Variable-Structure Systems from Graphs and Grammars

Eric Mjolsness
Department of Computer Science
University of California, Irvine
emj@uci.edu

UCI ICS TR# 05-09
(c) July 2005; Revised October 2005

Abstract

Variable-structure systems are those for which the available objects and their relationships can enter or leave the problem as a function of time or depending on the values of other variables. We seek to define probabilistic generative models for variable-structure systems in sufficient generality for use in complex scientific and pattern recognition applications. We define a semantic map, Ψ, by which model specifications map to model semantics for variable-structure systems. The specification takes the form of (a) a labelled graph or "dependency diagram" (some of whose nodes are labelled with random variables), or (b) a context-sensive stochastic parameterized grammar (SPG), a particular kind of "dynamical grammar". The semantics takes the form of a joint probability density function (pdf), in the case of a graph, or an infinite-dimensional time evolution operator on joint probability density functions for a dynamical grammar. We illustrate these frameworks by treating, with dependency diagrams and dynamical grammars, an elementary example: context-free but resource-bounded trees of conditionally dependent feature vectors. This model can serve as a scaffold for many other variable-structure systems.

Fixed-structure Dependency Diagram (DD) classes include graphical models such as Markov Random Fields, Bayes Nets, and Factor Graphs, as well as Constraint Networks. By adding new kinds of node and link labels to the graph specification we can extend such DD classes to efficiently represent variable-structure systems and other highly structured probabilistic architectures, with a formal semantics. The DD notation extensions proposed here include: interaction-gating links (for conditional links and nodes); index nodes and links (with and without weight sharing for structured architectures) that generalize "Plates" with multiple levels of indexing and controllable scope; undirected constraint links; and derived types such as node existence and time delay links. Diagrams for the feature tree model illustrate the differences in principle between dependency diagrams for fixed and variable-structure systems.

For SPG's we relate the probability distributions on continuous-time and discrete-time execution results for such grammars, and show how they express resource-limited feature trees. We show how perturbation theory gives rise to an approach both to analysing and to simulating continuous-time executions of dynamical grammars. Finally we discuss the relationships between DD and SPG frameworks, including their synergistic use in multi-scale modeling.

1 Introduction

Probabilistic models of complex dynamical systems are the basis for much progress in machine learning, pattern recognition, decision theory, and scientific inference. In all of these fields, there is a basic distinction between problems with static structure and variable structure. From the graph theory point of view, the distinction corresponds to that between dynamics of variables related by a fixed graph, and dynamics that governs a varying graph connectivity structure as well as the values of variables defined on its nodes. Examples in scientific modeling include the production of particular molecules as a result of chemical reactions, and their dynamic association into loosely bound multimolecular complexes or their enclosure within compartments. A cell biology example is the birth and death of cells, and their spatially contingent mechanical and signaling relationships within a developing organism. Examples in vision and robotics result from common-sense reasoning about movable extended objects and their relationships.

Variable-structure systems are more difficult to model and infer than are dynamical systems with static structure. The existence of domain objects, and their relationships, vary over time; typically this means that mathematical equations expressing such models change over time or take a more elaborate form than do fixed-structure systems. In addition, variable-structure systems can also be potentially infinite ones, although intrinsic resource constraints can be included to prevent this.

Dependency diagrams (DD's, [1]) are defined here as labelled graphs with enough label information that there exists a single standard semantic function Ψ that acts on each member of a DD class and produces, as its value, a joint probability distribution for the random variables that label selected nodes in the diagram. For example, Factor Graphs (FGs) provide a semantics (in the form of factored probability distributions) that encompasses Markov Random Fields (MRFs) and Bayes Nets (BNs), provided that the probability factor nodes are each labelled with a member of a suitable function space $\mathcal{F}$. Fixed-structure Dependency Diagram classes include graphical models such as MRFs, BNs, and FGs [1], as well as Constraint Networks.

In this paper, by adding new kinds of node and link labels to the graph specification we will extend Dependency Diagram (DD) classes to efficiently represent variable-structure systems, with a formal semantics. We will illustrate this procedure with a simple example of a variable-structure system in Section 2: a resource-bounded but otherwise arbitrarily shaped tree of locally interrelated feature vectors, created by a context-free birth-and-death process (Proposition 1). This feature tree can serve as a generative model for hierarchical clustering, for phylogenetic trees, or for noninteracting cell lineages in biology. In Section 3 we define the Dependency Diagram framework and show how it expresses the feature tree and other variable-structure systems (Section 3.6, Proposition 2). This and other resource-bounded variable structure systems can be reduced to fixed-structure factor graphs by what turns out to be an inefficient albeit uniform mapping, $\mathcal{E}$ (Section 3.6, Proposition 2, and Section 3.8, Proposition 3). We hypothesise that the non-uniformity of efficient reduction reflects essential differences between fixed and variable-structure DD's. Related but inequivalent formalisms for extending graphical models have been proposed in [2][1,3-4]. Related but inequivalent frameworks for modeling variable-structure systems are defined in [5] (branching or birth-and-death processes), [6] (marked point processes), [7] (MGS modeling language using topological cell complexes), [8] (interacting particle systems), [9] (BLOG probabilistic object model) [10] (adaptive mesh refinement with rewrite rules), and colored Petri Nets [11].

It is also possible to provide a mathematical definition (and derive some of its consequences) for the semantics of a very different and more dynamical notation for variable-structure probabilistic models: *Stochastic Parameterized Grammars* (SPG's), a subclass of "dynamical grammars". The key function of interest for the SPG framework is again a semantic map Ψ from syntactic expressions of grammars to time evolution operators on probability distributions. We show that SPG's are also capable of expressing the context-free random tree of feature vectors generated by an arbitrary fixed distribution $q(n)$ on the number of branches or children, n, branching from each node. In Section 2 we introduce a grammar for this especially simple example of a variable-structure system. We show how the size-conditioned tree distribution can be sampled efficiently by transforming it to another context-free SPG. In Section 4 we define the semantics of context-sensitive SPG's in both continuous-time and serial discrete-time execution models, relate the two, and show they specialize to the context-free feature tree (Section 4.3, Proposition 4). In Section 5 we relate the DD and SPG frameworks for variable-structure systems and discuss their implications.

2 The Hierarchical Feature Tree

We consider a simple example of a variable-structure system: a (probabilistic) context-free feature tree, with a fixed arbitrary distribution $q(n)$ on the number of children at each node. We will impose a resource limitation on this tree, by conditionalizing on the size of the generated tree defined as its number of nodes. We show (Proposition 1) that the resulting resource-bounded variable-structure system is equivalent to another context-free stochastic parameterized grammar and its distribution on feature trees.

2.1 Unbounded model formulation

Consider a (probabilistic) tree of feature vector nodes with only local dependencies between the features, and an arbitrary fixed distribution $q(n)$ on the number n of children at each node. We will impose a resource limitation on this tree, by conditionalizing on the size of the generated tree defined as its number of nodes N. This model incorporates a birth-and-death process [5] for the tree nodes and has applications to hierarchical clustering in $\mathbb{R}^d$, cell lineage trees with a state vector for each cell, and evolutionary phylogeny trees with a genotype (e.g. a discrete sequence string in $\mathbb{Z}_2^d$ or $\mathbb{Z}_4^d$) for each species. It can also serve as a scaffold structure (e.g. a cell lineage tree) for many other more complex variable-structure dynamical systems. We show (Proposition 1) that if we conditionalize on tree size, the resulting resource-bounded variable-structure system is equivalent to another context-free SPG and distribution on feature trees. The size-conditionalized feature tree is an efficiently computable object with an exchangeable distribution, and is important both for practical use and for the basic theory of variable-structure systems. Any potentially infinite system raises a computational obstacle, and we demonstrate here a strategy (size conditionalization by grammar transformation) to removing such obstacles.

A tree of feature vectors is "context-free" in our terminology if it can be generated by a context-free stochastic grammar, i.e. one in which each rule has only one term on the left hand side. In the first grammar below, this means each node in the tree has a feature vector that is conditionally independent of all others except for the feature vector of its parent node, and the number of child nodes is conditionally independent of everything except the existence of the parent node within the tree. What is novel in this Section is the highly generalizable grammar syntax (to be generalized in Section 4), the introduction and use of a grammar-level transformation to a closely related resource-bounded system (proving Proposition 1), and (as far as we know) the relation between q and new probability distributions r, Q, and R in that variable-structure system.

Figure 1 shows two randomly-generated feature trees. The probability of the first structure is $q(1)\,q(2)^2\,q(0)^3$ because is has one node with one child, two nodes with two children and three nodes with zero children. Likewise the probability of the second structure is $q(3)\,q(2)\,q(1)\,q(0)^4$. Given the structure, we can compute feature probabilities. The conditional probability density of the feature vectors in the first tree is $\phi(x_1 \mid x)\,\phi(x_{11} \mid x_1)\,\phi(x_{12} \mid x_1)\,\phi(x_{111} \mid x_{11})\,\phi(x_{112} \mid x_{11})$, and that for the second tree is $\phi(x_1 \mid x)\,\phi(x_2 \mid x)\,\phi(x_3 \mid x)\,\phi(x_{11} \mid x_1)\,\phi(x_{12} \mid x_1)\,\phi(x_{21} \mid x_2)$, according to the usual rules of a BN with fixed structure.

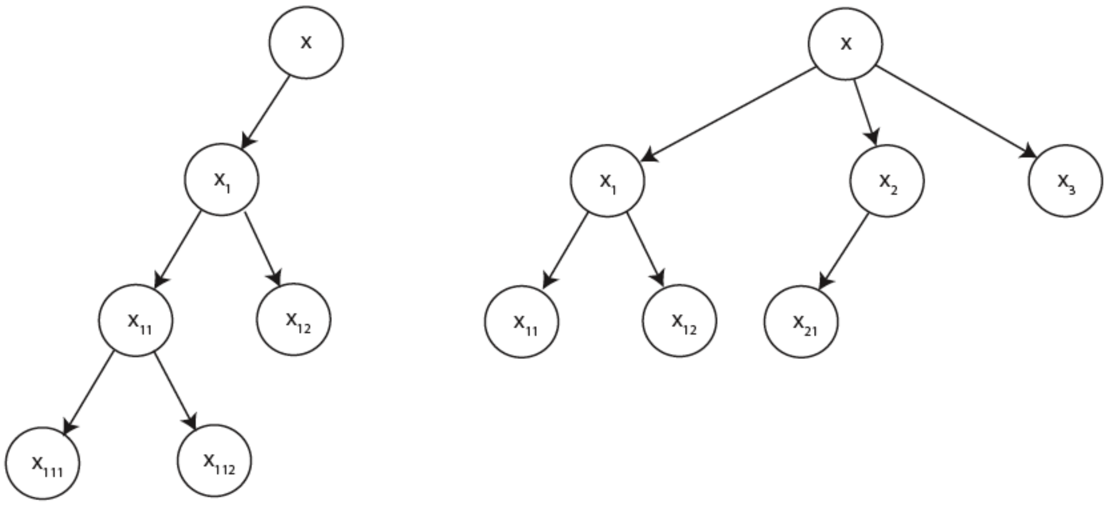


Figure 1: Two feature trees generated by the *clustergen* stochastic parameterized grammar. $\Pr = q(1)\,q(2)^2\,q(0)^3 \quad \times\phi\,(x_1 \mid x)\,\phi(x_{11} \mid x_1)\,\phi(x_{12} \mid x_1) \quad \times\phi(x_{111} \mid x_{11})\,\phi(x_{112} \mid x_{11})$. (b) $\Pr = q(3)\,q(2)\,q(1)\,q(0)^4$ $\times\phi\,(x_1 \mid x)\,\phi(x_2 \mid x)\,\phi(x_3 \mid x) \;\; \times\phi(x_{11} \mid x_1)\,\phi(x_{12} \mid x_1)\,\phi(x_{21} \mid x_2)$.

In the classic programming style of rule-based expert systems, we can generate clusters or other random objects hierarchically with a Stochastic Parameterized Grammar [12]. The following grammar has two rules R1 and R2, and it operates on three kinds of objects: "node" objects (the final output tree nodes), "nodeset" objects (a single one of which is the input nodeset object), and intermediate "child" objects that eventually become nodesets (via R2) that give rise to nodes and more children (via R1). Each rule is eligible to "fire" (to transform its left hand side into its right hand side) whenever an object matching its left hand side is present. Only one rule-firing can occur at each discrete time step. The probability of each eligible rule-firing (i.e. each combination of a rule and a set of objects that matches its left hand side) is uniform for this particular grammar, since both rule-specific distributions q and ϕ are normalized. However, the relative probabilities of the results of a rule firing, i.e. of the right-hand-side of a rule, are given by the conditional probability distributions q and ϕ specified in the "**with**" clause of each rule. The grammar is:

```
grammar (serial discrete-time context-free) clustergen (nodeset($x$) $\to$ {node($x_i$)}) {
        R1 : nodeset($x$) $\to$ node($x$), {child($x$) | $1 \leq i \leq n$} with $q(n)$, $n \geq 0$.
        R2 : child($y$) $\to$ nodeset($x$) with $\phi(x \mid y)$
}
```

In the syntax above, "**grammar**" is a keyword that declares what follows to be a grammar; "(serial discrete-time context-free)" specifies the algorithm for rule firing appropriate for context-free grammars i.e. those in which the left hand side (LHS) of each rule consists of a single term. (A formal semantics for all SPG's is exhibited in Section 4.) "*clustergen*" is the name of the grammar; and "$\text{nodeset}(x) \to \{\text{node}(x_i)\}$" specifies that the input of the grammar is a single nodeset object and its output is a set of node objects each with a value for a random variable x_i (arbitrarily numbered). Thus the grammar implements a single rule which could be invoked recursively by this or another grammar, using the **via** keyword (not shown) in place of the **with** keyword. R1 and R2 are the two rules whose function has been outlined. The rule labels "R*r*:" are optional and rule order is arbitrary. Thus a branching stochastic process starts with a "nodeset" with feature vector x, and (under rule R1) generates a terminal "node" term along with n "child" terms according to a probability distribution $q(n)$. For example, $q(n)$ could follow a power law. Each child (under rule R2) changes its real-valued feature vector x according to the conditional distribution ϕ, becoming a nodeset object eligible to reproduce under (R1). With *clustergen*, we have thus defined the context-free feature tree family $\mathcal{T}(q, \phi)$ as a probability distribution on trees of all sizes.

As examples, for real-valued feature vectors x the conditional distribution could be $\phi(x \mid y) = \text{Gaussian}\, G(x - y)$ resulting in a hierarchical mixture of Gaussians model; for binary (Boolean) vectors x it could be a Bernoulli distribution on whether $x_a = y_a$ or $x_a = \neg\, y_a = 1 - y_a$. Another example ϕ suggested above uses the tree generation number or depth to look up a fixed ratio $\overline{\sigma}_{\text{ratio}}$ by which the standard deviation of a Gaussian changes in each generation. The formulae for these distributions are:

$$\phi(x \mid y) = \text{Gaussian } G(x - y)$$

$$\phi(x \mid y) = \text{Bernoulli Br}(x_1 - y_1; p) \quad \text{// for binary-valued vectors}$$

$$\phi(x \mid y) = \text{Gaussian } G((x_1 - y_1)/y_2)\, \delta(x_2 - y_2\, \overline{\sigma}_{\text{ratio}}(y_3))\, \delta(x_3 - (y_3 + 1))$$

Such grammars are "stochastic", owing to the use of probability; they are "parameterized" since the terms produced carry numeric or other parameters; they are "context-free" since each rule has only one term on the left hand side. Hence: they are particular examples of context-free Stochastic Parameterized Grammars or SPG's. Distributions defined by such grammars can be randomly sampled using an interpreter we have written in the *Mathematica* computer algebra environment.

A key feature of the *clustergen* grammar is that because it is context-free, *every subtree behaves independently of its siblings*. That is, the presence of sibling subtrees only serves to postpone, not to alter, sampling of the distribution of the tree size and shape of a particular child node's descendants.

Next, we define a family $\mathcal{T}(q, \phi, N)$ of feature trees with a fixed, finite size N. The "*clustergen*" grammar results in a certain joint distribution $\Pr(\{\text{node}(x_I) \mid 1 \leq I \leq N\}, N)$. Let us design a resource-limited version of the grammar, so that $\Pr(\{\text{node}(x_I) \mid 1 \leq I \leq N\} \mid N)$ is the same as for the foregoing grammar. Thus, we conditionalize on N, the total number of nodes in the feature tree. For this we need to compute $\Pr(N)$ for $\mathcal{T}(q, \phi)$.

2.2 Size distribution calculations: Pr(N) and Pr(n, N)

$\Pr(N)$ is determined by the function q. Its derivation requires an understanding of birth-and-death processes [5], which can be provided by the generating function point of view. Let $g(z)$ be the generating function for $q(n)$. Also define the generating function $f(x)$ for $\Pr(N)$:

$$g(z) = \sum_{n=0}^{\infty} z^n\, q(n) \;\text{ and }\; f(x) = \sum_{N=1}^{\infty} x^N\, \Pr(N)$$

For example, if q is a geometric distribution, $g(z) = (1-p)/(1-p\,z)$. The power law is of particular interest for many applications.: if q is a power law with power $-\alpha$, $g(z) = p_0 + (1-p_0)\,\mathrm{Li}_\alpha(z)/\zeta(\alpha)$. $f(x)$ is the generating function for $\Pr(N)$ for finite integer values of N only; $1 - f(1)$ is the probability of obtaining $N = \infty$. Some important pairs (g, f) are listed in Table 1.

Table 1. Generating functions for q_i

Distribution name	Distribution $q(n)$	Generating function $g(z)$	Generating function $f(x)$
Geometric	$q_i = p^i\,(1\text{-}p)$	$g(z) = \frac{1-p}{1-p\,z}$	$f(x) = \frac{1-\sqrt{1-4\,p(1-p)\,x}}{2\,p}$
Linear Fractional	$q_n = \delta_{i0}\,\frac{1-b-p}{1-p} + (1-\delta_{n0})\,b\,p^{n-1}$	$\frac{(1-b-p)+(b-p+p^2)\,z}{(1-p)\,(1-p\,z)}$	$f(x) = \left((1-p)-(b-p+p^2)\,x^2 - \sqrt{\left((p-1)^2 + 2\,(p-1)\,(b+2\,b\,p+3\,(p-1)\,p)\,x + (b+(p-1)\,p)^2\,x^2\right)}\right)\Big/ (2\,p(1-p))$
Binomial	$q_n = \binom{N}{n}\,p^i\,(1-p)^{N-n}\,x^n$	$g(z) = ((1-p)+p\,z)^n$	$f(x) = (1-p)^n\,x + \frac{n\,(1-p)^{2\,n}\,p\,x^2}{p-1} + \left((1-p)^{3\,n}\,(-n\,p^2 + 3\,n^2\,p^2)\,x^3\right)\Big/\left(2\,(p-1)^2\right) + \ldots$
binary binomial tree	$q_n = \binom{2}{n}\,p^n\,(1-p)^{2-n}\,x^n$ $n \in \{0, 1, 2\}$	$g(z) = ((1-p)+p\,z)^2$	$f(x) = \frac{1-\sqrt{1-4\,x(1-p)/p}}{2\,x} - \frac{1-p}{p}$
Power law	$q_n = \begin{cases} p_0 & n = 0 \\ (1-p_0)\,n^{-\alpha}/\zeta(\alpha) & n > 0 \end{cases}$	$g(z) = p_0 + (1-p_0)\,\mathrm{Li}_\alpha(z)/\zeta(\alpha)$	$f(x) = p_0\,x + \frac{(1-p_0)\,p_0\,x^2}{\zeta(\alpha)} + (2^{-\alpha}\,(p_0-1)\,p_0\,(-2^\alpha + 2^\alpha\,p_0 - p_0\,\zeta(\alpha))\,x^3)\Big/(\zeta(\alpha))^2 + \ldots$

Surprisingly, the fundamental relationship between f and g is given by the beautiful functional equation [13]

$$f(x) = x\, g(f(x))\,. \tag{1}$$

This follows from the recursion relation

$$\Pr(N) = \sum_{n=0}^{\infty} q(n) \sum_{\left\{ N_i \,\middle|\, 1 \le i \le n \,\wedge\, 1 \le N_i \,\wedge \sum_{i=1}^{n} N_i = N-1 \right\}} \prod_i \Pr(N_i)$$

This functional equation can be solved iteratively using Taylor series expansions and

$$f_{L+1}(x) = x\, g(f_L(x)) \quad \text{and} \quad f_0(x) = 1\ .$$

Convergence is observed to be reliable: one new coefficient is fixed per iteration.

An even more effective solution is by reversion of power series [Pitman 98]. Solve the functional equation to find:

$$f(x) = \tilde{g}^{-1}(1/x),\ \text{where}$$
$$\tilde{g}(y) = g(y)/y$$

In *Mathematica* or other computer algebra systems this can be implemented (either numerically or symbolically) in a single line.

Example: for $g = \frac{(1-p)}{(1-p\,x)}$ in *Mathematica*, the input

```
InverseSeries[Series[1/y ((1 - p)/(1 - p y)), {y, 0, 10}]][[3]]
```

yields the result

```
{1 - p, (1 - p)^2 p, 2 (1 - p)^3 p^2, 5 (1 - p)^4 p^3, 14 (1 - p)^5 p^4,
 42 (1 - p)^6 p^5, 132 (1 - p)^7 p^6, 429 (1 - p)^8 p^7, 1430 (1 - p)^9 p^8,
 4862 (1 - p)^10 p^9, 16796 (1 - p)^11 p^10, 58786 (1 - p)^12 p^11}
```

The appearance of the Catalan numbers $\frac{1}{N+1}\binom{2N}{N}$ in the case of the exponential distribution can be understood from the standard bijection between arbitrary and binary trees, whose number is counted by these integers. The exact solution for this example is:

$$f(x) = \frac{1 - \sqrt{1 - 4\,p(1-p)\,x}}{2\,p},$$

and for the linear fractional case it is

$$f(x) = \frac{1}{2\,p(1-p)}$$
$$\left((1-p) - \left(b - p + p^2\right)x^2 - \sqrt{(p-1)^2 + 2\,(p-1)\,(b + 2\,b\,p + 3\,(p-1)\,p)\,x + (b + (p-1)\,p)^2\,x^2}\right).$$

Note also that the joint distribution $\Pr(n, N) = \Pr(N \mid n)\, q(n)$ satisifes the recursion relation

$$\Pr(n, N) = q(n) \Pr(N \mid n) = q(n) \sum_{\left\{ N_i \,\middle|\, \begin{array}{l} 1 \leqslant i \leqslant n \wedge 1 \leqslant N_i \\ \wedge \sum_{i=1}^{n} N_i = N-1 \end{array} \right\}} \prod_i \sum_{n_i=0}^{\infty} \Pr(n_i, N_i)$$

and therefore has generating a function

$$f(x, y) = \sum_{N=1}^{\infty} x^N \sum_{n=0}^{\infty} y^n \Pr(n, N)$$

which satisfies the functional equation

$$f(x, y) = x\, g(y\, f(x, 1)).$$

These two generating functions are related by $f(x, 1) = f(x)$. So, given $f(x)$, we can very simply compute

$$f(x, y) = x\, g(y\, f(x)). \tag{2}$$

Example: for $g(x) = \frac{(1-p)}{(1-p\,x)}$ we find

$$\Pr(n, N) = (1-p)^N \; p^{N+2n-1} \; \frac{n}{N-1} \binom{2N-n-3}{N-2} \Theta(n < N).$$

where $\Theta(P) = 1$ if proposition P is true, and $\Theta(P) = 0$ if proposition P is false.

To be very clear: the recursion equations can now be solved by direct power series calculations to any order desired, and the results used in an algorithm. What we don't have yet is just the closed form expression or the asymptotics for all useful node degree distributions $q(n)$. Further calculations and comparisons to other approaches are given in the Appendix.

2.3 Resource bounded model

The *clustergen* grammar above results in the joint distribution $\Pr(\{\text{node}(x_I) \mid 1 \leqslant I \leqslant N\}, N)$. Let us design a resource-limited version of the grammar, *rclustergen*, by choosing r and Q below so that $\Pr(\{\text{node}(x_I) \mid 1 \leqslant I \leqslant N\} \mid N)$ is the same as for the *clustergen* grammar:

grammar (serial discrete-time context-free) *rclustergen* (nodeset(x, N) $\to$ {node(x_i)}) {

R1 : nodeset(x, N) $\to$ node(x), {child(x, N_i) $\mid$ $1 \leqslant i \leqslant n$}

with $r(n \mid N)\, Q(\{N_i\}_1^n \mid N-1, n)$, $n \geqslant 0$.

R2 : child(y, N) $\to$ nodeset(x, N) with $\phi(x \mid y)$

}

A further grammar refinement sequentializes the emission of children according to deterministic Q, so that we get an efficient (few-variable) procedure for the resource-limited generation of samples from the same conditional probability $\Pr(\{\text{node}(x_I) \mid 1 \leqslant I \leqslant N\}, \mid N)$ as that of the original grammar. The distribution R must again be chosen accordingly.

grammar (serial discrete-time context-free) *rseqclustergen* (nodeset(x, N) $\to$ {node(x_i)}) {

R1 : nodeset$(x, N) \to$ node(x), children$(x, n, N-1) \mid 1 \leqslant i \leqslant n\}$
with $r(n \mid N)$

R2 : children$(x, n, N) \to$ child(x, N'), children$(x, n-1, N-N')$
with $R(N' \mid n, N)$

R3 : children$(x, 0, N) \to \varnothing$

R4 : child$(y, N) \to$ nodeset(x, N) with $\phi(x \mid y)$

}

Given the arbitrary distribution q, we seek r, Q, and R such that the conditional distributions $\Pr(\{\text{node}(x_I) \mid 1 \leqslant I \leqslant N\} \mid N)$ that emerge from these three grammars are identical, for any ϕ. Given $f(x)$ and $f(x, y)$ from Section 2.1, we can compute $r(n \mid N)$:

$$r(n \mid N) = \Pr(n \mid N) = \Pr(n, N) / \Pr(N)$$

in terms of quantities $\Pr(N)$ and $\Pr(n, N)$ known from the functions $f(x)$ and $f(x, y)$ respectively. Note that $r(n \mid 0) = \delta_{n,0}$. Next solve for Q:

$$Q(\{N_i\}_1^n \mid N, n) = \Pr(\{N_i\}_1^n \mid N, n) = \Pr(\{N_i\}_1^n, N \mid n) / \Pr(N \mid n), \text{ i.e.}$$

$$Q(\{N_i\}_1^n \mid N, n) = \delta\left(N - \sum_{i=1}^{n} N_i\right)\left(\prod_{i=1}^{n} \Pr(N_i)\right) \frac{q(n)}{\Pr(N, n)}. \tag{3}$$

This distribution Q is manifestly exchangeable. It remains to find the distribution R. For $n = 1$ we must have $R(N' \mid 1, N) = \delta(N', N)$ to conserve nodes. For $n > 1$,we calculate that

$$Q(\{N_i\}_1^n \mid N, n) = \delta\left(N - \sum_{i=1}^{n} N_i\right)\left(\prod_{i=1}^{n} \Pr(N_i)\right) \frac{q(n)}{\Pr(N, n)}$$

$$= \delta\left(N - N_n - \sum_{i=1}^{n-1} N_i\right)\left(\prod_{i=1}^{n-1} \Pr(N_i)\right) \Pr(N_n) \left(\frac{q(n-1)}{\Pr(N - N_n, n-1)} \; \frac{\Pr(N - N_n, n-1)}{q(n-1)}\right) \frac{q(n)}{\Pr(N, n)}$$

$$= Q\left(\{N_i\}_1^{n-1} \,\middle|\, N - N_n, n-1\right) \Pr(N_n) \frac{\Pr(N - N_n, n-1)}{\Pr(N, n)} \; \frac{q(n)}{q(n-1)}$$

Therefore

$$Q(\{N_i\}_1^n \mid N, n) = Q\left(\{N_i\}_1^{n-1} \,\middle|\, N - N_n, n-1\right) R(N_n \mid n, N) \tag{4}$$

where

$$R(N_n \mid n, N) = \Pr(N_n) \frac{\Pr(N - N_n, n-1)}{\Pr(N, n)} \; \frac{q(n)}{q(n-1)}, \tag{5}$$

which is again given in terms of quantities known from the generating functions $f(x)$, $f(x, y)$, and $g(z)$.

