Labeled graph notations for graphical models
Extended Report

Eric Mjolsness

Department of Computer Science
School of Information and Computer Science
University of California, Irvine
emj@uci.edu

March 2004
UCI ICS TR# 04-03

Abstract

We introduce new diagrammatic notations for probabilistic independence networks (including Bayes nets and graphical models). These notations include new node and link types that allow for natural representation of a wide range of probabilistic data models including complex hierarchical models. The diagrammatic notations also support models defined on variable numbers of complex objects and relationships. Node types include random variable nodes, index nodes, constraint nodes, and an object supernode. Link types include conditional dependency, indexing and index limitation, variable value limitation, and gating a dependency between nodes or objects by an arbitrary graph. Examples are shown for clustering problems, information retrieval, unknown graph structures in biological regulation, and other scientific domains. The diagrams may be taken as a shorthand notation for a more detailed syntactic representation by an algebraic expression for factored probability distributions, which in turn may be specified by stochastic parameterized grammar or graph grammar models. We illustrate these ideas with previously described applications and potential new ones.

1. Extending graph notation for dependency networks

In this section we will outline current practice in labeled-graph description of probabilistic models, and propose several useful extensions of the notation syntax, along with their semantics in terms of probability distributions. In section 2 we will show how to create hierarchical versions of nonhierarchical models. In section 3 we will apply the work to models of variable-structure systems, and we will express object models using the extended graph notation. In sections 4 and 5 we will discuss scientific application domains in biology and geology. In section 6 we will briefly discuss stochastic graph grammars, and conclude.

1.1 Dependency graph notation, with plates

Consider a Directed Probabilistic Independence Network (DPIN) [Smyth et al. 1997] or Bayes Network [Pearl 1988] as applied to the representation of a mixture-of-Gaussians

data model for clustering. We can introduce N real vector-valued, observable random variables $\boldsymbol{x}_i$, each sampled from one of A normal distributions with mean $\boldsymbol{y}_a$ and standard deviation or covariance $\boldsymbol{\sigma}_a$. The choice of Gaussian is governed by a class membership variable a_i, which is chosen from a discrete distribution on the A classes with probability ρ_a. In the simplest situation, a and $\boldsymbol{x}$ are random variables and all other quantities are externally imposed and known constants. A minimal inference problem is to observe $\boldsymbol{x}_i$ and determine a_i. (The usual inference formulation of clustering also takes the $\boldsymbol{y}$'s and perhaps the σ's as random variables to be inferred, as will be discussed below.) For random a and $\boldsymbol{x}$, the network of probabilistic dependencies can be diagrammed as in Figure 1.

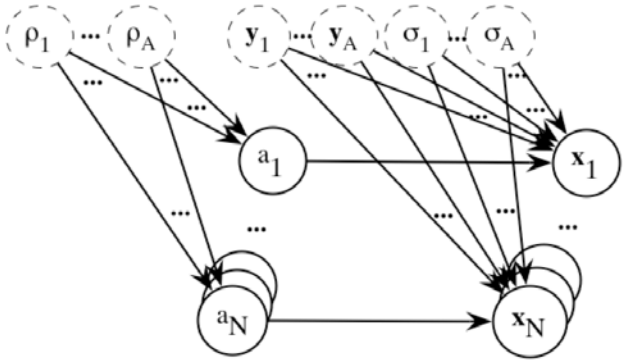


Figure 1. Directed probabilistic dependence network for a mixture of Gaussians model. Note repetition (marked by ellipses) of nodes and arrows in both *{1..N}* and *{1..A}* axes. A full clustering problem is obtained by making at least the cluster means, **y**, also be random variables rather than constants.

If the diagram of Figure 1 were fully labeled with conditional probability distributions, the mapping to a large multidimensional joint probability distribution would be the "semantics" of the diagrammatic notation. Without such labels, the semantic mapping is ambiguous but often sufficient for human communication. Both the conditional and the joint distributions are in general arbitrary probability densities but are usually also specifiable as symbolic algebraic expressions whose structure is reflected in the diagram. In this particular case the probability associated with a particular data vector is

$$\Pr(a,\mathbf{x} \mid \mathbf{y},\sigma) = \rho_a G(\mathbf{x} \mid \mathbf{y}_a,\sigma_a), \qquad (1)$$

where G is the Gaussian or Normal distribution, and therefore the global joint probability distribution is

$$\Pr(\{a_i,\mathbf{x}_i \mid i \in 1..N\} \mid \mathbf{y},\sigma) = \prod_i \rho_{a_i} G(\mathbf{x}_i \mid \mathbf{y}_{a_i},\sigma_{a_i}). \qquad (2)$$

This distribrution may be reexpressed using 0/1 cluster membership indicator variables M as

$$\Pr\left(\left\{M_{ia}, \mathbf{x}_i \mid i \in 1..N, a \in 1..A\right\} \mid \mathbf{y}, \sigma\right) = \prod_{ia} \left[\rho_a G\left(\mathbf{x}_i \mid \mathbf{y}_a, \sigma_a\right)\right]^{M_{ia}} . \quad (3)$$

For this particular problem, more elaborate diagrams and their distributions could include a prior for **y** (usually Gaussian as it is the conjugate distribution) and σ (Gamma or more generally a Wishart distribution), and even *N*, in which case statistical inference encompasses (supervised or unsupervised) learning as well as classification. This problem can be solved with the Expectation-Maximization algorithm [Bishop 1995]. As an aside, $\boldsymbol{\sigma}$ may be decorated with index *a* or not, and may be indexed by the internal d-dimensional index either zero times (scalar covariance), one time (diagonal covariance), or twice (symmetric covariance matrix), for a total of six different models.

Such DPIN (Bayes Net) diagrams can also be related to Undirected Probabilistic Independence Networks (UPIN) [Smyth et al. 1997] and thus to Markov Random Field formulations [Kinderman and Snell 1980] [Hinton and Sejnowski 1983] [Geman and Geman 1984] and to the more general Boltzmann distributions long employed in statistical physics.

The DPIN diagram above necessarily includes ellipses representing the large number of random variables present in a network with an arbitrary number A of classes giving rise to an arbitrary number N of data vectors. The iteration over many variables can be expressed diagrammatically, and the ellipses can be removed, by using the "plates" notation [Buntine 1994] [Jordan 2003] in such diagrams as figure 2(a) and 2(b).

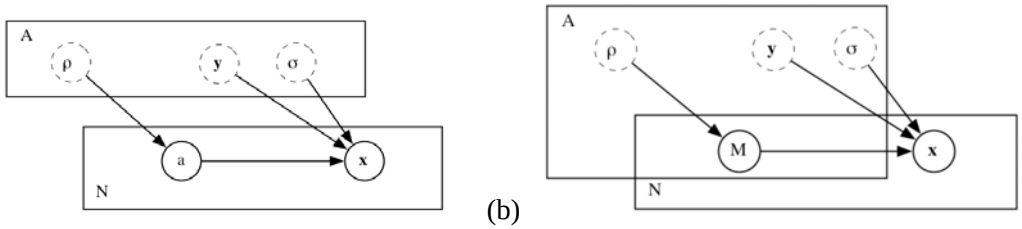


Figure 2. "Plate" diagrams for the mixture of Gaussians model. Here, each box or "plate" indicates replication by the number of times indicated with an enclosed index limit, *A* or *N*. Random variables in multiple plates are multiply indexed and replicated as a cross product. This specifies the same topology as Figure 1, without the ellipses.

Associated with every such model is at least one underlying probability distribution which provides the "meaning" of the diagram. A general formulation of the distributions compatible with a given UPIN diagram is

$$\Pr\left(\left\{x_i \mid i \in \{1..N\}\right\}\right) = \prod_c \phi_c\left(\left\{x_j \mid x_j \in C_c\right\}\right) = \exp\left[-\beta \sum_c V_c\left(\left\{x_j \mid x_j \in C_c\right\}\right)\right] \quad (4)$$

Here each ϕ_c is a nonnegative function of the variables in a set C_c indexed by *c*. These sets may correspond to cliques in the "triangulated" UPIN derived from the DPIN, but

more generally they can be arbitrary interactions among variables that can't be reduced solely to a combination of lower order interactions. Equivalently the energy function potentials V_c are functions of the same variable sets C_c.

1.2 Index nodes and links

The "plates" notation has the drawback that random variable nodes must be grouped together geometrically in order to share the same index or set of indices. With rectangular or even convex plates in 2D, this can only be done in general for a small number of variables. Furthermore, the plate notation is not literally a labeled graph representation and thus not subject to graph composition operations or automatic manipulation. In practice, the number of overlaps is sometimes reduced by omitting important details of the model.

To remove these problems and make dependency graph notation more usefully expressive, we propose to "shrink" the plates down to nodes of a particular type. These special "index nodes" can be distinguished from "random variable nodes" and "external parameter" nodes by enclosing their symbolic names in small squares rather than circles, or dotted circles (or even no enclosure) respectively. Now the clustering model looks as shown in Figure 3.

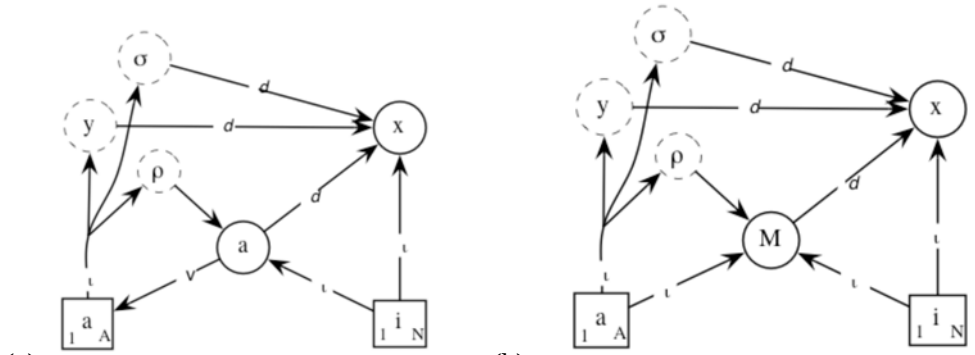


Figure 3. (a) Clustering model with cluster number variables a_i. (b) Equivalent clustering model with cluster membership indicator variables M_{ia}.

In these indexed probabilistic diagrams (IPD's), links labeled by "*d*" (or not labeled at all) are probabilistic dependency links, those labeled by "ι" are indexing links from an index node to a random variable node, and those labeled by "*v*" indicate that a random variable takes values in an index set. The "*M*" variable in Figure 3b is an indicator variable taking values in {0, 1} and constrained to indicate a unique class *a* for each datapoint $\mathbf{x}_i$.

Figure 4 shows a fuller diagrammatic specification of the clustering model. Still the only random variables are M_{ia} (or a_i) and $\mathbf{x}_i$, but now the full indexing as shown including the internal index *j* for the vectors **x** and **y**. A further refinement of the notation allows index variables to be limited by upper and/or lower bounds, or other descriptors.

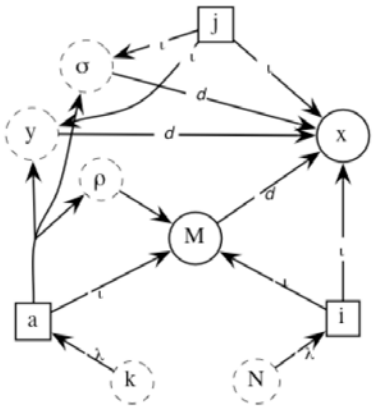


Figure 4. Mixture model showing addition of index *j* for dimensions of the feature vector space, and constant nodes for index limits.

A fourth arrow type labeled by "λ" (Figure 4) specifies the index range limits such as *A* (or *k*) and *N* as variables pointing into the index nodes, and a few more node and arrow types for IPD's will be introduced in the remainder of the paper. An alternative notation would be to use different kinds of lines (solid, dotted, wavy, springlike, …) for the different arrow types, as well as default arrow labels such as "ι" for arrows coming out of index nodes and "*d*" for arrows coming out of random variable nodes. These assignments may best be done after gaining more experience with the diagrams.

