad \forall r \in \mathcal{R}, \forall t \in \mathcal{T} \tag{10}$$

$$\sum_{b\in\mathcal{B}_k} A_b \cdot x_{bt} \leq P_{kt}^+ \quad \forall k \in \mathcal{K}_r : r \in \mathcal{R}, \forall t \in \mathcal{T} \tag{11}$$

$$\sum_{t\in\mathcal{T}_\tau}\sum_{b\in\mathcal{B}} (Q_b - Q^-) \cdot A_b \cdot x_{bt} \geq 0 \quad \forall \tau \in \{1, \ldots, \tau_{\max}\} \tag{12}$$

$$\sum_{t\in\mathcal{T}_\tau}\sum_{b\in\mathcal{B}} (Q^+ - Q_b) \cdot A_b \cdot x_{bt} \geq 0 \quad \forall \tau \in \{1, \ldots, \tau_{\max}\} \tag{13}$$

$$\overline{q}_\tau - Q_\tau \leq \delta_\tau^+ \quad \forall \tau \in \{1, \ldots, \tau_{\max}\} \tag{14}$$

$$Q_\tau - \overline{q}_\tau \leq \delta_\tau^- \quad \forall \tau \in \{1, \ldots, \tau_{\max}\} \tag{15}$$

$$x_{bt} \in \{0, 1\} \quad \forall b \in \mathcal{B}, \quad \forall t \in \mathcal{T} \tag{16}$$

$$c_b \geq 0 \quad \forall b \in \mathcal{B} \tag{17}$$

$$\delta_\tau^+, \delta_\tau^- \geq 0 \quad \forall \tau \in \{1, \ldots, \tau_{\max}\} \tag{18}$$

We minimize the average of the positive and negative deviations of the quality of the output from a given quality target value over the predetermined sub-intervals (cf. objective function (1)). A block can be excavated at most once within the entire planning horizon (constraint set (2)). By definition of decision variable x_{bt}, if $x_{bt} = 1$, the point in time t represents the completion time of block b. Constraint set (3) determines the completion times of blocks. Note that if a block is not excavated, the completion time is 0. Constraint set (4) guarantees that the completion time of a block must be greater than or equal to its processing time, i.e., the mining of a block must start during the considered planning horizon. Constraint set (4) is active only if a block is excavated. In the corresponding big-M formulation, we can choose M equal to $T_{\max}$. Constraint sets (5) and (6) observe a precedence relation

between blocks b and l. If the ordered pair (b, l) is in $\mathcal{V}$, block l cannot be mined if block b is not excavated (constraint set (5)). Moreover, constraint set (6) ensures that the completion time of block l must be at least by Z_l time units greater than the completion time of block b if block l is ever processed. Constraint sets (7) to (11) ensure the minimum and maximum limits of the output. Constraint sets (12) and (13) guarantee that the weighted average of the quality value of the excavated blocks during every sub-interval $(\overline{q}_\tau)$ is within a permitted tolerance range. Those constraints are like "grade control constraints" or "grade blending constraints" explained in Sect. 2.

Constraint sets (14) and (15) determine the lower bounds for δ_τ^+ and δ_τ^-, respectively. Since objective function (1) must be minimized, δ_τ^+ and δ_τ^- take the value of the positive and negative deviations of the quality of the output from Q_τ. However, the constraint sets are inherently nonlinear. We write the mathematical formula of $\overline{q}_\tau$ in the inequalities and reduce the left-hand side of each inequality to a common denominator. If we multiply both sides of the inequalities by the common denominator, we obtain the following nonlinear inequalities:

$$\sum_{t\in\mathcal{T}_\tau}\sum_{b\in\mathcal{B}}(Q_b - Q_\tau)\cdot A_b\cdot x_{bt} \le \sum_{t\in\mathcal{T}_\tau}\sum_{b\in\mathcal{B}} A_b\cdot x_{bt}\cdot \delta_\tau^+ \quad \forall\tau\in\{1,\ldots,\tau_{\max}\}$$

$$\sum_{t\in\mathcal{T}_\tau}\sum_{b\in\mathcal{B}}(Q_\tau - Q_b)\cdot A_b\cdot x_{bt} \le \sum_{t\in\mathcal{T}_\tau}\sum_{b\in\mathcal{B}} A_b\cdot x_{bt}\cdot \delta_\tau^- \quad \forall\tau\in\{1,\ldots,\tau_{\max}\}$$

In order to formulate a mixed-integer linear program, we introduce positive continuous decision variables θ_{bt}^+ and θ_{bt}^- to substitute the products $x_{bt}\cdot\delta_\tau^+$ and $x_{bt}\cdot\delta_\tau^-$ on the right-hand sides of the above inequalities, respectively. We, therefore, have the following constraint sets:

$$\sum_{t\in\mathcal{T}_\tau}\sum_{b\in\mathcal{B}}(Q_b - Q_\tau)\cdot A_b\cdot x_{bt} \le \sum_{t\in\mathcal{T}_\tau}\sum_{b\in\mathcal{B}} A_b\cdot \theta_{bt}^+ \quad \forall\tau\in\{1,\ldots,\tau_{\max}\} \tag{14-1}$$

$$\sum_{t\in\mathcal{T}_\tau}\sum_{b\in\mathcal{B}}(Q_\tau - Q_b)\cdot A_b\cdot x_{bt} \le \sum_{t\in\mathcal{T}_\tau}\sum_{b\in\mathcal{B}} A_b\cdot \theta_{bt}^- \quad \forall\tau\in\{1,\ldots,\tau_{\max}\} \tag{15-1}$$

By substituting a product of a binary decision variable and a continuous decision variable, the upper bound for the substitution variable (here θ_{bt}^+ and θ_{bt}^-) must be determined. The upper bound is related to the maximum value that the continuous decision variable can take if the binary decision variable is 1. Furthermore, the substitution variable takes the value 0 if the binary decision variable is 0. Constraint sets (14-2) and (15-2) guarantee that decision variables θ_{bt}^+ and θ_{bt}^- take the value of zero for all $t\in\mathcal{T}_\tau$ if block b is not excavated within sub-interval τ. Note that each block can be mined only once within the entire planning horizon. Thus, it is sufficient if we consider the summation of decision variables θ_{bt}^+ (θ_{bt}^-) and x_{bt} over the points in time $t\in\mathcal{T}_\tau$. Otherwise, if a block is removed at any point in time $t\in\mathcal{T}_\tau$, the left-hand side of constraint set (14-2) (constraint set (15-2)) can at most have the value of $Q^+ - Q_\tau$ $(Q_\tau - Q^-)$:

$$\sum_{t\in\mathcal{T}_\tau}\theta^+_{bt} \le (Q^+ - Q_\tau)\cdot\sum_{t\in\mathcal{T}_\tau}x_{bt} \quad \forall b\in\mathcal{B},\forall\tau\in\{1,\ldots,\tau_{\max}\} \tag{14-2}$$

$$\sum_{t\in\mathcal{T}_\tau}\theta^-_{bt} \le (Q_\tau - Q^-)\cdot\sum_{t\in\mathcal{T}_\tau}x_{bt} \quad \forall b\in\mathcal{B},\forall\tau\in\{1,\ldots,\tau_{\max}\} \tag{15-2}$$

