Real-Time Truck Dispatch System for Short-term Production Planning in an Industrial Mining Complex: A fast mechanism using reinforcement learning

COMP 652 – Machine Learning

Ashish Kumar (260656558)
Ph.D., Mining and Material Engineering Department,
McGill University
Cosmo Stochastic Mine Planning Laboratory
ashish.kumar@mail.mcgill.ca

Joao Pedro de Carvalho (260642102)
Ph.D., Mining and Material Engineering Department,
McGill University
Cosmo Stochastic Mine Planning Laboratory
joao.decarvalho@mail.mcgill.ca

A mining complex is an integrated supply chain network where the material is first extracted from the mine and then sent to different processing destinations to produce the sellable products. New digital technologies including the development of advanced sensors and monitoring devices have enabled such a supply chain network to acquire real-time information about its different components. For instance, measuring the performance, availability, productivity of trucks, shovels, processing mills, the quality of material handled with the trucks, shovels. A minor change in the performance of any component can result in massive changes in the state of the mining complex and consequently in the net revenue. This work focuses on developing a reinforcement learning framework that can adapt truck assignment/dispatch decisions based on real-time information gathered during the mining operation and state of the mining complex. The proposed framework uses a discrete event simulator to build a realistic environment that reflects the complexity of operations in a mining complex. The discrete event simulator simulates the response of the environment based on state-dependent actions which are then used to train a neural network agent through reinforcement learning. The reinforcement learning based neural network truck dispatch policies are compared with fixed truck dispatch, random truck dispatch, and, greedy truck dispatch policies

Keywords — real-time truck dispatching, reinforcement learning, mining complexes.

I. Introduction

A mining complex is a network of interrelated operations that integrate all aspects in a mineral value chain, starting from the materials extracted from the ground culminating with its transformations into a final product delivered to the mineral market. Uncertainty is a characteristic of a mining complex, starting from the supply of different types of materials extracted from the mineral deposits involved and the performance of its different components. With the advent of inexpensive sensors and digital storage, increasing amounts of data about a mining complex can be easily collected. Sensors installed on shovels, trucks, conveyor belt, and processing mills continuously measure the performance of the mining equipment, processing facilities, and, handling facilities (Koellner et al., 2004), as well as different pertinent properties of the material being handled (Dalm et al., 2017). Fig. 1 shows a conceptual mining complex where the material is extracted with a shovel. The sensor on the shovel measures the quality of, and time required to extract the material from the mine. The shovel loads the extracted material into a truck, which travels to a destination. The sensor on the truck measures the time required to reach the destination. The truck dumps the material at the destination if there is no queuing. The sensor at the destination measures the quality of, and the time required to process the material. The emptied truck is then dispatched/assigned to one of the shovels at the mine. Existing technologies make these real-time truck dispatching decisions greedily to avoid queuing time/maximize truck utilization/maximize shovel utilization (Nguyen and Bui, 2015; Chaowasakoo et al., 2017), without accounting for the state of the mining complex. The state of the mining complex includes (i) the cycle time and queue time to extract, load, haul, dump, and process material, (ii) performance of the mining equipment, processing facilities, and, handling facilities, and (iii) quality of material extracted and processed.

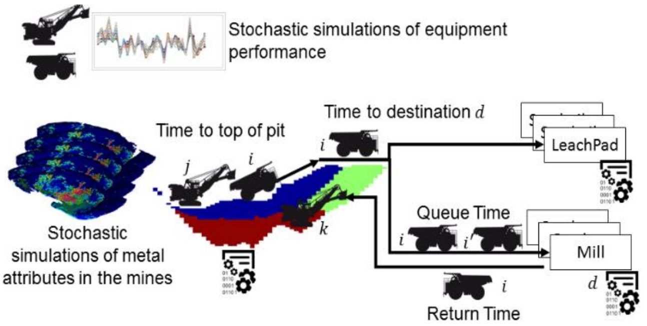
Stochastic simulations of equipment performance
Time to destination d
Time to top of pit
i
j
i
LeachPad
Stochastic simulations of metal attributes in the mines
k
Queue Time
i
i
Mill
i
d
Return Time

Fig. 1. A conceptual mining complex with real-time information.

