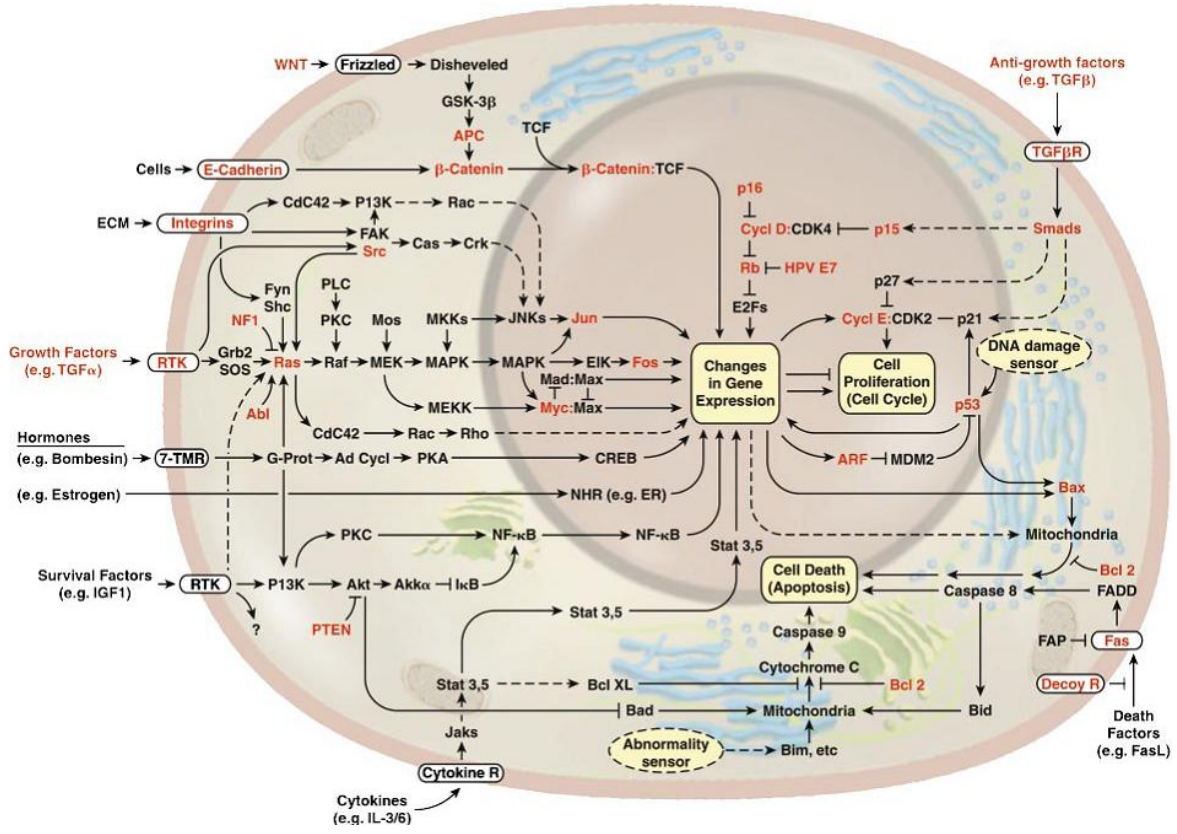
Approximate methods for DBN models of Bio-pathway systems

S Akshay

joint work with
Blaise Genest, Sucheendra K Palaniappan and P S Thiagarajan,
IPAL-CNRS and National University of Singapore

ACTS workshop,
Chennai Mathematical Institute, 27-29 Jan 2011

Overview: Biological pathways in a cell



- To model their dynamics as a Dynamic Bayesian Network.
- To perform computations (efficiently) using this model.

Bio chemical equations and ODEs

Bio chemical equations: $S + E \underset{k_2}{\overset{k_1}{\rightleftharpoons}} ES \xrightarrow{k_3} E + P$

Modeled as ODEs:

$$\frac{d[S]}{dt} = k_2[ES] - k_1[S][E] \qquad \frac{d[E]}{dt} = (k_2 + k_3)[ES] - k_1[S][E]$$

$$\frac{d[P]}{dt} = k_3[ES] \qquad \frac{d[ES]}{dt} = k_1[E][S] - (k_2 + k_3)[ES]$$

Bio chemical equations and ODEs

Bio chemical equations: $S + E \underset{k_2}{\overset{k_1}{\rightleftharpoons}} ES \xrightarrow{k_3} E + P$

Modeled as ODEs:

$$\frac{d[S]}{dt} = k_2[ES] - k_1[S][E] \qquad \frac{d[E]}{dt} = (k_2 + k_3)[ES] - k_1[S][E]$$

$$\frac{d[P]}{dt} = k_3[ES] \qquad \frac{d[ES]}{dt} = k_1[E][S] - (k_2 + k_3)[ES]$$

But, ODEs are generally too big to admit closed form solutions

- Resort to numerical simulations giving rise to trajectories.
- Parameters are imprecise so need LOTS of trajectories which are recomputed often.

