Les Cahiers du GERAD ISSN: 0711–2440

Short-term underground mine planning with uncertain activity durations using constraint programming

Y. Aalian, G. Pesant, M. Gamache

G–2022–44

October 2022

La publication de ces rapports de recherche est rendue possible grâce au soutien de HEC Montréal, Polytechnique Montréal, Université McGill, Université du Québec à Montréal, ainsi que du Fonds de recherche du Québec – Nature et technologies.

Dépôt légal – Bibliothèque et Archives nationales du Québec, 2022
– Bibliothèque et Archives Canada, 2022

The publication of these research reports is made possible thanks to the support of HEC Montréal, Polytechnique Montréal, McGill University, Université du Québec à Montréal, as well as the Fonds de recherche du Québec – Nature et technologies.

Legal deposit – Bibliothèque et Archives nationales du Québec, 2022
– Library and Archives Canada, 2022

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

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

Short-term underground mine planning with uncertain activity durations using constraint programming

Younes Aalian [a, b]

Michel Gamache [a, b]

Gilles Pesant [c]

[a] *GERAD, Montréal (Qc), Canada, H3T 1J4*

[b] *Département de mathématiques et de génie industriel, Polytechnique Montréal, Montréal (Qc), Canada, H3T 1J4*

[c] *Département de génie informatique et génie logiciel, Polytechnique Montréal, Montréal (Qc), Canada, H3T 1J4*

younes.aalian@polymtl.ca
michel.gamache@polymtl.ca
gilles.pesant@polymtl.ca

October 2022
Les Cahiers du GERAD
G–2022–44

Abstract : The short-term scheduling of activities in underground mines is an important step in mining operations. This procedure is a challenging optimization problem since it deals with many resources and activities conducted in a confined working space. Moreover, underground mining operations deal with multiple uncertainties such as the variation of activity durations. In this paper, a Constraint Programming (CP) model is proposed for short-term planning in underground mines. The developed model takes into account the technical requirements of underground operations to build realistic mine schedules. Furthermore, two different approaches are proposed based on the CP model for robust short-term underground mine scheduling. The first approach aims to create a robust schedule using multiple scenarios of the problem. This stochastic CP model enables to find a set of ordered robust sequences of activities performed by each available disjunctive resource over several scenarios. In the second approach, a confidence constraint is introduced in the CP model to specify the probability that the schedule generated won't underestimate the duration of activities. The model allows the mine planner to control the risk level with which an optimized solution should be produced such that it can be implemented given the actual activity durations. The presented approaches are tested on real data sets of an underground gold mine in Canada. An evaluation model is designed to evaluate the robust performance of the proposed models. The experiments demonstrate that both scenario-based and confidence-constraint approaches outperform the deterministic model by generating schedules that are more robust to uncertainties in underground operations.

Keywords : Mine planning, constraint programming, short-term planning, underground mine, scheduling

1 Introduction

Short-term underground mine scheduling assigns mining equipment to activities and determines the start time of each activity as well as the activities' ordering. Available activities are scheduled for each shift on a planning horizon of one to two weeks. Underground operations are performed in a dynamic and complex environment dealing with many uncertainties such as additional delays in activities and machine breakdowns. Therefore, short-term scheduling is performed on a shift-based time frame where the schedules can be revised according to the latest changes (Åstrand et al. (2020)).

In an underground mine, there are two main groups of activities: development and production. Development activities are performed in waste rocks with no financial value to access economically valuable deposits, while production activities are conducted in valuable rocks to extract ore material in places called stopes (minable shapes). Mining activities are performed thanks to a cyclic process at a site. A site is a workplace where mining activities are carried out.

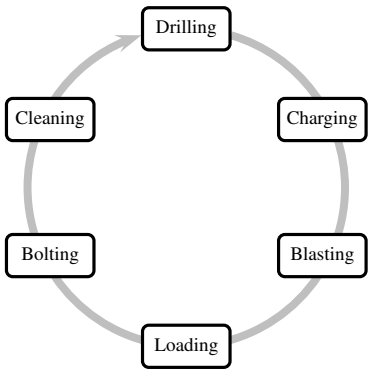


Figure 1: An example of a cycle for the development activities in underground mining

A cycle includes a series of specific activities at each site. For example, for a development site, series of activities include drilling, charging, blasting, loading, bolting, and cleaning and are conducted in a sequence-dependent order as presented in Figure 1. The description of activities and the equipment type used to perform each activity is shown in Table 1. As a consequence, short-term scheduling assigns equipment to associated activities in the cycle and determines the start and end times of those activities (Åstrand et al. (2018a)).

Table 1: Required equipment of each activity type

Activity type	Machine	Description
Drilling	Drilling rigs	Drilling blast holes in the rock face
Charging	Anfo loader	Charging drilled holes with explosives
Loading	Scooptram	Removing broken rocks after blasting
Bolting	Bolter	Stabilizing drifts by installing bolts into the rock mass
Cleaning	Scooptram	Removing small rocks from the site (the gallery)

Short-term planning in underground mines is a complicated procedure as it deals with many resources and activities of several types that are processed in a confined working space. Additionally, various operational constraints should be considered in this process. Furthermore, available machines have different capacities and speeds. Each machine can process a particular activity type in a working site at a time. Some equipment can only perform one activity type in the cycle while others can be used for several activities. Moreover, machines must travel between different site locations. Due to precedence constraints between activities, the next activity can begin only when the previous activity is completed. Based on safety regulations in the underground mine production, blasts are permitted

between two working shifts while other activities are paused, personnel is evacuated from the mine, and equipment is removed from the working site (Wang et al. (2020)).

Underground mining deals with numerous uncertainties. As long as the rock mass is not accessed physically, its properties remain uncertain. Therefore, the different characteristics of extracting rocks cause variation in the duration of mining activities. Furthermore, underground operations are performed in a confined and limited working space with dusty air. This harsh condition of the underground mine environment leads to frequent breakdowns of operating machines (Åstrand et al. (2018a)). This uncertainty in the scheduling process is a significant challenge for mine planners (Manríquez et al. (2020)). Conventional approaches for mine production scheduling do not consider uncertainty in the scheduling decisions which often results in infeasible solutions. Therefore, there is a need to develop an efficient tool for short-term underground mine planning that considers both the operational constraints and the uncertain environment of underground operations to generate more reliable and robust schedules.

Previous studies have proven the value and efficiency of Constraint Programming (CP) in solving scheduling problems. CP is an exact method for modelling and solving combinatorial optimization problems. This approach has been used in diverse sectors such as planning, scheduling, transportation, and automated systems scheduling problems (Laborie et al. (2018)). CP benefits from a high computational performance by applying constraint propagation that reduces the domains of variables and using powerful search strategies to reduce the search space (Pesant (2014)). Furthermore, CP offers more flexible and intuitive formulations by employing its rich set of variable types, functions, and global constraints (Kanet et al. (2004)). Thus, CP allows more compact models with fewer decision variables and constraints compared to other mathematical programming approaches such as MIP. These strengths of CP make it suitable for modeling and solving large-scale scheduling problems. In the underground mining context, by employing the rich dictionary of CP functions, the mining-specific constraints in the short-term scheduling problem can be modeled more easily than with other optimization techniques.