This work proposes the use of state-dependent truck dispatch policies, which can make real-time dispatch decision based on the new information gathered from sensors, and the state of the mining complex. The state-dependent truck dispatch policies (in this work neural network agents) are trained using reinforcement learning. In recent years, reinforcement learning based methods have shown exceptional performance at generating neural network agents that are capable of making very efficient decisions for different complex environments (Aissani et al., 2012; Mnih et al., 2013; Silver et al., 2016). Paduraru and Dimitrakopoulos (2018) propose a reinforcement learning policy framework to optimize the neural network destination policies of materials in a mining complex while accounting for supply and equipment performance uncertainty. The neural network destination policies increased the expected cash flow by 6.5 % compared to an optimized state-independent destination policy for a copper mining complex.

II. Method

The state-dependent truck dispatch policies rely on building a realistic model of the environment that can encapsulate all the factors affecting the response of the environment for a given action. The environment for truck dispatch policies is the various components and their interaction in a mining complex, which starts from the extraction of material until the product is shipped. Also, the different components have an inherent uncertainty associated with its performance, for instance, the quality of material extracted from the ground, the performance of the shovels, trucks and mills. The uncertainty with the quality of material in the ground is captured through a set of Monte Carlo based geostatistical simulations (Goovaerts, 1997; Boucher and Dimitrakopoulos, 2012). The uncertainty with the performance of the different equipment is captured through different distributions such as Gaussian distribution for shovel extraction time, moving time, truck hauling time, amongst others. A discrete event simulator is built to model realistically the flow of material in a mining complex considering the uncertainty related to geology, and equipment. The discrete event simulator is used to generate responses of the environment for the actions taken with the truck dispatch policies. The truck dispatch policies are trained via reinforcement learning using these response-action combinations.

A. *Discrete event simulation*

A possible configuration of the connections is represented in Fig. 2, where each shovel has a fixed destination where the material should be sent (black connections) and potential places where the trucks can be sent after dumping (red connections). At each location, shovel or destinations, there is, potentially, a queue of trucks waiting to be loaded or to dump. Then after the trucks operate their duty, they are assigned to a different destination.

The sequence of events can be modelled through a discrete event framework. Three main events considered are: “Shovel Finishes,” “Truck Moves” and “Dumps at Destination,” which are explained in the next sub-sections. These events are ranked according to their starting time in a priority queue “queue of events.”

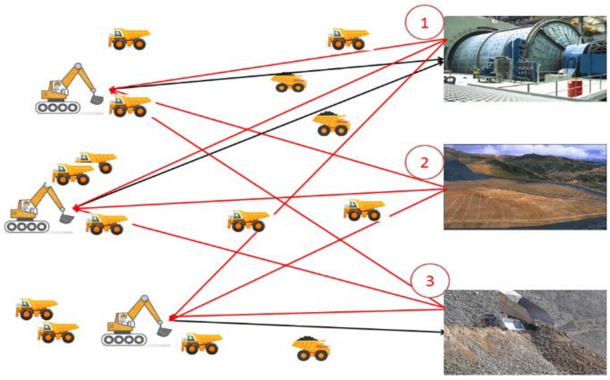

Fig. 2. Possible configuration of the connections in a mining complex.

1) *Shovel Finishes*

The “Shovel Finishes” event is related to the shovel extracting the material and loading into the truck. After this event is finished (Shovel Finishes), potentially two new events are triggered First, the truck is sent (“Truck Moves”) to the pre-defined destination (mill, leach pad or waste dump). Second, if trucks are waiting at this shovel, a new truck is popped out from the shovels' priority queue and “Shovel Finishes” event is created and placed in the event’s queue.

2) *Truck moves*

This event is related to the transfer of the truck from one location to another: from shovel to the destination or vice-versa. Once this event is finished, a new “Shovel Finishes” or “Dumps at Destination” event is created, depending on where the truck is headed.

