Control and Cybernetics

vol. **50** (2021) No. 3

Learning causal theories with non-reversible MCMC methods[*]

by

Antonina Krajewska

NASK National Research Institute
Kolska 12, 01045 Warszawa, Poland
antonina.krajewska@nask.pl

Abstract: Causal laws are defined in terms of concepts and the causal relations between them. Following Kemp et al. (2010), we investigate the performance of the hierarchical Bayesian model, in which causal systems are represented by directed acyclic graphs (DAGs) with nodes divided into distinct categories. This paper presents two non-reversible search and score algorithms (Q_1 and Q_2) and their application to the causal learning system. The algorithms run through the pairs of class-assignment vectors and graph structures and choose the one which maximizes the probability of given observations. The model discovers latent classes in relational data and the number of these classes and predicts relations between objects belonging to them. We evaluate its performance on prediction tasks from the behavioural experiment about human cognition. Within the discussed approach, we solve a simplified prediction problem when object classification is known in advance. Finally, we describe the experimental procedure allowing in-depth analysis of the efficiency and scalability of both search and score algorithms.

Keywords: Bayesian inference, causal systems, directed acyclic graph, MCMC, non-reversible Markov processes, search and score methods

1. Introduction

Causal reasoning is one of the most profound capabilities of the human mind. Knowledge about causal structure helps us in predicting future events and plan relevant interventions. Often causal relations form simple theories. In such systems, causal laws are defined in terms of concepts and causal relations between them. For example, learning the theory of magnetic interactions requires learning three concepts: magnets, magnetic objects, non-magnetic objects, and the laws describing their causal relations (see Kemp et al., 2010)

[*]Submitted: May 2021; Accepted: July 2021.

Following Kemp et al. (2010), we revisit an experiment about causal reasoning inspired by learning the theory of magnetism. The experiment uses a custom-built graphical interface. Participants move square objects around and observe the interactions between them. Items are divided into two categories, A and R. When objects of different categories touch each other, the R objects light up and play sounds. Objects are labelled either by letters from the alphabet or else they are undifferentiated. Participants are not told which class the items belong to, and they are not told the various laws governing each class. Their task is to learn this from the experiment. The experiment aims to determine whether people can acquire the causal theory and make predictions about unobserved relations between objects. We compare the performance of humans learning these rules in an experimental situation with the performance of a machine learning algorithm.

Kemp, Griffiths and Tenenbaum (2004) introduced an infinite ordered block model, a Bayesian framework for learning abstract relational knowledge. The model assumes that objects are divided into distinct categories and that there is a natural ordering of those categories. A particular relation holds for any pair of items with a probability that depends only on their classes. The algorithm discovers latent classes in the data and predicts relations between objects. The model was evaluated both on learning the structure of kinship systems and learning causal theories. In this paper, we focus on the latter one. The natural representation of the causal system from the experiment scenario is a directed acyclic graph (DAG). The nodes of the graph represent objects and belong to distinct categories. If an item was activated when touched against, we draw an arrow from the latter. The probability of an edge between two nodes depends on their classes. The acquisition of the described causal theory is equivalent to the learning of structure of a graphical model, defined as inferring the set of edges of the graph.

Koller et al. (2007) distinguish two approaches to the structure learning task, sampling-based and optimization-based (also referred to as search and score methods). Markov processes and the Metropolis-Hastings (M-H) algorithm play the essential yet different roles in both of them. Sampling-based methods approximate posterior distribution and rely on the fact that the underlying Markov process is ergodic. Most commonly used algorithms are based on Gibbs and M-H sampling (see Goudie and Mukherjee, 2016; Koller, et al. 2007). On the other hand, optimization-based solvers run through the space of all possible graph structures and examine their scores, expressing their posterior probability given the set of observations. M-H algorithm is widely used to run through the search space (see Madigan and York, 1993; Friedman and Koller, 2001). Since the goal of the underlying Markov chain is to visit pairs with relatively high scores, it is not required that the chain satisfy conditions of the ergodic theorem, such as reversibility (Koski and Noble, 2012). Corander, Gyllenberg and Koski (2006), and Corander, Ekdahl and Koski (2008) introduced the Metropolis-Hastings method with simplified, non-reversible kernels. Empirical results demonstrated the advantages of their approach, including flexibility in choosing intelligent proposal mechanisms, efficient computation, and parallel implementations, in which Markov chains exchange information about posterior distribution.

Several extensions of the infinite ordered block model might be found in the literature. Mansinghka et al. (2006) restricted it to directed acyclic graphs (DAG) and applied it to Bayesian networks. Their model is also suitable for the experimental scenario. Kemp, Griffiths and Tennenbaum (2004), Kemp et al. (2010) and Mansinghka et al. (2006) based structure learning on Gibbs sampling. However, Mansinghka et al. (2006) noted that developing search and score methods would be unquestionably helpful. Basing on their work, Rios, Noble and Koski (2015) introduced prior probability distribution, which uses the ordering of the node classes. The advantage of this model is that it offers a closed, computable form of the prior probability over the hypothesis spaces, which might be used to calculate score values by search and score methods. They have shown that the Gibbs sampler converged slowly to the posterior distribution and overcame this problem with the search and score method using the non-reversible Metropolis-Hastings processes.

This paper presents two non-reversible search and score algorithms, referred to as Q_1 and Q_2, and their application to the causal system learning, described above. Both algorithms used prior probability distribution from the work of Rios, Noble and Koski (2015) and were evaluated on the prediction tasks from the cognitive experiment. Moreover, we carried in-depth convergence analysis in the experimental scenario with a limited computational budget. We used empirical cumulative distribution (ECDF) to display the proportion of trials that achieved given accuracy. We found that although Q_1 reached better scores than Q_2, its predictions were more dependent on the initial points of the Markov chain and favoured sparser graphs. Q_2 was designed to avoid this phenomenon and performed better in prediction tasks. Finally, we applied Q_1 to solve the simplified problem, when object categorization is known in advance. The predictions made in that case were accurate.

2. Related work

Modern statistics and machine learning allow cognitive scientists to construct new representations of the human mind. Hierarchical Bayesian models have been criticized for focusing on the computational character of learning processes and simplifying other mechanisms (see Jones and Love, 2011; McClelland et al., 2010). However, they provide us with a theoretical tool for investigating the effectiveness of human inductive reasoning (Griffiths et al., 2010). Since they provide a formal representation of the abstract knowledge, they serve as a point of reference for operations performed by the human mind (Tenenbaum et al., 2011). They help us answer how much data do we need? and: which representations are helpful for inductive reasoning?

Bayesian networks are a natural language to describe a wide class of causal systems (see Koski and Noble, 2009). Many structure learning algorithms focus on discovering them in data (see Scanagatta, Salmeron and Stella, 2019; Madsen et al., 2017; Gao and Huang, 2020; Silander et al., 2018; Dai, Ren and Wu, 2020; Contaldi, Vafaee and Nelson, 2019). Mansinghka et al. (2006) notice that many models assume generic prior constraints on the graph structure. However, this may not be adequate for many real-world learning problems, in which variables can be categorized and play different roles in the causal system. This knowledge imposes constraints on prior probability distributions over search space and facilitates learning from small data sets. In this paper, we use prior in which the probability of an edge between two nodes is a function of their classes. Constraining the space to DAGs (directed acyclic graphs) may be used in search and score algorithms for learning structure of Bayesian networks, see Mansinghka et al. (2006). The prior distribution over graph space has been also considered in the work of Flores and al. (2011).

Koski and Noble (2012) outline three main approaches for learning graph structure from data (*structure learning*): *search and score*, *constraint based* and *hybrid*. Gibbs sampling (see Peters, 2008), used by Kemp, Griffiths and Tennenbaum (2004), incorporates the Metropolis-Hastings (M-H) algorithm and is classified as a hybrid method. In this paper, we introduce two search and score algorithms for the same task.

Search and score methods are commonly used in structure learning, as they find a graph that optimizes some scoring criterion (see Friedman, Nachman and Pe'er, 1999; Moore and Wong, 2004; Chickering, 2002; Lee and van Beek, 2017). Various scoring functions have been examined, including *Cooper Herskovitz likelihood* (Cooper and Herskovitz, 1992), *Bayesian Information Criterion* (BIC) (Schwarz, 1978), *minimum description length* ($MDL = -BIC$) (Rissanen, 1978) or *Akaike Information Criterion* (AIC) (Akaike, 1974).

Markov chain Monte Carlo (MCMC) algorithms have gained considerable popularity for obtaining posterior distribution in Bayesian inference (Ravenzwaaij, Cassey and Brown, 2018). A review of scalable MCMC techniques applied in Bayesian inference may be found in the survey written by Angelino, Johnson and Adams (2016). On the other hand, Markov processes may also be used to run through model space in search and score methods (Madigan et al., 2000; Corander, Ekdahl and Koski, 2008). Robinson (1973) proved that a recursive function gives the number of DAG structures with d nodes

$$N(d) = \sum_{i=1}^{d} (-1)^{i+1} \binom{d}{i} 2^{i(d-i)} N(d-i), \tag{1}$$

which means that it grows super-exponentially. Therefore, it is not possible to ensure that the chain visits each site even once. The Metropolis-Hastings (M-H) algorithm can be adopted to construct an efficient Markov chain. Within the M-H framework, the transition between steps is split into two steps: proposal and acceptance-rejection. The candidate generating density is defined by matrix Q and is crucial for the successful performance of the algorithm. We are not interested in empirical distribution approximating the Markov chain's stationary distribution in search and score algorithms. The convergence of the Markov process to a stationary distribution is irrelevant and gives us flexibility in constructing Markov processes, which may not satisfy conditions such as reversibility or Markov property. Recently, the M-H algorithm's extensions, based on non-reversible processes gain an increasing popularity (see Bierkens, 2016; Corander, Gyllenberg and Koski, 2006). This paper introduces the non-reversible search and score algorithms and investigates their performances in the hierarchical Bayesian model, described by Kemp et al. (2010).

3. Problem formulation

The experiment consists of eight phases, see Kemp et al. (2010). In each stage, three new objects are introduced. The items are divided into two classes: A (activators) and R (reactors). When objects from different categories touch each other, the R-objects get activated. Participants do not know about the division and the causal law describing the interaction; their task is to discover this.

Phase 0 Three objects - two from category A, one from category R - are in the staging area. Participants are asked to play around with them to see what lights up. Before proceeding, participants describe how, in their opinion, objects work.

Phase 1-7 Each phase has four parts.

Pre-test. Three new objects are added to the staging area. One of them serves as the probe object x. On odd and even-numbered phases, the categories of the three new items are R, R, A and A, A, R, respectively. The probe object belongs to the category to which two of the new objects belong. Participants answer the questions about the interaction of x with the familiar items from A and R classes (which appeared one phase before):

Consider, what will happen when objects x and a touch
Will x activate?
Will a activate?

Consider, what will happen when objects x and r touch
Will x activate?
Will r activate?

At this moment, participants are not able to predict how the x will interact with these objects.

Single interaction Participants are asked to touch the x object against the y object - the last one from the three items which appeared one phase before. The target y is always of the opposite category to the x. If participants have discovered the rule determining the interactions and classified objects which appeared a phase before correctly, they can classify x after observing this single interaction. Items from different categories touched, therefore one of them activated. Knowing the class of y and the rule determining the interactions allows the participants to classify x in the correct category.

Post-test Participants answer the same questions as in the pre-test.

Play Participants are asked to play around with all the objects currently on the screen and test all possible interactions. By moving items around, participants can organise them spatially. Participants proceed to the next phase at their own pace.

Figure 1 presents a screenshot of the single interaction in phase 1. Objects are labelled by random alphabet letters to allow referring to them in pre-test and post-test questions. After the first and last phase, participants were asked to describe the rule, explaining the interactions between objects. Half of the participants distinguished the two categories of objects and identified the relation between them correctly. This result shows the relative difficulty of the discussed theory and brings out the efficiency of the presented algorithms.