Note that the maximum values of the positive deviation (δ^+_τ) and the negative deviation (δ^-_τ) are bounded by $Q^+ - Q_\tau$ and $Q_\tau - Q^-$, respectively. Subsequently, the following constraint sets determine the values of δ^+_τ and δ^-_τ that are used in (1):

$$\sum_{t\in\mathcal{T}_\tau}\theta^+_{bt} \le \delta^+_\tau \quad \forall b\in\mathcal{B},\forall\tau\in\{1,\ldots,\tau_{\max}\} \tag{14-3}$$

$$\sum_{t\in\mathcal{T}_\tau}\theta^-_{bt} \le \delta^-_\tau \quad \forall b\in\mathcal{B},\forall\tau\in\{1,\ldots,\tau_{\max}\} \tag{15-3}$$

If we replace constraint sets (14), as well as (15) by constraint sets (14-1), (14-2), and (14-3) as well as constraint sets (15-1), (15-2), and (15-3), respectively, we have a mixed-integer linear program, where constraint set (16) indicates that decision variables x_{bt} are binary, and in addition to non-negativity constraint sets (17) and (18), the following non-negativity constraint set must be considered:

$$\theta^+_{bt},\theta^-_{bt}\ge 0 \quad \forall b\in\mathcal{B},\forall t\in\mathcal{T} \tag{19}$$

We denote the proposed mixed-integer linear program for our **b**lock **s**election and **s**equencing **p**roblem with **BSSP**.

In what follows, we show that our block selection and sequencing problem is $\mathcal{NP}$ -hard in the strong sense. We introduce a restricted case of BSSP (RBSSP) to which a pseudo-polynomial transformation from the well-known 3-PARTITION problem can be easily constructed.

We consider an underground mine that has one mining district, over a planning horizon of $T_{\max} = T$ work shifts with $\tau_{\max} = 1$. Let the number of blocks be $n = 3T$. We assume that there is only one block in each underground location ($\mathcal{V} = \emptyset$). Let the processing time of all blocks be equal to 1 work shift. Hence, constraint sets (3) to (6) do not have to be observed. Moreover, we assume that the maximum output for each tipple area is a very large number with the result that constraint set (11) is always satisfied. Since there is only one mining district in the restricted problem, constraint sets (7) and (9) are the same. By assuming the minimum output for the entire planning horizon equal to $\sum_{t\in\mathcal{T}}P^-_t$, constraint sets (7) and (9) are redundant. Furthermore, let the quality value of all blocks be equal to Q with $Q^- < Q < Q^+$. Thus, the deviation from the quality target value is a constant number regardless of which blocks have been excavated. Consequently, all of the quality-related constraint sets are met. We assume $P^+_{1t} = P^-_t = (\sum_{b=1}^{3T}A_b)/T$. Hence, we can denote $P^+_{1t} = P^-_t$ with P. Consequently, the restricted problem, RBSSP, can be formulated as follows:

Min. Constant number

$$\text{s. t.} \quad \sum_{b=1}^{3T} x_{bt} \leq 1 \quad \forall t \in \{1, \ldots, T\}$$

(cf. constraint set (2))

$$\sum_{b=1}^{3T} (A_b \cdot x_{bt}) \geq P \quad \forall t \in \{1, \ldots, T\}$$

(cf. constraint set (8))

$$\sum_{b=1}^{3T} (A_b \cdot x_{bt}) \leq P \quad \forall t \in \{1, \ldots, T\}$$

(cf. constraint set (10))

$$x_{bt} \in \{0, 1\} \qquad \forall b \in \{1, \ldots, 3T\},\ \forall t \in \{1, \ldots, T\}$$

RBSSP is not an optimization, but a so-called feasibility problem, i.e., a specific decision problem. In this restricted problem, we intend to determine T subsets $\mathcal{B}_1$, $\mathcal{B}_2, \cdots, \mathcal{B}_T$ of blocks that are removed at points in time $1, 2, \ldots, T$, respectively, subject to the constraint sets of the problem.

Garey and Johnson (1979) showed that 3-PARTITION is $\mathcal{NP}$-complete in the strong sense.

- **3-PARTITION**

  Input: a finite set $\mathcal{U} = \{u_1, u_2, \ldots, u_{3m}\}$, a bound $X \in \mathbb{N}$, and a size $s(u_b) \in \mathbb{N}$ for each $b = 1, \ldots, 3m$, such that $\frac{X}{4} < s(u_b) < \frac{X}{2}$ for each b and $\sum_{b=1}^{3m} s(u_b) = mX$.

  Question: are there m disjoint subsets $\mathcal{U}_1, \mathcal{U}_2, \ldots, \mathcal{U}_m$ of $\mathcal{U}$ such that:

$$\sum_{u_b \in \mathcal{U}_b} s(u_b) = X \qquad \forall b \in \{1, \ldots, m\}?$$

If we consider each element u_b of the given set $\mathcal{U}$ in 3-PARTITION as a block $b \in \mathcal{B}$ in RBSSP, a transformation from an arbitrary instance of 3-PARTITION to an instance of RBSSP is given by $T = m$, $A_b = s(u_b)$, and $P = X$. This transformation

can be performed in time polynomially in the input length. The length of the constructed RBSSP-instance is polynomially related to the length of the given 3-PARTITION instance. Furthermore, the largest number in the constructed instance is equal to the largest number of the given 3-PARTITION instance; consequently, the conditions of a pseudo-polynomial transformation are met. In a solution of RBSSP, we have T subsets of $\mathcal{B}$. Those subsets play the same role as the sets $\mathcal{U}_1, \ldots, \mathcal{U}_m$ in the desired partition of $\mathcal{U}$ in the 3-PARTITION. As a result, a solution for RBSSP exists if and only if the desired partition exists for the given 3-PARTITION instance. Thus, RBSSP is $\mathcal{NP}$-complete in the strong sense, and optimization problem BSSP is $\mathcal{NP}$-hard in the strong sense.

For $\mathcal{NP}$-hard problems in the strong sense, an optimal solution, especially for large and challenging problem instances, cannot generally be found using a MILP-solver within a reasonable amount of time. In the next section, we propose a heuristic approach that solely seeks a subset of the feasible region using some rules to provide good, feasible solutions.

4 Heuristic solution procedure

In this section, we introduce a constructive heuristic that is embedded in a so-called *multi-start algorithm*. Constructive heuristics gradually generate a complete solution based on a partial solution. In our heuristic algorithm, at each point in time t, eligible blocks are determined according to two different factors. On the one hand, a block is eligible if it can be completed at the considered point in time according to its processing time and the completion status of its predecessors. On the other hand, a block is not eligible if its extraction results in an overrun of the upper limit of the output in the associated tipple area and the related mining district. Based on a specific priority rule, the elements of the eligible set are prioritized. Then, a roulette-wheel selection procedure is applied to randomly select a block that is scheduled next in the planning horizon (its completion time is set to t). After that, the status of the mine and, accordingly, the eligible set are updated. The selection procedure continues until the eligible set at the considered point in time is empty. By the use of the roulette-wheel selection procedure, the next block to be scheduled is randomly selected from the eligible set. That means, if the method is carried out several times

in a multi-start environment, it is very likely that a large number of feasible solutions are generated. Table 4 outlines the sets, parameters, and variables used in our heuristic algorithm.

Algorithm 1 describes the developed multi-start heuristic in detail.