3) *Dumps at Destination*

The truck arriving at a destination (*mill, leach pad or waste dump*) triggers the “Dumps at Destination” event, where the material is delivered to a feed pile positioned in front of the destination. After this event finishes, a new “Truck Moves” event is placed in the event’s queue and if another truck is waiting to dump at this destination, a new “Dumps at Destination” events is created.

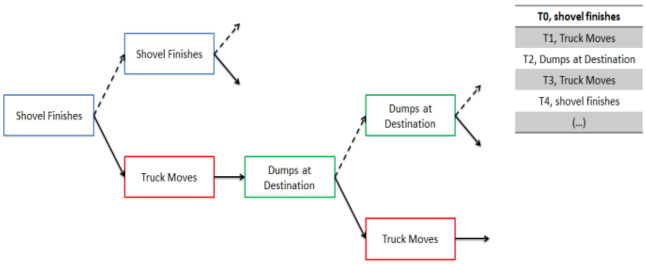

Fig. 3. Discrete event simulator and queue of events

Fig. 3 represents the sequence of events that can be generated from the three events described above. After each event is finished, some others are enqueued in a priority queue that is sorted by starting time.

B. Reinforcement Learning Truck Dispatch Policies

The current work assumes that the shovel has the information of where the material should be sent. However, after a truck has dumped its material at a certain destination, a decision needs to be made regarding which shovel the truck should go back. This setting is a dynamic and complex environment, where the global best policy is not known. Additionally, the policy that is taken now can trigger different events which are associated with many stochastic parameters: times taken to load, move and dump the truck. A smart strategy should not only consider if there is a queue in the system, but also the quality of material available to extract, the quantity of material being processed. The paper proposes the use of a robust strategy (state-dependent policies) that can capitalize on different possible configuration of the mine complex. The truck dispatch policies are a neural network agent that decides where to send the truck given the state of the mining complex (Fig. 4). The neural network is trained using reinforcement learning (Fig. 5) which includes using the discrete event simulator to generate the state of the mining complex, take decisions with the neural network agent given this state, evaluate reward and then perform gradient descent to adjust the weights of the neural network. The reward is an expected value over different geostatistical scenario and is calculated as the value of products generated minus the penalty for deviating from different targets. The rewards are not instantaneous; instead, they come after an episode of 30 minutes when the material gets processed. The rewards are then linked to all the different actions taken during this episode. The complete algorithm for the RL truck dispatch policies is mentioned below.

RL Truck Dispatch Policies Algorithm

This section outlines the complete RL algorithm for neural network-based truck dispatching policies. The different notation used in this section is mentioned in Table 2.

```
1. Read the set of geostatistical simulations for training, and the parameters related to the performance of equipment.
2. Do while rep<nIter:
    a. Initialize the configuration of the mining complex that includes the number of trucks, shovels, destination, capacities, schedule (blocks to be extracted with the shovels).
    b. Randomly initialize the location of trucks to a shovel
    c. Read the ε_RL, nCool, c.f, and the neural network parameters
    d. Start the discrete event simulator
    e. Do while t<HL:
        i. Do while t`<EL:
            1. Extract state of mine (features) using discrete event simulator
            2. Take actions with either the ε greedy policies or RL policies depending on the value of ε_RL. The ε greedy policies are used during the training phase to ensure exploration of solution space.
            3. Trigger the action taken and evaluate the time required for the action (sampling the equipment performance distribution) in the discrete event simulator
        ii. Process material and return reward = expected value of the product of geostatistical simulations- penalty for deviation from production targets (destination target, shovel production target)
        iii. Link reward to each action within the duration of the episode to generate labels
        iv. Store feature and labels to generate training data
        v. If rep = trainInterval:
            1. Train the neural network (Adam optimizer) by selecting randomly mini batches of size bSize from the training data for (nEpochs) number of epochs
            2. Save the neural network weights
        vi. nEpochs+=nEpochs+1; bSize= bSize+1
        vii. If rep % nCool:
            1. ε_RL=ε_RL*c.f
3. Read the set of geostatistical simulations for testing and the parameter related to the performance of equipment
4. Test the trained policies on 100 new repetitions
5. Compute results regarding p10, p50, and p90 risk profiles for different production indicators
```

