Representation, Reasoning, and Learning for a Relational Influence Diagram Applied to a Real-Time Geological Domain

Matthew C. Dirks[*], **Andrew Csinger**[†], **Andrew Bamber**[†], **David Poole**[*]
[*]University of British Columbia (UBC), [†]Minesense Technologies Ltd.
mcdirks@cs.ubc.ca

Abstract

Mining companies typically process all the material extracted from a mine site using processes which are extremely consumptive of energy and chemicals. Sorting the good material from the bad would effectively reduce required resources by leaving behind the bad material and only transporting and processing the good material. We use a relational influence diagram with an explicit utility model applied to the scenario in which an unknown number of rocks in unknown positions with unknown mineral compositions pass over 7 sensors toward 7 diverters on a high-throughput rock-sorting machine developed by MineSense™ Technologies Ltd. After receiving noisy sensor data, the system has 400 ms to decide whether to activate diverters which will divert the rocks into either a keep or discard bin. We learn the model offline and do online inference. Our result improves over the current state-of-the-art.

This paper considers the problem of sorting rock ore based on mineralogy, separating the valuable, high-grade rocks from the low-grade rocks as they pass over an array of electromagnetic sensors. By sorting more effectively, we reduce costs and help preserve the environment, because the amount of material sent to further downstream mining processes is reduced. MineSense™ is a Vancouver-based company developing conductivity-based sensing and sorting systems to sort ore more effectively and have developed a rock sorting platform called SortOre™ pictured in 1.

We have developed an anytime algorithm (Zilberstein 1996) for sorting called Rock Predictor Sorting Algorithm (RPSA) which uses the SortOre™ rock sorting platform on which we have performed training and evaluation.

In SortOre™, rocks are dumped on a conveyor belt, moving downward on the y-axis as in the schematic in Figure 2. As rocks travel, they pass over a **sensor array** consisting of 7 electromagnetic coils which measure magnetic and electromagnetic flux and respond to materials which have conductive or magnetic properties. Each sensor takes readings of the magnitude of the magnetic field disrupted by the rock. The computer processes the sensor data while the rocks travel to the end of the **conveyor belt** where they fall off onto the **diverter array** which may have one or more di-

Figure 1: MineSense™ SortOre™

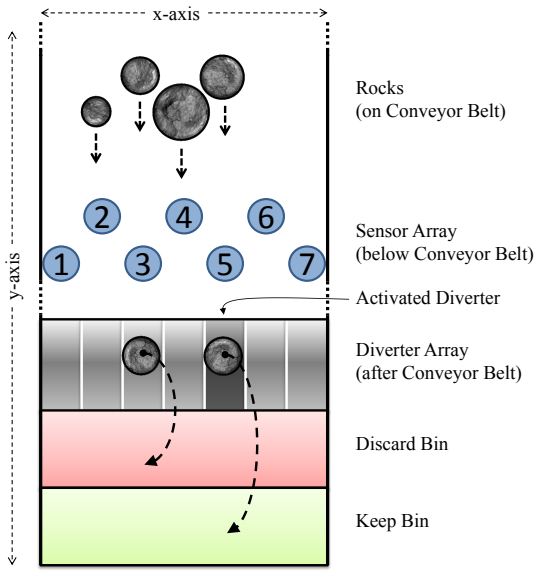

Figure 2: A schematic of the sorter.

