Causal Structure Learning for Dependency Analysis of Performance Data

Jan Lemeire  
ETRO lab  
Vrije Universiteit Brussel, Belgium  
jan.lemeire@vub.ac.be

Sam Maes  
COMO lab on Machine Learning  
Vrije Universiteit Brussel, Belgium  
sam.maes@vub.ac.be

Stijn Meganck  
COMO lab on Machine Learning  
Vrije Universiteit Brussel, Belgium  
stijn.meganck@vub.ac.be

Erik Dirkx  
ETRO lab  
Vrije Universiteit Brussel, Belgium  
erik.dirkx@vub.ac.be

Submitted for IPDPS’2006

Abstract

This paper proposes causal models for representing relational information about the variables of a performance analysis. Causal structure learning algorithms are used for constructing such models from experimental data. However, the existing algorithms had to be extended to capture the complexity of performance models, since they contain a mixture of continuous and discrete variables, nonlinear relations and deterministic relations. To handle the first two cases, we use a general conditional independence test based on the mutual information between probabilistic variables, where the underlying probability distribution of the experimental data is estimated by Gaussian kernel density estimation. Deterministic relations between variables make that these variables contain the same information about other related variables. We use the complexity of the relations as a criterion to decide upon which of the equivalent relations is the direct relation. Experiments with the aztec benchmark application for solving linear equations in parallel gave accurate models, providing insight in how each variable affects the overall performance.

1. Introduction

The design and implementation of high-performance applications requires knowledge of many factors that influence the performance. This task can be facilitated by tools that automate the performance analysis and offer multifunctional performance models. The models should fulfill many requirements, such as provide information on the expected performance, offer insight in the causes of performance degradation, should be constituted of reusable submodels, tell which variables should be known for predicting others, should make it possible to estimate the effects of optimizations and reason under uncertainty. Causal models offer a formalization of these properties. A causal model represents the independency relations among variables, it reduces a model into independent submodels in which the directly related variables of a variable contain all information about that variable. Furthermore, a causal model aims at representing the underlying physical mechanisms that generated the data, which makes it possible to reason about modifications - called interventions - like algorithm or system optimizations.

This paper reports on experiments with existing learning algorithms for finding the causal structure of performance variables from experimental data. Current tools that support multiple experiment analysis plot performance variables (SCALEA [11]) and inefficiencies (Aksum [3]) in function of application and system parameters. Others provide regression analysis (AIMS [15]). Our approach provides support for extending current tools and automate certain tasks of the performance expert, like getting insight in the effect of design decisions, detecting unexpected dependencies, testing independency assumptions and finding the reasons of outliers.

A causal model represents the conditional independence relations among the variables by a directed acyclic graph. The correspondence of the conditional independencies in the graph and the data is called faithfulness. Causal structure learning algorithms are based on the conditional independencies found in the experimental data. Performance data is however more complex than the data typically encountered in research about causal analysis: it contains a mixture of continuous and discrete variables and the re-
lations between the variables are not always linear, as assumed in most research. All these aspects are handled by the extensions we developed and integrated into the Tetrad tool [9], developed at the Dept. of Philosophy of Carnegie Mellon University. To measure dependencies, independent of the form of the relation, we applied the general information-theoretic concept of mutual information of two stochastic variables [2]. It is based on the entropy of a stochastic variable. We used a Gaussian kernel density estimator to estimate the underlying probability distribution from the experimental data. The learning algorithms also fail on data containing deterministic relations since it violates the faithfulness assumption. Two variables that are deterministically related contain equivalent information about other variables and conditional independencies does not provide a way of deciding upon which is the direct relation, since both variables become conditional independent by conditioning on the other variable. We propose to choose the less complex relation. This is motivated by the well-known principle of Ockham’s Razor, which states that if multiple hypotheses can explain the data, we should choose the simplest one.

Causal performance models are defined in Chapter 2, the structural learning algorithms and the extensions implemented by us are explained in chapter 3. Chapter 4 describes the experimental setup and discusses the experimental results from a performance analysis of the Aztec benchmark parallel application.

2. Causal Performance Models

This chapter will briefly introduce causal models, that are a special kind of Bayesian models. See [7, 8, 10] for a complete theoretic elaboration.

2.1. Bayesian Models

Bayesian models are defined with the properties of correlation and the Markov condition. Two stochastic variables $X$ and $Y$ with distributions $P(X)$ and $P(Y)$ are probabilistically independent when the joint distribution $P(X,Y)$ can be decomposed as $P(X,Y) = P(X).P(Y)$. On the other hand, $X$ and $Y$ being dependent implies that by knowing the value of one variable, something is known about the other. In information-theoretic terms [2], due to a dependency, the conditional entropy of the unknown variable $H(Y | X)$ decreases and the mutual information $I(Y ; X) = H(Y) - H(Y | X)$ between both variables is bigger than zero. Both properties are defined in section 3.1. The Markov condition states that all variables get independent from a variables by conditioning on the parents of that variable. Take the model $A \rightarrow B \rightarrow C$ for instance.