Table 1 presents the average values of answers given by the experiment participants after a single interaction phase. The response scale spans from 0 (definitely not) to 10 (definitely yes). At this moment, theory learners should correctly predict interactions between x and both types of target objects: activators and reactors. In odd-numbered phases, x is an activator. It does not activate when touched against both the reactor object r and the activator object a. Furthermore, object r will activate, but a will not. In even-numbered phases, x is a reactor. None of the objects a and r will activate when touched against x. Probe object x will activate when touched against the activator object a.

4. Bayesian model of causal theory formation

In this section, we present the model, which may describe the causal system learning, outlined before. We consider this system as an example of a simple theory. Following Kemp et al. (2010), Carey (1985) and Wellman and Gelman (1992), we characterize theories as sets of concepts and relationships between those concepts. This definition includes scientific theories and intuitive ones, organizing our everyday knowledge (see Carey, 1985; Murphy and Medin, 1985; Wellman and Gelman, 1992). Often, concepts and relationships in such systems define each other. This raises the question: how a learner can break into a closed system and acquire its components? The discussed algorithm gives us a computational answer.

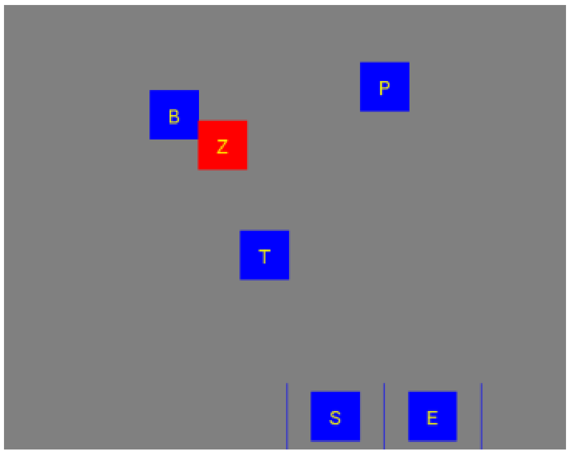


Figure 1: Screenshot of the single interaction phase. Objects are labelled by random alphabet letters to distinguish them. Object B (activator) serves as probe object x and object Z (reactor) serves as target object y

Question	Ideal	Phase number						
		$l = 1$	$l = 2$	$l = 3$	$l = 4$	$l = 5$	$l = 6$	$l = 7$
$x \in A$								
Will x activate?	0		4.17		2.39		2.11	
Will a activate?	0		4.94		2.89		3.39	
Will x activate?	0		3.78		3.33		2.94	
Will r activate?	10		5.89		7.44		7.28	
$x \in R$								
Will x activate?	10	7.28		7.28		6.28		7.94
Will a activate?	0	3.00		3.00		2.78		2.59
Will x activate?	0	7.11		7.11		3.73		4.17
Will r activate?	0	6.22		6.22		2.33		0.67

Table 1: Average values of answers given by participants of the experiment (37 students of Warsaw University) to the post-test questions) in phases $l = 1, \dots, 7$. Response scale spans from 0 (definitely not) to 10 (definitely yes).

The theory we speak about may be represented by a graph G with nodes assigned to the two classes: activators (A) and reactors (R) and edges directed according to the causal relation. At each stage, some of the edges are known from the previous observations. We define the prior probability distribution over space of pairs - graphs and classifications of their nodes - given the set of known edges. The pair with the highest probability is then chosen. A Markov chain Monte Carlo algorithm with the Metropolis-Hastings style transitions is used.

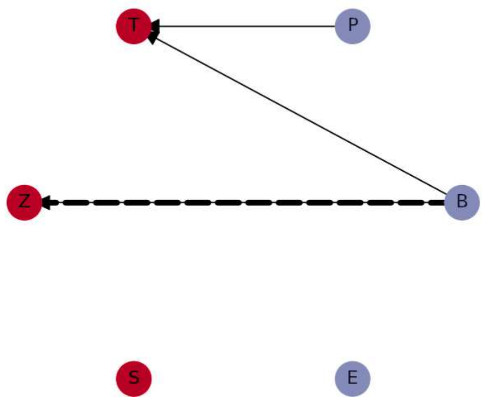


Figure 2: DAG representation of the causal system, presented in the phase 1, shown in Fig. 1. Edges $(P,T),(B,T)$ are known before the single interaction. Edge (B,Z) is established after the single interaction. Violet objects belong to the class A, red to the class R

The here presented algorithm discovers the classification of objects based on observed interactions and can predict future interactions between them. Although interactions between objects from different categories occur with probability equal 1 in the experimental scenario, the model assumes uncertainty about this probability. Moreover, in Section 5.5, we present a modification of this model that solves prediction tasks concerning known categorization.

The designed algorithm does not detect certain features of the experiment described, where objects appear in a certain order. Two reactors and one activator follow two activators and one reactor in odd and even-numbered phases in the experiment. For every interaction, objects touched against each other belong to different classes, although the participant does not know this. When there are exactly two categories of objects, introduced as AAR and RRA in odd and even phases, respectively, it is straightforward for a human to detect that this is the pattern after a few phases.

A machine learning algorithm that accommodates the possibility of introducing different classes in successive batches of three is harder to construct. It could be done by considering each group of three as an observation of a time-homogeneous Markov chain, with a prior distribution over the transition probabilities, which is then updated to a posterior given the observed groups of three. As $n \to +\infty$, the posterior will converge to a Dirac mass over: $P_n(AAR_{\text{next}}|RRA_{\text{now}}) = 1$ and $P_n(RRA_{\text{next}}|AAR_{\text{now}}) = 1$. This would result in a hierarchical model, which is only useful if the groups of three were being introduced by the experimenter according to a Markov model rather than simply at random. For any approach with a level of flexibility, the machine learning algorithm would take longer than a human to detect the pattern.

The algorithm used should learn that there are only two categories: the probability of introducing a third class in future phases tends to zero as more phases are observed. It will not learn the AAR / RRA pattern in any case.

In the situation, where the *activator* activated the *reactor* with probability 1, the humans outperformed the machine learning algorithm; in the experiment, where the probability of activation was 0.5, the results by humans were poor; the algorithm has better chances of achieving correct classification.

4.1. Directed Acyclic Graph representation

Directed acyclic graph (DAG) with nodes referring to objects may represent the causal system under consideration. If an object y activates an object x when they touch each other, there is a directed arrow from y to x. Only objects from category A activate objects from category R, and there are no interactions between objects of the same category. Therefore, the graph has no cycles. After phase l, $l = \{0, ..., 6\}$, there are $3(l+1)$ objects on the screen. Given all possible observations, edges between all $3(l+1)$ nodes are known. At the beginning of phase $l+1$, three new objects are introduced. One of them serves as a probe object. After a single interaction, an edge between nodes representing the probe object and one of the familiar objects is established. Fig. 2 shows the edges that are known in phase $l = 1$.

The algorithm gives, as output, the DAG structure with nodes representing all of the objects on the screen. It predicts the interactions between the probe object and objects which appeared a phase before.

4.2. Prior probability distribution over the search space

This section provides the definition of the prior probability distribution over the search space of pairs of graph structures and node classifications. The closed and relatively simple formula allows the application of the search and score approach, particularly the non-reversible M-H algorithms, presented in this paper. The here described prior captures higher-level knowledge that relation between objects depends on the category they belong to. Encoded information facilitates learning from small data sets. Moreover, the number of classes is not known in advance. Thus, the model predicts unknown relations and discovers correct categorization of the data set.

The causal system in the described experiment is fairly simple. There are only two types of objects, and the probability of the relationship between objects of different types is equal to 1. However, the considered prior is suitable for more complex systems, with different types of relations (including symmetric ones, represented by undirected graph edge) or systems where the probability of a relation is smaller than 1. Evaluation of the discussed distribution on medical data might be found in the paper of Rios, Noble and Koski (2015).

The construction of the prior distribution $P_{\mathcal{G},\underline{Z}}$ consists of three steps. Let $\mathcal{G}$ be a space of all possible DAG structures with d nodes and let $\underline{Z}$ denote a d-dimensional random vector, assigning class labels to each element of $\{1, ..., d\}$. We begin with defining the prior probability distribution for a random vector $\underline{Z}$. Having assigned graph nodes to categories, we consider $P_{\mathcal{G}|\underline{Z}}$. Finally, we derive explicit formula for $P_{\mathcal{G},\underline{Z}}$.

Probability of a node classification

Firstly, we consider prior probability distribution for a random vector $\underline{Z}$. To represent accurately the intuitive theories, developed by humans, suitable prior should favour vectors $\underline{z}$ with a small number of classes and allow this number to grow as more objects are seen (Kemp et al., 2010). Let K be the number of non-empty classes. Since K is unknown, $K \in \{1, ..., d\}$ and $\underline{Z}$ may take values in $\{1, ..., d\}^d$. We will assign each node to a given class in the order $\sigma(1), ..., \sigma(d)$, according to the Chinese Restaurant Process with hyper-parameter α (CRP(α)). We place the uniform distribution over the permutation space of d nodes $\Sigma^{(d)}$

$$P(\sigma) = \frac{1}{d!} \text{ for } \sigma \in \Sigma^{(d)} \tag{2}$$

and set $\underline{Z}_{\sigma(1)} = 1$. According to the CRP($\alpha$), subsequent nodes will be assigned to one of the existing classes with the probability proportional to the class size or form a new class with the probability controlled by the parameter α. Formally, let $Z_{\sigma_1}, \dots Z_{\sigma_j}$ have values $z_1, \dots, z_j$ and $m_k^{z,j}$ be the count of the number of nodes, which are labelled by first j values $\{z_1, \dots, z_j\}$ and assigned to the class k. We will denote $m_k^{z,d}$ as m_k. Suppose that there are already k_j classes. The node labelled as $j+1$ will belong

1. to the new class $(k_j + 1)$ with probability $\frac{\alpha}{\alpha+j}$,
2. or to the one of the existing classes (k) with probability $\frac{m_k^{z,j}}{\alpha+j}$.

The following lemma results from the definition of the CRP(α) and provides a closed formula for the probability of the classification of nodes determined in the order σ.

Lemma 4.1 *Let $\underline{Z}$ denote a d-dimensional random vector assigning class labels to each element of $V = \{1, \dots, d\}$. Let $\Sigma^{(d)}$ be a permutation space of V and let $\sigma \in \Sigma^{(d)}$ be an ordering, in which we assign class labels to elements of V, according to CRP(α). Let $S_d = \{\underline{z} | z_{\sigma(1)} = 1, z_{\sigma(1)} \leq z_{\sigma(2)} \leq \dots \leq z_{\sigma(d)}\}$. The distribution over $\underline{Z}$ for a given parameter α may be computed explicitly. If $\underline{z} \in S_d$*

$$P_{\underline{Z}|\Sigma^{(d)}}(\underline{z}|\sigma) = \alpha^K \frac{\Gamma(\alpha)}{\Gamma(\alpha+d)} \prod_{k=1}^{K} m_k! \tag{3}$$

for $K = 1, \dots, d$, where K denotes the number of non empty classes and m_k counts the number of the nodes labelled as k by $\underline{z}$. Otherwise, $P_{\underline{Z}|\Sigma^{(d)}}(\underline{z}|\sigma) = 0$.

Proof See Appendix 1.

Conditional probability of a graph given node classification

Now, we construct the conditional probability of a graph given the classification of its nodes: $P_{\mathcal{G}|\underline{Z}}$. The probability of an edge between two nodes depends only on their categories. Let $\underline{M}$ be a random matrix of size $K \times K$, representing the probabilities of drawing an edge from a node of class a to a node of class b

$$\underline{M}_{ab} := P((z_i, z_j) \in E | z_i = a, z_j = b), \tag{4}$$

where E denotes the set of graph edges. For $b > a$ we draw an entry $\underline{M}_{ab}$ from the Beta distribution with parameters $\beta_{a,b,1}$, $\beta_{a,b,2}$. Since we are interested only in the acyclic graphs, we set $\underline{M}_{ab} = 0$ for $a \geq b$. Based on the values of $\underline{M}$, we can store probabilities of drawing each of the graph edges in a $(d \times d)$-dimensional random matrix Ψ. Since $\underline{M}$ determines $\underline{\Psi}$, $P_{\mathcal{G}|\underline{Z},\underline{M}} = P_{\mathcal{G}|\underline{Z},\Psi}$, and

$$P_{\mathcal{G}|\underline{Z},\underline{\Psi}}(G|\underline{z}, \underline{\psi}) = \prod_{i=1}^{d} \prod_{j=1}^{d} \psi_{ij}^{\underline{G}_{ij}} (1 - \psi_{ij})^{1 - \underline{G}_{ij}}, \tag{5}$$

where $\underline{G}$ is an adjacency matrix of a graph G. The following lemma provides an explicit formula for $P_{\mathcal{G}|\underline{Z}}$.