2.4 Summary

Thus we have established

Proposition 1. There exists a size-bounded version $\mathcal{T}(q, \phi, N)$ of the context-free cluster tree distribution $\mathcal{T}(q, \phi)$, which shares the same probability distribution conditioned on tree size N, and is also generated by a context-free grammar whose single-rule probability distributions can be calculated from the distribution q.

Proof: The context-free stochastic parameterized grammar “clustergen” for hierarchical clusters has a joint probability density function which, when conditionalized on the total number of descendant nodes N, is the pdf of the context-free stochastic parameterized grammar “rclustergen”.

This establishes the existence of the resource-bounded context-free cluster tree distribution $\mathcal{T}(q, \phi, N)$, along with an efficient procedure “rseqclustergen” for sampling it requiring only the few-parameter distributions $r(n \mid N)$ and $R(N' \mid n, N)$. We use $\mathcal{T}(q, \phi, N)$ as a paradigmatic variable-structure system, even though it is much simpler than most we need to study. Simplifications include: it is context-free, and the relationships between random variables are determined upon their birth. However, more complex variable-structure systems can be constructed mathematically by starting with $\mathcal{T}(q, \phi, N)$.

In Section 4 we will introduce a formal semantics for all context-free stochastic parameterized grammars that covers the three grammars introduced above, as a special case of a much broader family of context-sensitive stochastic parameterized grammars or the still broader family of “dynamical grammars”. First, however, we consider the relationship of the context-free tree $\mathcal{T}(q, \phi, N)$ to Boltzmann distributions and graphical models.

3 Semantics for Variable-structure systems: Dependency Diagrams

Based on standard algebraic notation for domain-specific probability distributions and for the energy functions of Boltzmann distributions, we propose an augmented class of graphical models to describe the architecture of variable-structure systems. These models are specified by a labelled graph or “dependency diagram”, some of whose nodes are labelled with random variables. But the diagram also includes new node and link types to express highly structured dependency network architectures (i.e. those of low Kolmogorov complexity, generatable from a short computer program) including but not limited to variable-structure systems.

Dependency diagrams (DD’s) are defined here as labelled graphs with enough label information that there exists a single standard semantic function Ψ that acts on each member of a DD class and produces, as its value, a joint probability distribution function for the random variables that label selected nodes in the diagram. For example, Factor Graphs provide a semantics (in the form of factored probability distributions) that encompasses Markov Random Fields (MRF’s) and Bayes Nets (BN’s), provided that the probability factor nodes are each labelled with a member of a suitable function space $\mathcal{F}$.

A major motivation for formalising the semantic map Ψ is to create software and algorithms that generate problem-specific machine learning and pattern recognition algorithms. Another is to search for more powerful mathematical frameworks to assist human generation of such algorithms. The properties of Ψ will be summarized in Proposition 2 of Section 3.7 below.

With fixed-topology networks it is possible to express the resource-bounded cluster tree of N nodes, $\mathcal{T}(q, \phi, N)$, using a graphical model of $2^N - 1$ random variable nodes and 2^{N-1} dependency links. This is done by preallocating a full binary tree of depth N and standardly encoding the nonbinary tree in a binary tree. We should expect, however, to be able to express this model more efficiently using not much more than $O(N \log N)$

random variable nodes and a constant $O(1)$ amount of specification or "program" information, as for example in the grammar itself. We will instead provide an $O(N^2)$, $O(1)$ solution as summarized in Proposition 3 below.

Clearly, the feature tree has a highly structured architecture since the amount of model specification information is far less than the number of random variables. In general, a variable stucture system has a set of variables and dependencies which is potentially very large or infinite, compared to the constant amount of information required to specify the nonrandom aspects of the dependency link structure. Thus, the graph of dependencies is highly structured as measured by Kolmogorov information content: there is a short program to specify a large object, except for the random variable values that also influence the structure. For these reasons, we require a flexible way to specify highly structured dependency graph architectures. This is provided by indexing nodes and links. The short program is given by the expansion map $\mathcal{E}$ that removes these nodes and links, in what follows.

To begin, we review factor graphs (including directed and undirected probabilistic dependency links). Then we define several further labeled graph elements that are required to define variable-structure systems and the semantic function Ψ [1]: interaction gating links, node existence links, indexing nodes and links, and index constraint links. The definitional relationships among link types are summarized in Table 2. We then provide two different DD's for the context free feature tree as discussed above, and establish some basic properties of the semantics Ψ in Lemma 1 (Section 3.2), Proposition 2 (Section 3.7) and Proposition 3 (Section Section 3.9). Further link type definitions and examples follow, including time delay links that generalize Dynamic Bayes Networks.

3.1 Factor Graphs

Following [14] we recall that undirected (MRF) and directed (BN) graphical models may be incorporated into a common graph framework by introducing probability factor nodes ϕ_α into graphs that denote probability distributions. We assume each factor node ϕ_α is labelled by a member of some function space $\mathcal{F}$ whose values are nonnegative; for example, they may be exponentials of real-valued "potential functions" in another function space $\mathcal{F}'$.

For a dependency diagram, the semantics function Ψ maps such labelled graphs to probability density functions which we can write as follows:

$$\Pr(\boldsymbol{x} \mid \boldsymbol{x}_{\text{initial}}) = \frac{1}{Z} \prod_{\kappa \in \mathcal{K}_1} \phi_\kappa(\{x_i \mid F_{i\kappa} = 1\}) \prod_{\kappa \in \mathcal{K}_2} \phi_\kappa(\{x_i \mid D_{i\kappa} = 1\} \mid \{x_j \mid D_{\kappa j} = 1\}) \tag{6}$$

Here F and D are 0/1-valued adjacency matrices for two separate link types: undirected (MRF-like) and directed (BN) dependency links. An F link represents participation in a potential function in a Boltzman distribution or Markov Random Field. Although the graph F is itself directed, its Boolean-algebra square $U = F F^T$ symmetrically connects random variables that are related by one or more potential functions. By contrast the directed links D form a Directed Acyclic Graph (DAG) and represent a generalized form of conditional distribution in which each probability factor participates in the normalization relationship for directed factor graphs [14] [15], and specializes to a conditional distribution if a variable x_i and a factor κ are uniquely connected by a D-link. In addition, some of the variable nodes x may actually be labeled as fixed parameters. These may be interpreted as conditionalized random variables. F, U, and D type dependency links are distinguished by being labeled with "*f*", "*u*", and "*d*" in the labeled graph representation of a dependency diagram.

The conversion of MRF and BN links into FG f and d links is illustrated in Figure 2. With these interconversions, we can freely intermix f, u, and three types of d links (depending on the types of nodes connected) provided that sthe set of all the d links form a DAG.

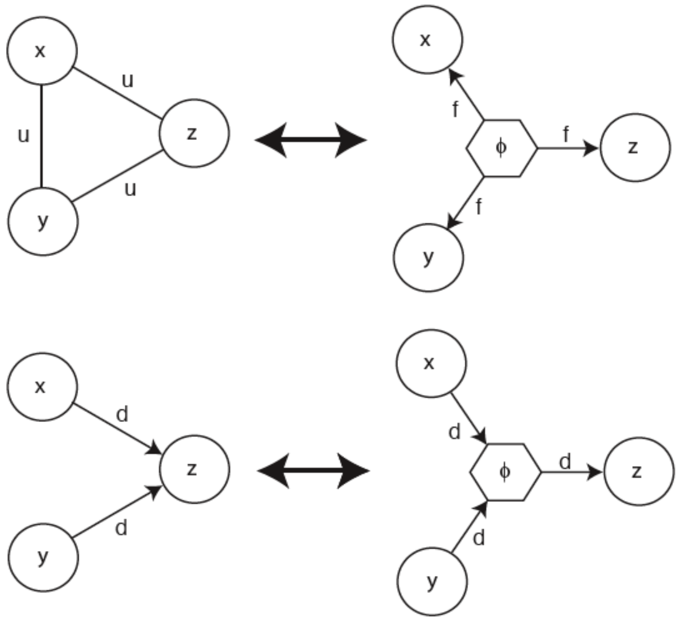


Figure 2: Conversions between MRF u undirected dependency links (top) and BN d directed dependency links (bottom), on the left, and FG f and d links, on the right. There exists a more precise definition of the factor ϕ in the top panel that would require additional pairwise factors between each pair of variables in the FG on the right (not drawn).

Let $\mathcal{P}(\mathcal{F}, N_{\text{rv}}, N_{\text{factor}})$ be the space of probability density functions that can be constructed according to Equation 6 from N_{rv} random variables and from N_{factor} probability factors each in the space $\mathcal{F}$. Let $\mathcal{P}(\mathcal{F}, N_{\text{rv}}) = \mathcal{P}(\mathcal{F}, N_{\text{rv}}, \infty)$.

We denote dependency diagram classes $\text{DD}(\{f, d\}, \{\mathbb{Z}, \mathbb{R}\}, \mathcal{F}, (N_{\text{node}}, N_{\text{link}}))$ for diagrams containing both f and d links, both integer-valued and real-valued random variables, factor functions in a function space $\mathcal{F}$, and bounds $(N_{\text{node}}, N_{\text{link}})$ on the numbers of nodes and links, assuming the above form for the semantics fuction Ψ from class members to probability distributions. More restricted classes such as $\text{DD}(\{f\}, \mathbb{Z}, \mathcal{F}, (N_{\text{node}}, N_{\text{link}}))$ (integer-valued Boltzmann distributions) can be described in a similar way.

Factor graphs as defined have a fixed structure of dependencies given by the matrices F and D, regardless of the values of other variables or parameters such as time. We seek to remove these limitations by defining new node and link types and their semantics. We also consider wherever possible the reduction to previously defined node and link types, and the effects of such reduction on size and complexity parameters such as numbers of nodes and links. The essential new link types are: factor *gating* links labelled by "γ", node and factor *indexing* nodes and links labelled by "ι" (which allow for replication), and node *existence* links labelled by "ϵ". Other convenenient node and link types will be defined in terms of these.

3.2 Gating links

A special case of dependency link is of particular interest for variable-structure systems: the gating link. We assign meaning to such links by extending the semantic function Ψ to dependency diagrams (labelled graphs that denote distributions) that include them. Examples of gating links include all 0/1-valued multiplicative indicator variables or functions in MRF's, such as line processes in region segmentation, cluster membership variables in mixture models, and graph matching assignment matrices [16].

The semantic function Ψ now assigns to each such diagram the probability distribution:

$$\Pr(\{x\}) = \frac{1}{Z} \prod_{\kappa \in \mathcal{K}_1} \phi_\kappa(\{x_i \mid F_{i\kappa} = 1\})^{\left[\prod_{k \mid \gamma(\kappa,k)=1} \Theta\left(x_k > 0\right)\right]}$$
$$\times \prod_{\kappa \in \mathcal{K}_2} \phi_\kappa(\{x_i \mid D_{i\kappa} = 1\} \mid \{x_j \mid D_{\kappa j} = 1\})^{\left[\prod_{k \mid \gamma(\kappa,k)=1} \Theta\left(x_k > 0\right)\right]} \tag{7}$$

Here, the 0/1-valued Heaviside functions Θ(predicate) are in the exponent and are applied to each integer- or real-valued random variable x_k that gates factor κ according to the gating graph links whose adjacency matrix is $\gamma(\kappa, k) = \gamma_{\kappa,k}$. Each product of Heavide functions takes value 0 or 1. Only if all its gating constraints (if any) are met is a probability factor ϕ_κ multiplied into the joint probability distribution. Both D and F type interactions can be gated. Since $\phi_\kappa{}^0 = 1$, in the absence of all gating links, each product $\prod_k \Theta$is empty, hence =1, and the definition for Ψ reduces to the previous one for FG's, Boltzmann distributions, MRF's, and BN's. For this definition to work, we define $0^0 = 1$.

The "standard expansion" map $\mathcal{E}$ for gating links eliminates all such links in favor of F and D links. This can be done by replacing all gating links with ungated ones of D type if possible using *local* renormalization of ϕ_κ, and with F type links if not; also it changes the probability factor ϕ_κ accordingly by raising it to the power of the product all incident 0/1 gating variable values, assuming that is possible within the function class $\mathcal{F}$. The resulting ungated graphical model has the same "meaning" (maps under Ψ to the same joint distribution on the same random variables) as the gated one but is devoid of gating links. In this way, we can *reduce* dependency diagrams with gating links to those without. The cost of doing so is an increase in the number of arguments and the generality of the allowed probability factor functions. The number of nodes and links remains constant.

Two essentially different classes of diagram may be considered: those in which the gating links are constrained to form a DAG when added to D, and those that aren't. If the D and associated γ links form a DAG, then we may attempt a reduction of gated directed links to ungated directed links. If for some reason the normalization property of D-links is lost, any affected probability factors can be moved from the "D" product to the "F" product and all corresponding dependency link labels changed accordingly. Either way we establish the following lemma.

Lemma 1. There exists a semantics function Ψ from DD($\{f, d, \gamma\}$; $\mathcal{F}$; $(N_{\text{node}}, N_{\text{link}})$) to probability density functions $\mathcal{P}(\mathcal{F}, N_{\text{node}})$ on N_{node} variables such that:

(a) Ψ_{fg} specializes to FG's, MRF's and BN's, i.e. it agrees with the standard $\Psi : \text{DD}(\{f,\ d\},\ \mathcal{F},\ (N_{\text{node}},\ N_{\text{link}})) \to \mathcal{P}(\mathcal{F},\ N_{\text{node}})$ on diagrams without γ links. Evaluated on such diagrams, $\Psi = \Psi_{\text{fg}}$.

(b) There exists a "standard expansion" map $\mathcal{E}$ which reduces $\text{DD}(\{f,\ d,\ \gamma\};\ \mathcal{F};\ (N_{\text{node}},\ N_{\text{link}}))$ to $\text{DD}(\{f,\ d\};\ \mathcal{F};\ (N_{\text{node}},\ N_{\text{link}}))$, such that on the domain $\text{DD}(\{f,\ d,\ \gamma\};\ \mathcal{F};\ (N_{\text{node}},\ N_{\text{link}}))$, $\Psi = \Psi_{\text{fg}} \circ \mathcal{E}$. In other words, the following diagram commutes:

$$\begin{tikzcd}
\text{DD}(\{f,\ d,\ \gamma\};\ \mathcal{F};\ (N_{\text{node}},\ N_{\text{link}})) \arrow[rr, "\mathcal{E}"'] \arrow[dr, "\Psi"] & & \text{DD}(\{f,\ d\};\ \mathcal{F};\ (N_{\text{node}},\ N_{\text{link}})) \arrow[dl, "\Psi_{\text{fg}}"'] \\
& \mathcal{P}(\mathcal{F},\ N) &
\end{tikzcd}$$

As we will see, however, this "standard" expansion map $\mathcal{E}$ can be far from the most efficient reduction for particular variable-structure systems. This fact is important to understanding the difference in principle between variable-structure and fixed-structure systems.

3.3 Node existence links

In the context-free tree example, a large number of potential random variables are *not* involved in any given tree, depending on the values (and involvement) of their parent variables. We may indicate non-involvement, or effective non-existence, of a random variable in a tree by using special gating links (labelled "ϵ") to cut off *all* of its interactions with other variables. Thus the semantics $\Psi(D)$ of $\text{DD}(f,\ d,\ \gamma,\ \epsilon\};\ \mathcal{F};\ (N_{\text{node}},\ N_{\text{link}}))$ is:

$$\begin{aligned}
\Pr(\{x\}) = \frac{1}{Z} \prod_{\kappa \in \mathcal{K}_1} \phi_\kappa(\{x_i \mid F_{i\kappa} = 1\})^{\left[\prod_{k \mid \gamma(\kappa,k)=1} \Theta\left(x_k > 0\right) \prod_{k \mid \epsilon(i,k)=1} \Theta\left(x_k > 0\right)\right]} \\
\times \prod_{\kappa \in \mathcal{K}_2} \phi_\kappa(\{x_i \mid D_{i\kappa} = 1\} \mid \{x_j \mid D_{\kappa j} = 1\})^{\left[\prod_{k \mid \gamma(\kappa,k)=1} \Theta\left(x_k > 0\right) \prod_{k \mid \epsilon(i,k)=1} \Theta\left(x_k > 0\right)\right]}
\end{aligned} \tag{8}$$

The products in the exponent are more simply expressed in terms of Boltzmann/Gibbs distribution energy functions, where they just multiply the potential functions $V_\kappa = -\log \phi_\kappa$:

$$E = \sum_{\kappa\in\mathcal{K}_1} V_\kappa(\{x_i \mid F_{i\kappa} = 1\}) \prod_{k \mid \gamma(\kappa,k)=1} \Theta\Big(x_k > 0\Big) \prod_{k \mid \epsilon(i,k)=1} \Theta\Big(x_k > 0\Big)$$
$$+ \sum_{\kappa\in\mathcal{K}_2} V_\kappa(\{x_i \mid D_{i\kappa} = 1\} \mid \{x_j \mid D_{\kappa j} = 1\}) \prod_{k \mid \gamma(\kappa,k)=1} \Theta\Big(x_k > 0\Big) \prod_{k \mid \epsilon(i,k)=1} \Theta(x_k > 0\Big) \quad (9)$$

We may equivalently *reduce* existence links to gating links as follows: constrain

$$\epsilon(i, k) = 1 \Leftrightarrow (\forall\, \kappa : F_{i\kappa} = 1 \vee D_{i\kappa} = 1 \Rightarrow \gamma'(\kappa, k) = 1)$$

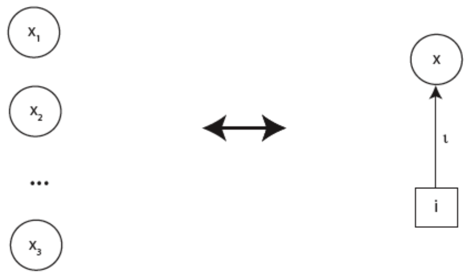
which may be achieved by defining

$$\gamma'(\kappa, k) = \gamma(\kappa, k) \vee (\exists\, i \mid \epsilon(i, k) \wedge (F_{i\kappa} = 1 \vee D_{i\kappa} = 1)).$$

In this way we extend the standard expansion map of Proposition 1 to include ϵ links: Define $\mathcal{E}$ on $\mathrm{DD}(\{f, d, \gamma, \epsilon\}; \mathcal{F}; (N_{\text{node}}, N_{\text{link}}))$ by first mapping to $\mathrm{DD}(\{f, d, \gamma\}; \mathcal{F}; (N_{\text{node}}, N_{\text{link}}))$ (using the identity function if there are no ϵ links), and then by the previously defined $\mathcal{E}$ to $\mathrm{DD}(\{f, d\}; \mathcal{F}; (N_{\text{node}}, N_{\text{link}}))$. This extends Lemma 1 to its obvious analog for $\mathrm{DD}(\{f, d, \gamma, \epsilon\}; \mathcal{F}; (N_{\text{node}}, N_{\text{link}}))$ and is a step towards proving Proposition 2(a) and (b) in Section 3.7 below.

3.4 Indexing links

One more significiant notational extension is required to express variable-structure system architectures in which a fixed amount of network description information controls a variable number of random variables and dependencies. In structured applications such as those involving time, space, or other architectural regularities, conventional algebraic notation for generative models expands to include subscripts, indices, or their equivalent. Here we incorporate such indices as part of the formal specification and semantics of Dependency Diagrams. Using index nodes, a fixed amount of network description information can specify a variable number of random variables and dependencies. Index nodes are a reformulation and extension [1] of the Plates notation of [2]. The key idea is that, whenever variables or functions could algebraically appear with subscripts indicating multiplicity, a dependency diagram has an "index node" with an "index link" to the corresponding variable or function node. Figure 4 shows the replacement of repeated random variables, with or without interactions, by index nodes and index links labelled "ι" (iota).



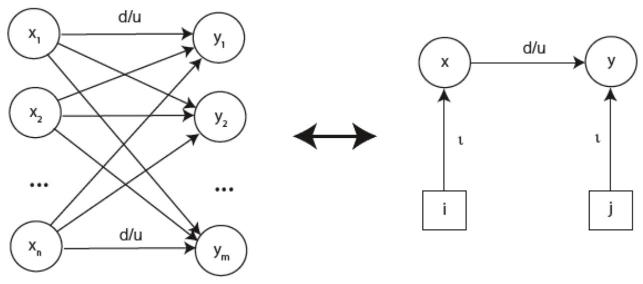


Figure 3: Indexing nodes. (a) An indexed set of random variables replaced by a variable node and an index node. (b) An indexed set of random variables and their directed or undirected dependencies, replaced by two indexed random variables and their directed or undirected dependence.

In the absence of gating, we may express indexing with ι (iota) links most simply using sparse matrices (also denoted by iota) as follows:

$$\iota(i, \alpha) \in \{0, 1\}, \ \iota(\kappa, \alpha) \in \{0, 1\}$$

where implicitly we constrained i to index the variable nodes x_i, α to index the index nodes a_α, and κ to index the probability factor nodes ϕ_i. Of course, the symbols α, i, and κ are not (necessarily) themselves index nodes but rather meta-indices in the mathematical language we are using to describe dependency diagrams. Indeed, by augmenting the metaindex i of random variable x_i with an ordered set of indices (a_α) that index x_i as specified by $\iota(i, \alpha) = 1$, we obtain the indexed random variable $x_{i,(a_\alpha | \iota(i,\alpha)=1)}$. A particular example would be $x_{5,(a_1,a_3)}$ or, even more specifically, "grade$_{(\text{student, course})}$".

The semantics of index nodes and links are given, under the restrictive assumption of a single level of indexing in DD(d,ι), by the following probability density formula. Define $J(\kappa)$ to be the *ordered* set $(a_\mu \,|\, \iota(\kappa, \mu) = 1)$, and likewise define $J(i) = (a_\alpha \,|\, \iota(i, \alpha) = 1)$ and $J(j) = (a_\beta \,|\, \iota(j, \beta) = 1)$. Then the value of Ψ is

$$\begin{aligned}\Pr(\{x\}) = \quad & \frac{1}{Z} \prod_{\{a_\alpha \in \mathcal{I}_\alpha\}} \prod_{\kappa \in \mathcal{K}_1} \phi_{\kappa, J(\kappa)}(\{x_{i, J(i)} \,|\, F_{i\kappa} = 1\}) \\ & \times \prod_{\kappa \in \mathcal{K}_2} \phi_{\kappa, J(\kappa)}(\{x_{i, J(i)} \,|\, D_{i\kappa} = 1\} \,|\, \{x_{j, J(j)} \,|\, D_{\kappa j} = 1\})\end{aligned} \quad (10)$$

Note that all products over indices act within the same global scope - there is no nesting of parenthisized subproducts over indices. This is only the simplest situation. In general, index nodes introduce the need for a compatible tree of index scope nodes (with an implicit root node for the whole diagram) that determines which probability factors are within scope for which index products.

Already this notation generalizes Plates in the following ways: (a) Flexible weight sharing is possible: there is no longer a constraint that if two variables are connected by a dependency link, and both are indexed by the same index a_α, that their interaction factor is always (or is always not) similarly indexed. The degree of weight sharing is specified flexibly by $\iota(i, \alpha) \in \{0, 1\}$ and $\iota(\kappa, \alpha) \in \{0, 1\}$. (b) There is no longer a constraint that all nodes indexed by a common index node must be arranged "inside" a compact grouping in a two dimensional layout. When these groupings have partial overlap, the problem of drawing such a layout becomes that of drawing Venn diagrams with many independent sets.

The following generalizations of the foregoing indexing mechanism can be added, go further beyond Plates, and are straightforward to express in terms of more general probability formulas: (a) multiple levels of indexing (subscripts on the subscripts, in a DAG of iota-relationships); (b) combination of indexing with gating links "γ" and therefore with existence links "ϵ"; (c) numerical index constraint links δ_ι, enforced by Kronecker delta function factors; (d) reordering the indices of some variable and factor nodes (from the default numerical order) to express multidimensional transpose operations; (e) constraints δ_u (expanded into probability factors ϕ_κ using Kronecker and Dirac delta functions) relating random variable values to index values and/or to each other; (f) $\tilde{\iota}(\kappa, \alpha) \in \{0, 1\}$ relationships that allow an indexed set of variables as arguments to a single probability factor; (g) upper index limits λ that can be variable rather than constant as assumed so far. In the proof of Proposition 2, below, we will only need (a), (b), and (c) above.

For example, valid indexing and gating (ι and γ) links can be combined (as required by generalization (b) above) by the following formula. Again define $J(\kappa)$ to be the ordered set of index symbols $J(\kappa) = (a_\mu \mid \iota(\kappa, \mu) = 1)$, and define $J(i) = (a_\alpha \mid \iota(i, \alpha) = 1)$, and so on. Then Ψ is

$$\begin{aligned}\Pr(\{x\}) = \frac{1}{Z} \prod_{\{a_\alpha \in \mathcal{I}_\alpha\}} &\left(\prod_{\kappa \in \mathcal{K}_1} \phi_{\kappa, J(\kappa)}(\{x_{i, J(i)} \mid F_{i\kappa} = 1\})^{\left[\prod_{k \mid \gamma(\kappa,k)=1} \Theta\left(x_{k, J(\kappa)} > 0\right) \prod_{k \mid \epsilon(i,k)=1} \Theta\left(x_{k, J(\kappa)} > 0\right)\right]} \right) \\ \times &\left(\prod_{\kappa \in \mathcal{K}_2} \phi_{\kappa, J(\kappa)}(\{x_{i, J(i)} \mid D_{i\kappa} = 1\} \mid \{x_{j, J(j)} \mid D_{\kappa j} = 1\})^{\left[\prod_{k \mid \gamma(\kappa,k)=1} \Theta\left(x_{k, J(\kappa)} > 0\right) \prod_{k \mid \epsilon(i,k)=1} \Theta\left(x_{k, J(\kappa)} > 0\right)\right]} \right)\end{aligned} \tag{11}$$

For multiple levels of indexing (as required by generalization (a) above), we have a more complicated formula for the PDF. Since $\iota(\alpha, \beta)$ is a DAG, we may self-consistently define the *ordered* sets of symbols

$$\begin{aligned} J(i) &= (a_{\alpha, J(\alpha)} \mid \iota(i, \alpha) = 1) \\ J(\kappa) &= (a_{\alpha, J(\alpha)} \mid \iota(\kappa, \alpha) = 1) \\ J(\alpha) &= (a_{\beta, J(\beta)} \mid \iota(\alpha, \beta) = 1) \end{aligned}$$

This generalizes the foregoing single-level definition in which $J(\alpha)$ was null, and allows for arbitrary levels of indexing in a DAG. If (without loss of generality) α is numbered in an order compatible with the $\iota(\alpha, \beta)$ DAG, so that $\iota(\alpha, \beta) = 1 \Rightarrow \alpha > \beta$), then $\alpha < \beta \Rightarrow \alpha \leqslant \beta \Rightarrow \neg\, \iota(\alpha, \beta)$. Then Ψ is

$$\begin{aligned}\Pr(\{x\}) &= \frac{1}{Z} \prod_{\{a_1\}} \prod_{\{a_{2,J(2)}\}} \cdots \prod_{\{a_{\alpha,J(\alpha)}\}} \cdots \prod_{\{a_{A,J(A)}\}} \\ &\times \left(\prod_{\kappa \in \mathcal{K}_1} \phi_{\kappa,J(\kappa)}(\{x_{i,J(i)} \mid F_{i\kappa} = 1\})^{\left[\prod_{k \mid \gamma(\kappa,k)=1} \Theta\left(x_{k,J(\kappa)} > 0\right) \prod_{k \mid \epsilon(i,k)=1} \Theta\left(x_{k,J(\kappa)} > 0\right)\right]} \right) \\ &\times \left(\prod_{\kappa \in \mathcal{K}_2} \phi_{\kappa,J(\kappa)}(\{x_{i,J(i)} \mid D_{i\kappa} = 1\} \mid \{x_{j,J(j)} \mid D_{\kappa j} = 1\})^{\left[\prod_{k \mid \gamma(\kappa,k)=1} \Theta\left(x_{k,J(\kappa)} > 0\right) \prod_{k \mid \epsilon(i,k)=1} \Theta\left(x_{k,J(\kappa)} > 0\right)\right]} \right.\end{aligned} \tag{12}$$

The other successive generalizations (c-g) have similar general formulae which grow larger, typographically.