![Figure 1. Medical causal model](image)

When the state of $B$ is known, $A$ contains no more information about $C$ and $C$ gets conditionally independent from $A$. Appearance of both properties reduces the complexity of the joint distribution, in which in general all variables are dependent from each other. The independencies can be represented by a directed acyclic graph (DAG) which consists of a set $V = \{V_1, \ldots, V_n\}$ of vertices, representing variables of interest, and a set $E$ of directed edges that connect the vertices and represent probabilistic relations. Pearl developed a method, called $d$-separation, to retrieve all independence relations of a DAG [4].

Fig. 1 shows an example of a medical model, in which diseases (middle row) are influenced by environmental factors (top row) and produce symptoms (bottom row). A Bayesian model offers a reduced description of the joint distribution. The distribution of each variable can be calculated from the distributions of its parents alone. From Fig. 1, the distribution of cancer is $P(\text{Cancer}) = P(\text{Pollution}).P(\text{Smoker}).P(\text{Cancer} | \text{Pollution}, \text{Smoker})$. A Bayesian model thus consists of a DAG and the conditional independency distributions.

2.2. Causal Models

A causal model interprets all relations of the Bayesian model to be of causal nature. A causal relation represents a local and autonomous physical mechanism, which is conceivable to change one relationship without changing the others. Contrary to the symmetrical correlation, it is an asymmetrical relation, but which can only be observed after an intervention. [7] defines an intervention as ’surgically’ setting a variable to a certain state, resulting in a mutilated model in which all links from the parents are removed. Hence, the asymmetry comes into play, since only the effects remain connected to the variable. For the model of Fig. 1, if in an experiment, a person would intentionally being ’intoxicated’ by the virus to get bronchitis, the relations with the environmental factor Smoker would be cut and only the effects of the disease would show a correlation. Causality thus implies modularity which en-
Figure 2. General model of parallel performance

ables us to predict the effects of specific modifications of the model. Moreover, due to the modularity properties of a causal model, causal models allow a reductionistic approach in which a variable can be studied with its parents alone.

2.3. Causal Performance Models

The model of parallel performance shown by Fig. 2 is constructed by an expert in an intuitive way, for what he expects as being a true and useful model. It shows the most common variables involved in the performance analysis of a parallel application. The worksize of the application and the number of processors \#processors are very general input parameters. Other system parameters are depicted by gray ovals. The performance variables at the right side are the output of the model. Intermediate variables like the communication time $T_{\text{comm}}$ and idle time $T_{\text{idle}}$ measure the overheads that can explain poor performance results. Other variables on their turn can explain the values for the overheads. Most of them can be measured, in the application (like the number of basic operations \#operations$_{\text{comp}}$ of the algorithm), in the system (like the cache misses) or by the MPI profiling capabilities (like the size of the communication data or the number of messages). Finally, the number of instructions per operation \#instr$_{\text{op}}$, the partitioning and the communication scheme denote characteristics of the application.

By analyzing of the properties attributed to this model, we find that they correspond with the definition of a causal model. We expect a variable to be determined by its parents, this corresponds with the Markov condition. We assume the model for not containing redundant relations (indirect relations), thus being minimal. Furthermore, the paths between the variables show the dependencies among the variables. This corresponds to the faithfulness condition. In the approach used by TETRAD [8], the 3 properties - the Markov condition, minimality and faithfulness - are taken as axioms for defining a causal model. With these properties, which can be measured experimentally, the model is assumed to come close to the reality and contain causal information.

3. Causal Structure Learning

Besides the formal treatment of causality by Pearl, another important advance in the field of causality was the construction of algorithms for learning causal models from experimental data by [13, 8]. Various tools exist that implement these algorithms, for an overview see [5] or (<http://www.cs.ubc.ca/~murphyk/Software/BNT/bnsoft.html>). We chose TETRAD (free available at <http://www.phil.cmu.edu/projects/tetrad/>), because it is open source software written in Java and contains an extensive set of algorithms. There are two types of structure learning algorithms: constraint-based and scoring-based algorithms. All algorithms of TETRAD are based on the work of [8] and are of the constraint-based type. The algorithms consist of two parts. First, they construct an undirected graph by finding direct relations. This part is the same for all algorithms. In the second part, the algorithms try to direct the edges using orientation rules. We are mainly interested in the first step, since the direction of the edges is quite trivial in performance models. Moreover, the knowledge of the input and output variables will help to direct the edges and any confusion about the direction can be resolved by knowledge added by an expert. Therefore, we will discuss the first step of the algorithm, called adjacency search, the detection of direct relations. It is based on the property that direct relations cannot become probabilistically independent upon conditioning on some other set of vertices (see the Markov condition). On the other hand, indirectly related variables become independent by conditioning on some variables on the path between both variables. For deciding upon an edge in the graph, the algorithm will go through all subsets of variables and check for conditional independency. If a test is successful, the edge is removed. The algorithm start by checking unconditional correlations and then gradually adds nodes to the conditioning set. It selects the nodes in an optimized way to minimize the tests it has to perform.