The index nodes have many advantages including the natural correspondence between graph-editing and model-editing operations. For example, any random variable node may be turned into a vector by indexing it with an appropriate new index node.

1.3 Connections to Stochastic Grammars and Bayes Networks

The actual probability distribution summarized by Figures 3 and 4 is the same as for Figures 1 and 2. It can be specified in full by a stochastic parameterized grammar [Mjolsness 1997]. For example, Figure 5 shows a stochastic parameterized grammar associated with the mixture mode above. The stochastic grammar formalism can be used to concisely express much more elaborate probabilistic models as will be shown later.

grammar $\text{mix}(\text{dataset} \to \{\text{datapoint}(\mathbf{x}_i) \mid i \in I\})$

{

$\text{dataset} \to \{\text{classmember}(a_i) \mid i \in I\}$ // a = class number

with $\Pr(a_i) = \begin{cases} \rho_{a_i} & \text{if } a_i \in \{1..A\} \\ 0 & \text{otherwise} \end{cases}$

$\text{classmember}(a_i) \to \text{datapoint}(x_i),\ \text{membership}(i, a_i)$

with $\mathbf{x}_i \sim G(\mathbf{y}_{a_i}, \sigma_{a_i})$

}

Figure 5. Reexpression of a Gaussian mixture model using two probabilistic rules in a stochastic grammar.

Each structured probabilistic model may be presented as an indexed probabilistic diagram (IPD), as an algebraic expression for probability or energy, and/or as a stochastic parameterized grammar (SPG). The IPD is a somewhat ambiguous shorthand for the other two specifications, unless it is annotated in detail.

As a second example, we describe the conventional static Bayes Network formalism in three ways: a probability distribution formula, a stochastic parameterized grammar (SPG), and a graph diagram that acts as a shorthand for either of the other descriptions. Given random variables X_i indexed by i in $I=\{1\ldots N\}$, and given a Directed Acyclic Graph of dependencies $G_{ij} \in \{0,1\}$ for which (after permuting the index values i) $G_{\{i<j\}} = 0$, we can write

$$\Pr\left(\left\{X_i = x_i \mid i \in I\right\}\right) = \prod_i \Pr\left(x_i \mid \left\{x_j \mid G_{ij} = 1\right\}\right), \qquad (5)$$

and the very simple SPG is

grammar StaticBayesNet(start → $\{X_i\ (x_i) \mid i \in I\}$) {

start → $\{ X'_i \mid i \in I\}$

X'_i , $\{X_j\ (x_j) \mid G_{ij}=1\}$ → $X_i\ (x_i)$, $\{X_j\ (x_j) \mid G_{ij}=1\}$

with $\Pr\left(x_i \mid \left\{x_j \mid G_{ij} = 1\right\}\right)$

}

Any legal order of rule firing will produce a valid sample from the joint probability distribution. Note that variables with no predecessors in the DAG G have their prior distributions $P(x_i \mid \{\ \}) = P(x_i)$ used automatically in both of these descriptions, which is correct. (Here {} is the empty set of random variables.) Also note that this grammar uses many terms on the left hand side of a rule, so that it is no longer context-free.

The corresponding dependency graph description is shown in Figure 6 below:

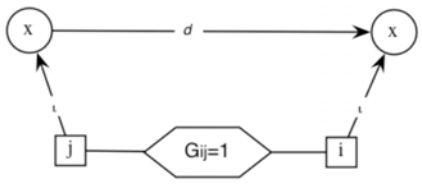


Figure 6. Bayes network diagram with index nodes.

Implicit in this representation is the constraint that the same random variable is indexed, for the same numeric value of i and j, even if it appears in two different places in the diagram. If G were not a DAG, then there would be a further ambiguity in attempting to

unroll a cycle in the graph *G* for which a variable can depend (perhaps indirectly) on itself. But here, *G* is assumed to be a DAG.

1.4 Interaction/Constraint nodes

Each dependency link in the foregoing diagrams is associated with a dependency in a conditional probability distribution P(node | predecessor nodes). Each such conditional distribution is a multiplicative factor ϕ in the global joint distribution (equation (4)), or equivalently an additive term (such as a clique potential in a Markov Random Field) in the global energy function. For some purposes it is useful to explicitly represent such multiplicative probability factors (or additive energy terms) in the diagrams. For example, some dependencies may naturally factor into several conditional distribution terms with each with fewer random variables. Also, constraints can be represented as probability factors ϕ such as Kronecker or Dirac delta functions which take the value zero wherever the constraint is violated, and one variable may be limited by several different constraints. These models may be represented by a new "interaction" node type, represented here as a hexagon enclosing textual or algebraic constraint information pertaining to probability factor ϕ. The semantics of the whole diagram is then given by equation (4), with any (index node, constraint node) pair connected through at least one random variable mapping to a summation of interactions over that index node in the energy function.

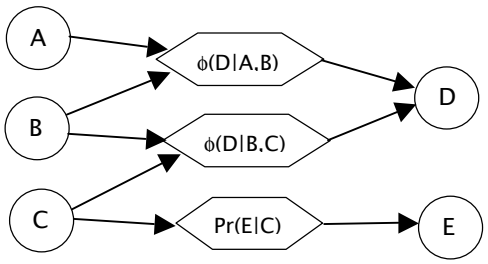


Figure 7. Illustration of probability factor/constraint node type. This factorization cannot be expressed by a Bayes Net [Frey 2003].

Undirected edges can be used for multiplicative factors in the overall joint probability distribution (such as monomials in an energy function, or clique potentials in an MRF). The constraint notation can also be combined with index nodes to impose constraints on several indices and/or random variables. Very similar ideas have been worked out in the "factor graphs" of [Kschischang et al. 2001] [Frey 2003] which are bipartite graphs with variable and constraint nodes with essentially the semantics of Boltzmann distributions, as in equation (4). They have been extended so that directed and undirected links may be mixed. They do not have index nodes or the other node and link types introduced below.

Given this full graph notation, all indexing relationships in a model can be displayed without omission, and the labeled graph representation can be the substrate for new graph editing and composition operations. Some of the new graph composition operations have

natural semantics of relevance to probabilistic data modeling. Two major improvement possible with the improved graph notation are (a) easy introduction of hierarchical structure into generative models, and (b) ability to express "variable-structure systems", with objects whose number and relationships vary as a function of other random variables.

2. Introduction of Model Hierarchy

As another example of the expressive power of the diagrammatic index notation, subscripts can have subscripts as in $x_{(i1, \ldots id)}$ as in Figure 9a or (abbreviated) 9b.

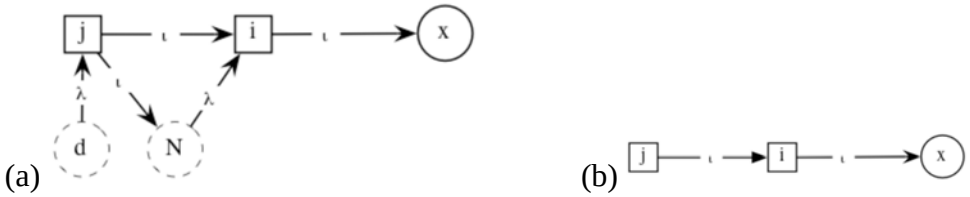


Figure 9. Hierarchical index specification, (a) with index limit nodes, and (b) without.

This is useful for representing d-dimensional multi-resolution image pyramids in vision, where for example an image variable x might be indexed as $x_{(l,(i_{11},\ldots i_{1d}),\ldots(i_{L1},\ldots i_{Ld}))}$, diagrammed in Figure 10a below. Then every lower-level index such as $(i_{21},\ldots i_{2d})$ takes on the same set of value regardless of value of the parent index such as$(i_{11},\ldots i_{1d})$ so that the tree is automatically balanced complete. For incomplete, unbalanced index trees one can allocate a separate lower level index, e.g.

$$i_{i_1} \text{ or } \left(i_{i_{11}1},\ldots i_{i_{1d}d}\right),$$

for each value of the higher level index i_1 or $(i_{11},\ldots i_{1d})$. This leads to unsightly sub-sub-sub indices in the textual representation, but is natural in the diagrammatic representation as shown in Figure 10b.

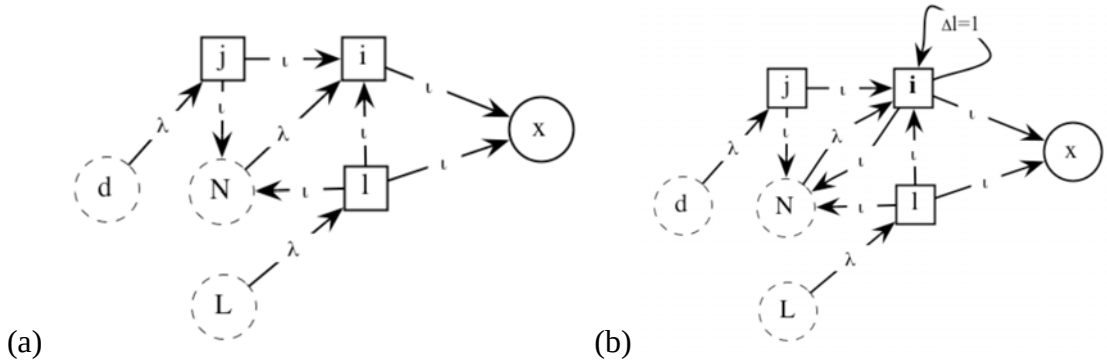


Figure 10. Alternative indexing schemes for multiscale grids. (a) Regular pyramid. (b) Spatially variable depth of resolution in a pyramidal grid. In this scheme, $N_{id} = 0$ determines that $\boldsymbol{i}$ is a leaf node in a tree of indices for dimension d. Otherwise, the tree goes deeper (to higher level number

l) for that value of i. Also, for x_{il} at level l , indexing of x by ***i*** is gated by l (see section 4 for a diagrammatic notation) so that only ***i***'s indexed by level l can index x_{il} .

2.1 Example: Hierarchical Clustering

Here is a two-level hierarchical generalization of the mixture model above.

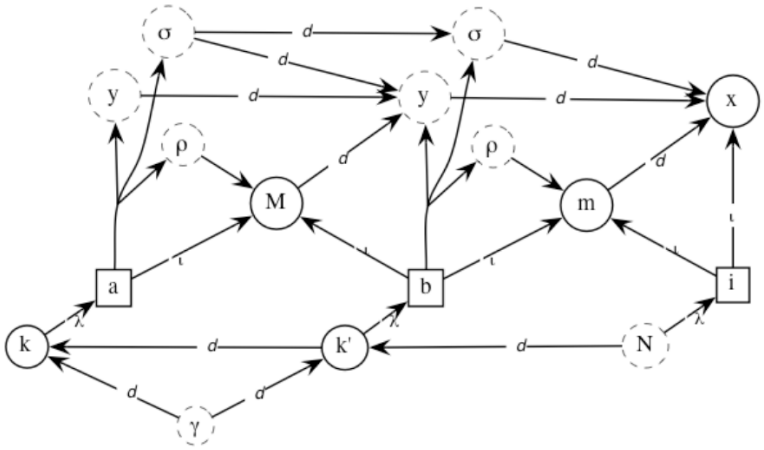


Figure 11. Two-level hierarchical mixture model.

which in a fully hierarchical version would be:

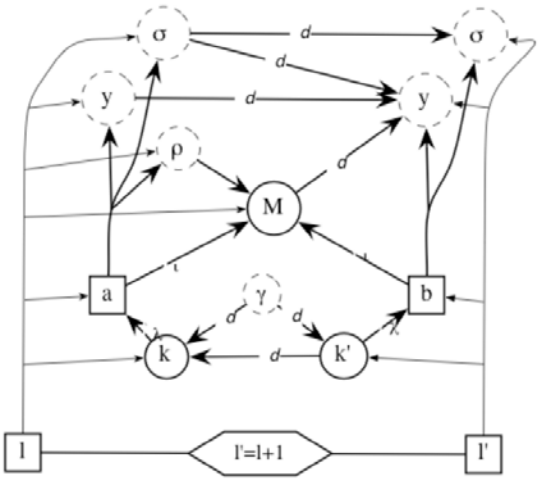


Figure 12. Arbitrary depth hierarchical mixture model.