A second class of index link generalizations is best combined with the idea of “scope” for products over index variables. We introduce (h) a tree or forest of index scope nodes and links $\sigma(\{\alpha, \kappa, i, \chi'\}, \chi) \in \{0, 1\}$, whose leaves are index, factor, or variable nodes. Here χ and χ' index scope nodes. These $\sigma = 1$ links determine the placement of parentheses in the repeated products over probability factors. Indexing relationships ι must “respect” i.e. stay within the scope σ of the source index node, recursively defined. By default there is a single global scope. Unlike Plates, scopes may not overlap except by nesting since they represent a parse tree of the algebraic expression for the probability density. There is also a generalization to (i) pure index constraints δ_ι (here, 0/1-valued gating exponents) that obligatorily relate index node values within any scope including the global one. These index constraint links are expanded (transformed away) by gating all interactions in the smallest common scope of the indices involved. Given this generalization, (j) repeated index nodes of the same name (e.g. i and i) are defined as different indices (e.g. ${}^1i$ and ${}^2i$) having Kronecker delta function constraints (numerical identity) between them. Finally (k) argument indexing $\tilde{\iota}(\kappa, \alpha)$ within a scope means that factor ϕ_κ has its own bound or dummy variable b_α for index α and does not appear within the product scope of a_α if any. Therefore all a_α -indexed components of an argument x to ϕ_κ are jointly arguments to the same instantiation of the function ϕ_κ .

In the simplest case, and also in all the above generalizations of indexing, there is once again a “standard expansion” map $\mathcal{E}$ that maps indexed diagrams to nonindexed ones of potentially much greater size. To evaluate $\mathcal{E}$, simply replace each node x_i with a large number of nodes $x_{i,J(i)}$, and each factor node ϕ_κ with a large number of factor nodes $\phi_{\kappa,J(\kappa)}$, connected as indicated in the probability density formula for Ψ.

The new link types defined by the expansion map, and the link types in terms of which they are defined, are summarized in Table 2.

Table 2. Dependency Diagram link types: definitional dependencies (itself a DAG)

Link type	Symbol	Definable in terms of ...
*f*actor dependency	f	stat mech, axiomatic, or d
*u*nconditional dependency	u	f, stat mech, or axiomatic
*d*irected (conditional) dependency	d	u, or axiomatic
gated interaction	γ	(f, d)
node existence	ϵ	γ
constraint on random variables	δ_{u}	$(u, \delta_{\mathrm{Kronecker}}, \delta_{\mathrm{Dirac}})$
identity of variable node names	repetition	δ_u
index	ι	repetitive expansion
scope	σ	ι
constraint on indices	δ_ι	$(\iota, \gamma \{, \sigma\})$
identity of index node names	repetition	δ_ι
argument indexing	$\tilde{\iota}$	$\iota \{, \sigma\}$, repetitive expansion
time delay	$(f \mid d)\delta\mathrm{t}$	$(\iota, (f \mid d), \delta_\iota)$
variable index limit	λ	ϵ
value type	v	(δ_u, ι)
permutable indexing	ι_σ	repetitive expansion

Example Constrained and multilevel indexing is illustrated in the following diagram fragment. It contains no probability factors but hierarchically indexes a single random variable $x_{l,(i_1, \ldots i_k, \ldots i_l)}$.

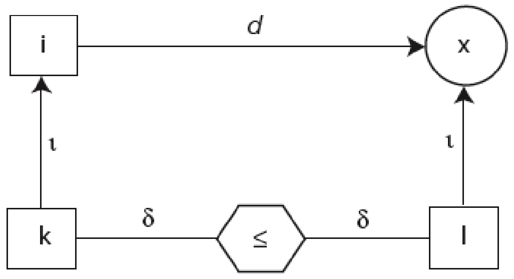


Figure 4: Random variable node x (circle) indexed by (squares) level number l, lineage indices i_k, where k is constrained to be $\leq l$ (hexagon).

Diagrams can be drawn more simply by omitting selected link labels according to the default link type conventions proposed and listed in Table 3.

Table 3. Default arrow types as a function of node types

Source node type	Target node type	Arrow D → /U —	Default link type	Other link types
probability factor	random variable	D →	f, factor dependency	d, δ
random variable	random variable	U —	u, unconditional dependency	δ_u
random variable	probability factor	D →	d, directed conditional dependency	ϵ, γ
random variable	random variable	D →	d, directed conditional dependency	-
index	random variable	D →	ι, index	ι_σ
index	index	D →	ι, index	-
index constraint	index	U —	δ_ι, index constraint	-
random variable	index	D →	λ	-
scope	index, scope	D →	σ, scope	-

With these node and link types, translation to and from Boltzmann distribution and related architectures is possible.

Example: A single-level clustering model can be modeled as a mixture of Gaussians (Figure 5).

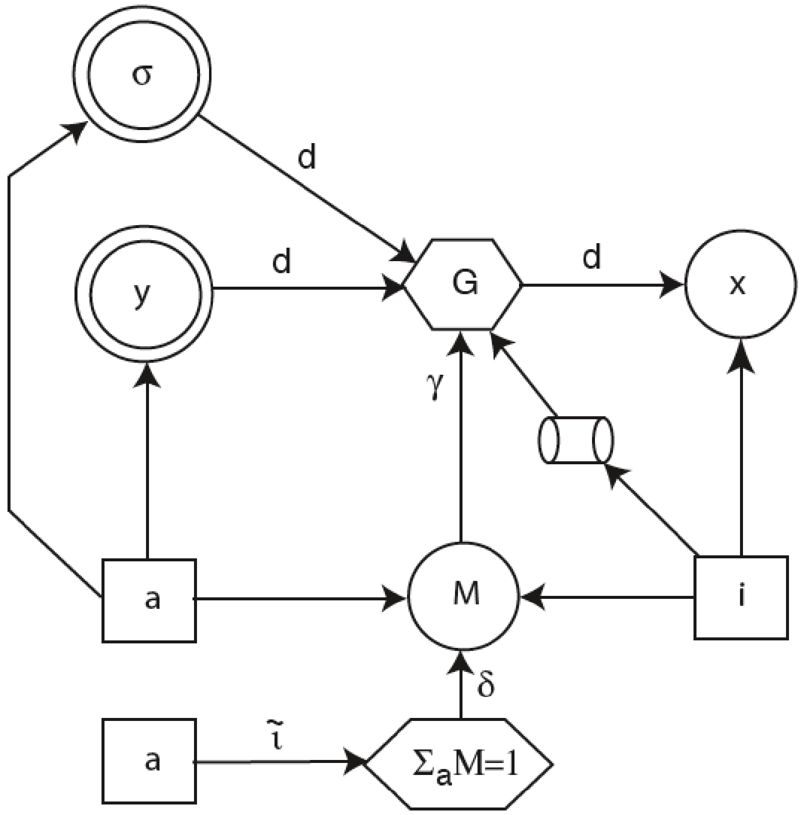


Figure 5: Dependency diagram for a mixture of Gaussians $G(x;\,y,\,\sigma) = G((x-y)/\sigma)$ (hexagonal probability factor node at top) with unique cluster membership constraint (hexagonal node at bottom). Unlabelled links are of default type "ι" or "σ" (cf. Table 3). Cluster membership indicator variables are $M_{i\,a}$; data vectors are x_i (circles); fixed cluster centers are y_a; fixed standard deviations are σ_a (double circles). Cluster index a is in the global scope, which encloses a smaller scope containing data vector index i (squares). Scope node (unlabelled tube) nests scope of G inside the scope of i which is global.

Including the global scope as outer parentheses and the indicated scope node as inner parentheses, the semantics $\Psi(D)$ for this diagram is is

$$\left(\prod_{i=1}^{i_{\max}} \delta\left(\sum_{a=1}^{a_{\max}} M_{i\,a} - 1\right)\left(\prod_{a=1}^{a_{\max}} G(x_i - y_a \mid \sigma_a)^{M_{i\,a}}\right)\right)$$

Example: There exists a diagram for 2D region segmentation with pixel index nodes and neighborhood index constraints. $x_{i\,j}$ is the reconstructed pixed values; $l_{i\,j}^{(x|y)}$ is the 0/1-valued discontinuity detection indicator variables. The diagram uses gating (by l) and index constraints (on $i' = i+1,\ j' = j+1$). For example, the

Boltzmann energy summand for gated continuity in the x direction is $E = \sum_{i\,j} l_{i\,j}^{(x)} (x_{i\,j} - x_{i+1\,j})^2$. The 2D region segmentation energy function [17] is:

$$\begin{aligned} E &= \sum_{i\,j} l_{x\,i\,j}(x_{i\,j} - x_{i+1\,j})^2 + \sum_{i\,j} l_{x\,i\,j}\, l_{x\,i\,j+1} + \sum_{i\,j} l_{y\,i\,j}(x_{i\,j} - x_{i\,j+1})^2 + \sum_{i\,j} l_{y\,i\,j}\, l_{y\,i+1\,j} + \sum_{i\,j} (x_{i\,j} - I_{i\,j})^2 \\ &= \sum_{i\,j\,i'\,j'} \delta_{i'\,,\,i+1}\, \delta_{j'\,,\,j+1} \left[l_{x\,i\,j}(x_{i\,j} - x_{i'\,j})^2 + l_{x\,i\,j}\, l_{x\,i\,j'} + l_{y\,i\,j}(x_{i\,j} - x_{i\,j'})^2 + l_{y\,i\,j}\, l_{y\,i'\,j} + (x_{i\,j} - I_{i\,j})^2 \right] \end{aligned}$$

and the diagram is:

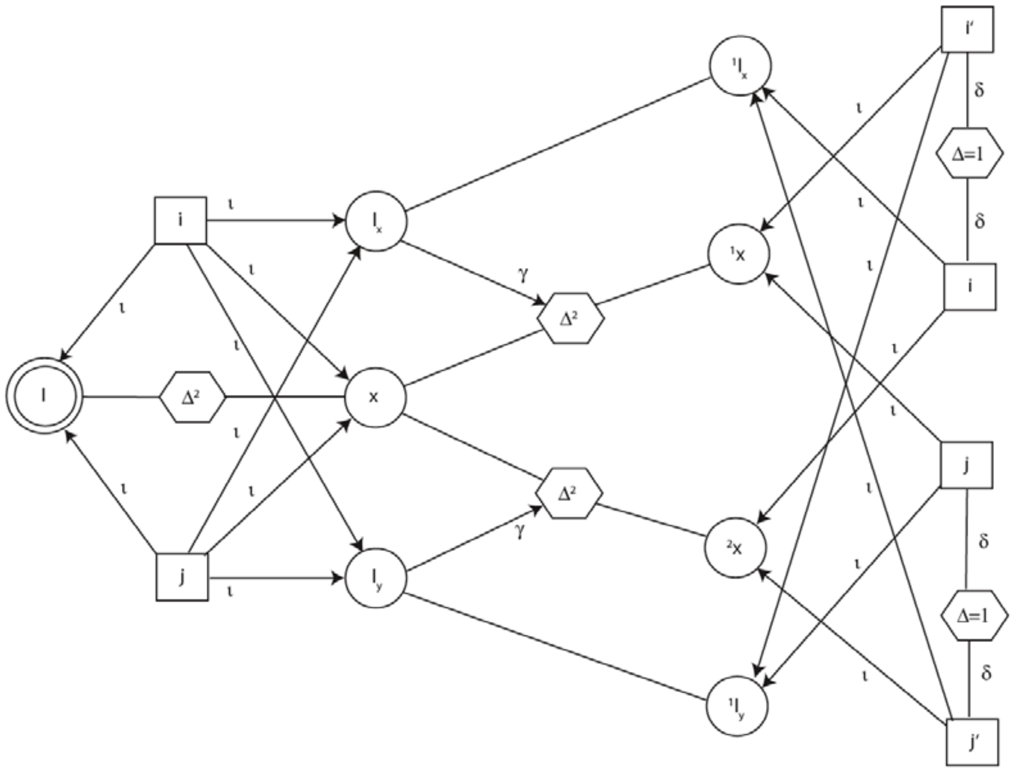


Figure 6: Dependency diagram for region segmentation energy function. Unlabelled links are of default type "u" (cf. Table 3).

3.4.1 Textual encoding of diagrams

A textual encoding of the foregoing dependency diagrams can be given by use of mapping arrows such as "$y \mapsto_d x$" to indicate that y maps (perhaps nonuniquely) to x under the directed d relationship, and "$y \leftrightarrow_u x$" to indicate an undirected arrow of type u. Thus Figure 5 can be represented textually in this way:

$$\text{random variables}: x,\, M,\, \overline{y},\, \overline{\sigma}$$
$$\text{factors}:\ G,\, \Sigma;\ \text{indices}:\ a,\ i;\ \text{scopes}:\ s$$

$$\begin{gathered} y \mapsto_d G,\ \sigma \mapsto_d G,\ M \mapsto_\gamma G,\ G \mapsto_d x,\ \Sigma \mapsto_\delta M \\ a \mapsto_\iota y,\ a \mapsto_\iota \sigma,\ i \mapsto_\iota x,\ a \mapsto_\iota M,\ i \mapsto_\iota M,\ a \mapsto_{\tilde{\iota}} \Sigma, \\ i \mapsto_\sigma s,\ s \mapsto_\sigma a,\ s \mapsto_\sigma G \end{gathered}$$

and likewise for Figure 6.

3.5 Useful derived node and link types

3.5.1 Constraint links for random variables

Arbitrary constraints with multiple levels of indexing are possible in diagrams of the form $\text{DD}((u, d, \iota, \delta))$, using both Dirac and Kronecker delta functions for continuous and discrete valued variables and indices respectively. The semantics are:

$$\begin{aligned} \Pr(\{x\}) \quad &= \frac{1}{Z} \prod_{\{a_1\}} \prod_{\{a_{\alpha=2,J(2)}\}} \cdots \prod_{\{a_{\alpha,J(\alpha)}\}} \cdots \prod_{\{a_{A,J(A)}\}} \\ &\times \prod_{\kappa \in \mathcal{K}(\delta)} \delta\left(g_{\kappa,J(\kappa)}\left(\left\{a_{\alpha,J(\alpha)} \,\middle|\, (\tilde{F}')_{\alpha,\kappa} = 1\right\}, \left\{x_{i,J(\kappa)} \mid \tilde{F}_{i,\kappa} = 1\right\}\right)\right) \\ &\times \prod_{\kappa\in\mathcal{K}_1} \phi_{\kappa,J(\kappa)}(\{x_{i,J(i)} \mid F_{i\,\kappa} = 1\}) \prod_{\kappa\in\mathcal{K}_2} \phi_{\kappa,J(\kappa)}(\{x_{i,J(i)} \mid D_{i\,\kappa} = 1\} \mid \{x_{j,J(j)} \mid D_{\kappa\,j} = 1\}) \end{aligned}$$

This goes beyond function nodes [2] to encompass relation nodes, due to the presence of undirected δ links as specified by matrices $\tilde{F}$ and $\tilde{F}'$, and the functions g_κ.

Example: $\delta\left(\sum_{i=1}^{n+1} x_i^2 - 1\right)$is a rotationally invariant specification of the embedding of the n-sphere into $\mathbb{R}^{n+1}$. Its function g_κ is $\sum_{i=1}^{n+1} x_i^2 - 1$. This constraint is unlike a functional constraint such as $x_{n+1} = \pm\sqrt{\sum_{i=1}^{n} x_i^2}$ as would be required by function nodes as defined in [2].

Inverse images $y = g(x) = 0$ can be used to define embeddings of many smooth manifolds into $\mathbb{R}^d$. The regular value theorem [18] provides conditions on g under which $g^{-1}(y = 0)$ is a submanifold of the manifold containing x. This is also a key idea in level set methods. Here g is a smooth function taking values in $\mathbb{R}^d$ or possibly in another manifold, itself defined by an inverse. According to the Whitney embedding theorem [18], every compact n-dimensional manifold with C^r differentiable structure ($r \geqslant 2$) can be C^r (r-times differentiably) embedded into $\mathbb{R}^{2n+1}$. Often, but not always, these embeddings can be defined as inverse images of C^r functions. For these reasons, Dirac delta functions composed with smooth functions $\delta(g_\kappa(x))$ on $\mathbb{R}^d$ can be expected to provide a powerful way to express probability distributions on manifolds. We can then add manifold-valued variables to the DD notation using “v” links (cf. Table 2) and a manifold description language, and define them in terms of constraint links.

3.5.2 Repeated nodes

Nodes repeated in a diagram can be distinguished textually by different pre-indices (e.g. ${}^1x_i$ vs. ${}^2x_i$) but are constrained to be numerically equal. Expression involving them are standardly expanded by inserting the appropriate Dirac or Kronecker delta functions into the product of probability factors . It is important to repeat nodes e.g. in order to instantiate them with different index symbols in different probability factors and/or scopes, without giving them an unintended Cartesian product structure.

3.5.3 Time delay links

Time delay links (defined in terms of indexing, gating, and constraint links) can be defined by the following expansion.

$$\mathcal{E}_{\delta\mathrm{t}} : (x_j \mapsto_{d\,\delta\,t} x_i) \mapsto (x_j \mapsto_d x_i \ \wedge\ t \mapsto_\iota x_j \ \wedge\ t' \mapsto_\iota x_i \ \wedge\ t \leftrightarrow_{\delta\,(t'-(t+\Delta\,t))} t');$$
$$\mathcal{E}_{\delta\mathrm{t}} : (x_j \mapsto_{u\,\delta\,t} x_i) \mapsto (x_j \leftrightarrow_u x_i \ \wedge\ t \mapsto_\iota x_j \ \wedge\ t' \mapsto_\iota x_i \ \wedge\ t \leftrightarrow_{\delta\,(t'-(t+\Delta\,t))} t')$$

For example, a dynamical chain graph version of a Markov Random Field (which generalizes Dynamic Bayes Nets [19]) is given by the diagram:

$$\{x_{j\,t} \leftrightarrow_u x_{i\,t} \mid j \leftrightarrow_U i\} \bigcup \{x_{i\,t} \mapsto_d x_{i\ t+\Delta\,t}\}$$

for which the expansion is:

$$\mathcal{E}\left[\left\{x_j \leftrightarrow_u x_i \ \middle|\ \left(U\,U^T\right)_{i\,j} = 1\right\} \bigcup \{x_j \mapsto_{d\,\delta\,t} x_i \mid D_{i\,j} = 1)\right] =$$
$$\left\{x_{j\,t} \leftrightarrow_u x_{i\,t} \ \middle|\ \left(U\,U^T\right)_{i\,j} = 1\right\} \bigcup \{x_{j\,t} \mapsto_d x_{i\,t+\Delta\,t} \mid D_{i\,j} = 1\}$$

The Dependency Diagram can be drawn compactly (as a meta-diagram) using index links in this way, and using repeated nodes as defined above in Section 3.5.2. The diagram is drawn in Figure 7 for which a textual representation is:

$$\left\{j \mapsto_\iota {}^{(1)}x,\quad i \mapsto_\iota {}^{(2)}x,\ i \mapsto_\iota {}^{(3)}x,\ {}^{(1)}x \leftrightarrow_{d\,\mathrm{or}\,u} {}^{(2)}x,\quad j \mapsto_{\delta(D\,\mathrm{or}\,U=1)} i,\ t \mapsto_\iota {}^{(1)}x,\ {}^{(2)}x_i \mapsto_{\delta\,t} {}^{(3)}x_i\right\}$$

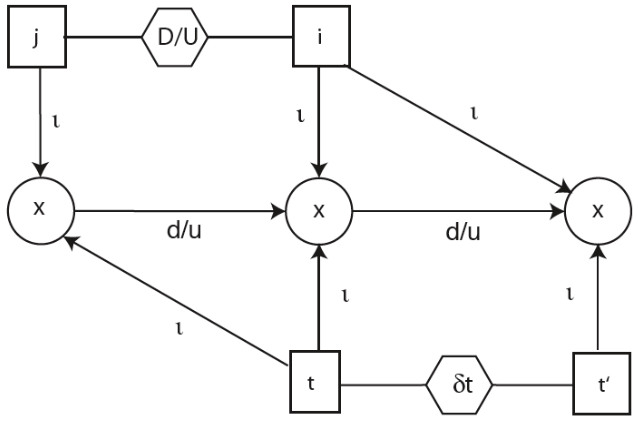


Figure 7: A dynamic version of a Markov Random Field. This is a “metadiagram”; a particular dependency diagram is found by expanding out the D or U matrix and the *i* and *j* (but not *t*) indices.

3.5.4 Subdiagrams

Another architectural element easily defined in DD notation is groups of nodes of any type, for which we allow some kinds of connections to or from a group to expand automatically into connections to all appropriate group members. This can be achieved by extending the use of scope nodes and links. For example, an index link from an index node to an entire scope would expand to a set of index links from the index node to all random variable nodes in the scope. By not also indexing the probability factor nodes, we add a level of weight sharing to such an architecture.

3.5.5 Dependency Diagram Type Notation

A systematic naming convention for many classes C of "Dependency Diagram" was introduced in Section 3.1. It allows a class of diagrams to be specified by describing a list of allowed types and label information, and a list of constraints on the graphs, and including all graphs that obey the resulting constraints. Thus we write "DD(list of allowed compatible link types; allowed value spaces or base types for variables, optional function spaces for probability factors, optional diagram size parameters)". Then (depending on somewhat arbitrary definitions and omitting unconstrained fields) BN = DD(d, $\mathbb{Z}$); Directed Probabilistic Independence Networks = DPIN = DD(d, $(\mathbb{R}, \mathbb{Z})$) ; Undirected Probabilistic Independence Networks = UPIN = DD(u, $\{\mathbb{R}, \mathbb{Z}\}$); PDD = factor graphs = DD($\{d, f\}$, $\{\mathbb{R}, \mathbb{Z}\}$); DD($\{d, \iota\}$, $\{\mathbb{R}, \mathbb{Z}\}$) = IPDD (Indexed Probability Dependency Diagrams); DD($\{\iota, \delta\}$, $\{\mathbb{R}, \mathbb{Z}\}$)= ICDD (Indexed Constrained Dependency Diagrams with no probability factors, for use in constraint systems), and so on. Further parameterization of DD can specify the function space $\mathcal{F}$ for the probability factors associated for example with directed or undirected links DD($\{f, u, d\}$, $\mathcal{F}$, $\{\mathbb{R}, \mathbb{Z}\}$), and can parametrically limit the maximum numbers of nodes and links in each diagram, for example as in DD($\{f, u, d\}$, $\mathcal{F}$, $\{\mathbb{R}, \mathbb{Z}\}$, $(N_{\text{random variables}}, N_{\text{dependency links}})$), which will allow reductions to be stated including their costs. In this regard, index nodes and links require the introduction of a new set of parameters: in addition to the maximal number of variable nodes nodes N_{rv} or probability dependency links N_{links} in the actual diagram at hand, there may also be a limit on the number of random variable nodes $\hat{N}_{\text{rv}}$ or dependency links $\hat{N}_{\text{links}}$ that are obtained after the expansion mapping $\mathcal{E}$ replicates the indexed random variables - these characterize the size of the expanded graph with indices removed. We may add another set of parameters to the DD notation for this purpose: $\text{DD}\big(\{f, u, d\}, \mathcal{F}, \{\mathbb{R}, \mathbb{Z}\}, (N_{\text{rv}}, N_{\text{links}}), \big(\hat{N}_{\text{rv}}, \hat{N}_{\text{links}}\big)\big)$.