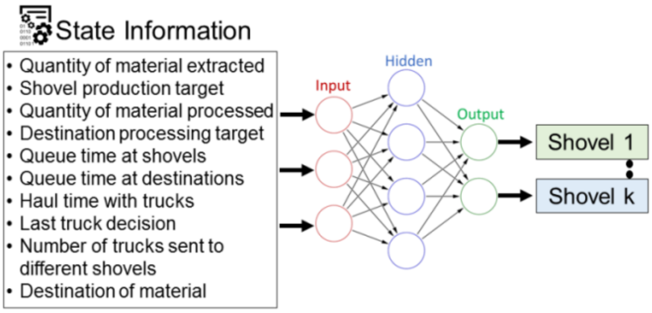
State Information
• Quantity of material extracted
• Shovel production target
• Quantity of material processed
• Destination processing target
• Queue time at shovels
• Queue time at destinations
• Haul time with trucks
• Last truck decision
• Number of trucks sent to different shovels
• Destination of material
Input
Hidden
Output
Shovel 1
Shovel k

Fig. 4. State vector of RL neural network dispatch policies

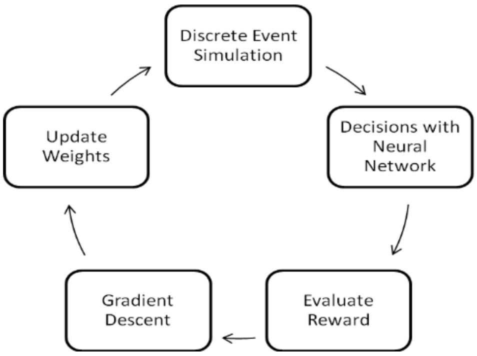

Fig. 5. Reinforcement learning for real-time truck dispatching policy.

III. CASE STUDY – COPPER MINING COMPLEX

The method proposed in this study considers a mine complex, composed of three shovels, one mill, one leach-pad, a waste dump, and 12 trucks of different sizes, is proposed. General parameters used are displayed in Table 1. The results for the RL dispatch policies are obtained using the parameters displayed in Table 2 and are compared with three different types of truck-dispatch policies. The results are presented after 100 repetitions, considering ten geostatistical simulations (the ones not used for training the RL dispatch policies).

The neural network agent used in the RL truck dispatch policies consist of two hidden layers with 100 and 50 neurons. The optimizer used for optimizing the neural network is Adam optimizer available with Tensorflow. The optimization parameters of the Adam optimizer are set to default values. The network is not very deep therefore only L2 regularization is used with a cost of 0.001. The RL framework uses a ε greedy policy mentioned below during the training phase to avoid local optima and explore the solution space. The ε value is gradually decreased during training with a cooling factor to ensure exploitation using the learned policies of the neural network and to ensure a balance between exploration and exploitation.

1) Fixed Truck Assignment

One operationally acceptable dispatch policy used by the industry is the one based on a fixed allocation. If a truck is assigned to operate with a specific shovel, it is never assigned to another.

2) Random Truck Assignment

As a means to evaluate a worst-case scenario, a random allocation is developed. After dumping the material at the destination, one of the shovels is chosen randomly to receive the truck.

3) E-greedy Truck Assignment

In this case, an E-greedy policy is adapted to choose a shovel with the smallest queue (local optima), in (1 - ε) % of the time. The value of ε was set to 0.85.

Table 1 – Configuration of the mineral value chain

Parameters	Value
Number of mines	1
Number of shovels	3(capacities of 75, 50 and 50 tonnes per bucket)
Destination and related target	3 (Mill - 750 tonnes, LeachPad 350 tonnes and Waste dump no limit)
Number of trucks	12 (capacity of 150, 200 or 250 tonnes)
Truck speed empty	40 km /h
Truck speed loaded	17 km/h
Number of geostatistical simulations for training and testing the policies	10 simulations
Copper price	8000 $/tonne Cu
Processing cost Mill	9 $ / tonne
Processing cost Leach Pad	9 $ / tonne
Mining cost	1 $ / tonne
Destination target	(6.25, 2.70, 16.6) tonne/minutes
Shovel target	(150, 100 100) tonne/minutes

Table 2 – Parameters of the RL Neural Network dispatching policies.

