Dependency Graphs

Oliver Richardson, oli@cs.cornell.edu

November 27, 2019

Abstract

We introduce Probabilistic Dependency Graphs, which can be regarded both as a probabilistic graphical model, and as a collection of soft, local, constraints. The additional representational flexibility allows us to represent both under and over-constrained belief states, as well as a modularity that makes the models easier to modify. They also have a clean theoretical backing, and reduce appropriately to existing, commonly used representations such as Bayesian Networks and constraint graphs. Some algorithms, notably including belief propagation, lift to the more general setting.

1 Introduction

I'm not exactly sure how to structure this, but the most important things, that I think could be centerpieces of a story are: I wouldn't do *any* of this. It's too complicated, going off into too many different directions. See below for what I would suggest.

1. Inconsistency is an important resource and can be viewed as the driver of change in many areas. However, the appeal of avoiding being ever wrong by design and fondness of soundness have occluded this in the past.

   (a) Anything that is shorthand for a distribution (e.g., BNs, Factor Graphs, Markov Networks), or lets you be strictly more agnostic about things cannot represent inconsistency. The same goes for many other notions of uncertainty: belief functions, and sets of probability distributions are under-constrained from the perspective of wanting a distribution. This seems to be the standard angle of attack against being a Bayesian—but it also seems inevitable that at some point end up with beliefs that aren't compatible with one another, even if you're extremely careful, because your possible worlds allow you to have two distinct beliefs about one thing. **This is the interaction effect between this and the second point**: your "possible worlds" might be impossible possible worlds.

   (b) Eliminating it in the representation means that the only way to interact with it is to resolve it immediately, which may be worse than waiting until there's new information.

   (c) Because the representation is so constrained, tasks that really don't need to be dealing with consistency (like putting two pieces of knowledge together, or updating) end up having to do this "every time they save", and in a way that is more ham-fisted than doing it with a dedicated algorithm. It also dramatically increases the computational cost required to do such things.

   The intermediate steps are often already useful enough to get the information you care about, and so jamming everything back into the same constricted representation is not really necessary. Also, the rigidity of the representation means there is no language for analyzing what happens between "commits" (e.g., what an agent's mental state looks like if halfway through a Bayesian update).

   (d) For cognitively bounded agents, the step above is often intractable: calculating normalization constants in Markov Networks and factor graphs is NP hard, as are constraint satisfiability problems. I would say that our approach seems cognitively simpler.

No need to make such strong claims I think it suffices to observe that the standard graphical representations of uncertainty can't represent inconsistency.

Again, I'm not sure that this is a point worth making. People will agree that the standard approaches can't represent inconsistency. That's good enough for our purposes.

I would simplify this by just saying "we can't represent "local" information well, giving the example of smoking causes cancer + tanning causes cancer.

Although inconsistency was our major motivation, I'm not sure that that's how I would write the introduction. Here's a possible alternative: There are a number of quite successful graphical representations of uncertainty. We're going to introduce one more. What makes our novel and interesting? Two things: (1) it can represent local information and (2) it can represent inconsistency. Here are some examples to show that both features are quite useful in practice. The examples will also give some intuition for our representation. [[GIVE EXAMPLES.]] Point out that our approach seems cognitively simpler. Discuss (at a high level) the two possible semantics for the language. Say that the "weighted probabilities" approach has two significant advantages over the more obvious "set of probability measures" approach: (1) It allows us to consider inconsistent scenarios, since the distributions do not have to be consistent with the graph. (2) It gives us a natural way of going from a local picture to a global one; there are various choices, and we can consider the "best" one(s). (3) It allows us to connect our approach to Bayesian networks (and factor graphs?) in a straightforward way.

I'm not quite sure what this means, but I would cut it.