In the next section, a review of related works on short-term underground mine planning is provided. Section 3 presents three Constraint Programming (CP) models for short-term underground mine planning. Section 4 describes the implementation of the proposed models on real data sets followed by the computational results and discussion. Finally, Section 5 concludes the paper.

2 Literature review

In recent years, several models and solution approaches have been proposed in the literature to address the problem of short-term planning in underground mines. Most of these approaches use mathematical programming, but constraint programming papers have been published very recently. Indeed, Laborie (2018) demonstrated that CP techniques show better results in comparison with mixed-integer programming (MIP) approaches especially for scheduling problems.

2.1 Mathematical programming approaches for short-term underground mine scheduling

Nehring et al. (2010) developed a MIP model for short-term scheduling and loader-truck allocation in sublevel stoping mines that enables reassignment of equipment based on the changes in underground mining operations. The model was tested on a conceptual underground copper mine and resulted in satisfactory tonnage deviations from predefined amounts for each shift over the entire planning period. O'Sullivan and Newman (2015) proposed an integer programming (IP) model to determine the short-term scheduling of activities in an underground lead and zinc mine in Ireland to maximize the discounted quantity of produced metal. Two general types of constraints have been considered in the model: resource and precedence constraints. The authors implemented exact and heuristic solutions

to reduce the number of variables in the problem. Moreover, they created an optimization-based decomposition heuristic to address complicated problem instances and create feasible schedules in less computation time. Song et al. (2015) studied the problem of scheduling mobile equipment in underground mines. They proposed a decision support instrument to optimize the scheduling of activities in the mine production process. The developed tool is applied to a real mine data set in Finland which considerably decreases the makespan compared to manual scheduling methods and improves the operational performance. However, the proposed method does not consider the uncertainty associated with unexpected activities in underground mining.

Schulze et al. (2016) scheduled the mobile machines while determining the sequence of block excavation in an underground potash mine. A mixed-integer linear programming (MILP) model is developed with the objective of minimizing the makespan and solved on a small-scale mine data set using CPLEX. Moreover, a priority rule-based construction procedure is used to develop a basic and an advanced multi-start algorithm. The advanced multi-start algorithm integrates waiting times of jobs to satisfy safety constraints. Schulze and Zimmermann (2017) proposed a construction solution for short-term production scheduling in underground mining in order to assign staff and machines to mining activities. Several operational constraints have been considered in this approach to address the problem. The objective is to minimize deviations from a targeted production in a potash mine. The developed model was evaluated on both randomly created and real-world case studies. Results demonstrate that the proposed multi-start algorithm remarkably outperforms the manual scheduling method. Seifi et al. (2019) developed a two-stage solution approach for scheduling machines and staff in a working shift of an underground potash mine located in Germany. In the first step, the relaxation of the MIP model is solved. Next, the solutions obtained by the relaxation model are modified to achieve feasible schedules using a heuristic algorithm. The experiments on realistic instances show that the proposed approach outperforms the heuristic procedure developed by Schulze and Zimmermann (2017).

Campeau and Gamache (2020) developed a MIP model for optimizing short-term planning in underground mines. The objective function is to maximize the discounted extraction of material while maintaining a minimum ore production rate to keep the mill active. Furthermore, operational limitations and resource constraints are considered in the model to generate feasible schedules. The model is applied to a gold mine data set with 385 workplaces and produced an optimal short-term schedule. Campeau et al. (2022) proposed a new MIP model to integrate and solve short- and medium-term scheduling problems for underground mining. In this approach, continuous variables for time discretization are incorporated into the model to address both short- and medium-term planning and generate realistic solutions. The model is solved on a data set of a Canadian gold mine and demonstrated promising results. A genetic algorithm (GA) is employed by Wang et al. (2020) to optimize the scheduling of underground mining equipment. The problem is considered an NP-hard problem. They proposed a non-linear programming (NLP) model with a large number of decision variables related to several mining sites and equipment types with their specific limitations. The objective is defined as minimizing the total completion time and working intervals.

2.2 Constraint programming approaches for short-term underground mine scheduling

A Constraint Programming (CP) model is proposed by Åstrand et al. (2018b) to schedule mobile fleet in underground mining. The developed model was tested on an underground mine data set and resulted in promising schedules. Åstrand et al. (2020) extended the CP model of Astrand et al. (2018) by considering the travel times of mobile machines in an underground cut-and-fill mine. Furthermore, the authors developed a new revised CP model in which the blast times are compressed, and the generated solutions are post-processed to acquire schedules for the main problem. A domain-specific neighborhood definition is also developed and employed in the Large Neighborhood Search (LNS) algorithm to increase the quality of schedules and obtain them in less computation time. The proposed

model and solution approach are evaluated on multiple instances created from underground mine data sets located in Sweden. Result demonstrates the efficiency of the proposed method to improve the initial feasible solution and obtain high-quality schedules. A CP model is proposed by Campeau and Gamache (2022) to address short- and medium-term scheduling in underground mines. The proposed model is tested on five different data sets obtained from an underground gold mine in Canada for a planning horizon up to a year to consider long-term production planning goals. Results indicate that the CP model outperforms the corresponding MIP model in both computational and application aspects. However, the proposed CP model is deterministic and does not consider the stochastic aspect of mining operations.

As can be seen from the literature, several models and solution approaches have been presented to solve the short-term underground mine planning problem. However, the developed methods do not consider the uncertainties associated with the variance in the activity durations, resulting in impractical short-term schedules. Therefore, this makes it necessary to develop new models that consider mining-specific requirements and the uncertainty of activity durations to produce more realistic and robust short-term schedules for underground mines.

3 Three models

This section presents three Constraint Programming (CP) models for short-term planning in underground mines. The models are specifically designed for development activities in an underground mine using the long-hole stopping method. The first model allocates machines to related activities and determines the sequences of activities on each machine as well as activities start time, taking into account operational constraints of underground mining, but does not yet consider uncertainty. The next two models will present two different approaches to deal with uncertainty.

3.1 Deterministic CP model

The model is developed using CP Optimizer (CPO) from IBM Optimization Studio. CPO is a CP-based tool designed to model and solve constraint-based scheduling problems. It is developed based on the model and run procedure that makes it simpler to understand and use. The CP Optimizer benefits from an efficient and complete automatic search algorithm. The automatic search employs Large Neighborhood Search (LNS) combined with Failure-Directed Search (FDS) that improves the performance and solution speed of the CP Optimizer automatic search. Therefore, CP Optimizer has the ability to quickly find good feasible solutions. Based on the literature, CP Optimizer has been successfully applied to address diverse scheduling problems in the industry (Laborie et al. (2018)).

In the proposed CPO model, three types of variables are employed: interval variables, optional interval variables, and sequence variables. The interval variable denotes activities in the scheduling model. An interval variable a has a given starting time s and an end time e with integer values such that $a \in \Big\{[s,e) \mid s,e \in \mathbf{Z}, e \geq s\Big\}$. The optional interval variable b represents an activity that may be unperformed and absent from the solution of the scheduling problem: $b \in \Big\{\emptyset\Big\} \cup \Big\{[s,e) \mid s,e \in \mathbf{Z}, e \geq s\Big\}$. The sequence variable defines a set of interval variables. This variable type is used to impose the no overlap constraint in the scheduling model. This constraint restricts activities in the sequence not to overlap in time. Furthermore, several types of functions and constraints used in the CP Optimizer scheduling model are defined as follows (Laborie et al. (2018)):

Presence of: This Boolean function is applied in logical constraints to ensure that a given interval variable is present in the solution of the problem.