Parameter	Value
Episode length (EL)	30 minutes
Horizon length (HL)	1440 minutes
Input features	31 features
Hidden layer 1	100 neurons
Hidden layer 2	50 neurons
Output layer	3 classes (number of shovels)
Hidden layer activation	ReLu
Output layer activation	tanh
Optimizer	Adam optimizer
Cost function	Mean squared error
Regularization	L2 regularization of weights
Regularization cost	0.001
Weight initialization	Random normal
Library	Tensorflow with GPU
Batch size (bSize)	500 +1 with every repetition
Number of training repetition (nIter)	5000
Training epochs (nEpochs)	20 + 1 with every repetition
Cooling factor (c.f)	0.95
Cooling schedule (nCool)	Every 10 repetitions
Initial ε_RL	1
trainInterval	500 repetition

To compare the dispatching policy proposed in the current paper, three other policies are considered, and they are summarized in **Error! Reference source not found.**. The results are shown in terms of p10, p50 and p90, representing the 10th, 50th and 90th percentiles, respectively.

Table 3 - Summary of cases tested and expected profit obtained with each case.

	Policy	**Expected profit obtained**
Case 1	Fixed truck allocation	0.165 M$
Case 2	Random truck allocation	0.0835 M$
Case 3	E-Greedy strategy	0.135 M$
Case 4	RL - policy	0.307 M$

The RL policy improves substantially the cash-flow generated in 24 hours of production, as presented in Fig. 6. This strategy is more informed than the others because it can analyze many aspects of the current state of the value chain and, consequently, make a more profitable decision. The main reason for this increase in cash flow is that the RL finds configurations where the shortfall in production at the mill is minimized, Fig. 7. On the other hand, Case 2 does especially bad, and that results in two shortfalls in one day. Case 4 avoids extracting waste since the objective function maximizes profit; it attempts to defer the extraction from such shovel, Fig. 8. In the long-run, this is not the best strategy since mining waste is necessary to give access to new ore material in the future. Interesting to note that, as the e-greedy strategy minimizes queue times from all shovels, it is one of the cases which produce the most amount of waste, e.g. it makes the shovel extracting waste quite productive. This high productivity is presented in Fig. 9 for the shovel 1 (with the highest capacity), where Case 3 and 4 presents the highest production.

Fig. 10 shows the queue times for all the cases considered. Note that even though the random assignment does not have the objective to reduce queue time, it is still the one that performs best in this attribute at that shovel. Case 1 provides an average queue time that in general hovers below the average of Case 3, however, the worst-case scenario of Case 1 is considerably higher.

The main drawback of the current RL policy is the excessive queue time at the shovel. The optimizer prefers parking the trucks in the queue at the most productive shovel, rather than sending it to a less profitable shovel, that works with leach pad or waste materials. This can be an indicator that a smaller number of trucks can be considered for such a simplified version of the value chain. For future work, it will be considered to train the RL policy for a longer horizon length to obtain instances where different configurations of the value chain, for example, several shovels sending material to the same location. Also, it will be considered to input a feature that allows the truck to look ahead, such that it understands that it must extract more waste now to release more ore to be processed in the future.

In short, the RL is considered the best policy due to significant improvement in cash flow and the worst policy tested is the random allocation, and surprisingly the fixed truck generates more cash flow than the e-greedy case. Note that Cases 1, 2 and 3 do not optimize the generation of cash flow.

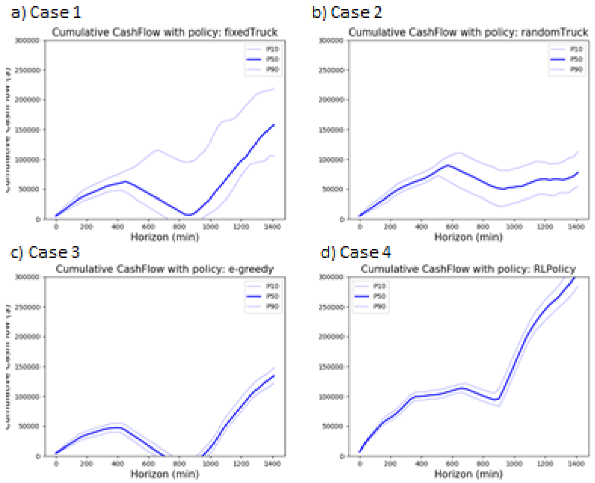

Fig. 6. Cumulative cash flow obtained with Cases 1, 2, 3 and 4.