Algorithm 1 Multi-start algorithm with priority rules

1: Input: a problem instance, optimality interval $[0, \xi]$, priority rule Ψ, and a time limit
2: **while** time limit is not achieved, and no solution has an objective value within the predefined optimality interval **do**
3: initialization;
4: **for all** $t \in \mathcal{T}$ **do**
5: determine τ;
6: **for all** $b \in \mathcal{E}$ **do**
7: **if** $t - Z_b - es_b \geq 0$ **then**
8: insert b into set $\mathcal{E}^{\mathcal{T}}$;
9: **if** $\mathcal{E}^{\mathcal{T}} = \emptyset$ **then**
10: terminate; (no feasible solution found)
11: $\mathcal{E}^{\mathcal{CT}} := \emptyset$;
12: **for all** $b \in \mathcal{E}^{\mathcal{T}}$ **do**
13: **if** $A_b \leq \rho^{\text{res}}_{R_b,t} \wedge A_b \leq \rho^{\text{res}}_{K_b,t}$ **then**
14: insert b into set $\mathcal{E}^{\mathcal{CT}}$;
15: **repeat**
16: choose $b^* \in \mathcal{E}^{\mathcal{CT}}$ with respect to priority rule Ψ;
17: update $\rho^{\text{res}}_{R_{b^*},t}$ and $\rho^{\text{res}}_{K_{b^*},t}$;
18: update $\overline{q}_\tau$;
19: **if** block l is the successor of b^* **then**
20: insert l in $\mathcal{E}$;
21: $es_l := t$;
22: remove b^* from $\mathcal{E}$;
23: remove b^* and all $b \in \mathcal{E}^{\mathcal{CT}}$ with $R_b = R_{b^*}$ and $K_b = K_{b^*}$ for which $A_b > \rho^{\text{res}}_{R_{b^*},t}$ or $A_b > \rho^{\text{res}}_{K_{b^*},t}$ from $\mathcal{E}^{\mathcal{CT}}$;
24: **until** $\mathcal{E}^{\mathcal{CT}} = \emptyset$
25: **if** constraint sets (7), (8), (9), (12), and (13) are satisfied **then**
26: store the solution;
27: **else**
28: discard the solution;
29: **return** the best solution found

In our minimization problem, all of the decision variables are non-negative. Thus, the objective function cannot take a value smaller than 0. With the prescribed value of $\xi > 0$, we denote a tiny quality tolerance value of the production process. Accordingly, a feasible solution is called an optimal solution if the associated value of the objective function lies in the narrow interval $[0, \xi]$. For a given problem instance, feasible solutions are generated using priority rule Ψ until the objective value of a feasible solution is within the predefined interval or a given time limit is exceeded (while-loop at line 2). Within an *initialization* step (line 3 in Alg. 1), we store for

Table 4 Sets, parameters, and variables used in the heuristic algorithm

Sets	
$\mathcal{E}$	Set of blocks that have no predecessor, or their predecessor blocks are already excavated
$\mathcal{E}^{\mathcal{CT}}$	Set of blocks that can be excavated regarding the considered point in time and the considered output capacities ($\mathcal{E}^{\mathcal{CT}} \subset \mathcal{E}^{\mathcal{T}}$)
$\mathcal{E}^{\mathcal{T}}$	Set of blocks that can be excavated regarding the considered point in time ($\mathcal{E}^{\mathcal{T}} \subset \mathcal{E}$)
Parameters	
ϵ	A tiny number that is set to 0.0001
K_b	Tipple area that contains block $b \in \mathcal{B}$
Ψ	Prescribed priority rule; $\Psi \in \{1, 2, 3, 4\}$
R_b	Mining district that contains block $b \in \mathcal{B}$
ξ	A tiny quality tolerance value of the production process
Variables	
b^*	Selected block from $\mathcal{E}^{\mathcal{CT}}$ (using a roulette-wheel-selection procedure)
es_b	Earliest start time of block b according to the completion time of its predecessor; if block b does not have any predecessor, es_b is 0
ψ_b	Probability value of block b
π_b	Priority value of block b according to prescribed priority rule Ψ
ρ_{kt}^{res}	Residual capacity of the output for tipple area k in time interval $(t-1, t]$
ρ_{rt}^{res}	Residual capacity of the output for mining district r in time interval $(t-1, t]$
$\overline{q}_\tau^b$	New value of $\overline{q}_\tau$ if block b is mined next in sub-interval τ in %

each block $b \in \mathcal{B}$ the associated mining district R_b and tipple area K_b. Moreover, we store the blocks with no predecessor block in set $\mathcal{E}$. For all $b \in \mathcal{E}$, the earliest start time is 0 ($es_b = 0$). Furthermore, the residual capacities ρ_{rt}^{res} and ρ_{kt}^{res} are set to the corresponding maximum limits of the output P_{rt}^{+} and P_{kt}^{+}, respectively.

After the initialization step, we pass the entire planning horizon in a for-loop (line 4). For the current point in time t, we determine the related sub-interval τ and put those blocks from $\mathcal{E}$ into set $\mathcal{E}^{\mathcal{T}}$ that can be completed until t (if-condition at line 7). At the beginning of the planning horizon ($t = 1$), the earliest start times of all blocks $b \in \mathcal{E}$ are 0. Accordingly, only the blocks $b \in \mathcal{E}$ with processing times $Z_b = 1$ can be added to $\mathcal{E}^{\mathcal{T}}$. If no block can be inserted into $\mathcal{E}^{\mathcal{T}}$, the output at the point in time t is 0. Hence, constraint set (8) is violated, and no feasible solution can be found. Thus, the algorithm terminates (line 10). Otherwise, blocks from $\mathcal{E}^{\mathcal{T}}$ are reconsidered according to the residual capacities of the corresponding mining districts and tipple areas. For block $b \in \mathcal{E}^{\mathcal{T}}$, if A_b does not exceed the associated residual capacities (if-condition at line 13), b is inserted into $\mathcal{E}^{\mathcal{CT}}$. In a realistic problem instance, we always have:

$$\max_{b \in \mathcal{B}} \{A_b\} \leq P^+_{R_b,t} \quad \forall t \in \mathcal{T}, \text{ and}$$
$$\max_{b \in \mathcal{B}} \{A_b\} \leq P^+_{K_b,t} \quad \forall t \in \mathcal{T}.$$

Therefore, at least one block from $\mathcal{E}^{\mathcal{T}}$ (if there is any) can always be inserted into $\mathcal{E}^{\mathcal{CT}}$ at each point in time. The block that must be scheduled next is selected from $\mathcal{E}^{\mathcal{CT}}$ according to the given priority rule Ψ (line 16). For our heuristic, we consider four different priority rules. Based on priority rule $\Psi \in \{1, 2, 3, 4\}$, we determine for each block b in $\mathcal{E}^{\mathcal{CT}}$ a priority value π_b. In the following, we explain the way how π_b is calculated according to a specific priority rule.

priority rule 1: $\pi_b = \dfrac{1}{|Q_b - Q_\tau| + \epsilon}$

For this rule, we put emphasis on the quality value of a block. A block with a smaller deviation from the quality target value takes a larger priority value. This priority rule gives a block that may contribute to a better objective function value a greater probability to be extracted next. Since some blocks may have no deviation from the quality target value, we add a tiny number $\epsilon > 0$ to the denominator of the fraction. In our work, we set $\epsilon = 0.0001$.

priority rule 2: $\pi_b = \dfrac{A_b}{|Q_b - Q_\tau| + \epsilon}$

For priority rule 2, we additionally consider the amount of material extracted from a block. If quality values of blocks are the same, a block with a larger amount of material is given a larger priority value, or—in other words—for the same amount of material, a block that has a smaller deviation from the quality target receives a larger priority value (like priority rule 1). Using priority rule 2, the lower limits of output are also considered to avoid generating infeasible solutions.

priority rule 3: $\pi_b = \dfrac{1}{|\overline{q}^b_\tau - Q_\tau| + \epsilon}$

Here, we calculate the value of $\overline{q}^b_\tau$ that represents the change of $\overline{q}_\tau$ if block b is excavated next. A block takes a larger priority value if its excavation results in a smaller deviation from the quality target value. We can, therefore, give a more considerable priority value to the blocks that allow the best possible improvement in the objective function value in each step.

priority rule 4: $\pi_b = \begin{cases} \epsilon, & \text{if } P^-_{R_b} \text{ is achieved;} \\ A_b, & \text{otherwise.} \end{cases}$

Priority rule 4 helps to avoid infeasible solutions in terms of constraint set (9). For block b, if the lower limit of the output for the corresponding mining district R_b is achieved, we set the priority value of block b equal to a tiny number $\epsilon > 0$. Otherwise, block b receives a priority value equal to A_b. Using priority rule 4, we focus only on finding feasible solutions regarding the lower limits of output.