End of: This function is an integer expression that gives access to the end of the interval variable if it is present. If the interval variable is absent, then the expression value is equal to zero.

Step functions: These elementary functions are used to model a known function of time, for instance, the amount of resources employed to perform an activity at a particular time (period). A step function returns a non-negative constant value between the start and end point of an interval variable and a zero value outside of this interval. The non-negative constant value is known as the height of the step function.

Alternative: The constraint alternative $(i, \{j_1, .., j_n\})$ makes sure that if the interval variable i is present, then exactly one of the optional interval variables in $\{j_1, .., j_n\}$ is chosen with start and end values identical to the ones of the interval variable i.

No overlap: This constraint makes sure that the set of interval variables defined by the sequence variable do not overlap each other. In the set of interval variables $\{j_1, .., j_n\}$ of the sequence s, the constraint $noOverlap(s, D)$ ensures that if j_1 takes place before j_2 in the sequence value, then j_1 must end before the start of j_2 while a minimum non-negative distance $d_{j_1 j_2}$ defined by a transition distance matrix (D) is maintained between these two variables.

End before start: The constraint $endBeforeStart(i, j, d_{ij})$ makes sure that if both interval variables i and j are present then the interval variable i ends before the start of interval variable j with an optional minimum delay of d_{ij} time units between these two variables.

Forbid extent: In constraint $forbidExtent(i, F)$, if the interval variable i is present, then it cannot overlap with the forbidden region where the value of the step function (F) is equal to zero. Therefore, the interval variable must either end before the forbidden region or start after that.

Lists of sets, parameters, and variables of the deterministic CP model are defined in Tables 2 and 3.

Table 2: Sets and parameters of the deterministic CP model

Sets	**Description**
J	Set of activities
M	Set of all available equipment
M_j	Set of eligible machines to perform activity j
$Succ_j$	Set of successor activities for activity j
B	Set of blast activities
Parameters	**Description**
p_j	Processing time of activity j
D	Matrix of transition time between sites where the value of its element is equal to 0 for the same site and greater than 0 otherwise
$Blast\_calendar$	The time periods where only blasting activities are allowed (all activities except the blasting are forbidden to be performed during period t where $F(t) = 0$)

Table 3: Decision variables of the deterministic CP model

Variables	**Description**
Y_j	The interval variable for activity j with size p_j
X_{jm}	The optional interval variable with size p_j to perform activity j using machine m
S_m	Sequence variable for machine m (S_m *in all* X_{jm} $\forall\, j \in J, m \in M$)

Objective function

$$Minimize\,[Max\,(endOf(Y_j))] \tag{1}$$

Constraints

$$presenceOf\,(X_{jm}) = 0 \qquad \forall j \in J, m \in M \backslash M_j \tag{2}$$

$$alternative\,(Y_j, \{X_{jm} \mid m \in M\}) \qquad \forall j \in J \tag{3}$$

$$noOverlap\,(S_m, D) \qquad \forall m \in M \tag{4}$$

$$endBeforeStart\,(Y_j, Y_i) \qquad \forall j \in J,\ i \in Succ_j \tag{5}$$

$$forbidExtent\,(Y_j, Blast\_calendar) \qquad \forall j \in J \backslash B \tag{6}$$

The objective of the model is to minimize the makespan of the project. Constraint (2) makes sure that each activity is allocated to the eligible machine type. Constraint (3) ensures to select only one optional variable for a given interval variable where both variables start and end simultaneously. Constraint (4) restricts the equipment not to overlap in time. In other words, each machine can be assigned to only one activity at a time. Constraint (5) considers the precedence relations between activities performed in a particular site. Activities have at most one predecessor, and few activities do not have any predecessor. Constraint (6) makes sure that blasts are performed between shifts. This constraint prohibits all activities except blasting to overlap with the forbidden region defined by the blast calendar.

3.2 Scenario-based approach

A solution of the deterministic CP model is subject to extensive changes due to multiple uncertainties such as the variation of activity durations and machine breakdowns. Therefore, a scenario-based approach is presented to generate robust sequences of activities performed on available machines over multiple scenarios. A scenario is a representation (realization) of activity durations based on statistical distributions. This approach aims to reduce the negative impact of potential delays and breakdowns on the makespan.

Our proposed scenario-based model is adapted from the model of **?**. In the improved model, the site index has been removed from the interval variables, which reduces the number of variables and constraints (reduces the problem size). Moreover, the model achieves higher quality solutions (more robust schedules) using more scenarios. In this method, the duration of activities are random variables with known distributions. Activities durations are sampled through multiple scenarios (simulations) using a stochastic sampling process. The presented model solves multiple scenarios of the original deterministic CP model. An additional constraint called "Same sequence" (Constraint (13)) is used in the CP model to ensure that activities are scheduled on available machines in the same order across all scenarios. The objective is to minimize the average makespan over all generated scenarios. As presented in Table 4, the stochastic CP model includes a new set and a new parameter in addition to the previous ones. An index $k \in K$ is added to the decision variables in order to take into account different scenarios, as defined in Table 5.

Table 4: Sets and parameters of the stochastic CP model

Sets and parameters	Description
K	Set of all scenarios
p_{jk}	Processing time of activity j in scenario k

Table 5: Decision variables of the stochastic CP model

Variables	Description
Y_{jk}	The interval variable for activity j with size p_{jk}
X_{jmk}	The optional interval variable with size p_{jk} to perform activity j using machine m in scenario k
S_{mk}	Sequence variable for machine m in scenario k (S_{mk} *in all* X_{jmk} $\forall j \in J, m \in M, k \in K$)

Objective function

$$Minimize \left[\sum_{k=1}^{|K|} Max\left(endOf(Y_{jk})\right)/|K| \right] \tag{7}$$

Constraints

$$presenceOf\left(X_{jmk}\right) = 0 \qquad \forall j \in J, m \in M \backslash M_j, k \in K \tag{8}$$

$$alternative\left(Y_{jk}, \{X_{jmk} \mid m \in M\}\right) \qquad \forall j \in J, k \in K \tag{9}$$

$$noOverlap\,(S_{mk}, D) \qquad \forall m \in M, k \in K \tag{10}$$
$$endBeforeStart\,(Y_{jk}, Y_{ik}) \qquad \forall j \in J, i \in Succ_j, k \in K \tag{11}$$
$$forbidExtent\,(Y_{jk}, Blast\_calendar) \qquad \forall j \in J \backslash B, k \in K \tag{12}$$
$$SameSequence(S_{mk}, S_{mk'}) \qquad \forall m \in M, \forall (k, k' \mid k \geq k') \in |K| \times |K| \tag{13}$$
$$\tag{14}$$

3.3 Confidence-constraint approach

This approach introduces new chance constraints called confidence into the deterministic CP model. The confidence-constraint model allows for risk/reward optimization by taking the appropriate amount of risk at the right time. The method ensures that the constructed (optimized) schedule tolerates uncertainty by meeting a risk threshold. Mercier-Aubin et al. (2020) presented a confidence constraint as follows (Constraint (15)):

$$CONFIDENCE([X_1, ..., X_n], [D_1, .., D_n], \gamma) \Longleftrightarrow \prod_{i=1}^{n} cdf_i(X_i) \geq \gamma \tag{15}$$

