Parallel Importance Sampling in Conditional Linear Gaussian Networks*

Antonio Salmerón[1], Darío Ramos-López[1], Hanen Borchani[4], Ana M. Martínez[4], Andrés Masegosa[2], Antonio Fernández[1], Helge Langseth[2], Anders L. Madsen[3,4], and Thomas D. Nielsen[4]

[1] University of Almería, ES-04120 Almería, Spain,
{antonio.salmeron,drl102,afalvarez}@ual.es
[2] Norwegian University of Science and Technology, NO-7491 Trondheim, Norway,
andres@cs.aau.dk, helgel@idi.ntnu.no
[3] Hugin Expert A/S, DK-9000 Aalborg, Denmark,
anders@hugin.com
[4] Aalborg University, DK-9220 Aalborg, Denmark,
{hanen,ana,tdn}@cs.aau.dk
http://www.amidst.eu

Abstract. In this paper we analyse the problem of probabilistic inference in CLG networks when evidence comes in streams. In such situations, fast and scalable algorithms, able to provide accurate responses in a short time are required. We consider the instantiation of variational inference and importance sampling, two well known tools for probabilistic inference, to the CLG case. The experimental results over synthetic networks show how a parallel version importance sampling, and more precisely evidence weighting, is a promising scheme, as it is accurate and scales up with respect to available computing resources.

Keywords: Importance sampling, variational message passing, Conditional Linear Gaussian networks, hybrid Bayesian networks

1 Introduction

Today, omnipresent sensors are continuously providing streaming data on the environments in which they operate. For instance, a typical monitoring and analysis system may use streaming data generated by sensors to monitor the status of a particular device and to make predictions about its future behaviour, or diagnostically infer the most likely system configuration that has produced the observed data. Sources of streaming data with even a modest updating frequency can produce extremely large volumes of data, thereby making efficient and accurate data analysis and prediction difficult. One of the main challenges is related to handling uncertainty in data, where principled methods and algorithms for

* Published by Springer: Lecture Notes in Computer Science Volume 9422, 2015, pp 36-46. The final publication is available at link.springer.com. DOI: 10.1007/978-3-319-24598-0_4

dealing with uncertainty in massive data applications are required. Probabilistic graphical models (PGMs) provide a well-founded and principled approach for performing inference and belief updating in complex domains endowed with uncertainty.

In this paper, we are interested in a particular type of PGMs, the so-called Bayesian networks [13], and more precisely, *hybrid* Bayesian networks, where discrete and continuous variables coexist. Our goal is to analyse the performance of probabilistic inference in hybrid Bayesian networks in scenarios where data come in streams at high speed, and therefore a quick response is required. Because of that, we will focus our analysis on conditional linear Gaussian (CLG) models [10, 11], instead of more expressive alternatives such as mixtures of exponentials [12], mixtures of polynomials [18] and mixtures of truncated basis functions in general [9], as inference in the latter models is in general more time consuming [15].

The remainder of the paper is organised as follows. Section 2 establishes the necessary background and contains the problem formulation. Section 3 describes the algorithms we consider in this paper. The core of the contributions is in Section 3.2, where we develop a parallel algorithm based on importance sampling for CLG networks. Its performance is tested in Section 4 and the paper ends with the conclusion in Section 5.

2 Preliminaries

Bayesian networks (BNs) [3, 8, 13] are a particular type of PGM that has enjoyed widespread attention in the last two decades. Attached to each node, there is a conditional probability distribution given its parents in the network, so that in general, for a BN with N variables $\mathbf{X} = \{X_1, \ldots, X_N\}$, the joint distribution factorizes as $p(\mathbf{X}) = \prod_{i=1}^{N} p_i(X_i|Pa(X_i))$, where $Pa(X_i)$ denotes the set of parents of X_i in the network. A BN is called *hybrid* if some of its variables are discrete while some others are continuous.

