Factor Graph Tutorial

HU, Pili

March 2, 2012

Abstract

Contents

1 A Motivating Example

1.1 Marginalization Problem

Consider a joint probability distribution:

$$p(\vec{x}) \tag{1}$$

where $\vec{x} = \{x_1, x_2, \dots, x_n\}$.

Marginalization operator:

$$p_1(x_1) = \sum_{x_2} \dots \sum_{x_{n-1}} \sum_{x_n} p(\vec{x}) \tag{2}$$

Assuming discrete variable. For continous ones, substitute sum with integral accordingly.

For simplicity of notation, introduce the shorthand "summary" notation:

$$\begin{aligned} p_1(x_1) &= \sum_{\sim\{x_1\}} p(\vec{x}) && (3) \\ &= \sum_{\{x_2, x_3, \dots, x_n\}} p(\vec{x}) && (4) \\ &= \sum_{x_2} \dots \sum_{x_{n-1}} \sum_{x_n} p(\vec{x}) && (5) \end{aligned}$$

The marginalization problem is defined as: Given $p(\vec{x})$, find $p_i(x_i)$. This problem can be generalized as summary for more than one variables.

1.2 Inference Problem

The inference problem is defined as: Given $p(\vec{x}, \vec{y})$, and the observed value of $\vec{y}$, say $\hat{\vec{y}} = \{\hat{y_1}, \hat{y_2}, \dots, \hat{y_n}\}$, find the most probable configuration of $\vec{x}$:

$$\vec{x}^* = \arg\max_{\vec{x}} \{p(\vec{x}|\hat{\vec{y}})\} \tag{6}$$

The conditional probability can be rewritten as:

$$\begin{aligned} p(\vec{x}|\hat{\vec{y}}) &= \frac{p(\vec{x}, \hat{\vec{y}})}{p(\hat{\vec{y}})} && (7) \\ p(\vec{x}|\hat{\vec{y}}) &\propto p(\vec{x}, \hat{\vec{y}}) && (8) \end{aligned}$$

Thus eqn(6) can be rewritten as:

$$\vec{x}^* = \arg\max_{\vec{x}} \{p(\vec{x}, \hat{\vec{y}})\} \tag{9}$$

We'll bridge the gap between the inference problem and marginalization problem defined above later. Now, we start with a toy example.

1.3 Inference with Four Binary Variables

Assume we have a joint distribution $p(a, b, c, d)$. Without any knowledge of the internal structure of the distribution, we can always write it as:

$$p(a, b, c, d) = p(a)p(b|a)p(c|a, b)p(d|a, b, c) \tag{10}$$

Now assume the distribution can be factorized in the following way:

$$p(a, b, c, d) = p(a)p(b)p(c|b)p(d|c) \tag{11}$$

We'll compare two ways of commputing

$$\max_{abcd} p(abcd)$$

using the following data:

$$p(a) = \begin{bmatrix} 0.1 & 0.9 \end{bmatrix} \tag{12}$$

$$p(b) = \begin{bmatrix} 0.2 & 0.8 \end{bmatrix} \tag{13}$$

$$p(c|b) = \begin{bmatrix} 0.4 & 0.6 \\ 0.6 & 0.4 \end{bmatrix} \tag{14}$$

$$p(d|c) = \begin{bmatrix} 0.3 & 0.7 \\ 0.8 & 0.2 \end{bmatrix} \tag{15}$$

1.3.1 Search Max on Joint Distribution Directly

First, we pretend there is no structure information available. Thus a naive way to compute the max is to evaluate the joint distribution everywhere on tha complete alphabets of variables, and then get maximum by comparison.

$$p(abcd) = \left[\begin{array}{c|c} \text{abcd} & \text{Probability} \\ \hline 0000 & 0.0024 \\ 0001 & 0.0056 \\ 0010 & 0.0096 \\ 0011 & 0.0024 \\ 0100 & 0.0144 \\ 0101 & 0.0336 \\ 0110 & 0.0256 \\ 0111 & 0.0064 \\ 1000 & 0.0216 \\ 1001 & 0.0504 \\ 1010 & 0.0864 \\ 1011 & 0.0216 \\ 1100 & 0.1296 \\ 1101 & 0.3024 \\ 1110 & 0.2304 \\ 1111 & 0.0576 \end{array}\right] \tag{16}$$

$$\begin{aligned} p(abcd^*) &= \max_{abcd} p(abcd) && (17) \\ &= p(1101) && (18) \\ &= 0.3024 && (19) \end{aligned}$$