Introducing probabilities

Markov chain

- States: Concentration of each species in [0,1] or [1,2] or [2,3] or [3,4] or [4,5] (discretization).
- Transitions: If Concentrations are in state s_1
$$E \in [0,1],\ S \in [3,4],\ ES \in [2,3],\ P \in [2,3]$$
at time t, then probability to be in state s_2
$$E \in [1,2], S \in [3,4], ES \in [2,3], P \in [2,3]$$
at time $t+1$ is 0.3...

That is,

the fraction of trajectories that start from state s_1 at time t and reach s_2 in one time unit is 0.3.

Quantitative Model Checking

- We are interested in simple questions. For instance,
  - What is the concentration of protein species X_i at time point t? That is, $P(X_i^t = v) = ?$
- We can phrase it as a Probabilistic CTL model checking question over the Markov chain.
- Use a probabilistic model checker (say, PRISM) to solve it.

Quantitative Model Checking

- We are interested in simple questions. For instance,
  - What is the concentration of protein species X_i at time point t? That is, $P(X_i^t = v) = ?$
- We can phrase it as a Probabilistic CTL model checking question over the Markov chain.
- Use a probabilistic model checker (say, PRISM) to solve it.

However, there are two problems:

1. Large size of the Markov chain model
2. Large number of computations

Compact Representation of the Markov chain

Problem 1: Size of Markov chain is huge

- matrix of $5^{\text{no. of species}}$ rows and columns!

- PRISM considers representation as product of Markov chains.
- But, we do not know how to decompose our large Markov chain into a product of Markov chains.

Compact Representation of the Markov chain

Problem 1: Size of Markov chain is huge

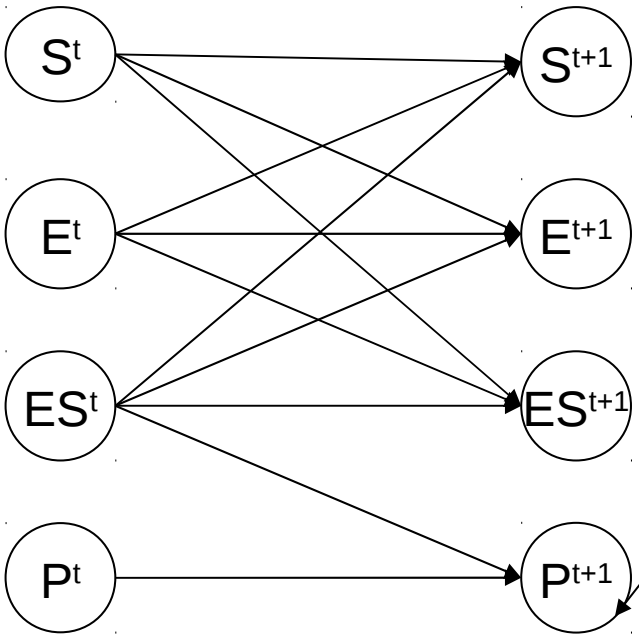
- matrix of $5^{\text{no. of species}}$ rows and columns!

- PRISM considers representation as product of Markov chains.
- But, we do not know how to decompose our large Markov chain into a product of Markov chains.

Solution: a "probabilistic graphical model"

- an underlying graph to describe relation between variables over a time step
- Markovian properties/assumptions of the biological system.

The Dynamic Bayesian Network



a Random Variable per species per time point, with its value being the concentration (interval)

Conditional Probability table :

eg., Pr ($P^{t+1} = I \mid P^{t} = I', ES^{t} = I'') = 0.7$

The graph structure is invariant over time

Semantics of a DBN

Joint probability in a DBN is defined recursively as:

$$P(X^t = \vec{v}) = \sum_{\vec{u}} P(X^{t-1} = \vec{u}) \prod_i P(X_i^t = \vec{v}_i \mid X^{t-1}_{parents\{i\}} = \vec{u}_{parents\{i\}})$$

Marginal probability is defined by "summing out":

$$P(X_i^t = \vec{v}_i) = \sum_{\vec{w} \mid \vec{w}_i = \vec{v}_i} P(X^t = \vec{w})$$