We will use lowercase letters to refer to values or configurations of values, so that x denotes a value of X and boldface $\mathbf{x}$ is a configuration of the variables in $\mathbf{X}$. Given a set of observed variables $\mathbf{X}_E \subset \mathbf{X}$ and a set of variables of interest $\mathbf{X}_I \subset \mathbf{X} \setminus \mathbf{X}_E$, *probabilistic inference* consists of computing the posterior distribution $p(x_i|\mathbf{x}_E)$ for each $i \in I$, where X_i can be either discrete or continuous. If we denote by $\mathbf{X}_C$ and $\mathbf{X}_D$ the set of continuous and discrete variables not in $\{\mathbf{X}_i\} \cup \mathbf{X}_E$, and by $\mathbf{X}_{C_i}$ and $\mathbf{X}_{D_i}$ the set of continuous and discrete variables not in $\mathbf{X}_E$, the goal of inference can be formulated as computing

$$p(x_i|\mathbf{x}_E) = \frac{p(x_i, \mathbf{x}_E)}{p(\mathbf{x}_E)} = \frac{\displaystyle\sum_{\mathbf{x}_D \in \Omega_{\mathbf{X}_D}} \int_{\mathbf{x}_C \in \Omega_{\mathbf{X}_C}} p(\mathbf{x}; \mathbf{x}_E) \mathrm{d}\mathbf{x}_C}{\displaystyle\sum_{\mathbf{x}_{D_i} \in \Omega_{\mathbf{X}_{D_i}}} \int_{\mathbf{x}_{C_i} \in \Omega_{\mathbf{X}_{C_i}}} p(\mathbf{x}; \mathbf{x}_E) \mathrm{d}\mathbf{x}_{C_i}}, \tag{1}$$

where $\Omega_{\mathbf{X}}$ is the set of possible values of a set of variables $\mathbf{X}$ and $p(\mathbf{x};\mathbf{x}_E)$ is the joint distribution in the BN instantiated according to the observed values $\mathbf{x}_E$.

Often, one is not interested in the full posterior distribution of X_i, but rather in the probability of the variable taking values on a given interval (a,b), which amounts to computing

$$p(a < X_i < b|\mathbf{x}_E) = \frac{\displaystyle\int_a^b \left(\sum_{\mathbf{x}_D \in \Omega_{\mathbf{X}_D}} \int_{\mathbf{x}_C \in \Omega_{\mathbf{X}_C}} p(\mathbf{x};\mathbf{x}_E)\mathrm{d}\mathbf{x}_C \right) \mathrm{d}x_i}{\displaystyle\sum_{\mathbf{x}_{D_i} \in \Omega_{\mathbf{X}_{D_i}}} \int_{\mathbf{x}_{C_i} \in \Omega_{\mathbf{X}_{C_i}}} p(\mathbf{x};\mathbf{x}_E)\mathrm{d}\mathbf{x}_{C_i}}, \tag{2}$$

if X_i is continuous. If it is discrete, instead of the variable taking values on an interval, we are interested in one of its possible values, i.e.

$$p(X_i = x_i|\mathbf{x}_E) = \frac{\displaystyle\sum_{\mathbf{x}_D \in \Omega_{\mathbf{X}_D}} \int_{\mathbf{x}_C \in \Omega_{\mathbf{X}_C}} p^{R(X_i=x_i)}(\mathbf{x};\mathbf{x}_E)\mathrm{d}\mathbf{x}_C}{\displaystyle\sum_{\mathbf{x}_{D_i} \in \Omega_{\mathbf{X}_{D_i}}} \int_{\mathbf{x}_{C_i} \in \Omega_{\mathbf{X}_{C_i}}} p(\mathbf{x};\mathbf{x}_E)\mathrm{d}\mathbf{x}_{C_i}}, \tag{3}$$

where $p^{R(X_i=x_i)}(\mathbf{x};\mathbf{x}_E)$ denotes the restriction of function $p(\mathbf{x};\mathbf{x}_E)$ to the value x_i of variable X_i, if X_i is discrete. We call the probabilistic inference tasks described in Eqs. (2) and (3) a *query*.