3.6 Example: Feature Tree

Example: Consider the resource-limited context-free feature tree $\mathcal{T}(q, \phi, N_{\text{active}})$, which may be used for hierarchical clustering or to model cell lineage. It is an irregular cluster tree of arbitrary depth, with a maximum number of tree nodes $N_{\text{active}} = L_{\max}$. Use index set $\mathcal{I}_L$ isomorphic to the bounded strings $\{(i_1, i_2, \ldots i_{l\leq L}) \mid i_l \in \{0, \ldots, b_{\max} - 1\} = \mathbb{Z}_{b_{\max}}\}$ (for example $b_{\max} = 2$ or $b_{\max} = N_{\text{active}}$). The diagram is:

$$\begin{gathered}
\forall\, l \in \{1, \ldots L\}\ \forall\, p \in \{1, \ldots l\} : \text{ indexing relationships} \\
l \mapsto_\iota \eta,\ i_p \mapsto_\iota \eta_l \in \mathbb{Z}_2,\ \ l \mapsto_\iota y,\ i_p \mapsto_\iota y_l \in \mathbb{R},\ l \mapsto_\iota N,\ i_p \mapsto_\iota N_l \in \mathbb{N}, \\
l \mapsto_\iota n,\ i_p \mapsto_\iota n_l \in \mathbb{N},\ l \mapsto_\iota \sigma \in \mathbb{R} \\
\forall\, l \in \{1, \ldots L\}: \text{ node existence relationships} \\
\eta_l \mapsto_\epsilon y_l\,,\ \eta_l \mapsto_\epsilon N_l \\
\forall\, l \in \{1, \ldots L-1\}: \text{ dependency relationships} \\
\eta_l \mapsto_d \eta_{l+1},\ \ y_l \mapsto_d y_{l+1}\,,\ \ N_l \mapsto_\delta N_{l+1}\,,\ \overline{\sigma}_l \mapsto_d y_{l+1}, \\
\text{as well as } N_{l=0} = N_{\text{active}}
\end{gathered} \tag{13}$$

with

$$
\begin{gathered}
\phi_{y\,y\,\eta}(y_{l+1,(i_1\,\ldots\,i_{l+1})} \mid y_{l,(i_1\,\ldots\,i_l)}, \eta_{l+1,(i_1\,\ldots\,i_{l+1})}, \overline{\sigma}_l) = \text{Gaussian}(y_{l,(i_1\,\ldots\,i_{l+1})}; y_{l,(i_1\,\ldots\,i_{l+1})}, \overline{\sigma}_l)^{\eta_{l+1,(i_1\,\ldots\,i_{l+1})}} \\
\phi_{\eta\,N}(\eta_{l,(i_1\,\ldots\,i_l)} \mid N_{l,(i_1\,\cdots\,i_l)}) = \Theta(\eta_{l,(i_1\,\ldots\,i_l)} = 0 \Leftrightarrow N_{l,(i_1\,\ldots\,i_l)} = 0) \\
\phi_{n\,N}(n_{l,(i_1\,\cdots\,i_l)} \mid N_{l,(i_1\,\cdots\,i_l)}, \eta_{l,(i_1\,\ldots\,i_l)}) = r(n_{l,(i_1\,\cdots\,i_l)} \mid N_{l,(i_1\,\cdots\,i_l)}) \\
\phi_{N\,N\,\eta\,n}(N_{l+1,(i_1\,\ldots\,i_{l+1})} \mid N_{l,(i_1\,\ldots\,i_l)}, \eta_{l,(i_1\,\ldots\,i_l)}, n_{l,(i_1\,\ldots\,i_l)}) = Q(\{N_{l+1,(i_1\,\cdots\,i_{l+1})}\} \mid N_{l,(i_1\,\cdots\,i_l)} - \eta_{l,(i_1\,\ldots\,i_l)}, n_{l,(i_1\,\ldots\,i_l)})
\end{gathered}
\tag{14}
$$

Here Q and r are defined in terms of q as in Section 2.3, particularly Equation 3. The second distribution includes the constraint that enforces quiescence inheritance down the index hierarchy, via the Θ term (which is 1 if its predicate argument is true and 0 otherwise) and since the equation for Q includes the factor

$$\delta\left(N_{l,(i_1\,\ldots\,i_l)} - \left(\eta_{l,(i_1\,\ldots\,i_l)} + \sum_{k=1}^{i_{\max}} N_{l+1,(i_1\,\cdots\,i_l\,k)}\right)\right).$$

This factor implies that $\eta \leqslant N$, consistent with $\eta_{l,(i_1\,\ldots\,i_l)} = 0 \Leftrightarrow N_{l,(i_1\,\ldots\,i_l)} = 0$. Also recall that $r(n \mid 0) = \delta_{n\,0}$, so that $N_{l,(i_1\,\cdots\,i_l)} = 0 \Rightarrow n_{l,(i_1\,\ldots\,i_l)} = 0$. This system is sparsely active, since

$$
\begin{gathered}
N_{\text{node}} = o\left(b_{\max}{}^{L}\right) \text{ and } N_{\text{active}} \leq L \text{ imply} \\
N_{\text{active}} \ll (b_{\max})^{N_{\text{active}}} \leq (b_{\max})^{L} \leq (\text{constant})\, N_{\text{node}}\,.
\end{gathered}
$$

Figure 8 illustrates the dependency diagram (but the actual diagram is a mathematical object).

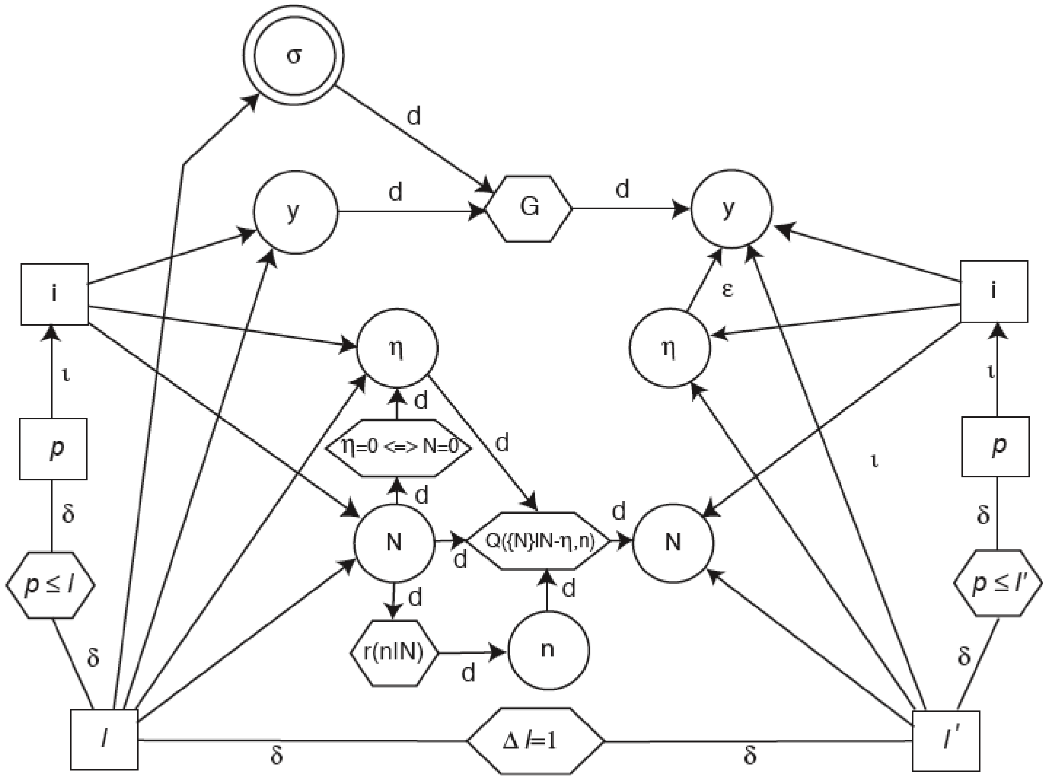


Figure 8: DD for resource-bounded context-free feature tree. Unlabelled links are of default type "ι" (cf. Table 3). Alternatively, Q could be replaced with a product of R factors (Equation 4).

This diagram D can be mechanically translated into a probability density formula as follows:

$$\Psi(D) = \prod_{l=1}^{L} \prod_{l'=1}^{L} \left(\prod_{{}^{1}i_1=1}^{i_{\max}} \cdots \prod_{{}^{1}i_l=1}^{i_{\max}} \right.$$

$$\left. \prod_{{}^{2}i_1=1}^{i_{\max}} \cdots \prod_{{}^{2}i_{l'}=1}^{i_{\max}} \left(X_{l,({}^{1}i_1 \cdots {}^{1}i_l),l',({}^{2}i_1 \cdots {}^{2}i_{l'})} \right)^{\prod_{p|1\leqslant p \wedge p\leqslant l \wedge p\leqslant l'} \delta({}^{1}i_p - {}^{2}i_p)} \right)^{\delta(l'-(l+1))}$$

where

$$X_{l,(i_1 \ldots i_l),l',(j_1 \cdots j_{l'})} = [G(y_{l',(j_1 \cdots j_{l'})} \,|\, y_{l,(i_1 \cdots i_l)}, \sigma_{l,(i_1 \cdots i_l)})]^{\eta_{l',(i_1 \cdots i_{l+1})}}$$
$$Q(\{N_{l,(j_1 \cdots j_{l'})}\} \,|\, N_{l',(i_1 \cdots i_l)} - \eta_{l,(i_1 \ldots i_l)})\, \Theta(\eta_{l,(i_1 \ldots i_l)} = 0 \Leftrightarrow N_{l,(i_1 \ldots i_l)} = 0)\, r(n_{l',(i_1 \cdots i_l)} \,|\, N_{l',(i_1 \cdots i_l)})$$

Hand calculation simplifies this to

$$\begin{aligned}\Psi(D) = \prod_{l=1}^{L} \prod_{\{i_p \in \{1..i_{\max}\} | p \in \{1..l+1\}\}} & [G(y_{l+1,(i_1 \dots i_{l+1})} \mid y_{l,(i_1 \dots i_l)}, \sigma_l)]^{\eta_{l+1,(i_1 \dots i_{l+1})}} \\ Q(\{N_{l+1,(i_1 \dots i_{l+1})}\} \mid N_{l,(i_1 \dots i_l)} - \eta_{l,(i_1 \dots i_l)})\, & \Theta(\eta_{l,(i_1 \dots i_l)} = 0 \Leftrightarrow N_{l,(i_1 \dots i_l)} = 0)\, r(n_{l,(i_1 \dots i_l)} \mid N_{l,(i_1 \dots i_l)})\end{aligned} \tag{15}$$

which, if $L \geqslant N_{\max}$, is the solution to the recursion relation for the full probability density of the resource-bounded context-free cluster tree.

3.7 DDs for Variable-Structure Graphical Models

Combining Lemma 1 with the conclusions of Section 3.3 (defining ϵ links), Section 3.4 (ι indexing links), and Section 3.6 (context-free feature tree), we deduce the following Proposition:

Proposition 2. There exists a semantic function Ψ from $\mathrm{DD}(\{f, d, \iota, \gamma, \epsilon, \delta_\iota\}, \mathcal{F}, (N_{\text{node}}, N_{\text{link}}))$ to $\mathcal{P}(\mathcal{F}, N_{\text{node}})$ (probability density functions on N variables) such that:

(a) Ψ specializes to FG's, MRF's and BN's, i.e. it agrees with the standard $\Psi_{\text{fg}} : \mathrm{DD}(\{f, d\}, \mathcal{F}, (N_{\text{node}}, N_{\text{link}})) \to \mathcal{P}(\mathcal{F}, N_{\text{node}})$ when restricted to diagrams with only *f* and *d* links. Evaluated on such diagrams, $\Psi = \Psi_{\text{fg}}$.

(b) There exists a "standard expansion" map $\mathcal{E}$ which reduces $\mathrm{DD}(\{f, d, \iota, \gamma, \epsilon, \delta_\iota\}, \mathcal{F}, (N_{\text{node}}, N_{\text{link}}))$ to $\mathrm{DD}\big(\{f, d\}, \mathcal{F}, \big(\tilde{N}_{\text{node}}, \tilde{N}_{\text{link}}\big)\big)$, such that on the domain $\mathrm{DD}(\{f, d, \iota, \gamma, \epsilon, \delta_\iota\}, \mathcal{F}, (N_{\text{node}}, N_{\text{link}}))$, $\Psi = \Psi_{\text{fg}} \circ \mathcal{E}$.

(c) There exists a diagram D, of constant size N_{DD} as a function of N_{tree}, for which $\Psi(D)$ is the distribution of the resource-limited context-free cluster tree $\mathcal{T}(q, \phi, N_{\text{tree}})$.

Note that (b) implies that the following diagram commutes:

$$\begin{tikzcd}
\mathrm{DD}(\{f, d, \iota, \gamma, \epsilon\}, \mathcal{F}, N) \arrow[rr, "\mathcal{E}"'] \arrow[dr, "\Psi"'] & & \mathrm{DD}(\{f, d\}, \mathcal{F}, N) \arrow[dl, "\Psi_{\text{fg}}"] \\
& \mathcal{P}(\mathcal{F}, N) &
\end{tikzcd}$$

3.8 Alternative reductions for sparse activation

3.8.1 New complexity measures for sparse activation

A fundamental change in expressive power comes from considering classes of $\mathrm{DD}(\{f,\ d,\ \gamma,\ \iota,\ \delta,\ \epsilon\},\ \ldots,(N_{\text{rv vertex}},\ N_{\text{interaction vertex}},\ N_{f\text{ link}}, N_{d\text{ link}}, N_{\text{active vbl}}, N_{\text{active interaction}}))$ networks parameterized by a bound $N_{\text{active interaction}}$ on the number of f or d interactions that can be gated "on" in any configuration that satisfies the constraints, and by $N_{\text{active vbl}}$ = a bound on the number of active random variables, ie. the number of random variables not gated out of existence by ϵ links in any valid state, in addition to the usual bounds on number of random variables $N_{\text{rv vertex}}$ and dependency links of various types $(N_{\text{interaction vertex}},\ N_{f\text{ link}}, N_{d\text{ link}})$. The gating interactions γ and ϵ can be eliminated in favor of general f and d dependency interactions using the expansion map $\mathcal{E}$ of Lemma 1, which leaves $N_{\text{rv vertex}}, N_{\text{interaction vbl}}, N_{f\text{ link}}, N_{d\text{ link}}$ essentially unchanged and takes no advantage of the sparse activation of variables $N_{\text{active vbl}} \ll N_{\text{rv vertex}}$ or interactions $N_{\text{active interaction}} \ll N_{\text{interaction vertex}}$. However, for at least some probabilistic model classes C with sparse interaction activation, we will show in Proposition 3 below that there exist much deeper reductions than the expansion map $\mathcal{E}$, so that:

$$C \subseteq \Psi(\mathrm{DD}(\{f,\ d,\ \delta\},\ \ldots,(N_{\text{rv vertex}},\ N_{\text{interaction vertex}},\ N_{f\text{ link}}, N_{d\text{ link}}, N_{\text{active vbl}})))$$
$$\text{but also}\, C \sim C' \subseteq \Psi\big(\mathrm{DD}\big(\{f,\ d,\ \delta\},\ \ldots,\ (\tilde{N}_{\text{rv vertex}},\ \tilde{N}_{\text{interaction vertex}},\ \tilde{N}_{f\text{ link}}, \tilde{N}_{d\text{ link}})\big)\big)$$

can be satisfied simultaneously with $\tilde{N}_{\text{rv vertex}} \ll N_{\text{rv vertex}}$, etc., by exploiting the constrained-in sparsity with a nontrivial change of variables. Here the size parameters N's are chosen so that each is minimal, given the others, for model class C and for *equivalent* probabilistic models, under a certain change of variables, in model class C'.

3.8.2 Efficient Reduction

A change of variables is available for the context-free feature tree. It follows the logic for changing variables in Boltzmann distributions introduced in [12], but with a somewhat simpler result in the present case. First consider the subset inclusion which simply adds to the above variables, the arbitrary mapping

$$M_{I;\,l,\,(i_1\,\ldots\,i_l)} \in \mathbb{Z}_2$$

of possible parse tree nodes to unique but arbitrary indices $I \in \{1,\ \ldots N_{\text{active node}}\}$which act as memory addresses. Then there is no risk of contradiction in adding the unique assignment constraints

$$\sum_{I=1}^{N_{\text{active node}}} M_{I;\,l,\,(i_1\,\ldots\,i_l)} = \eta_{i_1\,\ldots\,i_l} \quad \text{and} \quad \sum_{l=1}^{L} \sum_{\{i_1\,\ldots\,i_l\}} M_{I;\,l,\,(i_1\,\ldots\,i_l)} \equiv \zeta_I \leqslant 1.$$

The resulting DD for $\{\eta, y, n, N, M\}$ can be nontrivially mapped to another one with exponentially fewer random variables $\{z, \zeta, m, \hat{N}, C\}$as follows. Define:

$$C_{I J s} = \sum_{l=1}^{L} \sum_{\{i_1 \ldots i_l\}} M_{I; l, (i_1 \ldots i_l)} M_{J; l, (i_1 \ldots i_l\, s)} ,$$

$$z_I = \sum_{l=1}^{L} \sum_{\{i_1 \ldots i_l\}} M_{I; l, (i_1 \ldots i_l)} y_{l, (i_1 \ldots i_l)} , \quad \zeta_I = \sum_{l=1}^{L} \sum_{\{i_1 \ldots i_l\}} M_{I; l, (i_1 \ldots i_l)} \eta_{l, (i_1 \ldots i_l)} = \sum_{l=1}^{L} \sum_{\{i_1 \ldots i_l\}} M_{I; l, (i_1 \ldots i_l)},$$

$$m_I = \sum_{l=1}^{L} \sum_{\{i_1 \ldots i_l\}} M_{I; l, (i_1 \ldots i_l)} n_{l, (i_1 \ldots i_l)} , \quad \hat{N}_I = \sum_{l=1}^{L} \sum_{\{i_1 \ldots i_l\}} M_{I; l, (i_1 \ldots i_l)} N_{l, (i_1 \ldots i_l)} .$$

The new constraints are

$$\sum_{I=1}^{N_{\text{active node}}} C_{I J s} \leqslant \zeta_J , \quad \sum_{J=0}^{N_{\text{active node}}} C_{I J s} \leqslant \zeta_I , \quad \sum_{s=0}^{b_{\max}} C_{I J s} \leqslant \zeta_I\, \zeta_J .$$

$$\zeta_I = 0 \Rightarrow \left(m_I = 0 \wedge \hat{N}_I = 0\right)$$

The inverse mapping is given by

$$M_{I; l, (i_1 \ldots i_l)} = \sum_{\{I_0 \ldots I_l\}} \delta(I_l - I) \prod_{l'=1}^{l} C_{I_{l'-1}\, I_{l'}\, i_{l'}} ,$$

$$y_{l, (i_1 \ldots i_l)} = \sum_{I=1}^{N_{\text{active node}}} M_{I; l, (i_1 \ldots i_l)} z_I , \quad \eta_{l, (i_1 \ldots i_l)} = \sum_{I=1}^{N_{\text{active node}}} M_{I; l, (i_1 \ldots i_l)} \zeta_I ,$$

$$n_{l, (i_1 \ldots i_l)} = \sum_{I=1}^{N_{\text{active node}}} M_{I; l, (i_1 \ldots i_l)} m_I , \quad N_{l, (i_1 \ldots i_l)} = \sum_{I=1}^{N_{\text{active node}}} M_{I; l, (i_1 \ldots i_l)} \hat{N}_I$$

The actual probability distribution (from Equation 15) becomes under this change of variable:

$$\Psi(D) = \prod_{l=1}^{L} \prod_{\{i_p \in \{1..i_{\max}\} | p \in \{1..l+1\}\}} \left[G\left(\sum_J M_{J; l+1, (i_1 \ldots i_{l+1})} z_J \,\middle|\, \sum_I M_{I; l, (i_1 \ldots i_l)} z_I , \sigma_l \right) \right]^{\sum_J M_{J; l+1, (i_1 \ldots i_{l+1})} \zeta_J}$$

$$Q\left(\left\{ \sum_J M_{J; l+1, (i_1 \ldots i_{l+1})} \hat{N}_J \right\} \,\middle|\, \sum_I M_{I; l, (i_1 \ldots i_l)} \hat{N}_I - \sum_I M_{I; l, (i_1 \ldots i_l)} \zeta_I \right)$$

$$\Theta\left(\sum_I M_{I; l, (i_1 \ldots i_l)} \zeta_I = 0 \Leftrightarrow \sum_I M_{I; l, (i_1 \ldots i_l)} \hat{N}_I = 0 \right) r\left(\sum_I M_{I; l, (i_1 \ldots i_l)} m_I \,\middle|\, \sum_I M_{I; l, (i_1 \ldots i_l)} \hat{N}_I \right)$$

We use the 0/1 values and row- and column-sum constraints on M to simplify this expression.

$$\Psi(D) = \prod_{l=1}^{L} \prod_{\{i_p \in \{1..i_{\max}\} \mid p \in \{1..l+1\}\}} \prod_{I} \Bigg\{ \left[G\left(\sum_J M_{J;\, l+1,\, (i_1 \,\ldots\, i_{l+1})}\, z_J \,\middle|\, z_I\,,\, \sigma_l \right) \right]^{M_{I;\, l,\, (i_1 \,\ldots\, i_l)} \sum_J M_{J;\, l+1,\, (i_1 \,\ldots\, i_{l+1})}\, \zeta_J}$$

$$\times\, Q\left(\left\{ \sum_J M_{J;\, l+1,\, (i_1 \,\ldots\, i_{l+1})}\, \hat{N}_J \right\} \,\middle|\, \hat{N}_I - \zeta_I \right)^{M_{I;\, l,\, (i_1 \,\ldots\, i_l)}}$$

$$\times\, \Theta\big(\zeta_I = 0 \Leftrightarrow \hat{N}_I = 0\big)^{M_{I;\, l,\, (i_1 \,\ldots\, i_l)}}\; r\big(m_I \mid \hat{N}_I\big)^{M_{I;\, l,\, (i_1 \,\ldots\, i_l)}} \Bigg\}$$

where in this expression we define $0^0 = 1$, so that a multiplicative factor of zero that does not enter into the product (due to the zero exponent) doesn't set the product to zero or otherwise affect it. Continuing,

$$\Psi(D) = \prod_{l=1}^{L} \prod_{\{i_p \in \{1..i_{\max}\} \mid p \in \{1..l\}\}} \prod_{s \in \{1..i_{\max}\}} \prod_{I} \Bigg\{ \Bigg(\prod_{\{J_s \mid 1 \leqslant s \leqslant m_I\}} [G(z_J \mid z_I\,,\, \sigma_l)]^{M_{I;\, l,\, (i_1 \,\ldots\, i_l)}\, M_{J_s;\, l+1,\, (i_1 \,\ldots\, i_l,s)}\, \zeta_J}$$

$$\times\, Q\big(\{ \hat{N}_{J_s} \mid 1 \leqslant s \leqslant m_I \} \,\big|\, \hat{N}_I - \zeta_I \big)^{M_{I;\, l,\, (i_1 \,\ldots\, i_l)} \prod_{\{J_s \mid 1 \leqslant s \leqslant m_I\}} M_{J_s;\, l+1,\, (i_1 \,\ldots\, i_l,s)}} \Bigg)$$

$$\times\, \Theta\big(\zeta_I = 0 \Leftrightarrow \hat{N}_I = 0\big)^{M_{I;\, l,\, (i_1 \,\ldots\, i_l)}}\; r\big(m_I \mid \hat{N}_I\big)^{M_{I;\, l,\, (i_1 \,\ldots\, i_l)}} \Bigg\}$$

Now interchange the products over I and over $(l,\ \{i\})$. Because of the row and column sum constraints on M, each $(l,\ \{i\})$ index value combination corresponds to at most one I or J index value. Thus

$$\Psi(D) = \prod_{I} \Bigg\{ \Bigg(\prod_{s \in \{1..i_{\max}\}} \prod_{\{J_s \mid 1 \leqslant s \leqslant m_I\}} [G(z_J \mid z_I\,,\, \sigma_l)]^{C_{IJs}\, \zeta_J}\; Q\big(\{ \hat{N}_{J_s} \mid 1 \leqslant s \leqslant m_I \} \,\big|\, \hat{N}_I - \zeta_I \big)^{\prod_{\{J_s \mid 1 \leqslant s \leqslant m_I\}} C_{IJs}} \Bigg)$$

$$\times\, \Theta\big(\zeta_I = 0 \Leftrightarrow \hat{N}_I = 0\big)^{\zeta_I}\; r\big(m_I \mid \hat{N}_I\big)^{\zeta_I} \Bigg\}$$

Since there are constraints $C_{IJs} \leqslant \zeta_J$ and $\zeta_I = 0 \Rightarrow \big(m_I = 0 \wedge \hat{N}_I = 0\big)$, and since $r(0 \mid 0) = 1$, this is equivalent to

$$\Psi(D) = \prod_{I} \Bigg\{ \Bigg(\prod_{s \in \{1..i_{\max}\}} \prod_{\{J_s \mid 1 \leqslant s \leqslant m_I\}} [G(z_J \mid z_I\,,\, \sigma_l)]^{C_{IJs}}\; Q\big(\{ \hat{N}_{J_s} \mid 1 \leqslant s \leqslant m_I \} \,\big|\, \hat{N}_I - \zeta_I \big)^{\prod_{\{J_s \mid 1 \leqslant s \leqslant m_I\}} C_{IJs}} \Bigg)$$

$$\times\, \Theta\big(\zeta_I = 0 \Leftrightarrow \hat{N}_I = 0\big)\, r\big(m_I \mid \hat{N}_I\big) \Bigg\} \tag{16}$$

This probability distribution, together with the constraints on (C, ζ) and $\big(\hat{N}, \zeta\big)$, is shown in the dependency diagram of Figure 9. Note that the constraint $\zeta_I = 0 \Rightarrow m_I = 0$ needn't be included explicitly since it follows from $\zeta_I = 0 \Rightarrow \hat{N}_I = 0$ and $r\big(m \mid \hat{N} = 0\big) = \delta_{m0}$.

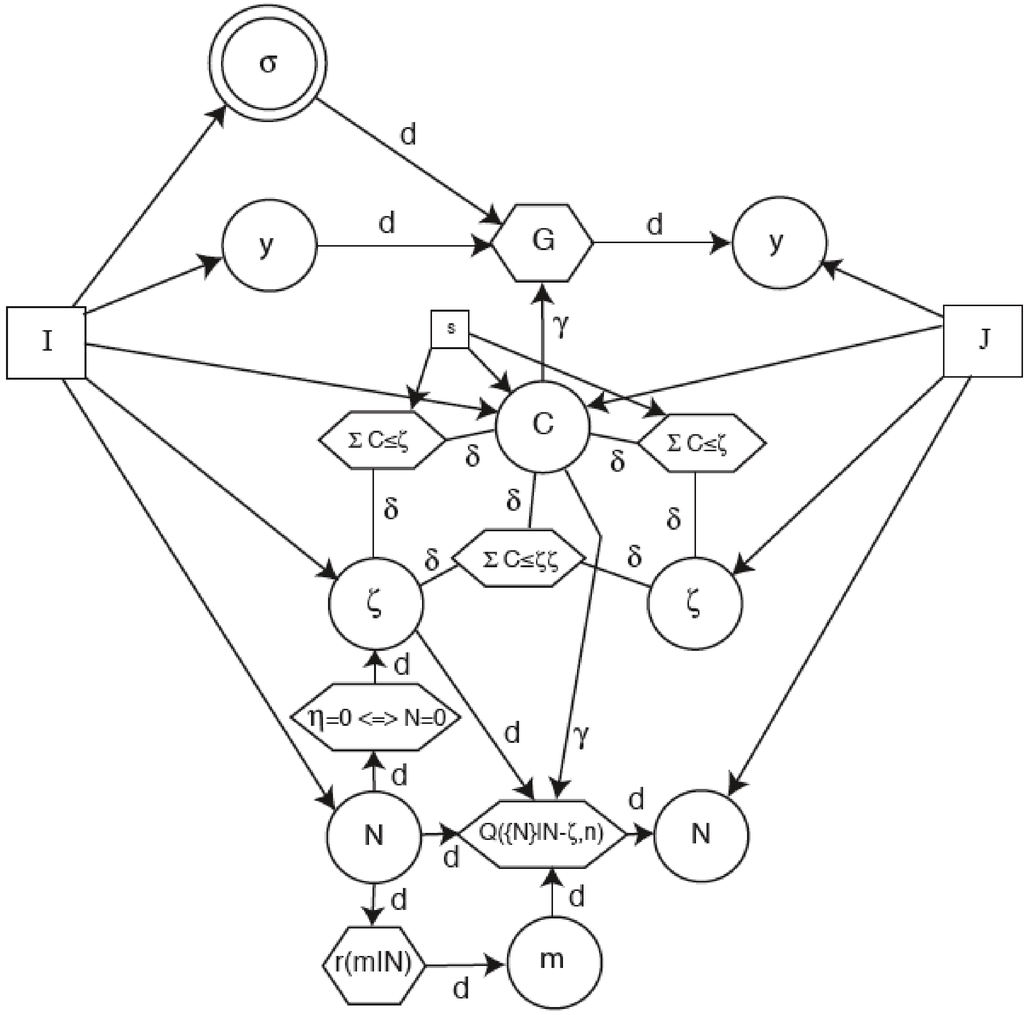


Figure 9: Efficient dependency diagram for resource-bounded context-free feature tree. Unlabelled links are of default type "ι" (cf. Table 3). Apply the expansion map $\mathcal{E}$ of Lemma 1 to obtain same-sized diagram without gating (γ). Apply the full expansion map $\mathcal{E}$ of Proposition 2 to obtain a diagram with exponentially fewer nodes than that obtained by applying $\mathcal{E}$ to the diagram of Figure 8, which denotes an equivalent probabilistic model. Cf. Proposition 3. Alternatively, Q could be replaced with a product of R factors (Equation 4).

