# A Bayesian Network Based Solution Scheme for the Constrained Stochastic On-line Equi-Partitioning Problem*

Sondre Glimsdal† and Ole-Christoffer Granmo‡

**Abstract**

A number of intriguing decision scenarios revolve around partitioning a collection of objects to optimize some application specific objective function. This problem is generally referred to as the Object Partitioning Problem (OPP) and is known to be NP-hard. We here consider a particularly challenging version of OPP, namely, the Stochastic On-line Equi-Partitioning Problem (SO-EPP). In SO-EPP, the target partitioning is unknown and has to be inferred purely from observing an on-line sequence of object pairs. The paired objects belong to the same partition with probability $p$ and to different partitions with probability $1-p$, with $p$ also being unknown. As an additional complication, the partitions are required to be of equal cardinality. Previously, only heuristic sub-optimal solution strategies have been proposed for SO- EPP. In this paper, we propose the first *Bayesian* solution strategy. In brief, the scheme that we propose, BN-EPP, is founded on a Bayesian network representation of SO-EPP problems. Based on probabilistic reasoning, we are not only able to infer the underlying object partitioning with superior accuracy. We are also able to simultaneously infer $p$, allowing us to accelerate learning as object pairs arrive. Furthermore, our scheme is the first to support a wide range of constraints on the partitioning (Constrained SO-EPP). Being Bayesian, BN-EPP provides superior performance compared to existing solution schemes. We additionally introduce Walk-BN-EPP, a novel WalkSAT inspired algorithm for solving large scale BN-EPP problems. Finally, we provide a BN-EPP based solution to the problem of order picking, a representative real-life application of BN-EPP.

## 1 Introduction

A number of intriguing decision scenarios revolve around grouping a collection of objects into partitions in such a manner that some application specific objective function is optimized. This type of grouping is referred to as the Object Partitioning Problem (OPP) and is in its general form known to be NP-hard.

*A preliminary version of parts of this paper was presented at ICMLA 2014 - the 13th International Conference on Machine Learning and Applications, Detroit, USA, December 2014.

†This author can be contacted at: Centre for Artificial Intelligence Research (CAIR), University of Agder, Postbox 422, 4604 Kristiansand, Norway. E-mail: sondre.glimsdal@uia.no.

‡Author's status: *Professor*. This author can be contacted at: Centre for Artificial Intelligence Research (CAIR), University of Agder, Postbox 422, 4604 Kristiansand, Norway. E-mail: ole.granmo@uia.no.

In this paper, we consider a particularly challenging variant of OPPs — the Constrained Stochastic Online Equi-Partitioning Problem (CSO-EPP). In CSO-EPP, objects arrive sequentially, in pairs[1]. Furthermore, the relationship between the arriving objects is stochastic: Paired objects belong to the same partition with probability $p$, and to different ones with probability $1-p$. As an additional complication, the partitioning is constrained, with the default constraint being that the partitions must be of equal cardinality, referred to as equi-partitioning. Unlike previous work, we relax this constraint and only require the size of each partition to be known beforehand. Under these challenging conditions, the overarching goal is to infer the underlying partitioning, that is, to predict which objects will appear together in future arrivals, from a history of object arrivals.

The CSO-EPP can be applied to solve a number of challenging tasks. We will here study a particularly fascinating one, *order picking*, which highlights the full spectrum of nuisances captured by CSO-EPP. Order picking is defined as *"the process of retrieving products from storage (or buffer areas) in response to a specific customer request"* [1]. Order picking occurs both in warehouses employing an Automated Storage/Retrieval System (AS/RS) and those depending on manual labor. Tompkins et al. identified travel time as the main factor when it comes to optimizing order-picking [2]. For this reason, to facilitate efficient retrieval of products, frequently ordered products should be placed in easy to reach locations. Additionally, products that are often ordered together should be placed in near-proximity of each other. By doing so, we can systematically reduce the total travel time needed to collect orders.

In more challenging order-picking scenarios, the governing product relationships may be unknown initially, and thus have to be learned over time by monitoring which products are ordered together. Additionally, non-related products may sporadically be ordered in conjunction, leading to *stochastic* order composition. This means that successful solution strategies must be able to operate in a stochastic environment. Furthermore, many order picking scenarios impose constraints when it comes to product placement. One could for instance require that a subset of the objects is located in a subset of the available locations, e.g., that all frozen objects should be in freezers, even when they are rarely purchased together. Other constraints could be that all products from a brand must be co-located on the request of the manufacturer, or that fragile objects must be placed in shelves close to the floor. To further exemplify the importance of dealing with constraints, several more are listed in Table 1[2]. Noting that each section of a warehouse can be represented as a CSO-EPP partition, and that products can be represented as CSO-EPP objects, we propose CSO-EPP as a model for order picking.

In this paper, we present the first *Bayesian* solution scheme for SO-EPP and CSO-EPP. Let $\mathcal{O} = \{O_1, O_2, \ldots, O_w\}$ be a set of $W$ objects. These are to be partitioned into $R$ different partitions $\mathcal{P} = \{P_1, P_2, \ldots, P_R\}$. The aim is to find some unknown underlying partitioning of the objects based on noisy observations. Succinctly, the problem can be described as a 2-tuple $(\mathcal{U}, p)$, where $\mathcal{U}$ is a set of tuples $(O_i, O_j)$. If $(O_k, O_m) \in \mathcal{U}$

[1]Note that the arrival of objects in *pairs* can easily be generalized to arrival of objects in *sets*, which in turn are transformed into pairwise combinations of the objects contained in each set. See Section 5.4 for an example of this.

[2]These are based on real-world point-of-sale transaction data from a grocery outlet [3].

Table 1: Example constraints governing the placement of products in a warehouse.

| Number | Products | Constraint |
|---|---|---|
| 1 | shopping bags | Must either be in the entrance- or counter section |
| 2 | whole milk, rolls/buns, tropical fruit | Cannot be in the same section |
| 3 | white wine, specialty chocolate | Must be in the same section |
| 4 | yogurt | Has to be in the cooler section |
| 5 | tropical fruit | Cannot be in the cooler section |

then object $k$ and $m$ belong to the same underlying partition, otherwise, they belong to different ones. Constraints can then naturally be formulated in terms of: (1) the cardinality of each partition; (2) what objects must be, or must not be, in the same partition; and (3) which subset of objects must be in which subset of partitions. The two latter types of constraints can be expressed by formulating restrictions on object pairs in $\mathcal{U}$, while the first type of constraint can be specified as a cardinality vector of size $R$. Finally, $p$ is the probability of a *convergent request* [4], i.e., the probability that a request (i.e., an observation) encompasses two objects from the same underlying partition. A request where the objects originate from different underlying partitions is called a *divergent request* [4], which occurs with probability $1 - p$.

Under the above model, an observation can be simulated by sampling from a Bernoulli distribution. With probability $p$, select a pair of objects randomly from $U$: $(O_i, O_j) \in \mathcal{U}, i \neq j$ (a *convergent* request). And with probability $1 - p$, randomly select a pair of objects not in $U$: $(O_k, O_m) \notin \mathcal{U}, k \neq m$ (a *divergent* request). This definition is equivalent to the definition given by Oommen et al. [5].

For completeness, we mention that solutions to SO-EPP are invariant to permutation of the partitions, as long as the objects grouped together inside each partition remain unchanged.

Previously, only heuristic sub-optimal solution strategies have been proposed for SO-EPP, and no solution exists for CSO-EPP. In this paper, for both of these problems, we propose the first *Bayesian* solution strategy. The solution strategy is based on a novel Bayesian network representation of CSO-EPP problems. To enable swifter computations with BN-EPP, we additionally introduce Walk-BN-EPP, an approximate reasoning approach that takes advantage of the unique structure of BN-EPP. The paper contribution can be summarized as follows:

1. We propose a novel Bayesian network model of the CSO-EPP problem (BN-EPP) that fully captures the nuances of CSO-EPP.

2. We provide a BN-EPP based algorithm for on-line object partitioning that outperforms the existing *state-of-the-art* SO-EPP solution schemes.