2.2 Example: Information Retrieval

Recall the mixture of Gaussians model (above). It is closely related to mixture models for document "topic" content [Hoffman 1999; Jordan et al. 2003]. The probability decomposition proposed by [Hofmann 1999] for latent semantic indexing approach to textual information retrieval is:

$$\Pr(\theta, w) = \Pr(\theta)\Pr(w \mid \theta) = \Pr(\theta)\sum_z \Pr(w \mid z)\Pr(z \mid \theta) \tag{6}$$

A elaborated by [Blei et al. 2003] this may be diagrammed as follows:

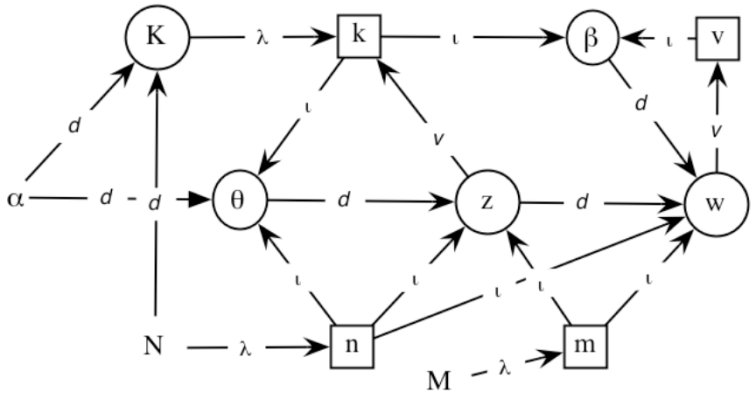


Figure 13. Document topic model.

In this diagram the notation is as follows. The index nodes are: $n \in \{1 \ldots N\}$ indexes the documents; $m \in \{1 \ldots M\}$ indexes the word positions in a document (padded out to maximal document length, or else subscripted as M_n); $k \in \{1 \ldots K\}$ indexes the topics a word or document can be "about"; $v \in \{1 \ldots V\}$ indexes the vocabulary of possible words.

The random variable nodes are: w_{nmv}=0 or 1 depending on whether or not word position m in document n is occupied by vocabulary word ω; z_{nmk} = 0 or 1 depending on whether word position m in document n is about topic k; θ_{nk} = the prior probability (mixture parameter) of topic k in document n, for all word positions.

The constant parameter nodes are: α = a parameter for the probability distribution on probability distributions such that $\Sigma_k \theta_{nk} = 1$ (e.g. the Dirichlet distribution); N and M, the number of documents and words in a document respectively.

Note similarities and differences compared to the clustering network. The inexact mapping between the two model graphs is: ρ → θ, M → z, x → w, i→m, N→N, k→K, a→k . Differences include the ω, n indices.

This is a very simple mixture model. As in clustering, one can view the number of words drawn from a particular topic as a multinomial distribution, or one can view the topic (z_{nm*}) of each word m in each document n as drawn independently from the same distribution (θ_*).

Natural elaborations for this model include the introduction of a hierarchical structure wherever it would make sense: on the topic index *k*, for hierarchical topic clustering $k \rightarrow (k_1, k_2 \ldots)$, and on the document index, n, for document segmentation into chapters, sections, paragraphs, and other parts. $n \rightarrow (n_1, n_2, \ldots m_1, m_2, \ldots)$. [Blei et al. 2003] proposed a hierarchical topic model using the "Chinese restaurant process". Note that (n, m) is already a hierarchical index on word positions in a corpus, so the new index would end in m as well. [Hearst 1994] introduced information retrieval models for topic boundaries within a document.

There are a variety of ways to formulate such models. Figure 14 provides a representation of the topic hierarchy model of [Blei et al 2003]. Figure 15 proposes a novel generalization to include both topic and document hierarchies.

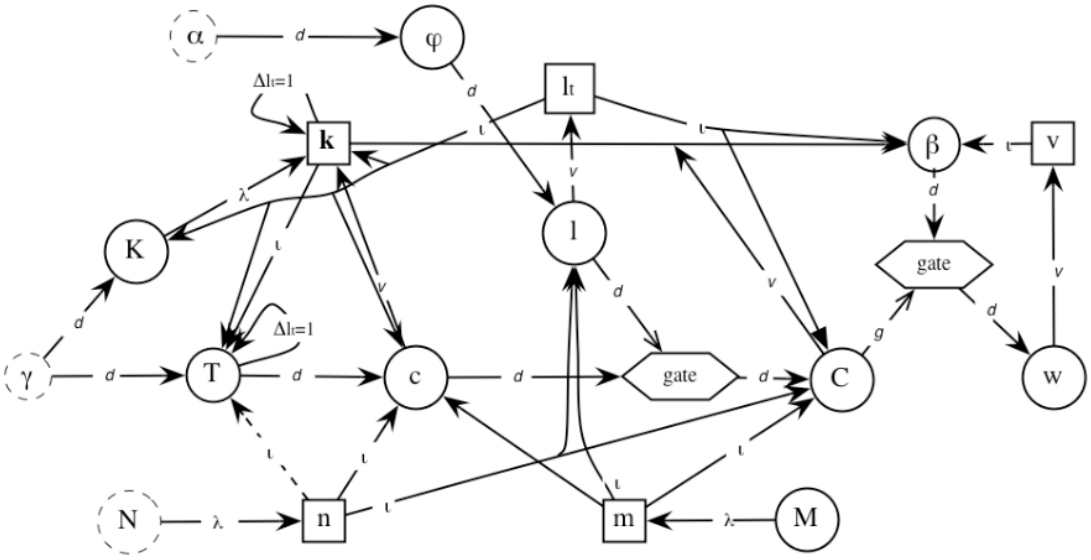


Figure 14. Hierarchical topic model from [Blei et al. 2003] model, redrawn and relabeled. The topic index **k** is hierarchical and indexed by a topic level number l_t. *T* is a hierarchical mixing proportion vector.

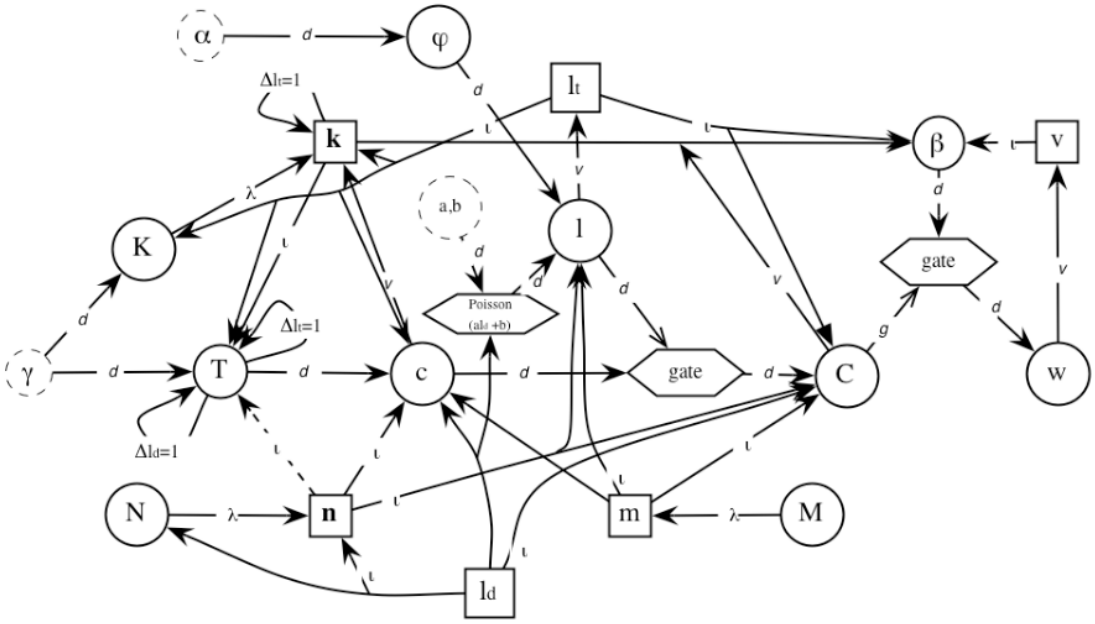


Figure 15. A generalization to express both topic and document hierarchies, as trees.

2.3 Example: Bioinformatics

Another example that shows how to reexpress structured probabilistic models in a scientific domain is a model relating gene sequence and mRNA expression data by way of regulator binding site and module membership variables [E. Segal et al. 2003]. It is diagrammed in Figure 16. The "logit" expression has a nontrivial algebraic structure. *S* is the sequence of nucleotides upstream of or otherwise capable of regulating expression in a particular gene *g*. *R* denotes the possession of regulatory motif *r* by gene *g*. *M* denotes the membership of gene *g* in module *m*. *E* denotes the expression value of mRNA coded for by gene *g* in experiment *j*.

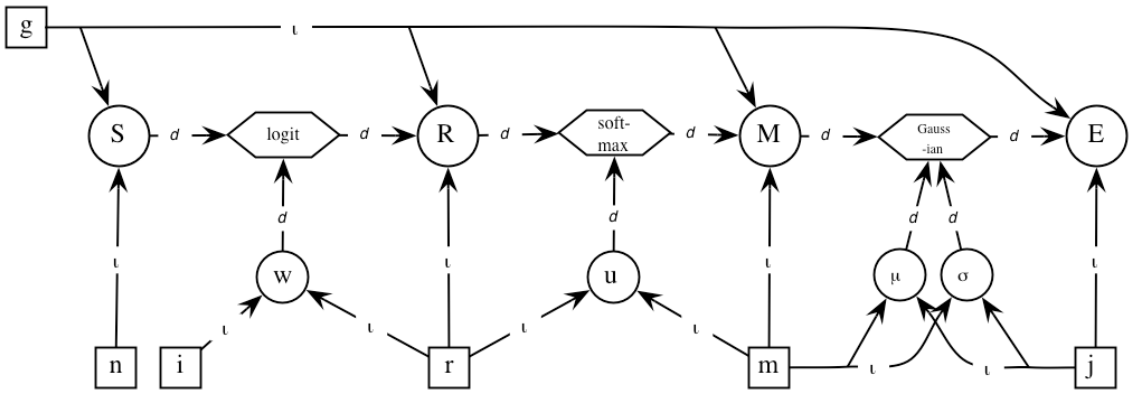


Figure 16. Model for regulation of mRNA expression level as a function of sequence information [Segal et al. 2003].

3. Variable-Structure Systems.

3.1. Variable number of nodes.
One approach to variable structure systems is to allow subscript limits to become random variables. For example the number of clusters, *k*, can become a variable in the mixture model described above:

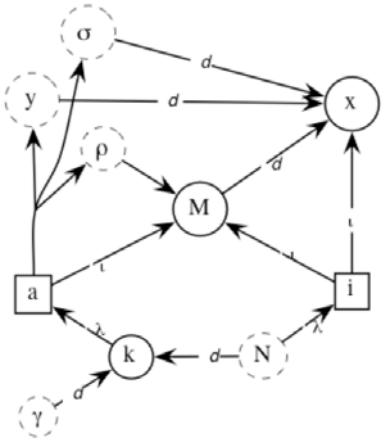


Figure 8. Mixture model with variable number of classes.

This is useful for example for expressing clustering models with an unknown or variable number *k* of cluster centers, and an unknown discrete probability distribution ρ on those *k* cluster centers. The prior on *k* and ρ can be given for example by "power law prior" such as a finite sampling of infinitely many clusters whose population frequencies are given by a power law with exponent γ. Alternatively it could be given by a Dirichlet distribution with its own exponents.