Semantics of a DBN

Joint probability in a DBN is defined recursively as:

$$P(X^t = \vec{v}) = \sum_{\vec{u}} P(X^{t-1} = \vec{u}) \prod_i P(X_i^t = \vec{v}_i \mid X^{t-1}_{parents\{i\}} = \vec{u}_{parents\{i\}})$$

Marginal probability is defined by "summing out":

$$P(X_i^t = \vec{v}_i) = \sum_{\vec{w} \mid \vec{w}_i = \vec{v}_i} P(X^t = \vec{w})$$

Problem 2: too many computations

- summation over all u means 5^{n-1} computations

Large number of computations

Problem 2: too many computations

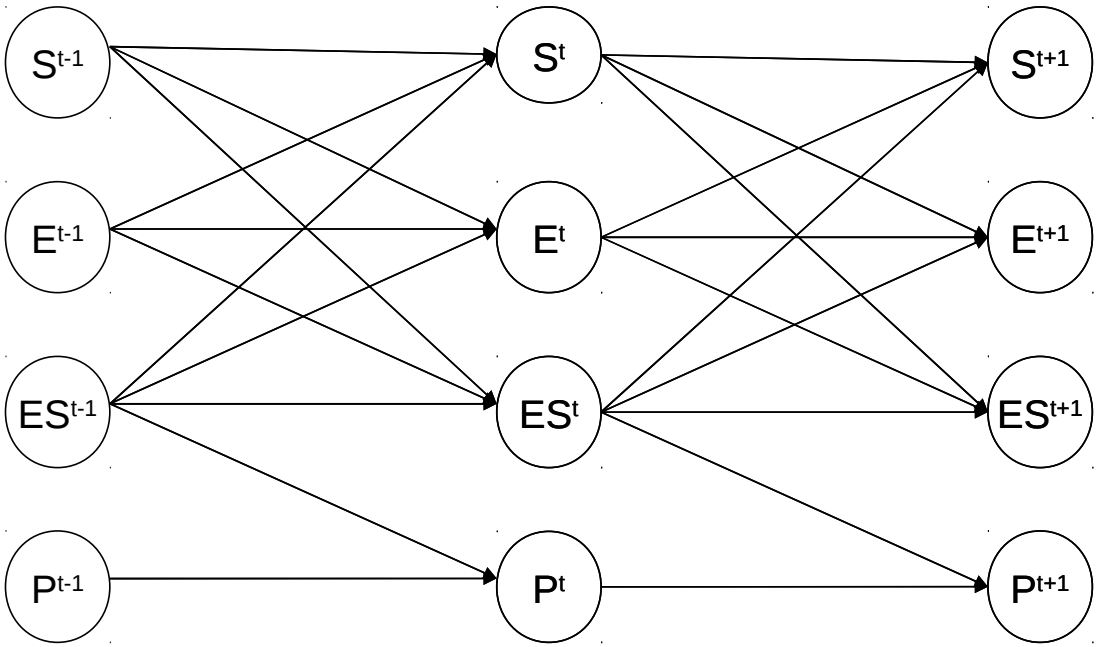
- summation over all u means 5^{n-1} computations

- Existing model checkers cannot handle this.
- For eg., PRISM can handle $\sim 10^7/10^8$ states.
[www.prismmodelchecker.org/manual/FrequentlyAskedQuestions]
- Whereas, our model has $5^{32} \sim 10^{22}$ states.
- Indeed this is routine in Biological systems...

Inferencing: Dependence over time

Can we use the structure of the DBN?

- the conditional dependencies (of the DBN) do not help, since *in a few time steps*, all variables typically become correlated.



Approximate Inferencing

- Idea: Approximation...
- Assume that the random variables at the previous step are independent, i.e., Replace the joint distributions:

$$P(X_1^{t-1} = u_1, X_2^{t-1} = u_2, \ldots, X_n^{t-1} = u_n)$$

by the product of the marginals:

$$\prod_{j=1}^{n} P(X_j^{t-1} = \vec{u}_j)$$

Approximate Inferencing

- Replace the joint distributions:

$$P(X_1^{t-1} = u_1, X_2^{t-1} = u_2, \ldots, X_n^{t-1} = u_n)$$

by the product of the marginals:

$$\prod_{j=1}^{n} P(X_j^{t-1} = \vec{u}_j)$$

- Then we can show that $P(X_i^t = v) =$

$$\sum_{\vec{u}_{parents\{i\}}} \left(\prod_{j \in parents\{i\}} P(X_j^{t-1} = u_j) \right) P(X_i^t = v \mid X_{parents\{i\}}^{t-1} = \vec{u}_{parents\{i\}})$$