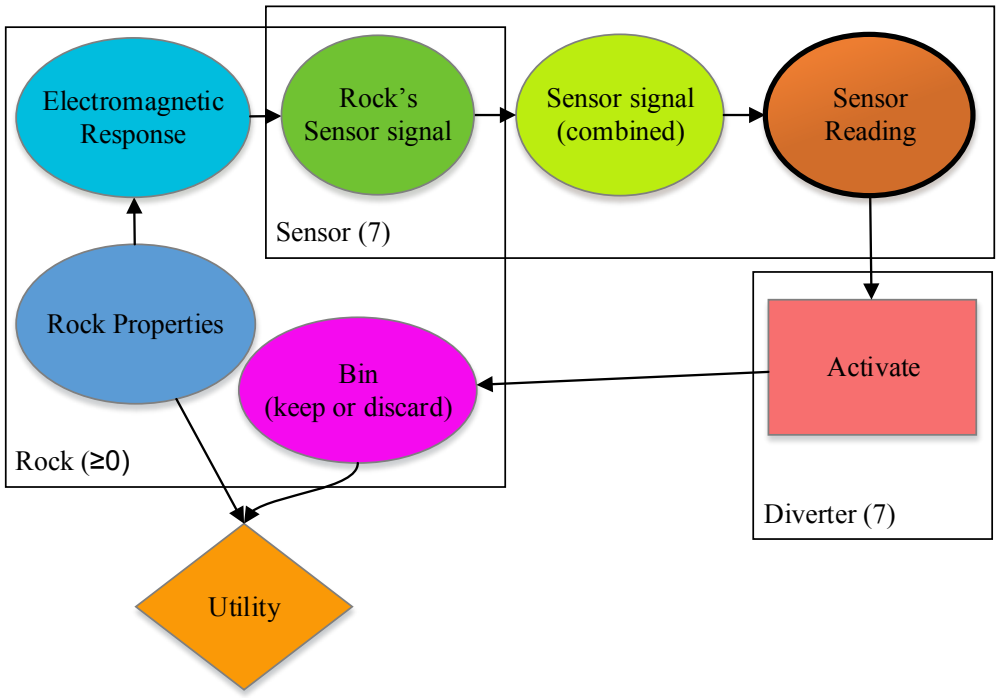

Figure 3: Relational Influence Diagram (High-Level)

verters **activated**, displacing rocks into the **keep bin** or else leaving them to fall into the **discard bin**.

Related Work

Surprisingly, little research has been done in the use of electromagnetic coils in sensors to detect and locate minerals. Similar research is that of detecting and locating unexploded ordnance beneath the ground, however at a long range and with the goal of identifying single objects at a time (Das et al. 1990). Likewise, inductive sensing has also been used in coin recognition, but again only considers the case of single objects (Passeraub et al. 1997).

The Model

The system can be described using a relational influence diagram (Shachter 1988) with plate notation (Buntine 1994). We chose to use a relational influence diagram to explain the model and to the reveal explicitly what is and isn't being modelled, though the solution we develop does not necessarily require it. Furthermore, we hope to use this diagram to discover new research areas by spotting where the diagram is lacking and to describe changes in the future in terms of adding or removing diagram elements (variables, arrows, plates, etc) making future work more clear. We have two diagrams: the first one (Figure 3), shows the system at a higher-level and we believe is conceptually easier to understand, and the second (Figure 4) describes the same system broken down into simpler components with their relations explained more clearly.

First, refer to the high-level influence diagram (Figure 3). There is an unknown number of rocks (zero or more), which may or may not be overlapping, each with **rock properties** describing the size, minerality, and position of the rock. To simplify the problem we model rocks as circles with some amount of mineral content spread equally throughout and currently we do not attempt to model mineral type.

A rock's properties influence the **electromagnetic response** it gives. The electromagnetic response variable models how a rock would respond to a sensor regardless of distance to the sensor.

The rocks do not necessarily pass directly over a sensor, but may pass between sensors or over multiple sensors. A **rock's sensor signal** is the signal of the rock over a particular sensor. The **sensor signal** combines the rock sensor signals from the rocks passing (partially or wholly) over it, and we explain it more later.

Finally, we model a **sensor reading** as a sample from the sensor signal at discrete times (milliseconds). Sensor reading is the only observed variable, and is taken at 1 millisecond intervals.

Given only the sensor signals of all 7 sensors, a decision must be made to **activate** or not activate each diverter for each millisecond in time. The diverters then guide each rock into either the **keep bin** or the **discard bin**. This action is stochastic. Please note, there are two ways for a good rock

to incorrectly end up in the discard bin (i.e. a false positive). First, imagine a good rock at some location, the model may incorrectly predict that the rock is bad at that location or simply that there are no rocks at this location and then will choose not to activate the diverter nearest this rock resulting in a false positive. Second, the model may correctly predict a good rock at its location and activate the diverter at the correct time but the rock may tumble unexpectedly and fall into the discard bin resulting in a false positive.