The described confidence constraint consists of a vector of (integer) decision variables $[X_1, .., X_n]$, a vector of statistical distributions $[D_1, .., D_n]$, and a confidence threshold γ. D_i denotes the distribution of the historical observed values of X_i. From each distribution D_i, we define a cumulative distribution function cdf_i. A set of independent random variables $W_1, .., W_n$ is defined. These variables follow the described distributions. The confidence constraint ensures that, with probability at least γ, all variables X_i are greater than or equal to their associated random variable W_i. The logarithmic form (log transformation) of constraint (15) is presented as Equation (16).

$$\sum_{i=1}^{n} ln(P[W_i \leq X_i]) \geq ln(\gamma) \tag{16}$$

In this paper, two versions of the confidence-constraint are introduced into the deterministic CP model to ensure sufficiently large values for activity durations with a given threshold probability (γ) without systematically overestimating the duration of those activities. The confidence threshold value is defined based on the risk tolerance in underground operations. The CP model with the confidence-constraint (CP chance model) aims to produce sequences of activities on machines that are (likely) robust to the activity durations uncertainty with a certain confidence threshold. The developed model can produce several (reliable) robust schedules using different confidence thresholds, enabling the mine planner to select the suitable threshold value according to the risk tolerance in underground operations. The two confidence constraints are presented as follows:

3.3.1 Confidence constraint 1

This chance constraint ensures that all available activities have delay values (Z_j) such that the combined cumulative distribution function ($CDFs$) of those delays is no smaller than the threshold probability (γ). In other words, the constraint ensures sufficiently long durations for delay of activities in accordance with a given threshold probability γ. In this constraint, the cdf_j is implemented as a table of integers indexed by the decision variable (Z_j). The developed constraint is defined in Table 6 and Equation (17).

$$\sum_{j=1}^{J} ln(cdf_j(Z_j)) \geq ln(\gamma) \tag{17}$$

Table 6: Decision variable and parameter of the CP chance model

Variables	Description
Z_j	Decision variable for delay duration of activity j
γ	Confidence threshold, a probability between 0 and 1

3.3.2 Confidence constraint 2

In the previous chance constraint, the threshold value is applied to all available activities, resulting in small decreases in activity durations once the threshold value is reduced. Therefore, a second version of the confidence constraint is proposed that is less restrictive (on the duration of activities) than the first one. This constraint considers the confidence threshold for each machine (performing related activities) rather than all available activities. The presented constraint makes sure that a set of activities performed by a particular machine m take sufficient delays according to the confidence threshold (γ_m). γ_m denotes the confidence threshold for machine m that has a probability between 0 and 1. The proposed confidence constraint is defined in Equation (18).

$$\sum_{j=1}^{J} ln(cdf_j(Z_j)) \cdot presenceOf(X_{jm}) \geq ln(\gamma_m) \quad \forall\, m \in M \tag{18}$$

In the developed confidence constraints, the delay of an activity (Z_j) is a decision variable that is set by the CP model. The total duration of an activity (P_j) is the sum of the initial (minimum) duration (d_j) that is a given parameter and the delay time (Z_j) determined using the CP model with the confidence constraint (Equation (19)). The total duration of an activity is represented in Figure 2.

$$P_j = d_j + Z_j \quad \forall\, j \in J \tag{19}$$

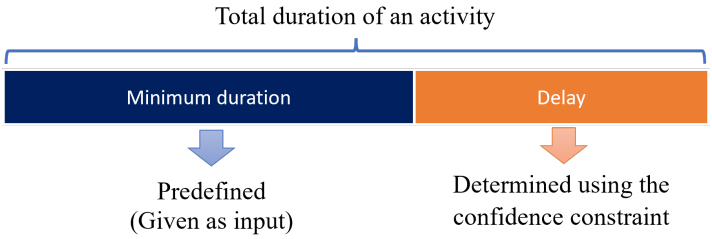


Figure 2: Total duration of an activity in the confidence-constraint approach

To further describe the proposed confidence constraints in the CP model, a simple instance is defined for the confidence-constraint model with ten activities of the same type performed on two machines with a maximum delay of eight units. All activities are assumed to have the same delay range which is picked from the integer interval: $x \in [0, 8]$. Table 7 shows the CDF of delays defined as a table of integers (in a tabular form) following the triangular distribution (0, 4, 8).

Table 7: The cdf of different delay values

x	1	2	3	4	5	6	7	8
$cdf(x)$	0.0313	0.125	0.2813	0.5	0.7188	0.875	0.9688	1
$Log(cdf(x))$	-1.5051	-0.9031	-0.5509	-0.301	-0.1434	-0.058	-0.0138	0

Figure 3 shows the obtained activity delays using the CP model with chance constraint 1 for different confidence thresholds. The presented delays in each column meet the chance constraint according to

the defined confidence threshold. Figure 4 demonstrates the delays for activities considering the second chance constraint where the confidence threshold is imposed on activities carried out on each machine. As shown in Figures 3 and 4, the second chance constraint is less restrictive than the first one since it results in lower delays for smaller threshold values.

		Confidence threshold									
Activity	Machine	**1**	**0.9**	**0.8**	**0.7**	**0.6**	**0.5**	**0.4**	**0.3**	**0.2**	**0.1**
1	M1	8	7	7	7	6	6	6	6	6	6
2		8	7	7	7	6	6	6	6	6	6
3		8	7	7	7	7	7	8	6	6	6
4		8	8	7	7	7	7	7	6	6	6
5		8	8	7	7	8	8	6	6	6	6
6	M2	8	8	7	7	7	6	7	7	6	6
7		8	8	7	7	7	6	6	6	6	5
8		8	8	8	7	7	7	6	6	6	5
9		8	8	8	7	7	7	7	7	6	4
10		8	8	8	7	7	7	6	6	5	6

Figure 3: The delay of activities for different threshold values using confidence constraint 1

		Confidence threshold									
Activity	Machine	**1**	**0.9**	**0.8**	**0.7**	**0.6**	**0.5**	**0.4**	**0.3**	**0.2**	**0.1**
1	M1	8	7	7	6	6	6	6	6	5	4
2		8	7	7	6	6	6	6	5	5	5
3		8	7	7	7	7	6	6	5	5	4
4		8	8	7	7	6	6	6	5	5	5
5		8	8	7	8	7	6	5	7	6	6
6	M2	8	7	7	6	6	6	5	5	5	4
7		8	7	7	6	6	6	6	6	5	4
8		8	7	7	7	7	6	6	5	5	5
9		8	8	7	7	6	6	6	5	5	5
10		8	8	7	8	7	6	6	7	6	6

Figure 4: The delay of activities for different threshold values using confidence constraint 2

4 Experiments and results

The CP models are implemented in IBM ILOG CPLEX Optimization Studio version 12.8.0 and solved using the Constraint Programming Optimizer. All tests were carried out on a computer with an Intel(R) Core(TM) i7-9750H CPU @ 2.60GHz processor and 16 GB of RAM. The presented models are tested on two real-world data sets from an operating underground gold mine located in Canada. The data sets include the set of activities, their processing time, predecessor activities, and the site where each activity must be performed. Furthermore, the set of all sites, available equipment types, and the matrix of machine travel times between sites are provided. The fleet of trucks is not considered in these data sets and it is assumed there are enough trucks available in the mine. The first instance is composed of 13 machines and 151 activities located in 10 sites. The second instance includes 15 machines, 241 activities and 18 sites. Available resources are divided into five equipment types where numbers of each equipment type are presented in Table 8.