According to the priority values, all blocks $b \in \mathcal{E}^{\mathcal{CT}}$ receive a probability value ψ_b as follows:

$$\psi_b = \frac{\pi_b}{\sum_{l \in \mathcal{E}^{\mathcal{CT}}} \pi_l}.$$

We apply a roulette-wheel selection procedure, where each block occupies an area on the roulette-wheel proportional to its ψ_b-value (Michalewicz and Fogel 2004, Sect. 6.1). That is equivalent to partitioning the interval [0, 1] into $|\mathcal{E}^{\mathcal{CT}}|$ parts, where the b-th sub-interval (part) has the width ψ_b representing block b. Subsequently, a random number between 0 and 1 is generated. The sub-interval that contains the random number determines block b^*. At lines 17 and 18, the associated residual capacities and the value of $\overline{q}_\tau$ are updated. If b^* has a successor l (line 19), we insert l in $\mathcal{E}$. The earliest start time of l is set equal to the completion time of b^*, i.e., $es_l = t$ (line 21). Block b^* is then removed from $\mathcal{E}^{\mathcal{CT}}$. Moreover, all blocks $b \in \mathcal{E}^{\mathcal{CT}}$ with $R_b = R_{b^*}$ and $K_b = K_{b^*}$, for which $A_b > \rho^{\text{res}}_{R_{b^*},t}$ or $A_b > \rho^{\text{res}}_{K_{b^*},t}$, have to be removed from $\mathcal{E}^{\mathcal{CT}}$ (lines 22 and 23). The repeat-until-loop in Alg. 1 will be executed until $\mathcal{E}^{\mathcal{CT}}$ is empty.

After violating the condition of the repeat-until-loop, t is incremented by one. The entire for-loop at line 4 will be executed for all $t \in \mathcal{T}$. We then check the feasibility of the solution found according to constraint sets (7), (8), (9), (12), and (13). Feasible solutions are stored, and infeasible solutions are discarded.

We observe the loops in Alg. 1 to determine the time complexity of the presented heuristic approach in every run. The for-loop at line 4 is executed $T_{\max}$ times. Moreover, the for-loop at line 6 and the repeat-until-loop are conducted, at most, $|\mathcal{B}|$ times. Line 11, the for-loop at line 12, as well as line 16, if-condition at line 19, and lines 22 and 23 within the repeat-until-loop have, at most, $|\mathcal{B}|$ steps. Accordingly, the algorithm has a time complexity of $\mathcal{O}(T_{\max} \cdot |\mathcal{B}|^2)$ that implies a pseudo-polynomial algorithm. Note that in practical applications, $T_{\max}$ is given as a constant number. Therefore, the time complexity of the algorithm is $\mathcal{O}(|\mathcal{B}|^2)$, and the heuristic proposed is polynomial.

In the next section, we compare the results achieved by a MILP-solver with the solutions provided by the proposed heuristic approach. Moreover, we introduce a sophisticated combination of the heuristic and the mathematical program to improve the results obtained by the heuristic procedure for the most challenging problem instances, for which our MILP-solver cannot find a feasible solution within a reasonable amount of time.

5 Computational study

In this section, we perform an experimental performance analysis for the proposed mixed-integer linear program and the constructive heuristic. The computational study is executed on randomly generated problem instances that are based on real-world data derived from Clausen (2013). Table 5 shows some parameters that are typical for a potash mine and used to generate realistic problem instances.

To normalize the problem instances, we set the number of blocks in each mining district r equal to 325. Thus, the problem instances with more underground locations have fewer blocks in each underground location and vice versa.

Table 5 Parameters used to generate realistic problem instances

Parameters	Symbols	Values
Number of mining districts in an underground mine		1–5
Number of tipple areas in each mining district		4–6
Number of underground locations in each tipple area		5–11
Number of blocks in each underground location		5–10
Amount of material contained in each block in tonnes	A_b	700–1200
Quality value of each block in %	Q_b	9.8–16.2
Lower limit of the quality value of the output in %	Q^-	11.1
Upper limit of the quality value of the output in %	Q^+	14.1
Quality target value for each sub-interval in %	Q_τ	12.6
Processing time of the first blocks in each underground location measured in work shifts	Z_b	1–9
Processing time of other (not the first) blocks in each underground location measured in work shifts		6–12

The allowed maximum output in each mining district r for time interval $(t-1, t]$ depends on the number of loaders available in the corresponding mining district during the associated time interval. A loader can typically transport 750 tonnes of crude material within a work shift. In each work shift, 1 to 4 loader(s) are available so that $P^+_{rt} \in \{750, 1500, 2250, 3000\}$. The minimum output in each mining district depends on the capacity of loaders, too. The parameter P^-_r can be determined as a certain percentage γ of the capacity of all the available loaders in mining district r within the entire time horizon:

$$P^-_r = \gamma \cdot \sum_{t \in \mathcal{T}} P^+_{rt} \qquad \forall r \in \mathcal{R}.$$

The parameter γ varies between 70% and 90%, with a step size of 5%. Furthermore, the lower limit of the total output is the sum of P^-_r over the mining districts contained in the underground mine:

$$P^- = \sum_{r \in \mathcal{R}} P^-_r.$$

The upper limits of the output for tipple areas P^+_{kt} are randomly chosen from the set $\{1000, 2000, \ldots, 5000\}$. Those numbers are given due to the characteristics of typical feeder breakers. Even though the parameters P^+_{kt} have different values in terms of tipple areas k, they are the same for all time intervals $(t-1, t]$ in the planning horizon. In each time interval $(t-1, t]$, an output greater than 0 has to be achieved. We set the parameter P^-_t as follows:

$$P^-_t = \min_{b \in \mathcal{B}} \{A_b\} \qquad \forall t \in \mathcal{T}.$$

We consider a planning horizon of one month that without loss of generality, is supposed to have only four weeks. Each sub-interval τ represents a time interval of one

week ($\tau_{\max} = 4$). A week contains 18 work shifts, so that the planning horizon has $T_{\max} = 72$ work shifts.

The quality values of blocks are stated in percentage and usually have decimals. We multiply the quality values of blocks and, accordingly, the quality parameters from Table 5 by 100 to avoid rounding errors. For example, a quality value of 13.7% is 1370 in a problem instance. As mentioned in Sect. 4, from a practical point of view, a solution is called an optimal solution if the associated value of the objective function lies within the small interval of $[0, \xi]$ with $\xi > 0$. Parameter ξ is the quality tolerance value of the production process and is chosen to be 0.1%. Like the quality values of blocks, we multiply ξ by 100 as well; ξ is, therefore, equal to 10.