It is with regard to the second which we model in the diverter's activate variable. Currently we have trained a simple Bernoulli model for when an activated diverter fails to divert the rock into the keep bin due to the rock tumbling or hitting another rock. Other unexpected results may also occur such as a rock tumbling over toward a diverter it would not otherwise have been hit by, colliding with another rock, or splitting into 2 smaller rocks; However, we did not model these because we deemed the accuracy to be good enough to compare one algorithm relative to another (see Evaluation Section for details).

The **utility** takes into account the number of false positives (bad rocks in the keep bin, FP) and false negatives (good rocks in the discard bin, FN) as well as the cost of false positives ($C(FP)$) and the cost of false negatives ($C(FN)$). To make an appropriate decision, we take into account utility and associated misclassification costs.

We use normalized expected cost ($Norm(E[Cost])$) (Drummond and Holte 2006) as the cost (or loss) function for utility, where smaller values are better.

$Norm(E[Cost])$:

$$\frac{FN \cdot C(FN) + FP \cdot C(FP)}{P(+) \cdot C(FN) + P(-) \cdot C(FP)} \tag{1}$$

Where FP and FN are the counts of bad rocks in the keep bin and good rocks in the discard bin. For example, FN is the number of rocks which ended up in the discard bin but whose true mineral content actually represents a good rock. $P(+)$ and $P(-)$ are the distributions of a good (positive) rock occurring and of a bad (negative) rock occurring respectively.

The costs, $C(FP)$ and $C(FN)$, will depend on various factors outside of the model (e.g. environmental costs, the cost of transportation, and the price of minerals). The ratio between the two is what matters.

As in Cost Curves (Drummond and Holte 2006), misclassification costs (utility) and class distributions are expressed in a single term called **Probability Cost** (PC):

$$PC(+) = \frac{P(+)C(FN)}{P(+)C(FN) + P(-)C(FP)} \tag{2}$$

We discretized the domain of $PC(+)$ into a set of 9 and learned the model for each. Only misclassification costs varies, the class distributions remained constant in our evaluation.

Model Details

The specifics of the relations in the system are described next, please refer to the detailed relational influence diagram in Figure 4. This detailed diagram is just as accurate as the previous, but helps to explain how each variable is connected. Rocks are denoted with an i subscript and sensors are denoted with a j subscript (1 thru 7).

As shown in the detailed diagram, we model rock properties as 4 components: mineral content (c_i), size as a radius (r_i), and position (x_i, y_i). Electromagnetic response is modelled as an un-normalized symmetric 2-D Gaussian function which is broken down into two parameters: width and height. The width (s_i) parameter is a linear function of the rock's size:

$$s_i(r_i) = (ar_i + b)^2 \tag{3}$$

where slope (a) and bias (b) are parameters which are learned. The height (h_i) is a linear function of the rock's mineral content:

$$h_i(c_i) = c_i m \tag{4}$$

where m is a parameter which is learned. The 2-D mean of the 2-D Gaussian is the same as the rock's position so is not modelled separately.

We model a rock sensor signal as an un-normalized 1-D Gaussian function. A 1-D Gaussian typically has 3 parameters: width, height, and mean. We exploit a property of 2-D Gaussians to relate electromagnetic response to rock sensor signal: a cut of a 2-D Gaussian will be a 1-D Gaussian with the same width, and if the cut is parallel to the y-axis, then the mean of the 1-D Gaussian will be equal to the y-position of the 2-D Gaussian's mean. Since the rocks move vertically on the y-axis, all cuts will be parallel to the y-axis.

Therefore, the 1-D Gaussian width is equal to the electromagnetic response's width and the mean is equal to the y-position of the electromagnetic response which is equal to the y-position of the rock. Hence, we reuse the x_i node from rock properties and the width (s_i) node from electromagnetic response. The Gaussian property is used during inference and is the reason we chose to use Gaussians.