3. The BN-EPP scheme is highly flexible in the sense that we can encode a wide range of partitioning constraints, leveraging the representation capacity of Bayesian networks.

4. BN-EPP is parameter-free, which means that performance is maximized without any fine tuning of parameters.

5. In addition to predicting the correct partitioning of objects, BN-EPP also estimates the noise parameter $p$ on-line.

6. We demonstrate that Walk-BN-EPP exhibits state-of-the-art performance on a large-scale real-world warehouse order picking problem.

7. We define a novel scheme that allows us to apply Spectral Clustering (SC) to the SO-EPP problem, demonstrating performance close to BN-EPPs state-of-the-art performance on artificial data.

The paper is organized as follows. In Sect. 2 we present related work. We then provide a brief overview of Bayesian networks in Sect. 3, before we proceed with providing the details of our BN-EPP scheme in Sect. 4. Then, in Sect. 5, we present our empirical results and demonstrate the superiority of BN-EPP when compared to existing state-of-the-art schemes. We conclude in Sect. 6 and provide pointers for further work.

## 2 Related Work

, The OPP is already a thoroughly studied problem [6, 7]. Yet, research on its fascinating variant, SO-EPP [8, 9, 10, 5, 11], is surprisingly sparse despite its many real-world applications, which includes software clustering [12] and keyboard layout optimization [13]. To cast further light on the unique properties of SO-EPP, we will here relate it to two similar problems, namely, the *Poset Ordering Problem* (POP) and the *Graph Partitioning Problem* (GPP).

**The Poset Ordering Problem (POP).** A *poset* is defined as a set of elements with a transitive partial order, where some elements may be incomparable [14]. A binary relation that is reflexive, antisymmetric, and transitive defines this ordering, referred to as a *less-than-or-equal* relation ($\leq$). The standard *less-than-or-equal* relation for integers forms for instance a partial ordering on the set of integers. In the poset ordering problem, the goal is to establish the partial ordering of a poset by comparing pairs of elements, typically using the *less-than-or-equal* relation as few times as possible. Accordingly, both in SO-EPP and POP, one must learn from paired elements to uncover an underlying more complex structure. That is, in POP, the *less-than-or-equal* relation is applied iteratively on pairs of elements, while in SO-EPP a *in-the-same-partition* relation is used instead. Whereas the *less-than-or-equal* relation found in POP is both reflexive and transitive, it is not symmetric, i.e., $A \leq B$ does not imply $B \leq A$. The *in-the-same-partition* relation, however, is symmetric. This means that the solution of SO-EPP is not a partial ordering, but a set of *equivalence classes*, leading to unique solution schemes.

**The Graph Partitioning Problem (GPP) and Spectral Clustering (SC).** The GPP is in its most general form an NP-complete problem [15]: Let $G = (V, E)$ be a graph with a set of vertices $V$ and a set of weighted edges $E$. In graph equipartitioning, the goal is to partition $V$ into $k$ subsets $V_1, V_2, \ldots, V_k$ of equal cardinality. In all brevity, the solution to a GPP instance is the partitioning that minimizes the sum of those edge weights that cross different vertex sets, $(V_i, V_j), i \neq j$ [16]. The SO-EPP can thus be cast as a GPP if

the frequencies of object co-occurrence are known for all object pairs. Then we could form a complete graph, $G = (V, E)$, where each vertex in $V$ represents an object. Further, the weight of an edge between a pair of objects is simply the frequency with which we observe that particular pair. The resulting GPP can then be solved by any GPP solver [17, 18, 19].

Another approach to solving GPPs is based on a Markov Random Walk. By defining a Markov Random Walk over $G$, one can perform clustering based on the eigenvalues of the resulting transition matrix [20]. This method is based on the usage of query statistics [11], e.g. to generate the transition matrix from the different frequency counts. The baseline for this approach is Spectral Clustering (SC) where the eigenvalues is used as a low-dimensional embedding of the problem space. SC can then effectively generate clusters using the MultiClass Normalized Cuts scheme [21]. In Section 5.2 we define a simple, yet effective scheme that allow us to apply SC to SO-EPP.

A main drawback of SC and other GPP solvers is that they do not support the type of real-world probabilistic constraints mentioned in the introduction (CSO-EPP), and as we shall see, do not either fully utilize of the problem specific characteristics of SO-EPP.

**State-of-the-art solution schemes for SO-EPP.** We now turn our attention to algorithms that are specifically designed to solve SO-EPP. The state-of-art solution scheme for SO-EPP is the Pursuit Object Migration Automaton (POMA), introduced by Shirvani et al. in 2017 [8]. POMA is based on the Object Migration Automaton (OMA) [5, 4]. The basic OMA is a statistics free scheme, meaning that it does not try to estimate object co-occurrence frequencies. Instead, each object navigates a finite state machine according to a few simple fixed rules, allowing the objects to *migrate* between the different partitions, gradually converging to a solution. POMA, on the other hand, leverages co-occurrence frequency estimates, through the following two phases:

1. An *estimation phase* adopts the previous state-of-art Object Migration Automaton (OMA) [5, 4] to generate an initial solution, while simultaneously estimating the pairwise-object frequencies.

2. A *fine-tuning phase* refines the initial solution by making use of the pursuit paradigm [22, 23] to filter diverging or noisy queries. Thus, the POMA is able to determine whether a pairwise query facilitates convergence.Theoretically, this would allow the underlying OMA to operate in a noise free environment, and, consequently, converge quickly.

However, POMA is still a heuristic rule based approach. While efficient, it is not optimal, which leads us to design the BN-EPP algorithm presented in this paper. BN-EPP is a probabilistic parameter-free algorithm that, as we shall see, is not only more flexible in terms of the requirements placed on the solution, but also able to infer the level of noise present in the environment.

# 3 A Bayesian Network Based Solution Scheme for the Constrained Stochastic On-line Equi-Partitioning Problem

In this section, we present our novel BN-EPP scheme — a generative modeling approach for solving CSO-EPP based on Bayesian networks (BNs). By taking advantage of the ability of BNs to construct interpretable models that encode probability distributions over complex domains [24], we capture the unique characteristics of CSO-EPP. We further propose an efficient reasoning algorithm for BN-EPP that allows uncertainty to be represented and managed explicitly.

A BN consists of a directed acyclic graph (DAG) representing the conditional dependencies between a set of random variables. When modeling causal relationships, an edge between the nodes A and B signifies that A *"causes"* B. Consider the BN shown in Figure 1. In this simple BN, we have three discrete random variables: *Weather*, *Sprinkler* and *Lawn*. Let us assume that the weather can have one of three different states: *Sunny*, *Cloudy*, or *Rainy*. Further, the lawn is either *Wet* or *Dry*, and the sprinkler can be *On* or *Off*. Adding directed edges, we can encode knowledge about cause and effect, such as the fact that rainy weather causes the lawn to be wet. Similarly, a long period of sunny weather triggers a need for turning the sprinkler on, hence weather indirectly causes the lawn to be wet though the sprinkler system.

The above qualitative description of cause and effect is further enriched with a quantitative description. The quantitative description takes the form of a probability distribution assigned to each node, conditioned on the state of the parents of the respective node. The purpose of the conditional probability distributions is to quantitatively describe the probabilistic independence relationships captured by the DAG. We assign these probabilities though Conditional Probability Tables (CPTs), one for each node in the graph. Note that a node without parents is assigned an unconditional probability distribution. A CPT for the sprinkler can be seen in Table 2, where the effect weather has on the state of the sprinkler is captured. The CPT here tells us, e.g., that the sprinkler turns on with probability 0.8 in sunny weather.