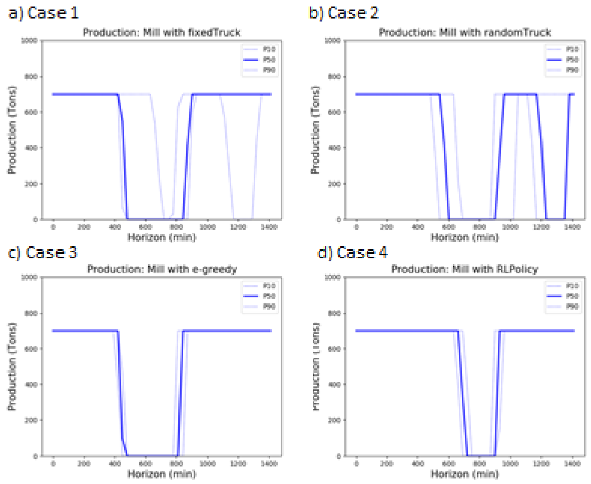

Fig. 7. Mill production obtained with Case 1, 2, 3 and 4.

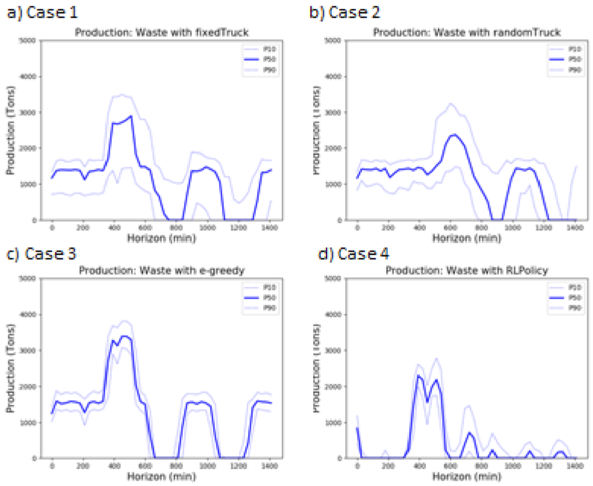

Fig. 8. Waste extraction obtained with Case 1, 2, 3 and 4.

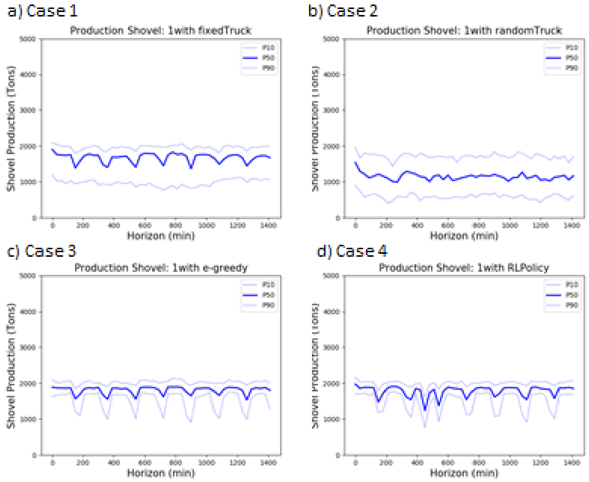

Fig. 9. Production of shovel 1 in Case 1, 2, 3 and 4.