2.1 Conditional Linear Gaussian Networks

A *Conditional Linear Gaussian Network* is a hybrid Bayesian network where the joint distribution is a conditional linear Gaussian (CLG) [11]. In the CLG model, the conditional distribution of each discrete variable $X_D \in \mathbf{X}$ given its parents is a multinomial, whilst the conditional distribution of each continuous variable $Z \in \mathbf{X}$ with discrete parents $\mathbf{X}_D \subseteq \mathbf{X}$ and continuous parents $\mathbf{X}_C \subseteq \mathbf{X}$, is given as a normal density by

$$p(z|\mathbf{X}_D = \mathbf{x}_D, \mathbf{X}_C = \mathbf{x}_C) = \mathcal{N}(z; \alpha_{\mathbf{x}_D} + \boldsymbol{\beta}_{\mathbf{x}_D}^{\mathsf{T}} \mathbf{x}_C, \sigma_{\mathbf{x}_D}), \tag{4}$$

for all $\mathbf{x}_D \in \Omega_{\mathbf{X}_D}$ and $\mathbf{x}_C \in \Omega_{\mathbf{X}_C}$, where α and $\boldsymbol{\beta}$ are the coefficients of a linear regression model of Z given its continuous parents; this model can differ for each configuration of the discrete variables $\mathbf{X}_D$. Therefore, the conditional mean of Z is a linear model on its continuous parents, while its standard deviation, σ_D, only depends on the discrete ones.

After fixing any configuration of the discrete variables, the joint distribution of any subset $\mathbf{X}_C \subseteq \mathbf{X}$ of continuous variables is a multivariate Gaussian whose parameters can be obtained from the ones in the CLG representation. For a set of M continuous variables $Z_1, \ldots, Z_M$ with a conditionally specified joint density

$p(z_1,\ldots,z_M) = \prod_{k=1}^{M} p(z_k|z_{k+1},\ldots,z_M)$, where the k-th factor, $1 \leq k \leq M$, is such that

$$p(z_k|z_{k+1},\ldots,z_M) = \mathcal{N}(z_k; \mu_{z_k|z_{k+1},\ldots,z_M}, \sigma_{z_k}),$$

it holds that the joint is $p(z_1,\ldots,z_M) = \mathcal{N}(z_1,\ldots,z_M; \boldsymbol{\mu}, \boldsymbol{\Sigma})$, where $\boldsymbol{\mu}$ is the n-dimensional vector of means and $\boldsymbol{\Sigma}$ is the covariance matrix of the multivariate distribution over random variables $Z_1,\ldots,Z_M$ and both $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are derived from the parameters in Eq. (4) [17].

3 Approximate inference in CLG networks

Exact inference in CLG networks is a computationally expensive task that requires the construction of a *strong* junction tree in order to guarantee that the continuous variables are marginalised out first [10]. Hence, in scenarios as stream processing, where quick responses are required, the use of approximate algorithms becomes necessary. In this section we analyse two approaches to approximate inference in CLG networks. Both are based on general techniques for probabilistic inference able to provide quick answers to queries, namely *variational inference* [1] and *importance sampling* [6].

3.1 Variational inference

Variational inference is a deterministic approximate inference technique, where we seek to iteratively optimise a variational approximation to the posterior distribution of interest [1]. Let $\mathcal{Q}$ be the set of possible approximations; then the variational approximation to a posterior distribution $p(\mathbf{x}_I|\mathbf{X}_E = \mathbf{x}_E)$ is defined as

$$q^*_{\mathbf{x}_E}(\mathbf{x}_I) = \arg\min_{q\in\mathcal{Q}} D(q(\mathbf{x}_I)||p(\mathbf{x}_I|\mathbf{X}_E = \mathbf{x}_E)),$$

where $D(q||p)$ is the KL divergence between q and p.