Corresponding computation complexity:

- Function evaluation: $16 \times 4 = 64$
- Product: $16 \times 3 = 48$
- Comparison(for max operator): 15

1.3.2 Search Max Intelligently

Indeed, eqn(11) conveys useful information by the factorization of the joint probability.

Let's expand the maximization $p(a, b, c, d)$

$$\begin{aligned} \max_{abcd}\{p(abcd)\} &= \max_{abcd}\{p(a)p(b)p(c|b)p(d|c)\} && (20) \\ &= \max_{a}\{p(a)\} \max_{bcd}\{p(b)p(c|b)p(d|c)\} && (21) \\ &= \max_{a}\{p(a)\} \max_{d}\{\max_{bc}\{p(b)p(c|b)p(d|c)\}\} && (22) \\ &= \max_{a}\{p(a)\} \max_{d}\{\max_{c}\{\max_{b}\{p(b)p(c|b)\}p(d|c)\}\} && (23) \end{aligned}$$

$$\max_{a}\{p(a)\} = \max_{a} f_a(a) = 0.9 \qquad (24)$$

$$\begin{aligned} \max_{b}\{p(b)p(c|b)\} &= \max_{b} f_{bc}(bc) && (25) \\ &= \max_{b} \left[\begin{array}{c|c} \text{bc} & \text{Probability} \\ \hline 00 & 0.08 \\ 01 & 0.12 \\ 10 & 0.48(*) \\ 11 & 0.32(*) \end{array} \right] && (26) \\ &= \left[\begin{array}{c|c} \text{c} & \text{Probability} \\ \hline 0 & 0.48 \\ 1 & 0.32 \end{array} \right] && (27) \end{aligned}$$

Denote $\max_b\{p(b)p(c|b)\}$ by $\mu_{bc}(c)$.

$$\max_c\{\mu_{bc}(c)p(d|c)\} = \max_c f_{cd}(cd) \tag{28}$$

$$= \max_c \left[\begin{array}{c|c} \text{cd} & \text{Probability} \\ \hline 00 & 0.144 \\ 01 & 0.336(*) \\ 10 & 0.256(*) \\ 11 & 0.064 \end{array}\right] \tag{29}$$

$$= \left[\begin{array}{c|c} \text{d} & \text{Probability} \\ \hline 0 & 0.256 \\ 1 & 0.336 \end{array}\right] \tag{30}$$

Denote $\max_c\{\mu_{bc}(c)p(d|c)\}$ by $\mu_{cd}(d)$.

$$\max_d\{\max_c\{\max_b\{p(b)p(c|b)\}p(d|c)\}\} \tag{31}$$

$$= \max_d\{\mu_{cd}(d)\} \tag{32}$$

$$= 0.336 \tag{33}$$

Thus we get final result:

$$\max_{abcd}\{p(abcd)\} = \max_a\{p(a)\} \times \max_d\{\mu_{cd}(d)\} \tag{34}$$

$$= 0.3024 \tag{35}$$

Again, we calculate the computation complexity:

- Function evaluation:

$$2 + 4 \times 2 + 4 \times 2 + 2 = 20$$

- Product:

$$0 + 4 \times 1 + 4 \times 1 + 0 + 1 = 9$$

- Comparison(for max operator):

$$1 + 2 \times 1 + 2 \times 1 + 1 = 6$$

1.3.3 Observation

We compare the complexity of two methods in table(1).

The observations:

- By properly using the structure of joint distribution, it's possible to reduce computation complexity.

Table 1: Comparison Between Two Methods

Items	Naive	Intelligent
Function	64	20
Product	48	9
Comparison	15	6

- The trick to reduce complexity in the second method is: "product" is distributive through "max". Thus we can separate some variables when evaluating the maximum of others. We'll address this issue in details later.
- How to reveal and utilize the structure in a systematic way is still a problem.