This change of variables establishes a bijection of states between the $\{y, \eta, n, N, M\}$ and $\{z, \zeta, m, \hat{N}, C\}$ systems, preserving probabilities and conditional probabilities. Hence it is a reduction. It also establishes also a reduction from $\{y, \eta, n, N\}$ to $\{z, \zeta, m, \hat{N}, C\}$, but in principle we also must multiply by the entropy term $\exp(S(m, \hat{N})) = \exp(S(n, N)))$ which counts the number of equivalent configurations of the M variables . In this particular case, the equivalent configurations of M are just parameterized by permutations that reassign N_0 tree nodes to $I_{\max} \geqslant N_0$ memory locations, of which there are $(I_{\max})_{N_0} = (I_{\max})!/(I_{\max} - N_0)!$, which is a constant and hence gets normalized out of any probabilities. Thus, for this particular problem, there is no entropy effect to take into account in computing the probabilities in the new $\{z, \zeta, m, \hat{N}, C\}$ variables.

In this way, we establish a bidirectional mapping or *equivalence* between probabilistic model classes C and C'.

Thus we have established (see Section 3.5.5 for DD notation)

Proposition 3. There exists a nonempty model class C, including the context free cluster tree $\mathcal{T}(N, q, \phi)$, and a class of equivalent probabilistic models C', for which

$$\begin{gathered} C \subseteq \Psi(\mathrm{DD}(\{f,\ d,\ \gamma,\ \iota,\ \delta, \epsilon\},\ \ldots,\ \mathrm{const},\ N_{\mathrm{rv}})) \subseteq \Psi(\mathrm{DD}(\{f,\ d,\ \delta\},\ \ldots,\ N_{\mathrm{rv}})) \\ \wedge\ C \nsubseteq \Psi(\mathrm{DD}(\{f,\ d,\ \gamma,\ \iota,\ \delta, \epsilon\},\ \ldots,\ \mathrm{const},\ N_{\mathrm{rv}}-1)) \subseteq \Psi(\mathrm{DD}(\{f,\ d,\ \delta\},\ \ldots,\ N_{\mathrm{rv}}-1)) \\ \wedge\ \ C \sim C' \subseteq \Psi\big(\mathrm{DD}\big(\{f,\ d,\ \gamma,\ \iota,\ \delta\},\ \ldots,\ \mathrm{const},\ \tilde{N}_{\mathrm{rv}}\big)\big) \subseteq \Psi\big(\mathrm{DD}\big(\{f,\ d,\ \delta\},\ \ldots,\ \tilde{N}_{\mathrm{rv}}\big)\big) \end{gathered}$$

with N_{rv}, which is minimal for class C, such that $N_{\mathrm{rv}} = O(2^N)$ and $\tilde{N}_{\mathrm{rv}} = O(N^2) = O\big(\log(N_{\mathrm{rv}})^2\big)$.

This proposition provides evidence that the sparse activation networks that can be defined with $\gamma,\ \iota,\ \delta$ links are essentially different from other network classes. Such links can be eliminated uniformly, or efficiently, but so far not both uniformly and efficiently.

3.9 Variable-Structure System Examples

Object Representation. With these diagrammatic notations it is straightforward to express novel network architectures and structures [1] for networks of “objects” with contingent existence, variable interrelationships, and a schema of types such as those of [12] or [3]; see also [4]. The basic idea is to use one index (e.g. i or j) to describe instances of an object of a given type and their relationships as determined by a sparse matrix $G_{ij} \in \{0, 1\}$. A second index (e.g. a or b) is the internal index for a vector of parameters describing an object; these variables interact (or not) between G-related instance according to another matrix $g_{ab} \in \{0, 1\}$. If there are many types of objects, it takes a third kind of index (α or β) to index the type and there is a “schema” $S_{\alpha\sigma\beta} = \mathrm{INA}_{\alpha\sigma\beta} \in \{0, 1\}$ whereby an object of type β may occupy slot σ of an object of type α. Two individual real-valued variables x and y will interact only if all three interaction conditions are satisfied. All three matrices g, G, and S gate the interaction. Ignoring for a moment the schema index, and making explicit the internal index a or b, Figure 10 shows the resulting DD.

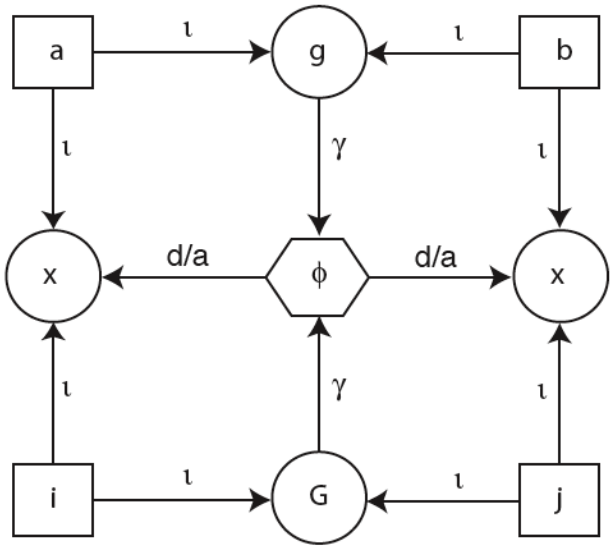
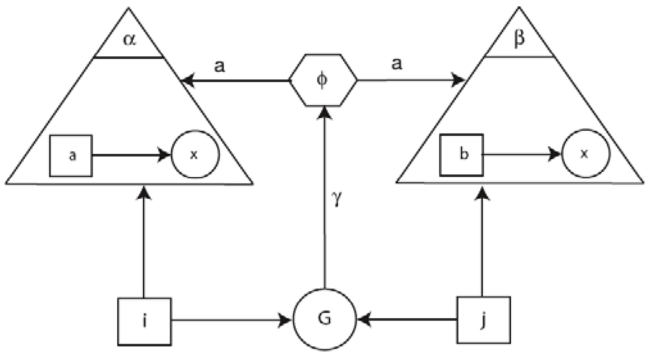
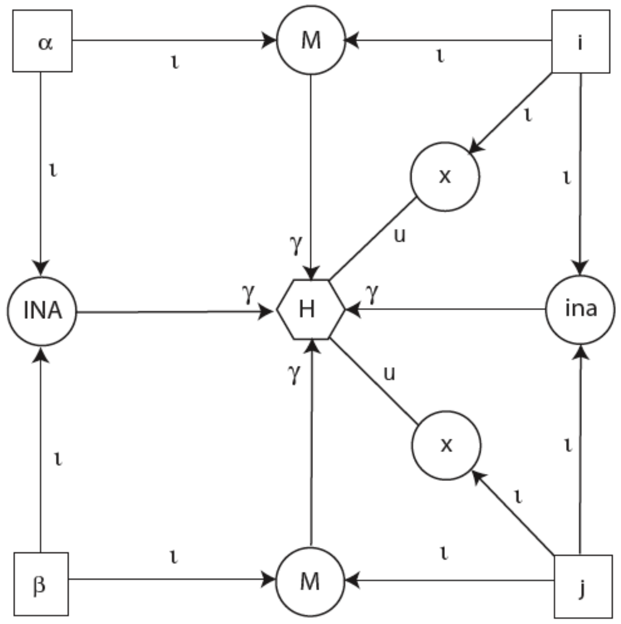


Figure 10: Diagram for gating of interaction by both object instance relationships, G, and internal object attribute relationships, g.



Subdiagrams can be used to encapsulate parameter vector (*a,b*) structure, instance replication (*i,j*) structure, and instance relatedness (*G*) structure, in the dependency diagram for objects containing attribute variables *x*. In Figure 11, subdiagrams are outlined by triangles and labelled with object type labels (α and β) . The basic object of type α, with attribute vectors *x* whose components are indexed by *a*, can be drawn 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, a, and α. This diagram is consistent with the previous notation for Frameville networks [Anandan et al. 1989; Mjolsness 1997] governed by Boltzmann energy functions, except that we have so far omitted "class" composition information whereby an object of class α can contain objects of class β via compositional "ina" links, and also the subtype "ISA" relationships between object classes. The compositional structure is restored by the Frameville energy function (and corresponding Boltzmann distribution) shown below in algebraic and diagrammatic form, supressing the internal vector indices *a* and *b*.



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

4 Semantics for Dynamical Grammars

Context-sensitive stochastic parameterized grammars are a substantial generalization from context-free ones. Here we provide the general formalism in both continuous and discrete time execution models, relate the two, and show they specialize trivially to the context-free cluster tree.

This same type of composition is the essential idea for the composition of stochastic processes representing chemical reaction networks: take the union of all chemical reaction nodes and links in two networks, identifying reactant nodes that "mean" the same thing in the two different networks, to get a new reaction/reactant graph. The same composition method works for stochastic parameterized grammars that generalize chemical reaction networks.

Unlike Dependency Diagrams, parameterized Dynamical Grammars are intrinsically dynamical and variable-structure models. One way to unify the two concepts is to formulate Dynamical Grammars whose rules each have a local conditional probability density given by a Dependency Diagram. In this way we include both Boltzmann distributions [4] and, in the limit $\Delta t \to 0$, differential equation systems [20-21] as elementary dynamical systems underlying each grammar rule.

4.1 Syntax for stochastic parameterized grammars

The grammar syntax

$$\{\tau_j(x_j)\} \to \{\tau_i(x_i')\} \text{ with } \rho_r \Pr(\{x_i'\} \mid \{x_j\}) \tag{17}$$

is a generalization of chemical reaction network syntax

$$\left\{m_i^{(r)} A_i\right\} \xrightarrow{k^{(r)}} \left\{n_i^{(r)} A_i\right\}$$

where parameters x, x' have been added and explicit stoichiometries m, n removed.

4.2 Operator algebra semantics for discrete-event, continuous-time rules

Given a continuous-time stochastic parameterized grammar rule (rule number r) of the form

$$\{\tau_j(x_j)\} \to \{\tau_i(x_i')\} \text{ with } \rho_r \Pr(\{x_i'\} \mid \{x_j\}), \tag{18}$$

we can write a *generator* for the transition probability rate of this rule. (Note that the term numberings i and j are arbitrary, so long as different terms get different index numbers.) A *generator* is an operator that advances the state of a system by an infinitiesimal amount, here an amount of time in a dynamical system. For non-negative integer valued variables we can create suitable generators if we first define the basic "creation operator" $\hat{a}$ and "annihilation operator" a:

$$\hat{a} = \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots \\ 1 & 0 & 0 & 0 & \\ 0 & 1 & 0 & 0 & \\ 0 & 0 & 1 & 0 & \\ \vdots & & & \ddots & \ddots \end{pmatrix} = \delta_{n,m+1} \text{ and } a = \begin{pmatrix} 0 & 1 & 0 & 0 & \cdots \\ 0 & 0 & 2 & 0 & \\ 0 & 0 & 0 & 3 & \\ 0 & 0 & 0 & 0 & \ddots \\ \vdots & & & & \ddots \end{pmatrix} = m\,\delta_{n+1,m} \,, \tag{19}$$

which have an algebra defined by the "commutator" $[a,\ \hat{a}]_{n\,m} \equiv (a\ \ \hat{a} - \ \hat{a}\,a\,)_{n\,m} = \delta_{n,m}$, i.e.

$$[a,\ \hat{a}] \equiv (a\ \hat{a} - \ \hat{a}\,a\,) = I = \begin{pmatrix} 1 & 0 & 0 & 0 & \cdots \\ 0 & 1 & 0 & 0 & \\ 0 & 0 & 1 & 0 & \\ 0 & 0 & 0 & 1 & \\ \vdots & & & & \ddots \end{pmatrix};\ \hat{a}\,a\ = N_a \equiv \begin{pmatrix} 0 & 0 & 0 & 0 & \cdots \\ 0 & 1 & 0 & 0 & \\ 0 & 0 & 2 & 0 & \\ 0 & 0 & 0 & 3 & \\ \vdots & & & & \ddots \end{pmatrix}. \tag{20}$$

This is a statistical rather than the usual quantum representation of the Heisenberg algebra $[a,\ \hat{a}] = I$, which has no finite-dimensional representations. (Nevertheless, we will see that conserved or nonincreasing resources can be introduced to ensure that only a finite number of dimensions ever receive a nonzero probability.) For 0/1 binary valued variables, use instead the operators

$$a' = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix},\ a = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}$$
$$\{a,\ a'\} = a\,a' + a'\ a = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = I\,;\ a'\ a\ = N_a \equiv \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}$$

Either way, operators corresponding to different grammar terms commute, as do all powers of identical operators:

$$[a, a] = [a', a'] = [a, b] = [a', b'] = [O^n, O^m] = 0$$

By truncating the infinite-dimensional matrices to finite size $n^{(\max)} < \infty$ we may compute that for some polynomial $Q(N,\ n^{(\max)})$ of degree $n^{(\max)}$ -1 with rational coefficients,

$$[a,\ \hat{a}] = I + N\,Q\big(N,\ n^{(\max)}\big).$$

Eg. if $n^{(\max)}$ =1 then $Q = -2$. If the parameters x are continuous e.g. real-valued, then the general commutator relation is

$$[a(x),\ \hat{a}(y)] = \delta(x-y)\big[I + N\,Q\big(N,\ n^{(\max)}\big)\big]$$

where δ is the Dirac delta (generalized) function.

For the syntactic form of the grammar of Equation 17, each *term(value)* combination $\tau_j(x_j)$ is assigned its own $(a(\tau_j, x_j),\ \hat{a}(\tau_i, x_j))$ operator pair that annihilate or create a parameterized term $\tau_j(x_j)$ with parameter value x_j. Each rule, r, is assigned its own "generator" operator for time evolution, O_r, by the technique of simultaneously annihilating all terms on the left hand side and creating all terms on the right hand side of the rule. For the very special case in which there are no parameters, the resulting operator is

$$\begin{array}{c} \hat{O}_r\ =\ \rho_r\left[\prod_{i\in \text{rhs}(r)} \hat{a}(\tau_i)\right]\ \left[\prod_{j\in \text{lhs}(r)} a(\tau_j)\right] \\ O_r = \hat{O}_r - \text{diag}\big(\mathbf{1}^T \cdot \hat{O}_r\big) \end{array}.$$

The diagonal term enforces conservation of probability, $\mathbf{1}^T \cdot O_r = 0$. The product over j makes the grammar context-sensitive. In the presence of parameters, the rule generator $\hat{O}_r$ is obtained by summing the creation-annihilation product over all possible values of the term parameters. Therefore each rule r receives the following operator O_r:

$$\begin{gathered} \hat{O}_r = \sum_{\{x'_i, x_j\}} \rho_r(\{x_j\})\, \phi_r(\{x'_i\} \mid \{x_j\}) \left[\prod_{i \in \mathrm{rhs}(r)} \hat{a}(\tau_i, x_i')\right] \left[\prod_{j \in \mathrm{lhs}(r)} a(\tau_j, x_j)\right] \\ O_r = \hat{O}_r - \mathrm{diag}\left(\mathbf{1}^T \cdot \hat{O}_r\right). \end{gathered} \tag{21}$$

The rule generators can be summed over rules to get the time evolution operator of the whole grammar. Summing the rule-specific generators, we define the Hamiltonian:

$$H = \sum_r O_r = \hat{H} - D = \sum_r \hat{O}_r - \sum_r \mathrm{diag}\left(\mathbf{1}^T \cdot \tilde{O}_r\right). \tag{22}$$

This then determines the *Master Equation* [22]: if we let $\Pr(\{n\};\ t \mid \{m\};\ 0)$ denote the probability of having n particles or terms of each possible type and state after time t, starting from m particles or terms of each possible type and state at time zero, then

$$\begin{gathered} \frac{d}{dt} \Pr(\{n\};\ t \mid \{m\};\ 0) = \sum_{\{p\}} H_{\{n\}\{p\}} \Pr(\{p\};\ t \mid \{m\};\ 0), \quad \text{i.e. in matrix notation} \\ \frac{d}{dt} \Pr(t \mid 0) = H \Pr(t \mid 0) \end{gathered} \tag{23}$$

with initial condition

$$\Pr(\{n\};\ 0 \mid \{m\};\ 0) = \delta_{\{n\},\{m\}}. \tag{24}$$

This system has the following abstract solution in matrix notation, which can be taken to define the semantics of our grammar:

$$\Pr(t \mid 0) = \exp(t H) \ , \tag{25}$$

which implies

$$\Pr(t) = \exp(t H) \cdot \Pr(0). \tag{26}$$

Recently, the form $\exp\left(t\left(\hat{H} - D\right)\right)$ has also been used to define the “Diffusion Kernel” on graphs [23]; here we may interpret $\hat{H}$ as a graph of allowed state transitions weighted by their rates.

4.2.1 Operator exponential

The meaning of the operator exponential is given by the Taylor series expansion for the exponential, or more generally by the Trotter product formula as follows:

$$\begin{aligned} \exp[t(H_0 + H_1)] &= \lim_{n\to\infty} \left[I + \frac{t}{n}(H_0 + H_1)\right]^n \\ &= \lim_{n\to\infty} \left[\left(I + \frac{t}{n} H_0\right)\left(I + \frac{t}{n} H_1\right)\right]^n . \end{aligned}$$

Then the abstract operator solution of the master equation can be taken to define the semantics $\Psi_c\big(H\big(\hat{H}(\Gamma)\big)\big) = \exp(t\,H)\cdot \Pr(0)$ of the continuous-time grammar Γ. This establishes part (a) of Proposition 4 of Section 4.4 below.

4.2.2 Chapman-Kolmogorov equation

We now verify this system satisfies the postulates of a stochastic process in continuous time. We assume that time is continuous. We also assume it is homogeneous, so that the starting time doesn't matter. This is expressed in two equations. For $t \geqslant t'$:

$$\Pr(n_1,\ \ldots\ n_k;\ t\,|\,m_1, m_k;\ t') = \Pr(n_1,\ \ldots\ n_k;\ t - t'\,|\,m_1,\ \ldots\ m_k;\ 0)\,.$$

The second equation is the Chapman-Kolmogorov equation for stochastic processes, which says that evolving forward from time t to time t' and then from time t' to time t'' through all possible intermediate states at time t' is the same as evolving forward from the initial to the final time $(t \leqslant t' \leqslant t'')$:

$$\Pr(n_1,\ \ldots\ n_k;\ t''\,|\,m_1,\ \ldots\ m_k;\ t) = \sum_{\{p(1),\ldots\ p(k)\}} \Pr(n_1,\ \ldots\ n_k; t''\,|\,p_1,\ \ldots\ p_k;\ t')\Pr(p_1,\ \ldots\ p_k;\ t'\,|\,m_1,\ \ldots\ m_k;\ t) \qquad (27)$$

Note that the Chapman-Kolmogorov equation automatically follows from the abstract solution, since even for matrices, multiples of H commute and therefore $\exp[(t'' - t')\,H]\exp[(t' - t)\,H] = \exp[(t'' - t)\,H]$.

4.3 Discrete-time SPG's

The state of the discrete-time grammar after n rule firing steps is the normalized version of $\hat{H}^n \cdot p_0$:

$$c_n\,\hat{H}^n \cdot p_0 = \big(\hat{H}^n \cdot p_0\big)\big/\big(\mathbf{1}\cdot \hat{H}^n \cdot p_0\big)$$

This depends on a normalization constant $c_n = 1\big/\big(\mathbf{1}\cdot\hat{H}^n \cdot p_0\big)$. For unbounded operators of infinite dimension this criterion can be state-dependent and hence dependent on n, so $c_n \neq c^n$. This is a critical distinction between stochastic grammar and Markov chain models, for which $c_n = c^n$.

Example: For the unbounded context-free unlabelled tree $\mathcal{T}(q, \phi)$,

$$\hat{H} = \sum_{k=0}^{\infty} q_k\,\hat{a}^k\,a = g(\hat{a})\,a;\ \ H = g(\hat{a})\,a - N$$

$$\hat{H}^2 = g(\hat{a})^2\,a^2 + g(\hat{a})\,g'(\hat{a})\,a;\ \ \mathbf{1}\cdot\hat{H}^2 = (1 + g'(1))\,\mathbf{1}\cdot a = (1 + g'(1))\,\boldsymbol{N}$$

$$\hat{H}^3 = g(\hat{a})^3\,a^3 + 3\,(g(\hat{a}))^2\,g'(\hat{a})\,a^2 + g(\hat{a})\,(g'(\hat{a}))^2\,a + (g(\hat{a}))^2\,g''(\hat{a})\,a;\ \ \mathbf{1}\cdot 3 = \big(1 + 3\,g'(1) + g'(1)^2 + g''(1)\big)\boldsymbol{N}$$

Elaborations of this example include: graph node labels x (real-valued or discrete-valued):

$$H = \sum_{n=0}^{\infty} q_n \int_{-\infty}^{\infty} d\,x_1\ \phi(x_1\,|\,y)\,\hat{a}(x_1)\cdots\int_{-\infty}^{\infty} d\,x_n\ \phi(x_n\,|\,y)\,\hat{a}(x_n)\int_{-\infty}^{\infty} d\,y\,a(y) - \int_{-\infty}^{\infty} d\,y\,N_{a(y)}$$

and/or resource limitations via a countdown operator:

$$H = g(b\,\hat{a})\,a - N_a$$

Here the transposed operator b makes the grammar context-sensitive but provides a uniform-rate “countdown” from m_b initially available event tokesn to zero, whereupon the grammar must stop due to lack of available event tokens. This provides an implementation of the resource-limited cluster tree structure. Both concepts can be combined to yield another operator algebra version of the resource-limited feature tree $\mathcal{T}(N, q, \phi)$:

$$\hat{H}(q, \phi) = \sum_{n=0}^{\infty} q_n\, b^n \int_{-\infty}^{\infty} d x_1\, \phi(x_1 \mid y)\, \hat{a}(x_1) \cdots \int_{-\infty}^{\infty} d x_n\, \phi(x_n \mid y)\, \hat{a}(x_n) \int_{-\infty}^{\infty} d y\, a(y),$$

$$p_0 = \delta(m_b, N)$$

4.4 Relation of discrete-time and continuous-time grammars

The continuous and discrete-time grammar executions are related as follows. After continuous time t, the joint probability density on the states of the original system and on the number of discrete rule firings, n, has the generating function

$$S(z) = \sum_{n=0}^{\infty} s_n\, z^n = \exp\left(t\left(\hat{H}\, z - D\right)\right) \cdot p_0$$

so that

$$s_n = \operatorname{Coef}_n\left(\exp\left(t\left(\hat{H}\, z - D\right)\right), z\right) \cdot p_0 .$$

An alternative approach to the semantics of the discrete-time grammar is to take the short-time limit of the continuous-time grammar’s conditional distribution given that n rule firings occurred:

$$\lim_{t\to 0} \left[s_n / 1 \cdot s_n \right] = \lim_{t\to 0} \left[\operatorname{Coef}_n\left(\exp\left(t\left(\hat{H}\, z - D\right)\right), z\right) \cdot p_0 / 1 \cdot \operatorname{Coef}_n\left(\exp\left(t\left(\hat{H}\, z - D\right)\right), z\right) \cdot p_0\right].$$

This result follows by a short calculation from the following general expression for S:

$$S(z) = \sum_{n=0}^{\infty} s_n\, z^n = \exp\left(t\left(\hat{H}\, z - D\right)\right) \cdot p_0$$

$$= \sum_{n=0}^{\infty} \frac{z^n}{n!} \left[\partial_z{}^n \exp\left(t\left(\hat{H}\, z - D\right)\right)\right]_{z=0} \cdot p_0$$

$$= \sum_{n=0}^{\infty} \frac{z^n}{n!} \left[\partial_z{}^n \sum_{k=0}^{\infty} \frac{\left(t\left(\hat{H}\, z - D\right)\right)^k}{k!}\right]_{z=0} \cdot p_0$$

$$= \sum_{n=0}^{\infty} \frac{z^n}{n!} \left[\sum_{k=n}^{\infty} \frac{1}{k!} \sum_{\{0 \leq i_p \leq k-n\} \wedge \sum_{p=0}^{n} i_p = k-n} n!\, (-t\, D)^{i_n}\, t\, \hat{H} (-t\, D)^{i_{n-1}} \cdots t\, \hat{H}\, (-t\, D)^{i_0}\right] \cdot p_0$$

$$= \sum_{n=0}^{\infty} z^n \left[\sum_{k=0}^{\infty} \frac{1}{(k+n)!} \sum_{\{0 \leq i_p \leq k\} \wedge \sum_{p=0}^{n} i_p = k} t^n (-t\, D)^{i_n}\ \hat{H} (-t\, D)^{i_{n-1}} \cdots \hat{H}\, (-t\, D)^{i_0}\right] \cdot p_0$$

From this expression we can take the small-time limit, picking out only the $i_p = 0$ terms:

$$\lim_{t\to 0} S(z) = \sum_{n=0}^{\infty} z^n \left[\frac{1}{(k+n)!} \Big|_{(k=0)} t^n \, \hat{H}^n \right] \cdot p_0 = \sum_{n=0}^{\infty} \frac{z^n \, t^n}{n!} \hat{H}^n \cdot p_0$$

Thus

$$\lim_{t\to 0} \left[s_n / 1 \cdot s_n \right] = \hat{H}^n \cdot p_0 \Big/ \left(1 \cdot \hat{H}^n \cdot p_0 \right)$$

If the denominator happens to be a power of n, which in general it is not, then we specialize to a discrete-time Markov process as a function of the number of rule firings, n.

Thus we have established:

Proposition 4. Given the stochastic parameterized grammar (SPG) rule syntax of Equation 17,

(a) There is a semantic function Ψ_c mapping from any continuous-time, context sensitive, stochastic parameterized grammar Γ via a time evolution operator $H(\hat{H}(\Gamma))$ to a joint probability density function on the parameter values and birth/death times of grammar terms, conditioned on the total elapsed time, t.

(b) There is a semantic function Ψ_d mapping any discrete-time, sequential-firing, context sensitive, stochastic parameterized grammar Γ via a time evolution operator $\hat{H}(\Gamma)$ to a joint probability density function on the parameter values and birth/death times of grammar terms, conditioned on the total discrete time defined as number of rule firings, n.

(c) The short-time limit of the density $\Psi_c(\Gamma)$ conditioned on $t \to 0$ and conditioned on n is equal to $\Psi_d(\Gamma)$.

(d) There is a serial context-free grammar Γ_{tree} whose asymptotic probability distribution is that of the context-free feature tree $\mathcal{T}(q, \phi)$, and another context-free grammar $\Gamma_{\text{rl-tree}}$ whose asymptotic probability distribution is that of the resource-limited context-free feature tree $\mathcal{T}(N, q, \phi)$.

Corollary. The following diagram commutes:

$$\begin{tikzcd}
 & \hat{H}(\Gamma) \arrow[dl, "\Psi_c"'] \arrow[dr, "\Psi_d"] & \\
\mathcal{P}(H(\Gamma) \mid t, n) \arrow[rr, "(\Delta t \to 0)"] & & \mathcal{P}(\hat{H}(\Gamma) \mid n)
\end{tikzcd}$$

4.5 Perturbative expansion for stochastic field theory

A fundamental method for deriving simulation algorithms can be borrowed from quantum field theory and transposed to stochastic processes. The version given here is derived probabilistically and has the advantage of being recursively self-applicable.