(e) Bayesian updating is a greedy algorithm for finding truth. It takes one observation at a time and immediately uses all of the inconsistency in it to get a better picture of the truth. **This is only guaranteed to be independent of event order if the set of worlds is static**. Otherwise collecting all of the observations all at once and then reducing inconsistency globally could yield a different answer. Applications of Jeffrey's rule are not in general independent of event order even if the set of possible worlds is static.

Questions:

There may be, but we're not going to discuss it in this paper.

(a) Is there an interesting moral analog of Godel's 2nd theorem? Can I use this to argue for the potential for inconsistency?

(b) Can all inconsistency be framed as logical inconsistency? What does the language need to include so that this is true? I have no idea what this means, but I don't think we should discuss it.

2. Nobody's subjective "possible worlds" are static or strictly decrease over time. This is not due entirely to failure of perfect recall: people invent concepts, learn things about the world, etc. This causes a shifting of the underlying space of what is considered possible. Common representations of uncertainty and knowledge do not deal with this well, and often posit the existence of a fixed "true" set of possible worlds, which needs to be known by the modeler in order to make predictions. I do think we should talk about the domain "expanding", but this is not how I would introduce it.
3. Alternatively, I can paint a picture of defining a number of small domains where everything is locally consistent, but globally there are no guarantees; we only need to make local decisions and know things in local contexts, and so there's get away with not maintaining expensive global things like full joint distributions, so long as you also take care to clean up inconsistencies when they occur. Yes! This should be a centerpiece of our story.

   The process of sorting through local beliefs and going from local picture to a global one can be thought of as a "sheafification" of belief updating. NO!!!! No hint of category theory.

Inconsistency is bad. Believing a logically inconsistent formula can lead you to arbitrarily bad conclusions, having an infeasible set of constraints makes all answers you could give wrong, and having inconsistent preferences can lose you infinite money. We don't want to build inconsistent systems or agents with incoherent views of the world, and so, where possible, we design them so they cannot possibly be broken in this way. Suppose, for example, that we are trying to represent some quantity that must be a point on the unit circle. We could do it with an x and y coordinate, but this could be problematic because $x^2 + y^2$ might not be 1 — it would be safer and harder to go awry if we parameterize it by an angle $\theta \in [0, 2\pi)$ instead. In the absence of performance benefits (like needing to regularly use the y-coordinate and not wanting to compute a sine), why would we take the first approach, introducing a potentially complex data-invariant, when we could avoid it?

This line of thought, though common and defensible, is flawed if we are not perfectly confident in the design of both our system and the ways it can interact with the outside world. Using similar logic, we might ask ourselves: Why ask programmers for type annotations when all instructions are operationally well-defined at run-time? Why use extra training data if there's already enough there to specify a function? Why estimate a quantity in two ways when they will yield different answers? Why repeat and rephrase your ideas when this could make you contradict yourself? Why write test cases when they could fail and make the project inconsistent? Why conduct an experiment if it could just end up contradicting your current knowledge?

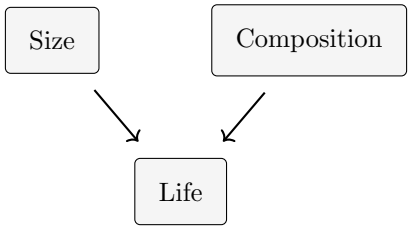
These questions may seem silly, but there is a satisfying information theoretic answer to all of them: redundancy, though costly, is the primary tool that we use to combat the possibility of being wrong. Maintaining data invariants can be expensive but provides diagnostic information; in the example above, settings of x and y that don't lie on the unit circle provide diagnostic information that something has gone wrong. In many cases, it is also possible to paper over problems by forcibly re-instating local data invariants: for instance, we could re-normalize any values of x and y (so long as $xy \neq 0$; we can chose an arbitrary point otherwise) at every step. While this would reduce inconsistency, it also hides red flags.

Using a Bayesian Network to represent a probability distribution is like representing a circle with $\theta \in [0, 2\pi)$. By construction, the result must be a distribution, and nothing can possibly go wrong so long as we can always decide on exactly one distribution which is sufficient for our purposes.