We distinguish between 5 different test sets according to the number of mining districts. Hence, there is 1 mining district in the first test set, 2 mining districts in the second test set, and so on (5 mining districts in the fifth test set). We randomly generated 20 problem instances for each test set using the introduced parameter ranges. Additionally, we consider five different levels of γ, i.e., $\gamma \in \{70\%, 75\%, \ldots, 90\%\}$, concerning the lower limit of the output (P_r^- and accordingly P^-, see above). Thus, we have 100 problem instances for each γ-level. Test sets with 5 mining districts represent the largest problem instances. On the other hand, the problem instances with a γ-level of 90% are the most challenging problem instances to solve.

We used GAMS 24.9 to implement our mixed-integer linear program and solved the problem instances using CPLEX 12.7.1. We set the solver parameters in the way that the solution procedure terminates in the following cases:

1. if a solution found is optimal;
2. if a solution found is in the predefined interval [0, 10]; or
3. if a time limit of 1800 s is exceeded.

The corresponding solution procedure is called **CPLEX**. According to the cases mentioned above, if we say that CPLEX finds an optimal solution for a problem instance, case 1. or case 2. holds.

The heuristic algorithm is implemented in the programming language **C++** and executed with the compiler Microsoft Visual Studio 2010. Since we use four different priority rules in the heuristic, we set the upper limit of the solution time equal to 450 s for each priority rule. Thus, for each problem instance, we run the multi-start heuristic approach for 1800 (4×450) s to have a relatively fair comparison with the results of CPLEX, which has a time limit of 1800 s, too. The resulting multi-start heuristic approach with four randomized priority rules is called **MSH-4R**.

All tests are executed on an Intel(R) i7-7700K@4.20GHz machine with 64 GB RAM under Windows 10.

We can compare the results achieved by CPLEX and MSH-4R from different aspects. On the one hand, the number of problem instances for which no feasible solution can be provided is important. On the other hand, we want to know whether a feasible solution obtained is optimal, and if not, what can be said about the solution quality. We discussed that the problem instances with a γ-level of 90% are the most challenging problem instances to solve, and the ones with 5 mining districts

are the largest problem instances. Accordingly, the results of each solution procedure for each γ-level and regarding the size of the problem instances are of most interest. Our computational study suggests the following results:

- In general, the greater the problem instances, the more time CPLEX needs to find an optimal solution.
- For γ-levels of 70% to 85%, CPLEX finds an optimal solution for almost all problem instances. However, for the most challenging problem instances with $\gamma = 90\%$, CPLEX can hardly find any feasible solution.
- MSH-4R can find for all of the problem instances at least one feasible solution that is not far from the best solutions found.
- MSH-4R using priority rule 3 provides the most high-quality solutions in comparison to the other priority rules.
- If we use the solutions found by MSH-4R using priority rule 3 as initial solutions for CPLEX, the results found for the most challenging problem instances are improved by, on average, 92.5%.

A detailed explanation of the results achieved by the solution approaches is given in the following.

Table 6 compares the results achieved by CPLEX with the solutions found by MSH-4R. Each row in Table 6 shows the information for a test set that consists of 20 problem instances. Columns # *Optimal* depict the number of test instances for which CPLEX and MSH-4R could find an optimal solution (see the explanation for an optimal solution above). The number of test instances for which a feasible solution could be found but the optimality of the solution could not be proven or shown is given under columns # *Feasible* for both solution procedures. Columns # *Unknown* present the number of test instances for which no feasible solution is found within the considered time limit (1800 s). Note that the numbers appearing under each column cannot be greater than 20. Columns *Solution time* report the average solution time in seconds for the considered 20 problem instances in each test set. According to the setting, the average solution time cannot be greater than 1800 s. Analogous to CPLEX, MSH-4R terminates if a solution found by a given priority rule is in the predefined interval [0, 10]. In that case, we say that MSH-4R finds an optimal solution for a problem instance. Note that MSH-4R is not able to state whether a feasible solution with an objective function value greater than 10 is optimal or not. For a problem instance solved by MSH-4R, an optimal solution may be found by using any priority rule. In those cases, the procedure terminates before the time limit (450 s) is exceeded for the respective priority rule. The time in seconds under *Solution time* MSH-4R is the sum of solution times taken by every priority rule and cannot be greater than 1800 s (4×450).

We see in Table 6 that CPLEX could find for the whole 100 problem instances with $\gamma = 70\%$ an optimal solution. The greater the number of mining districts in a test set, the more time CPLEX needs to find an optimal solution. For $\gamma = 70\%$, the solution time is, on average, 14 s for the 20 problem instances with 1 mining district and 132 s for problem instances with 5 mining districts. For the whole 100 problem instances with $\gamma = 70\%$, CPLEX needed, on average, 53 s to prove

Table 6 Comparison of the results achieved by CPLEX with the solutions found by MSH-4R (each row represents a test set including 20 problem instances)

γ[in %]	$\lvert\mathcal{R}\rvert$	# Optimal		# Feasible		# Unknown		Solution time [s]		Ave. deviation from best. [$\frac{\%}{100}$]	
		CPLEX	MSH-4R	CPLEX	MSH-4R	CPLEX	MSH-4R	CPLEX	MSH-4R	CPLEX	MSH-4R
70	1	20	7	0	13	0	0	14	1392	0.00	7.42
	2	20	3	0	17	0	0	22	1615	0.00	8.40
	3	20	0	0	20	0	0	44	1800	0.00	14.60
	4	20	2	0	18	0	0	52	1740	0.00	12.74
	5	20	0	0	20	0	0	132	1800	0.00	13.35
75	1	20	7	0	13	0	0	26	1392	0.00	7.42
	2	20	3	0	17	0	0	62	1621	0.00	8.40
	3	20	0	0	20	0	0	241	1800	0.00	14.60
	4	20	2	0	18	0	0	279	1740	0.00	12.74
	5	20	0	0	20	0	0	330	1800	0.00	13.35
80	1	18	7	2	13	0	0	202	1392	0.00	6.48
	2	20	3	0	17	0	0	230	1615	0.00	8.40
	3	20	0	0	20	0	0	404	1800	0.00	14.60
	4	20	2	0	18	0	0	473	1740	0.00	12.75
	5	20	0	0	20	0	0	676	1800	0.00	13.35
85	1	18	7	2	13	0	0	324	1392	0.02	5.74
	2	17	3	3	17	0	0	568	1615	0.00	7.13
	3	18	0	2	20	0	0	788	1800	0.00	14.04
	4	19	2	1	18	0	0	904	1740	0.00	12.51
	5	18	0	2	20	0	0	1020	1800	0.00	12.86
90	1	7	6	1	14	12	0	1241	1460	0.00	1.07
	2	0	2	1	18	19	0	1800	1668	0.00	0.36
	3	0	0	0	20	20	0	1800	1800	–	0.00
	4	0	1	0	19	20	0	1800	1762	–	0.00
	5	0	0	0	20	20	0	1800	1800	–	0.00

the optimality of the solutions found. For $\gamma = 75\%$, there is the same trend. All of the test instances are solved to optimality by CPLEX; the solution time is, on average, 26 s for problem instances with 1 mining district and 330 s for problem instances with 5 mining districts. Notably, the computational times are higher (for the 100 problem instances with $\gamma = 75\%$, the computational time is, on average, 187.6 s) since the problem instances are more challenging. For γ-levels of 80% and 85%, the same trend can be seen. For $\gamma = 80\%$ ($\gamma = 85\%$), CPLEX could optimally solve 98 (90) problem instances. At the γ-level of 80% (85%), the solution time is, on average, 202 (324) s for the 20 problem instances with 1 mining district and 676 (1020) s for the 20 problem instances with 5 mining districts. For the other problem instances with $\gamma = 80\%$ and $\gamma = 85\%$ that could not be solved

to optimality, CPLEX could find a feasible solution (cf. columns # *Feasible*). For the most challenging problem instances with $\gamma = 90\%$, CPLEX could find for only seven problem instances with 1 mining district an optimal solution within, on average, 1241 s. Furthermore, for only two other problem instances, a feasible solution could be found by CPLEX. In other words, for 91 problem instances at the γ-level of 90%, CPLEX cannot find any feasible solutions within a reasonable amount of time.