Lemma 4.2 *Let $\mathcal{G}$ be a space of all possible DAG structures with d nodes and let $\underline{Z}$ be a d-dimensional random vector, assigning K class labels to $\{1, \dots, d\}$. Let probability $P_{\mathcal{G}|\underline{Z},\underline{M}}$ be given by (5) for matrix $\underline{M}$, defined by eq.(4), then*

$$P_{\mathcal{G}|\underline{Z}}(G|\underline{z}) = \prod_{1 \leq a \leq b \leq K} \frac{B(\beta_{a,b,1} + q_{a,b}, \beta_{a,b,2} + r_{a,b})}{B(\beta_{a,b,1}, \beta_{a,b,2})}, \tag{6}$$

where $q_{a,b}$ is the number of present edges from nodes of class a to nodes of class b, $r_{a,b}$ is the number of absent edges and B denotes the Beta function.

Proof See Appendix 2.

Joint probability for a graph and classification of its nodes

Finally, we apply Lemmae 4.1 and 4.2 to calculate joint probability distribution $P_{\mathcal{G},\underline{Z}}$. We begin with the following observation: after the nodes had been classified, σ gives no further information about the graph structure. Hence,

$P_{\mathcal{G}|\underline{Z}} = P_{\mathcal{G}|\underline{Z},\Sigma^{(d)}}$. This gives a joint probability distribution over classifications and graph structures

$$P_{\mathcal{G},\underline{Z}}(G,\underline{z}) = \sum_{\sigma\in\Sigma^{(d)}} \frac{1}{d!} P_{\mathcal{G}|\underline{Z}}(G|\underline{z}) P_{\underline{Z}|\Sigma^{(d)}}(\underline{z}|\sigma). \tag{7}$$

Lemma 4.3 *Let $\mathcal{G}$ be a space of all possible DAG structures with d nodes and let $\underline{Z}$ be a d-dimensional random vector assigning K class labels to $\{1,\dots,d\}$. Let $P_{\mathcal{G}|\underline{Z}}$ be given by (6) and $q_{a,b}$, $r_{a,b}$ be the number of present and absent edges from nodes of class a to nodes of class b, respectively. Let $P_{\underline{Z}|\Sigma^{(d)}}$ be given by Lemma 4.1 and let $P_{\mathcal{G},\underline{Z}}$ satisfy (7). Then, $P_{\mathcal{G},\underline{Z}}(G,\underline{z})$ is given by*

$$\frac{1}{d!}\frac{\Gamma(\alpha)\alpha^K}{\Gamma(\alpha+d)}\Big(\prod_{k=1}^{K}(m_k!)\Big)^2 \prod_{1\le a\le b\le K} \frac{B(\beta_{a,b,1}+q_{a,b},\beta_{a,b,2}+r_{a,b})}{B(\beta_{a,b,1},\beta_{a,b,2})}. \tag{8}$$

Proof See Appendix 3.

4.3. Model inference

Let $\mathcal{G}_0$ denote a subset of $\mathcal{G}$, established on the basis of the observed interactions. At each stage of the experiment we find the DAG structure / classification $G_s, \underline{z}_s$ which maximizes posterior probability $P_{\mathcal{G},\underline{Z}|\mathcal{G}_0}$. This is done by the search and score algorithms described below.

Non-reversible MCMC search and score algorithms

When finding a pair with high posterior probability, the Metropolis-Hastings (M-H) algorithm is used as a method of finding a process that moves through the configurations $(G,\underline{z})$, in such a way that it favors configurations with high posterior probability. For each pair visited, we compute a score function

$$score(G,\underline{z}) = P_{\mathcal{G},\underline{Z}}(G,\underline{z}), \tag{9}$$

where G is consistent with the observations. The random walk is restricted to such graphs. The pair with the highest score is then chosen.

Following Corander, Gyllenberg and Koski (2006), we simplify the M-H algorithm by using acceptance probability $min\Big\{1,\frac{\pi'}{\pi}\Big\}$ of transition from state $(G,\underline{z})$ to state $(G',\underline{z}')$, where $\pi' := P_{\mathcal{G},\underline{Z}}(G',\underline{z}')$ and $\pi := P_{\mathcal{G},\underline{Z}}(G,\underline{z})$. This is computationally efficient and allows for using a transition mechanism, for which proposal probabilities do not have to be computed explicitly. The pseudo-code for the search and score algorithm with non-reversible M-H scheme is shown in the Algorithm 4.

Since we do not try to build up an empirical distribution, instead of this we score each configuration visited and choose the configuration of the maximal score, here it is unnecessary for the chain to have reversibility property, nor that the stationary distribution be equal to the target distribution. With this in mind, we present two M-H fashioned algorithms optimizing the score function. We construct two transition matrices, Q_1 and Q_2, which are relatively straightforward to compute and give the chain's reasonable mobility through the search space. To allow the Markov process to visit various classifications of the objects, in both algorithms, we pick a node from the graph and re-assign it according to the $CRP(\alpha)$. Then, we construct a temporary graph, consistent with the obtained classification.

Algorithms differ in the last step, which allows the Markov process to visit more graph structures. In the Q_1 transition kernel, we pick randomly two classes and then uniformly choose a node from each of the classes. The new graph contains an edge between the selected nodes with the probability of 0.5. At each iteration, all of the edges in one category are removed and only one edge might be added. As a result, for long-running simulations, the generated DAG structures become sparse. To avoid the above-described behaviour, in the Q_2 algorithm, we select randomly only one node and re-insert all incidence edges according to the Beta distribution.

$\mathbf{Q_1}$ transition kernel

The first three steps are common to both of the algorithms.

1. First, we choose node i at random according to uniform distribution and remove it. If the i-th node was in the class of its own - class k_i - we re-label classes $\{k+1,...,K\}$, $k+1 \to k$ for $k \geq k_i$.

2. We re-assign the i-th node according to the CRP(α). If there are currently K classes, we put it in the $K+1^{st}$ class with probability $\frac{\alpha}{\alpha+(d-1)}$ and in class k_j for $1 \leq k_j \leq K^{st}$ class with probability $\frac{m_{k_j}}{\alpha+(d-1)}$. If i is now in the class of its own, $(K+1)$, we choose uniformly distributed k_l in $\{1,...,K+1\}$ and label the class containing i by k_l. We re-label classes $k \geq k_l$ $k \to k+1$ (see Algorithm 1). This step allows the algorithm to evaluate different node classifications. The probability of introducing a new class in each iteration is controlled by the parameter α. The number of discovered classes determines the distance between nodes in the output graph. The more classes are discovered, the greater the distance between the nodes that can be achieved.

3. We construct a temporary graph $\tilde{G}$ to be consistent with the current classification of the nodes: we remove all edges between the i-th node and the nodes that are now in the same class. Then we redirect edges which now contradict the new class structure.

4. We create a new graph G' in the following way: We choose a pair of classes (a, b), $a < b$ at random, then we choose a node $j \in a$ and $l \in b$. For G', we select with, probability 0.5, a graph $\tilde{G}$ with or without an edge from j to l. This step allows the Markov process to visit more DAG structures and is computationally cheap. The corresponding pseudocode for the Q_1 proposal step is presented in Algorithm 2.

$\mathbf{Q_2}$ transition kernel

We modify the last step of the Q_1 algorithm. Instead of choosing two classes and modifying only one edge, we choose one class and select only one of its nodes: i, according to the uniform distribution. We draw an edge from the i-th node to any other node according to the Beta distribution used in the prior probability definition. This step increases computational complexity. However, the proposed transition kernel is is derived from the prior probability distribution and can lead to better results. The whole Q_2 transition is shown in the Algorithm 3.

The non-reversible M-H methods offer freedom in the choice of transition kernels. For other prior distributions, other proposal mechanisms may be designed.

Algorithm 1 Transition between categorization of nodes in the graph. Used in Q_1 and Q_2 proposal kernels.

```
function Reassign class labels($\underline{z}_{t-1}$)
    $\underline{z}' \leftarrow \underline{z}_{t-1}$
    $i \sim \mathcal{U}\{1, \dots d\}$                        ▷ Select one node $i$
    $k_i \sim \mathrm{CRP}(\alpha)$                           ▷ Reassign class of node $i$
    $\underline{z}'[i] \leftarrow k_i$
    if $\underline{z}'[i] = K + 1$ then                    ▷ $i$ is in the class of its own
        $k_l \sim \mathcal{U}\{1, \dots K + 1\}$              ▷ Shuffle class labels
        $\underline{z}'[i] \leftarrow k_l$
        for $k = 1, 2, \dots, K + 1$ do
            if $k \geq k_l$ then
                $k \leftarrow k + 1$
            end if
        end for
    end if
    return $\underline{z}'$
end function
```

Algorithm 2 Q_1 transition kernel

```
function $\mathbf{Q_1}(G_{t-1}, \underline{z}_{t-1})$
    $\underline{z}' \leftarrow$ Reassign class labels$(\underline{z}_{t-1})$
    $G' \leftarrow G_{t-1}$
    Redirect contradicting edges and remove edges between nodes of the same classes in $G'$
    Sample without replacement two class labels $a < b$
    Select uniformly node $j$ from class $a$ and node $l$ from class $b$
    Remove an edge $(j, l)$ if exists
    Draw an edge $(j, l)$ with probability 0.5
    return $(G', \underline{z}')$
end function
```

Algorithm 3 $\mathbf{Q_2}$ transition kernel

```
function $\mathbf{Q_2}(G_{t-1}, \underline{z}_{t-1})$
    $\underline{z}' \leftarrow$ Reassign class labels$(\underline{z}_{t-1})$
    $G' \leftarrow G_{t-1}$
    Redirect contradicting edges and remove edges between the nodes of the same classes in $G'$
    $i \sim \mathcal{U}\{1, \dots d\}$                                    ▷ Select one node $i$
    Remove all edges incident to the node $i$
    for $j = 1, 2, \dots, d$ do
        if $\underline{z}'[j] < \underline{z}'[i]$ then
            Draw $(j, i) \sim \text{Beta}(\beta_{1,\underline{z}'[j],\underline{z}'[i]}, \beta_{2,\underline{z}'[j],\underline{z}'[i]})$
        end if
        if $\underline{z}'[j] > \underline{z}[i]'$ then
            Draw $(j, i) \sim \text{Beta}(\beta_{1,\underline{z}'[i],\underline{z}'[j]}, \beta_{2,\underline{z}'[i],\underline{z}'[j]})$
        end if
    end for
    return $(G', \underline{z}')$
end function
```

Algorithm 4 Search and score with non-reversible M-H

```
1: Initialize $G_0, \underline{z}_0$
2: $G_s, \underline{z}_s \leftarrow G_0, \underline{z}_0$    ▷ Initialize solution
3: for $t = 1, 2, \ldots,$ MAX_ITER do
4:     $(G', \underline{z}') \leftarrow \mathbf{Q}(G_{t-1},\, \underline{z}_{t-1})$    ▷ Propose $(G', \underline{z}')$
5:     $u \sim \mathcal{U}(0, 1)$
6:     $\alpha \leftarrow min\Big\{1, \frac{P_{\mathcal{G},\underline{Z}}(G',\underline{z}')}{P_{\mathcal{G},\underline{Z}}(G_{t-1},\underline{z}_{t-1})}\Big\}$
7:     if $\alpha \leq u$ then
8:         $(G_t, \underline{z}_t) \leftarrow (G', \underline{z}')$    ▷ Accept $(G', \underline{z}')$
9:     else
10:        $(G_t, \underline{z}_t) \leftarrow (G_{t-1}, \underline{z}_{t-1})$    ▷ Reject $(G', \underline{z}')$
11:    end if
12:    if score$(G_t, \underline{z}_t) \geq$ score$(G_s, \underline{z}_s)$ then
13:        $(G_s, \underline{z}_s) \leftarrow (G_t, \underline{z}_t)$    ▷ Update solution
14:    end if
15: end for
16: return $(G_s, \underline{z}_s)$    ▷ Return the pair with the highest score
```

5. Numerical example

5.1. Introduction

In the present section of the paper, we provide an ample illustration for the performance of the algorithms proposed in terms of prediction. For this purpose, we use the experimental setup, which was described in Section 3 before. In the respective experiments, we run the MCMC algorithm both before and after a single interaction part of each phase of the procedure.