Approximate Inferencing

- Replace the joint distributions:

$$P(X_1^{t-1} = u_1, X_2^{t-1} = u_2, \ldots, X_n^{t-1} = u_n)$$

by the product of the marginals:

$$\prod_{j=1}^{n} P(X_j^{t-1} = \vec{u}_j)$$

- Then we can show that $P(X_i^t = v) =$

$$\sum_{\vec{u}_{parents\{i\}}} \Big(\prod_{j \in parents\{i\}} P(X_j^{t-1} = u_j) \Big) P(X_i^t = v \mid X_{parents\{i\}}^{t-1} = \vec{u}_{parents\{i\}})$$

- The Factored Frontier (FF) algorithm [Murphy & Weiss'01] computes this, denoted $P^{FF}(X_i^t = v)$.

The computations and the questions

- Typical question: What is the concentration of protein species X_i at time point t? That is, $P(X_i^t = v) = ?$

The computations and the questions

- Typical question: What is the concentration of protein species X_i at time point t? That is, $P(X_i^t = v) = ?$
- From previous slide:
  If $P^{FF}(X_i^t = v) = k$ and the error due to FF approximation is bounded by ϵ, then

$$k - \epsilon \leq P(X_i^t = v) \leq k + \epsilon$$

- For example: Suppose we want $P(X_i^{100} = v) < 0.5$ and we obtain $P^{FF}(X_i^{100} = v) = 0.1$ and $\epsilon = 0.2$, then :-)

The computations and the questions

- Typical question: What is the concentration of protein species X_i at time point t? That is, $P(X_i^t = v) = ?$
- From previous slide:
  If $P^{FF}(X_i^t = v) = k$ and the error due to FF approximation is bounded by ϵ, then

$$k - \epsilon \leq P(X_i^t = v) \leq k + \epsilon$$

- For example: Suppose we want $P(X_i^{100} = v) < 0.5$ and we obtain $P^{FF}(X_i^{100} = v) = 0.1$ and $\epsilon = 0.2$, then :-)

Goal:

To give a theoretical bound for ϵ.

The computations and the questions

- Typical question: What is the concentration of protein species X_i at time point t? That is, $P(X_i^t = v) = ?$
- From previous slide:
  If $P^{FF}(X_i^t = v) = k$ and the error due to FF approximation is bounded by ϵ, then

$$k - \epsilon \leq P(X_i^t = v) \leq k + \epsilon$$

- For example: Suppose we want $P(X_i^{100} = v) < 0.5$ and we obtain $P^{FF}(X_i^{100} = v) = 0.1$ and $\epsilon = 0.2$, then :-)

Goal:

To give a theoretical bound for ϵ.

- But, how bad is the error in practice?

The computations and the questions

- In practice, FF performs surprisingly well on most species in many models. But in some cases, ϵ can be as big as 0.4.

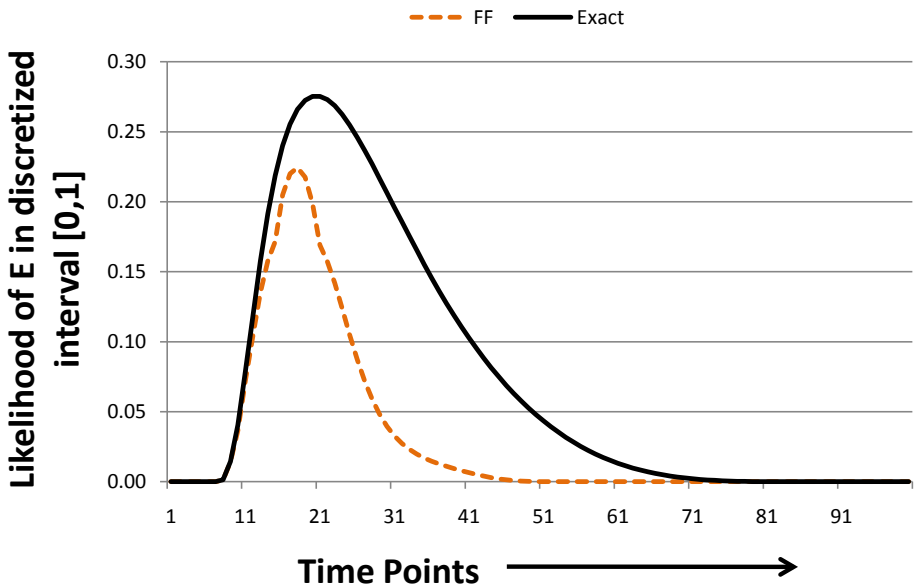


Figure: (marginal) prob of conc of E being in $[0, 1]$ over time

- Can we do better (perhaps, at the cost of spending more time doing computations)?
- Rest of this talk: Partial answer to the above questions...