Here is a stochastic parameterized grammar associated with the above model:

grammar MixGaussian (dataset(N, γ) → {datapoint(**x**) | $i \in \{1..N\}$ }) {
dataset(N, γ) → clusterset(k, ρ, N)
with (ρ, k) ~ PowerLawPrior(N, γ) // or a Dirichlet distribution
clusterset(k, ρ, N) → {cluster(a, $\mathbf{y}_a$, σ_a, n_a)}
with $\mathbf{y}_a$ ~ Gaussian(0,1)
Pr(σ) = $\Gamma(\sigma|\alpha,\beta)$ // Gamma or Wishart distribution
n_a ~ Multinomial(N, k, ρ_a)
where σ = diag(σ)
cluster(a, $\mathbf{y}_a$, σ_a, n_a) → {datum(x), $b \in \{1..\ n_a\}$ }, filledclass(a, n_a)
with x_{ab} ~ Gaussian(y, σ)
{datum($\mathbf{x}_{ab}$) | a in 1 .. k and b in 1.. n_a }, {filledclass(a, n_a) | a in $\{1..k\}$}
→ {datapoint($\mathbf{x}_i$) | $i \in \{1..N\}$}
with {M} ~ uniform permutations M such that $\sum_{ib} M_{i,ab} = n_a$
subject to $\mathbf{x}_i = \sum_{a,b} M_{i,ab}\ \mathbf{x}_{ab}$
}

The SPG is a useful tool for describing DPIN's in which the structure of the conditional independence network, including the existence of particular variables or dependency relationships, depends on the values of other random variables. In this case, the number of classes (and hence *a* or *M* variables) is determined by random variable *k* (or *A*), and the dependency between y_a and x_i is present or not dependent on the value of a_i or M_{ai}. This kind of dynamic structure is not well captured by a fixed-structure Bayes Network; it requires large increases in dependency fanin that are problematic for general-purpose inference algorithms.

3.2 Variable Link Structure

We have already seen how to make the number and nature of the objects in a data model depend on other random variables and therefore vary from one sample to the next. Now we generalize to variable link or relationship structure, arriving at variable-structure systems such as those required in biological development [Mjolsness, Sharp and Reinitz 1991].

Recall the constraint node representation for factoring conditional or joint densities from section 1.4. As an example involving directed edges, in equation (1) for a mixture of Gaussians, the separate factors

$$\phi(\mathbf{x}_i \mid \mathbf{y}_a, \sigma_a, M_{ia}) = G(\mathbf{x}_i \mid \mathbf{y}_a, \sigma_a)^{M_{ia}} \qquad (7)$$

would each correspond to an interaction node which "gates" the influence of **y** and σ on **x** under the control of the matrix random variable *M* (here, used in the indicator variable form $M \in \{0,1\}$). *M* itself is regulated by the constraint that for each data point *i*, its sum over all classes *a* is one. Both constraints can be diagrammed in a useful template or idiom for gated interactions:

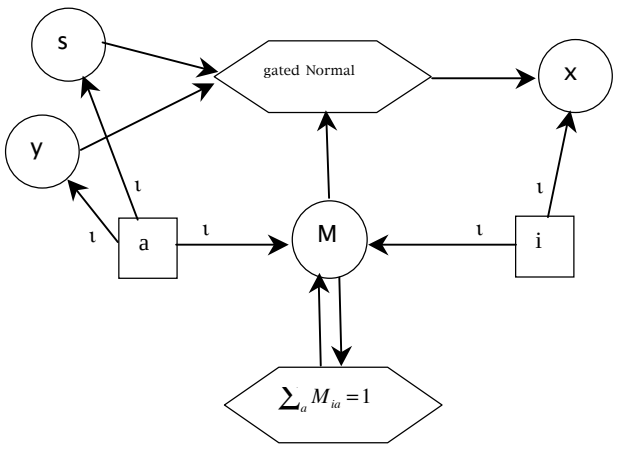


Figure 17. Constraint node for the cluster membership variable *M*.

Using interaction nodes one may naturally diagram sparse random adjacency matrices (data graph or relationship structures). In Figure 17 above, simply replace the cluster membership node M with a general adjacency matrix node $G \in \{0,1\}$, and replace the constraint node $\sum_a M_{ia} = 1$ with a different interaction such as a Boltzmann distribution with energy function on G's total number of edges, its indegree distribution, and its outdegree distribution:

$$E_{\text{degree}}(G) = \sum_{ia} \mu_{ia} G_{ia} + \sum_i f^{\text{in}}\left(\sum_a G_{ia}\right) + \sum_a f^{\text{out}}\left(\sum_i G_{ia}\right) . \qquad (8)$$

A Boltzmann distribution on adjacency matrices with this form of energy function can serve as a prior probability distribution on sparse graphs.

3.3 Observations on the graph prior

By comparison with [Morris et al. 2003], equation (8) is a factorable and solvable statistical mechanical system when there is either an indegree or an outdegee term alone. Morris et al. use just one degree prior for a symmetric (undirected) graph, but incorporate a posterior that depends on multiple graphs each of which has a degree prior. Similar "exponential" probability distributions on graphs are discussed in the review by [Newman 2003]. To understand the Boltzmann distribution in the more complex situations of both indegree and outdegree terms constraining a directed graph, or a degree term constraining an undirected graph, we can calculate the volume (= exponential of the entropy) of a constant-energy slice of adjacency matrix space. We do this by calculating the number C of graph adjacency matrices corresponding to a given degree distribution $(m_1, \ldots m_N)$ where m_i is the integer degree of the i'th node, and where self-connections are allowed. The number $C(m_1, \ldots m_N)$ is given by the generating function:

$$\begin{aligned} J_N(\mathbf{z}) &= \sum_{\{m\}} C(m_1, \ldots m_N) \prod_{i=1}^{N} z_i^{m_i} \\ &= \sum_{\left\{\begin{array}{l} G_{ij} \mid \\ G_{ij} \in \{0,1\} \\ \wedge \quad i \le j \end{array}\right\}} \prod_{i=1}^{N} z_i^{G_{ii}} \prod_{\substack{i,j=1 \\ i<j}}^{N} \left(z_i z_j\right)^{G_{ij}} \\ &= \prod_{i=1}^{N} (1 + z_i) \prod_{\substack{i,j=1 \\ i<j}}^{N} \left(1 + z_i z_j\right) \end{aligned} \qquad (9)$$

for undirected graphs. For directed graphs there are both outdegrees m and indegrees n, and the number $C(m_1, \ldots m_N, n_1, \ldots n_N)$ of graph adjacency matrices compatible with all these degree constraints is given by

$$K_N(\mathbf{x},\mathbf{y}) = \sum_{\{m\}} C(m_1,..m_N,n_1,...n_N)\prod_{i=1}^{N} x_i^{m_i} \prod_{j=1}^{N} y_j^{n_j}$$
$$= \sum_{\left\{\begin{matrix} G_{ij} | \\ G_{ij}\in\{0,1\}\end{matrix}\right\}} \prod_{i,j=1}^{N} \left(x_i y_j\right)^{G_{ij}}$$
$$= \prod_{i,j=1}^{N} \left(1 + x_i y_j\right) \quad . \qquad (10)$$

Other additive energy function terms can express more detailed prior biases on the graph structure of *G*. For example, one may count and reward or penalize graph "motifs" such as loops in the graph G by adding the penalty term $\mathrm{tr}[G^T/(1 - \varepsilon G)]$, which weights longer loops with higher powers of ε. Related ideas are reviewed by [Newman 2003] as "exponential random graphs" along with preferential attachment and vertex duplication distributions. [Ziv et al. 2003] describe a graph feature vocabulary built from the adjacency matrix, used to discriminate graphs from different distributions. As an example of the expressive power of features built from functions of the adjacency matrix, the prior (a) below expresses prevalence of triangles i→j→k→i, following [Ziv et al.], and the prior (b) below expresses occurrences of an arbitrary subgraph, some of whose edges (g_{ab}=1) must map directly to edges of *G* and some of whose edges ($\hat{g}_{ab}$=1) can appear indirectly as a sequence of edges of *G*:

$$\sum_{ijk} G_{ij} G_{jk} G_{ki} \qquad (a)$$

$$\sum_{\{i_1..i_n\}} \prod_{ab} \left(G_{i_a j_b}\right)^{g_{ab}} \left(\left(\frac{G}{1-\varepsilon G} \right)_{i_a j_b} \right)^{\hat{g}_{ab}} \quad . \qquad (b)$$

Expressible graph motifs include bottlenecks, feedback loops, feedforward loops, lock-and-key node connection biases, and so on. However, each motif must be present exactly to be counted and the objective functions are high order. A more flexible option would be to add an inexact graph-matching objective function [Gold et al. 1996] between *G* and *g* to the Boltzmann distribution. This objective includes new matching variables M_{ia} between nodes of *G* and of *g* and it maximizes the number of consistent "rectangles" of links in G_{ij}, g_{ab}, M_{ia} and M_{jb}. If we integrate probability over configurations of *M*, rather than maximizing probability over them, then the probability of *G* is increased by its possession of multiple possibly inexact copies of a graph motif *g*.

3.4 Gating graphs and relationships

Taking $E_{degree}(G)$ to refer to any such Boltzmann distribution on graph structures, the resulting template diagram is:

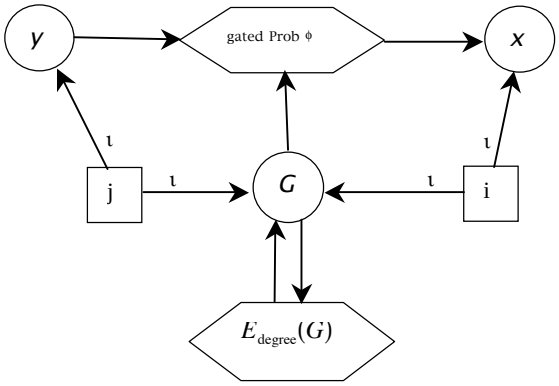


Figure 18. Constraint node for the graph sparse adjacency matrix variable *G*.

As a diagrammatic abbreviation, one can allow *G* to become another (user-defined, dynamically variable) link type and link label, like d, ι, λ, and v, in the diagrammatic notations of Figure 19a or even 19b.

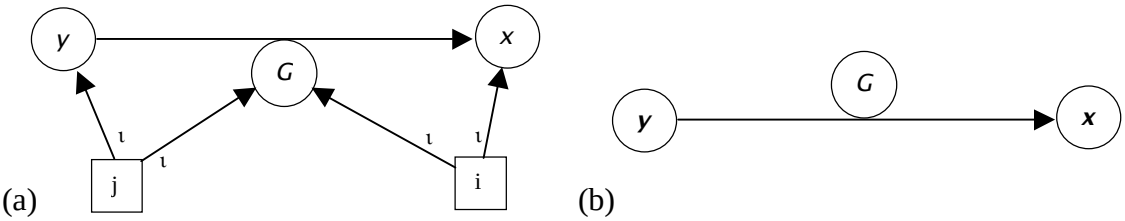


Figure 19. Abbreviated notation for gating of the interaction between x and y by G. (a) with index nodes, or (b) with vector notation for the random variables **x** and **y**.

(By convention, every component x_i of **x** depends on every component y_j of **y** unless explicitly restricted.) This pattern may be iterated to make a network of possible relationships, e.g.

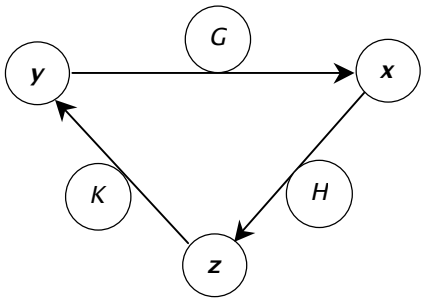


Figure 20. Network with multiple relationships each represented by a graph.

A biological example might relate *y* to states of genes, *x* to states of proteins, and *z* to states of transcription factor binding sites that act on genes, with apparent circularity to be removed by introduction of dynamics as described later.

The associated probability distribution for a *G*-mediated relationship between *x* and itself may be taken to be, following equation (4), one of the sequence of increasingly general Boltzmann distributions below:

$$E = \sum_{ij} G_{ij} V\left(x_i, x_j \mid T_{ij}\right) + E_{\text{degree}}(G)$$
$$= \sum_{ijc} G_i^c G_j^c V\left(x_i, x_j \mid \mathbf{T}^c\right) + E_{\text{degree}}(G)$$
$$E = \sum_c \sum_{\{i\}} \left(\prod_j G_j^c \right) V\left(\left\{ x_j \mid G_j^c = 1 \right\} \mid \mathbf{T}^c \right) + E_{\text{degree}}(G) \quad .$$

In this way we can introduce one or many dynamic or data-dependent relationships, such as *G* above, between a set of random variables. For example there could be one graph *A* representing adjacency and another graph *C* representing containment among volume elements in a geometric model. Note also that the conditional distribution of *G* given x is influenced by compatibility between the node variables as measured by pairwise potentials *V*.