In order to capture the ability of the model to learn the causal theories, we investigate the predicted relations between the probe object x and all the other objects before and after the single interaction. This makes a slight difference with respect to the previously described experimental procedure, in which the participants were asked regarding only one reaction. In this manner, though, we obtain a direct and reliable measure of success. We establish a binary vector, representing the answers to the yes/no questions, concerning the predicted interactions at each stage. For the evaluation of the performance of the model we use F1 score, precision and recall metrics.

In this section, we present the algorithms' performance for prediction within the experimental setup described in Section 3. We run the MCMC algorithm before and after a single interaction part of each phase.

To capture the model's ability to learn causal theory, we investigate predicted relations between the probe object x and all other objects before and after single interaction. This differs slightly from the experimental procedure, where participants were asked about only one reaction. However, it provides a direct and reliable measure of success. We create a binary vector representing answers to yes/no questions about predicted interactions at each stage. We use F_1-score, precision, and recall to evaluate the model.

We run the computations for both Q_1 and Q_2 transition kernels. We vary input data: initial points (graphs and class assignment vectors), as well as model parameters α, $\underline{\beta_1}$ and $\underline{\beta_2}$. Subsection 5.3 describes the generation of the initial points. Subsections 5.4, 5.5 present the results for Q_1 and Q_2. Furthermore, we examined the predictions of the model once categories of objects are known and ran an MCMC algorithm which used the same score function, but moved only through the graph structures (subsection 5.6).

5.2. The data set

Data set in each phase $l = 1, \ldots, 7$ is formatted as an adjacency $3(l+1) \times 3(l+1)$-matrix of the graph of relations between objects. The subset of the entries of size $3l \times 3l$ is established from the previous observations.

5.3. Generation of initial points

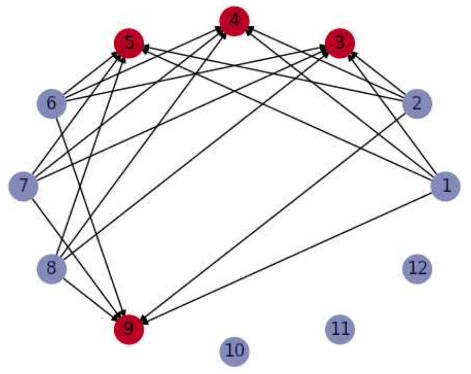
To investigate the impact of initial points on algorithm performance, we developed three methods of initial graph (G_0) generation. We will refer to nodes introduced in each phase of the experiment as *new nodes.* In all simulations, we assigned *new nodes* to one category.

(A) Firstly, we generate initial graphs with *new nodes* disconnected from other nodes.
(B) Secondly, we draw initial edges from each of the *new nodes* with the probability of 0.5 to every other node.
(C) Lastly, we generate initial graphs with the edges directed from a *new node* to any other node. The prior probability of G_0 is equal to 0, thus $score(G_0, \underline{z}_0) = 0$.

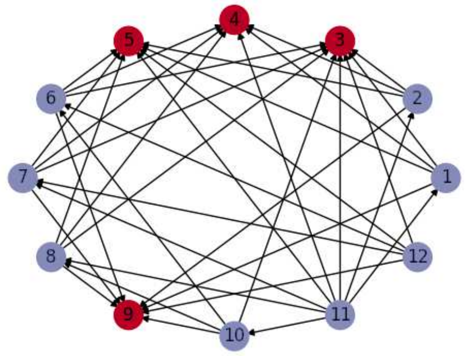
The initial points (G_0, z_0) with their scores in phase 3 are shown in Fig. 3.

5.4. Results for algorithm Q_1

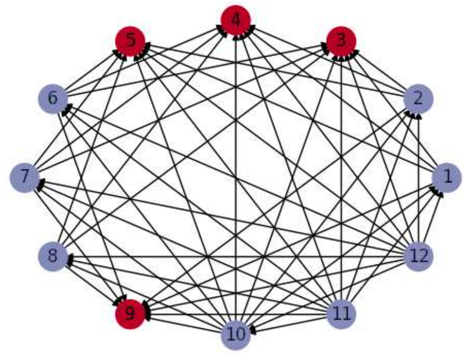
In this section, we present the results for the first choice of the transition matrix Q_1. We set model parameters to $\alpha = 1$, $\beta_{1,a,b} = \beta_{1,a,b} = 1$ for each pair (a, b) of the discovered classes. Figure 7 presents output graphs obtained in phase $l = 3$. Figure 4 shows the categorization of objects in each phase of the experiment for different initial points generation. Commonly, the objects introduced are assigned the same classification. This is to be expected; after the classification of the 'probe' object has been determined, the probe object's class now has greater weight than the other classes and hence greater probability according



(a) $score(G_0, \underline{z}_0) = 5.48e^{-16}$



(b) $score(G_0, \underline{z}_0) = 0$



(c) $score(G_0, \underline{z}_0) = 0$

Figure 3: Initial points (G_0, z_o) and their scores in phase $l = 3$. Violet objects belong to class 1 and red ones to class 2.

to the Chinese Restaurant Process. For initial points, generated by methods A and B, objects were divided into two categories. For initial points, generated by the method C, algorithm discovered three classes.

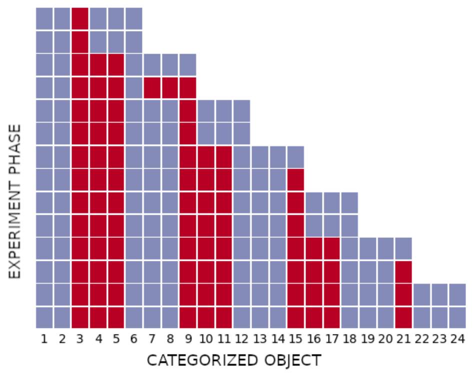
Figure 5 presents F_1, recall and precision in all phases of the experiment. F_1 mean values were significantly greater after single interaction at each stage. Having no information about relations between probe object x and other objects, leads to algorithm predictions being random and standard deviation greater. In phase 7 F_1 mean and standard deviation values were $\mu = 0.52$, $\sigma = 0.42$ before interaction and $\mu = 0.68$, $\sigma = 0.32$ afterwards. In phase 6 those values were equal to, respectively, $\mu = 0.08$, $\sigma = 0.25$ and $\mu = 0.72$, $\sigma = 0.33$.

We investigated the impact of the initial points choice (Fig. 6). The algorithm succeeded in predicting interactions with probe objects for the initial points, generated by A and B methods. In the last phase of the experiment for the method A, F_1 mean and standard deviation values were $\mu = 0.92$, $\sigma = 0.02$ and for the method B: $\mu = 0.83$, $\sigma = 0.12$. However, for the method C the algorithm performed poorly, with $\mu = 0.27$, $\sigma = 0.15$.

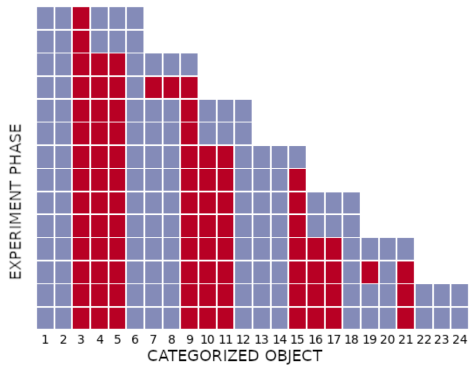
The model answers to all the post-test questions were correct in case when probe object was an activator in all phases (Table 2). Unlike the participants, the model was able to give yes/no answers to post-test questions. If the model learned the labelled DAG structure representing causal system it should give correct answers depending on the edge between the node representing object x and the nodes representing activator and reactor objects.

Question	Ideal	Phase number						
		$l = 1$	$l = 2$	$l = 3$	$l = 4$	$l = 5$	$l = 6$	$l = 7$
$x \in A$								
Will x activate?	No		No		No		No	
Will a activate?	No		No		No		No	
Will x activate?	No		No		No		No	
Will r activate?	Yes		Yes		Yes		Yes	
$x \in R$								
Will x activate?	Yes	No		No		No		No
Will a activate?	No	No		No		No		No
Will x activate?	No	No		No		No		No
Will r activate?	No	No		No		No		No

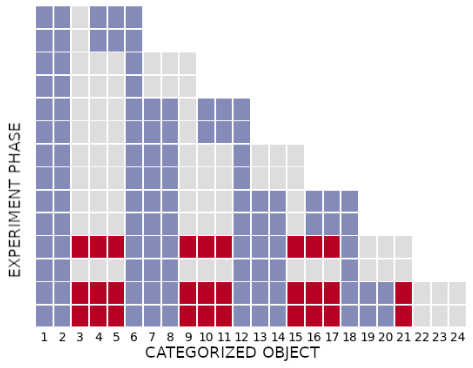
Table 2: Predictions in each phase for $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$, number of iterations $N = 2000$. In odd-numbered phases x is an activator, thus it does not activate when touched against both reactor object r and activator object a. Furthermore, object r will activate, but a will not. In even-numbered phases, x is a reactor. None of the objects a and r will activate, when touched against x. Probe object x will activate only when they touch against the activator object a



(a) Class assignments for initial points generated by the method A

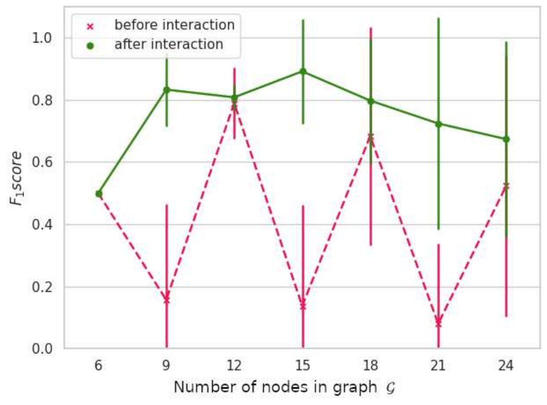


(b) Class assignments for initial points generated by the method B

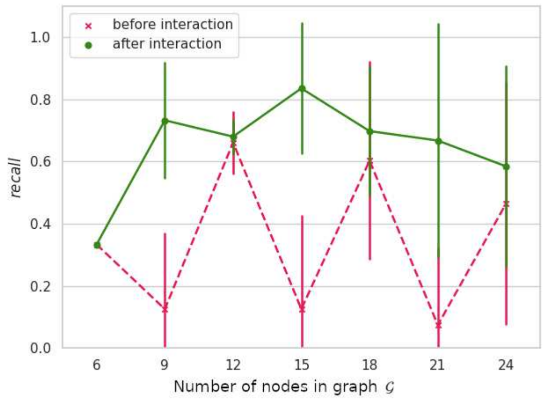


(c) Class assignments for initial points generated by the method C

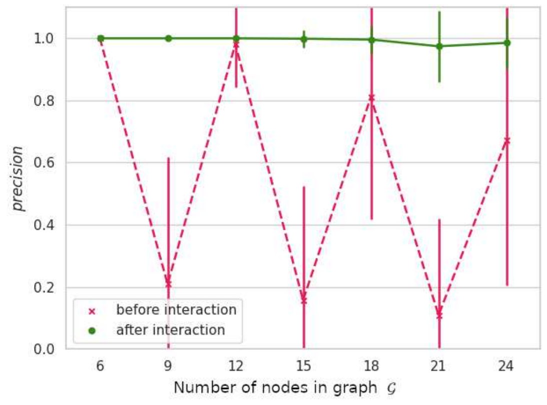
Figure 4: Class assignments of the nodes introduced in each phase (before and after single interaction) obtained by Q_1 algorithm, $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$ and number of iterations $N = 2000$. Violet objects belong to class 1, red to class 2 and grey to class 3