Rest of the talk

- A quick error analysis for the FF approximation scheme.
- Introducing a parametrized version of the FF algorithm called Hybrid FF.
- An intuitive example.
- Experimental results.
- Conclusion.

Quick error analysis

- At each step of FF, the joint distribution is approximated by the product of the marginals.
- Let P^t denote the exact joint distribution at time t. Observe that, $P^t = T(P^{t-1})$ where T is the transition matrix of the underlying markov chain.
- Let B^t denote the product of marginals given at time t by FF.

Overall error at time t:

$$\Delta_t = |P^t - B^t| = |T(P^{t-1}) - B^t|$$

Quick error analysis

- At each step of FF, the joint distribution is approximated by the product of the marginals.
- Let P^t denote the exact joint distribution at time t. Observe that, $P^t = T(P^{t-1})$ where T is the transition matrix of the underlying markov chain.
- Let B^t denote the product of marginals given at time t by FF.

Overall error at time t:

$$\Delta_t = |P^t - B^t| = |T(P^{t-1}) - B^t|$$

- Then,

$$\Delta_t = |T(P^{t-1}) - B^t| \leq |T(P^{t-1}) - T(B^{t-1})| + \underbrace{|T(B^{t-1}) - B^t|}_{\text{single step error } \delta_t}$$

Quick error analysis

Thus, overall error at time t is bounded by

1. $\Delta_t \leq |T(P^{t-1}) - T(B^{t-1})| + \delta_t$
2. From [Boyen & Koller'98] we obtain $|T(P^{t-1}) - T(B^{t-1})| \leq \lambda |P^{t-1} - B^{t-1}|$ where $0 \leq \lambda \leq 1$ is a contraction factor depending on T.
3. Thus,

$$\Delta_t \leq \lambda \Delta_{t-1} + \delta_t$$

Lemma

Let us denote the max single step error as δ. Then,

$$\epsilon = \max_t \Delta_t \leq \frac{\delta}{1-\lambda}, \text{ if } \lambda < 1$$

The single step error

- Thus, the overall error made by FF can be bounded in terms of the single step error and the contraction factor.

The single step error

- Thus, the overall error made by FF can be bounded in terms of the single step error and the contraction factor.
- The contraction factor will be the same for any approximation scheme that uses Bayesian inferencing. There exist (purely) theoretical bounds for it [Boyen & Koller'98] (more in Conclusion).

The single step error

- Thus, the overall error made by FF can be bounded in terms of the single step error and the contraction factor.
- The contraction factor will be the same for any approximation scheme that uses Bayesian inferencing. There exist (purely) theoretical bounds for it [Boyen & Koller'98] (more in Conclusion).
- But even the single step error can be high theoretically (close to 1) and in practice (as high as 0.2).

An intuitive example

Suppose we start with a joint distribution:

$$P = \begin{array}{c} M_x \backslash M_y \\ .44 \\ .04 \\ .52 \end{array} \begin{array}{c} \begin{array}{ccc} .02 & .44 & .54 \end{array} \\ \begin{pmatrix} .02 & .4 & .02 \\ & .02 & .02 \\ & .02 & .5 \end{pmatrix} \end{array}$$

Then, FF does the following:

$$B^{FF} = \begin{pmatrix} .44 \\ .04 \\ .52 \end{pmatrix} \times \begin{pmatrix} .02 & .44 & .54 \end{pmatrix} \left[= \begin{pmatrix} .0088 & .1936 & .2376 \\ .0008 & .0176 & .0216 \\ .0104 & .2288 & .2808 \end{pmatrix} \right]$$

Thus, $\max |P - B^{FF}| = .22$.

The hybrid factored frontier algorithm (idea)

Our approach:

Reduce the error made at a single step by FF, and so reduce overall error!

The hybrid factored frontier algorithm (idea)

Our approach:

Reduce the error made at a single step by FF, and so reduce overall error!

1. Observe that the single step error is high only if the joint probability distribution itself has a high value.

The hybrid factored frontier algorithm (idea)

Our approach:

Reduce the error made at a single step by FF, and so reduce overall error!

1. Observe that the single step error is high only if the joint probability distribution itself has a high value.
2. However, there can only be few high values (for instance, only one greater than 0.5).

The hybrid factored frontier algorithm (idea)

Our approach:

Reduce the error made at a single step by FF, and so reduce overall error!