The process of mechanistically forcing invariants is homologous to the standard practice for factor graphs: practitioners will often just assume that the density it defines is normalizable, and either forcibly re-normalize or cleverly avoid computing the normalization constant while still assuming that one exists; behavior is usually left unspecified in the unlikely event that it is not defined or zero.

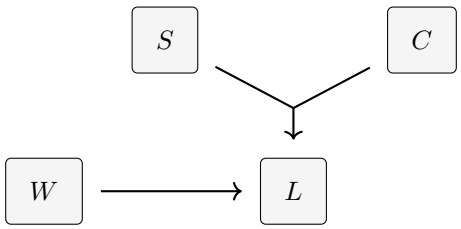
It is clear that, while inconsistency is bad, being able to identify it is extremely useful. To that end, we introduce a more general representation of knowledge and uncertainty which can be both under and over-constrained (from the perspective of producing a probability distribution), can be used to emulate a large class of both probabilistic models and constraint sets, and enjoys many additional properties which make them more useful than the specific variants.

Example 1.1. Suppose we have a belief about how size and composition affect the habitability of a planet: say we're astrobiologists, and we have some sense of how likely we are to find life on a given planet, supposing we knew how big it is and whether it's mostly made of rocks or gas. That is to say that we have a conditional probability table $\Pr(\text{Life} \mid \text{Size}, \text{Composition})$, which we are used to graphically depicting like this:



This picture looks like a Bayesian Network, which is somewhat misleading. In order to interpret this graph as a quantitative BN, we also would need probability distributions over Size and Composition — things we may not be willing to give.[1] From a probabilistic perspective, our beliefs are under-specified. This is a problem (though not a novel one) with Bayesian networks.

A bigger problem occurs when our biologist friend reminds us that life requires water, and gives us a probability estimate for the existence of life on a planet, with and without water. We trust this friend completely, and totally believe these probabilities. Unfortunately, but there's no way to incorporate it into our picture, because we don't know what the correlations are between water, size, and composition; neither are we prepared to give a probability of live given a full description of the three, and may not even have the space to keep such a thing.[2] Let S, C, W, L be shortenings of Size, Composition, Water, and Life, respectively. Intuitively, we want to draw instead a picture that looks more like this:



which represents having two conditional probability tables on L: one from $S \times C$ and the other from W. This would allow us to combine the two facts that we know, without also providing more information than we're prepared to equivocate on. It is worth noting that there is now a possibility of being inconsistent, in the sense that is possible to specify the conditional distributions in the links in such a way that no joint distribution on all variables will marginalize out to them — for instance, if all estimates of L from the W are strictly smaller than any probability estimate of L from $S \times C$.

[1] Situations like these are incredibly common and important. In statistical learning theory, this is the difference between a generative and a discriminative (or conditional) model

[2] If *Size* and *Composition* each have $\approx \sqrt{N}$ elements, and *Water* had $\approx N$ elements, it would be $O(N^2)$ to store a full joint table, compared to $O(N)$ for the two individual ones.

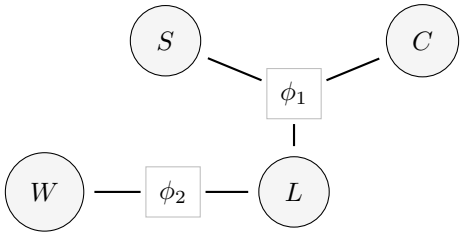
The probabilists in us might not be willing to so easily give up the notion that these data define a probability distribution, at least implicitly. Perhaps the reason Bayesian networks are insufficient to represent this epistemic state is because the state is not a distribution and hence invalid; maybe there's something simple we can do to turn it into a one? It turns out that we can (almost always) get a distribution and simultaneously commit to preserving the relative ratios of the specified probabilities within the links, and even more clearly exposing the independence structure that we think of Bayesian Networks as giving.