(a) F1

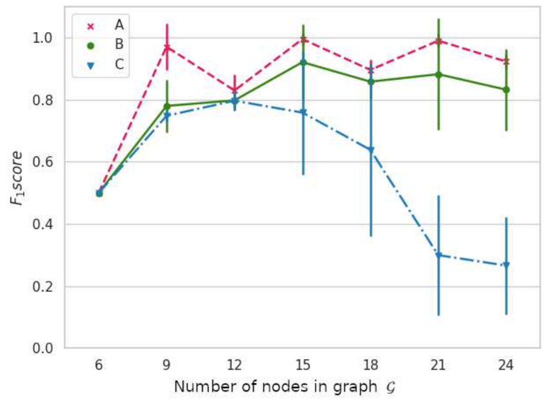


(b) Recall

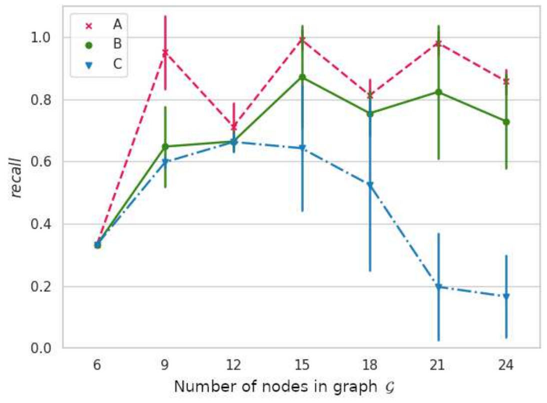


(c) Precision

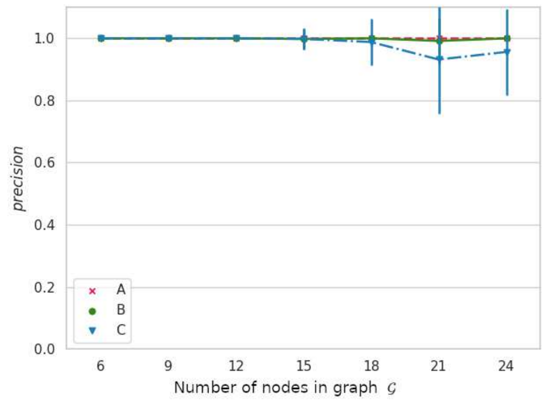
Figure 5: F_1, recall and precision before and after single interaction for Q_1 algorithm, $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$, $N = 2000$



(a) F1

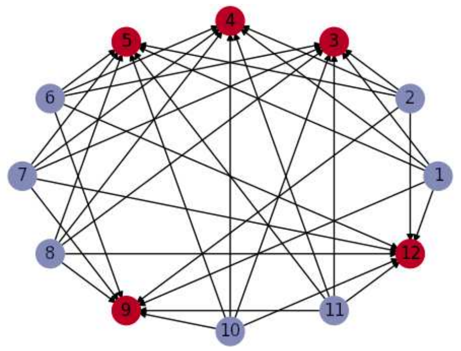


(b) Recall

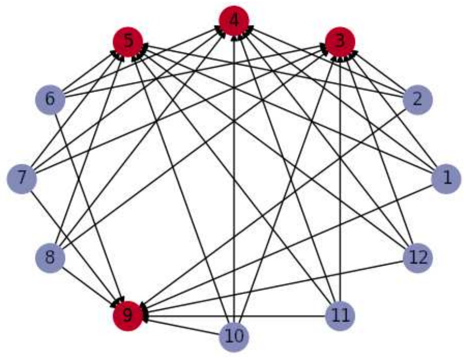


(c) Precision

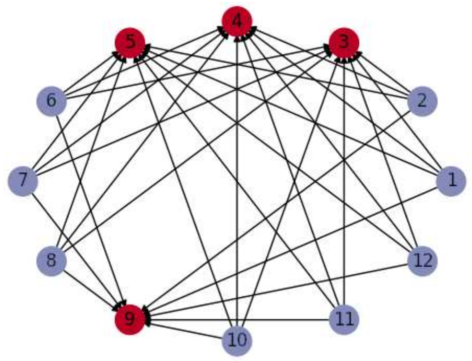
Figure 6: F_1, recall and precision for different methods of generation of initial points for Q_1 algorithm, $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$, $N = 2000$



(a) Output graph for the initial points generated by the method A



(b) Output graph for the initial points generated by the method B



(c) Output for the initial points generated by the method C

Figure 7: Output graphs in phase $l = 3$ for Q_1 algorithm, $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$, $N = 2000$. Violet objects belong to class 1, red ones to class 2

5.5. Results for algorithm Q_2

In this section, we present the results the for the second choice of transition matrix, Q_2. Item classification during all phases is shown in Fig. 8. For all three methods of generation of the initial points, the model distinguished three types of objects.

Figure 9 presents F_1, recall and precision for different graph sizes. As for the first Q_1, after single interaction F_1 mean values are greater, while standard deviations are significantly lower. In the last phase of the experiment F_1 mean and standard deviation values were $\mu = 0.76$, $\sigma = 0.29$ before interaction and $\mu = 0.88$, $\sigma = 0.03$ afterwards.

This time, the algorithm succeeded in discovering relations between objects regardless of the initial point generation method. In the last phase the F_1 scores were equal: $\mu = 0.91$, $\sigma = 0.03$; $\mu = 0.88$, $\sigma = 0.03$, and $\mu = 0.87$, $\sigma = 0.03$ for the initil points, generated by the methods A, B and C, respectively.

5.6. Fixed class assignment vectors

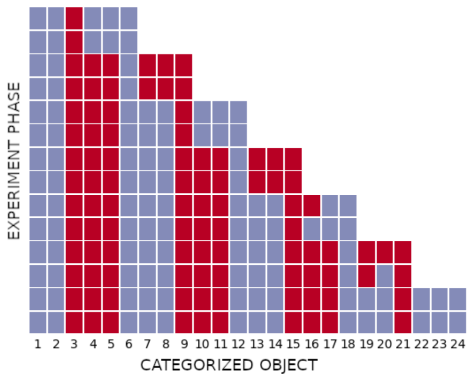
In this section, we investigate the algorithms' performance in the prediction task exclusively. We construct MCMC which moves only over graph structures, while the correct class assignment vector is fixed. Transition between the graphs is defined just as in the Q_1 algorithm. Given a graph G_1, we create a graph G_2 in the following way. We choose a pair of classes (a, b) $a < b$ at random, and we then choose node $j \in a$ and $l \in b$. For G_2 we select with the probability 0.5 a graph $\tilde{G}$ with or without a node from j to l.

The algorithm was outstandingly successful in prediction tasks in all phases of the experiment (Fig. 11).

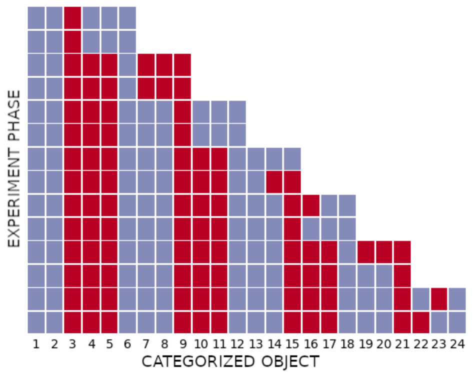
6. Score function convergence

Finally, we evaluated the computational efficiency of both proposed algorithms. We carried an in-depth analysis of score function behaviour. Figure 12 presents the trace plots of 300 Markov Chains for 3000 iterations in the seventh phase. The number of possible DAG structures is equal to 1.4×10^{103}. For Q_1 algorithm, score function converges to the value of the solution pair $G_s, \underline{z}_s$ after less than 500 iterations (Fig. 12). The trace plot for Q_2 shows a significantly wider range of sampled values.

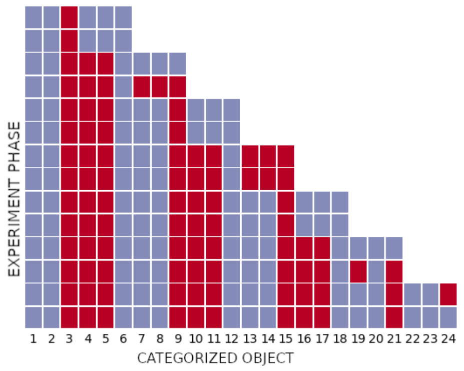
We benchmarked both proposed algorithms in scenarios with limited computational budget in the following experimental setup. For phases: 5, 7, 10 and 12 we performed 500 runs of each algorithm with the computational budget (maximal number of iterations) set to 5000. The main measure of efficiency was the number of iterations until algorithm reaches the prescribed accuracy ϵ,



(a) Class assignments for the initial points generated by the method A

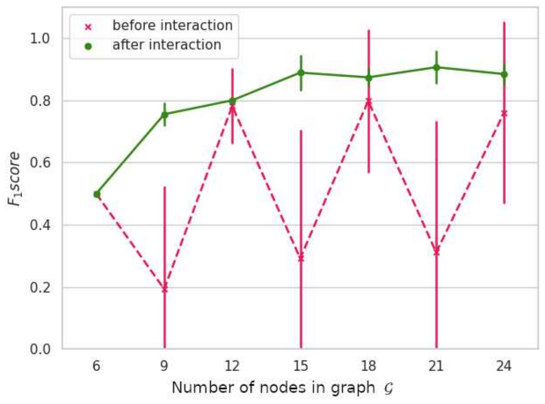


(b) Class assignments for the initial points generated by the method B

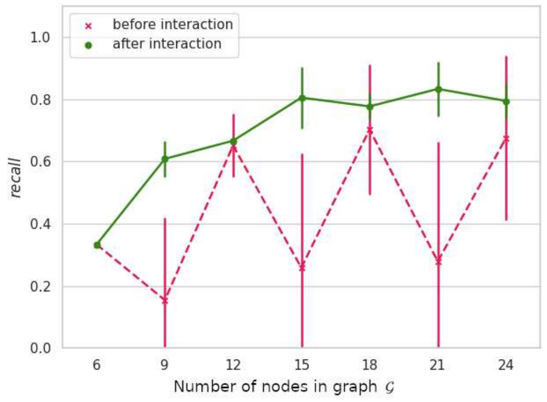


(c) Class assignments for the initial points generated by the method C

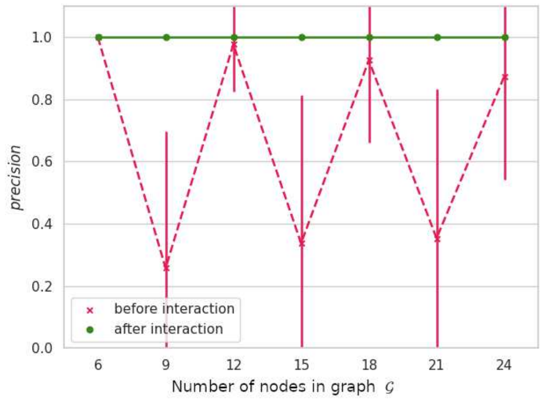
Figure 8: Class assignments of the nodes introduced in each phase (before and after single interaction) obtained by Q_2 algorithm, $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$ and number of iterations $N = 2000$. Violet objects belong to class 1 and red ones to class 2