The height of the 1-D Gaussian is modelled as the variable height (h_{ij}), shown below:

$$h_{ij}(h_i, x_j, x_i, s_i) = h_i \mathrm{e}^{-1/2 \frac{(x_j - x_i)^2}{s_i}} \tag{5}$$

Where h_{ij} is the height of the rock sensor signal for rock i on sensor j, x_j is the x-position of the j^{th} sensor, x_i is the x-position of rock i, and s_i is the width of the electromagnetic response for rock i.

Lastly, we model sensor signal as a sum of all the rock sensor signals. Since rock sensor signal is a 1-D Gaussian, summing creates a sum of Gaussians.

$$f_j(t) = \sum_{i=1}^{n} h_{ij} \mathrm{e}^{-1/2 \frac{(t - y_i)^2}{s_i}} \tag{6}$$

Where t is continuous time and y_i is rock i's y-position.

Runtime Inference

The problem is resource bounded for a variety of reasons including: the speed of the rocks, the length of the conveyor belt, and communication overhead. In short, we have 400ms to make a decision about which diverters to activate, and we can only observe sensor readings of each sensor. To meet the

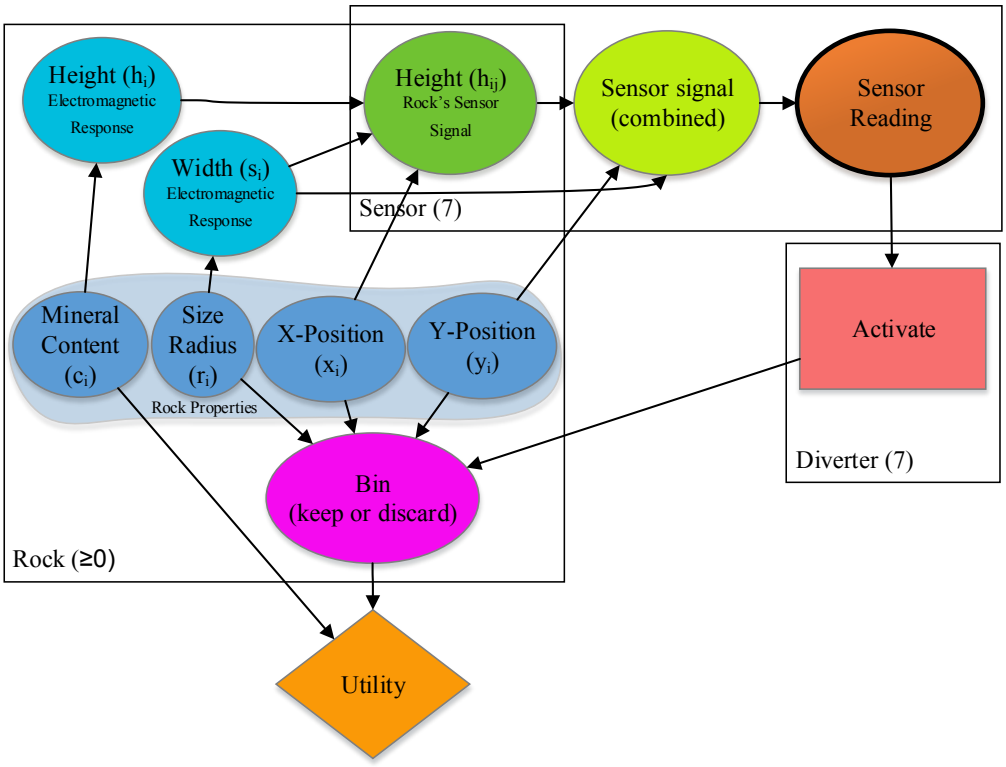

Figure 4: Relational Influence Diagram (Detailed)

400ms deadline, we collect a group (called a **data window**) of 200 sensor readings at a time and run inference on each group independently.

At runtime our algorithm hypothesizes the positions, sizes and mineral contents of the rocks on the conveyor belt within the data window and determines for each diverter whether to activate it.

We have experimented with a number of inference algorithms to determine the positions and compositions of the rocks, including MCMC (Robert and Casella 2009), gradient-based methods (Snyman 2005), and various methods within the SciPy library (Jones et al. 2001). We found that what works the best is a mix between random and coarse exhaustive search.