3.5 Introduction of "Objects"

For knowledge representation it is useful to consider a special case in which the entities related by a link in *G* are groups of variables (representing "objects") rather than single variables. Simply make each variable *x* above be a vector by adding an index to the *x*'s but not the *G*'s:

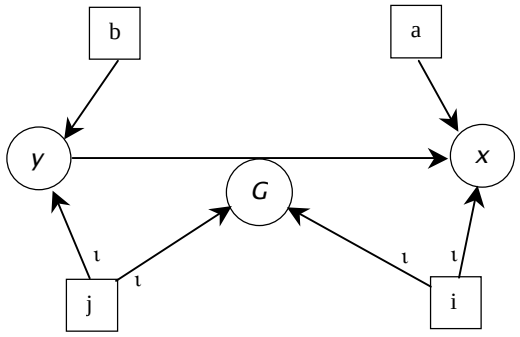


Figure 21. Introduction of objects.

This dependency graph allows all internal variables y_{jb} of object *j* to influence all internal variables x_{ia} of object *i*, provided only that the relationship G_{ij} between the two objects is nonzero. If instead there is one or more preferred wiring patterns g_{ab} among the internal variables in any two objects, then we can use a graph decomposition form such as

$$G_{(ia)\,(jb)} = \Sigma_\alpha G^\alpha_{ij}\, g^\alpha_{ab}$$

to gate the connections between y_{jb} and x_{ia}. For example, the wiring pattern index α could take values α =1 for Adjacency and α=2 for Containment in a geometric model. Each wiring pattern α could gate a separate potential function V. Note that, if wiring patterns α are as numerous as course-scale index pairs (ij), then we can recover a modular but otherwise unstructured hierarchical network $G_{(ia)\,(jb)} = G^{(coarse)}_{ij}\, g^{ij}_{ab}$.

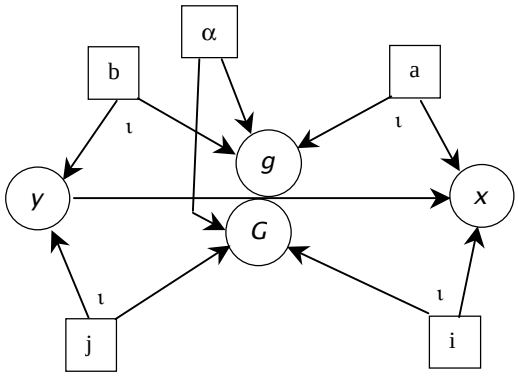
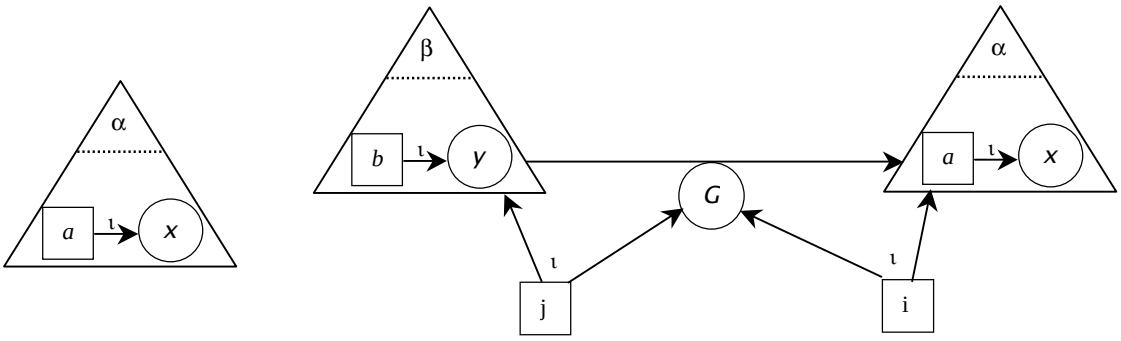


Figure 22. Multiple relationships among objects.

The variables in such an object may be grouped together diagrammatically with another shape such as a triangle, representing an object instance. Optionally objects may be assigned types α. The resulting compound object node is shown in Figure 23(a). Unlike the plate notation, overlaps and recursive substructure are not allowed within the node in such a diagram but must be represented by relationship nodes in a graph structure as shown in Figure 23(b). These triangle object nodes also have similarity to class nodes in a Universal Modeling Language (UML) diagram, though UML is a much more elaborate notation without a probabilistic interpretation.



(a) (b)

Figure 23. (a) triangle node groups essential elements of an object: its random variables x, its internal index a, and its type value α. (b) Relationship G between two sets of object nodes gates their probabilistic dependencies.

3.6 Relation to Probabilistic Relational Model (PRM)

A PRM as described in [Friedman et al. 1999] is a relational database model with an added dependency graph structure, consistent with the relational model schema. We map PRM's into the object dependency graphs introduced above as follows:

Classes or *entity types* such as X and Y are mapped into indexed sets of random variable groups such as $\{x_i\}$ and $\{y_j\}$. *Class instances* or *entities* such as $x \in X$ are mapped into particular random variable groups such as x_i and y_j. *Attributes* such as *x.a* and *y.b* are mapped into particular random variables such as x_{ia} and y_{jb}. *Binary relations* such as $\sigma(X,Y)$ are mapped into graphs $G^{\sigma}(X,Y) = \{ G^{\sigma}_{ij} \}$. These also define single *slots*.

Slot chains such as $\sigma.\tau.\omega$ are mapped into matrix multiplication of the corresponding adjacency matrices, e.g. $\Sigma_{jk} G^{\sigma}_{ij} G^{\tau}_{jk} G^{\omega}_{kl}$. Given a starting object indexed by *i*, this computes a set of possible related objects indexed by *l* via slot chain $\sigma.\tau.\omega$. All possible results of slot chains of any length (including zero) may be computed from the components of matrix $I/(I-\varepsilon G')$, as $\varepsilon \to 1$, where $G' = \Sigma_{\sigma} G^{\sigma}$. *Dependencies and Parent sets* such as Pa(x.a) are mapped into attributes reachable by *g*, for *i=j*, or by $G'/(I-\varepsilon G')$ g. There is room for debate here about the best extent of a parent set, since for some databases, $I/(I-G')$ is nearly dense. A choice of *aggregation operators* over multiple attributes reached from a given slot chain is a feature of PRM that is not modeled in detail by the diagram above.

The *schema* for a PRM is the pattern of wiring (σ, x, y) by which binary relations indexed by α relate variable classes such as *x* and *y*. It is implicit in the diagrams above, but can be made explicit with the addition of yet another index α on the random variables *x* to designate their type, and a schema wiring pattern $S^{\sigma}_{\alpha\beta}$ among types and binary relations. Then we obtain a metamodel for all PRM's:

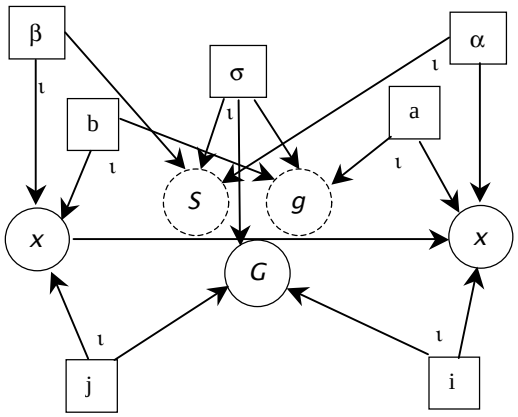


Figure 24. Probabilistic relational model.

Note that each G^{σ} is variable and data-dependent in a PRM, but g is constant and so is the schema S.

3.7 Frames and Semantic Networks

The basic object x, of type α, with attributes indexed by a, possibly related to object y, can be redrawn as a "frame instance" in a very simple semantic network by using the new object node (here denoted with a triangular shape) to group x, α, and a as in Figure 21. This diagram is consistent with the previous notation for Frameville networks [Mjolsness, Gindi and Anandan 1989; Anandan Letovsky and Mjolsness 1989; Mjolsness 1997] governed by Boltzmann energy functions, except that we have instances without the "class" information such as S and g via the potentials H. Those can be restored, as shown in Figure 25.

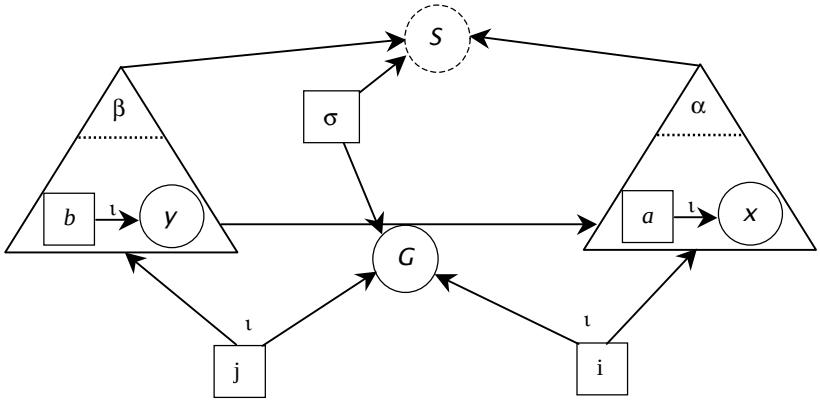


Figure 25. Frame instances with schema.

Compare this object model to the following Frameville diagram and objective function, in which (σ,τ) and (α,β) play similar roles. *INA* is *S*, *ina* is *G*, *M* is internal to the hard-typed objects. However, differences between the objects above and Frameville include: (a) the fact that in the latter system the assignment of instances i to types α is flexible, allowing for type uncertainty during inference and the use of inheritance; (b) constraints on S enforcing the interpretation of a compositional hierarchy (*INA* or part-of) and/or inheritance hierarchy (*ISA* relationship) in Frameville; (c) Frameville is usually regarded as having undirected dependencies so that inference can proceed in any direction; (d) the omitted H parameter-check term is similar to g and the omitted clique potentials, except that H is indexed by object type rather than by relation.

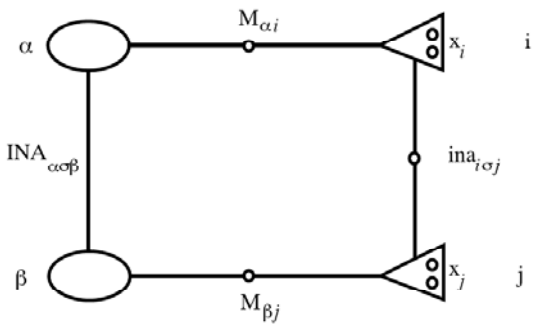


Figure 26. Frameville objective function diagram, from [Mjolsness 1997].

$$E(A,x) = H_0\left(x^0\right) + \sum_{l=1}^{L} \sum_{\alpha\sigma\beta} \sum_{ij} \mathrm{INA}_{\alpha\sigma\beta} \mathrm{ina}_{i\sigma j}^{l} M_{\alpha i}^{l-1} M_{\beta j}^{l} \left[H^{(\alpha\sigma\beta)}\left(x_j^{l}, x_i^{l-1}, u^{(\alpha\sigma\beta)} \right) - \mu^{(\sigma\beta)} \right]$$

In this way we are able to group random variables into larger units of "memory" analogous to record structures in a programming language or frame instances in a semantic network style knowledge representation scheme, and to index these frame instances in such a way that they can point to one another by way of sparse graphs. We thus arrive at a graphical notation for objects and relationships similar to "Frameville" networks.

3.8 Local node dynamics

The variable-structure relationship diagrammed above can provide the topology within which node dynamics are local, for example in a recurrent neural network or other feedback-capable model of transcriptional and other biological regulatory networks [Mjolsness Sharp and Reinitz 1991]. Other examples include modeling internal dynamics of extended objects such as storms in the atmosphere, or geological processes resulting in changes to strata, or articulated mechanical objects in robotics and vision.