(a) F1

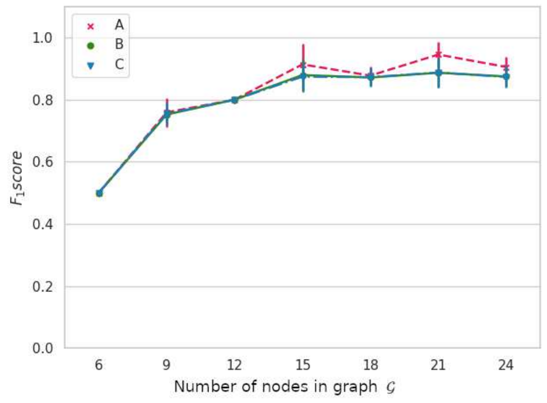


(b) Recall

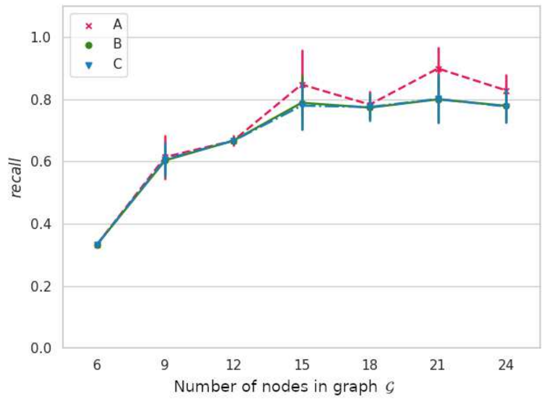


(c) Precision

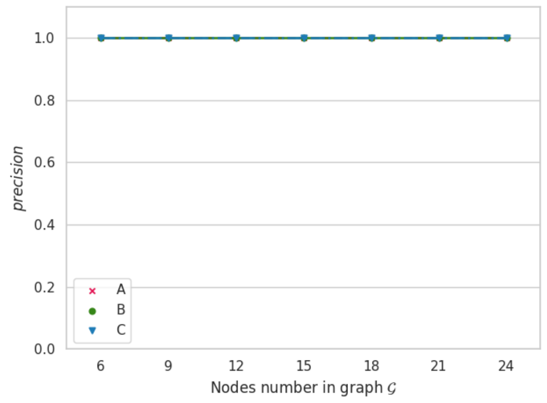
Figure 9: F_1, recall and precision before and after single interaction for Q_2 algorithm, $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$, $N = 2000$



(a) F1

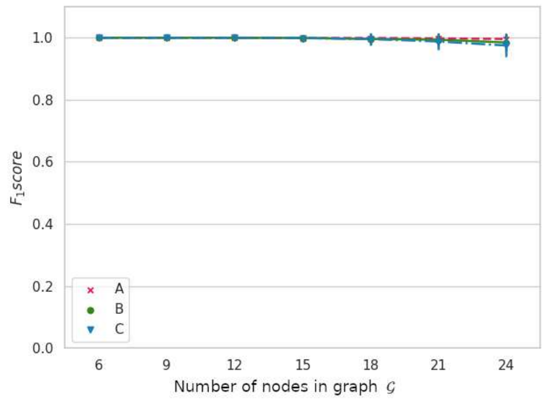


(b) Recall

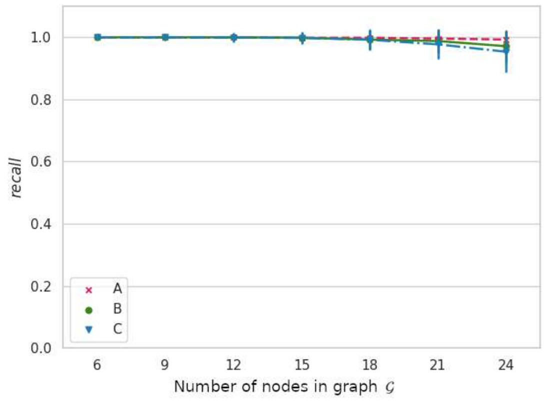


(c) Precision

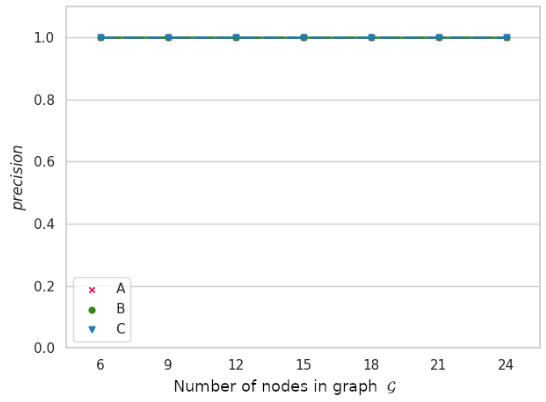
Figure 10: F_1, recall and precision for different methods of generation of initial points for Q_2 algorithm, $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$, $N = 2000$



(a) F1

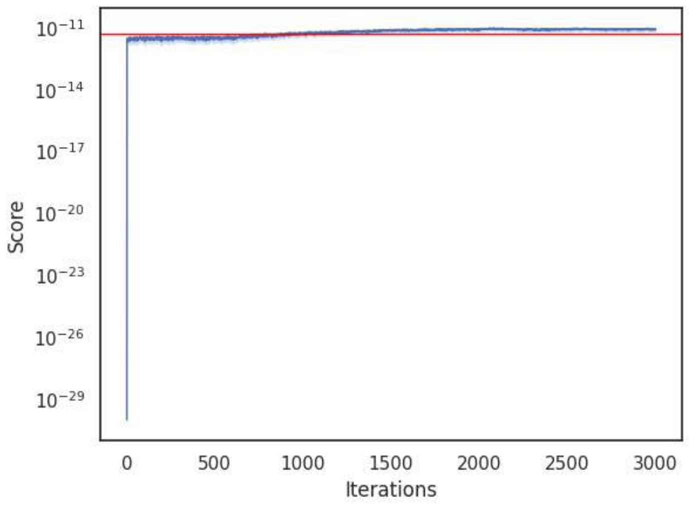


(b) Recall

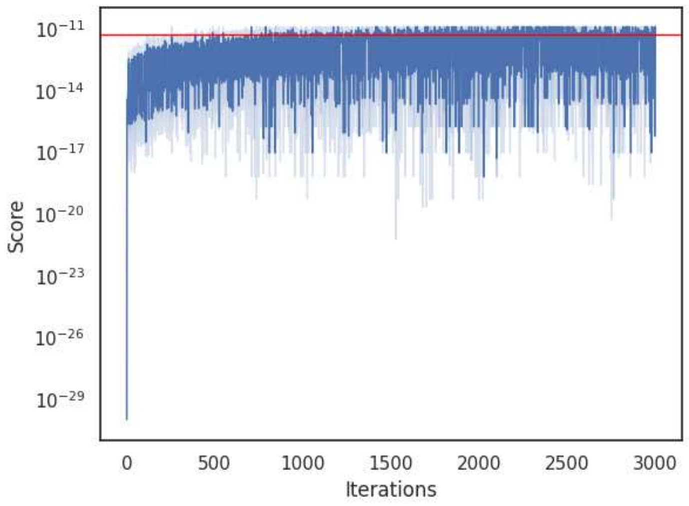


(c) Precision

Figure 11: F_1, recall and precision for different methods of generation of initial point for the algorithm moving only on graph structures (with fixed, correct $\underline{z}$), $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$, $N = 2000$



(a) Q_1 algorithm



(b) Q_2 algorithm

Figure 12: Score function values in phase $l = 7$. Score of the correct pair is equal to $5.16e^{-12}$. Traces are drawn for 300 Markov chains

meaning it outputs with the pair $(G, \underline{z})$ such that

$$score(G, \underline{z}) \in |score(G_s, \underline{z}_s) - \epsilon, score(G_s, \underline{z}_s) + \epsilon|. \tag{10}$$

Without loss of generality we set $score(G, \underline{z}) = \log_{10} P_{\mathcal{G},\underline{Z}}(G, \underline{z})$. This is done in order to simplify convergence analysis. However, with the exponential growth of the search space, the probabilities of visited graphs become very low. For larger problems, we suggest using the logarithmic score function to improve numerics. Following Hansen et al. (2016); Szynkiewicz (2018); Dolan and More (2001); More and Wild (2009) we use empirical cumulative distribution function (ECDF) (see Vaart, 1998) to display the proportion of trials which achieved accuracy $\epsilon \in \{1, 2, 3, 5, 8, 10\}$. Figures, depicting ECDF curves prove that the Q_1 algorithm outperforms the Q_2 algorithm in all the phases considered. Q_1 achieves the accuracy of 1 within less than 10 000 iterations even when the search space consists of 4×10^{191} graphs. We find these results particularly promising, and we are going to examine the application of Q_1 in discovering relations in larger data sets. Although the Q_2 algorithm succeeded in the prediction task, it struggled to achieve the prescribed accuracy of the score function. This raises the question about the choice of the scoring criterion and will be tackled in further research.

7. Conclusion

We described a hierarchical Bayesian model that discovers causal theory explaining given observations and predicting future interactions. This article aims at presenting two non-reversible Metropolis-Hastings algorithms, validating both of them on learning causal system similar to inductive theories, learned by humans, and evaluating their efficiency in scenarios with a limited computational budget. We believe that this research completes the state-of-art work. In further research, we will focus on the evaluation of the presented algorithm on other data sets and on comparison with other structure learning algorithms.

The discussed approach defines discovering theory as establishing the effect of searching through large space of relational systems and choosing the one with the highest score. Since such systems are often complex, it is crucial to provide scalable and efficient searching methods. This paper introduced non-reversible MCMC search and score algorithms, which visit causal systems represented as pairs of DAG structures and classification vectors of their nodes. We have also described an experiment about causal reasoning, where results may be interpreted in terms of model prediction. As mentioned in Section 3, the model is not specific for the causal system, which was the experiment's subject; therefore, it was not sensitive to some information accessible for participants. Still, only half of the participants were classified as those who learned causal theory. In further research, we will draw a more detailed comparison between machine learning and human inductive reasoning.

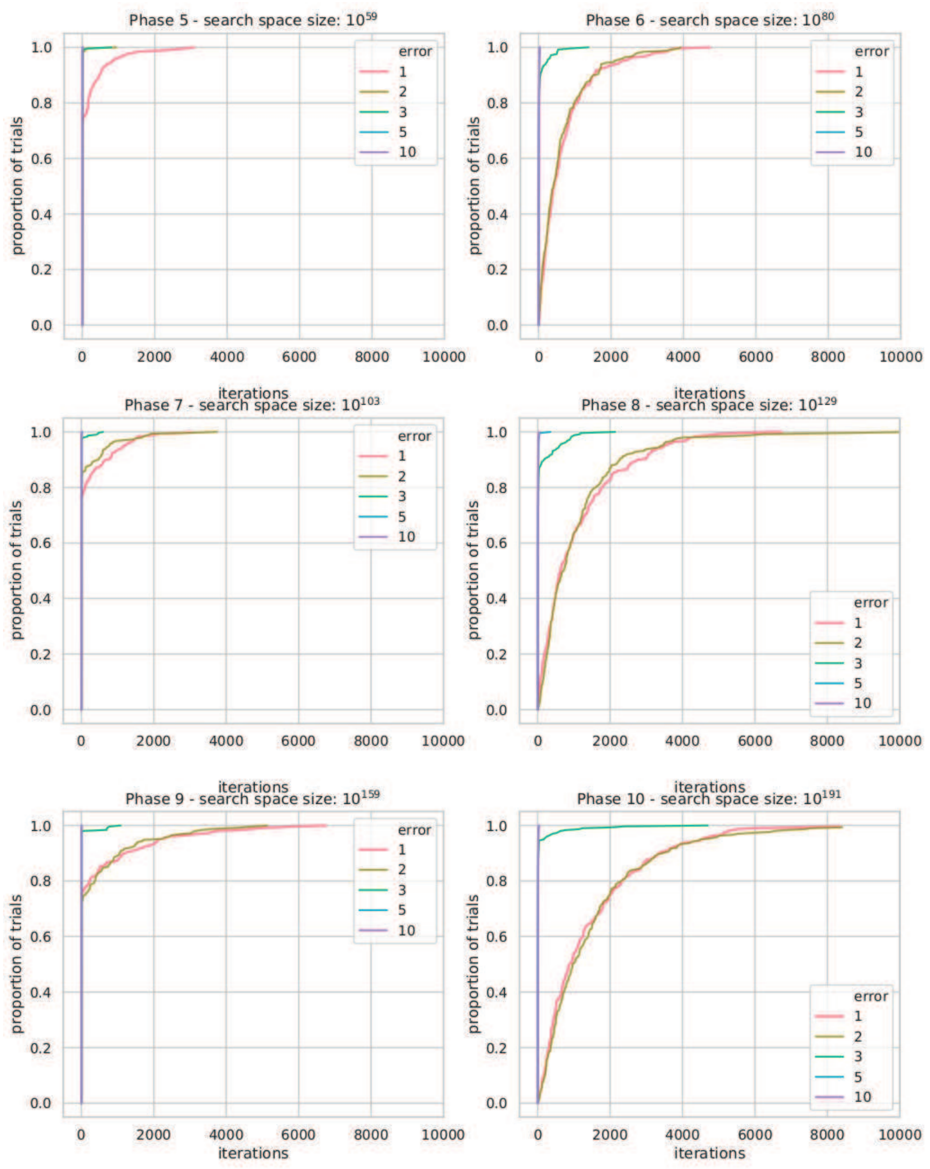


Figure 13: Empirical cumulative distribution function of runtimes of Q_1 algorithm in phases $l = 5, \ldots 10$ until it reaches the prescribed accuracy ϵ for $\epsilon \in \{1, 2, 3, 5, 8\}$ and $score(G, \underline{z}) = \log_{10} P_{\mathcal{G},\underline{Z}}(G, \underline{z})$. Model parameters set to $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$.