A common approach is to employ a *variational mean-field* approximation of the posterior distribution, so that the approximation factorises over the individual variables involved, i.e.,

$$q^*_{\mathbf{x}_E}(\mathbf{x}_I) = \prod_{i\in I} q^*_{\mathbf{x}_E}(x_i). \tag{5}$$

During the optimisation of the variational mean-field one performs a coordinate ascent, where we iteratively update the individual variational distributions while holding the others fixed [7]. Updating a variational distribution essentially involves calculating the variational expectation of the logarithm of the original conditional distributions of the model. This can be done efficiently and in closed form when the distributions involved are conjugate-exponential [2]. A general architecture for supporting *variational message passing* (VMP) in graphical models is presented in [20], highlighting how distributions that are conjugate-exponential families can be utilised to efficiently represent the messages by the

expected natural statistics. In this paper, we consider the application of VMP to CLG networks, and therefore the posterior distribution of the variables in the network will be the factors in Eq. (5), represented as normal densities for continuous variables and as multinomials for the discrete ones.

3.2 Importance sampling

Importance sampling [6] is a versatile simulation technique that in the case of inference in BNs amounts to transforming the numerator in Eq. (2) by multiplying and dividing by a distribution p^* that, unlike $p(\mathbf{x}, \mathbf{x}_E)$, is easy to handle and, more precisely, from which samples can easily be drawn.

Let θ denote the numerator of Eq. (2), i.e. $\theta = \int_a^b h(x_i)\mathrm{d}x_i$ with

$$h(x_i) = \sum_{\mathbf{x}_D \in \Omega_{\mathbf{X}_D}} \int_{\mathbf{x}_C \in \Omega_{\mathbf{X}_C}} p(\mathbf{x}; \mathbf{x}_E)\mathrm{d}\mathbf{x}_C.$$

Then, we can write θ as

$$\theta = \int_a^b h(x_i)\mathrm{d}x_i = \int_a^b \frac{h(x_i)}{p^*(x_i)} p^*(x_i)\mathrm{d}x_i = E_{p^*}\left[\frac{h(X_i^*)}{p^*(X_i^*)}\right], \tag{6}$$

where p^* is a probability density function on (a, b) called the *sampling distribution*, and X_i^* is a random variable with density p^*. Let $X_i^{*(1)}, \ldots, X_i^{*(m)}$ be a sample drawn from p^*. Then it is easy to prove that

$$\hat{\theta}_1 = \frac{1}{m} \sum_{j=1}^{m} \frac{h(X_i^{*(j)})}{p^*(X_i^{*(j)})} \tag{7}$$

is an unbiased estimator of θ.

As $\hat{\theta}_1$ is unbiased, the error of the estimation is determined by its variance, which is

$$\begin{aligned} \mathrm{Var}(\hat{\theta}_1) &= \mathrm{Var}\left(\frac{1}{m} \sum_{j=1}^{m} \frac{h(X_i^{*(j)})}{p^*(X_i^{*(j)})}\right) = \frac{1}{m^2} \sum_{j=1}^{m} \mathrm{Var}\left(\frac{h(X_i^{*(j)})}{p^*(X_i^{*(j)})}\right) \\ &= \frac{1}{m^2} m \mathrm{Var}\left(\frac{h(X_i^*)}{p^*(X_i^*)}\right) = \frac{1}{m} \mathrm{Var}\left(\frac{h(X_i^*)}{p^*(X_i^*)}\right). \end{aligned} \tag{8}$$

The key point in importance sampling is the selection of the sampling distribution since, according to Eq. (8), it determines the accuracy of the estimation. The rule is that the closer p^* is to h, the lower the variance is [4].