From the BN CPTs, we can conduct diagnostic and predictive reasoning, simply by asking questions about the state of the random variables. One could for instance ask: "if the lawn is wet, what are the chances that it was caused by rain or by the sprinkler?" or "if the sprinkler is on, does that indicate that there is sun outside?".

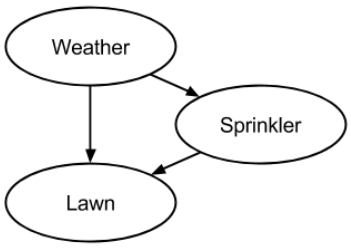


Figure 1: Simple BN.

| Weather – State | Sunny | Cloudy | Rainy |
|---|---|---|---|
| $P$(Sprinkler = on\|Weather) | 0.8 | 0.15 | 0.05 |
| $P$(Sprinkler = off\|Weather) | 0.2 | 0.85 | 0.95 |

Table 2: CPT of Sprinkler

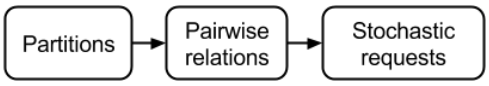


Figure 2: Overview of BN-EPP.

The generative model that we propose, BN-EPP, can be described in terms of three interacting BN fragments, as shown in Figure 2. Firstly, a dedicated BN fragment, referred to as "Partitions" in the figure, captures the actual placement of objects into partitions. This includes any constraints on the partitioning, such as equi-partitioning. Since the objects arrive in pairs, we need to further generate an intermediate BN fragment — the "Pairwise relations" fragment — that explicitly extracts all pairwise object relations from the "Partitions" fragment. Finally, an observation model is derived from "Pairwise relations", capturing generation of convergent and divergent request. This latter fragment, "Stochastic requests", is based on the noise parameter $p$ and the "Pairwise relations" fragment.

Based on the BN-EPP, our on-line solution strategy for CSO-EPP can be summarized as follows. In operation, arriving object pairs (requests) are entered into the "Stochastic requests" part of the BN-EPP as observations (evidence). From these observations, pairwise relations are inferred in the intermediate fragment, which, finally, leads to a probability distribution over allowed partitions of objects in the "Partitions" fragment. Every object pair observed provides new information, and gradually, with successive observations, the probability distribution over object partitions converges to a single partitioning that solves the underlying CSO-EPP. In the case of multiple equally probable solutions, BN-EPP will arbitrarily select a single solution.

The detailed construction of BN-EPP is outlined in Algorithm 1. The BN-EPP needs to:

**Requirement 1** Handle constraints, such as only considering partitions of equal cardinality.

**Requirement 2** Infer whether two objects belong to the same partition.

**Requirement 3** Correctly handle both converging and diverging requests.

**Requirement 4** Encode the actual object partitioning.

We will now explain how the BN-EPP algorithm fulfills the above requirement. First of all, recall that $\mathcal{O} = \{O_1, O_2, \ldots, O_w\}$ is a set of $W$ objects. These are to be partitioned into $R$ different partitions $\mathcal{P} = \{P_1, P_2, \ldots, P_R\}$. The aim is to find some unknown underlying partitioning of the objects based on noisy observations of object pairs (convergent and divergent requests).

**(Requirement 1) Only consider partitions that fulfill governing constraints (Lines 1-5)**
The first part of the algorithm builds the "Partitions fragment" from Figure 2. Briefly stated, we represent

**Algorithm 1:** Constructing BN-EPP

**Data:** Objects $\mathcal{O} = \{O_1, O_2, \ldots, O_w\}$; Partitions $\mathcal{P} = \{P_1, P_2, \ldots, P_R\}$; and Noise resolution $N$
**Result:** A BN-EPP model $\beta$: Noise $p_\beta$; Partitions $\mathcal{O}_\beta$; Pairwise relations $\mathcal{A}_\beta$; Stochastic requests $\mathcal{X}_\beta$

`/* Create partitions fragment` $\mathcal{O}_\beta$`. */`
$\mathcal{O}_\beta := \emptyset$
**for** $i := 1$ *to* $W$ **do**
  `/*` $O_{\beta_i}$ `assigned to a partition in` $\mathcal{P}$`, given preceding assignments` $\mathcal{O}_\beta$ `*/`
  `/* and the CPT of object` $i$`:` $F_{O_{\beta i}}$ `*/`
  $O_{\beta_i} :=$ Node(States=$[P_1, P_2, \ldots, P_R]$, Parents=$\mathcal{O}_\beta$, Distr=$F_{O_{\beta i}}$)
  $\mathcal{O}_\beta := \mathcal{O}_\beta \cup O_{\beta_i}$
**end**
`/* Create pairwise relations fragment. */`
$\mathcal{A}_\beta := \emptyset$
**for** $i, j \in [W \times W]$ *s.t.* $i < j$ **do**
  `/*` $A_{\beta_{ij}}$ `is true if and only if` $O_{\beta_i}$ `and` $O_{\beta_j}$ `are in the same partition. */`
  $A_{\beta_{ij}} :=$ Node(States=[True, False], Parents=$\{O_{\beta_i}, O_{\beta_j}\}$, Distr=$F_{A_\beta}$)
  $\mathcal{A}_\beta := \mathcal{A}_\beta \cup \{A_{\beta_{ij}}\}$
**end**
`/* Create stochastic requests fragment. */`
$p_\beta :=$ Node(States=$[\frac{0}{N}, \frac{1}{N}, \ldots, \frac{N}{N}]$, Parents=$\emptyset$, Distr=$\mathrm{F}_{p_\beta}$) `// Noise probability.`
$\mathcal{X}_\beta := \emptyset$
**for** $i, j \in [W \times W]$ *s.t.* $i < j$ **do**
  $X_{\beta_{ij}} :=$ Node(States=$[\mathbb{N}_0]$, Parents=$\{A_{\beta_{ij}}, p_\beta\}$, Distr=$F_{X_\beta}$) `// Pair observation count.`
  $\mathcal{X}_\beta := \mathcal{X}_\beta \cup \{X_{\beta_{ij}}\}$
**end**

each EPP object, $O_i \in \mathcal{O}$, using a corresponding BN node, $O_{\beta i}$. Each BN node, $O_{\beta i} \in \mathcal{O}_\beta$, has one state per partition, $P_i \in \{P_1, \ldots, P_R\}$, representing the partition assigned to $O_i$. For instance, if we have two partitions then there will be two states per object, one for partition $P_1$ and one for partition $P_2$.

We now model the governing constraints, including equal cardinality of partitions, by means of the BN DAG. Because of the reciprocal relationships among objects (objects are either in the same partition or not), we can order the BN object nodes arbitrarily. Without loss of generality, assume that A is the first BN node in the ordering. This means that A can be freely placed in any partition (the placement does not depend on the placement of any other object, because none of the other objects have been placed yet). Then the next object in the ordering, object B, only needs to take into account object A's choice of partition. Likewise object C, the third object, is only restricted by the previous objects' choice of partition (the choices of object A and B). Continuing in this manner, we can always represent the partition of the next object as solely being dependent on the already partitioned objects. It is for this purpose we maintain the gradually increasing object set $\mathcal{O}_\beta$, containing all the already partitioned predecessor objects. This organization of objects is thus leading to a BN DAG structure, as exemplified in Figure 3, capturing two partitions and four objects.