1.4 Inference with One Observed Variable

Here's another example. Assume c is observed to be 1 in the last example. What's the new most probable configuration of other variables?

1.4.1 Naive

As before, simply restrict the evaluation of functions only on points where $c = 1$.

$$p(abcd) = \left[\begin{array}{c|c} \text{abcd} & \text{Probability} \\ \hline 0010 & 0.0096 \\ 0011 & 0.0024 \\ 0110 & 0.0256 \\ 0111 & 0.0064 \\ 1010 & 0.0864 \\ 1011 & 0.0216 \\ 1110 & 0.2304 \\ 1111 & 0.0576 \end{array}\right] \tag{36}$$

The most probable configuration of the four variables is 1110, and corresponding probability is 0.2304(the joint probability, not the probability of $a = 1, b = 1, d = 0$ conditioned $c = 1$).

1.4.2 Intelligent

With the observation of $c = 1$, the joint distribution can be decomposed as:

$$\max_{abd}\{p(ab1d)\} = \max_{abd}\{p(a)p(b)p(c=1|b)p(d|c=1)\} \tag{37}$$

$$= \max_{a}\{p(a)\} \max_{b}\{p(b)p(c=1|b)\} \max_{d}\{p(d|c=1)\} \tag{38}$$

$$\max_a\{p(a)\} = \max_a f_a(a) = 0.9 \tag{39}$$

$$\begin{aligned}
\max_b\{p(b)p(c=1|b)\} &= \max_b f_{bc}(bc, c=1) && (40)\\
&= \max_b \left[\begin{array}{c|c} \text{bc} & \text{Probability} \\ \hline 01 & 0.12 \\ 11 & 0.32 \end{array}\right] && (41)\\
&= 0.32 && (42)
\end{aligned}$$

$$\begin{aligned}
\max_d\{p(d|c=1)\} &= \max_d \left[\begin{array}{c|c} \text{cd} & \text{Probability} \\ \hline 10 & 0.8 \\ 11 & 0.2 \end{array}\right] && (43)\\
&= 0.8 && (44)
\end{aligned}$$

Thus the final maximum probability is given by:

$$\begin{aligned}
& \max_a\{p(a)\} \max_b\{p(b)p(c=1|b)\} \max_d\{p(d|c=1)\} && (45)\\
= \; & 0.9 * 0.32 * 0.8 && (46)\\
= \; & 0.2304 && (47)
\end{aligned}$$

1.4.3 Observation

Now that we validated the correctness of our intelligent method, we again compare the complexity as is in table(2).

Table 2: Comparison Between Two Methods

Items	Naive	Intelligent
Function	32	8
Product	24	4
Comparison	7	3

Besides previous observations on the value of "structure", we highlight one more thing:

- When the variable c is observed, the joint distribution function can be further decomposed! That is, in previous example, there is a relationship between b and d, so we evaluate max operator in the order of b, then c, then d. However, with the observation of c, the sub functions involving b and d are fully decoupled, this further reduces the complexity.
- This observation is indeed the notion of conditional independence, as is one major concern in some graphical models like MRF and BN.

2 Factor Graph Specification

3 Factor Graph Transformation

Transformation of loopy factor graph can result in tree structures. Thus ordinary sum-product algorithm can be used to obtain an exact solution.

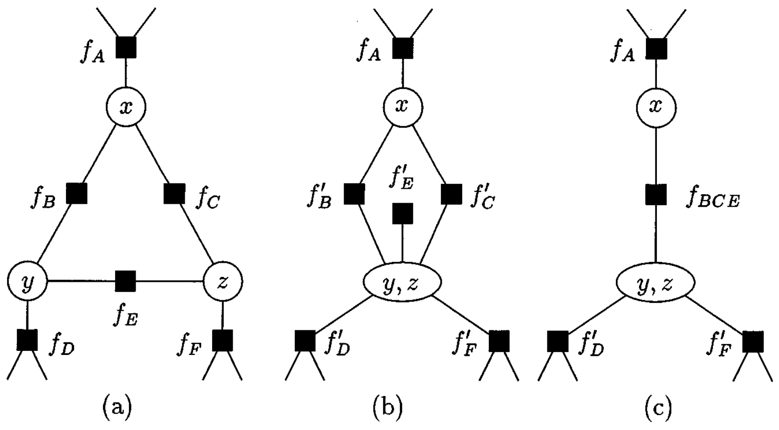

Figure 1: Transformation: Clustering

Effect of clustering:

- Cluster variable nodes: domain enlarged; no complexity increasing in local functions; increase complexity in sum-product algorithm (function evaluation).
- Cluster function nodes: do not increase complexity of variables; increase complxity in sum-product algorithm (sizes of messages are increased).

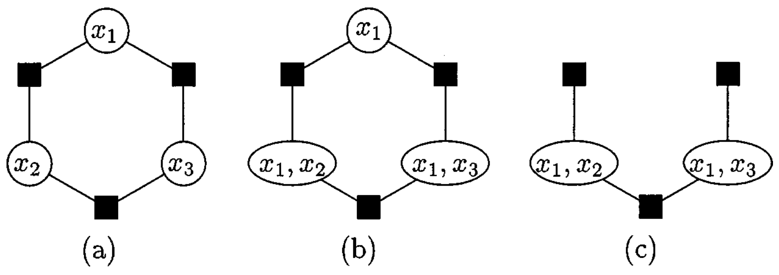

Figure 2: Transformation: Streching

Effect of stretching:

- Simply stretching variables adds positional variables to some local functions. Local function complexity is not changed. Variable alphabets are enlarged.

- After stretching, redundant links or even redundant variables can be removed. By systematic stretching, all cycles can be eliminated.

4 Selected Applications

4.1 Model System Dynamics

This section follows the paper:
P. Mirowski and Y. LeCun. Dynamic factor graphs for time series modeling. *Machine Learning and Knowledge Discovery in Databases*, v:128–143, 2009.

Declaration: figures in this section are borrowed from the orignal paper[4]. We omit further citation for simplicity.

4.1.1 Quick Note of the Main Idea

Problem settings:

- System is composed of state variables and observation variables.
- Observation function maps state to observation.
- Dynamic function governs state transitions.
- Aim at modeling high dimensional underlying dynamic, probably non-linear, but deterministc.

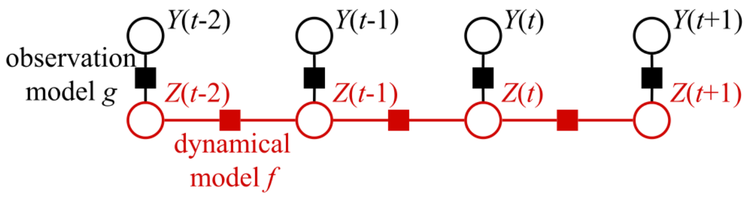

Figure 3: Single State Dependency(HMM)

When current state is only dependent on previous one state, it is like an HMM model.

The state transition model can be more general that it may depend on several previous states, or even observations.

Formulation of paramterized FG:

$$L(\mathbf{W}, Y, Z) = E(\mathbf{W}, Y) + R_z(Z) + R(\mathbf{W}) \tag{48}$$

$$\tilde{Z} = \arg\max_{Z} L(\tilde{\mathbf{W}}, Y, Z) \tag{49}$$

$$\tilde{\mathbf{W}} = \arg\max_{\mathbf{W}} L(\mathbf{W}, Y, \tilde{Z}) \tag{50}$$