One key to this application is to introduce a new index node "*t*" for time. This corresponds to loop unrolling in a conventional neural network. As before, we can also introduce internal vector indices a (not shown) for the state variables v.

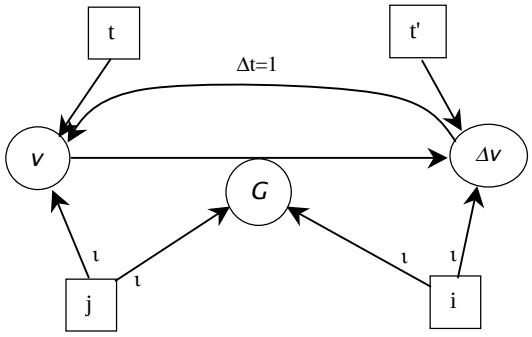


Figure 27. Objects with node dynamics.

The return arrow from Δv to v allows the change of state sampled at time t to be added into the state vector at time t+1. The resulting graph of dependencies is still a Directed Acyclic Graph.

A generic energy function for dynamics, local to the network, is then:

$$E = C_1 \sum_t \sum_{ija} G_{ij}^a \left[u_{ia}(t) - F^{\alpha(i)\beta(j)}\left(v_j(t), p_{ija}\right)\right]^2 + C_2 \sum_{t,i,a} \left(u_{ia}(t)\right)^2$$
$$+ C_3 \sum_t \sum_i \left[\left(v_i(t+\Delta t) - v_i(t)\right)/\Delta t - \sum_a g_{ia}\left(u_{ia}(t)\right)\right]^2$$

where

$$G = \sum_a G_{ij}^a \quad .$$

The complete collection of node and link types can be put into a type hierarchy. Inheritance relationships between object types can be understood as inheritance of parent type parameters, relationships (slots), and optionally also terms in the energy function.

In the applications to follow, we will adopt the frame instance notation and terminology (including inheritance relationships) for object-level probabilistic dependency diagrams.

With implicit graph priors, every network diagram corresponds to a probability distribution over networks including their dynamics. Expected temporal behaviors of high probability dynamical networks in such a distribution could be characterized in many ways, including their approximability by simpler networks including but not limited to those exhibiting attractor dynamics.

4. Application: Biological Regulatory Networks

Here we outline application of object-level probabilistic dependency graphs to biological regulatory networks.

4.1 Regulatory network dynamics

A generic dynamic regulatory model, local to graph $G = \sum_a G_{ij}^a$, can be written:

$$E = \sum_t \sum_{ija} \sum_{s\in\{-1,+1\}} G_{ij}^{sa} \left[u_{ia}^s(t) - F^{s\alpha(i)\beta(j)}\left(v_j(t), p_{ija}^s\right)\right]^2 / \sigma_{\alpha(i)\beta(j)}\left(u_{ia}^s(t)\right)^2 + C \sum_{t,i,s} \left(u_i^s(t)\right)^2$$
$$+ \sum_t \sum_i \left[\left(v_i(t+\Delta t) - v_i(t)\right)/\Delta t - \sum_a g_{ia}^+\left(u_{ia}^+(t)\right) + \sum_a v_i(t) g_{ia}^-\left(u_{ia}^-(t)\right)\right]^2 / \sigma_{\alpha(i)}\left(v_i(t)\right)^2 \quad .$$

For example, the law of mass action can be expressed with F=*log*(.) and g = *exponential*(.), and recurrent neural network style Gene Regulation Networks can be written using F = identity and g = a sigmoidal function. Other dynamics can be implemented with hidden units. The (optional) a index serves to distinguish different interaction nodes that lead from j to i, corresponding for example to different reactions with j as an input and i as an output. Interactions are also classified as synthetic or degradational, s = +1 or –1, with slightly different dynamics. The standard deviation function serve to implement a Langevin equation stochastic model, in which standard deviations are typically proportional to the square root of the number of molecules, or to model measurement error which may be for example a positive linear function of the concentration.

The form of the dynamics $F^{s\alpha\beta}$ assumed for regulatory interactions between regulators of type β and regulates of type α is diverse. It depends on modeling choices for a wide variety of biological processes including enzyme kinetics (represented with mass action or Michaelis-Menten kinetics, or much more detailed models for allosteric enzymes), transcriptional regulation, translational control, protein complex formation and degradation, active degradation of protein, and so on [Shapiro et al. 2003]. Many different regulatory models may be required in a single network. Hence, F is indexed by object types α and β.

The resulting model class shares features with more specialized ones of [Segal, Yelensky and Koller 2003], [Imoto, Goto, Miyano. 2002], and other recent attempts to create a probabilistic framework for inferring cellular regulatory networks.

4.2 Central Dogma Networks

If we use a general form for regulatory dynamics such as the above and define the object types α to be genes, proteins and the like, we can outline a more detailed Central Dogma (CD) network (Figure below). It is constructed by considering the most basic categories of reactants α (such as DNA, RNA, and protein) and the likely reaction types $\alpha\beta$ among them (cf. dynamical equation above). As more basic biological mechanisms are understood, such as regulation by small RNA's, they can be added to the diagram with consequent global effects on the model. Dotted circles are constant or observed values.

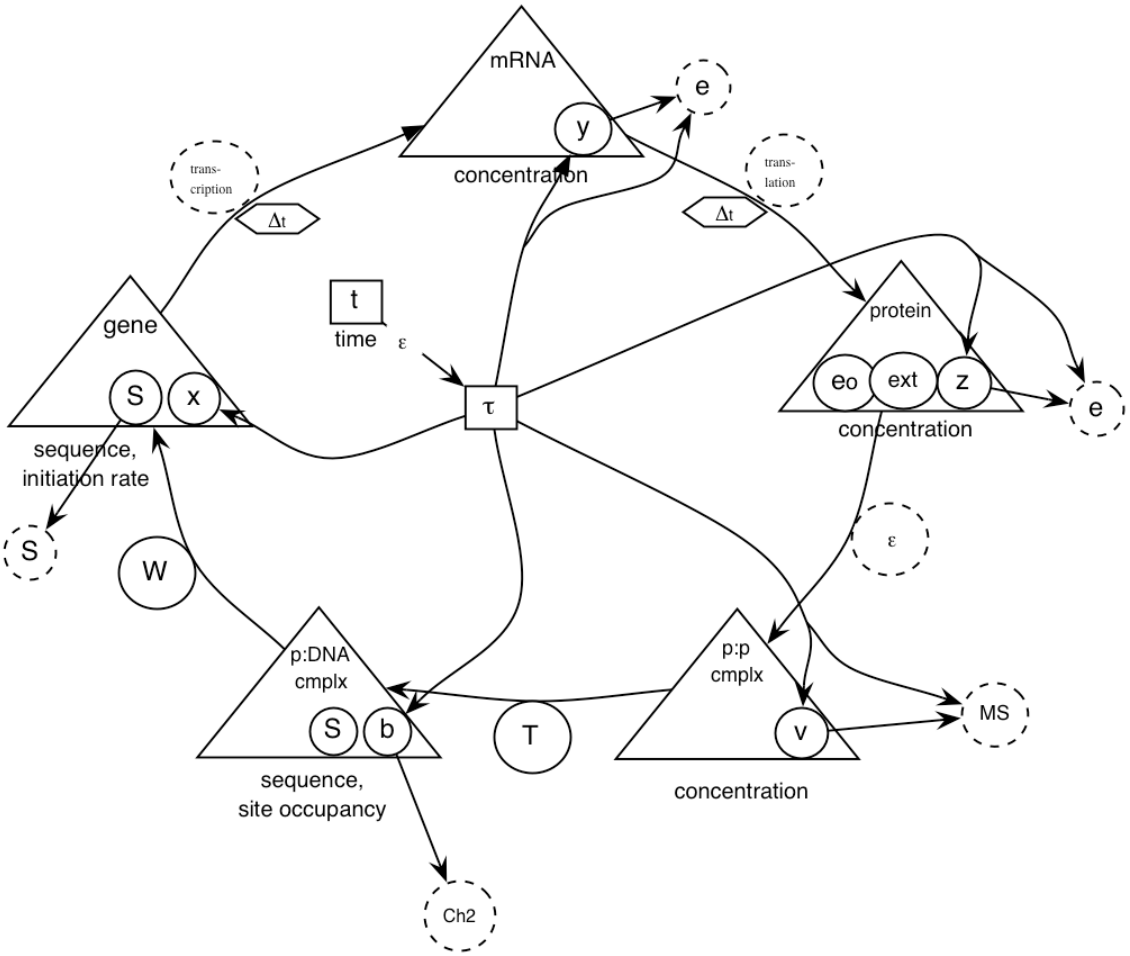


Figure 28. A Central Dogma network.

Note the multiple types of observational data that serve to constrain the model. The version of the diagram above assumes the organism in question has been sequenced. In greater detail, the network is:

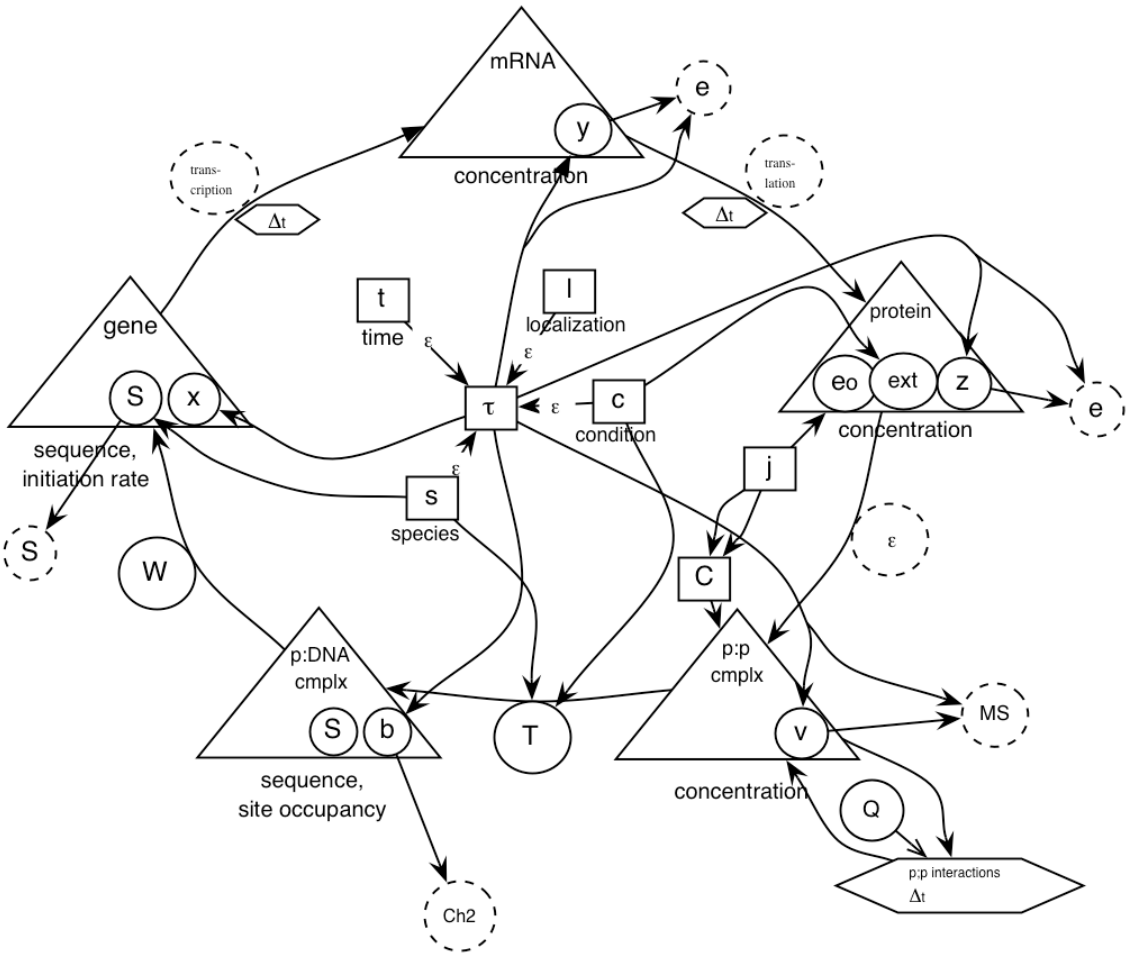


Figure 29. Central Dogma network, refined.