A simple procedure for selecting the sampling distribution is the so-called *evidence weighting* (EW) [5]. In EW, each variable is sampled from a conditional density given its parents in the network. The sampling order is therefore from parents to children. The observed variables are not sampled, but instead they are instantiated to the observed value. A version of this algorithm in which the conditional densities are dynamically updated during the simulation procedure was

introduced in [19]. In this paper we will only use static sampling distributions, as that is the fastest alternative.

Hence, adopting EW means that h involves the product of all the conditional distributions in the Bayesian network, while p^* involves the same conditional distributions except those ones corresponding to observed variables.

Note that the denominator in Eq. (2) is just the probability of evidence, which has to be estimated as well in order to have an answer to a query (recall that $\hat{\theta}_1$ is just an estimator of the numerator). It was shown in [4] that numerator and denominator can be estimated using the same sample. To achieve this, instead of taking a sampling distribution defined on (a, b) it must be defined on the entire range of X_i. In such case, the estimator in Eq. (7) becomes an estimator of the denominator (probability of evidence) and the same estimator, evaluated only in the points in the sample that fall inside (a, b), is an estimator of θ.

Function EW($\mathbf{X}$,P,$\mathbf{x}_E$,X,a,b,M)
Input: The set of variables in the network, $\mathbf{X} = \{X_1, \ldots, X_N\}$ in topological order. The distributions in the network $P = \{p_1, \ldots, p_N\}$. Evidence $\mathbf{X}_E = \mathbf{x}_E$. The target variable X. Sample size M.
Output: An estimation of $P(a < X < b|\mathbf{X}_E = \mathbf{x}_E)$
begin
  Initialization:
  $s_1 \leftarrow 0$; $s_2 \leftarrow 0$.
  for $j \leftarrow 1$ **to** M **do**
    Sample generation:
    $w_1 \leftarrow 1$; $w_2 \leftarrow 1$.
    for $i \leftarrow 1$ **to** N **do**
      if $X_i \notin \mathbf{X}_E$ **then**
        Simulate a value $x_i^{(j)}$ for X_i using $p_i(x_i|Pa(x_i))$.
        $w_2 \leftarrow w_2 * p_i(x_i^{(j)}|Pa(x_i))$.
      end
      else
        Let $x_i^{(j)}$ be the value of X_i in $\mathbf{X}_E$.
      end
      $w_1 \leftarrow w_1 * p_i(x_i^{(j)}|Pa(x_i))$.
    end
    if $w_1 \neq 0$ **then**
      Let $x^{(j)}$ be the value of X in the simulated configuration $x_1^{(j)}, \ldots, x_N^{(j)}$.
      if $x^{(j)} \in (a, b)$ **then**
        $s_1 \leftarrow s_1 + w_1/w_2$
      end
      $s_2 \leftarrow s_2 + w_1/w_2$
    end
  end
  return s_1/s_2 .
end

Algorithm 1: The EW algorithm for answering a probabilistic query.

The details of the inference procedure are given in Alg. 1. In the For loop devoted to sample generation, w_1 and w_2 represent, respectively, the values of h and p^* for the simulated configurations. Each variable is simulated using its con-

ditional distribution. In fact, we can only simulate from marginals rather than conditional distributions. That's why EW starts simulating from root nodes, where marginal distributions are attached to them. Once root nodes are simulated (i.e. we have a value for them), their children are simulated by first instantiating their conditional distributions to the simulated values for the roots, obtaining, therefore marginal densities. As we are operating with CLG networks, the marginal distribution for each discrete variable is a multinomial while it is a normal for each continuous variable. In both cases, simulating values from them is straightforward [14].

Some configurations can be useless for the estimation procedure. That is the case in which w_1 becomes zero which happens when a simulated configuration is incompatible with the observations. As an example, consider a BN with two binary variables X and Y, where $P(X = 0) = 0.9$, $P(Y = 0|X = 0) = 0$ and $P(Y = 0|X = 1) = 0.5$. It means that, approximately, 90% of the times the value simulated for X will be 0. Assume that we have observed $Y = 0$. As $P(Y = 0|X = 0) = 0$, 90% of the times the simulated configuration will be discarded. This problem only arises when simulating discrete variables, as the normal density for a continuous variable is never equal to 0.