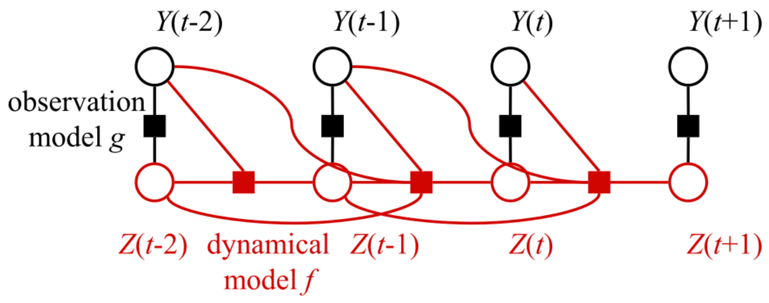

Figure 4: Multiple States and Observation Dependency

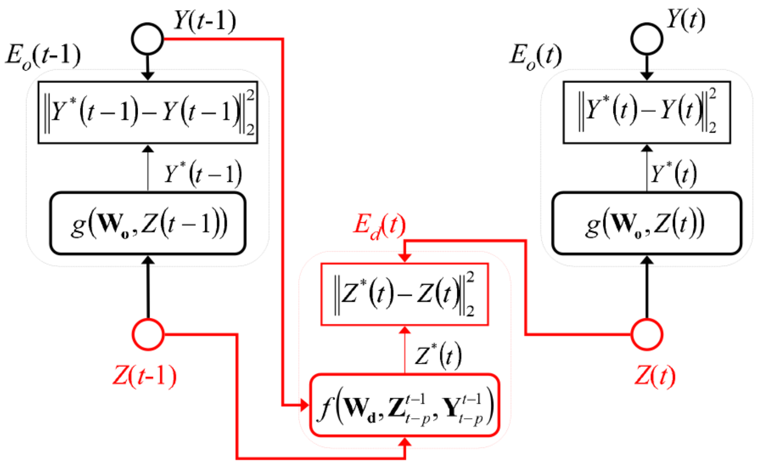

Figure 5: System Details

Since the paper operates in the negative log domain, they want to minimize L. The more likely some configuration is, the energy E is lower. As is depicted in fig(), energy function is the square error.

When parameters are fixed, eqn(49) acts as what ordinary factor graph does, namely, given all explicit function definitions, compute the marginalization problem. In ths application, paramters $\mathbf{W}$ is not determined. An usual way to tackle with this problem is by Expectation-Maximization schema (sometimes called Generalized Expectation Maximization algorithm). eqn(49) resembles the E-step, while eqn(50) resembles the M-step.

To decouple 5 superimposed sine waves, the authors choose the architecture:

- 5 dimensional state variable.
- Dynamic function(f): 5 independent FIR filter of order 25.

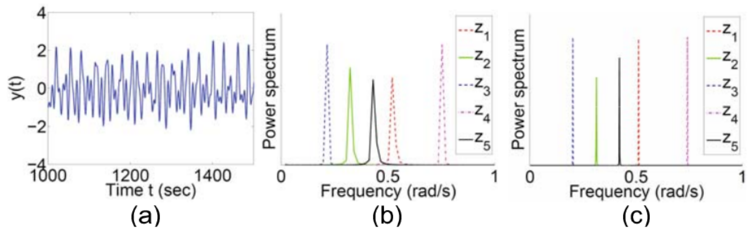

Figure 6: Training and Evaluation on Superimposed Sine Waves

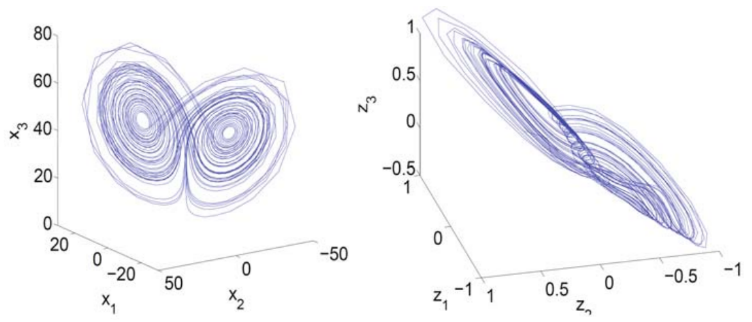

Figure 7: Training and Evaluation on Lorenz Curve

To capture the observation of 1D variable whose underlying dynamic is 3D-Lorenz curve, the authors choose the architecture:

- 3 dimensional state variable.
- Dynamic function(f): 3 layered convelutional network.

Besides the two evaluation mentioned above, the original paper contains more. Interested readers can refer to [4] for more information.