I'm not sure exactly what this means, but I think we can defer it to the technical discussion in later sections, if it's to go anywhere.

This can be done by treating the conditional distributions $p(L = l \mid S = s, C = c)$ and $p(L = l \mid W = w)$ as *factors*, which multiplied together give the relative probability density of any setting of variables $S \times C \times W \times L$

$$\Pr(s, c, w, l) \propto \phi_1(s, c, l)\phi_2(w, l)$$

where $\phi_1(s, c, l) = p(L = l \mid S = s, C = c)$, $\phi_2(w, l) = p(L = l \mid W = w)$. This can also be represented graphically, with a *factor graph*—a commonly used graphical model hailed as a strict generalization of Bayesian Networks and Markov networks.



We now have a distribution that combines our beliefs, but this is not exactly what we had in mind earlier. Beyond simply the inevitable effects of requiring a distribution, such as curing of us of our lack of distribution on S, C, and W, this factor representation has additional unwanted effects: ??

1. We can't weight the pieces of information differently. Though the scale of each factor ϕ_i gives us a degree of freedom in which to encode this information, it cannot be used, as $(a\phi_1)(b\phi_2) = (ab)(\phi_1\phi_2)$, and the aggregate coefficient ab too is normalized out to form the distribution.

2. The resulting picture does not encode conditional probabilities in quite the way that we had wanted: now updating on S does not preserve $L \mid C, S$, bringing L along as required, but rather does something unclear and very global: we've lost the dependency structure we had in the first few pictures. Relatedly, we have lost the directedness of the edges, and with it, hope that the edges represent anything causal. Furthermore, the addition of new factors can dramatically change the meanings of existing ones. For all of these reasons, it is incredibly difficult to interpret part of the graph by itself.For instance, knowing the joint distribution does not determine the values of the factors.

3. If at least one factor is zero for every setting of S, C, W, L, no distribution is defined — in the face of inconsistency, the entire formalism ceases to work at all.

In the introduction I would definitely say that factor graphs also try to represent "local information" as we do. But I would say this succinctly. Here's a quick pass
However, our approach has a number of advantages: (1) we can weight pieces of information differently [how do we do this?]; (2) factor graphs can't deal well wi
inconsistency; if at least one factor is 0 for every setting of the variables, no distribution is defined; (3) updating does not preserve the depending structure well (w
return to this point in Section ??

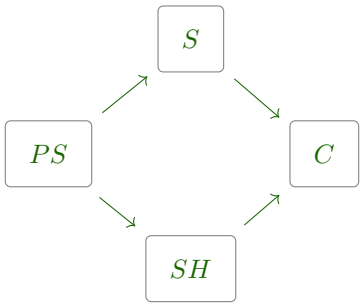
While offer a solution of great generality, they sacrifice a great deal of interpretability and destroy a lot of the important internal features of our original belief representation, so that they can represent distributions.

This is not good, but much worse is the way that they sweep under the rug issues wherever possible. In example 1.1,

Akin to a in some ways destroy inconsistency that may have been a big deal had it come to the surface.

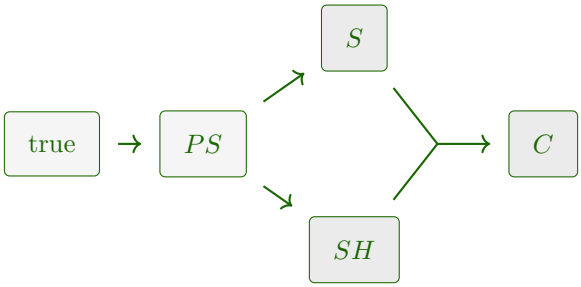
Example 1.2. Consider the classic example used to introduce Bayesian nets, in which the four variables are interest are booleans indicating whether a person (C) develops cancer, (S) smokes, (SH) is exposed to 2nd

hand smoke, and (PS) has parents who smoke, presented graphically as follows:



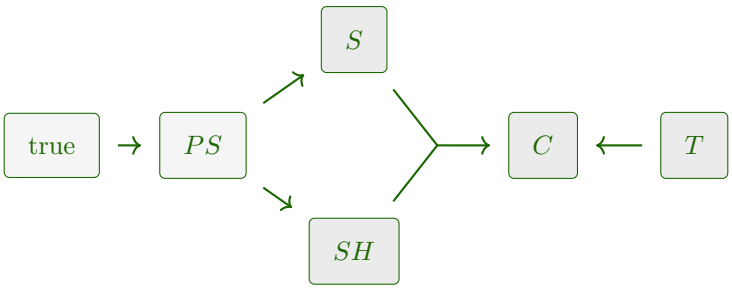
This is a compact representation of a joint distribution over all four variables, which achieves compactness by taking advantage of independence between variables. It encodes an assumption that every node is independent of its non-descendants given its parents.

Most of the time, we do not make the independence assumption because we know for certain that the variables are independent; rather, we just suspect that the identified links are by much more important than the others. Determining for sure that smoking and second hand smoke are independent, controlling for parents' smoking habits, would extremely difficult, and to do properly would require much more empiricism to validate. Why even bother jumping through this hoop? Because we wanted a BN to be shorthand for a probability distribution. But we have freed ourselves from these shackles, and make no such assumption.



The node on the far left is a special node which only takes one value, and allows us to represent unconditional distributions as arrows, visually making clear the difference between a lack of distribution, and an unconditional one.

Now, suppose you read a very thorough empirical study which demonstrates that people who use tanning beds have a 10% incidence of cancer, compared with 1% in the control. Just as in the previous example, this cannot be encoded directly into the Bayesian Network. The PDG on the other hand, has no trouble, and is simply the union of the two pieces of information:



As I said, I like this example, but I think you should shorten it significantly. Just give the standard Bayesian network for smoking and talk about would would be involved for BN folks vs us in adding the information about tanning beds to the picture.

Note that the right half of the diagram (shaded slightly darker) has the same topology as in example 1.1. △

Benefits of this representation:

All these point (at least the ones that I understand) should be made in the intro. It may make sense to have this summary at the end of the intro.

1. We can represent both over-constrained and under-constrained mental states, both of which we argue are an important component of an agent's state.
2. Over-constrained models may be inconsistent; such inconsistencies provide a natural way of prescribing changes in mental state. Moreover, many standard algorithms, such as belief updating via Jeffrey's rule, as well as marginalization algorithms such as belief propagation, can be regarded as special cases of consistency reduction.
3. We can emulate the functionality of not only other graphical models (such as Bayesian Networks, and to a large extent, factor graphs), but also other non-probabilistic notions of uncertainty.
4. The local interpretation of arrows makes it much less invasive to add, remove, and partially interpret parts of the model, compared to other graphical models.
5. This modularity makes it possible to add explicit rules to embed logic within the model
6. By allowing agents to merge, split, and compress variables, we also make it possible for agents to design their own representations. With these tools, in conjunction with consistency
7. In contrast with a simple collection of constraints, inconsistencies are local, and individual mistakes have limited impact on expectations.

What does this mean? Do you have a good example?

I'm not sure what this means. Why is it more true for our representation than others?

This is an important point. Can you give an example of this?

2 Related Work

3 Worlds

Other than allowing for inconsistency, the biggest difference between Probabilistic Dependency Graphs and other graphical models is their interaction with the underlying space: a PDG .

The standard approach to probabilistic modeling is to start by selecting a measurable space of possible outcomes Ω, and then put a normalized measure on it and compute desired quantities with it. Before you can begin to think about random variables, which are defined as set $X_i : \Omega \to V_i$ from outcomes to the set of values V_i that X_i can take on, you have to specify Ω. This construction works well, so long as Ω is large enough to express everything you ever cared to conceptualize. Because agents are expected to have probability distributions over Ω, the set of worlds that they consider possible must effectively stay constant over time, to use mechanisms such as conditioning as a sole way of changing a mental state.