In regards of scalability, it is worth pointing out that the iterations in the For loop for sample generation can be executed in parallel. This is due to the fact that the items in the sample are independent of each other. As that loop constitutes the fundamental workload of the algorithm, the scalability is potentially high, using, for instance, a multi-threaded implementation. Our proposal for scaling up the algorithm consists of using parallelisation in the above mentioned For loop. We have used Java 8 streams in our implementation.

Even though all the discussions above were focused on queries involving a continuous variable, similar arguments can be developed for discrete queries, where instead of an interval, we seek the probability of a variable taking on a fixed value. For the sake of simplicity, we omit here the details for the discrete case.

4 Experimental evaluation

In order to test the accuracy and scalability of EW with respect to available computing resources, we conducted an experiment over two randomly generated CLG networks with 10 and 500 variables respectively, half of them continuous and the rest binary discrete variables. The aim of this choice is to test the behaviour of the algorithm when dealing with small as well as with large models. The number of links was set to double the number of variables. We have not considered any parallelisation issue for VMP, but we have included it in the experimental analysis as a bench mark.

For each network, we randomly generated a set of observations for 5% of the variables. Queries were also selected at random, by choosing a variable and generating a number α from a standard normal distribution and taking the interval (a, b) with $a = \alpha - 0.5$ and $b = \alpha + 0.5$. Each query was answered

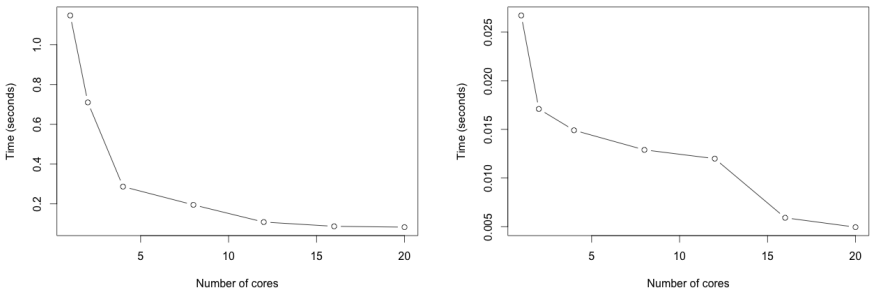

Fig. 1. Run time as a function of the number of cores for EW over a randomly generated CLG network with 500 variables (left) and 10 variables (right).

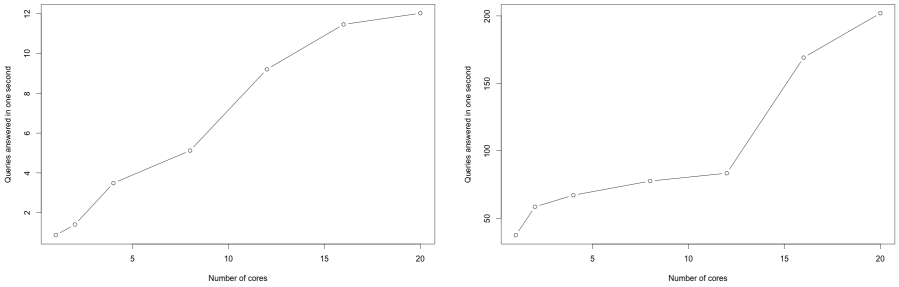

Fig. 2. Number of queries answered per second as a function of the number of cores for EW over a randomly generated CLG network with 500 variables (left) and 10 variables (right).

using VMP and EW, the latter with samples of size 1000, 5000 and 10000. Each experiment was replicated for an increasing number of cores ranging from 1 to 20. The experiments were run on a dual-processor AMD Opteron 2.8GHz server with 32 cores and 64GB of RAM, running Linux Ubuntu 14.04.1 LTS.