The corresponding CPTs for the EPP objects ($F_{O_{\beta i}}$ in the algorithm) are generated as a function of the constraints set by CSO-EPP (the constraints governing the partitioning, e.g., equi-partitioning). As an example, the CPT of object C (the third object) can be seen in Table 3. From the table we observe for

| A - State | $P_1$ | | $P_2$ | |
|---|---|---|---|---|
| B - State | $P_1$ | $P_2$ | $P_1$ | $P_2$ |
| C in $P_1$ | 0.0 | 0.5 | 0.5 | 1.0 |
| C in $P_2$ | 1.0 | 0.5 | 0.5 | 0.0 |

Table 3: CPT of object C

instance that $P(C = P_1|A = P_1, B = P_1) = 0.0$, that is, if object A and B is in $P_1$ then the probability of C being in $P_1$ is zero. On the other hand, if object A and B is located in different partitions then object C is equally likely to be in partition $P_1$ as in partition $P_2$. Thus, by constructing the CPT of each node (representing an object) in this manner, a solution that fulfills all of the constraints is always ensured because a partitioning that violates constraints is assigned a probability of zero.

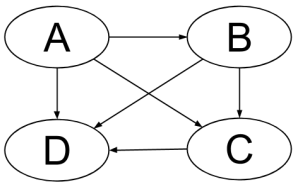


Figure 3: Object dependencies for 4 objects with 2 partitions.

**(Requirement 2) Infer pairwise relations between the objects (Lines 6-10)**
Now that the "Partitions fragment" has determined the partition of each object, it is a simple task to determine whether an object pair belongs to the same partition. In the "Pairwise relations" fragment, we represent every pair of objects as a deterministic node with two states: *True* if the pair is in the same partition, and *False* when they are not (distribution $F_{A_\beta}$ in the algorithm).

Figure 4 provides an example of a "Pairwise relations" fragment, obtained following the above procedure for four objects and two partitions. The corresponding CPT for the pair node for object A and object C (node AC) can be found in Table 4. From the truth table it is evident that if object A and C belong to the same partition, the state of node AC state is *True*, and *False* otherwise.

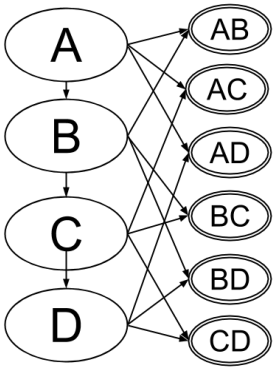


Figure 4: Pairwise object relations for 4 objects with 2 partitions.

| A - State | $P_1$ | | $P_2$ | |
|---|---|---|---|---|
| C - State | $P_1$ | $P_2$ | $P_1$ | $P_2$ |
| AC - State | True | False | False | True |

Table 4: Truth-table for node AC

Table 5: The CPT of an observation node conditioned on the state of the parent pair $A_{\beta_{ij}}$ and the $p_\beta$ node.

| | |
|---|---|
| $P(X_{\beta ij} = n \mid A_{\beta_{ij}} = \text{True}, p_\beta = p)$ | $B\left(n, p \cdot \frac{1}{P} \cdot \binom{\frac{W}{P}}{2} \frac{1}{\frac{W}{P}} \cdot \frac{1}{\frac{W}{P}-1}\right)$ |
| $P(X_{\beta ij} = n \mid A_{\beta_{ij}} = \text{False}, p_\beta = p)$ | $B\left(n, (1-p) \cdot \binom{P}{2} \frac{1}{P} \cdot \frac{1}{P-1} \cdot \frac{1}{\frac{W}{P}} \cdot \frac{1}{\frac{W}{P}}\right)$ |

The "Pairwise relations" fragment gives BN-EPP the capability to infer object relations from pairwise observations, such as in the following scenario: Given the above example, assume that we know that (1) Object A is known to be in partition $P_1$, and (2) Object B and object D should be in the same *unknown* partition, i.e., the BD-node is set to *True*. BN-EPP will then correctly infer the only possible partitioning, namely the two partitions $P_1 : \{A, C\}$ and $P_2 : \{B, D\}$. While similar result could have been obtained though the usage of a propositional logic solver, as we shall see, the stochastic nature of CSO-EPP rules out such a solution.

**(Requirement 3) Stochastic requests (Lines 11-15)**

The BN model obtained through the "Partitions"- and "Pairwise relations" fragments allows us to infer the correct object partitions, given that we know the state of a sufficient number of the pairwise relation nodes. However, the CSO-EPP involves both convergent and divergent requests. Consequently, we need a mechanism for handling noisy information.

Firstly, we introduce a BN node $p_\beta$ representing $p$ — the convergent request probability. The state space of $p$ is a discretization of potential values for $p$, each with an equal prior probability. Attached to this $p_\beta$ node is a series of *observation* nodes $X_{\beta_{ij}} \in \mathcal{X}_\beta$, each dependent on the state of the $p_\beta$ node, and whether or not its corresponding pair node $A_{\beta ij}$ is *True* or *False*. The CPT for each observation node ($F_{X_\beta}$ in the algorithm) is a function of the number times $n \in \mathbb{N}_0$ that particular pair has been observed, as well as the states of the $p_\beta$ node, as shown in Table 5. As seen, $F_{X_\beta}$ is distributed according to a Bernoulli distribution, $B(n, p)$.

**(Requirement 4) Decode the object partitioning from the BN representation**

While the BN correctly models the CSO-EPP, it does not directly present us with a solution in the form of a partition for each object. However, we obtain the partitioning indirectly by finding the Maximum a Posteriori (MAP[3]) configuration of the BN-EPP. In all brevity, a MAP query identifies the most probable solution given the observations [25, 24].

$$\text{solution(BN)} = \text{MAP}(\mathcal{O}_\beta \cup \mathcal{A}_\beta \cup p_\beta | \mathcal{X}_\beta) = \underset{\mathcal{O}_\beta \cup \mathcal{A}_\beta \cup p_\beta}{\arg\max}\ P(\mathcal{O}_\beta \cup \mathcal{A}_\beta \cup p_\beta | \mathcal{X}_\beta)$$

[3]Also known as Most Probable Explanation (MPE).

For an example of the outcome of the final step, see Figure 5. Note that the observation nodes for an object pair $XY$ in the figure is denoted by $O(XY)$. The complete BN-EPP for four objects and two partitions is shown, ready for MAP inference. As can be seen, the resulting BN-EPP has a complex structure. In the next section we take advantage of this structure to propose an efficient and novel inference algorithm for large scale CSO-EPP problems.

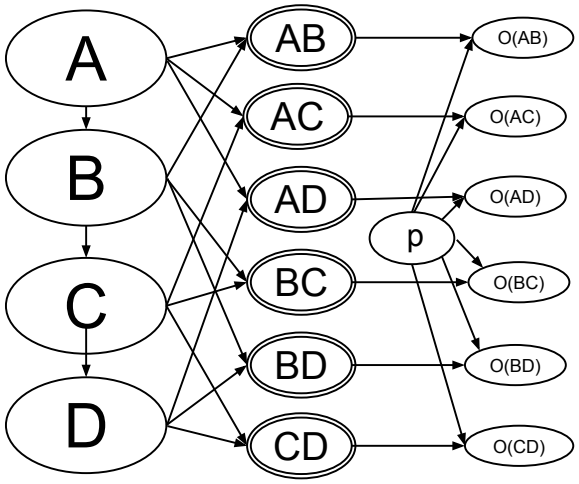


Figure 5: BN for solving an EPP with 4 objects, 2 partitions, and Binomially distributed observation nodes.

Note that the BN-EPP solution strategy is related to the Thompson Sampling (TS) principle that was introduced by Thompson in 1933 [26], and forms the basis for several of the leading solution schemes for so-called Multi-Armed Bandit (MAB) Problem. The classical MAB problem is a sequential resource allocation problem. At each time step, one pulls one out of multiple available bandit arms. Each arm pulled provides a reward with a certain probability, and the objective is to maximize the total number of rewards obtained through the sequence of arms pulled [27, 28]. In the Learning Automata (LA) literature this scheme is referred to as Bayesian Learning Automata (BLA) [28].