Under 6 assumptions the algorithm will find the correct equivalence class of indistinguishable equivalent causal models. Model of the same class have the same undirected
For measuring the entropy of continuous variables, we will use the entropy of the discretized distribution. For a variable \( X \) with density \( f(x) \) and dividing the range of \( X \) in bins of length \( \Delta \), the entropy of the quantized distribution is (Cover 1991, p228)

\[
H(X^\Delta) = -\int_{-\infty}^{+\infty} f(x) \Delta \log(f(x)\Delta)
\]

The errors induced by the quantification will be small, since the estimated distributions will be smooth by construction, as will be explained in the next section. The discretized version of entropy is of the same form as Eq. 1. This approach makes it possible for handling continuous and discrete variables in a similar way. \( I(Y;X) \) sums for every \( x \) the decrease in uncertainty of \( y \). Each value of \( x \) is considered independently, where correlation measures a relation between all points (for all values of \( x \)). For mutual information, the relation between \( x \) and \( y \) can be arbitrary. The disadvantage of this test is that more data points are needed. Actually, for every \( x \) value, multiple data points are needed. This would be the same as discretizing the continuous variable and would be a great disadvantage. Kernel density estimation overcomes these problems: the number of data points needed can be limited because \( I(Y;X) \) in \( x \) is influenced by its neighboring points by assuming a smoothly changing distribution.

3.2. Kernel Density Estimation

For applying the information-theoretic concept of dependency, it is necessary to have an estimation of the probability distributions from the data. For discrete variables the distribution can be estimated by simply counting the occurrences of the values and divide the counts by the number of data points. For continuous variables, we chose kernel density estimation for estimating the distributions. The idea is to construct the distribution by spreading the information of each data point over the neighboring points. See http://www.maths.uwa.edu.au/~duoKt/seminars/intro2kde/ for a gentle introduction. The distribution is the result of a convolution of the data points with a well-chosen kernel [14]:

\[
p(x) = \frac{1}{nb} \sum_{i=1}^{n} K\left(\frac{x-x_i}{b}\right)
\]

with \( n \) the number of data points \( x - i \) and \( K(\cdot) \) the multivariate kernel function, which is symmetric and satisfying \( \int k(x)\Delta x = 1 \). The factor \( b \) is the smoothing bandwidth and determines the width of the Gaussian curve. Theoretic analysis and simulation have shown that the choice of the kernel is not crucial, but that bandwidth is the determining factor for having good estimations. We chose the Gaussian
function for kernel and for the bandwidth the 3 times the range divided by the number of data points.

For estimating a multivariate distribution, a multidimensional kernel is used with a specific bandwidth in each dimension. The independence test so created could easily be plugged in into the TETRAD tool.

3.3. Deterministic Relations

A variable $X$ is functionally related to a set of variables $(Y_1, Y_2, \ldots, Y_n)$, denoted by $Y$, if $X = f(Y)$ is a function. This implies that the variables $Y$ contain all information about $X$, and the conditional entropy of $X$ becomes 0, due to the knowledge of $Y$. This implies, that $X$ gets conditionally independent from any variable by conditioning on $Y$:

$$I(Z; X | Y) = H(X | Y) - H(X | Y, Z) = 0 \quad (8)$$

for any $Z$. The second term is also 0, since conditioning never increases entropy. Thus

$$0 \leq H(X | Y, Z) \leq H(X | Y) = 0 \quad (9)$$

The result of the deterministic relation is that $X$ cannot contain additional information about any other variable. The result is that some edges are removed erroneously. This happens when 2 variables $X$ and $Y$ contain equivalent information about $Z$, both $I(Z; X | Y) = 0$ and $I(Z; Y | X) = 0$. $Z$ and $X$ become independent by conditioning on $Y$ and $Z$ and $Y$ by conditioning on $X$. The adjacency search procedure, described in the previous section, will fail, since both edges will be removed. No path between $Z$ and both variables remain, but they clearly contain information about $Z$. We call both edges equivalent, since they represent the same amount and the same kind of information. We introduce the criteria of uncertainty and complexity to decide upon which edge represents the direct relation. By the first criterion, we choose the variable with the least entropy. For example, if $X = f(Y)$ is a function, the entropy of $X$ is smaller than that of $Y$. By the second criterion, we prefer simpler relations. In the case of a choice between a discrete and a continuous variable, the relation with the continuous variable can be modeled by an analytical function, whereas the relation with the discrete variable is described by discrete probability values. If both variables are continuous, a regression analysis is applied to measure the complexity of the relations. This criterion is valid if we assume that the complexity of relations increase along a causal path. Take the model $A \rightarrow B \rightarrow C$, the relation between $A$ and $C$ will in general be more complex than the relations between $A$ and $B$ or between $B$ and $C$. The adjacency search procedure was modified for finding equivalent edges by comparing conditional independence tests. An extra step was added to the algorithm in which the direct edge among the equivalent edges was decided.