Each run was repeated 10 times and the run time and error of the estimations were averaged over the 10 runs. The error of the estimations was computed using the χ^2 divergence. Let p_i, $i = 1, \ldots, 10$ be the exact probability corresponding to the query in run i, and let q_i be the estimated value. The χ^2 divergence is computed as

$$\chi^2 = \frac{1}{10} \sum_{i=1}^{10} \frac{(q_i - p_i)^2}{p_i}.$$

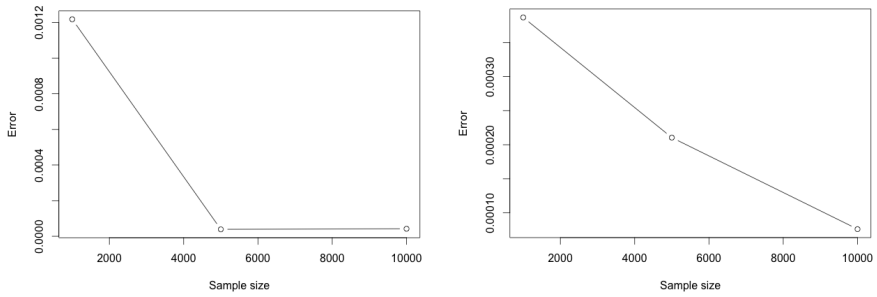

Fig. 3. Error attained by EW as a function of the sample size for a network with 10 variables (left) and with 500 variables (right).

	10 vars.	500 vars.
Run time (seconds)	0.0739	9.6917
Error	0.4657	2.2759

Table 1. Error and run times for VMP.

The χ^2 divergence is specially appropriate for measuring errors in probability estimations, as it is measured taking into account the magnitude of the value to estimate, and not simply the absolute or square deviation.

The results of the experiments in terms of run time for EW are shown in Fig. 1. The results for VMP are given in Tab. 1. The plots correspond to a sample of size 1000 for EW, and show the evolution of the run time as a function of the number of cores used during the computation. It can be seen how in both networks EW scales up with respect to the number of cores. we conjecture that the jump in the curve at 12 cores is probably due to the small magnitude of the run time (of the order of miliseconds) and hence, any small variation due to any issue external to the algorithm can cause it, specially taking into account that the server where the experiments were run was shared with other users.

The ability of the algorithm for processing streams is illustrated in Fig. 2, where the number of queries answered per second is given as a function of the number of cores. It can be seen that the algorithm is able to process up to 12 queries in a second for the 500 variable network, and over 200 per second for the 10 variable network, when using 20 cores.

There is a big difference in favour of EW with respect to VMP in what concerns computing time, according to Tab. 1. For instance, when using 20 cores, EW gives an answer in less than 0.1 seconds, while VMP takes around 9, in the large network. In the small network, with only 10 variables, the results are similar, resulting EW as a much faster procedure, reaching response times for 20 cores below 0.005 seconds. It means that the method is able to answer

around 200 queries in 1 second, which is of special interest when processing queries coming in streams.

The behaviour of the EW algorithm in terms of error is summarised in the plots in Fig. 3, where the χ^2 divergence is represented versus the sample size. As expected, the error goes down as the sample size increases. Even with the lowest sample size considered (1000), the errors are fairly low for the large and small networks. The errors reported by VMP are considerably higher, as reported in Tab. 1

5 Conclusion

In this paper we have analysed the problem of approximate inference in CLG networks with special interest in parallelisation issues. We have tested the behaviour of two general approaches to probabilistic inference when applied to CLG networks. Importance sampling, and more precisely EW, has shown to be preferable to VMP both in terms of speed and accuracy. The quick responses provided by EW suggest that it is potentially an appropriate inference method for answering queries when evidence comes in form of a stream.