In TS, to quickly shift from exploring reward probabilities to reward maximization, one recursively estimates the reward probability of each arm using a Bayesian filter. To determine which arm to play, one obtains a reward probability sample from each arm, and the arm that provides the highest value is pulled. The selected arm triggers a reward, which in turn is used to perform a Bayesian update of the arm's reward probability estimate. As a result, TS selects arms with a frequency proportional to the posterior probability that the arm is optimal.[28]

TS has turned out to be among the top performers for traditional MAB problems [28, 29], supported by theoretical regret bounds [30, 31]. It has also been been successfully applied to contextual MAB problems [32], Gaussian Process optimization [33], Distributed Quality of Service Control in Wireless Networks [34], Cognitive Radio Optimization [35], as well as a foundation for solving the Maximum a Posteriori Estimation problem [36].

# 4 Walk-BN-EPP

The MAP problem is NP-complete [24], and thus cannot be solved efficiently for large networks in general. Accordingly, to allow solutions to be found for large CSO-EPPs, we will in this section introduce a novel inference scheme, Walk-BN-EPP. Walk-BN-EPP is designed to take advantage of the particular characteristics of the BN-EPP DAG structure and is based on WalkSAT [37], a well-known and effective solver for the NP-complete Boolean satisfiability (SAT) problem.

Note that our decision to design a dedicated algorithm for BN-EPP does not mean that existing general MAP solvers, such as Variable Elimination, Belief Propagation and the various evolutionary algorithms [38], cannot be used. On the contrary, they work quite well on small- and medium sized CSO-EPPs. However, since they do not take advantage of the BN-EPP's unique structure, they scale poorly. Thus, by introducing Walk-BN-EPP we expand the class of problems that can be solved with BN-EPP.

Walk-BN-EPP is based on WalkSAT [37], a successful algorithm for solving the NP-complete Boolean satisfiability (SAT) problem [39]. In all brevity, in SAT the goal is to find a truth value assignment for the variables of a Boolean expression that makes the overall expression evaluate to "True", thus *satisfying* the expression. The Boolean expression is a propositional logic formula that consists of a conjunction of Boolean clauses. The overall strategy of WalkSAT can be summarized as follows. One repeatedly selects one of the Boolean variables randomly, negate its value, and then observe whether the new truth value increases the total number of Boolean clauses satisfied. If the number of satisfied clauses does not increase, then with high probability one reverts the negated Boolean variable to its original state. Otherwise, the new state is kept. This simple iterative procedure is repeated until all the clauses are satisfied. Hence, one could say that WalkSAT performs a random walk with a drift towards "better" truth value assignments, that is, assignments with an increasing number of clauses satisfied.

Walk-BN-EPP is inspired by Walk-SAT in the sense that we divide Walk-BN-EPP into two steps: (1) Generate an initial configuration that partitions the objects by sampling from BN-EPP using forward sampling. (2) Improve the initial partitioning by applying a random walk with a drift towards more probable partitionings, that is, BN variable state configurations with higher MAP. The two steps are laid out in Algorithm 2, and we here explain them in more detail.

**Initialization Step.** In order to perform a Walk-SAT inspired random walk, we need an initial state configuration for the variables in BN-EPP. This initial configuration should ideally be as close as possible to the solution we seek, to reduce the length of the random walk. To achieve this, we sample an initial configuration from a rough estimate of the posterior probability distribution, one object $O_{\beta_i}$ at a time, starting with $O_{\beta_1}$. That is, the state assigned to $O_{\beta_i}$ is sampled from $P(O_{\beta_i}|\mathcal{X}_\beta, \{O_{\beta_k}\}_{k=1}^{i-1})$ using the traditional Likelihood-Weighted (LW) sampling algorithm [24]. Since the "Pairwise relations" fragment follows deterministically from the "Partitions" fragment, the states of the nodes $\mathcal{A}_\beta$ are then also given. When all of the object nodes, $O_{\beta_i} \in \mathcal{O}_\beta$, have been assigned a state in this manner, we use this configuration as an initial solution candidate for the random walk. The details of the initialization step are covered by

lines 1-5 in Algorithm 2.

Note that constraints forces the posterior probability of any violating assignment to zero, with the remaining probabilities renormalized. As an example, assume that we have 16 objects and 4 partitions. We have already placed 4 objects into partition number 3. To place the 5th object, use LW sampling and obtain $P(O_{\beta_5}) = \{0.1, 0.7, 0.2, 0.0\}$ from the BN. Note that the fourth probability becomes zero due to the previous assignment of objects to the corresponding partition, reflecting a full partition. To place the 5th object we then sample a partition from $P(O_{\beta_5})$. That is, we select partition 1 w.p. 0.1, partition 2 w.p. 0.7 and partition 3 w.p. 0.2. In this example, let us assume that we sampled partition 1. The 5th object is thus assigned to this partition. We repeat this process for each object, taking into account the choices of all previously assigned objects, until all objects have been assigned to a partition.

**The Walk-SAT Based Search.** In the second step of our algorithm (lines 6-22), we seek to iteratively improve the initial configuration from the initialization step. We do this by performing a Walk-SAT inspired random walk over the state space of candidate partitions. The random walk consists of iteratively swapping the partition of randomly selected pairs of objects, $(O_{\beta_i}, O_{\beta_j}) \in \mathcal{O}_\beta \times \mathcal{O}_\beta$, with the intent of gradually moving towards more probable object partitions, and ultimately, the most probable partitioning (i.e., the solution to the MAP problem). Let the set $\mathcal{O}_\beta^t = \{O_{\beta_1} = o_1, O_{\beta_2} = o_2, \ldots, O_{\beta_i} = o_i, \ldots, O_{\beta_j} = o_j, \ldots, O_{\beta_n} = o_n\}$ be the current configuration of the network before two randomly selected objects, $O_{\beta_i}$ and $O_{\beta_j}$, swap partitions. Further, let $\mathcal{O}_\beta^{t+1} = \{O_{\beta_1} = o_1, O_{\beta_2} = o_2, \ldots, O_{\beta_i} = o_j, \ldots, O_{\beta_j} = o_i, \ldots, O_{\beta_n} = o_n\}$ be the configuration produced by the swap. Finally, let the log probability, $C^q$, of a configuration $q$ be defined as follows:

$$C^q = \log P(\mathcal{O}_\beta^q) = \sum_{1 \leq k \leq N} \log P(O_{\beta_k} = o_k | \mathit{parents}(O_{\beta_k}))$$

with $\mathit{parents}(O_{\beta_k})$ being the parents of the node $O_{\beta_k}$ in BN-EPP.

To systematically refine the current configuration, we always switch from configuration $\mathcal{O}_\beta^t$ to configuration $\mathcal{O}_\beta^{t+1}$ if the log probability $C^t$ is greater than $C^{t+1}$ (we accept the new configuration). If, on the other hand, the log probability decreases, we instead reject the new configuration, $\mathcal{O}_\beta^{t+1}$, with probability $1-\epsilon$. Otherwise, we accept the new configuration. Note that in the algorithm, $U(0,1)$ refers to a uniform distribution over the interval $[0, 1]$.

As an example assume that $C^4 = -15.3$, we then pick one objects from two different partitions, say object number 4 and 10 and swap their location. Calculating $C^5 = -14.9$ we observe that $C^5$ is greater than $C^4$, thus we accept the new state $\mathcal{O}_\beta^5$. For the next step, we select object 1 and 2 and swap their locations. However, calculating $C_6 = -20.2$ we see that the previous state, $\mathcal{O}_\beta^5$, has a larger log probability than the new configuration. Therefore we revert to the original configuration with probability $1 - \epsilon$, else, w.p. $\epsilon$ we keep the new, though inferior configuration. This process is then repeated for a predefined number of steps $T$ and the best observed configuration is presented as the solution.

**Algorithm 2:** Walk-BN-EPP

**Data:** Bayesian network BN-EPP, $\epsilon$ - probability of accepting an inferior state, and $T$ - the number of steps to execute.