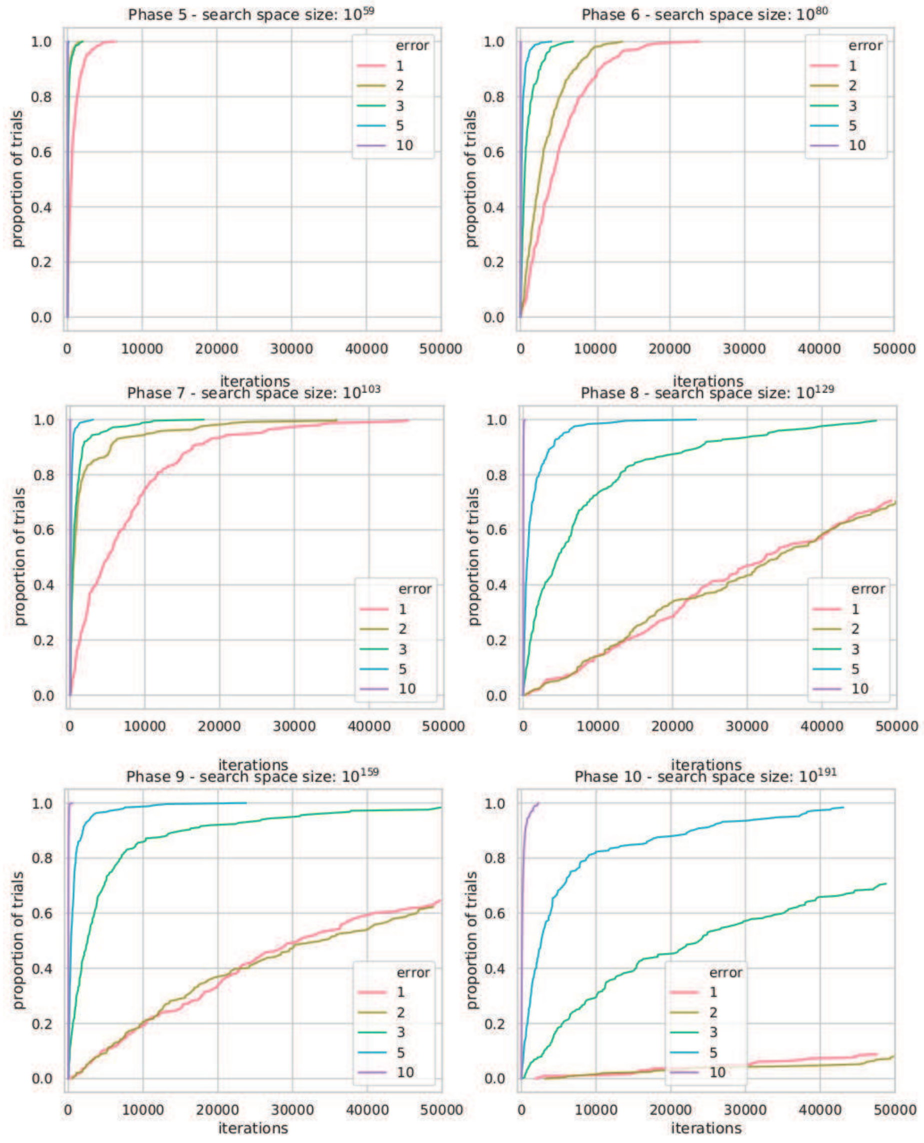


Figure 14: Empirical cumulative distribution function of runtimes of Q_2 algorithm in phases $l = 5, \dots 10$ until it reaches the prescribed accuracy ϵ for $\epsilon \in \{1, 2, 3, 5, 8\}$ and $score(G, \underline{z}) = \log_{10} P_{\mathcal{G},\underline{Z}}(G, \underline{z})$. Model parameters set to $\alpha = 1$, $\underline{\beta_1} = \underline{\beta_2} = \underline{1}$

To evaluate its ability to learn causal theory based on observations, we compared model predictions before and after single interaction between objects. For both algorithms proposed, F_1 values for yes/no questions about causal relations were significantly larger after single interaction. This demonstrates the capacity of knowledge acquisition. We explored the impact of the choice of the initial point for both algorithms. Q_2 algorithm succeeded for all of the initial points, while Q_1 performed poorly for one method of the starting points generation. In further research, we will address this problem and consider parallel computational strategies. We have also considered a problem of categorization of the input data. Algorithms classified correctly all of the objects introduced before. All of the nodes were grouped into two or three classes. Frequently, new objects were put into one category. We find these results promising. In this paper, we focused on a simple causal system. However, several applications of the discussed model, including learning ontologies and learning the structure of kinship systems, might be found in the literature (Kemp, Griffiths and Tenenbaum, 2004; Kemp et al., 2010). In further research, we will adopt this approach to discover dependencies between computational tasks in energy-aware allocation problems, described by Arabas (2019, 2021).

Furthermore, we focused on the predictive ability of the model in the simplified task. We assumed node classification to be known and constructed Markov chain moving only through graph structures. The algorithm's outstanding performance encourages us to apply it to larger data sets with more detailed classification. Finally, we carried out a numerical experiment illustrating the efficiency and scalability of the tested algorithms Q_1 and Q_2. Aggregated results revealed differences in the computational budget, required for score function to converge with a given accuracy. We find Q_1 performance promising enough to apply it to larger and more complex data sets. Although the Q_2 algorithm outperformed Q_1 in prediction tests, it struggled to achieve the given accuracy of score function within a limited computational budget. This observation brings up the question about the choice of the scoring criterion, which will be tackled. Regardless of that, further study will concern enhancing Q_2 performance with parallel strategies.

Appendix 1

LEMMA 4.1 *Let $\underline{Z}$ denote a d-dimensional random vector assigning class labels to each element of $V = \{1, \ldots, d\}$. Let $\Sigma^{(d)}$ be a permutation space of V and let $\sigma \in \Sigma^{(d)}$ be an ordering, in which we assign class labels to elements of V, according to CRP(α). Let $S_d = \{\underline{z} | z_{\sigma(1)} = 1, z_{\sigma(1)} \leq z_{\sigma(2)} \leq \ldots \leq z_{\sigma(d)}\}$. The distribution over $\underline{Z}$ for a given parameter α may be computed explicitly. If $\underline{z} \in S_d$*

$$P_{\underline{Z}|\Sigma^{(d)}}(\underline{z}|\sigma) = \alpha^K \frac{\Gamma(\alpha)}{\Gamma(\alpha + d)} \prod_{k=1}^{K} m_k! \tag{3}$$

for $K = 1, \ldots, d$, where K denotes the number of non empty classes and m_k counts the number of the nodes labelled as k by $\underline{z}$. Otherwise, $P_{\underline{Z}|\Sigma^{(d)}}(\underline{z}|\sigma) = 0$.

Proof If $\underline{z} \in S_d$, then $P_{\underline{Z}|\Sigma^{(d)}}(\underline{z}|\sigma)$ is equal to

$$P(Z_{\sigma(j+1)}{=}z_{\sigma(j+1)}|Z_{\sigma(j)}{=}z_{\sigma(j)}, \ldots, Z_{\sigma(1)}{=}z_{\sigma(1)}, \Sigma^{(d)}{=}\sigma)$$
$$= \frac{\alpha^K \prod_{k=1}^{K}(m_k^{(d)})!}{\prod_{j=0}^{d-1}(\alpha + j)} = \alpha^K \frac{\Gamma(\alpha)}{\Gamma(\alpha + d)} \prod_{k=1}^{K} m_k!. \tag{11}$$

Straightforwardly from the definition of CRP(α), if $\underline{z} \notin S_d$, then $P_{\underline{Z}|\Sigma^{(d)}}(\underline{z}|\sigma) = 0$. □

Appendix 2

Lemma 4.2 *Let $\mathcal{G}$ be a space of all possible DAG structures with d nodes and let $\underline{Z}$ be a d-dimensional random vector, assigning K class labels to $\{1, \ldots, d\}$. Let probability $P_{\mathcal{G}|\underline{Z},\underline{M}}$ be given by (5) for matrix $\underline{M}$, defined by eq.(4), then*

$$P_{\mathcal{G}|\underline{Z}}(G|\underline{z}) = \prod_{1 \le a \le b \le K} \frac{B(\beta_{a,b,1} + q_{a,b}, \beta_{a,b,2} + r_{a,b})}{B(\beta_{a,b,1}, \beta_{a,b,2})}, \tag{6}$$

where $q_{a,b}$ is the number of present edges from nodes of class a to nodes of class b, $r_{a,b}$ is the number of absent edges and B denotes the Beta function.

Proof Let $\psi_{ij} := \underline{M}_{ab}$ for $z_i = a$, $z_j = b$, and let

$$P_{\mathcal{G}|\underline{Z},\underline{M}}(G|\underline{z}, \underline{m}) = \prod_{i=1}^{d}\prod_{j=1}^{d} \psi_{ij}^{\underline{G}_{ij}} (1 - \psi_{ij})^{\underline{G}_{ij}} \tag{12}$$

for

$$\underline{G}_{ij} = \begin{cases} 1 & (z_i, z_j) \in E, \\ 0 & (z_i, z_j) \notin E, \end{cases} \tag{13}$$

where E denotes set of edges of graph G.