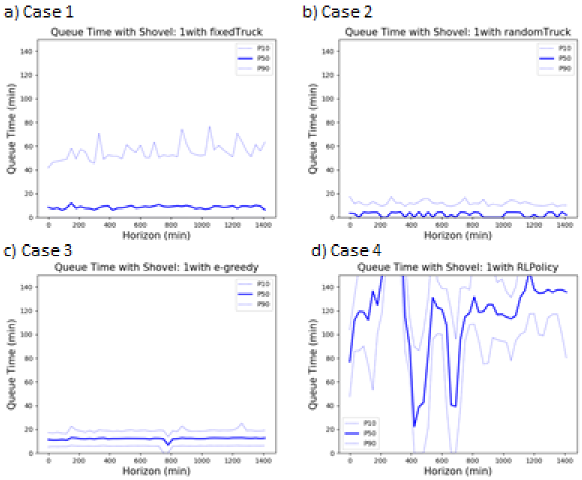

Fig. 10. Queue time at Shovel 1 obtained in Case 1, 2, 3 and 4.

IV. Additional Tests

The RL neural network agent in the previous section was trained using multiple repetitions of 24 hours of the discrete event simulation. As a means to assess the power of generalization of such policy, this section presents the results tested in 10 days of the discrete event simulation and compares to the E-greedy heuristic policy. The results displayed herein represent the p10, p50 and p90 related to 10 repetitions of the ten days simulation.

Results presented by Table 4 and Fig. 11 confirm the robustness of the trained policy. The RL neural network agent (Case 4) can provide a cash flow profile 30% higher than the one obtained the Case 3. Even though, Case 3 results in a slightly more consistent feed rate at the mill, Fig. 12, Case 4 defers the extraction of waste material to later periods, Fig. 13. Consequently, impacting the objective function positively and the RL agent learns such a sequence of actions.

Fig. 14 presents the productivity of the shovel 3, and it shows that Case 4 rarely uses the shovel in consideration. From an operational point of view, this may not be the most interesting strategy. However, it shows that the capability of RL policy to make more profitable decisions. These results motivate additional future work where the objective function can also aim to provide a more stable throughput at the mill and more consistent use of the equipment.

Table 4 – Summary of cases tested for 10 days of discrete event simulation and expected profit obtained.

	Policy	**Expected profit obtained**
Case 3	E-Greedy strategy	1.758 M$
Case 4	RL - policy	2.296 M$

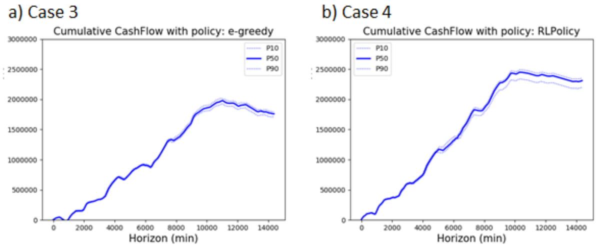

Fig. 11. – Cumulative cash flow obtained with Cases 3 and 4 for 10 days of dispatching policies.

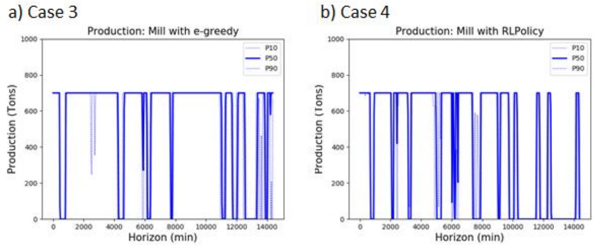

Fig. 12. – Mill production obtained with Cases 3 and 4 for 10 days of dispatching policies.

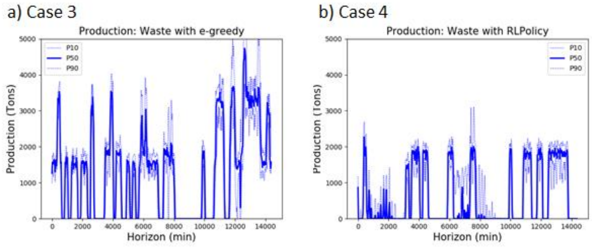

Fig. 13. – Waste production obtained with Cases 3 and 4 for 10 days of dispatching policies.