**Result:** MAP configuration

**for** $i := 1$ *to* $W$ **do**
  Estimate $\pi_i = P(O_{\beta_i}|\mathcal{X}_\beta, \{O_{\beta_k}\}_{k=1}^{i-1})$ using LW.
  Draw a single sample from $\pi_i$: $s \sim \pi_i$.
  Set the state of object $i$: $O_{\beta_i} = s$
**end**
$\mathcal{O}_\beta^0 = \{O_{\beta_1} = o_1, O_{\beta_2} = o_2, \ldots, O_{\beta_n} = o_n\}$
$C_0 = \text{CalculateLogProbability}(\mathcal{O}_\beta^0)$
$\mathcal{O}_\beta^{max} := \mathcal{O}_\beta^0$
$C^{max} := C^0$
**for** $t := 1$ *to* $T$ **do**
  $O_{\beta_i}, O_{\beta_j} = \text{PickTwoRandomObjects}()$
  $\mathcal{O}_\beta^t := \text{SwapPartitionsOfObjects}(O_{\beta_i}, O_{\beta_j}, \mathcal{O}t - 1_\beta)$
  $C^t := \text{CalculateLogProbability}(\mathcal{O}_\beta^t)$
  |
  **if** $C^{max} < C^t$ **then**
    $\mathcal{O}_\beta^{max} := \mathcal{O}_\beta^t$
    $C^{max} := C^t$
  **end**
  **if** $C^t < C^{t-1}$ *and* $U_{(0,1)} < 1 - \epsilon$ **then**
    $\mathcal{O}_\beta^t := \mathcal{O}_\beta^{t-1}$
    $C^t := C^{t-1}$
  **end**
**end**
**return** $\mathcal{O}_\beta^{max}$

Table 6: Walk-BN-EPP results for different configurations on the r4w16 problem with 100 observations and p=0.75. Each data point is the average of a 1000 independent trials.

| **Walk Iterations** | **50** | **100** | **500** | **1000** | **2000** | **4000** |
|---|---|---|---|---|---|---|
| Random Prior | 0.005 | 0.02 | 0.039 | 0.05 | 0.057 | 0.064 |
| TS with 50 samples | 0.007 | 0.039 | 0.048 | 0.056 | 0.069 | 0.070 |
| TS with 250 samples | 0.061 | 0.066 | 0.063 | 0.085 | 0.11 | 0.10 |

# 5 Experimental Results on Walk-BN-EPP

To evaluate the on-line performance of BN-EPP and Walk-BN-EPP, we will here study convergence speed and accuracy empirically. The main question is how many observations, or queries, are required to obtain a correct partitioning of the objects, for various stochastic environments. Since the response to queries is stochastic, we will measure average performance over a large ensemble of independent trials.

We will explore two different kinds of stochastic environments. The first one is generated environments, where data is generated directly from an underlying SO-EPP problem. The data could for instance be generated from a SO-EPP with three partitions and nine objects (abbreviated r3w9) and a predefined level of noise. The second one is real-world environments, where an underlying perfect partitioning does not necessarily exist. Here data is separated into two parts, one part for training and one part for testing. The goal is then to solve the SO-EPP at hand using the training data, in such as manner that we maximize the performance on the unseen test data.

## 5.1 Impact of Walk-BN-EPP Parameter Settings

To evaluate the impact of the various parameters available in Walk-BN-EPP, we first solve the r4w16 (four partitions and 16 objects) problem using likelihood-weighted sampling with different number of random walk steps, as well as the number of samples used to estimate a maximum posterior initial configuration. Not surprisingly, as seen in Table 6, increasing the number of steps in the random walk significantly enhances the performance of Walk-BN-EPP. In addition, we observe that increasing the number of samples used to estimate an initial configuration increases performance further. Indeed, by applying our likelihood-weighted sampling algorithm by the modest number of 250 samples per object, we obtain an 1120% increase in probability of finding the configuration that provides the maximum posterior probability.

## 5.2 Applying Spectral Clustering (SC) to SO-EPP

The SC algorithm cannot be directly applied to SO-EPP. In order to compare our BN-EPP scheme with SC, we therefore here introduce a new variant of the SC algorithm. Vanilla SC takes a graph $G = (V, E)$ represented by a transition matrix $T$ as input. By inspecting the eigen-vectors of $T$, SC then generates a predefined number, $R$, of clusters $C = \{C_1, C_2, \ldots, C_R\}$ [21]. To find $T$ we simply row normalize the count matrix $M = \{m_{ij}\}$ where $m_{ij}$ is the number of times object $i$ and $j$ have appeared together in a query.

However, the number of objects in each cluster is not constrained to be $n$ as SO-EPP requires. So to balance $C$ we find the subset of clusters $C_- \subset C$ that have an incorrect number of objects. Let $\mu_i$ be the euclidean mean of the objects of $C_i$ as given by $T$. The fitness of a single object $o_k$ in $C_i$ is then defined as the cosine similarity between object $o_k$ row in $T$ and $\mu_i$.

We then iteratively remove the least fitting object from all clusters that have a surplus of objects. Once all clusters have $n$ or less objects, we greedily insert the removed objects, one-by-one, into the cluster where the object fits the most, and where there also is available space for the object.

This simple, yet effective, opens up for using SC to solve SO-EPP.

### 5.3 Empirical Comparison with Pursuit Object Migration Automaton (POMA) and Spectral Clustering

The Pursuit Object Migration Automaton (POMA) [8] represents state-of-the-art for solving EPP. We here compare our novel BN-EPP approach with POMA and other state-of-the-art approaches, focusing on:

- Accuracy of convergence, i.e., how many requests do we need to observe before we are able to correctly partition the objects.
- Probability of convergent requests (degree of noise).

For each experiment configuration, ten thousand individual trials were performed in order to minimize variance in our results. To avoid bias, we further independently selected a random optimal partitioning of the objects for every trial, and made sure that all algorithms were exposed to an identical sequence of incoming queries.

Unlike POMA, which requires a predetermined number of parameters (denoted $N, \tau$ and $\kappa$ as in the original paper), BN-EPP is a parameter free scheme. Note that the Walk-BN-EPP scheme for doing inference on BN-EPP does require two parameters, namely, the number of steps for the random walk and the number of samples used to estimate an initial maximum posteriori distribution. We here report the results of POMA using the standard choice of 10 states ($N = 10$) [8, 5, 4] for all of the experiment configurations. For $\tau$ (noise tolerance) and $\kappa$ (estimation phase length) we use the settings from the original paper [8]. For the warehouse experiment, a random search singles out the parameter values $\kappa = 9000$ and $\tau = 0.0003$ for high performance.

We have generated a diverse range of scenarios, and each scenario has been used to generate 1000 independent random trials to minimize variance. The results can be found in Table 7. Here, the notation rXwY refers to an EPP problem where X is the number of partitions and Y is the total number of objects. In the table, we observe that BN-EPP's accuracy on generated scenarios greatly exceed the state-of-the-art POMA as well as SC. The reasoning behind this is that POMA uses a very simple threshold scheme based on Maximum Likelihood to determine if a request is convergent or divergent. If this predefined threshold ($\tau$) is wrong then POMA will either assume that all requests are convergent or that all are divergent. BN-EPP, on the other hand, directly quantifies the uncertainty associated with the requests by estimating $p$ – the

probability of a convergent request. In Figure 6 we have plotted the probabilities BN-EPP assigned to the different $p$ values from time step to time step. A major feature of BN-EPP is that it maintains a probability distribution spanning the whole object partitioning solution space, while POMA only works from a single configuration instance. We further believe that the ability of BN-EPP to track $p$ explains why BN-EPP infers the correct partitioning significantly faster than POMA. From Table 7 it is clear that BN-EPP and SC the superior choices for solving generated SO-EPP scenarios where the data clearly forms a solution, with BN-EPP outperforming SC slightly. However, we shall see in Section 5.4 that SC does not exhibit this level of performance when faced with real-word data that does not conform as strictly to the problem definition. The reason for this is that SC implicitly imposes very strong bias onto how the data is formed, making it excellent when the problem data fits perfectly with the bias.