Still, a we might wiggle our way around this using only classical probability: we could say that Ω is some very large set of outcomes that is guaranteed to be expressive enough to capture anything we care about, and then

To be clear, we've already given up on the possibility of being a Bayesian, because we clearly don't have priors on arbitrary concepts we haven't considered yet if we don't even know the extent of the space, but we might be able to do this with some under-constrained mechanism.

This strategy, works so long as you are an omniscient modeler. If you are modeling a system in which you know the set of all possible outcomes, either implicitly or explicitly, you can just collect them and use this to be Ω, marginalize out appropriately, and let agents figure out their own distributions on subspaces of Ω. Still, this is not entirely satisfying, for several reasons.

[todo: *most of these are unfinished thoughts, some need to be deleted*]

1. While it is true that the there will be subspaces of Ω which are isomorphic to the sets of worlds W_i that the agents are modeling in their heads, the embedding is not at all clear [todo:]
2. There may not be an omniscient modeler, and even if there were one, it seems very strange for an agent to have any access to it. Suppose you are using probabilities to describe real uncertainties in your life. To do this the standard way, you need to chose the subspace of the one true Ω [todo: *why is this problematic? for reasons other than inability to change?*]
3. Agents can never gain access via any standard mechanism to new worlds. There's no principled way to add worlds from Ω to W_i. Effectively, they can never learn new concepts.

4. Any updates must be done on the entire space of things you consider possible [todo: *response: of course, this is all handled implicitly, so it's taken care of*].
5. While agents are free to merge multiple states of Ω into a single state in W_i, they cannot do the reverse: an agent cannot have a finer granularity than Ω for discerning events. This would . This also implies that agents are logically omniscient.

For all of these reasons, we take the view that probability should be thought of subjectively,

At the same time, starting with a set of random variables $\{X_i\}$, and setting $\Omega = \prod_i V_i$ to be every possible assignment of variables is also an abuse of the word "possible".

4 Equivalent Definitions and Variants

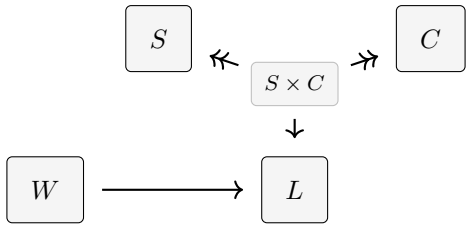
Definition 4.1. A *strict Probabilistic Dependency Graph* is a tuple $(N, L, \boldsymbol{\mu}, \mathcal{V})$ where

- $\mathcal{N} : \mathbf{Set}$ is a finite collection of nodes
- $\mathcal{L} \subseteq N \times N$ is a set of directed links between nodes
- $\mathcal{V} : \mathcal{N} \to \mathbf{MeasSet}$ is an N-indexed family of measurable sets, representing the values that a node can take
- $\boldsymbol{\mu} : ((A, B) : \mathcal{L}) \to \mathcal{V}(A) \to \Delta(\mathcal{V}(B))$ is a family of conditional probability distributions on $\mathcal{V}(B)$ indexed by the values of A for every link $(A, B) \in \mathcal{L}$

The definition of $\boldsymbol{\mu}$ is probably more familiar than it looks. If every $\mathcal{V}(N)$ is finite with all subsets measurable, then $\boldsymbol{\mu}_{A,B}$ is just a conditional probability table, just as in a Bayesian Network. For those more familiar with stochastic processes, this is a stochastic matrix.