4.5.1 Time-ordered operator expansion

The general expansion formula for S is given by the time-ordered product (Equation 2.14 of [24]equation 4.29[24][25]) which we can derive by elementary probabilistic means as follows.

We obtain extra insight by continuing the calculation of $S(z)$ from the previous section:

$$= \sum_{n=0}^{\infty} z^n\, t^n \left[\sum_{k=0}^{\infty} \sum_{\{0 \leqslant i_p \leqslant k\} \wedge \sum_{p=0}^{n} i_p = k} \frac{\prod_{p=0}^{n} (i_p)!}{\left(\sum_{p=0}^{n} i_p + n\right)!} \frac{(-t\, D)^{i_n}}{(i_n)!} \hat{H} \frac{(-t\, D)^{i_{n-1}}}{(i_{n-1})!} \cdots \hat{H} \frac{(-t\, D)^{i_0}}{(i_0)!} \right] \cdot p_0$$

$$= \sum_{n=0}^{\infty} z^n\, t^n \left[\sum_{\{0 \leqslant i_p \leqslant \infty\}} \frac{\prod_{p=0}^{n} (i_p)!}{\left(\sum_{p=0}^{n} (\, i_p + 1) - 1\right)!} \frac{(-t\, D)^{i_n}}{(i_n)!} \hat{H} \frac{(-t\, D)^{i_{n-1}}}{(i_{n-1})!} \cdots \hat{H} \frac{(-t\, D)^{i_0}}{(i_0)!} \right] \cdot p_0$$

$$= \sum_{n=0}^{\infty} z^n\, t^n \left[\sum_{\{0 \leqslant i_p \leqslant \infty\}} \frac{\prod_{p=0}^{n} \Gamma(i_p + 1)}{\Gamma\left(\sum_{p=0}^{n} (\, i_p + 1)\right)} \frac{(-t\, D)^{i_n}}{(i_n)!} \hat{H} \frac{(-t\, D)^{i_{n-1}}}{(i_{n-1})!} \cdots \hat{H} \frac{(-t\, D)^{i_0}}{(i_0)!} \right] \cdot p_0$$

Now we use the Multinomial-Dirichlet normalization integral

$$\frac{\prod_{p=0}^{n} \Gamma(i_p + 1)}{\Gamma\left(\sum_{p=0}^{n} (\, i_p + 1)\right)} = \int_0^1 d\theta_0 \cdots \int_0^1 d\theta_n\, \delta\!\left(\sum_{i=1}^{n} \theta_p - 1 \right) \prod_{p=0}^{n} (\theta_p)^{i_p} \,.$$

Accordingly,

$$S(z) =$$

$$\sum_{n=0}^{\infty} z^n\ \ t^n \left[\sum_{\{0 \leqslant i_p \leqslant \infty\}} \int_0^1 d\theta_0 \cdots \int_0^1 d\theta_n\, \delta\!\left(\sum_{i=1}^{n} \theta_p - 1 \right) \left(\prod_{p=0}^{n} (\theta_p)^{i_p} \right) \frac{(-t\, D)^{i_n}}{(i_n)!} \hat{H} \frac{(-t\, D)^{i_{n-1}}}{(i_{n-1})!} \cdots \hat{H} \frac{(-t\, D)^{i_0}}{(i_0)!} \right] \cdot p_0$$

$$= \sum_{n=0}^{\infty} z^n\ \ t^n \left[\sum_{\{0 \leqslant i_p \leqslant \infty\}} \int_0^1 d\theta_0 \cdots \int_0^1 d\theta_n\, \delta\!\left(\sum_{i=1}^{n} \theta_p - 1 \right) \frac{(-\theta_n\, t\, D)^{i_n}}{(i_n)!} \hat{H} \frac{(-\theta_{n-1}\, t\, D)^{i_{n-1}}}{(i_{n-1})!} \cdots \hat{H} \frac{(-\theta_0\, t\, D)^{i_0}}{(i_0)!} \right] \cdot p_0$$

$$= \sum_{n=0}^{\infty} z^n \; t^n \left[\int_0^1 d\theta_0 \cdots \int_0^1 d\theta_n \; \delta\left(\sum_{i=1}^{n} \theta_p - 1 \right) \right.$$

$$\left. \sum_{\{0 \leq i_0 \leq \infty\}} \frac{(-\theta_n \, t\, D)^{i_0}}{(i_n)!} \hat{H} \sum_{\{0 \leq i_1 \leq \infty\}} \frac{(-\theta_{n-1} \, t\, D)^{i_1}}{(i_{n-1})!} \cdots \hat{H} \sum_{\{0 \leq i_n \leq \infty\}} \frac{(-\theta_0 \, t\, D)^{i_n}}{(i_0)!} \right] \cdot p_0$$

$$= \sum_{n=0}^{\infty} z^n \; t^n \left[\int_0^1 d\theta_0 \cdots \int_0^1 d\theta_n \; \delta\left(\sum_{i=1}^{n} \theta_p - 1 \right) \exp(-\theta_n \, t\, D) \, \hat{H} \exp(-\theta_{n-1} \, t\, D) \cdots \hat{H} \exp(-\theta_0 \, t\, D) \right] \cdot p_0$$

$$= \sum_{n=0}^{\infty} z^n \left[\int_0^t d\tau_0 \cdots \int_0^t d\tau_n \; \delta\left(\sum_{i=1}^{n} \tau_p - t \right) \exp(-\tau_n \, D) \, \hat{H} \exp(-\tau_{n-1} \, D) \cdots \hat{H} \exp(-\tau_0 \, D) \right] \cdot p_0$$

In summary (since p_0 was never used in the above calculations),

$$\exp\left(t \left(\hat{H} - D\right)\right)$$

$$= \sum_{n=0}^{\infty} \left[\int_0^t d\tau_0 \cdots \int_0^t d\tau_n \; \delta\left(\sum_{i=1}^{n} \tau_p - t \right) \exp(-\tau_n \, D) \, \hat{H} \exp(-\tau_{n-1} \, D) \cdots \hat{H} \exp(-\tau_0 \, D) \right] .$$

Alternatively, define $t_1 = \tau_0$, $t_2 = t_1 + \tau_1$, ... $t_{n+1} = t_n + \tau_n = t$. Then the evolution of the state vector is given by

$$\exp\left(t \left(\hat{H} - D\right)\right) \cdot p_0 \; =$$

$$\sum_{n=0}^{\infty} \left[\int_0^t d t_1 \int_{t_1}^t d t_2 \cdots \int_{t_{n-1}}^t d t_n \exp(-(t - t_n) \, D) \, \hat{H} \exp(-(t_n - t_{n-1}) \, D) \cdots \hat{H} \exp(-t_1 \, D) \right] \cdot p_0$$

Since D is diagonal, the terms $\exp(-\tau\, D)$ are analytically calculable and easy to simulate with large jumps in time. Between these easy terms are interposed single powers of $\hat{H}$ representing the occurrence of discrete-time grammar events that must be simulated.

These last two expression for $\exp\left(t \left(\hat{H} - D\right)\right)$ have a significant interpretation in the case of reaction kinetics: they correspond to the Gillespie algorithm for stochastic simulation. The exponential distribution of waiting times until the next reaction is given by $\exp(-\tau\, D)$, which depends on the state of the system but doesn't change it, and the reaction events are modeled by the interdigitated powers of $\hat{H}$.

This perturbative approach is equivalent to the use of perturbative methods including Feynman diagram calculations in quantum field theory, except for an occasional factor of $\sqrt{-1}$ which would turn our probabilities into the complex-valued probability factors of quantum mechanics. It can be accomplished for any decomposition of H into a solvable part H_0 (here, $-D$) plus a more difficult term H_1 (here, $\hat{H}$):

$$\exp(t\,(H_0 + H_1)) \cdot p_0 \;=\; \sum_{n=0}^{\infty} \left[\int_0^t d t_1 \int_{t_1}^t d t_2 \cdots \int_{t_{n-1}}^t d t_n \exp((t - t_n)\, H_0)\, H_1 \exp((t_n - t_{n-1})\, H_0) \cdots H_1 \exp(t_1\, H_0)\right] \cdot p_0 \tag{28}$$

Another decomposition of H, enabling another application of this expansion will be described in Section 5.1. This form of the time-ordered product expansion is recursively self-applicable.

4.6 Standard stochastic processes with operator algebra

Using the above operator algebra formalism we can express and solve the following standard continuous-time stochastic processes. These processes illustrate very simple dynamical grammars. Solutions are expressed as generating functions, from which any moment can be calculated. Each parenthesised term in a Hamiltonian corresponds to a grammar rule in the foregoing formalism.

4.6.1 Solution methods for the master equation

An equivalent representation of the annihilation and creation operators is given by their respective effects on a generating function $G(z)$, with one symbolic variable $z_{(\tau,x)}$ for each allowed term type and parameter combination:

$$G(z) = \sum_{\{n(\tau,x)=0\}}^{\infty} \Pr_{\{n(\tau,x)\}} \prod_{\{(\tau,x)\}} (z_{(\tau,x)})^{n(\tau,x)} = \sum_{\{n(\tau,x)=0\}}^{\infty} \Pr_{\{n(\tau,x)\}} \exp\left[\sum_{\tau} \int d x\, n(\tau, x)\, \mu(\tau, x)\right]. \tag{29}$$

The variable z corresponds to the "fugacity" for creation of instances of a term $\tau(x)$ in the grand canonical ensemble in statistical mechanics, and $\mu(\tau, x) \equiv \log z(\tau, x)$ is the corresponding "chemical potential" . Then we map

$$a_{z(\tau,x)} \mapsto \partial_{z(\tau,x)},\;\; \hat{a}_{z(\tau,x)} \mapsto z(\tau, x),\;\text{ and } [a_{z(\tau,x)}, \hat{a}_{z(\tau',x')}] = \delta_{\tau\,\tau'}\, \delta(x - x')\,. \tag{30}$$

The most involved part of this translation is just subtracting the diagonal probability-balance term, to go from $\tilde{O}_r$ to O_r. In this representation, the "Master equation" is first-order *linear* partial differential equation in many variables. In addition, boundary conditions are linear in G, second- and higher-order correlations are expectations computed as linear operators acting on G, and separation of time and space factors in the PDE solution gives a further linear analysis (analogous to an eigenvalue problem) of G.

4.6.2 Solvable Examples

Exponential decay of a single particle: Hamiltonian $H = \begin{pmatrix} 0 & \lambda \\ 0 & -\lambda \end{pmatrix} = \lambda\,(a - N)_{2\times 2}$, in which the creation and annihilation operators have been projected into the two-dimensional subspace corresponding to presence of either zero or one copies of a particle. The generating function is $G_0 = 1$, $G_1 = 1 - e^{-\lambda t} + e^{-\lambda t} z$.

Poisson process: Hamiltonian $H = \rho\,(\hat{a} - I)$. Equivalent to a single chemical reaction denoting synthesis, $\left\{\varnothing \xrightarrow{\rho_i} \mathrm{A}\right\}$. Generating function starting from m-particle initial condition:

$$G_m(z,t) = z^m \exp(\rho(z-1)t) = e^{-\rho t}\sum_{n=0}^{\infty}\frac{(\rho t)^n}{n!} z^{n+m}$$

Population *decay* process: $H = \rho\,(a - N)$. Equivalent to a single chemical reaction denoting decay or uncatalysed degradation $\left\{\mathrm{A}\xrightarrow{\rho_\mathrm{d}}\varnothing\right\}$. Generating function:

$$G_m(z,t) = ((z-1)\,e^{-\rho t}+1)^m = \sum_{n=0}^{m}\binom{m}{n} e^{-n\rho t}(1-e^{-\rho t})^{m-n}\,z^n,\ \ n \sim \text{Binomial}(m, e^{-\rho t}).$$

(This can be calculated from the PDE in (z, t) or seen from the single-particle decay generating function.)

Galton-Watson birth-death process, equivalent to a chemical reaction network with decay and a "chain reaction" $\left\{\mathrm{A}\xrightarrow{\rho_\mathrm{b}} 2\,\mathrm{A},\ \mathrm{A}\xrightarrow{\rho_\mathrm{d}}\varnothing\right\}$:

$$H = \rho_b\left(\hat{a}^2\,a - N\right) + \rho_d\,(a-N)$$

$$G_m(z,t) = \left(\frac{(\rho_b - \rho_d\,e^{(\rho_b-\rho_d)t})\,z + \rho_d(e^{(\rho_b-\rho_d)t}-1)}{\rho_b(1-e^{(\rho_b-\rho_d)t})\,z + (\rho_b\,e^{(\rho_b-\rho_d)t}-\rho_d)}\right)^m.$$

Galton-Watson birth-death process *with immigration*, equivalent to a chemical reaction network with synthesis, decay, and a "chain reaction" $\left\{\mathrm{A}\xrightarrow{\rho_\mathrm{b}} 2\,\mathrm{A},\ \varnothing\xrightarrow{\rho_\mathrm{i}}\mathrm{A},\ \mathrm{A}\xrightarrow{\rho_\mathrm{d}}\varnothing\right\}$:

$$H = \rho_b\left(\hat{a}^2\,a - N\right) + \rho_d\,(a-N) + \rho_i\,(\hat{a} - I)$$

$$G_m(z,t) = \left(\frac{(\rho_b - \rho_d)}{\rho_b(1-e^{(\rho_b-\rho_d)t})\,z + (\rho_b\,e^{(\rho_b-\rho_d)t}-\rho_d)}\right)^{\rho_i/\rho_b}\left(\frac{(\rho_b - \rho_d\,e^{(\rho_b-\rho_d)t})\,z + \rho_d(e^{(\rho_b-\rho_d)t}-1)}{\rho_b(1-e^{(\rho_b-\rho_d)t})\,z + (\rho_b\,e^{(\rho_b-\rho_d)t}-\rho_d)}\right)^m.$$

The detailed mathematical theory of such birth/death processes can be extended to multiple object types, and includes conditions for the exclusion of "explosions" in which the number of terms generated grows to infinity in a finite amount of time.

Synthesis-decay chemical reaction network $\left\{\varnothing\xrightarrow{\mathrm{k_s}}\mathrm{A},\ \mathrm{A}\xrightarrow{\mathrm{k_d}}\varnothing\right\}$:

$$H = k_s\,(\hat{a} - I) + k_d\,(a - N)$$

Defining $\kappa = k_s / k_d$, the differential equation and its solution by separation of variables are

$$G_m(z,t) = \int g_{m\,\lambda}(z)\,h_{m\,\lambda}(z)\,d\lambda$$

$$\left[k_d(z-1)\left[\ \tfrac{\partial}{\partial z} - \kappa\right] - \lambda\right] g_{m\,\lambda}(z) = 0 = \tfrac{\partial}{\partial t}\,h_{m\,\lambda}(z) + \lambda\,h_{m\,\lambda}(z)$$

$$g_{m\,\lambda}(z) = c_m(\lambda)\,(z-1)^{\lambda/k_d}\,e^{z\,\kappa}$$

and finally

$$g_m(z,t) = \left(1 + (z-1)\,e^{-k_d\,t}\right)^m e^{\kappa(z-1)\left(1-e^{-k_d\,t}\right)}.$$

Bidirectional conversion chemical reaction: $\left\{\mathrm{A}\xrightarrow{\mathrm{k_f}}\mathrm{B},\ \mathrm{B}\xrightarrow{\mathrm{k_r}}\mathrm{A}\right\}$:

Note that conservation of $A+B$ makes this actually a finite-dimensional system, provided that the initial conditions give zero probability to all values of A and B above some finite limit. Whether this condition is satisfied or not, the solution is as follows.

$$H = k_f\,(\hat{a}_2\, a_1 - N_1) + k_r\,(\hat{a}_1\, a_2 - N_2)$$

Defining $g_{\{m\}\lambda}(z) = z_1^{m(1)}\, z_2^{m(2)}\, g_\lambda(\zeta)$, $\zeta = z_1 / z_2$, and $\kappa = k_f / k_r$, the method of characteristics gives

$$g_{\{m\}}(\zeta, t) = e^{\zeta\, m_2}\,(\zeta+\kappa)^{-\kappa\,(m_1+m_2)}\, C\left[t - \frac{\log(\zeta-1) + \kappa\log(\zeta+\kappa)}{k_f + k_r}\right]$$

and using the initial condition $g_{\{m\}}(\zeta, 0) = 1$ we find

$$g_{\{m\}}(\zeta, t) = e^{m_2(\zeta - \varphi(\zeta,t))}\left(\frac{\varphi(\zeta, t) + \kappa}{\zeta+\kappa}\right)^{\kappa\,(m_1+m_2)}$$

where

$$\begin{array}{c}\psi_{\kappa,k_r}(\zeta) \equiv \log(\zeta-1) + \kappa\log(\zeta+\kappa)\\ \psi_{\kappa,k_r}(\varphi(\zeta,t)) = -(1+\kappa)\,k_r\,t + \psi_{\kappa,k_r}(\zeta)\end{array}.$$

This solves the model. For example we can compute the average amount of reaction product $\langle n_2\rangle$ as a function of time:

$$\begin{aligned}\langle n_2(t)\rangle &= \left.\frac{\partial \log G_m(z,t)}{\partial z_2}\right|_{z=1} = m_2 + \langle n(t)\rangle\\ &= m_2\, e^{-(k_f+k_r)\,t} + \frac{k_f}{k_f+k_r}\,(m_1+m_2)\left(1 - e^{-(k_f+k_r)\,t}\right)\end{aligned}$$

Correlations can be calculated to any order. A few more elaborate reactions have been solved in equilibrium using hypergeometric functions [26].

4.7 Biological Examples

4.7.1 Multiscale Dynamical Grammars in Biological Development

The growing tip (shoot apical meristem) of the plant Arabidopsis thaliana provides an illustrative example of a variable-structure dynamical system at three different scales: the molecular, cellular, and organ levels. At the molecular level, the primary molecules are auxin (a plant growth hormone) and PIN1 (a membrane-bound auxin pump). In a simple model [27], PIN1 in the membrane of cell i bounding cell j acts as a catalyst in removing auxin molecules from cell i (modeled annihilation) and simultaneously inserting them into cell j (creation). Reciprocally, auxin acts on PIN1, both enhancing its synthesis and directing its incorporation into the membranes of nearby cells. The reactions in this positive feedback loop are

$$\left\{\text{auxin}[i] \overset{\text{PIN1}[i,j]}{\Longrightarrow} \text{auxin}[j],\ \emptyset \overset{\text{auxin}[i]}{\Longrightarrow} \text{PIN1}[i],\ \text{PIN1}[i] \overset{\text{auxin}[j]}{\Longrightarrow} \text{PIN1}[i,j],\ \text{PIN1}[i,j] \to \emptyset,\ \text{PIN1}[i,j] \to \text{PIN1}[i]\right\},$$

where for example

$$\text{auxin}[i] \overset{\text{PIN1}[i,j]}{\Longrightarrow} \text{auxin}[j] \quad = \quad \{\text{auxin}[i] + \text{PIN1}[i,\ j] \to \text{Complex1},$$
$$\text{Complex1} \to \text{auxin}[i] + \text{PIN1}[i,\ j],\ \text{Complex1} \to \text{auxin}[j] + \text{PIN1}[i,\ j]\}$$

At the cell level, cells have internal state including the above reactions and also cell mass and position. When mass exceeds a threshold cell divide. Mass influences the resting length of elastic springs connecting neighboring cells which determine their positions. Positions determine which cells are neighbors, therefore which regulatory subnetworks are connected. There is variable structure both in the objects (cells) and their relationships (communicating neighbors; lineage trees of cell ancestry).

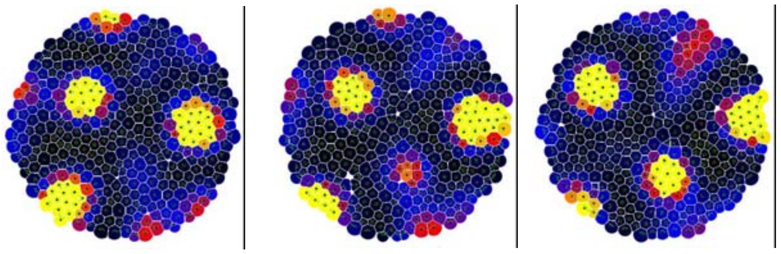


Figure 11: SAM, top view, three time slices. Color = auxin concentration. Emergent peaks correspond to floral meristem primordia.

Figure 11 shows the resulting dynamical pattern of auxin (yellow/blue scale) evolving over time. Emergent phenomena are the auxin peaks that form off center and move out radially to make room for new peaks. The peaks are hypothesized to determine the position of the primordia for new floral meristems in the phyllotactic pattern of flowers, leaves and branches for the above-ground part of the plant. The variable-structure objects at this scale are the primordia.

The variable structure dynamics illustrated by this model is not analytically tractable but is expressible at each of three levels (the molecular, cellular, and organ levels) using the dynamical grammars formalism. The ability of a cell to divide and interact with its neighbors gives rise, at a coarser spatial and temporal scale, to the ability of a shoot apical meristem to branch and create the floral meristems. Whether the multiscale relationships can be fully understood within this formalism is a challenge for future work.

4.7.2 Multiparent clustering

In contrast to the context-free feature tree, this grammar allows either one or two parents for each node. This model can be applied to clustering. It can also be applied to lineage trees of individual organisms in a population.

grammar (continuous-time) *multiparent* ($\{\text{nodeset}(x_i)\} \to \{\text{node}(x_j)\}$) {

R1 : $\text{nodeset}(x) \to \text{node}(x), \{\text{nodeset}(\tilde{x}_j)\,|\}$ with $\rho_1\, q(n)\, \phi_1(x_1)$ // single-parent reproduction

R2 : $\text{nodeset}(x_1),\ \text{nodeset}(x_2) \to \text{node}(x_1),\ \text{node}(x_2),\ \{\text{nodeset}(\tilde{x}_i) \mid 1 \leqslant i \leqslant n\}$

with $\rho_2\, C(x_1, x_2)\, q(n) \prod_{i=1}^{n} \phi_2(\tilde{x}_i \mid x_1, x_2)$ // two-parent reproduction

}

R1 summarizes both rules of the context-free clustergen grammar. R2 adds something new: a context-sensitive rule for two-parent reproduction. This model can be applied to clustering or, as shown below, to lineage trees of individual organisms. Resource bounds can be imposed as in the case of the the context-free feature tree, now with separate costs for single and multiple parentage. There is again a connection to Bayes Nets: given the DAG B of which nodes are parents to which others, the feature vectors x have conditional distributions for which B is the Bayes network. However, B is determined only when the grammar is executed, by sampling according to q and C.

4.7.3 Evolution of Genotypes

As another example of a multiscale dynamical grammar, we mention the dynamics of the lineage DAG (at the fine scale) in a sexually reproducing and evolving population of model organisms. As an emergent phenomenon such a population may undergo speciation and reproductive isolation, to produce a coarse-scale branching event in a phylogeny, which can in turn be modeled as a context-free random binary feature tree.

Darwinian evolution proceeds by selection on heritable variation, which we here decompose as two processes: selection, and reproduction with variation. We exhibit a continuous-time, context-sensitive grammar for modelling the evolution of such organisms, each of which has a genotype vector g and a somatic state vector x. To study the emergence of reproductively isolated species we may define an average phenotype $\overline{g}$ for any population or subpopulation, along with a feature similarity or distance criterion for individual reproductive compatibility $C(g_1, g_2)$. Four processes (selection, heritable variation, individual experience, and social interaction) are modeled by four rules that encode reasonable first approximations about the flow of information, including a strong distinction between inheritence of genetic vs. nongenetic information. Resource conservation constraints are not explicitly included, though some components of the somatic vector x could represent scarce resources. Other components of x could be devoted to signaling genotype, g. Many variations on this grammar are possible, and for any such grammar, choice of the functions C, q, etc will strongly affect the emergent dynamics.

```
**grammar** (continuous-time) *evolver* ({organism($g_i$, $e_i$)} → {organism($g_j$, $e_j$)}) {
        R1 : organism($g_1$, $e_1$) → ∅ with $\rho_d(g_1, e_1)$
                // death: selection by the environment
        R2 : organism($g_1$, $e_1$), organism($g_2$, $e_2$) → organism($g_1$, $e_1$), organism($g_2$, $e_2$),
                    {organism($\tilde{g}_i$, $\tilde{e}_i$) | $1 \leq i \leq n$}
                with $\rho_h\, C(g_1, g_2, e_1, e_2)\, q(n) \prod_{i=1}^{n} \phi_2(\tilde{g}_i \mid g_1, g_2)\, \varsigma(\tilde{e}_i)$
                // heritable variation of $\tilde{g}_i$
        R3 : organism($g$, $e$) → organism($g$, $\tilde{e}$) with $\rho_e\, \psi(\tilde{e} \mid e)$
                // individual experience
        R4 : organism($g_1$, $e_1$), organism($g_2$, $e_2$) → organism($g_1$, $\tilde{e}_1$), organism($g_2$, $\tilde{e}_2$)
                with $\rho_s\, \chi(\tilde{e}_1, \tilde{e}_2 \mid e_1, e_2, g_1, g_2)$
                // social interaction
}
```

Here s is a vector of “epigenetic” information acquired from experience but not passed on by inheritance, and g is a vector of inherited genotypic information. Grammar *evolver* is really a *schema* for an imporant family of complex dynamical systems rather than an individual model.

4.7.4 BN Grammar

Bayes Nets have been widely used in bioinformatics [28-29]. There is an elementary transformation from any BN to an equivalent grammar ([1], Section 1.3). Essentially, each random variable X in the BN is transformed into two parameterized terms: an undetermined random variable X' and a sampled or determined version $X(x)$ parameterized by its value, x. Each random variable is also assigned a rule $R_X : X', \{Y(y)\} \to X(x), \{Y(y)\}$ which can fire and determine that variables' value only when X is undetermined but all its predecessors Y have been determined. With suitable index paremeters, all of these rules can be expressed in a single SPG meta-rule.

5 Discussion

The Stochastic Parameterized Grammar (SPG) semantics function $\Psi(\Gamma)$ can be used to give a precise meaning to the context free feature tree and to many other variable-structure systems specified by SPG's in either continous or discrete time execution models. Likewise, the Dependency Diagram semantics function $\Psi(D)$ can be used to give a precise meaning to dependency diagrams for the feature tree and other variable structure systems, as well as other highly structured graphical models, by defining new node and link types including interaction gating, index nodes for replication of structure, contingent node existence, undirected constraints, time delay dependencies, and others. Both frameworks can be used to express highly structured graphical models including but not limited to variable-structure systems. Having formally defined the meanings of these notations, it is now possible to study their meaning-presevering mappings or transformations. We expect this to be a fruitful area of inquiry.

Since we have defined two different formal mathematical descriptions of classes of variable structure systems, that both encompass at least the same context free feature tree model, the question of the general relationship between these two frameworks arises. It will not be fully answered here but there are several relevant observations. Most notably, an SPG is intrinsically a dynamical system that can be simulated or executed numerically, whereas the PDF corresponding to any DD (except for pure Bayes Nets, which can each be represented as a simple SPG with one rule per random variable Section 4.7.4) is a static object with no unique implied sampling dynamics. Roughly, SPG's represent processes and DD's statically represent the outcomes of those processes. This relationship is consistent with the historical role of the Boltzmann distribution in statistical mechanics as a theory of long-time, coarse-scale equilibria arising from more fundamental equations of motion.