1. Observe that the single step error is high only if the joint probability distribution itself has a high value.
2. However, there can only be few high values (for instance, only one greater than 0.5).

Main Idea:

Maintain the higher values of joint distribution separately and update them directly (as joints). For the rest use the FF algorithm.

The hybrid factored frontier algorithm (idea)

Our approach:

Reduce the error made at a single step by FF, and so reduce overall error!

1. Observe that the single step error is high only if the joint probability distribution itself has a high value.
2. However, there can only be few high values (for instance, only one greater than 0.5).

Main Idea:

Maintain the higher values of joint distribution separately and update them directly (as joints). For the rest use the FF algorithm.

- For details, refer to the report at `www.comp.nus.edu.sg/~suchee/hybridlong.pdf`.

An intuitive example contd.

In HFF, start by fixing a threshold=.4.

1. Run FF to obtain marginals.

$$\text{for } P = \begin{pmatrix} .02 & .4 & .02 \\ & .02 & .02 \\ & .02 & .5 \end{pmatrix}, \; B^{FF} = \begin{pmatrix} .44 \\ .04 \\ .52 \end{pmatrix} \times \begin{pmatrix} .02 & .44 & .54 \end{pmatrix}$$

An intuitive example contd.

In HFF, start by fixing a threshold=.4.

1. Run FF to obtain marginals.

$$\text{for } P = \begin{pmatrix} .02 & .4 & .02 \\ & .02 & .02 \\ & .02 & .5 \end{pmatrix}, \; B^{FF} = \begin{pmatrix} .44 \\ .04 \\ .52 \end{pmatrix} \times \begin{pmatrix} .02 & .44 & .54 \end{pmatrix}$$

2. Then we find 4 candidate spikes, i.e., positions in joint corresponding to high values of all marginals. We maintain these values almost exactly (by an inductive step).

$$\begin{pmatrix} 0 & .38 & .02 \\ 0 & 0 & 0 \\ 0 & .02 & .48 \end{pmatrix}$$

An intuitive example contd.

For the rest, we maintain them as a product of marginals (as in FF). However, the new marginals need to be normalized. With this,

$$B^{HFF} = \begin{pmatrix} 0 & .38 & .02 \\ 0 & 0 & 0 \\ 0 & .02 & .48 \end{pmatrix} + (.1) \times \begin{pmatrix} .4 \\ .4 \\ .2 \end{pmatrix} \times \begin{pmatrix} .2 & .4 & .4 \end{pmatrix}$$

where

$.1 = 1 - (.38 + .48 + .02 + .02)$, $.4 = \frac{(.44 - .40)}{1 - .90} = \frac{\text{marginal} - \text{spikerowsum}}{1 - \text{spikesum}}$.

An intuitive example contd.

For the rest, we maintain them as a product of marginals (as in FF). However, the new marginals need to be normalized. With this,

$$B^{HFF} = \begin{pmatrix} 0 & .38 & .02 \\ 0 & 0 & 0 \\ 0 & .02 & .48 \end{pmatrix} + (.1) \times \begin{pmatrix} .4 \\ .4 \\ .2 \end{pmatrix} \times \begin{pmatrix} .2 & .4 & .4 \end{pmatrix}$$

where
$.1 = 1 - (.38 + .48 + .02 + .02)$, $.4 = \frac{(.44 - .40)}{1 - .90} = \frac{\text{marginal} - \text{spikerowsum}}{1 - \text{spikesum}}$.

$$B^{HFF} = \begin{pmatrix} .008 & .396 & .036 \\ .008 & .016 & .016 \\ .04 & .028 & .488 \end{pmatrix}. \text{ Recalling } P = \begin{pmatrix} .02 & .4 & .02 \\ & .02 & .02 \\ & .02 & .5 \end{pmatrix}$$

$\max|P - B^{HFF}| = .04$ $(\leq .1 = 1 - \text{spikesum})$, $\max|P - B^{FF}| = .22$.

An intuitive example contd.