Though the experimental results are promising, they are still limited. We intend to study the inference problems in more complex settings, involving networks with more variables and more links. Also, the randomly generated networks did not include a high concentration of extreme probabilities for the discrete variables, i.e. zeros in the probability tables. In scenarios of extreme probabilities, EW is known to be not so accurate [16], and therefore more sophisticated methods as the ones proposed in [4] for mixtures of truncated exponentials, are to be developed.

Acknowledgments. This work was performed as part of the AMIDST project. AMIDST has received funding from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no 619209.

References

1. H. Attias. A variational Bayesian framework for graphical models. *Advances in neural information processing systems*, pages 209—215, 2000.
2. M.J. Beal. *Variational algorithms for approximate Bayesian inference*. PhD thesis, Gatsby Computational Neuroscience Unit, University College London, 2003.
3. R.G. Cowell, A.P. Dawid, S.L. Lauritzen, and D.J. Spiegelhalter. *Probabilistic Networks and Expert Systems*. Statistics for engineering and information science. Springer, 1999. ISBN 0-387-98767-3.
4. A. Fernández, R. Rumí, and A. Salmerón. Answering queries in hybrid Bayesian networks using importance sampling. *Decision Support Systems*, 53:580–590, 2012.

5. R. Fung and K. C. Chang. Weighting and integrating evidence for stochastic simulation in Bayesian networks. In M. Henrion, R.D. Shachter, L.N. Kanal, and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence*, volume 5, pages 209–220. North-Holland (Amsterdam), 1990.
6. J.M. Hammersley and D.C. Handscomb. *Monte Carlo Methods*. Chapman & Hall, 1964.
7. T.S. Jaakkola and Y. Qi. Parameter expanded variational Bayesian methods. In *Advances in Neural Information Processing Systems*, pages 1097–1104, 2006.
8. D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
9. H. Langseth, T.D. Nielsen, R. Rumí, and A. Salmerón. Mixtures of truncated basis functions. *International Journal of Approximate Reasoning*, 53(2):212–227, 2012.
10. S. L. Lauritzen and F. Jensen. Stable local computation with conditional Gaussian distributions. *Statistics and Computing*, 11(2):191–203, 2001.
11. S.L. Lauritzen and N. Wermuth. Graphical models for associations between variables, some of which are qualitative and some quantitative. *The Annals of Statistics*, 17:31–57, 1989.
12. S. Moral, R. Rumí, and A. Salmerón. Mixtures of truncated exponentials in hybrid Bayesian networks. In *EQSCARU'2001*, volume 2143 of *Lecture Notes in Artificial Intelligence*, pages 145–167. Springer, Berlin, Germany, 2001.
13. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., San Mateo, CA., 1988.
14. R. Y. Rubinstein. *Simulation and the Monte Carlo Method*. Wiley (New York), 1981.
15. R. Rumí and A. Salmerón. Approximate probability propagation with mixtures of truncated exponentials. *International Journal of Approximate Reasoning*, 45:191–210, 2007.
16. A. Salmerón, A. Cano, and S. Moral. Importance sampling in Bayesian networks using probability trees. *Computational Statistics and Data Analysis*, 34:387–413, 2000.
17. R.D. Shachter and C. Kenley. Gaussian ináuence diagrams. *Management Science*, 35:527–550, 1989.
18. P.P. Shenoy and J.C. West. Inference in hybrid Bayesian networks using mixtures of polynomials. *International Journal of Approximate Reasoning*, 52:641–657, 2011.
19. W. Sun and K.C. Chang. Probabilistic inference using linear Gaussian importance sampling for hybrid Bayesian networks. In *Signal Processing, Sensor Fusion, and Target Recognition XIV. Proc. of SPIE*, volume 5809, pages 322–329, 2005.
20. J.M. Winn and C.M. Bishop. Variational message passing. *Journal of Machine Learning Research*, 6:661–694, 2005.