In this light, the following technical relationships between DD's and SPG's are understandable. (1) Given a fixed-structure DD there are many ways to use the "detailed balance" principle to create an executable SPG whose discrete-time dynamics tend in the limit of long time to the PDF of the DD. (2) Given an SPG the asymptotic long-time behavior of the grammar (or of any dynamical system) can be used to define an effective single grammar rule with conditional probability distribution on output terms given input terms. This single grammar rule has an unnormalized probability distribution that can be invoked recursively within other grammars using the **via** keyword in place of **with**. It can also be modeled approximately by variable-structure DD's of varying complexity. (3) In particular, a dependency diagram with a large number of time delay links and dense existence links between immediately succeeding temporal variables can in principle be "unrolled" to express discrete-time grammar executions. (4) The elementary distributions associated with individual rules in a SPG or

DG may be specified by a fixed-structure DD or by the Boltzmann distribution of an MRF [30]. At a much higher level of abstraction, (5) *term(parameter-vector)* combinations correspond to instantiated "objects" in a high-level object/relationship network.

As a result of the relationships (1)-(4), multiscale models of processes and/or dynamical systems may be formulated in terms of an alternation of dynamical SPG and summarizing DD at each scale.

The SPG formalism we have outlined here includes the possibility of a MRF Boltzmann distribution for each rule, and can straightforwardly be extended to include stochastic graph grammars, differential equations [20-21], and stochastic differential equations; we take those extensions to be within the purview of "dynamical grammars" (DG's). Many scientific applications can now be expressed in terms of context-sensitive dynamical grammars including modeling the growing tip (shoot apical meristem) of the plant *Arabidopsis thaliana* at three different scales: the molecular, cellular, and organ levels [31].

Although inference algorithms for variable structure system models lie outside the scope of this paper, it is natural to consider quantitative sampling algorithms expressed as discrete-time variable-structure systems, just as Markov Chain Monte Carlo training algorithms are fixed-structure Markov chains.

Acknowledgements. This research was supported by NIH BISTI grant GM069013, NSF:FIBR award number EF-0330786, and the School of Information and Computer Sciences at the University of California, Irvine. Helpful discussions occurred with Ashish Bhan, Alex Sadovsky, Michael Turmon, Max Welling, and Guy Yosiphon.

References

[1] Mjolsness, E. (2004). *Labeled Graph Notations for Graphical Models: Extended Report*. University of California, Irvine. UCI ICS TR #04-03, http://www.ics.uci.edu/~emj

[2] Buntine, W. L. (1994). *Operations for Learning with Graphical Models*. Journal of Artificial Intelligence Research.

[3] Getoor, L., Friedman, N., Koller, D., & Pfeffer, A. (2001). *Learning Probabilistic Relational Models*. In S. Dzeroski & N. Lavrac (Ed.), *Relational Data Mining* Springer.

[4] Heckerman, D., Meek, C., & Koller, D. (2004). *Probabilistic Models for Relational Data*. Microsoft Research, Redmond. Technical Report MSR-TR-2004-30.

[5] Athreyea, K. B., & Ney, P. E. (1972). *Branching Processes*. Springer-Verlag; Dover.

[6] Snyder, D. L., & Miller, M. I. (1991). *Random Point Processes in Time and Space*. New York: Wiley.

[7] Giavitto, J., & Michel, O. (2001). *MGS: a Programming Language for the Transformations of Topological Collections*. CNRS Université d'Evry Val d'Essonne, Evry. 61-2001 http://mgs.lami.univevry.fr/PUBLICATIONS/publication.html#Documentation

[8] Liggett, T. M. (1985). *Interacting Particle Systems*. New York: Springer-Verlag.

[9] Milch, B., Marthi, B., Russell, S., & Sontag, D. (2005). *BLOG: Probabilistic Models with Unknown Objects*. Proceedings of the International Joint Conference on Artificial Intelligence.

[10] Smith, C., Prusinkiewicz, P., & Samavati, F. (2003). *Local specification of surface subdivision algorithms*. In J. Pfaltz, M. nagl & B. Bohlen (Ed.), *Applications of Graph Transformations with Industrial Relevance*

(AGTIVE 2003): Lecture Notes in Computer Science 3062 (pp. 313–327). Springer-Verlag. http://www.algorithmicbotany.org/papers/localspec.agtive2003.pdf

[11] Jensen, K. (1997). *Coloured Petri Nets I, II, III.* Springer-Verlag.

[12] Mjolsness, E. (1997). *Symbolic Neural Networks Derived from Stochastic Grammar Domain Models*. In R. Sun & R. Alexandre (Ed.), *Connectionist Symbolic Integration*. Lawrence Erlbaum Associates.

[13] Hawkins, D., & Ulam, S. (1944). *Theory of Multiplicative Processes, 1.*. Los Alamos Scientific Laboratory, Los Alamos. LA-171.

[14] Frey, B. (2003). *Extending Factor Graphs so as to Unify Directed and Undirected Graphical Models*. Proceedings of the Nineteenth Conference on Uncertainty in Artificial Intelligence.

[15] Lauritzen, S. (1996). *Graphical Models*. Oxford University Press.

[16] Gold, S., Rangarajan, A., & Mjolsness, E. (1996, May 15). *Learning with Preknowledge: Clustering with Point and Graph Matching Distance Measures*. Neural Computation, **8**(4).

[17] Mjolsness, E., Garrett, C., & Miranker, W. (1991, March). *Multiscale Optimization in Neural Networks*. IEEE Transactions on Neural Networks, **2**(2).

[18] Hirsch, M. W. (1976). *Differential Toplogy*.. New York: Springer-Verlag.

[19] Murphy, K. P. (2002). *Dynamic Bayesian Networks: Representation, Inference and Learning* (Doctoral Dissertation, University of California, Berkeley).

[20] Mjolsness, E., Sharp, D. H., & Reinitz, J. (1991). *A Connectionist Model of Development*. Journal of Theoretical Biology, **152**(4), 429–454.

[21] Prusinkiewicz, P., Hammel, M. S., & Mjolsness, E. (1993). *Animation of Plant Development*. SIGGRAPH '93 Conference Proceedings.

[22] van Kampen, N. G. (1981). *Stochastic Processes in Physics and Chemistry*. North-Holland.

[23] Kondor, R., & Lafferty, J. (2002). *Diffusion Kernels on Graphs and Other Discrete Input Spaces*. In (Ed.), *Proceedings of the International Conference on Machine Learning*.

[24] Mattis, D. C., & Glasser, M. L. (1998). *The uses of quantum field theory in diffusion-limited reactions*. Reviews of Modern Physics, **70**, 979–1001.

[25] Risken, H. (1984). *The Fokker-Planck Equation*. Berlin: Springer.

[26] McQuarrie, D. A.. *Stochastic Approach to Chemical Kinetics*. J. Appl. Prob., 413–478.

[27] Jönsson, H., Heissler, M., Shapiro, B,. Meyerowitz, M. & Mjolsness, E.. *An auxin-driven polarized transport model for phyllotaxis*. Manuscript in preparation.

[28] Baldi, P., & Brunak, S. (2001). *Bioinformatics: The Machine Learning Approach*. MIT Press.

[29] Segal, E., Yelensky, R., & Koller, D. (2003). *Genome-wide discovery of transcriptional modules from DNA sequence and gene expression*. Bioinformatics, **19**(Supplement 1), i273–i282 .

[30] Mjolsness, E. (1994). *Connectionist Grammars for High-Level Vision*. In V. Honavar & L. Uhr (Ed.), *Artificial Intelligence and Neural Networks: Steps Toward Principled Integration* Academic Press.

[31] Jönsson, H., et al. (2005). *Modeling the Organization of the WUSCHEL Expression Domain in the Shoot Apical Meristem*. In *Procedings of Intelligent Systems in Molecular Biology*.

[32] Ghahramani, Z. (2005). *Non-parametric Bayesian Methods*. Uncertainty in Artificial Intelligence (UAI) Tutorial notes. Retrieved from http://www.gatsby.ucl.ac.uk/~zoubin/talks/uai05tutorial-b.pdf

A Appendices

5.1 Operator algebra for continuous-event rules

Following [20] or [21], we may introduce extended-time process rules which change the real-valued parameters but not the numbers or types of the interactors during an extended, finite time interval rather than a pointlike transition or event time. In such a process, for each infinitesimal time interval Δt we have a rate of transition to altered parameters as follows:

$$\begin{aligned} \tilde{O}_r &= \int d\{x_i\} \int d\{y_i\}\, \rho_r \left[\prod_{i\in \mathrm{lhs}(r)} \hat{a}(\tau_i, y_i)\, a(\tau_i, x_i)\right] \\ &\quad \times \rho_r(\{x_i\})[\lim_{\Delta t\to 0} \mathrm{Pr}_r(\{y_i\} \mid \{x_i\}, \Delta t)/\Delta t] \\ O_r &= \tilde{O}_r - \mathrm{diag}\left(\mathbf{1}^T \cdot \tilde{O}_r\right) = \tilde{O}_r - \rho_r \prod_{i\in \mathrm{lhs}(r)} N(\tau_i, x_i) \end{aligned} \tag{31}$$

Two issues arise with this formulation: the limit of the conditional probability within the integral, and the computation of an integral over operator expressions. The small-time transition probability $\mathrm{Pr}_r(\{x_i + \Delta x_i\} \mid \{x_i\}, \Delta t)$

$$\rho_r(\{y_i\} \mid \{x_i\}) = \rho_r(\{x_i\}) \lim_{\Delta t\to 0} [\mathrm{Pr}_r(\{y_i\} \mid \{x_i\}, \Delta t) - \mathrm{Pr}_r(\{x_i\} \mid \{x_i\}, \Delta t)]/\Delta t$$

is dependent on the process being modeled, but there are a few standard types corresponding to differential equations such as diffusion and transport.

5.1.1 Ordinary and Stochastic Differential Equations

There are SPG rule forms corresponding to stochastic differential equations governing diffusion and transport. Given the SDE or equivalent Langevin equation (which specializes to a system of ordinary differential equations when $\eta(t) = 0$):

$$\begin{gathered} d x_i = v_i(\{x_k\})\, d t + \sigma(\{x_k\})\, d W \quad \text{or} \\ \frac{d x_i}{d t} = v_i(\{x_k\}) + \eta_i(t) \end{gathered} \tag{32}$$

under some conditions on the noise term $\eta(t)$ [25] the dynamics can be expressed as a Fokker-Planck equation for the probability distribution $P(\{x\}, t)$:

$$\frac{\partial P(\{x\}, t)}{\partial t} = -\sum_i \frac{\partial}{\partial x_i} v_i(\{x\})\, P(\{x\}, t) + \sum_i \frac{\partial^2}{\partial x_i\, \partial x_j} D_{ij}(\{x\})\, P(\{x\}, t)$$

where D is related to η and W. The first term on the right is the drift of probability due to v, and the second is its diffusion due to D or η. Let $P(\{y\}, t \mid \{x\}, 0)$ be the solution of this equation given initial condition $P(\{y\}, 0) = \delta(\{y\} - \{x\}) = \prod_k \delta(y_k - x_k)$ (with Dirac delta function appropriate to the particular measure μ used for each component). Then at $t = 0$,

$$\frac{\partial P(\{y\}, 0 \mid \{x\}, 0)}{\partial t} \equiv \rho(\{y_i\} \mid \{x_i\}) = -\sum_i \frac{\partial}{\partial y_i} v_i(\{x\})\, \delta(\{y\} - \{x\}) + \sum_i \frac{\partial^2}{\partial y_i\, \partial y_j} D_{i\,j}(\{x\})\, \delta(\{y\} - \{x\})$$

Thus the probability rate $\rho(\{y_i\} \mid \{x_i\})$ is given by a differential operator acting on a Dirac delta function. It can be decomposed into drift and diffusion:

$$\rho_{\text{drift}}(\{y_i\} \mid \{x_i\}) = -\sum_i \frac{\partial}{\partial y_i} v_i(\{x\}) \prod_i \delta(y_i - x_i) \tag{33}$$

$$\rho_{\text{diffusion}}(\{y_i\} \mid \{x_i\}) = \sum_{i\,j} \frac{\partial^2}{\partial y_i\, \partial y_j} D_{i\,j}(\{x\}) \prod_i \delta(y_i - x_i) \tag{34}$$

from which by (Equation 31) we construct the evolution generator operators $O_{\text{FP}} = O_{\text{drift}} + O_{\text{diffusion}}$, where

$$O_{\text{drift}} = -\int d\{x\} \int d\{y\}\, \hat{a}(\{y\})\, a(\{x\}) \left(\sum_i \nabla_{y_i} v_i(\{y\}) \prod_k \delta(y_k - x_k) \right) \tag{35}$$

$$O_{\text{diffusion}} = \int d\{x\} \int d\{y\}\, \hat{a}(\{y\})\, a(\{x\}) \left(\sum_{i\,j} \nabla_{y_i} \nabla_{y_j} D_{i\,j}(\{y\}) \prod_k \delta(y_k - x_k) \right) \tag{36}$$

The second order derivative terms give diffusion dynamics and also regularize and and promote continuity of probability in parameter space both along and transverse to any local drift direction. (Equation 35) and (Equation 36) constitute an operator algebra expression of ordinary and stochastic differential equation dynamics.

If a grammar includes such (ordinary or stochastic) Differential Equation rules along with non-DE rules, a solver can be used to compute $\exp((t_{n+1} - t_n)\, O_{\text{FP}})$ in the time-ordered product for $\exp(t\, H)$ as a hybrid simulation algorithm for discontinuous (jump) stochastic processes combined with stochastic differential equations. This gives a second application of the time-ordered product expansion of Section 4.5.1.

5.1.2 Connection to quantum field theory calculations

The most important connection to QFT methods is the time-ordered operator expansion of (Section 4.5.1). It is also informative, however, calculate matrix entries as follows.

A systematic use of the foregoing formalism reveals connections to many methods of quantum field theory, without the conventional reliance on momentum space methods which doesn't fit all of our problems. As an example, we can use the relations

$$\mid z \rangle = \hat{a}(\{z\}) \mid 0 \rangle\,, \quad \langle w \mid = \langle 0 \mid a(\{w\})$$
$$[a(\{x\}),\ \hat{a}(\{y\})] = \delta(\{y\} - \{x\})[1 + N(\{x\})\, Q(N(\{x\}), n^{\max})]$$
$$\langle w \mid z \rangle = \delta(\{w\} - \{z\})$$

(where Q is a polynomial of degree $n^{\max}$ -1 and $n^{\max}$ is the largest number of identical objects allowed for a given type) to calculate the matrix element for a single particle's time evolution under diffusion alone, with a constant diffusion coefficient $D = 1$, starting from a single parameter vector y and ending at parameter vector z. First we calculate some auxiliary matrix elements:

$$\langle z \mid y \rangle = \langle 0 \mid a(z)\, \hat{a}(y) \mid 0 \rangle = \delta(z - y) = \int d x\, \delta(x - y)\, \delta(x - z)$$

and

$$\begin{aligned} \langle z \mid H_{\text{diff}} \mid y \rangle &= (0 \mid a(z)\, H_{\text{diff}}\, \hat{a}(y) \mid 0 \rangle \\ &= \int d x_1 \int d x_2\, \langle 0 \mid a(z)\, \hat{a}(x_2)\, a(x_1)\, \hat{a}(y) \mid 0 \rangle\, \nabla_{x_1}^2\, \delta(x_1 - x_2) \\ &= \int d x_1 \int d x_2\, \delta(z - x_2)\, \delta(x_1 - y)\, \nabla_{x_1}^2\, \delta(x_1 - x_2) \\ &= \int d x\, \delta(x - y)\, \nabla_x^2\, \delta(x - z) \end{aligned}$$

and (using $\int d x\, \hat{a}(x)\, a(x) = \mathbf{1}$)

$$\begin{aligned} \left\langle z \,\middle|\, {H_{\text{diff}}}^2 \,\middle|\, y \right\rangle &= \left\langle 0 \,\middle|\, a(z)\, {H_{\text{diff}}}^2\, \hat{a}(y) \,\middle|\, 0 \right\rangle \\ &= \int d x_2\, \langle 0 \mid a(z)\, H_{\text{diff}}\, \hat{a}(x_2)\, a(x_2)\, H_{\text{diff}}\, \hat{a}(y) \mid 0 \rangle \\ &= \int d x_2\, \langle 0 \mid a(z)\, H_{\text{diff}}\, \hat{a}(x_2) \mid 0 \rangle\, \langle 0 \mid a(x_2)\, H_{\text{diff}}\, \hat{a}(y) \mid 0 \rangle \\ &= \int d x_2 \left(\int d x_3\, \delta(x_3 - x_2)\, \nabla_{x_3}^2\, \delta(x_3 - z) \right) \left(\int d x_1\, \delta(x_1 - y)\, \nabla_{x_1}^2\, \delta(x_1 - x_2) \right) \\ &= \int d x_1 \int d x_2 \int d x_3\, \delta(x_3 - x_2) \left(\nabla_{x_3}^2\, \delta(x_3 - z) \right) \delta(x_1 - y) \left(\nabla_{x_1}^2\, \delta(x_1 - x_2) \right) \\ &= \int d x_1 \int d x_3\, \delta(x_1 - y) \left(\nabla_{x_3}^2\, \delta(x_3 - z) \right) \left(\nabla_{x_1}^2\, \delta(x_1 - x_3) \right) \\ &= \int d x_1 \int d x_3\, \delta(x_1 - y) \left(\nabla_{x_3}^2\, \delta(x_3 - z) \right) \left(\nabla_{x_3}^2\, \delta(x_1 - x_3) \right) \\ &= \int d x_1 \int d x_3\, \delta(x_1 - y)\, \delta(x_3 - z) \left(\left(\nabla_{x_3}^2 \right)^2 \delta(x_1 - x_3) \right) \\ &= \int d x\, \delta(x - z)\, (\nabla_x^2)^2\, \delta(y - x)\, . \end{aligned}$$

The last three calculations can be generalized to arbitrary powers of H_{diff} :

$$\langle z \mid {H_{\text{diff}}}^n \mid y \rangle = \int d x\, \delta(x - z) \left(\nabla_x^2 \right)^n \delta(y - x)$$

from which we can calculate

$$\begin{aligned} \langle z \mid \exp(t\, H_{\text{diff}}) \mid y \rangle &= \int d x\, \delta(x - z)\, \exp\left(t\, \nabla_x^2 \right) \delta(y - x) \\ &= \exp\left(t\, \nabla_z^2 \right) \delta(y - z)\, . \end{aligned} \tag{37}$$

Using (Johnson and Lapidus 10.2.6)

$$\left(e^{t\, \nabla_x^2}\, f \right)(x) = (4\, \pi\, t)^{-d/2} \int d u\, f(u)\, e^{-\| x - u \|^2 / 2\, t},$$

we find

$$\langle z \mid \exp(t\, H_{\text{diff}}) \mid y \rangle = (4\,\pi\, t)^{-d/2} \int d\,x \int d\,u\, \delta(x - z)\, \delta(y - u)\, e^{-\| x-u \|^2/2\,t}$$

i.e.

$$\langle z \mid \exp(t\, H_{\text{diff}}) \mid y \rangle = (4\,\pi\, t)^{-d/2}\, e^{-\| z-y \|^2/2\,t}\,. \tag{38}$$

We can calculate with drift in a similar way:

$$\begin{aligned}
\langle w \mid H_{\text{drift}} \mid z \rangle &= -\left\langle \{w\} \middle| \int d\{x\} \int d\{y\} \left(\sum_i \nabla_{y_i}\, v_i(\{y\})\, \delta(\{y\} - \{x\}) \right) \hat{a}(\{y\})\, a(\{x\})\, \hat{a}(\{z\}) \middle| 0 \right\rangle \\
&= -\left\langle \{w\} \middle| \int d\{x\} \int d\{y\} \left(\sum_i \nabla_{y_i}\, v_i(\{y\})\, \delta(\{y\} - \{x\}) \right) \hat{a}(\{y\})\, \delta(\{z\} - \{x\})[1 + N(\{x\})] \middle| 0 \right\rangle \\
&= -\int d\{x\} \int d\{y\} \left(\sum_i \nabla_{y_i}\, v_i(\{y\})\, \delta(\{y\} - \{x\}) \right) \delta(\{z\} - \{x\})\, \langle \{w\} \mid \{y\} \rangle \\
&= -\int d\{y\} \left(\sum_i \nabla_{y_i}\, v_i(\{y\})\, \delta(\{y\} - \{z\}) \right) \delta(\{w\} - \{y\}) \\
&= +\int d\{y\}\, \delta(\{y\} - \{z\}) \left(\sum_i v_i(\{y\})\, \nabla_{y_i}\, \delta(\{w\} - \{y\}) \right) \\
&= \sum_i v_i(\{z\})\, \nabla_{z_i}\, \delta(\{w\} - \{z\})
\end{aligned}$$

and similarly

$$\begin{aligned}
\langle z \mid H_{\text{drift}}{}^n \mid y \rangle &= \int d\,x\, \delta(x - z)\, (v(x) \cdot \nabla_x)^n\, \delta(y - x) \\
\langle z \mid e^{t\, H_{\text{drift}}} \mid y \rangle &= \int d\,x\, \delta(x - z) \exp[t\, v(x) \cdot \nabla_x]\, \delta(y - x)
\end{aligned}$$

Using Taylor’s theorem in differential operator form,

$$\left(e^{t\, c \cdot \nabla_x}\, f \right)(x) = f(x + c),$$

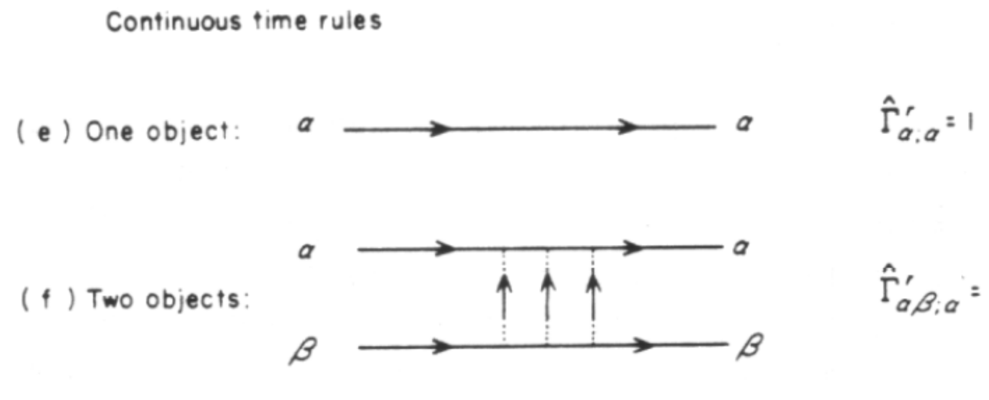
(where c is a constant) we find

$$\left\langle z \middle| e^{t\, H_{\text{drift}}} \middle| y \right\rangle = \int d\,x(0)\, \delta(x(0) - z)\, \delta\!\left(y - \left(x(0) + \int_0^t v(x(t))\, d\,t \right) \right) \tag{39}$$

which expresses a formal solution for the ordinary differential equation without noise:

$$\frac{d\,x_i}{d\,t} = v_i(\{x_k\})$$

Some possible types of extended-time continuous-valued transition rules are shown in the following diagram [20].



5.1.3 Partial Differential Equations and Stochastic PDE's

Finally, an important problem is to translate partial differential equations and stochastic partial differential equations of general form into the operator algebra. This can be done by relating PDE's and SPDE's to large systems of ODE's and SDE's, and taking the limit symbolically. Nontrivial mathematics will be needed to confirm whether the indicated limits really exist or not.

Consider the (possibly stochastic) PDE

$$\frac{\partial \Phi(x)}{\partial t} = F[\Phi](x) = F\left(\Phi(x), \frac{\partial \Phi(x)}{\partial x}, \ \ldots, \ \frac{\partial^n \Phi(x)}{\partial x^n}\right) + \eta(t). \tag{40}$$

where x may be a scalar or a vector, and likewise for Φ. We make the following mapping to (Equation 31):

Table 4. Ordinary vs. Partial differential objects

Ordinary differential object	Partial differential object
d/dt	$\partial/\partial t$
i	x
x_i	$\Phi(x)$
y_i	$\Phi'(x)$
$\partial/\partial x_i$ (partial derivative)	$\delta/\delta\Phi(x)$ (functional derivative)
D (homog. scalar diffusion coef.)	D (homog. scalar diffusion coef.)
$\delta(\boldsymbol{y}-\boldsymbol{x}) = \prod_i \delta(y_i - x_i)$	$\Delta(\Phi' - \Phi) = \int dx\, \delta(\Phi'(x) - \Phi(x))$
$\int dx\, g(x)$ (ordinary integral)	$\int \mathcal{D}\Phi\, G[\Phi]$ (functional integral)
$a_\tau(\boldsymbol{x}) = a_\tau(\{x_i\})$	$a_\tau(\Phi) = a_\tau(\{\Phi(x)\})$

With this table of translations, the drift and diffusion operators for PDE's and SPDE's become

$$O_{\text{drift}} = -\int\int \mathcal{D}\Phi\, \mathcal{D}\Phi' \ \hat{a}(\Phi')\, a_\tau(\Phi) \left(\int dx \frac{\delta}{\delta\Phi'(x)} F[\Phi'](x)\, \Delta(\Phi' - \Phi)\right) \tag{41}$$

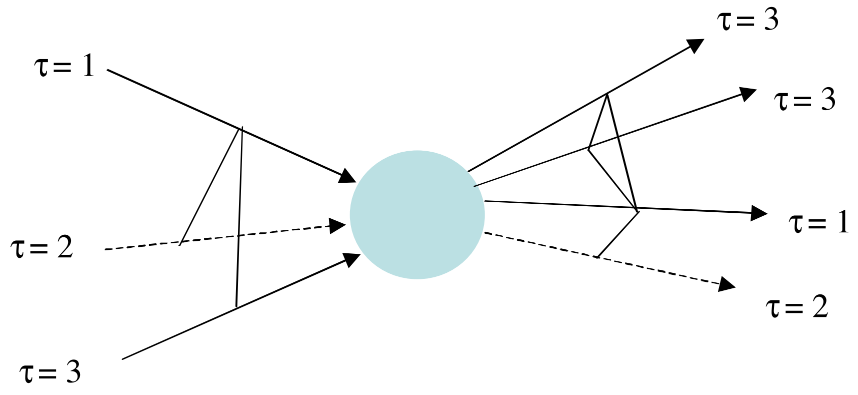
$$O_{\text{diffusion}} = D \int\int \mathcal{D}\Phi\, \mathcal{D}\Phi'\ \hat{a}(\Phi')\, a_\tau(\Phi) \left(\int dx\, \frac{\delta^2}{\delta\, \Phi'(x)^2}\, \Delta(\Phi' - \Phi) \right) \tag{42}$$

This gives another potential application of the time-ordered product expansion of Section 4.5.1, which can be used to create simulation algorithms.

With suitable PDE’s it becomes possible to represent dynamically changing manifolds, either by differential equations for the metric as in General Relativity, or for an explicit embedding into a higher dimensional space, or for an implicit embedding given by a function $f(\boldsymbol{x}) = 0$ (a level set method).