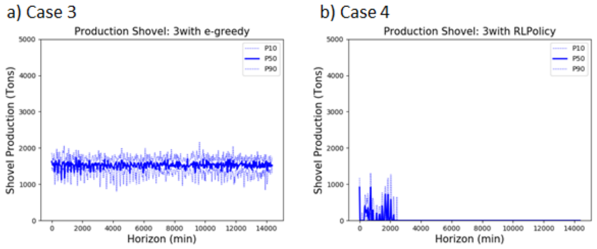

Fig. 14. – Shovel 3 production obtained with Cases 3 and 4 for 10 days of dispatching policies.

V. Conclusions

The paper proposes state-dependent neural network truck dispatch policies that can adapt the dispatch decisions of truck depending on the state of the mining complex, and capturing most synergies existing between these different components. The neural network truck dispatch policies are trained using reinforcement learning. The reinforcement learning algorithm includes a neural network agent that interacts with an environment (in our case a mining complex) to learn how to make truck dispatching decisions in a mining complex. The environment is modelled with a discrete event simulator that encapsulates the different transformation and interrelated activities that happens in a mining complex realistically. The RL framework uses different features that are extracted from the current state of the mining complex. This enables a real-time decision taking into consideration many different features, and not only queue times and targets, which are commonly prioritized in industrial operations.

The case study is performed at a copper mine composed by a mill, leach-pad, waste dump, three shovels and twelve trucks. First, the RL policy is trained for 24 hours of operation, and the results are compared to baseline policies (Case 1 and 2) and a heuristic approach (Case 3). When the RL policy is tested for 24 hours of operation, the cash flow obtained substantially overcomes all others. The mill is more consistently feed, and less waste is generated. Additional tests, consider the RL policy trained for 24 hours but applied to 10 days of production. Results show a proper power of generalization by presenting a cash-flow 30% than the Case 3 used as a comparison.

The limitation of the proposed policy lies in the operability of not using much one of the equipment and generating a long queue in the most efficient shovel. Future work will focus on making these decisions more operational by applying some daily targets. Additionally, to improve training phase multiple random rollouts of the future will be considered to generate better training information. More state features should also be tried to generate better policies and detailed sensitivity analysis with the different parameters of the RL framework.

References

[1] Dalm, M., Buxton, M.W.N., and van Ruitenbeek, F.J.A. (2017). Discriminating ore and waste in a porphyry copper deposit using short-wavelength infrared (SWIR) hyperspectral imagery. Minerals Engineering, 105, 10–18

[2] Koellner, W.G., Brown, G.M., Rodríguez, J., Pontt, J., Cortés, P., and Miranda, H. (2004). Recent advances in mining haul trucks. IEEE Transactions on Industrial Electronics, 51(2), 321–329

[3] Nguyen, D., and Bui, X. (2015). A real-time regulation model in multi-agent decision support system for open pit mining. In Proceedings: International Symposium Continuous Surface Mining-Aachen, Springer, Cham, 255–262

[4] Chaowasakoo, P., Seppälä, H., Koivo, H., and Zhou, Q. (2017). Digitalization of mine operations: Scenarios to benefit in real-time truck dispatching. International Journal of Mining Science and Technology, 27(2), 229-236

[5] Paduraru, C., and Dimitrakopoulos, R. (2018). Responding to new information in a mining complex: Fast mechanisms using machine learning. (Submitted), 1–30

[6] Aissani, N., Bekrar, A., Trentesaux, D., and Beldjilali, B. (2012). Dynamic scheduling for multi-site companies: A decisional approach based on reinforcement multi-agent learning. Journal of Intelligent Manufacturing, 23(6), 2513–2529

[7] Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. (2013). Playing Atari with deep reinforcement learning. ArXiv, 1312.5602, 1–9

[8] Silver, D., Huang, A., Maddison, C.J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. (2016). Mastering the game of Go with deep neural networks and tree search. Nature, 529(7587), 484–489

[9] Goovaerts, P. (1997). Geostatistics for Natural Resources Evaluation. Oxford University Press.

[10] Boucher, A., and Dimitrakopoulos, R. (2012). Multivariate Block-Support Simulation of the Yandi Iron Ore Deposit, Western Australia. Mathematical Geosciences, 44(4), 449-468.