The conditioning on the array $\underline{M}$ may by removed by integration

$$P_{\mathcal{G}|\underline{Z}} = \int P_{\mathcal{G}|\underline{Z},\underline{M}} \pi_{\underline{M}|\underline{Z}}. \tag{14}$$

Therefore,

$$P_{\mathcal{G}|\underline{Z}}(G|\underline{z}) = \int \prod_{j=1}^{d}\prod_{i=1}^{d} \psi_{ij}(\underline{M})^{\underline{G}_{ij}}(1-\psi_{ij}(\underline{M}))^{1-\underline{G}_{ij}}$$
$$\cdot \prod_{a,b} \frac{1}{B(\beta_{a,b,1},\beta_{a,b,2})} \underline{M}_{ab}^{\beta_{a,b,1}-1}(1-\underline{M}_{ab})^{\beta_{a,b,1}-1} d\underline{\psi}$$
$$= \prod_{1\le a\le b\le K} \frac{1}{B(\beta_{a,b,1},\beta_{a,b,2})}$$
$$\cdot \int_0^1 \underline{M}_{ab}^{q_{a,b}+\beta_{a,b,1}-1}(1-\underline{M}_{ab})^{r_{a,b}+\beta_{a,b,2}-1}$$
$$= \prod_{1\le a\le b\le K} \frac{B(\beta_{a,b,1}+q_{a,b},\beta_{a,b,2}+r_{a,b})}{B(\beta_{a,b,1},\beta_{a,b,2})}. \tag{15}$$

□

Appendix 3

Lemma 4.3 *Let $\mathcal{G}$ be a space of all possible DAG structures with d nodes and let $\underline{Z}$ be a d-dimensional random vector assigning K class labels to $\{1,\ldots,d\}$. Let $P_{\mathcal{G}|\underline{Z}}$ be given by (6) and $q_{a,b}$, $r_{a,b}$ be the number of present and absent edges from nodes of class a to nodes of class b, respectively. Let $P_{\underline{Z}|\Sigma^{(d)}}$ be given by Lemma 4.1 and let $P_{\mathcal{G},\underline{Z}}$ satisfy (7). Then, $P_{\mathcal{G},\underline{Z}}(G,\underline{z})$ is given by*

$$\frac{1}{d!}\frac{\Gamma(\alpha)\alpha^K}{\Gamma(\alpha+d)}(\prod_{k=1}^{K}(m_k!))^2 \prod_{1\le a\le b\le K}\frac{B(\beta_{a,b,1}+q_{a,b},\beta_{a,b,2}+r_{a,b})}{B(\beta_{a,b,1},\beta_{a,b,2})}. \tag{8}$$

Proof From (7)

$$P_{\mathcal{G},\underline{Z}}(G,\underline{z}) = \sum_{\sigma\in\Sigma^{(d)}} \frac{1}{d!} P_{\mathcal{G}|\underline{Z}}(G|\underline{z}) P_{\underline{Z}|\Sigma^{(d)}}(\underline{z}|\sigma)$$
$$= \frac{1}{d!}\frac{\Gamma(\alpha)}{\Gamma(\alpha+d)}\#\{\sigma|\underline{z}\}\alpha^K \prod_{k=1}^{K}(m_k!)$$
$$\cdot \prod_{1\le a\le b\le K}\frac{B(\beta_{a,b,1}+q_{a,b},\beta_{a,b,2}+r_{a,b})}{B(\beta_{a,b,1},\beta_{a,b,2})}, \tag{16}$$

where $\#\{\sigma|\underline{z}\}$ is the number of the permutation, under which $\underline{z}$ can be obtained. This is equal to the number of permutation of nodes within the same category, which is $\prod_{k=1}^{K}(m_k!)$. Therefore, $P_{\mathcal{G},\underline{Z}}(G,\underline{z})$ is equal to

$$\frac{1}{d!}\frac{\Gamma(\alpha)\alpha^K}{\Gamma(\alpha+d)}(\prod_{k=1}^{K}(m_k!))^2 \prod_{1\le a\le b\le K}\frac{B(\beta_{a,b,1}+q_{a,b},\beta_{a,b,2}+r_{a,b})}{B(\beta_{a,b,1},\beta_{a,b,2})}. \tag{17}$$

References

Akaike, H. (1974) A new look at the statistical model identification. *IEEE Transactions on Automatic Control*, **19**(6): 716–723.

Angelino, E., Johnson, M. J. and Adams, R. P. (2016) Patterns of scalable bayesian inference. *Foundations and Trends in Machine Learning*, 9: 119–247.

Arabas, P. (2019) Energy aware data centers and networks: a survey. *Journal of Telecommunications and Information Technology*, 4: 26–36.

Arabas, P. (2021) Modeling and simulation of hierarchical task allocation system for energy-aware hpc clouds. *Simulation Modelling Practice and Theory*, 107: 102–221.

Bierkens, J. (2016) Non-reversible Metropolis-Hastings. *Stat Comput*, 26: 1213—1228.

Carey, S. (1985) *Conceptual Change in Childhood*. MIT Press.

Chickering, D. (2002) Optimal structure identification with greedy search. *Journal of Machine Learning Research*, 3: 507–554. doi: 10.1162/153244 303321897717.

Contaldi, C., Vafaee, F. and Nelson, P.C. (2019) Bayesian network hybrid learning using an elite-guided genetic algorithm. *Artificial Intelligence Review*, **52**(1): 245–272.

Cooper, G. F. and Herskovits, E. (1992) A bayesian method for the induction of probabilistic networks from data. *Mach. Learn.*, **9**(4): 309–347. ISSN 0885-6125. doi: 10.1023/A:1022649401552.

Corander, J., Ekdahl, M. and Koski, T. (2008) Parallel interacting mcmc for learning of topologies of graphical models. *Data Mining and Knowledge Discovery*, 17: 431–456.

Corander, P., Gyllenberg, M. and Koski, T. (2006) Bayesian model learning based on a parallel mcmc strategy. *Statistics and Computing*, 16: 355–362.

Dai, J., Ren, J. and Du, W. (2020) Decomposition-based bayesian network structure learning algorithm using local topology information. *Knowledge-Based Systems*, 105602.

Dolan, E. and More, J. (2001) Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91. doi: 10.1007/s1010 70100263.

Flores, M. J., Nicholson, A. E., Brunskill, A., Korb, K. B. and Mascaro, S. (2011) Incorporating expert knowledge when learning bayesian network structure: A medical case study. *Artificial Intelligence in Medicine*, **53** (3):181–204. doi: https://doi.org/10.1016/j.artmed.20 11.08.004.

Friedman, N. and Koller, D. (2001) Being bayesian about network structure: A bayesian approach to structure discovery in bayesian networks. *Mach. Learn.*, 50. doi: 10.1023/A:1020249912095.

Friedman, N., Nachman, I. and Pe'er, D. (1999) Learning bayesian network structure from massive datasets: The sparse candidate algorithm. In:

Proceedings of the Fifteenth Conference on Uncertainty and Artificial Intelligence. Morgan Kaufmann Publishers, 206–215. doi: 10.13140/2.1.1125.2169.

Gao, F. and Huang, D. (2020) A node sorting method for k2 algorithm in bayesian network structure learning. In: *2020 IEEE International Conference on Artificial Intelligence and Computer Applications (ICAICA)*, 106–110. doi: 10.1109/ICAICA50127.2020.9182465.

Goudie, R. J. B. and Mukherjee, S. (2016) A gibbs sampler for learning dags. *Journal of Machine learning Research*, **17**(2): 1–39.

Griffiths, T. L., Chater, N., Kemp, C., Perfors, A., Tenenbaum, J. B. and Goodman, N. D. (2010) Probabilistic models of cognition: exploring representation and inductive biases. *Trends in Cognitive Sciences*, 14: 357–364.

Hansen, N., Auger, A., Mersmann, O., Tusar, T. and Brockhoff, D. (2016) Coco: A platform for comparing continuous optimizers in a blackbox setting. *Optimization Methods and Software*, 36: 114–144.

Hastings, W. K. (1970) Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57:97–109.

Jones, M. and Love, B. C. (2011) Bayesian fundamentalism or enlightenment? On the explanatory status and theoretical contributions of bayesian models of cognition. *Behavioral and Brain Sciences*, 34:169–231.

Kemp, C., Griffiths, T. L. and Tenenbaum, J.B. (2004) Discovering latent classes in relational data. Technical report, MIT, CSAIL, 2004.

Kemp, C., Tenenbaum, J. B., Niyogi, S. and Griffiths, T. (2010) A probabilistic model of theory formation. *Cognition*, 114: 165–196.

Koller, D., Friedman, N., Getoor, L. and Taskar, B. (2007) Graphical models in a nutshell. In: L. Getoor and B. Taskar, eds., *Introduction to Statistical Relational Learning*. The MIT Press, 13–55.

Koski, T. and Noble, J. M. (2012) A review of bayesian networks and structure learning. *Mathematica Applicanda*, 40: 51–103.

Koski, T. J. T. and Noble, J. M., eds. (2009) *Bayesian Networks: an Introduction*. Wiley.

Lee, C. and van Beek, P. (2017) Metaheuristics for score-and-search bayesian network structure learning. In: *Canadian Conference on Artificial Intelligence*, 129–141. Springer.

Madigan, D. and York, J. (1993) Bayesian graphical models for discrete data. *International Statistical Review*, 63: 215–232.

Madigan, D., Andersson, S., Perlman, M. and Volinsky, C. (2000) Bayesian model averaging and model selection for Markov equivalence classes of acyclic digraphs. *Communications in Statistics: Theory and Methods*, 25. doi: 10.1080/03610929608831853.

Madsen, A. L., Jensen, F., Salmerón, A., Langseth, H. and Nielsen, T. D. (2017) A parallel algorithm for bayesian network structure learning from large data sets. *Knowledge-Based Systems*, 117: 46–55.

Mansinghka, V., Kemp, C., Tenenbaum, J. B. and Griffiths, T. L.

(2006) Structured priors for structure learning. In: *Proceedings of the 22nd conference on uncertainty in artificial intelligence* (UAI). AUAI Press, 324–331.

McClelland, J., Botvinick, M. M., Noelle, D. C., Plaut, D. C., Rogers, T. T., Seidenberg, M. S. and Smit, L. B. (2010) Letting structure emerge: connectionist and dynamical systems approaches to cognition. *Trends in Cognitive Sciences*, 14: 348–356.

Moore, A. and Wong, W.-k. (2004) Optimal reinsertion: A new search operator for accelerated and more accurate bayesian network structure learning. In: *Proceedings of the Twentieth International Conference on Machine Learning (ICML'03)*. AAAI Press, 552–559.

More, J. and Wild, S. (2009) Benchmarking derivative-free optimization algorithms. *SIAM Journal on Optimization*, 20: 172–191. doi: 10.1137/080 724083.

Murphy, G. and Medin, D. (1985) The role of theories in conceptual coherence. *Psychological Review*, **92**(3): 289–316. ISSN 0033-295X. doi: 10.1037/0033-295X.92.3.289.

Peters, G. (2008) Markov chain Monte Carlo: stochastic simulation for bayesian inference. *Statistics in Medicine*, **27**(16): 3213–3214. doi: 10.1002 /sim.3240.

Ravenzwaaij, D., Cassey, P. and Brown, S. D. (2018) A simple introduction to Markov chain Monte-Carlo. *Psychonomic Bulletin & Review*, 25: 143–154.

Rios, F., Noble, J. and Koski, T. (2015) A prior distribution over directed acyclic graphs for sparse bayesian networks. arXiv: 1504.06701

Rissanen, J. (1978) Modeling by shortest data description. *Automatica*, **14**(5): 465–471. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(78)90005-5.

Robinson, R. (1973) Counting labeled acyclic digraphs. In: F. Harary, ed., *New Directions in the Theory of Graphs*, 239–273. Academic Press, New York, NY.

Scanagatta, M., Salmeron, A. and Stella, F. (2019) A survey on bayesian network structure learning from data. *Progress in Artificial Intelligence*, 8: 425–439.

Schwarz, G. (1978) Estimating the dimension of a model. *Ann. Statist.*, **6**(2): 461–464. doi: 10.1214/aos/1176344136.

Silander, T., Leppä-Aho, J., Jääsaari, E. and Roos, T. (2018) Quotient normalized maximum likelihood criterion for learning bayesian network structures. In: *International Conference on Artificial Intelligence and Statistics*, 948–957. [Publisher ????]

Szynkiewicz, P. (2018) Comparative study of pso and cma-es algorithms on black-box optimization benchmarks. *Journal of Telecommunications and Information Technology*, 4: 5–17.

Tenenbaum, J. B., Kemp, C., Griffiths, T. L. and Goodman, N. D. (2011) Statistics, structure, and abstraction. *Science*, 331: 1279–1285.

v. d. Vaart, A. W. (1998) *Asymptotic Statistics. Cambridge Series in Statistical and Probabilistic Mathematics.* Cambridge University Press. doi: 10.1017/CBO9780511802256.

Wellman, H. M. and Gelman, S. A. (1992) Cognitive development: Foundational theories of core domains. *Annual Review of Psychology*, **43**(1):3 3 7–375. doi: 10.1146/annurev.ps.43.020192.002005. PMID: 1539946.