Table 7: The average number of objects that are wrongly placed for BN-EPP, POMA, and SC for different generated scenarios with $p = 0.6$. The results are average values, obtained from 1000 independent trials to minimize variance.

| Scenario | T | BN-EPP | POMA | SC |
|---|---|---|---|---|
| r2w4 | 10 | **0.30** | 0.55 | 0.32 |
| r3w6 | 50 | **0.04** | 0.87 | **0.04** |
| r3w9 | 100 | **0.05** | 1.86 | 0.06 |
| r6w12 | 200 | **0.00** | 1.48 | 0.02 |
| r4w12 | 200 | **0.01** | 3.42 | 0.20 |
| r2w12 | 200 | **0.41** | 1.89 | 1.29 |
| r5w15 | 400 | **0.00** | 4.97 | 0.31 |
| r3w15 | 400 | **0.00** | 4.85 | 0.08 |
| r3w18 | 800 | **0.00** | 4.62 | 1.13 |
| r6w18 | 800 | **0.00** | 6.50 | 0.16 |
| r9w18 | 800 | **0.00** | 2.48 | 0.12 |

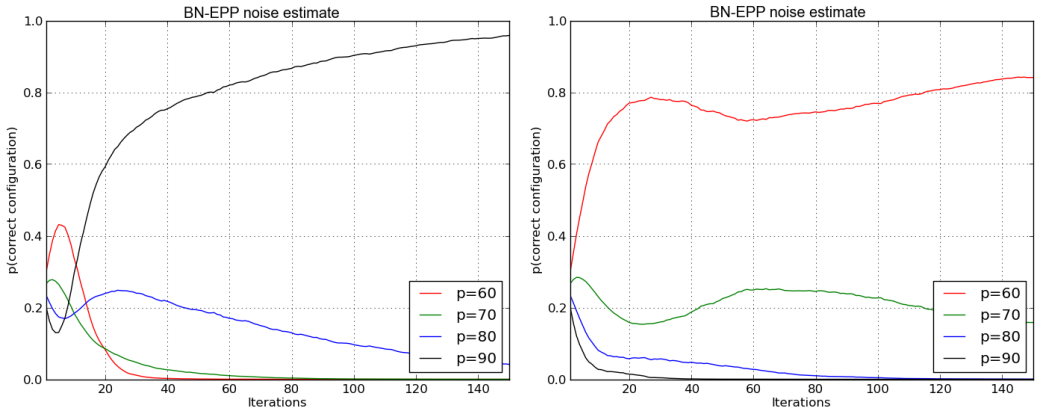


Figure 6: Probability of the different $p$ values for $R = 3$ and $W = 9$. The left plot covers the scenario where the true probability of convergent requests $p$ is 0.9, while the right plot shows the results for convergent request probability 0.6.

### 5.4 Empirical Results for Warehouse Optimization

To demonstrate the applicability of BN-EPP, we evaluate our scheme using one month (30 days) of real-world point-of-sale transaction data from a grocery outlet (collected in [3]). Each transaction $T_k$ is a subset of the set of all unique articles $O$, where $|O| = 169$. In total there are 9835 transactions. Each article is labeled by its type of product, e.g. ice-cream instead of the actual brand. The number of articles per transaction vary wildly from orders of size 32 down to single article transactions. The mean number of objects per transaction is 4.4, with a standard deviation of 3.5.

Note that the above data-set does not provide a physical layout of the grocery outlet. Therefore, we here assume that the objects are to be partitioned among 13 different sections of the store in such a manner that the time each customer spend travelling between sections is minimized when collecting the articles on their shopping list (transaction $T_k$).

In addition, as discussed in the introduction, we introduce constraints on the placement of objects, listed in Table 1 transforming the problem from a SO-EPP problem to a CSO-EPP problem. However, as neither OMA/POMA nor SC does support CSO-EPP, we will include results for SO-EPP.

To measure solution effectiveness, we track how many warehouse sections, $v$, a consumer must visit to collect all the wares on his shopping list. We then assume that the experienced cost of travel doubles for each new unique section the consumer must visit. As an example, if a customer needs to visit 3 different sections the cost of that transaction becomes $2^3 = 8$.

We evaluate the effectiveness of BN-EPP using 5 fold cross validation, where we select 1 fold for training and 4 folds for testing. We report the mean cost of the transactions in the test set. The 5-fold cross validation is performed 1000 times to estimate expected effectiveness. For Walk-BN-EPP, we used the parameter settings of likelihood-weighted sampling with 100 samples per object and 1000 iterations for the walk phase. Table 8 demonstrate that Walk-BN-EPP significantly outperform the state-of-the-art by obtaining nearly half the loss compared to POMA and SC. Even when imposing the rules from Table 1, rendering the optimization problem significantly harder, we obtain comparable loss to the other schemes, with the other contenders (solving the easier SO-EPP) having no such rules imposed on their solution. One reason for the effectiveness of Walk-BN-EPP can be the Bayesian global perspective used to guide the search, which is in contrast to the heuristic local search employed by the competing approaches.

We would finally like to remark that our experiments show that POMA [8] seems to be highly dependent on the hyper-parameters. During our random search we often observed POMA obtaining a low loss on parts of the data with a particular set of hyper-parameters, only for the loss to be much higher than average for other part of the dataset with the same hyper-parameters.

Table 8: Effectiveness of Walk-BN-EPP, POMA and SC on the grocery dataset [3] as measured in number of sections traveled (5-fold cross validation). In physical terms, we can see that Walk-BN-EPP roughly halves the number of sections a customer has to visit on average to find all groceries on the shopping list.

| | Walk-BN-EPP (CSO-EPP / SO-EPP) | POMA (SO-EPP) | OMA (SO-EPP) | SC (SO-EPP) |
|---|---|---|---|---|
| Mean | 61.6 / **30.6** | 56.0 | 68.1 | 61.6 |
| Std.Dev | 6.1 / **4.5** | 7.0 | 6.5 | 14.4 |

# 6 Conclusion and Further Work

In this paper we have presented a novel approach to the Constrained Stochastic Online Equi-Partitioning Problem (CSO-EPP), namely, the Bayesian Network EPP model and inference scheme. We have demonstrated how the various components of BN-EPP interact and that BN-EPP significantly outperform existing state-of-art, not only in speed of convergence, but also in its ability to estimate the stochastic properties of the underlying environment. From a history of object arrivals, we are able to predict which objects will appear together in future arrivals. To enable BN-EPP to deal with larger data sets we introduced Walk-BN-EPP, a WalkSAT inspired solver for BN-EPPs. Walk-BN-EPP was then applied to a real-world warehouse problem and shown to significantly outperform state-of-the-art inference schemes, even when constraining the solution space in terms of real-world constraints.

We also introduced an adaption of Spectral Clustering (SC) for SO-EPP and showed that its performance on generated SO-EPP scenarios came close to BN-EPP, however it was clearly outperformed by BN-EPP on more complex real-world datasets [3].

In our future work, we intend to investigate how the BN-EPP approach can be expanded to cover other classes of stochastic optimization problems such as graph partitioning and poset ordering problems, potentially outperforming generic off-line techniques such as Particle Swarm Optimization (PSO) [40], Genetic Algorithm (GA) [41] or Ant Colony Optimization (ACO) [42].

# References


[1] R. de Koster, T. Le-Duc, and K. J. Roodbergen, "Design and control of warehouse order picking: A literature review," *European Journal of Operational Research*, vol. 182, no. 2, pp. 481 – 501, 2007. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0377221706006473

[2] J. A. Tompkins, *Facilities planning.* Wiley, 2010.