On the contrary, MSH-4R was able to find an optimal solution for 12 problem instances each at the γ-levels of 70%, 75%, 80%, and 85% as well as for nine problem instances with $\gamma = 90\%$ (altogether 57 of the whole 500 problem instances). Hence, it cannot be said at which level MSH-4R works best. MSH-4R could find for every problem instance at every level and with every size at least one feasible solution (the numbers under # *Unknown* MSH-4R are all 0), even for the γ-level of 90%. To evaluate the solution quality of the results achieved by MSH-4R, we calculate the average of the absolute deviations from the best solutions found (in $\frac{\%}{100}$), which is stated in the last columns *Ave. deviation from best* of Table 6. The following example illustrates how the deviation from the best solution found is calculated.

Example 1 Let 6.64 be the objective function value of the best solution found by CPLEX (S_C), and 24.34 the objective function value of the best solution found by MSH-4R (S_H). Since $S_C \leq \xi = 10$ is true, we consider S_C as an optimal solution. Accordingly, the deviation of S_C from the best solution found is 0, and the deviation of S_H from an optimal solution is $S_H - \xi = 14.34\ \frac{\%}{100}$.

Now, let $S_C = 16.34$ and $S_H = 30.71$. The best solution found is S_C and, consequently, the deviation of S_C from the best solution found is 0. Independent of whether we can prove the optimality of S_C or not, the deviation of S_H from the best solution found is $S_H - S_C = 14.37\ \frac{\%}{100}$.

For the problem instances with $\gamma = 70\%$, the solutions found by MSH-4R deviate, on average, $11.30\ \frac{\%}{100}$ (0.113%) from the optimal solution found by CPLEX. The minimum value is for the test set with 1 mining district ($7.42\ \frac{\%}{100}$), and the maximum value belongs to the test set with 3 mining districts ($14.60\ \frac{\%}{100}$). For $\gamma = 70\%$, MSH-4R could find for only 12 problem instances an optimal solution. Hereby, seven problem instances that are solved to optimality by MSH-4R have 1 mining district. On the other hand, for the test set with 3 mining districts, no optimal solution is found by MSH-4R. Since the average is calculated over the whole 20 problem instances for each test set, the differences regarding the average deviations from the best solutions found can be explained by the different number of optimal solutions found for each test set. The same trend exists at γ-levels of 75%, 80%, and 85%, where the solutions found by MSH-4R deviate, on average, 11.3, 11.12, and $10.45\ \frac{\%}{100}$ from the best solution found, respectively. We can conclude that the quality of the solutions found by MSH-4R is quite promising for the γ-levels of 70%, 75%, 80%, and 85%, where optimal solutions for 388 of 400 problem instances are known, which is a strong measure to evaluate the results. In addition, we see that for $\gamma = 90\%$, where CPLEX could find for only nine problem instances a feasible solution, MSH-4R found for all of the

100 problem instances at least one feasible solution. Therefore, it is reasonable to use MSH-4R to solve the problem instances at the γ-level of 90%.

In the next step, we compare the applied priority rules in MSH-4R (cf. Sect. 4) in Table 7. The first columns on the left-hand side of Table 7 show the number of problem instances for which a specific priority rule **exclusively** found the best solution. For example, in the first row of Table 7, the sum of the numbers under *Number of best-known solutions exclusively found* for rules 1 to 4 is 14. That means there are six problem instances in the test set for which the best solution could be found by means of at least two different priority rules. At the same line, we see that, e.g., priority rule 2 could exclusively find the best solutions for two problem instances, which means that the other priority rules could not find the best solution for those problem instances. We see in Table 7 that for 407 of the whole 500 problem instances, the best solutions were exclusively obtained using priority rule 3,

Table 7 Comparison among 4 priority rules (each row represents a test set including 20 problem instances)

γ [*in%*]	$\lvert\mathcal{R}\rvert$	Number of best-known solutions exclusively found				Average t^{bs} (t^{fs}) for the best (first) feasible solution [s]			
		Rule 1	Rule 2	Rule 3	Rule 4	Rule 1	Rule 2	Rule 3	Rule 4
70	1	2	0	12	0	186 (0.00)	196 (0.00)	153 (0.00)	186 (0.00)
	2	0	0	18	0	249 (0.00)	182 (0.00)	209 (0.00)	260 (0.00)
	3	2	0	18	0	229 (0.00)	151 (0.00)	210 (0.00)	223 (0.00)
	4	1	0	18	0	256 (0.00)	213 (0.00)	250 (0.00)	258 (0.00)
	5	1	0	19	0	231 (0.00)	260 (0.00)	240 (0.00)	267 (0.00)
75	1	2	0	12	0	185 (0.00)	196 (0.00)	153 (0.00)	207 (0.00)
	2	0	0	18	0	249 (0.00)	181 (0.00)	209 (0.00)	148 (0.00)
	3	2	0	18	0	229 (0.00)	151 (0.00)	210 (0.00)	224 (0.00)
	4	1	0	18	0	256 (0.00)	213 (0.00)	250 (0.00)	200 (0.00)
	5	1	0	19	0	231 (0.00)	260 (0.00)	240 (0.00)	223 (0.00)
80	1	2	0	12	0	186 (0.00)	196 (0.00)	153 (0.00)	228 (0.00)
	2	0	0	18	0	249 (0.00)	181 (0.00)	209 (0.00)	174 (0.00)
	3	2	0	18	0	229 (0.00)	151 (0.00)	209 (0.00)	187 (0.00)
	4	1	0	18	0	256 (0.00)	213 (0.00)	236 (0.00)	224 (0.00)
	5	1	0	19	0	231 (0.00)	259 (0.00)	239 (0.00)	194 (0.00)
85	1	2	0	12	0	186 (0.00)	195 (0.00)	153 (0.00)	238 (0.00)
	2	0	0	18	0	249 (0.00)	182 (0.00)	210 (0.00)	194 (0.00)
	3	2	0	18	0	229 (0.00)	151 (0.00)	210 (0.00)	232 (0.00)
	4	1	0	18	0	256 (0.00)	213 (0.00)	236 (0.00)	204 (0.00)
	5	1	0	19	0	231 (0.00)	260 (0.00)	240 (0.00)	254 (0.00)
90	1	3	1	11	0	144 (0.00)	180 (0.00)	190 (0.00)	234 (0.00)
	2	0	3	15	0	216 (0.00)	217 (0.00)	227 (0.00)	197 (0.00)
	3	0	4	16	0	212 (0.05)	245 (0.00)	261 (0.00)	257 (0.00)
	4	1	3	13	0	250 (0.00)	196 (0.00)	214 (3.00)	246 (0.05)
	5	0	7	12	1	201 (0.25)	220 (0.05)	214 (0.10)	230 (0.00)

which suggests priority rule 3 as the most potent rule. Remember that priority rule 3 enables the best possible improvement regarding the objective function in each step.