In this diagram, note the role of the sparse matrix *C* of adjacency relationships in a protein complex, defined in terms of the protein index *j*; the compound observation index τ which includes time point, experimental condition, organism species or taxon, and subcellular localization; and the multiple protein-complex reaction types. The protein interaction network *Q* includes subnetworks Q_1, Q_2, and Q_3 of protein-protein interactions involved in complex assembly/disassembly, protein state modification, and controlled degradation, all likely to be modeled with different dynamical models. Sequence information from genomes of multiple related species can be used to better define coding/noncoding regions in DNA [Seipel and Haussler] and protein-DNA binding sites [Moses et al. 2004, Kellis et al. 2003].

4.3 Pathways

Overlapping and/or interacting pathways are also important. Each may, in principle, involve most or all of the central dogma network types. Hence a pathway index π would be "orthogonal" to a mechanism index $\alpha\beta$. Indeed, the global reaction connectivity graph *G* can be recursively decomposed first according to CD types (giving rise to connection

matrices $G^{p:DNA,p:p}=T$, $G^{gene,p:DNA}=W$, and so on), and then according to hierarchical overlapping pathway memberships. The pathway decomposition can be expressed by indexing each CD object type α by a hierarchical version of index π (not shown above) whose deepest level is the object instance index (*i, j*, etc., also omitted above except for protein index *j*). The connection matrices *T, W*, and so on would simply have a degree prior or bias which favored intra-pathway connections without prohibiting inter-pathway connections. In this way, pathway membership is a property of the model but not of the system being modeled.

A very simple example of such a pathway model is provided by the general architecture of a "core/leaf" regulatory model, in which a fully recurrent core network connects outwards, with little if any feedback, to a set of leaf nodes each of which receives input from a small number of core nodes.

4.4 Separation of time scales

Some variables may evolve much faster than others as a consequence of parameter settings with a large dimensionless ratio. This happens for example in the Michaelis-Menten approximation to mass action kinetics. As another example, a higher order (k>2) clique interaction may influence variable *x* as a function of a fast variable *y* and a normally slow-changing variable *z*. In this case, a slow input z may effectively gate a fast input, acting to change the topology of the fast timescale network. Such phenomena may be modeled by replacing the time index *t* with two indices (t_{fast}, t_{slow}), each with the full connectivity that *t* had. A true separation of time scales may then allow the graph to be simplified by eliminating the indexing of slow variables by t_{fast} and possibly simplifying the potential functions. In this way local node dynamics (as above) can give rise to network connectivity dynamics on a slower scale.

Another variation of this technique is to replace *t* by *(c, t)* where c is a condition index that is associated not with slow time scales but with altered experimental conditions reflected in (*c* indexing of) network structure, such as (a) altered initial conditions, (b) mutant genotypes including deletions or conditional knockouts, and (c) transgenic organisms.

A very different separation of timescales is that between organismal and evolutionary time. In this case the dynamics of connections is intrinsically different from that of nodes. The phylogeny relationship ψ in the diagram above can be used to define potential functions for integrated or interacting evolutionary/regulatory networks.

5. Application: Computational Field Geology

Here we outline an application of object-level probabilistic dependency graphs to computational field geology. Prior work in object-level computational approaches to geological inference include [Simmons 1983] [Ady 1993] and [Sakamoto 1994], but they are non-probabilistic. The goal here is to create a generative statistical model of layered strata resulting from repeated geological processes such as deposition, deformation, faulting, erosion, and so on. To this end we introduce relevant object types and relationships. The types include 2D and 3D geometrical objects (e.g. surface and volume elements) and compositions thereof (e.g. surfaces and volumes). Adjacency relationships are indicated in this particular example by sharing lower-dimensional substructures. The types also include rock facies, which are groups of rocks with similar observable properties such as spectra and visual texture. Processes are modeled as discrete or continuous-time firing of rules in a grammar of geological transformations. Object types and relationships are introduced using frame "slot", "parameter", and "inheritance" notation.

layeredStructure (contains {layer}, {boundarySurface})
layer (ISA thinVolume)
(between B1:boundarySurface, B2:boundarySurface)
(matrixComposition C1:composition)
(embedComposition C2:composition)
(cutby {F_i=fault}) (boundedby {B1, B2, F_i})
boundarySurface (ISA surface) (between layer1, layer2)
composition({faciesContent(p_α , facies$_\alpha$})

thinVolume (ISA volume)
(between B1:boundarySurface, B2:boundarySurface)
// Has simplified mechanics and boundaries compared to volume.
// p_α gives mixture probabilities for layer-wide facies content
volume (contains {volement} {vertex(**x**)}) (boundedby {surface})
surface (contains {facet} {vertex(**x**)}) (boundedby {curve})
(composition {faciesContent(p_α , facies$_\alpha$})
// surface weathering effects on volume facies
volement (contains {vertex(**x**)}) (boundedby {facet})
(embeddedDist{patch(**y**, $\boldsymbol{\sigma}$)})
// Maintain convexity so volement, and its facets,
// are all determined as the convex hull of its vertices.
// e.g. two dual surface tilings, offset in z, give tetrahedral tiling.
// Patches are Gaussian blobs in an optional spatial distribution
// of embedded material.
stressedVolement($\boldsymbol{\varepsilon}$, $\hat{\boldsymbol{\sigma}}$) (ISA volement) // local strain and stress tensors $\boldsymbol{\varepsilon}$, $\hat{\boldsymbol{\sigma}}$
curve (contains {segment})
facet (contains {vertex(**x**)}) (boundedby {segment})
segment (boundedby {vertex(**x**)})

facies(α,$\boldsymbol{\mu}$, **c**, ρ(s))
// $\boldsymbol{\mu}$ is the mixture probability vector for minerals in facies α
// **c** is a vector of observables for facies α:
Concatenate spectra, rotation invariants of visual texture, etc.
Spectral components of **c** can be a complicated function of $\boldsymbol{\mu}$.
// ρ(s) is a rock size distribution

faultedLayeredStructure (ISA layeredStructure) (contains {fault})
fault (ISA surface)

Major transformations to be modeled with stochastic grammar rules include:

Deformation: layeredStructure, forceSurface ➔ layeredStructure, forceSurface
// use elastic/viscoelastic dynamics to respond to stress through deformation
Relaxation: layeredStructure ➔ layeredStructure
// use viscoelastic/plastic dynamics to relax stress over time
Faulting: layeredStructure, fault ➔ faultedLayeredStructure
// add fault, redraw volements along it, and shift all volement positions
Deposition: layeredStructure, flowField2D, flowHistory, source ➔ layeredStructure
// adds a layer on top, from surface source
Sedimentation: layeredStructure, flowHistory, source ➔ layeredStructure
// adds a layer on top, from above e.g. marine deposition or airfall
Erosion: layeredStructure, flowField2D, flowHistory ➔ layeredStructure
// removes material, forming a new top surface for exposed layers
Impact: layeredStructure, volement, impact ➔ layeredStructure
// numerous alterations including excavated layers
Regridding: layeredStructure ➔ layeredStructure
// adjusts the grid for better use of limited grid points,
leaving interpolated field unchanged

Note the use of slot chains as in PRM. Note also the need for new flowField, flowHistory and forceSurface objects (below) and grammar rules to create them.

forceSurface (ISA surface) (contains {forceFacet(**f**) })
forceFacet(**f**) (ISA facet) (contains {vertex})
Hence:
forceSurface ({F'_l}, $Eq'_{lm,np}$)}, {F'_l =forceFacet($\mathbf{f}_l$,{X'_{lm}})},{X'_{lm} =vertex($\mathbf{x'}_{lm}$)}
// $\mathbf{f}_l$ = force (vector) per unit area of facet l.

The objects and relationships listed above can be interpreted in two closely related ways: as objects and relationships in an object-level probabilistic independence network, as defined above, and as collections of linked terms (corresponding to the object types above) that may appear on either side of a rule in a parameterized graph grammar. The rules in a grammar serve to define dependency relationships, with RHS random variables depending on LHS random variables with an earlier discrete "time" index.

Figure 30 shows a portion of the above network of objects and relationships in the present diagrammatic notation.

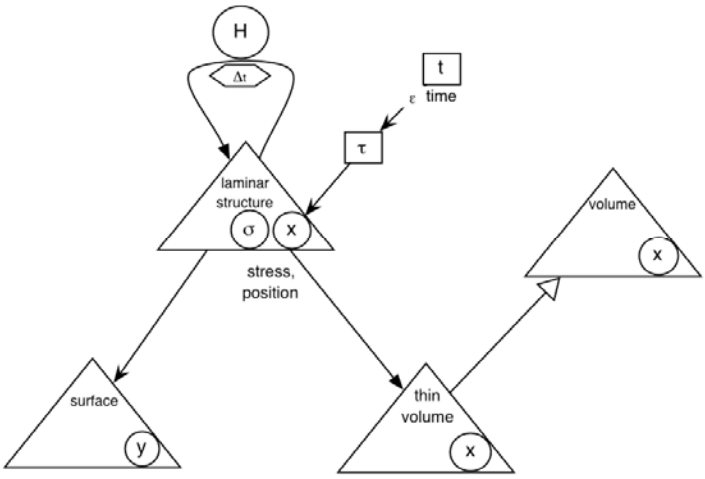


Figure 30. Geological objects.

5.1 Deformation

A deformation rule can take the form

layeredStructure($\{L_l\}$), $\{L_l$ = layer($\{V_i\}$)$\}$,$\{V_i$ =volement($\{X_k|V_{ik}=1\}$)$\}$,
$\{X_k$=vertex($\mathbf{x}_k$)$\}$, forceSurface($\{F'_m\}$)$\}$,
$\{F'_m$ =forceFacet($\mathbf{f}_m$, $\{X_k|S_{mk}=1\}$)$\}$
➔ layeredStructure($\{L_i\}$), $\{L_l$ = layer($\{V_i\}$)$\}$,$\{V_i$ =volement($\{X_k|V_{ik}=1\}$)$\}$,
$\{X_k$=vertex($\mathbf{x'}_k$)$\}$, forceSurface($\{F'_m\}$)$\}$,
$\{F'_m$ =forceFacet($\mathbf{f}_m$, $\{X_k|S_{mk}=1\}$)$\}$
where $Eq_{ik,i'k'} = V_{ik}\, V_{i'k'}\, \delta_{kk'}$ // alternative numberings name the same vertex
under $E(\mathbf{x}') = \Sigma_{ik}\, Vol_i\, E_{vol}(\{\mathbf{u}_k = \mathbf{x}'_k - \mathbf{x}_k\},\{V_{ik}\})$
$+ \Sigma_{mk}\, Area_m\, E_{surf}(\{\mathbf{u}_k = \mathbf{x}'_k - \mathbf{x}_k\},\{ S_{mk} \})$
// **x'** chosen under elastic energy function.
// Then x' becomes the new x, i.e. material is viscoelastically relaxed
// unless residual stress $\boldsymbol{\sigma}$ is stored e.g. with stressedVolements.

For relatively small displacements **u,** the energy function E_{vol} is a quadratic function of the strain $\boldsymbol{\varepsilon}_{ab} = \mathbf{u}_{(ab)} = (\mathbf{u}_{a,b} + \mathbf{u}_{b,a})/2$ [Landau and Lifshitz] incorporating material inhomogeneities, anisotropies, and a global Euclidean coordinate system invariance. Here ",b" subscript is the spatial derivative operator defined on an interpolated strain function of position. Stresses $\hat{\boldsymbol{\sigma}}$ are forces per unit area, derived as derivatives of E with respect to strain. For large displacements, more sophisticated invariant energies must be used [Terzopoulos et al. 1987, Grinspun et al. 2003].