In our data sets, the duration of different activity types follows the triangular distributions (a, b, c) as presented in Table 9. These statistical distributions are built using historical data of activity durations (including potential delays). We used a temporal discretization for the duration of activities, where a time unit is equal to ten minutes in reality.

Table 8: Number of machines per equipment type for Instances 1 and 2

Equipment	Number of available equipment Instance 1	Instance 2
Scooptram	2	2
Bolter	4	6
Scooptram clean face	1	1
Jumbo	3	3
Anfo loader	3	3

Table 9: Triangular distributions of activity durations

Activity types	Minimum(a)	Average(b)	Maximum(c)	Triangular distributions
Mucking	13	16	26	(13, 16, 26)
Bolting	36	43	72	(36, 43, 72)
Clean face	3	5	9	(3, 5, 9)
Drilling	16	24	33	(16, 24, 33)
Explosive charging	5	8	18	(5, 8, 18)

In the confidence-constraint approach, delays are chosen from integer intervals. Each activity type has a different range of delays as follows:

- Mucking: range of delays : [0, 13]
- Bolting: range of delays : [0, 36]
- Clean face: range of delays : [0, 6]
- Drilling: range of delays : [0, 17]
- Explosive charging: range of delays : [0, 13]

The CDFs of delays are defined in a tabular form (a table of integers) following triangular distributions shown in Table 9. Since different activity types take different delays, for every activity type, a specific range of CDF values is defined and used in the confidence constraint to determine the delay. The delay times of various activity types for different confidence threshold values are shown in Table 10.

Table 10: The delay time of activities for different confidence threshold values

Activity type	1	0.95	0.90	0.85	0.80	0.75
Mucking	13	11	10	9	8	8
Bolting	36	29	26	24	22	20
Clean face	6	5	5	5	4	4
Drilling	17	15	14	13	12	11
Explosive charging	13	11	10	9	8	8

Figures 5a and 5b show the minimum duration and the delay time of bolting and drilling operations for different confidence threshold values. As can be seen, lower threshold values in the confidence constraint result in smaller delays for activities. Figure 6 compares the delay time of several activity types for three threshold values.

An evaluation model is used to assess the robustness of the schedules (sequences of activities on machines) generated using different CP models. In the first step, a schedule is produced via a CP model. In the confidence-constraint approach, a threshold probability (γ) is given as input to generate a robust solution. For the scenario-based model, the number of scenarios implemented is predefined. Next, the created sequences of activities are applied to multiple scenarios with different durations where activities are sequenced on machines in the same order (with a fixed order) across all scenarios.

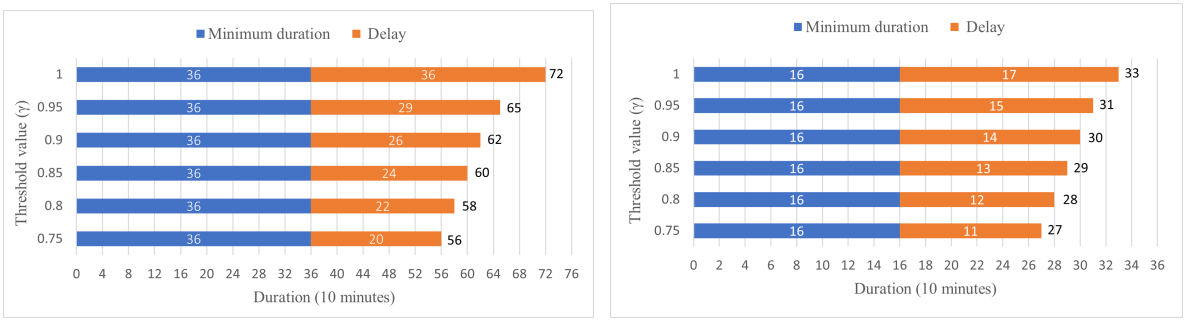


Figure 5: Minimum duration and delay of bolting (a) and drilling (b) activities for different threshold values

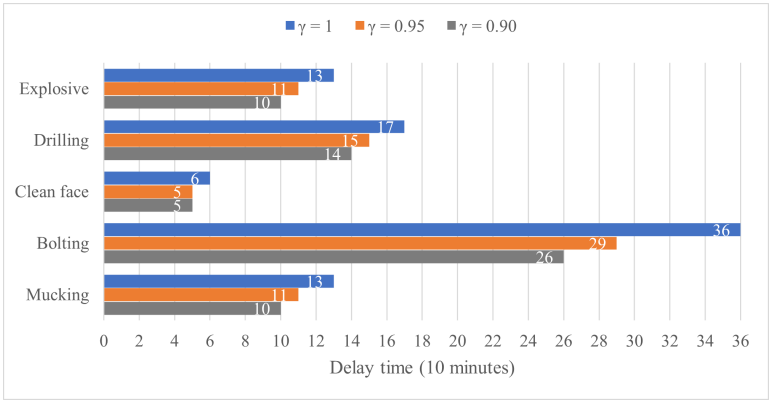


Figure 6: Delay time of several activity types for three threshold values

Each scenario represents a realization of the uncertainty about activity durations (variation of activities duration) based on statistical distributions. Historical data of activity durations are used to build statistical distributions. The evaluation procedure is applied to different CPO models and the average makespans of generated sequences are computed. Figure 7 shows the procedure of the evaluation model.

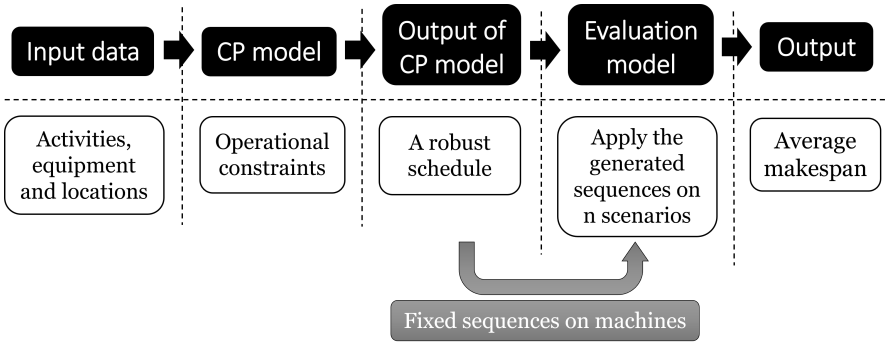


Figure 7: Main steps of the evaluation model (procedure)

In this paper, different schedules are generated using different (versions of) CP models defined as follows:

A: The deterministic CP model without confidence constraint that generates schedules considering the minimum durations as inputs.

B: The deterministic CP model without confidence constraint that generates schedules considering the average durations as inputs.

C: The deterministic CP model without confidence constraint that generates schedules considering the maximum durations as inputs. This model is equivalent to the CP confidence model with the confidence threshold value equal to 1 (γ=1).

$C1_\gamma$: The CP model that employs confidence constraint 1 to generate schedules considering sufficiently long durations for activities with a defined threshold probability ($\gamma < 1$). CP models with different γ are compared in this paper.

$C2_\gamma$: The CP model with confidence constraint 2 that ensures sufficient durations for activities in the schedule according to the given confidence threshold ($\gamma < 1$).