Let t^{bs} and t^{fs} be the times in seconds that are taken to find the best (*bs*) and the first feasible solution (*fs*), respectively. Those times are given on the right-hand side of Table 7. MSH-4R using priority rules 1, 2, 3, and 4 needed, on average, 225, 202, 213, and 220 s, respectively, to find the best feasible solutions. The first feasible solution could be quickly found by MSH-4R regardless of the applied priority rules (e.g., priority rules 2 and 4 find the first feasible solution by no longer than, on average, 0.05 s).

We saw that CPLEX performs for $\gamma = 90\%$ worse, in particular, with respect to the number of feasible solutions found (cf. Table 6). At the γ-level of 90%, where we have the most challenging problem instances, and where it is tough to find a feasible solution by a MILP-solver, MSH-4R can find at least one feasible solution for each problem instance. In the following, we introduce another solution approach to tackle the problem instances with $\gamma = 90\%$. The idea is to support CPLEX by starting with an initial feasible solution that is found by MSH-4R. We distinguish between the following kinds of initial solutions:

first solution (*fs*): MSH-4R using priority rules 2 and 4 is the fastest approach to find a feasible solution (cf. Table 7). That may be the case because both priority rules consider the amount of material of a block for determining the priority values to observe the lower limits of output. If we observe the quality of the solutions found, MSH-4R using priority rule 2 (priority rule 4) could exclusively find the best solution for 18 problem instances (one problem instance) with $\gamma = 90\%$. Remember that priority rule 2 considers both block's amount of material and the block's quality deviation from the quality target. By contrast, priority rule 4 focuses only on the amount of material, which justifies the higher quality of the solutions found using priority rule 2 than using priority rule 4. Thus, we solve the problem instances at the γ-level of 90% by MSH-4R using priority rule 2 and store the time t^{fs}. Then, we give the first feasible solution found as an initial solution to CPLEX and set the upper limit of time equal to $1800 - t^{fs}$ seconds. The corresponding solution approach is called **CPLEX with *fs***.

best solution (*bs*): Table 7 shows that MSH-4R using priority rule 3 performs best among all four priority rules. MSH-4R using priority rule 3 could exclusively find for 67 problem instances with $\gamma = 90\%$ the best solution within, on average, 221.2 s. Hence, we solve the problem instances at the γ-level of 90% by MSH-4R using priority rule 3, set the upper limit of the solution time equal to 250 s, and store the best feasible solution found. Then, we solve the problem instances by CPLEX using the best feasible solutions found by MSH-4R as an initial solution with a time limit of 1550 s. The corresponding solution approach is called **CPLEX with *bs***.

Table 8 depicts that using an initial solution leads to a much better performance of CPLEX at the γ-level of 90%. Without using an initial solution, CPLEX could find for only nine of 100 problem instances a feasible solution (thereof seven optimal solutions) within, on average, 1688 s (cf. Table 6, last 5 rows). If we apply CPLEX

Table 8 Comparison of the solutions found by CPLEX with and without initial solutions (each row represents a test set including 20 problem instances)

γ [*in%*]	$\lvert\mathcal{R}\rvert$	CPLEX with *fs*			CPLEX with *bs*		
		# Opt.	# Feas.	Solution time [s]	# Opt.	#Feas.	Solution time [s]
90	1	12	8	922	14	6	789
	2	12	8	1115	11	9	1298
	3	5	15	1626	7	13	1567
	4	8	12	1651	8	12	1575
	5	5	15	1757	5	15	1786

with *fs*, 42 problem instances can be solved to optimality within, on average, 1414.2 s. Typically, the larger the problem instances, the harder it is to solve. For the problem instances with 1 mining district, CPLEX with *fs* could find an optimal solution for 12 of 20 problem instances. For the problem instances with 5 mining districts, only five of 20 problem instances can be solved to optimality by CPLEX with *fs*. CPLEX with *bs* performs even better, where 45 problem instances can be optimally solved within, on average, 1403 s (some more problem instances within some less time in comparison to CPLEX with *fs*). The same trend regarding the size of the instances is recognizable. We can see that 14 of 20 problem instances with 1 mining district and five of 20 problem instances with 5 mining districts can be solved to optimality by CPLEX with *bs*.

Finally, we compare the results achieved by MSH-4R with the combination of MSH-4R and CPLEX. For this purpose, the average of the absolute deviations from the best solution known is depicted in Table 9 for each test set with $\gamma = 90\%$. Let S_{H_i} be the objective function value of the best solution found by MSH-4R for problem instance i. Moreover, let $S^{fs}_{C_i}$ and $S^{bs}_{C_i}$ denote the objective function values of the solutions found by CPLEX with *fs* and *bs* for problem instance i, respectively. The numbers in parentheses are the average of $(S_{H_i} - S^{fs}_{C_i})/(S_{H_i})$ as well as $(S_{H_i} - S^{bs}_{C_i})/(S_{H_i})$ over the problem instances in every corresponding test set. We see that the solutions found by CPLEX with an initial solution are much better than the solutions found by MSH-4R. In particular, if we first solve a problem instance heuristically using

Table 9 Comparison of the solutions found by MSH-4R with the results achieved by CPLEX using initial solutions (each row represents a test set including 20 problem instances)

γ [*in%*]	$\lvert\mathcal{R}\rvert$	Average deviation from the best solution found [$\frac{\%}{100}$]		
		MSH-4R	CPLEX with *fs* (improvement)	CPLEX with *bs* (improvement)
90	1	3.11	0.98 (68.5%)	0.03 (99.0%)
	2	5.11	1.96 (61.6%)	0.27 (94.7%)
	3	8.88	2.43 (72.6%)	0.72 (91.9%)
	4	8.11	1.93 (76.2%)	0.57 (93.0%)
	5	7.70	1.22 (76.2%)	1.21 (84.3%)

priority rule 3 and then give the best feasible solution found as an initial solution to CPLEX (CPLEX with *bs*), the solutions are improved by, on average, 92.5%.

6 Conclusion

In this paper, we introduced a mixed-integer linear program and a heuristic approach to solve a block selection and sequencing problem occurring in underground potash mines. In the problem under consideration, we minimized the deviations of the output quality value from a prescribed quality target value. The deviations were calculated for certain sub-intervals within a given planning horizon.

To evaluate the solution approaches, we randomly generated 100 problem instances (5 test sets of 20 problem instances each) based on realistic data. We solved the problem instances on 5 different levels in terms of the lower limit of the output using the proposed mixed-integer linear program and the suggested constructive heuristic. We can conclude that if the MILP-solver starts with an initial solution, an optimal solution for a problem instance can be found more quickly. We can also say, if we solve a problem instance within a reasonable amount of time heuristically using introduced priority rule 3 and then give the best feasible solution found as an input to the MILP-solver, we obtain quite promising results. Hence, practice-relevant problems in potash mines can be solved with an acceptable quality within an adequate time frame. Consequently, the results achieved could support the decisions of underground mining operators to provide the aboveground processing plants with homogenous output by a systematic comparison between the actual and target performance.

Further research will apply metaheuristics to improve the results achieved by the proposed constructive heuristic. Uncertainty regarding the processing times needed to excavate the blocks must also be investigated. On the other hand, the material excavated in a work shift can be stored in a bunker during a specific time interval. That can lead to another value of the output quality in comparison to having all the material conveyed directly to the surface. Ongoing research can distinguish those cases to make the generated solutions more realistic.

Funding Open Access funding enabled and organized by Projekt DEAL.