For the rest, we maintain them as a product of marginals (as in FF). However, the new marginals need to be normalized. With this,

$$B^{HFF} = \begin{pmatrix} 0 & .38 & .02 \\ 0 & 0 & 0 \\ 0 & .02 & .48 \end{pmatrix} + (.1) \times \begin{pmatrix} .4 \\ .4 \\ .2 \end{pmatrix} \times \begin{pmatrix} .2 & .4 & .4 \end{pmatrix}$$

where
$.1 = 1 - (.38 + .48 + .02 + .02)$, $.4 = \frac{(.44 - .40)}{1 - .90} = \frac{\text{marginal} - \text{spikerowsum}}{1 - \text{spikesum}}$.

$$B^{HFF} = \begin{pmatrix} .008 & .396 & .036 \\ .008 & .016 & .016 \\ .04 & .028 & .488 \end{pmatrix}. \text{ Recalling } P = \begin{pmatrix} .02 & .4 & .02 \\ & .02 & .02 \\ & .02 & .5 \end{pmatrix}$$

$\max|P - B^{HFF}| = .04$ ($\leq .1 = 1 - \text{spikesum}$), $\max|P - B^{FF}| = .22$.
Interesting question: How many joints do we need to get high (value of spikesum)? This parameter defines the threshold.

Comparing FF and HFF with exact inference

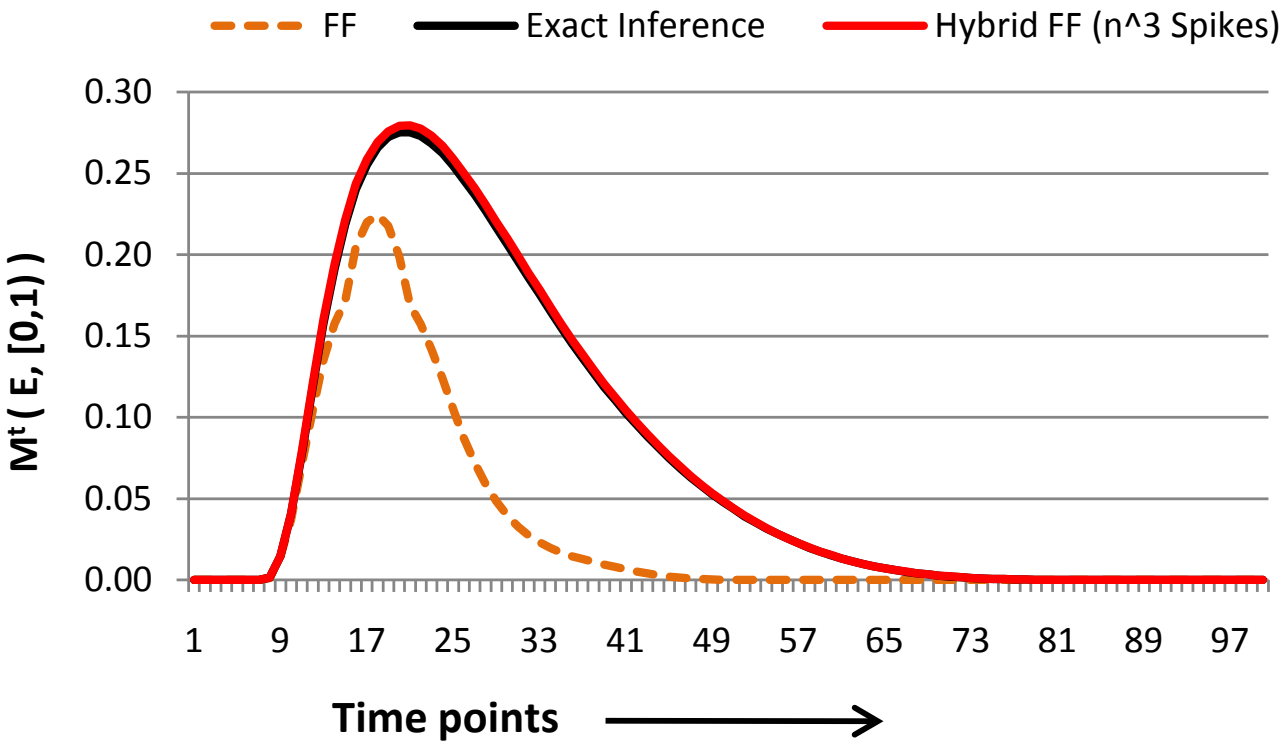


Figure: (marginal) prob of conc of E being in $[0, 1]$ over time

Comparing FF and HFF with exact inference(2)