S_n: The stochastic CP model with the same sequence constraint which is solved over n scenarios with different activity durations.

Different models produce different ordered sequences of activities on available machines. The proposed CP models are tested on two data sets from a Canadian underground gold mine. Results are presented in the following subsections.

4.1 Confidence-constraint models

Table 11 shows the results of deterministic CP models and CP models with confidence constraint 1 ($C1_\gamma$), solved on Instance 1. In this table, columns 2, 3, and 4 indicate the value of the makespan, the optimality gap, and the solving time to produce a schedule. As demonstrated in this table, decreasing the threshold threshold (γ) in the CP chance model increases the solving time and the solution gap. Models with smaller γ should select the best delay for each activity among larger delay ranges that increase the difficulty to solve the problem. In other words, the solver requires a longer computational time to choose the most appropriate delay value for every activity while satisfying the confidence constraint and minimizing the makespan. Thus, reducing γ increases the problem's difficulty where the search algorithm gets stuck in a local minimum and the CPO solver returns a non-optimal solution once the time runs out.

CP chance models that are not solved to optimality do not show significant changes in the solution gap after one hour of running. However, models have been executed (run) for up to 10 hours to achieve the best possible solution within this timeframe.

Table 11: Makespan of different models solved on Instance 1

Model	Makespan	Gap%	Solving time
A	365	optimal	3 sec
B	480	optimal	3 sec
C ($\gamma = 1$)	790	optimal	4 sec
$C1_{0.95}$	774	optimal	7 min 31 sec
$C1_{0.90}$	766	optimal	4 hours 6 min
$C1_{0.85}$	761	6.83%	10 hours
$C1_{0.80}$	754	9.55%	10 hours

Table 12 presents the evaluation model results for schedules generated using different models on Instance 1. In this table, the second and third columns show the average makespan and the standard deviation obtained using the schedule over 100 simulations. According to this table, $\gamma = 0.85$ seems to be the best threshold for the CP chance model, resulting in the lowest average makespan in the evaluation model. CP chance models generate more robust schedules with smaller average makespans than deterministic models.

As can be seen in Tables 11 and 12, there is a (significant) difference between the makespan (Table 11) and the average makespan (Table 12) of CP chance models. The makespan is the total

Table 12: Results of the evaluation model on Instance 1

Model	Average makespan	Standard deviation
A	578.24	19.945
B	570.25	19.932
C ($\gamma = 1$)	567.33	17.278
$C1_{0.95}$	566.04	17.041
$C1_{0.90}$	560.15	18.158
$C1_{0.85}$	558.96	17.915
$C1_{0.80}$	560.43	18.317

length of the schedule with sufficiently long activity durations respecting the confidence threshold. The average makespan is the mean makespan of 100 simulated schedules in the evaluation process with activity durations following the triangular distribution of Table 9.

The output of deterministic CP models and CP models with confidence constraint 1, tested on Instance 2 is presented in Table 13. Table 14 compares the average makespan and the standard deviation of generated schedules over 100 scenarios in the evaluation model. As shown in this table, $\gamma = 1$ results in the lowest average makespan when applied to the CP chance constraint model. Chance models with confidence thresholds less than one ($\gamma < 1$) fail to outperform models A and B in the evaluation procedure. A comparison of Tables 12 and 14 demonstrates that the CP model with confidence constraint 1 produces more robust solutions on Instance 1 than on Instance 2. Due to the larger size of Instance 2, chance models solved on this instance show higher solution gaps and therefore are less robust than models applied to Instance 1.

Table 13: Makespan of different models solved on Instance 2

Model	Makespan	Gap%	Solving time
A	375	optimal	6 sec
B	480	optimal	6 sec
C ($\gamma = 1$)	790	optimal	7 sec
$C1_{0.95}$	779	2.82%	10 hours
$C1_{0.90}$	777	7.98%	10 hours
$C1_{0.85}$	777	11.33%	10 hours
$C1_{0.80}$	775	12.77%	10 hours

Table 14: Results of the evaluation model on Instance 2

Model	Average makespan	Standard deviation
A	588.33	15.734
B	581.03	15.225
C ($\gamma = 1$)	577.13	16.481
$C1_{0.95}$	582.96	14.754
$C1_{0.90}$	585.40	12.715
$C1_{0.85}$	583.73	13.483
$C1_{0.80}$	585.32	13.948

Table 15 presents the results of deterministic CP models and CP models with confidence constraint 2 ($C2_\gamma$), applied to Instance 1. This table shows that using smaller confidence threshold values (γ) in the CP chance model increases the solution gap.

Table 16 shows the evaluation model results for different schedules over 100 simulations on Instance 1. As can be seen in this table, models with $\gamma = 0.85$ and $\gamma = 0.75$ produce more robust schedules with lower average makespans in the evaluation model. Although the solution of chance models with $\gamma < 1$ is not optimal, their average makespans are smaller than those of models A, B,

and C. Therefore, the CP model with confidence constraint 2 builds more robust schedules than the deterministic CP model on Instance 1.

Table 15: Makespan of different models solved on Instance 1

Model	Makespan	Gap%	Solving time
A	365	optimal	3 sec
B	480	optimal	3 sec
C ($\gamma = 1$)	790	optimal	4 sec
$C2_{0.95}$	737	1.89%	10 hours
$C2_{0.90}$	729	5.08%	10 hours
$C2_{0.85}$	705	5.39%	10 hours
$C2_{0.80}$	693	7.50%	10 hours
$C2_{0.75}$	686	9.18%	10 hours

Table 16: Results of the evaluation model on Instance 1

Model	Average makespan	Standard deviation
A	578.24	19.945
B	570.25	19.932
C ($\gamma = 1$)	567.33	17.278
$C2_{0.95}$	554.31	18.90
$C2_{0.90}$	548.63	19.749
$C2_{0.85}$	546.93	19.326
$C2_{0.80}$	547.91	20.993
$C2_{0.75}$	546.77	20.144

The results of deterministic CP models and CP models with confidence constraint 2, solved on Instance 2, are shown in Table 17.

Table 17: Makespan of different models solved on Instance 2

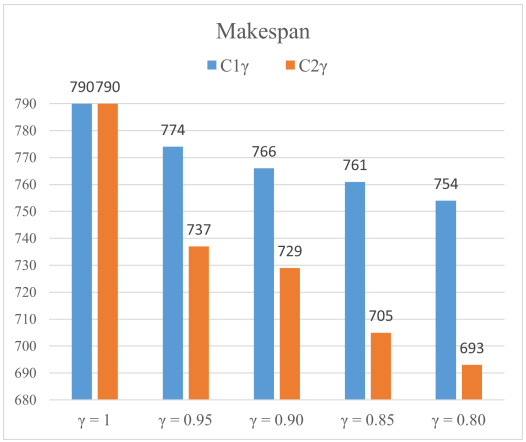
Model	Makespan	Gap%	Solving time
A	375	optimal	6 sec
B	480	optimal	6 sec
C ($\gamma = 1$)	790	optimal	7 sec
$C2_{0.95}$	750	3.87%	10 hours
$C2_{0.90}$	744	7.12%	10 hours
$C2_{0.85}$	742	10.24%	10 hours
$C2_{0.80}$	727	12.10%	10 hours
$C2_{0.75}$	723	13.97%	10 hours

Table 18 presents the average makespan and the standard deviation of schedules over 100 scenarios in the evaluation process. As demonstrated in this table, CP models with confidence constraint 2 generate schedules with lower average makespans than deterministic CP models A and B. Based on Table 18, $\gamma = 0.95$ appears to be the best threshold for the CP chance model since it results in the lowest average makespan. CP models with confidence constraint 2 perform better on Instance 1 than on Instance 2 by achieving smaller average makespans in the evaluation process, as shown in Tables 16 and 18.