4. Experiments

4.1. Experimental Setup

We will validate our approach by applying the causal structure learning on data retrieved from experiments with the Aztec benchmark library [12](http://www.cs.sandia.gov/CRF/aztec1.html). It provides an iterative algorithm for parallel solving of partial differential equations, defined over a 3-dimensional grid. It supports 2 sparse matrix formats, a point-entry modified sparse row (MSR) format and a block-entry variable block row (VBR) format. Experiments are run on a dedicated cluster of 8 PentiumII computers connected by a 100MHz non-blocking switch. The number of equations is varied between 100 and 400 and the number of grid points between $5^3$ and $20^3$. The same experiments are performed for both matrix formats. Our performance analysis tool EPPA [6](http://parallel.vub.ac.be/eppa) uses the MPI profiling facilities to automatically trace the MPI calls and writes them to a database. A small tool was developed to retrieve data from the database, write it to a TAB separated file which we load as mixed data into TETRAD 4.3. We then applied the PC algorithm with default options onto our data.

4.2. Overall performance

Fig. 3 shows the model of Aztec’s performance constructed by TETRAD. The variables are ordered from input to output, starting with the 4 input parameters at the left and the parallel runtime and speedup at the right. It is immediately clear that the direction of the relations are not correct, since we expect them to go from left to right. As explained before, this is not our main concern. At the time of writing the paper, the knowledge of the input and output variables could not be inserted yet and the effects of deterministic relations on the orientation rules were still unclear. The undirected graph that was learned is more interesting, since it shows how the performance is generated. To analyze this in depth, 2 variables were registered which the algorithm returns at the end of the execution, namely the total number of flops needed for the computation (totalFlops) and the number of iterations (totalIterations) that the algorithm performed. totalIterations is completely determined by the number of grid points and does not affect the performance in a direct way. This can be understood by the fact that totalFlops incorporates all information, except nbrProcessors, about the runtime, communication time $T_{\text{comm}}$ and idle time $T_{\text{idle}}$. We added an additional variable eqXpoints which is simply the product of nbrEquations and nbrGridPoints. This is a useful variable since it explains how totalFlops depends on
nbrEquations and nbrGridPoints. totalFlops is much bigger for the VBR matrix format as for the MSR format. Finally, the speedup depends on the number of processors, the computation, communication and idle time.

4.3. Communication performance

Fig. 4 shows the learned model of the communication performance. The 4 input parameters are at the left side and the total communication time $T_{comm}$ at the top right. The figure shows the result of the adjacency search step only, we omitted the orientation step because the results were unreliable as in the previous model. Nevertheless, the undirected graph shows how $T_{comm}$ is generated, when we imagine the relation going from left to right and down to top. The communication performance is completely defined by the number of processors, the number of messages and the size of the communicated data. We registered 4 variables that are calculated by the algorithm when the partitioning of the matrices is finished. The internalUnknowns are the elements that can be updated using only information on the current processor. externalUnknowns refers to the off-processor elements that are required during the calculations by the borderUnknowns elements. unknownsSentToNeighbors represents the number of elements actually sent. These definitions confirm the model. The communication is primary affected by unknownsSentToNeighbors, which on its turn is influenced by borderUnknowns and this by externalUnknowns. The VBR matrix format generates more messages than the MSR format. The variable eqXpoints, which is the product of nbrEquations and nbrGridPoints, determines internalUnknowns in combination with the number of processors. On the contrary, the relation of externalUnknowns with nbrEquations

5. Conclusions

Causal modeling provides a theoretical framework for capturing relational regularities in performance data. Algorithms can detect these regularities and construct causal models from experimental data and offer a founded way of dividing a complex model into independent submodels. We enlarged the scope of existing structure learning algorithms by using the form-free information-theoretic concept of mutual information and by introducing the complexity criterion for selecting direct relations among equivalent relations. Experiments with the Aztec benchmark application show that accurate models are learned. Causal modeling make explicit what is done by the scientist when analyzing performance and allow further automation.

References

mance problems in parallel and distributed programs by us-