5.2 Stochastic Parameterized Graph Grammars (SPGG's)

Here is an example of a continuous-time graph grammar rule:



The following syntax introduces Object Identifier (OID) labels L_i for each parameterized term, and allows labelled terms to point to one another through a graph of such labels . The graph is related to two subgraphs of neighborhood indices $N(i, \sigma)$ and $N'(j, \sigma)$ specific to the input and output sides of a rule. Like types or variables, the label symbols appearing in a rule are chosen from an alphabet $\{L_\lambda \mid \lambda \in \Lambda\}$. Unlike types but like variables X_c, the label symbols $L_{\lambda(i)}$ actually denote nonnegative integer values - unique addresses or object identifiers.

A GG rule is of the form, for some nonnegative-integer-valued functions $\lambda(i)$, $\lambda'(j)$, $N(i, \sigma)$, $N'(j, \sigma)$ for which $(\lambda(i) = \lambda(j)) \Rightarrow (i = j)$, $(\lambda'(i) = \lambda'(j)) \Rightarrow (i = j)$:

$$\begin{gathered} \left\{ L_{\lambda(i)} := \tau_i\left(x_{a(i)};\, \left(L_{N(i,\sigma)} \,\middle|\, \sigma \in 1..\sigma^{\max}_{a(i)}\right)\right) \,\middle|\, i \in \mathcal{I} \right\} \\ \to \{L_{\lambda(i)} \mid i \in \mathcal{I}_1 \subseteq \mathcal{I}\} \bigcup \left\{ L_{\lambda'(j)} := \tau_j\left(x'_{a'(j)};\, \left(L_{N'(j,\sigma)} \,\middle|\, \sigma \in 1..\sigma^{\max}_{a'(j)}\right)\right) \,\middle|\, j \in \mathcal{J} \right\} \\ \textbf{with}\ \rho_r\left(\left\{x'_{a'(j)}\right\} \,\middle|\, \{x_{a(i)}\}\right) \end{gathered} \tag{43}$$

Note that the fanout of the graph is limited by $\sigma^{\text{cur}}_i \leqslant \sigma^{\max}_{a(i)}$. Let

$$\mathcal{I} = \mathcal{I}_1 \bigcup \mathcal{I}_2 \text{ and } \mathcal{I}_1 \bigcap \mathcal{I}_2 = \varnothing$$
$$\mathcal{J} = \mathcal{J}_1 \bigcup \mathcal{J}_2 \text{ and } \mathcal{J}_1 \bigcap \mathcal{J}_2 = \varnothing$$

$$\mathcal{J}_1 = \{j \in \mathcal{J} \wedge (\exists\, i \in \mathcal{I}_2 \mid \lambda(i) = \lambda'(j)\}$$
$$\mathcal{J}_2 = \{j \in \mathcal{J} \wedge (\nexists\, i \in \mathcal{I}_2 \mid \lambda(i) = \lambda'(j)\}$$
$$\mathcal{I}_3 = \{i \in \mathcal{I}_2 \wedge (\nexists\, j \in \mathcal{J}_1 \mid \lambda(i) = \lambda'(j)\} \subseteq \mathcal{I}_2)$$

This syntax may be translated to the ordinary non-graph grammar rule (where NextOID is a variable, and OIDGen and Null are types reserved for the translation):

$$\begin{gathered}\{\tau_{a(i)}(L_{\lambda(i)}, x_{a(i)}, (L_{N(i,\sigma)} \mid \sigma \in 1..\sigma_i^{\text{cur}})) \mid i \in \mathcal{I}\}, \text{OIDGen(NextOID)} \\ \to \{\tau_{a(i)}(L_{\lambda(i)}, x_{a(i)}, (L_{N(i,\sigma)} \mid \sigma \in 1..\sigma_i^{\text{cur}})) \mid i \in \mathcal{I}_1\} \bigcup \\ \left\{\tau_{a'(j)}\left(L_{\lambda'(j)}, x'_{a'(j)}, \left(L_{N'(j,\sigma)} \,\middle|\, \sigma \in 1..\sigma_j^{\text{cur}}\right)\right) \,\middle|\, j \in \mathcal{J}_1 \wedge (i \in \mathcal{I}_2) \wedge (\lambda(i) = \lambda'(j))\right\} \bigcup \\ \left\{\tau_{a'(j)}\left(L_{\lambda'(j)}, x'_{a'(j)}, \left(L_{N'(j,\sigma)} \,\middle|\, \sigma \in 1..\sigma_j^{\text{cur}}\right)\right) \,\middle|\, j \in \mathcal{J}_2\right\} \bigcup \{\text{Null}(L_{\lambda(i)}) \mid i \in \mathcal{I}_3\} \\ \bigcup \{\text{OIDGen(NextOID} + |\mathcal{J}|)\} \\ \textbf{with } \rho_r\left(\left\{x'_{a'(j)}\right\} \,\middle|\, \{x_{a(i)}\}\right) \prod_{j \in \mathcal{J}_2} \delta_K(L_{\lambda'(j)}\ ,\ \text{NextOID} + j - 1)\end{gathered}$$

which already has a defined semantics. Note that all set membership tests can be done at translation time because they do not use information that is only available dynamically during the grammar evolution. Optionally we may also add a rule schema (one rule per type, τ_a) to eliminate any dangling pointers:

$$\begin{gathered}\tau_a(L_{\lambda(1)}, x, (L_{N(1,\sigma)} \mid \sigma \in 1..\sigma_1^{\text{cur}})),\ \text{Null}(L_{\lambda(2)}) \\ \to \tau_a(L_{\lambda(1)}, x, (L_{N(1,\sigma)} \mid (\sigma \in 1..\sigma_1^{\text{cur}}) \wedge (N(1, \sigma) \neq \lambda(2)))),\ \text{Null}(L_{\lambda(2)}) \\ \textbf{with } \rho_{\text{cleanup}} \sum_{\sigma \in 1..\sigma^{\max}} \delta_K(L_{N(1,\sigma)}\ ,\ L_{\lambda(2)})\end{gathered}$$

Note that the rule's target graph structure has been encoded in N, and N', and a search for all label sets that might satisfy these constraints is called for by the sum over label variables. We can compare this to the search for graph motifs g in a larger graph G.

Strings may be encoded as one-dimensional graphs using either a singly or doubly linked list data structure. String rewrite rules

$$(\tau_{a(i)}(x_i) \mid i \in \mathcal{I}_L) \to (\tau_{a'(j)}(y_j) \mid j \in \mathcal{I}_R)\ \textbf{with}\ \rho_r((x_i), (y_j))$$

(note ordering of arguments) are emulated as graph rewrite rules, whose semantics are defined above. This form is capable of handling many L-system grammars. If ρ_r is not supplied it can be set to 1.

5.3 Multiple Scale Modeling

A major reason for considering a modeling language that encompasses stochastic, deterministic, discrete, continuous and other characteristics of models in one framework, is its potential applicability to multi-scale modeling. There are many cases where a stochastic model at one level is approximated by a deterministic model at the next coarser scale, and vice versa. For example there is such an alternation between quantum mechanics, molecular dynamics, kinetic theory, and thermodynamics of fluids.

Here we outline the automatable search for coarse-scale dynamical models, given fine-scale dynamics.

5.3.1 Example: Neural network energy functions

In the simplest case the probability distribution to be approximated at a coarse scale is in thermodynamic equilibrium, so there is no dynamics. Then our technical goal is to approximate a Boltzmann distribution of some given form, such as

$$P(x) = \frac{1}{Z_P} \exp\left[-\beta\left(\sum_{ij} W_{ij}\, x_i\, x_j + \sum_i b_i\, x_i + \sum_i \varphi_i(x_i) + \sum_{ij} W_{ijk}\, x_i\, x_j\, x_k\right)\right]. \tag{44}$$

with a coarse-scale MRF of the same general form:

$$Q(y) = \frac{1}{Z_Q} \exp\left[-\beta\left(\sum_{ab} V_{ab}\, y_a\, y_b + \sum_a \hat{b}_a\, y_a + \sum_a \tilde{\varphi}_a(y_a) + \sum_{ab} V_{abc}\, y_a\, y_b\, y_c\right)\right] \tag{45}$$

where there are fewer coarse-scale variables y_a than fine-scale variables x_i, and fewer coarse-scale interactions V than fine-scale interactions W. If this course-scale approximate model reduction can be defined once, it can be defined recursively as well to get a series of reduced models $\{P_0 = P,\ P_1 = Q[P],\ P_2 = Q[P_1],\ \ldots\}$. However, we must relate the x variables to the y variables, and suggest how to find and then use the functional $Q = \text{coarse}[P]$. To keep the relationship between x and y simple we introduce the restriction relation

$$\rho(y \mid x) = \frac{1}{Z_\rho} \exp\left[-\beta\left(\sum_{ai} R_{ai}\, y_a\, x_i + \sum_a \hat{\varphi}_a(y_a)\right)\right] \tag{46}$$

where $\hat{\varphi}$ is quadratic, and the prolongation relation

$$\pi(x \mid y) = \frac{1}{Z_\pi} \exp\left[-\beta\left(\sum_{ia} K_{ai}\, x_i\, y_a + \sum_i \check{\varphi}_i(x_i)\right)\right] \tag{47}$$

where $\check{\varphi}$ is quadratic. Note that these relations have the property that x or y can easily be integrated out.

We may consider parameterized limits of restriction and prolongation relation parameters and obtain Dirac delta functions on linear combinations of conditioned variables, in which case the restriction or prolongation *relation* becomes a restriction or prolongation *map* as in ordinary deterministic multigrid methods.

Next we introduce a distortion measure as an objective function to be minimized by choice of ρ, π, and $Q(\{y_a\})$, such as:

$$\begin{aligned} D[P(\{x_i\}),\, Q(\{y_a\})] &= D_{\text{KL}}\left[P(\{x_i\}),\ \textstyle\int \pi(\{x_i\} \mid \{y_a\})\, Q(\{y_a\})\, d\{y_a\}\right], \text{ or} \\ D[P(\{x_i\}),\, Q(\{y_a\})] &= D_{\text{KL}}\left[\textstyle\int \rho(\{y_a\} \mid \{x_i\})\, P(\{x_i\})\, d\{x_i\},\ Q(\{y_a\})\right]. \end{aligned} \tag{48}$$

Minimization of one or both of these asymmetric distances with respect to Q, ρ, and/or π, along with any desired sparsity or network structure constraints on V, R, and K, provides the statement of the learning phase of the multiscale modeling problem. The minimization can be done by gradient descent which turns out [Mjolsness, Sharp, Lackner unpublished] to be a generalized form of the Boltzmann machine algorithm of Hinton Sejnowski 1986, or by other optimization algorithms which use sample and sampled gradient information.

Note that if x_i is already a hierarchical space such as an image scale space, then y_a introduces a potentially different hierarchy which may or may not coincide with dropping the finest-scale pixels from the i -indexed scale space.

5.3.2 Uses of the approximation

Given an approximation Q of P, one can use it to speed up the convergence to equilibrium of a simulation of P, by alternating Markov chains constructed to equilibrate P and Q. For example, Q can be used as an importance sampling heuristic for faster equilibration of P. Two mechanisms for such speedup are (a) proposing large steps to get out of local modes (local minima of the energy), which can speed things up exponentially by tunneling through energy barriers, and (b) propogating information long distances over a local grid (the latter mechanism is often important in speeding up deterministic multigrid methods for solving partial differential equations).

In the classic Boltzmann machine, random variables were divided into visible (input and/or output) and hidden units. For supervised learning, the visible units can be divided into distinct input and output units, and we seek a $Q(x_{\text{output}}, x_{\text{input}})$ distribution that approximates $P(x_{\text{output}}, x_{\text{input}})$, so that $Q(x_{\text{output}} \mid x_{\text{input}})$ models an observed $P(x_{\text{output}} \mid x_{\text{input}})$ and also $Q(x_{\text{input}})$ models $P(x_{\text{input}})$. For example, the input could be an image with a spectral intensity vector at each pixel, and the output could be a vector of binary feature presence/absence decisions at each pixel. Clearly this sitution fits into our multiscale framework as the first hierarchical step, if the first restriction and prolongation relationships are constrained to be the identity map. Recursively, then, one can use the more general multiscale step to create a series of more compact approximating models and accelerate the original Boltzmann machine learning algorithm. From past experience (our own and others' with algebraic multigrid) this is likely to be most effective if the recursive invocation scheme is the "W" cycle that calls cheaper, coarser models much more frequently than fine ones.

Ultimately a converged multiscale method can be used (a) to quickly approximate $P(x_{\text{output}} \mid x_{\text{input}})$ and thus predict from or classify a potentially novel input, with no further learning, and (b) to provide steps towards a scientific understanding of the original joint distribution $P(x)$ in terms of optimal reduced models that approximate it well, particularly if weight sharing is also used.

5.3.3 Goal-directed approximation

Both of the distortion measures of (Equation 48) may be subsumed within a more general, goal-oriented approximation framework in which both x and y spaces are mapped to an essential space of the "observables" of interest for a particular purpose, wherein the distortion measure is defined. Let $\zeta_x(\{z\} \mid \{x_i\})$ be a *defined* relationship (a given conditional distribution) between fine-scale variables x and a set of observables $\{z\}$, and let $\zeta_y(\{z\} \mid \{y_a\})$ be the relationship, to be optimized, between coarse-scale variables $\{y_a\}$ and the same set of observables $\{z\}$. Then a suitable distortion measure for optimizing $\zeta_y(\{z\} \mid \{y_a\})$ and $Q(\{y_a\})$ is

$$D[P(\{x_i\}),\, Q(\{y_a\})] = D_{\text{KL}}\left[\int \zeta_x(\{z\} \mid \{x_i\})\, P(\{x_i\})\, d\{x_i\},\ \int \zeta_y(\{z\} \mid \{y_a\})\, Q(\{y_a\})\, d\{y_a\}\right] \tag{49}$$

If ζ_x is the identity then ζ_y is the prolongation map and (Equation 49) is equivalent to Equation 48a. If the z's can be mapped to the y variables, on the other hand, then ζ_x plays the role of a fixed (unoptimized) restriction map and optimizing (Equation 48b) with respect to Q is equivalent to optimizing (Equation 49) with respect to $\zeta_y\, Q$.

The formulation of (Equation 49) has similarities to an autoencoder or more generally the “Information Bottleneck” formalism of Tishby, with y and its distribution $Q(y)$ playing the role of a bottleneck in the mapping between x and z.

5.3.4 Generalization to time series

Everywhere in the foregoing, we may split a collection of variables such as x, y, or z into a pair such as $(x(t),\ x(t+\Delta t))$ and regard the modeled probability distribution as a conditional distribution $P(x(t+\Delta t)\,|\,x(t))$ that is independent of t (depends only on Δt) and defines a semigroup for a dynamical system - a stochastic process obeying:

$$\begin{gathered}
P(x(t+\Delta t)\,|\,x(t)) = P(x(\Delta t)\,|\,x(0)),\ 0 \leqslant \Delta t \\
P(x'(t)\,|\,x(t)) = \delta(x'(t) - x(t)) \\
P(x(t+\Delta t)\,|\,x(t)) = \int_{-\infty}^{\infty} \{d\,x(t')\}\, P(x(t+\Delta t)\,|\,x(t'))\, P(x(t')\,|\,x(t)),\ t \leqslant t' \leqslant t+\Delta t
\end{gathered} \tag{50}$$

The multiscale approximation setting then becomes that of Figure A1.

$$\begin{tikzcd}
y(t) \arrow[rr] \arrow[dr] & & y(t+\Delta) \arrow[dr] & \\
& z(t) \arrow[rr, dotted] & & z(t+\Delta) \\
x(t) \arrow[uu] \arrow[ur] \arrow[rr] & & x(t+\Delta) \arrow[uu] \arrow[ur] &
\end{tikzcd}$$

Figure A1.

We require that the joint distributions such as $P(x(t+\Delta t),\ x(t))$ take a specific form such as (Equation 44), and adapt the K-L distance measures proposed above for MRF learning accordingly. In this way we arrive at a fundamental multiscale model-reduction approach to the prediction of time series, including the image sequences in our featured application domain.

5.4 Relation of cluster tree to Chinese Restaurant Process

5.4.1 Dirichlet and Chinese Restaurant processes

The stick-breaking construction of a Dirichlet process can be expressed with this discrete-time grammar (following [32]):

```
**grammar** (discrete-time) $DP$ (start($N$) $\to$ {cluster($i$, $\theta_k$, $\pi_k$) | $1 \leqslant k < \infty$}) {
        start($N$) $\to$ cluster'(0, 0, 0, 1, 0)
        cluster'($k$, $\theta_k$, $\beta_k$, $\Xi_k$, $\pi_k$) $\to$ cluster($k$, $\theta_k$, $\pi_k$),
                cluster'($k+1$, $\theta_{k+1}$, $\beta_{k+1}$, $\Xi_{k+1}$, $\pi_{k+1}$)
            **with** $\beta_{k+1} \sim \text{Beta}(\cdot\,|\,1,\alpha) = (\Gamma(1+\alpha)/\Gamma(\alpha))\,(\beta_{k+1})^{\alpha-1}$
            with $G_0(\theta_k)$
            where $\pi_{k+1} = \beta_{k+1}\,\Xi_k$
            where $\Xi_{k+1} = (1-\beta_{k+1})\,\Xi_k$
}
```

Then the Chinese restaurant process for cluster generation is:

```
grammar (discrete-time) CRP (start($N$) $\to$ {sample($x$) | $1 \leqslant k \leqslant N$}) {
        start($N$) $\to$ samples($N$), {cluster($k$, $\theta_k$, $\pi_k$) | $1 \leqslant k < \infty$} via DP
        samples($N$), $C$ = {cluster($i$, $\theta_k$, $\pi_k$) | $1 \leqslant k < \infty$} $\to$
                samples($N-1$), $C$, sample'($\theta_k$)
            with $\pi_k$
            subject to $N > 0$
        sample'($\hat{\theta}$) $\to$ sample($x$) with p($\cdot|\hat{\theta}$)
}
```

5.4.2 Cluster tree

The *clustergen* grammars can be specialized and limited so as to function in a very similar manner to DP and CRP above, with a Binomial Beta substituted for the Beta distribution. However, clustergen determines a more general family of distributions. For example one can control the histogram of cluster sizes.

Unwind *rclustergen* and limit to just two levels, but otherwise maintain equivalence to *rclustergen*:

```
grammar (discrete-time) L1clustergen (start($N$) $\to$ {cluster($i$, $\theta_k$, $\pi_k$) | $1 \leqslant k \leqslant n$}) {
        start($N$) $\to$ {cluster($k$, $\theta_k$, $\pi_k$) | $1 \leqslant k \leqslant n$}
            with $q_1(n)$
            with $\prod_k \phi(\theta_k \mid 0)$
            with $\Pr(\{\gamma_k \equiv n_k / N \mid 1 \leqslant k \leqslant n\} \mid N, n)$
                $= \delta\big(N - \sum_{j=1}^{n} n_j\big) \prod_{j=1}^{n} q_2(n_j) / \mathrm{Coef}[z^N, [g_2(z)]^n]$
            where $\pi_k = \mathrm{rank}_k(\mathrm{sort}(\{\gamma_k \mid 1 \leqslant k \leqslant n\}))$
                    // or just report unsorted clusters
            // optionally sample N too: with $N \sim N^{-\nu} / \varsigma(\nu)$
}
```

Exact version:

```
grammar (discrete-time) L2clustergen (start($N$) $\to$ {sample($i$, $\theta_i$, $\pi_i$) | $1 \leqslant i \leqslant N$}) {
        start($N$) $\to$ samples($N$), {cluster($i$, $\theta_k$, $\pi_k$) | $1 \leqslant k \leqslant n$} via L1clustergen
        samples($N$), $C$ = {cluster($i$, $\theta_k$, $\pi_k$) | $1 \leqslant k < \infty$} $\to \bigcup_{1 \leqslant k \leqslant n}$ {sample'($\theta_k$) | $1 \leqslant i \leqslant n_k = \mathrm{Round}(\pi_k N)$}, $C$
        sample'($\hat{\theta}$) $\to$ sample($x$) with $\phi(\cdot|\hat{\theta})$
}
```

Multinomial resampling approximate version:

```
grammar (discrete-time) L2clustergen (start($N$) $\to$ {sample($i$, $\theta_i$, $\pi_i$) | $1 \leqslant i \leqslant N$}) {
    start($N$) $\to$ samples($N$), {cluster($i$, $\theta_k$, $\pi_k$) | $1 \leqslant k \leqslant n$} **via** *L1clustergen*
    samples($N$), $C$ = {cluster($i$, $\theta_k$, $\pi_k$) | $1 \leqslant k < \infty$} $\to$
        samples($N - 1$), $C$, sample'($\theta_k$)
            **with** $\pi_k$
            **subject to** $N > 0$
    sample'($\hat{\theta}$) $\to$ sample($x$) **with** $\phi(\cdot|\hat{\theta})$
}
```

Note the detailed mapping to *DP* and *CRP* grammars above.

The cluster generation procedure can again be serialized for implementability, as in the meaning-preserving transformation from *rclustergen* to *rseqclustergen*:

$$\Pr(\{n_k \mid 1 \leqslant k \leqslant j+1\} \mid N, n) = \Pr(n_{j+1} \mid \{n_k \mid 1 \leqslant k \leqslant j\}, N, n) \Pr(\{n_k \mid 1 \leqslant k \leqslant j\} \mid N, n)$$

$$\Pr(n_{j+1} \mid \{n_k \mid 1 \leqslant k \leqslant j\}, N, n) = \frac{\Pr(\{n_k \mid 1 \leqslant k \leqslant j+1\} \mid N, n)}{\Pr(\{n_k \mid 1 \leqslant k \leqslant j\} \mid N, n)} = \frac{\mathrm{Coef}\left[x^N \prod_{k=1}^{j+1} {y_k}^{n_k}, \prod_{k=1}^{n} g_2(x\, y_k)\right] \Big|_{y_{k>j+1}=1}}{\mathrm{Coef}\left[x^N \prod_{k=1}^{j} {y_k}^{n_k}, \prod_{k=1}^{n} g_2(x\, y_k)\right]\Big] \Big|_{y_{k>j}=1}}$$

5.4.3 Resulting functions

E.g. if q_2 = a geometric distribution, we can compute this solution and get a *Binomial-Beta distribution* that plays the role of the *Beta distribution* in the CRP below. This can be calculated as follows:

$$\Pr(n_{j+1} \mid \{n_k \mid 1 \leqslant k \leqslant j\}, N, n) =$$

$$\Pr\left(n_{j+1} \mid \tilde{N} = N - \sum_{k=1}^{j} n_k, n\right) = \mathrm{Bb}\left(n_{j+1} \mid \hat{\alpha} = 1, \hat{\beta} = n - j - 1, \ \hat{n} = \tilde{N} \equiv N - \sum_{k=1}^{j} n_k, n\right)$$

$$\langle n_{j+1} \rangle \equiv E(n_{j+1}) = \hat{n} \frac{\hat{\alpha}}{\hat{\alpha} + \hat{\beta}} = \frac{\tilde{N}}{n - j} \quad \text{(equal sized portions)}$$

$$\left\langle (n_{j+1} - \langle n_{j+1} \rangle)^2 \right\rangle \equiv \mathrm{Var}(n_{j+1}) = \hat{n} \frac{\hat{\alpha}\, \hat{\beta}}{\left(\hat{\alpha} + \hat{\beta}\right)^2} \frac{\hat{\alpha} + \hat{\beta} + \hat{n}}{\hat{\alpha} + \hat{\beta} + 1} = \frac{\tilde{N}(n - j - 1)}{(n - j)^2} \frac{\left(\tilde{N} + n - j\right)}{(n - j + 1)}$$

$$\text{(zero variance for } j + 1 = n\text{)}$$

Instead of a "stick-breaking construction", let's call this the "bread-breaking construction": each n_{j+1} in turn tries to take an equal share of the $\tilde{N}$remaining samples. Only the last one is able to do this with no variance, since he just takes everything left over.

But in the more interesting case of a power law, $g_2(z) = \mathrm{Li}_\alpha(z) / \zeta(\alpha)$, we need to calculate the numerator Coef[·] in the following expression:

$$\Pr(n_{j+1} \mid \{n_k \mid 1 \leqslant k \leqslant j\} \mid N, n) = \frac{(n_{j+1})^{-\alpha}}{\zeta(\alpha)} \frac{\mathrm{Coef}\left[x^{N - n_{j+1}}, (\mathrm{Li}_\alpha(z))^{n-j-1}\right]}{\mathrm{Coef}\left[x^{N - n_j}, (\mathrm{Li}_\alpha(z))^{n-j}\right]\Big]}$$

Let's compute $(\mathrm{Li}_\alpha(z) / \zeta(\alpha))^n$ using *Mathematica*:

Simplify[Series[Power[PolyLog[a, z], n], {z, 0, 9}]]

$$\begin{gathered}
z^n \Bigl(1 + 2^{-a}\, n\, z + \bigl(3^{-a}\, n + 2^{-1-2a}\, (-1+n)\, n\bigr)\, z^2 + \\
\Bigl(4^{-a}\, n + 6^{-a}\, (-1+n)\, n + \frac{1}{3}\, 2^{-1-3a}\, (-2+n)\, (-1+n)\, n\Bigr)\, z^3 + \\
\frac{1}{24}\, \bigl(24\, 5^{-a} + 12\, \bigl(2^{1-3a} + 9^{-a}\bigr)\, (-1+n) + 12^{1-a}\, (-2+n)\, (-1+n) + 2^{-4a}\, (-3+n)\, (-2+n)\, (-1+n)\bigr) \\
n\, z^4 + \frac{1}{120}\, \bigl(5\, 2^{3-a}\, 3^{1-a} + 2^{3-2a}\, 15^{1-a}\, (5^a + 6^a)\, (-1+n) + 5\, 12^{1-2a}\, (8^a + 9^a)\, (-2+n)\, (-1+n) + \\
5\, 2^{2-3a}\, 3^{-a}\, (-3+n)\, (-2+n)\, (-1+n) + 2^{-5a}\, (-4+n)\, (-3+n)\, (-2+n)\, (-1+n)\bigr)\, n\, z^5 + \ldots\Bigr)
\end{gathered}$$

Rewrite by hand to reveal patterns that may be the key to solving it:

$$\begin{gathered}
z^n \Bigl\{\Bigl[1 + \bigl[2^{-a}\ z + 3^{-a}\, z^2 + 4^{-a}\, z^3 + 5^{-a}\, z^4 +\ 6^{-a}\, z^5 + \ldots\bigr]\, n + \\
\frac{1}{2}\bigl[4^{-a}\, z^2 + 2\ 6^{-a}\, z^3 +\ (2\ 8^{-a} + 9^{-a})\, z^4 +\ (10^{-a} + 12^{-a})\, z^5 + (2\ 12^{-a} + 2\ 15^{-a} + 16^{-a})\, z^6 + \ldots\bigr] \\
(-1+n)\, n\ + \frac{1}{6}\bigl[8^{-a}\, z^3 + 3\ 12^{-a}\, z^4 + 3\ \ (16^{-a} + 18\, -^{a})\, z^5 + (\,3\ 20^{-a}\ + 6\ 24^{-a} + 27^{-a})\, z^6 + \ldots\bigr] \\
(-2+n)\, (-1+n)\, n\ + \\
\frac{1}{24}\bigl[16^{-a}\ z^4 + 4\ 24^{-a}\ z^5 + 2\, (2\ 32^{-a} + 3\ 36^{-a})\, z^6 + \ldots\bigr]\, (-3+n)\, (-2+n)\, (-1+n)\, n \ldots\Bigr\}
\end{gathered}$$