4.1.2 Comments

- DFG(in this paper) is basically FG. "dynamic" only address the application aspects, rather than claiming an extension of FG.
- The framework of FG is too general. Designing observation function and state transition function, choice of regularizer, setting coefficients are really tricky. There doesn't seem to be a rule of thumb.
- The introduction of unknown parameters and regularizers make the formulation deviate from ordinary factor graph. Before final marginalization, there is a training stage to obtain opimal paramters.

4.2 Solving Marginalization Problem in Social Network

This section follows the paper:
C. Wang, J. Tang, J. Sun, and J. Han. Dynamic social influence analysis through time-dependent factor graphs. *ASONAM*, v:p, 2011.

Declaration: figures in this section are borrowed from the orignal paper [5]. We omit further citation for simplicity.

4.2.1 Quick Note of the Main Idea

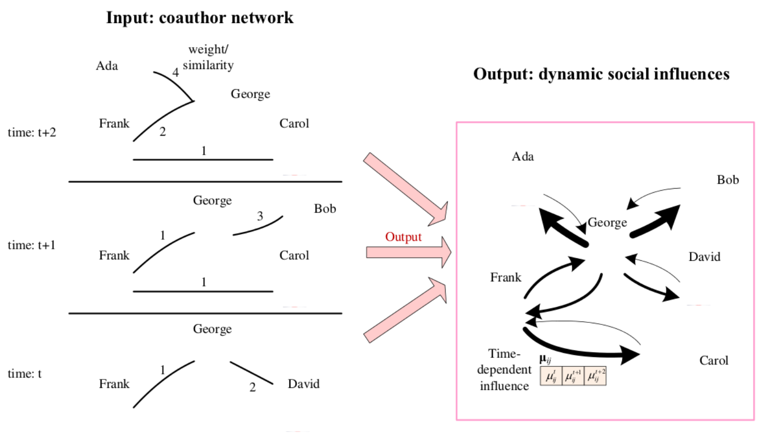

Figure 8: Problem Definition

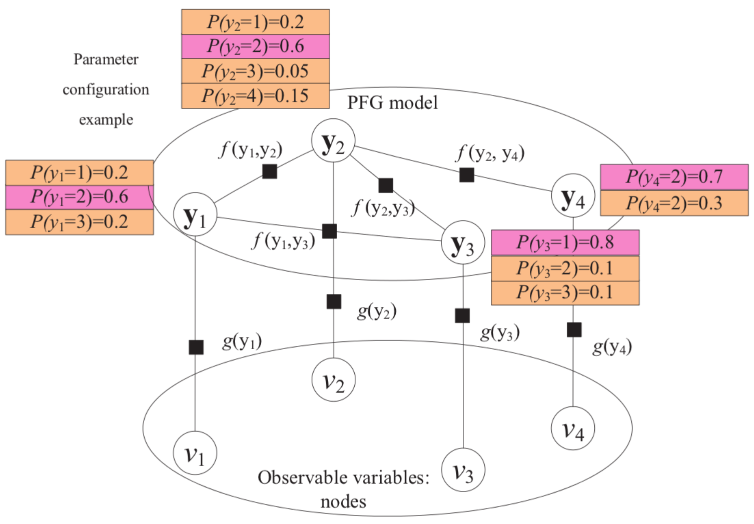

Figure 9: Visualization of the Factor Graph

$$g(y_i|V) \quad = \quad \begin{cases} \frac{w_{iy_i}}{\sum_{j\in NB(i)}(w_{ij}+w_{ji})} & y_i \neq i \\ \frac{\sum_{j\in NB(i)} w_{ji}}{\sum_{j\in NB(i)}(w_{ij}+w_{ji})} & y_i = i \end{cases}$$

Figure 10: Choice of Node Factor Function

$$f(y_i, y_j|V) = \begin{cases} \frac{u}{|SC(i)\cap SC(j)|} & y_i = y_j \\ \frac{1-u}{|SC(i)||SC(j)|-|SC(i)\cap SC(j)|} & y_i \neq y_j \end{cases}$$

Figure 11: Choice of Link Factor Function

4.2.2 Comments

5 Other Taste of Modeling

6 Related Models/Algorithms

Models:

- Bayesian Network. Directed Graphical Model.
- Markov Random Field. Undirected Graphical Model.
- Chain Graph. Directed + Undirected.
- Factor Graph.