For each sensor, and within the current data window, we exhaustively search over all values of rock size and rock mineral content.

However, rock size is limited to only 2 possible values and mineral content is limited to 11 values including 0. By including 0 we allow for an unknown number of rocks to be predicted, because a rock with 0 mineral content is effectively removing the rock (we treat non-valuable rocks and non-existent rocks the same). The space of a rock's position (x, y) is continuous; We randomly choose values for x and y, typically performing around 100 samples (we sample as many as we have time for within the time limit).

For each sample in the search, we generate the sensor signal due to the rocks and compare it to the observed sensor readings. Sampling repeats until time runs out, at which point the best hypothesis is chosen and decisions to activate are made.

Offline Learning

For offline learning we recorded sensor data and manually labelled the rocks and their properties. Generating data to use offline is expensive and time-consuming.

Using the recorded sensor data offline, we automatically configure our model's parameters using a state-of-the-art algorithm configuration method called SMAC (Hutter, Hoos, and Leyton-Brown 2011). SMAC searches over the space of parameters optimizing a given cost function — in our case we optimize for utility. We trained on 3/4 of our labelled data and held the remaining 1/4 aside as the test set.

For each $PC(+)$ (i.e. misclassification cost) we ran SMAC for 4 hours on 8 2.67-GHz Xeon x5550 cores on WestGrid's Hermes cluster.

Evaluation

The previously used sorting algorithm, which we call VBSA, looks at the sensor readings of each sensor, and when the magnitude of the reading passes some threshold it activates the diverter corresponding to the sensor. This sorting

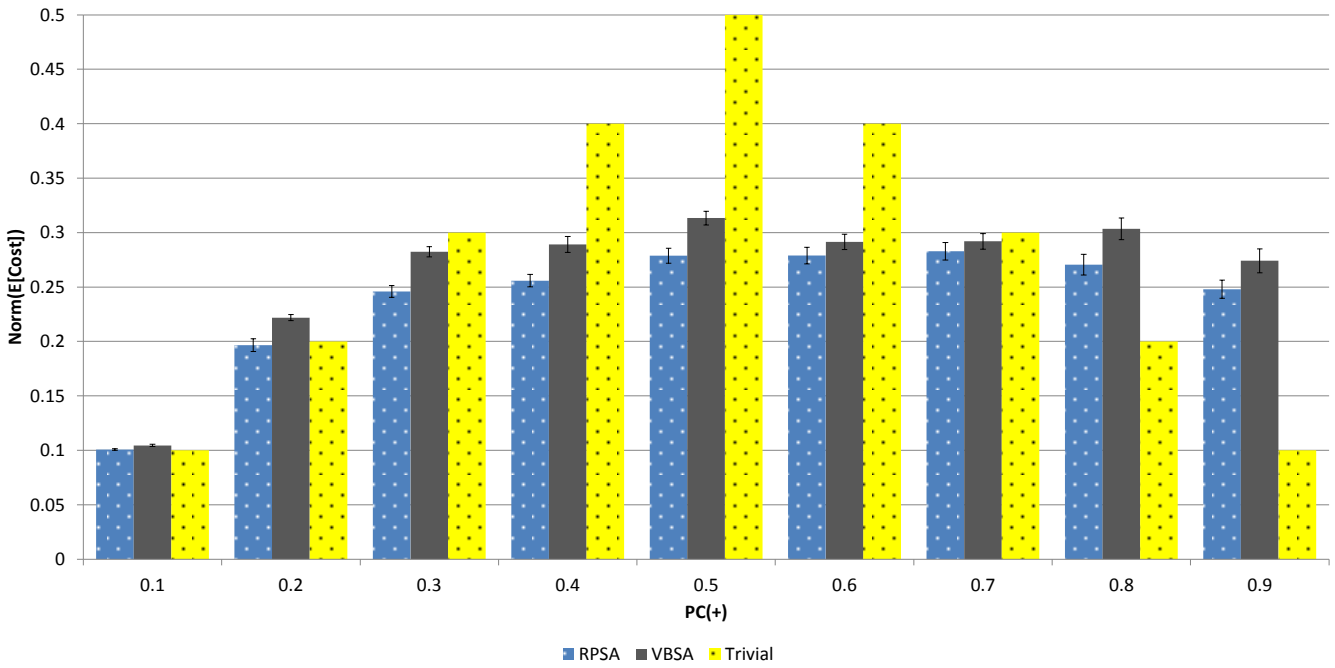

Figure 5: Comparison of two sorting algorithms using the learned parameters for each algorithm for each $PC(+)$ (representing misclassification costs), where lower $Norm(E[Cost])$ is better.