[3] M. Hahsler, K. Hornik, and T. Reutterer, "Implications of probabilistic data modeling for mining association rules," in *From Data and Information Analysis to Knowledge Engineering.* Springer, 2006, pp. 598–605.

[4] W. Gale, S. Das, and C. T. Yu, "Improvements to an algorithm for equipartitioning," *Computers, IEEE Transactions on*, vol. 39, no. 5, pp. 706–710, 1990.

[5] B. Oommen and D. Ma, "Deterministic learning automata solutions to the equipartitioning problem," *Computers, IEEE Transactions on*, vol. 37, no. 1, pp. 2–13, Jan 1988.

[6] R. Xu, D. Wunsch *et al.*, "Survey of clustering algorithms," *Neural Networks, IEEE Transactions on*, vol. 16, no. 3, pp. 645–678, 2005.

[7] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping multidimensional data*. Springer, 2006, pp. 25–71.

[8] A. Shirvani and B. J. Oommen, "On enhancing the object migration automaton using the pursuit paradigm," *Journal of Computational Science*, 2017.

[9] M. Hammer and A. Chan, "Index selection in a self-adaptive data base management system," in *Proceedings of the 1976 ACM SIGMOD international conference on Management of data*. ACM, 1976, pp. 1–8.

[10] C. T. Yu, M. K. Siu, K. Lam, and F. Tai, "Adaptive clustering schemes: general framework," in *Proc. IEEE COMPSAC Conf.* IEEE, 1981, pp. 81–89.

[11] D. Ciu and Y. Ma, "Object partitioning by using learning automata," Ph.D. dissertation, Carleton University, 1986.

[12] A. S. Mamaghani and M. R. Meybodi, "Clustering of software systems using new hybrid algorithms," in *Computer and Information Technology, 2009. CIT'09. Ninth IEEE International Conference on*, vol. 1. IEEE, 2009, pp. 20–25.

[13] B. J. Oommen, R. S. Valiveti, and J. R. Zgierski, "An adaptive learning solution to the keyboard optimization problem," *IEEE transactions on systems, man, and cybernetics*, vol. 21, no. 6, pp. 1608–1618, 1991.

[14] C. Daskalakis, R. M. Karp, E. Mossel, S. J. Riesenfeld, and E. Verbin, "Sorting and selection in posets," *SIAM Journal on Computing*, vol. 40, no. 3, pp. 597–622, 2011.

[15] K. Andreev and H. Racke, "Balanced graph partitioning," *Theory of Computing Systems*, vol. 39, no. 6, pp. 929–939, 2006.

[16] R. E. Burkard, *Quadratic assignment problems*. Springer, 2013.

[17] P. Galinier, Z. Boujbel, and M. C. Fernandes, "An efficient memetic algorithm for the graph partitioning problem," *Annals of Operations Research*, vol. 191, no. 1, pp. 1–22, 2011.

[18] J. Kim, I. Hwang, Y.-H. Kim, and B.-R. Moon, "Genetic approaches for graph partitioning: a survey," in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*. ACM, 2011, pp. 473–480.

[19] U. Gupta and N. Ranganathan, "A game theoretic approach for simultaneous compaction and equipartitioning of spatial data sets," *Knowledge and Data Engineering, IEEE Transactions on*, vol. 22, no. 4, pp. 465–478, 2010.

[20] M. Meila and J. Shi, "Learning segmentation by random walks," in *Advances in neural information processing systems*, 2001, pp. 873–879.

[21] S. X. Yu and J. Shi, "Multiclass spectral clustering," in *Proceedings of the Ninth IEEE International Conference on Computer Vision - Volume 2*, ser. ICCV '03. Washington, DC, USA: IEEE Computer Society, 2003, pp. 313–. [Online]. Available: http://dl.acm.org/citation.cfm?id=946247.946658

[22] M. Agache and B. J. Oommen, "Generalized pursuit learning schemes: New families of continuous and discretized learning automata," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 32, no. 6, pp. 738–749, 2002.

[23] B. J. Oommen and M. Agache, "Continuous and discretized pursuit learning schemes: Various algorithms and their comparison," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 31, no. 3, pp. 277–287, 2001.

[24] D. Koller and N. Friedman, *Probabilistic graphical models: principles and techniques*. MIT press, 2009.

[25] C. Yuan, T.-C. Lu, and M. J. Druzdzel, "Annealed map," in *Proceedings of the 20th conference on Uncertainty in artificial intelligence*. AUAI Press, 2004, pp. 628–635.

[26] W. R. Thompson, "On the likelihood that one unknown probability exceeds another in view of the evidence of two samples," *Biometrika*, vol. 25, no. 3/4, pp. 285–294, 1933.

[27] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems," *Machine Learning*, vol. 5, no. 1, pp. 1–122, 2012.

[28] O.-C. Granmo, "Solving two-armed bernoulli bandit problems using a bayesian learning automaton," *International Journal of Intelligent Computing and Cybernetics*, vol. 3, no. 2, pp. 207–234, 2010.

[29] O. Chapelle and L. Li, "An empirical evaluation of thompson sampling," in *Advances in Neural Information Processing Systems 24*, J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2011, pp. 2249–2257.

[30] S. Agrawal and N. Goyal, "Analysis of thompson sampling for the multi-armed bandit problem," in *Conference on Learning Theory, COLT*, 2012.

[31] ——, "Further optimal regret bounds for thompson sampling," in *Proceedings of the Sixteenth International Conference on Artificial Intelligence and Statistics*, 2013, pp. 99–107.

[32] ——, "Thompson sampling for contextual bandits with linear payoffs," in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, 2013, pp. 127–135.

[33] S. Glimsdal and O.-C. Granmo, "Gaussian process based optimistic knapsack sampling with applications to stochastic resource allocation," in *Proceedings of the 24th Midwest Artificial Intelligence and Cognitive Science Conference 2013*. CEUR Workshop Proceedings, 2013, pp. 43–50.

[34] O.-C. Granmo and S. Glimsdal, "Accelerated bayesian learning for decentralized two-armed bandit based decision making with applications to the goore game," *Applied intelligence*, vol. 38, no. 4, pp. 479–488, 2013.

[35] L. Jiao, X. Zhang, B. J. Oommen, and O.-C. Granmo, "Optimizing channel selection for cognitive radio networks using a distributed bayesian learning automata-based approach," *Applied Intelligence*, vol. 44, no. 2, pp. 307–321, 2016.

[36] D. Tolpin and F. Wood, "Maximum a posteriori estimation by search in probabilistic programs," in *Eighth Annual Symposium on Combinatorial Search*, 2015.

[37] B. Selman, H. A. Kautz, and B. Cohen, "Noise strategies for improving local search," in *AAAI*, vol. 94, 1994, pp. 337–343.

[38] P. Larrañaga, H. Karshenas, C. Bielza, and R. Santana, "A review on evolutionary algorithms in bayesian network learning and inference tasks," *Information Sciences*, vol. 233, pp. 109–125, 2013.

[39] T. Soh, M. Banbara, and N. Tamura, "Proposal and evaluation of hybrid encoding of csp to sat integrating order and log encodings," *International Journal on Artificial Intelligence Tools*, vol. 26, no. 01, p. 1760005, 2017.

[40] J. Kennedy, "Particle swarm optimization," in *Encyclopedia of Machine Learning*. Springer, 2011, pp. 760–766.

[41] J. H. Holland, "Genetic algorithms," *Scientific American*, vol. 267, no. 1, pp. 66–72, 1992.

[42] M. Dorigo, M. Birattari, and T. Stutzle, "Ant colony optimization," *IEEE Computational Intelligence Magazine*, vol. 1, no. 4, pp. 28–39, 2006.