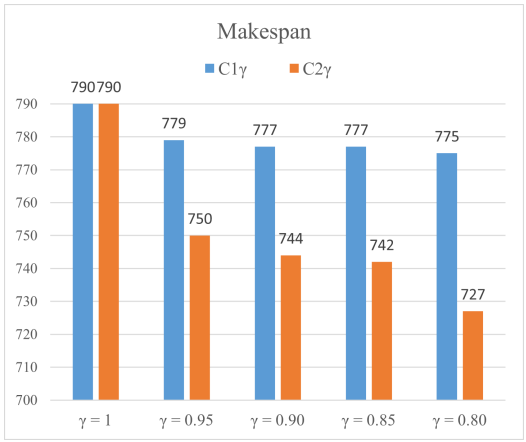
Figures 8a and 8b compare the makespan of schedules produced using CP models with confidence constraints and different confidence thresholds (γ) on Instances 1 and 2. According to these figures, the first confidence constraint imposes a greater restriction on the duration of activities than the second one, resulting in higher makespans. In Figure 8b, the CP model with the first confidence constraint yields the same objective value (makespan) at $\gamma = 0.90$ and $\gamma = 0.85$. Generally, decreasing γ results in a smaller makespan. However, due to the difficulty of the problem on Instance 2, the search algorithm is trapped in a local minimum that returns an inferior (poor) solution when the time runs out.

Table 18: Results of the evaluation model on Instance 2

Model	Average makespan	Standard deviation
A	588.33	15.734
B	581.03	15.225
C $(\gamma = 1)$	577.13	16.481
$C2_{0.95}$	575.77	13.516
$C2_{0.90}$	579.09	13.936
$C2_{0.85}$	580.72	14.685
$C2_{0.80}$	578.86	15.842
$C2_{0.75}$	580.50	15.470



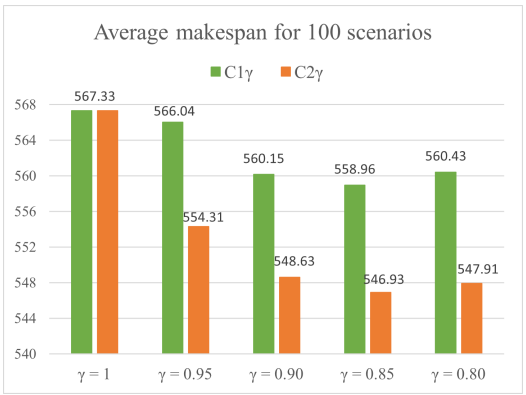
(a)



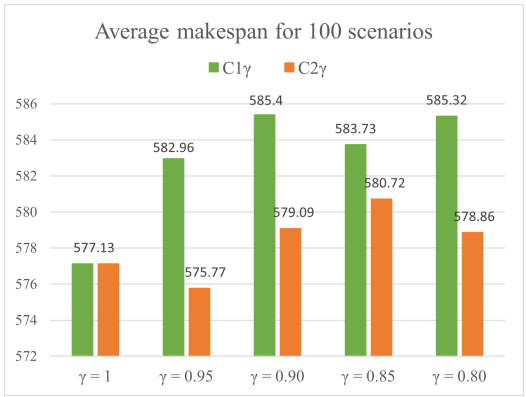
(b)

Figure 8: Makespan of schedules generated using confidence-constraint models with different threshold values on Instances 1 (a) and 2 (b)

Figures 9a and 9b show the average makespan of generated schedules using two CP confidence models with different γ over 100 scenarios on Instances 1 and 2. As demonstrated in these figures, CP models with confidence constraint 2 outperform CP models with confidence constraint 1 on both Instances 1 and 2 by achieving smaller average makespans in the evaluation process.



(a)



(b)

Figure 9: Results of the evaluation process for confidence-constraint models with different threshold values on Instances 1 (a) and 2 (b)

4.2 Scenario-based models

Table 19 presents the output of scenario-based models (stochastic CP models) for 10, 25, and 50 scenarios solved on Instance 1. Figure 10 demonstrates the evaluation model results over 100 simulations for schedules generated by scenario-based models on Instance 1.

Table 19: Makespan of scenario-based models solved on Instance 1

Model	Number of scenarios	Mean makespan	Gap%	Solving time
S_{10}	10	542.30	optimal	22 sec
S_{25}	25	543.40	optimal	1 min 20 sec
S_{50}	50	542.68	optimal	11 min

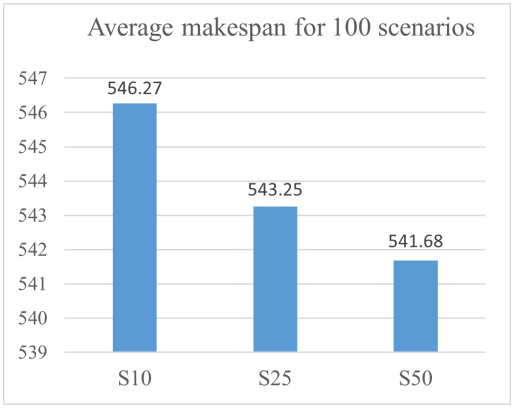


Figure 10: Results of the evaluation process for scenario-based models on Instance 1

The results of stochastic CP models solved with 10, 25, and 50 scenarios on Instance 2 are shown in Table 20. Figure 11 compares the average makespan of schedules produced using scenario-based models over 100 scenarios on Instance 2.

Table 20: Makespan of scenario-based models solved on Instance 2

Model	Number of scenarios	Mean makespan	Gap%	Solving time
S_{10}	10	554.90	2.27%	1 hour
S_{25}	25	561.28	3.19%	1 hour
S_{50}	50	565.56	4.05%	1 hour

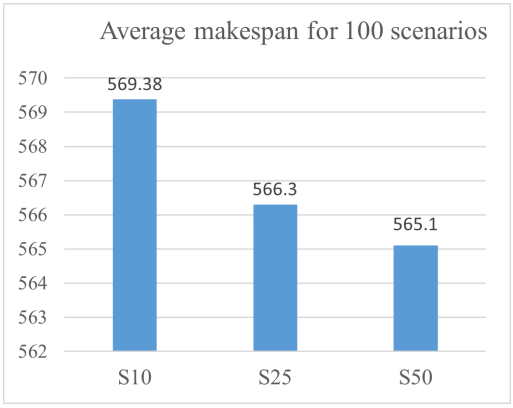


Figure 11: Results of the evaluation process for scenario-based models on Instance 2

As demonstrated in Figures 10 and 11, model S_{50} (which is solved using 50 scenarios) produces the most robust schedule with the lowest average makespans in both Instances 1 and 2. According to Figures 10 and 11, stochastic models solved on larger scenario numbers generate more robust schedules. However, solving time and/or solution gap increase when more scenarios are applied to the model, as shown in Tables 19 and 20. In the scenario-based method, a key challenge is to achieve a trade-off between the number of scenarios implemented and the robustness of the solution since increasing the scenario number reduces the resolution process (increases the computational time).