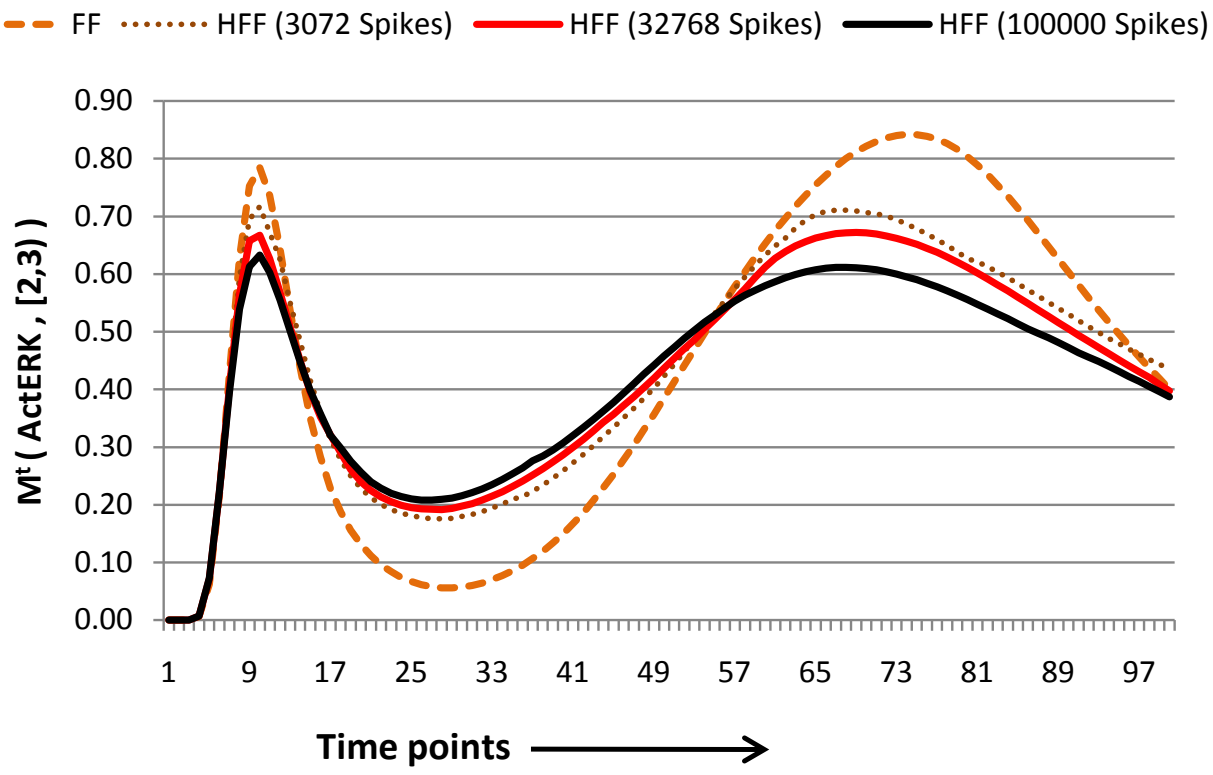


Figure: Time profile of $M^t(ActErk \in [2,3])$ for FF and HFF

Tradeoff

Runtime grows from 0.2 seconds to 150 sec (for 3072 spikes) in the big model with 32 variables.

Issues

Our current concerns

- We want a result like: the marginal prob can be wrong by atmost 0.1. For this, we need to bound on the contraction factor.

Issues

Our current concerns

- We want a result like: the marginal prob can be wrong by atmost 0.1. For this, we need to bound on the contraction factor.
- Improve the running time of HFF. Right now, we only have a naive sequential implementation.

Issues

Our current concerns

- We want a result like: the marginal prob can be wrong by atmost 0.1. For this, we need to bound on the contraction factor.
- Improve the running time of HFF. Right now, we only have a naive sequential implementation.

Again,

- Can we use HFF to perform (approximate) model checking? Probabilistic verification techniques based on logics such as PCTL, PLTL?
- How to use further properties from the system - sparsity, regularity etc?
- We ignore observations, but what happens when they are added?

Some biologically relevant questions

1. Complicated: If Gene encoding Protein A is knocked out is there at least 85% probability that concentration of protein B drops to zero with in the next 10 time points?
2. Behaviour over time: Is there a concomitant change in concentration of protein Y with changes in protein X?
3. Are our estimated parameter values good? How sensitive are the variables? Are some more important than others?...

Idea:

Write the above properties in (possibly extended version of) PCTL, and use the approximate methods developed to model check them!

Some references...

- [Bing & Thiagarajan & Hsu], Probabilistic Approximations of Signaling Pathway Dynamics, CMSB-2009.
- [Murphy & Weiss], The Factored Frontier Algorithm for Approximate Inference in DBN's, UAI-2001.
- [Boyen & Koller], Tractable Inference for Complex Stochastic Processes, UAI-1998.
- For more details, `www.comp.nus.edu.sg/~suchee/hybridlong.pdf`.

Optional slide

More experimental questions

1. Model validation: Do the results we compute match with the experimental data thus validating the model?
2. Prediction: Are our model's predictions consistent with the experimental data? If not, could it indicate some missing phenomenon?