algorithm made no attempt to infer rocks or their properties, but does work surprisingly well given its simplicity.

We ran both the previously used algorithm, VBSA, and our new algorithm, RPSA, on the test set. The sorting decisions outputted from VBSA is also fed through the same diverter and utility model as ours - for the purpose of comparing two algorithms, the diverter model does not have to be perfect for a relative comparison.

Results can be seen in Figure 5. The '"trivial" algorithm shows the cost of rejecting (for $PC(+) < 0.5$) or accepting (for $PC(+) > 0.5$) all the rocks all the time. At $PC(+)$ of 0.5 one can accept or reject all or randomly sort in order to achieve the trivial cost. We show improvement using our relational influence diagram based sorting algorithm versus VBSA across the 9 values of $PC(+)$, while still being able to make decisions online within the real-time constraint.

Our algorithm, RPSA, improves on VBSA by 9% on average across the 9 values of $PC(+)$ and is statistically significant for where $PC(+)$ is 0.5 and lower. Error bars represent standard error of the mean.

Future Work

Future challenges are to allow us to have a more detailed and accurate model. More specifically, a better model of rock properties; Including a model of various mineral types (one type per rock) as well as heterogeneous rocks (multiple mineral types per rock), and rocks of odd shapes and sizes (removing the assumption that rocks are circles). We would also like to consider alternate models for the sensor signal, other than Gaussians.

Furthermore, we would like to be able to draw an influence diagram and automatically perform inference and learning on the model, enabling us to quickly try new changes to the model and evaluate the results.

Conclusion

Sorting rocks using electromagnetic sensors is a challenging real-world planning problem where there is an unknown number of objects and we have to act in real-time. We created a new sorting algorithm, RPSA, by modeling the problem using a relational influence diagram and automatically configured its parameters offline. RPSA does online inference and beats the previously used algorithm, VBSA.

References

Buntine, W. L. 1994. Operations for learning with graphical models. *Journal of Artificial Intelligence Research* 2:159–225.

Das, Y.; McFee, J. E.; Toews, J.; and Stuart, G. C. 1990. Analysis of an electromagnetic induction detector for real-time location of buried objects.

Drummond, C., and Holte, R. C. 2006. Cost curves: An improved method for visualizing classifier performance. *Machine Learning* 65(1):95–130.

Hutter, F.; Hoos, H. H.; and Leyton-Brown, K. 2011. Sequential model-based optimization for general algorithm configuration. In *Proceedings of the 5th International Conference on Learning and Intelligent Optimization*, LION'05, 507–523. Berlin, Heidelberg: Springer-Verlag.

Jones, E.; Oliphant, T.; Peterson, P.; et al. 2001–. SciPy: Open source scientific tools for Python.

Passeraub, P. A.; Besse, P.-A.; de Raad, C.; Dezuari, O.; Quinet, F.; and Popovic, R. S. 1997. Coin recognition using an inductive proximity sensor microsystem. In *In Proceedings of International Conference on Solid State Sensors and Actuators*, volume 1, 389–392.

Robert, C., and Casella, G. 2009. A short history of Markov chain monte carlo: Subjective recollections from incomplete data.

Shachter, R. D. 1988. Probabilistic inference and influence diagrams. *Operations Research* 36(4):589–604.

Snyman, J. A. 2005. *Practical mathematical optimization : an introduction to basic optimization theory and classical and new gradient-based algorithms*. Applied optimization. New York: Springer.

Zilberstein, S. 1996. Using anytime algorithms in intelligent systems. *AI Magazine* 17(3):73–83.