Justification of modeling ability:

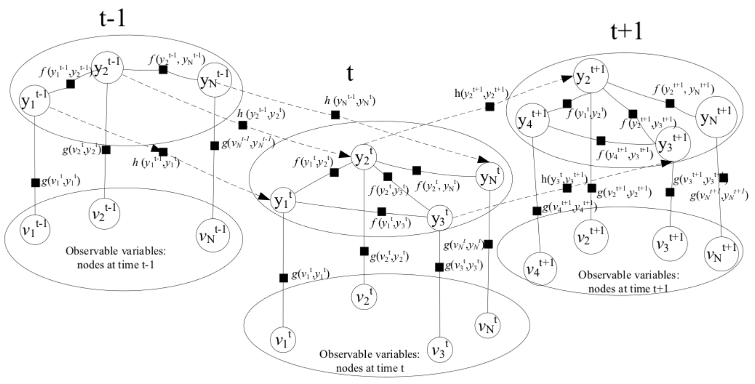
t-1
t
t+1
Observable variables: nodes at time t-1
Observable variables: nodes at time t
Observable variables: nodes at time t+1

Figure 12: Factor Graph with Time Dependency

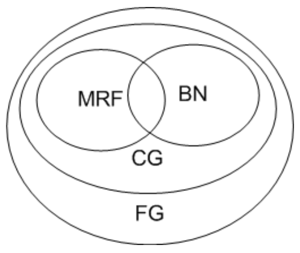

Figure 13: Venn Graph of Several Graphical Models

- BN $\to$ MRF. The moralization process hides certain stucture previously available in BN.
- MRF. An example of square located 4 nodes. If two diagonal variables are observed, the other two variables become independent. BN can not express such information.
- CG. Since both directed and undirected, it forms the superset of the union of BN and MRF.
- FG. Converting CG to FG reveals more details, namely, how the joint distribution is factorized(usually, one explicit normalization function exhibits). BTW, FG can model not only probabilistic problems, but also other problems like system dynamics and coding. It is super general in the sense.

Algorithms:

- Junction tree.
- Bayesian ball.
- (Loopy)Belief propagation.
- Sum-product.

7 Discussions

Since the development of factor graph is boosted in the past decade, different authors come up with different description of similar problems. Not to distinguish right from wrong, I just regard those stuffs out there as inconsistent. My opinion on some parts of past literature:

- In Bishop's book[1], chapter 8.4.5, P411. The example is not good. Actually, when talking about that probability maximization problem, we should know "product" corresponds to product operator, and "sum" corresponds to max operator. In this case, the maginalization operation for a single varialbe is indeed the maximization for each instancde of that variable. Using local marginalized function(max), we can certainly get the global probability maximization point considering all variables.

- As for Dynamic Factor Graph, the author of this paper do not advocate the abuse of this term like an extension of factor graph. FG itself is able to model system dynamics, as we've already seen in those examples above. Other authors may use the term DFG [5] [4] , but their DFG is application specific. Those graphs are essentially FG. Not until we examine the physical meaning of some factor nodes do we realize their "dynamic" property.

References

[1] C.M Bishop. *Pattern recognition and machine learning*, volume 4. springer New York, 2006.

[2] B.J. Frey. Extending factor graphs so as to unify directed and undirected graphical models. In *Proc. 19th Conf. Uncertainty in Artificial Intelligence*, pages 257–264, 2003.

[3] F.R. Kschischang, B.J. Frey, and H.A. Loeliger. Factor graphs and the sum-product algorithm. *Information Theory, IEEE Transactions on*, 47(2):498–519, 2001.

[4] P. Mirowski and Y. LeCun. Dynamic factor graphs for time series modeling. *Machine Learning and Knowledge Discovery in Databases*, v:128–143, 2009.

[5] C. Wang, J. Tang, J. Sun, and J. Han. Dynamic social influence analysis through time-dependent factor graphs. *ASONAM*, v:p, 2011.

[6] Moral Graph, Wikipedia, `http://en.wikipedia.org/wiki/Moral_graph`

[7] Markov Blanket, Wikipedia, `http://en.wikipedia.org/wiki/Markov_blanket`