4.3 Comparison of scenario-based, confidence-constraint, and deterministic models

Figures 12 and 13 compare evaluation model results for generated schedules using deterministic models (orange), CP models with confidence constraint 2 (green), and scenario-based models (blue) on Instances 1 and 2. As shown in these figures, the scenario-based approach outperforms confidence-constraint and deterministic methods by obtaining lower average makespans over 100 simulations in the evaluation model on both Instances 1 and 2. Although the confidence-constraint method does not outperform the scenario-based approach, it provides more robust schedules to uncertainties than the deterministic model.

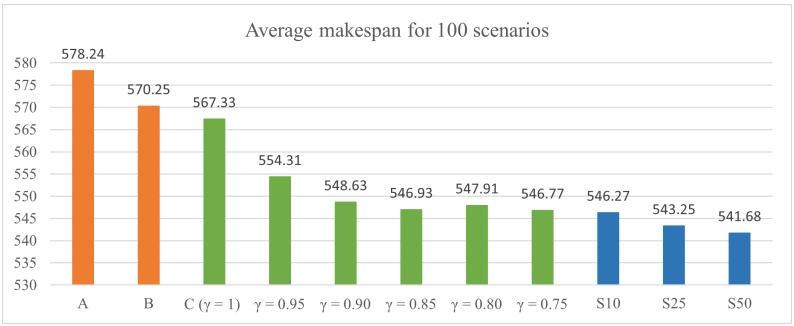


Figure 12: Average makespan of generated schedules using different models over 100 scenarios on Instance 1

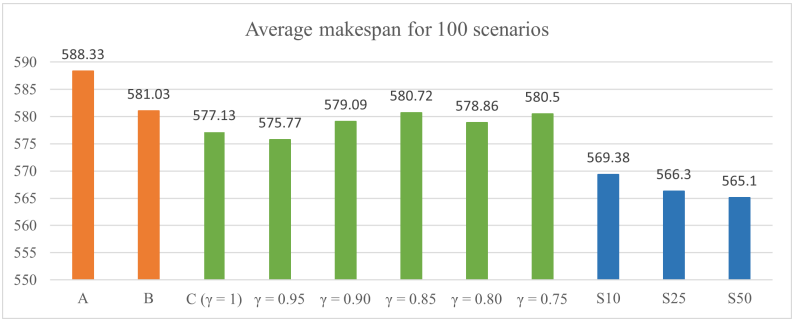


Figure 13: Average makespan of generated schedules using different models over 100 scenarios on Instance 2

5 Conclusion

This paper presented two different approaches called scenario-based and confidence-constraint using Constraint Programming (CP) to build feasible robust schedules. These approaches represent two ways to handle uncertainties in the short-term underground mine scheduling problem. Both models are tested on two real-world data sets from a Canadian underground mine. An evaluation model is used to evaluate the robustness of schedules (sequences of activities on resources) produced using different approaches. Compared to the deterministic model, the confidence-constraint and scenario-based

approaches create more robust schedules to uncertainties on new potential scenarios with different activity durations. The confidence-constraint model generates schedules that can be directly applied to underground mines under a certain level of risk tolerance. However, the scenario-based approach outperforms the confidence-constraint model on our instances. Future research should be devoted to developing a heuristic (search) algorithm to improve the quality of solutions in the chance model. In addition, further experiments could be conducted on the CP model with the second confidence constraint considering different threshold values for various machine types.

References

Max Åstrand, Mikael Johansson, and Jenny Greberg. Underground mine scheduling modelled as a flow shop: a review of relevant work and future challenges. Journal of the Southern African Institute of Mining and Metallurgy, 118(12):1265–1276, 2018a.

Max Åstrand, Mikael Johansson, and Alessandro Zanarini. Fleet scheduling in underground mines using constraint programming. In International conference on the integration of constraint programming, artificial intelligence, and operations research, pages 605–613. Springer, 2018b.

Max Åstrand, Mikael Johansson, and Alessandro Zanarini. Underground mine scheduling of mobile machines using constraint programming and large neighborhood search. Computers & Operations Research, 123: 105036, 2020.

Louis-Pierre Campeau and Michel Gamache. Short-term planning optimization model for underground mines. Computers & Operations Research, 115:104642, 2020.

Louis-Pierre Campeau and Michel Gamache. Short-and medium-term optimization of underground mine planning using constraint programming. Constraints, pages 1–18, 2022.

Louis-Pierre Campeau, Michel Gamache, and Rafael Martinelli. Integrated optimisation of short-and medium-term planning in underground mines. International Journal of Mining, Reclamation and Environment, 36 (4):235–253, 2022.

John J Kanet, Sanjay L Ahire, and Michael F Gorman. Constraint programming for scheduling. 2004.

P. Laborie. An update on the comparison of MIP, CP and hybrid approaches for mixed resource allocation and scheduling. volume 10848 LNCS of Lecture Notes in Computer Science, pages 403–411. Springer Verlag, 2018.

P. Laborie, J. Rogerie, P. Shaw, and P. Vilim. IBM ILOG CP optimizer for scheduling: 20+ years of scheduling with constraints at IBM/ILOG. Constraints, 23(2):210–50, 2018. doi: 10.1007/s10601-018-9281-x.

Fabián Manríquez, Javier Pérez, and Nelson Morales. A simulation–optimization framework for short-term underground mine production scheduling. Optimization and Engineering, 21(3):939–971, 2020.

Alexandre Mercier-Aubin, Ludwig Dumetz, Jonathan Gaudreault, and Claude-Guy Quimper. The confidence constraint: A step towards stochastic cp solvers. In International Conference on Principles and Practice of Constraint Programming, pages 759–773. Springer, 2020.

Micah Nehring, Erkan Topal, and Peter Knights. Dynamic short term production scheduling and machine allocation in underground mining using mathematical programming. Mining Technology, 119(4):212–220, 2010.

Dónal O'Sullivan and Alexandra Newman. Optimization-based heuristics for underground mine scheduling. European Journal of Operational Research, 241(1):248–259, 2015.

Gilles Pesant. A constraint programming primer. EURO Journal on Computational Optimization, 2(3):89–97, 2014.

Marco Schulze and Jürgen Zimmermann. Staff and machine shift scheduling in a german potash mine. Journal of Scheduling, 20(6):635–656, 2017.

Marco Schulze, Julia Rieck, Cinna Seifi, and Jürgen Zimmermann. Machine scheduling in underground mining: an application in the potash industry. OR spectrum, 38(2):365–403, 2016.

Cinna Seifi, Marco Schulze, and Jürgen Zimmermann. A two-stage solution approach for a shift scheduling problem with a simultaneous assignment of machines and workers. In The 39th International Symposium on Application of Computers and Operations Research in the Mineral Industry, page 377, 2019.

Zhen Song, Håkan Schunnesson, Mikael Rinne, and John Sturgul. Intelligent scheduling for underground mobile mining equipment. PloS one, 10(6):e0131003, 2015.

Hao Wang, Victor Tenorio, Guoqing Li, Jie Hou, and Nailian Hu. Optimization of trackless equipment scheduling in underground mines using genetic algorithms. Mining, Metallurgy & Exploration, 37(5):1531–1544, 2020.