References

Askari-Nasab H, Pourrahimian Y, Ben-Awuah E, Kalantari S (2011) Mixed integer linear programming formulations for open pit production scheduling. J Min Sci 47(3):338–359

Azzamouri A, Fénies P, Fontane F, Giard V (2018) Scheduling of open-pit phosphate mine extraction. Int J Prod Res 56(23):7122–7141

Bley A, Boland N, Fricke C, Froyland G (2010) A strengthened formulation and cutting planes for the open pit mine production scheduling problem. Comput Oper Res 37(9):1641–1647

Blom M, Pearce AR, Stuckey PJ (2019) Short-term planning for open pit mines: a review. Int J Min Reclam Environ 33(5):318–339

Blom ML, Pearce AR, Stuckey PJ (2016) A decomposition-based algorithm for the scheduling of open-pit networks over multiple time periods. Manag Sci 62(10):3059–3084

Campeau L-P, Gamache M (2020) Short-term planning optimization model for underground mines. Comput Oper Res. https://doi.org/10.1016/j.cor.2019.02.005

Chesworth W (2008) Encyclopedia of soil science. Springer, Dordrecht

Chicoisne R, Espinoza D, Goycoolea M, Moreno E, Rubio E (2012) A new algorithm for the open-pit mine production scheduling problem. Oper Res 60(3):517–528

Clausen E (2013) Konzept für einen integrierten Produktionssteuerungsansatz bei Anwendung eines Örterbaus. PhD diss., Clausthal University of Technology

Elsayed S, Sarker R, Essam D, Coello CC (2020) Evolutionary approach for large-scale mine scheduling. Inf Sci 523:77–90

Espinoza D, Goycoolea M, Moreno E, Newman A (2013) Minelib: a library of open pit mining problems. Ann Oper Res 206(1):93–114

Garey MR, Johnson MD (1979) Computers and intractability: a guide to the theory of NP-completeness. Freeman, New York

Hamrin H (2001) Underground mining methods and applications. In: Hustrulid WA, Bullock RL (eds) Underground mining methods: engineering fundamentals and international case studies. SME, Littleton, pp 3–14

Heinz A, von der Osten R (1982) ABC Kali und Steinsalz. Deutscher Verlag für Grundstoffindustrie, Leipzig

Jélvez E, Morales N, Nancel-Penard P, Peypouquet J, Reyes P (2016) Aggregation heuristic for the open-pit block scheduling problem. Eur J Oper Res 249(3):1169–1177

Jélvez E, Morales N, Nancel-Penard P, Cornillier F (2020) A new hybrid heuristic algorithm for the precedence constrained production scheduling problem: a mining application. Omega 94:102046

King B, Goycoolea M, Newman A (2017) Optimizing the open pit-to-underground mining transition. Eur J Oper Res 257(1):297–309

Kozan E, Liu SQ (2011) Operations research for mining: a classification and literature review. ASOR Bull 30(1):2–23

K+S AG (2013) A day in the mine. http://www.k-plus-s.com/en/ks-zum-anfassen/tag-in-der-grube.html. Accessed 10 Sep 2020

Lambert WB, Newman AM (2014) Tailored Lagrangian relaxation for the open pit block sequencing problem. Ann Oper Res 222(1):419–438

Lamghari A, Dimitrakopoulos R (2016) Network-flow based algorithms for scheduling production in multi-processor open-pit mines accounting for metal uncertainty. Eur J Oper Res 250(1):273–290

Leite LG, Marcelo J, Arruda EF, Bahiense L, Marujo LG (2020) Modeling the integrated mine-to-client supply chain: a survey. Int J Min Reclam Environ 34(4):247–293

Liu SQ, Kozan E (2016) New graph-based algorithms to efficiently solve large scale open pit mining optimisation problems. Expert Syst Appl 43:59–65

Mai NL, Topal E, Erten O, Sommerville B (2019) A new risk-based optimisation method for the iron ore production scheduling using stochastic integer programming. Resour Policy 62:571–579

Martinez MA, Newman AM (2011) A solution approach for optimizing long-and short-term production scheduling at LKAB's Kiruna mine. Eur J Oper Res 211(1):184–197

Michalewicz Z, Fogel D (2004) How to solve it: modern heuristics. Springer, Berlin

Montiel L, Dimitrakopoulos R (2015) Optimizing mining complexes with multiple processing and transportation alternatives: an uncertainty-based approach. Eur J Oper Res 247(1):166–178

Mousavi A, Sellers E (2019) Optimisation of production planning for an innovative hybrid underground mining method. Resour Policy 62:184–192

Mousavi A, Kozan E, Liu SQ (2016) Comparative analysis of three metaheuristics for short-term open pit block sequencing. J Heuristics 22(3):301–329
Musingwini C (2016) Optimization in underground mine planning-developments and opportunities. J South Afr Inst Min Metall 116(9):809–820
Nehring M, Topal E, Little J (2010) A new mathematical programming model for production schedule optimization in underground mining operations. J South Afr Inst Min Metall 110(8):437–446
Nehring M, Topal E, Kizil M, Knights P (2012) Integrated short-and medium-term underground mine production scheduling. J South Afr Inst Min Metall 112(5):365–378
Newman AM, Rubio E, Caro R, Weintraub A, Eurek K (2010) A review of operations research in mine planning. Interfaces 40(3):222–245
O'Sullivan D, Newman A (2015) Optimization-based heuristics for underground mine scheduling. Eur J Oper Res 241(1):248–259
O'Sullivan D, Brickey A, Newman A (2015) Is openpit production scheduling easier than its underground counterpart? Min Eng 67(4):68–73
Ramazan S, Dimitrakopoulos R (2013) Production scheduling with uncertain supply: a new solution to the open pit mining problem. Optim Eng 14(2):361–380
Reus L, Belbèze M, Feddersen H, Rubio E (2018) Extraction planning under capacity uncertainty at the Chuquicamata underground mine. Interfaces 48(6):543–555
Rivera Letelier O, Espinoza D, Goycoolea M, Moreno E, Muñoz G (2020) Production scheduling for strategic open pit mine planning: a mixed-integer programming approach. Oper Res. 4:5. https://doi.org/10.1287/opre.2019.1965
Samavati M, Essam D, Nehring M, Sarker R (2017) A local branching heuristic for the open pit mine production scheduling problem. Eur J Oper Res 257(1):261–271
Samavati M, Essam D, Nehring M, Sarker R (2018) A new methodology for the open-pit mine production scheduling problem. Omega 81:169–182
Schulze M, Zimmermann J (2017) Staff and machine shift scheduling in a German potash mine. J Sched 20(6):635–656
Schulze M, Rieck J, Seifi C, Zimmermann J (2016) Machine scheduling in underground mining: an application in the potash industry. OR Spectr 38(2):365–403
Seifi C, Schulze M, Zimmermann J (2019) A two-stage solution approach for a shift scheduling problem with a simultaneous assignment of machines and workers. In: Mining goes digital: proceedings of the 39th international symposium application of computers and operations research in the mineral industry (APCOM (2019) June 2019, Wroclaw, Poland. CRC Press, pp 377–385
Seifi C, Schulze M, Zimmermann J (2020) A new mathematical formulation for a potash-mine shift scheduling problem with a simultaneous assignment of machines and workers. Eur J Oper Res. https://doi.org/10.1016/j.ejor.2020.10.007
Smith ML, Wicks SJ (2014) Medium-term production scheduling of the Lumwana Mining Complex. Interfaces 44(2):176–194
USGS (2011) Metals and minerals: United States geological survey minerals yearbook. United States Government Printing Office, Washington
Vossen TWM, Kevin Wood R, Newman AM (2016) Hierarchical Benders decomposition for open-pit mine block sequencing. Oper Res 64(4):771–793