A key idea for this application is that one or more boundary surfaces of the layeredStructure, on the bottom and/or sides, are chosen and subjected to constant isostatic pressure (stress in the exterior volume due to pressure and gravity alone, as if the exterior volumes were fluid enclosed in a balloon). If necessary, first augment the layeredStructure with a deep bottom layer, then cut the layeredStructure by the desired force-application surface "forceSurface" to make it a boundary. During relaxation, displace this surface along with the layeredStructure volume but keep its forces compatible with those due to an exterior fluid volume, e.g. constant or isostatic. An extra grammar rule will be required to create the desired forceSurface. Major special cases that can be treated this way include uplift (centered at a point or on a line), slumping (likewise), lateral compression, and lateral extension.

To minimize E, one may use viscoelastic stress tensor dynamics. To sample from it will require other techniques.

A similar discussion applies to the "relaxation" rule which however will be augmented with velocity-dependent viscoelastic stress tensor terms resulting in dynamics for relaxation of stress. Deformation, relaxation, sedimentation and erosion together allow one to define a nontrivial scenario and test an inference algorithm.

5.2 Sedimentation

A sedimentation grammar rule can take the form:

$$\text{layeredStructure}(\{L_l \mid l \in \{1 .. l_{max}\}\}),\ \text{flowHistory}(\Delta z),$$
$$\text{source}(\text{matrixComposition} \rightarrow \text{MC}, \text{embedComposition} \rightarrow \text{EC})$$
$$\rightarrow \text{layeredStructure}(\{L_l \mid l \in \{0 .. l_{max}\}\}),$$
$$L_0 = \text{layer}(\text{matrixComposition} \rightarrow \text{MC}, \text{embedComposition} \rightarrow \text{EC}, \{V_i\}),$$
$$\{V_i = \text{volement}(\{X_k\}, \text{embeddedDist}(\{\text{patch}(\mathbf{y}, \boldsymbol{\sigma})\}))\ \}$$

In greater detail: construct a new thin volume L_0 on top of a triangulated surface S by constructing the dual tessellation, raising its height by $\Delta z/2$, and adding another copy of triangulated surface S, raised everywhere by Δz. Connect vertices locally for a volumetric triangulation of a new layer of thickness Δz. Supply a patchy distribution of embedded material (a superposition of spatial Gaussians) and restrict it to each volume element in turn.

6. Discussion

A more general and convenient graphical notation for generative models has been introduced. Its expressiveness is supported by examples described in previous work in biology, geology, hierarchical clustering models, and variable-structure systems. It

enables us to look forward to greatly simplified invention of new algorithms involving hierarchical models and variable-structure systems.

Such graph representations of generative models, and implicitly, of statistical inference algorithm families for their inversion, have several potential advantages. They may serve as a substrate for automated creation of inference algorithms. This can happen by converting equation-labeled graph structures into mathematical pseudocode and compiling that into conventional programming languages. Or, graphs can be manipulated automatically by graph composition and editing operations. Indeed, graphs can be generated by statistical models in the form of graph grammars. For example, hierarchical graphs of varying degrees of regularity can be generated by the grammar and recursion relations below:

grammar graph-recursion (start → {node(**i**), G-connection(a, **i**, **j**)}) {

start → node'((0)), G-connection(1, (0), (0))

N=node'(**i**) → N=node(**i**), { node'((**i**, i_n)) | A((**i**, i_n))=1 ^ $i_n < i_{max}$ }
under $E = \mu \Sigma_{(i, in)} A((\mathbf{i}, i_n))$

G-connection(*a*, **i**, **j**), N=node((**i**, i_n)), M=node((**j**, j_n))
→ { G-connection(*b*, (**i**, i_n), (**j**, j_n)) | $G^{ab}_{i_n j_n}=1$ }, N, M

}

In this grammar, boldface indices **i** etc. correspond to index tuples (i_1, i_2, … i_{n-1}) and (**i,** i_n) denotes the tuple extension to (i_1, i_2, … i_n).

In this way, $G^{ab}_{i_n j_n}$ acts as a reusable wiring pattern or *cable* or, when **i** = **j**, as a reusable subgraph.

The corresponding graph recursion relationship is [Mjolsness Sharp and Alpert 1989]

$$G^{a_0}_{(i_1 \ldots i_L)(j_1 \ldots j_L)} = \sum_{\{a_l\}} \prod_{l=1}^{L} \left[G^{a_{l-1} a_l}_{i_l j_l} \right]^{A_{(i_1 \ldots i_L)} A_{(j_1 \ldots j_L)}}$$

This form generalizes all previous ones mentioned above.

Acknowledgements. The author benefited from discussions with Michael Turmon and Max Welling, as well as Barbara Wold, Christopher Hart, Padhraic Smyth, Ashley Davies, Kenneth Hurst, and the students of ICS 280F Winter 2004. The work was supported by the Physical Sciences Division of the NASA's Office of Biological and Physical Research (OBPR), by the NASA Intelligent Systems program, by the National Institute for General Medical Sciences, BISTI program, and by the School of Information

and Computer Science at the University of California, Irvine.

References

B. E. Ady, "Towards a Theory of Spatio-Chronological Relations for Geoscience", Ontario Geological Survey Open File Report 5854, 1993.

P. Anandan, Stanley Letovsky, and Eric Mjolsness. "Connectionist Variable-Binding by Optimization", August 1989 Cognitive Science conference proceedings.

C. M. Bishop, *Neural Networks for Pattern Recognition*. New York: Oxford, 1995.

David M. Blei, Andrew Y. Ng, Michael I. Jordan, "Latent Dirichlet Allocation". *Journal of Machine Learning Research* 3 (2003) 993-1022 .

David M. Blei. Thomas L. Griffiths, Michael I. Jordan Joshua B. Tenenbaum, Hierarchical Topic Models and the Nested Chinese Restaurant Process. *Advances in Neural Information Processing* 2003.

W. L. Buntine. Operations for Learning with Graphical Models, Journal of Artificial Intelligence Research 1994.

Brendan Frey, "Extending Factor Graphs so as to Unify Directed and Undirected Graphical Models", *Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence*, Morgan Kaufmann, 2003.

Nir Friedman, Lise Getoor, Daphne Koller, Avi Pfeffer, Learning Probabilistic Relational Models. IJCAI, 1999.

Marti A. Hearst. "Context and structure in automated full-text information access". *University of California at Berkeley dissertation. Computer Science Division Technical Report*, 1994.

G. E. Hinton and T. J. Sejnowski. Optimal Perceptual Inference. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 448–453. Washington, DC. 1983.

Thomas Hofmann, "Probabilistic Latent Semantic Analysis". *Proc. of Uncertainty in Artificial Intelligence, UAI* 1999.

Thomas Hofmann. Probabilistic latent semantic indexing. *Proceedings of the Twenty-Second Annual International SIGIR Conference on Research and Development in Information Retrieval*, 1999.

Stuart Geman and Donald Geman, "Stochastic relaxation, Gibbs distribution and the Bayesian restoration of images". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721—741, 1984.

L. Getoor, N. Friedman, D. Koller, and A. Pfeffer. Learning Probabilistic Relational Models, in *Relational Data Mining*, S. Dzeroski and N. Lavrac, Eds., Springer, 2001.

Steven Gold, Anand Rangarajan, and Eric Mjolsness. "Learning with Preknowledge: Clustering with Point and Graph Matching Distance Measures", Advances in Neural Information Processing Systems 7, editors Tesauro, Touretzky, Leen, MIT Press, 1995.

E. Grinspun, A. Hirani, M. Desbrun, P. Schroder, "Discrete Shells", Eurographics/ SIGGRAPH Symposium on Computer Animation, 2003.

S. Imoto, T. Goto, S. Miyano. Estimation of genetic networks and functional structures between genes by using Bayesian networks and nonparametric regression. *Pacific Symposium on Biocomputing*, 7: 175–186, 2002. http://psb.stanford.edu/psb-online/

M. I. Jordan. Graphical models. In press: *Statistical Science (Special Issue on Bayesian Statistics),* 2003. http://www.cs.berkeley.edu/~jordan/publications.html, 2003.

M. Kellis, N. Patterson, M. Endrizzi, Birren B, E. S. Lander. Sequencing and comparison of yeast species to identify genes and regulatory elements. Nature. 2003 May 15;423(6937):241-54 .

R. Kinderman and J.L. Snell, *Markov Random Fields and Their Applications*. Providence, RI: Amer. Math. Soc., 1980.

F. R. Kschischang, B. J. Frey, and H.-A. Loelinger, "Factor graphs and the sum-product algorithm". *IEEE Transactions on Information Theory, Special Issue on Codes on Graphs and Iterative Algorithms*, 47(2):498-519.

L. D. Landau and E. M. Lifshitz, *Theory of Elasticity*, 3rd edition. Butterworth-Heinemann 1986.

Eric Mjolsness, David H. Sharp, and Bradley K. Alpert, "Scaling, Machine Learning, and Genetic Neural Nets". Advances in Applied Mathematics, June 1989.

Eric Mjolsness, Gene Gindi, and P. Anandan. "Optimization in Model Matching and Perceptual Organization", *Neural Computation*, vol 1 no 2, Summer 1989.

Eric Mjolsness, David H. Sharp, and John Reinitz, "A Connectionist Model of Development", Journal of Theoretical Biology, vol 152 no 4, pp 429-454, 1991.

Eric Mjolsness, "Symbolic Neural Networks Derived from Stochastic Grammar Domain Models", in Connectionist Symbolic Integration, eds. R. Sun and F. Alexandre, Lawrence Erlbaum Associates, 1997.

Q. Morris, B. Frey, C. Paige, "Denoising and untangling graphs using degree priors", Advances in Neural Information Processing Systems 2003.

A.M. Moses, D.Y. Chiang, and M.B. Eisen, "Phylogenetic Motif Detection by Expectation-Maximization on Evolutionary Mixtures". Pacific Symposium on Biocomputing 9:324-335 2004.

M. E. J. Newman "The structure and function of complex networks". SIAM Review 45, 167-256 (2003).

Judea Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers, 1988.

M. Sakamoto, "Mathematical Formulations of Geologic Mapping Process – Algorithms for an Automatic System". Journal of Geosciences, Osaka City University, vol 37, p. 243-292, March 1994.

Segal E, Shapira M, Regev A, Pe'er D, Botstein D, Koller D, Friedman N. Module networks: Identifying regulatory modules and their condition-specific regulators from gene expression data. Nat Genet. 2003 Jun; 34(2):166-76.

E. Segal, R. Yelensky and D. Koller. Genome-wide discovery of transcriptional modules from DNA sequence and gene expression.. *Bioinformatics*, Vol. 19 Suppl. 1, pages i273–i282 , 2003.

E. Segal, J. Stuart, D. Koller, and S. Kim, A Gene Co-Expression Network for Global Discovery of Conserved Genetics Modules,. Science, 2003 October, 302(5643): 249-55.

Bruce E. Shapiro, Andre Levchenko, Elliot M. Meyerowitz, Barbara J. Wold , and Eric D. Mjolsness Cellerator: extending a computer algebra system to include biochemical arrows for signal transduction simulations. Bioinformatics vol. 19 no. 5, pages 677–678, 2003.

Simmons, Reid G., *Representing and Reasoning about Change in Geological Interpretation*. Masters thesis, Massachusetts Institute of Technology, December 1983.

Siepel, A. and Haussler, D., Combining Phylogenetic and Hidden Markov Models in Biosequence Analysis. (2003). In *Proceedings of the 7th Annual International Conference on Research in Computational Molecular Biology* (RECOMB 2003), pp. 277-286.

Padhraic Smyth, David Heckerman, and Michael I. Jordan, "Probabilistic Independeince Networks for Hidden Markov Probability Models", Neural Computation vol. 9 no. 2, 1997.

D. Terzopoulos, J. Platt, A. Barr, and K. Fleisher, "Elastically deformable models". In Proceedings of SIGGRAPH, pp. 205-214, 1987.

Etay Ziv, Robin Koytcheff, Chris Wiggins "Novel systematic discovery of statistically significant network features" Preprint, Arxiv Condensed Matter, abstract cond-mat/0306610.