The definition $\boldsymbol{\mu}$ is slightly over-simplified if not everything is measurable. More generally if $\mathcal{V}(A) = (X, \mathcal{A})$ and $\mathcal{V}(B) = (Y, \mathcal{B})$, then for any link $L \in \mathcal{L}$, we're really referring to a function $\boldsymbol{\mu}_L : X \times \mathcal{B} \to [0, 1]$ such that $\boldsymbol{\mu}_L(x, -) : \mathcal{B} \to [0, 1]$ is a probability distribution and $\boldsymbol{\mu}_L(-, S)^{-1}(R) \in \mathcal{A}$ for any $S \in \mathcal{B}$, and measurable subset $R \subseteq [0, 1]$, technically making $\boldsymbol{\mu}_{A,B}$ a *Markov Kernel* from A to B.

Example 4.1. In example 1.1, we displayed the arrow from S and C to L as a directed hyper-edge. While we would like to maintain this intuition, it turns out that we can simplify our formalism by de-sugaring this picture into the following:



The double headed arrows are for degenerate conditional distributions, which are fully deterministic, but for now this is not terribly relevant. We can now present this PDG formally with the elements specified in definition 4.1; below we assume everything is measurable and omit this part of the formalism.

$$\mathcal{N} = \{S,\ C,\ L,\ W,\ S \times C\}$$
$$\mathcal{L} = \{(S \times C, L), (W, L), (S \times C, S), (S \times C, C)\}$$
$$\mathcal{V} = \begin{cases} \mathcal{V}(S) & = \{big, small\} \\ \mathcal{V}(C) & = \{rocky, gasseous\} \\ \mathcal{V}(L) & = \{l, \neg l\} \\ \mathcal{V}(W) & = \{none, some, mostly\} \\ \mathcal{V}(S \times C) & = \mathcal{V}(S) \times \mathcal{V}(C) \end{cases}$$

$$\boldsymbol{\mu} = \begin{cases} \boldsymbol{\mu}[S \times C, L] = \begin{array}{c} \quad l \quad \neg l \\ \left[\begin{array}{cc} .1 & .9 \\ .2 & .8 \\ .05 & 0.95 \\ 0.00001 & 0.99999 \end{array}\right] \end{array} \begin{array}{l} \text{big, rocky} \\ \text{small,rocky} \\ \text{big, gasseous} \\ \text{small,gasseous} \end{array} \qquad \boldsymbol{\mu}[W, L] = \begin{array}{c} \quad l \quad \neg l \\ \left[\begin{array}{cc} 0 & 1 \\ .005 & .995 \\ .05 & 0.95 \end{array}\right] \end{array} \begin{array}{l} \text{none} \\ \text{some} \\ \text{mostly} \end{array} \\ \boldsymbol{\mu}[S \times C, C] = \begin{array}{c} \text{rocky} \quad \text{gasseous} \\ \left[\begin{array}{cc} 1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 1 \end{array}\right] \end{array} \begin{array}{l} \text{big, rocky} \\ \text{small,rocky} \\ \text{big, gasseous} \\ \text{small,gasseous} \end{array} \qquad \boldsymbol{\mu}[S \times C, S] = \begin{array}{c} \text{small} \quad \text{big} \\ \left[\begin{array}{cc} 0 & 1 \\ 1 & 0 \\ 0 & 1 \\ 1 & 0 \end{array}\right] \end{array} \begin{array}{l} \text{big, rocky} \\ \text{small,rocky} \\ \text{big, gasseous} \\ \text{small,gasseous} \end{array} \end{cases}$$

△

This works pretty well for the two edges that we described before, but the structural overhead of the additional de-sugaring: the $\boldsymbol{\mu}[S \times C \to S]$ and $\boldsymbol{\mu}[S \times C \to C]$ tables, as well as the set $\mathcal{V}(S \times C)$ seem like they didn't need to be specified, and one might even feel that it would be a mistake to allow any other table. Some reasons for this design decision include:

- It is easier to prove things about graphs than directed hyper-graphs. Similarly, defining directed paths is a lot simpler.
- We can eliminate the clunkiness by fusing the model with an algebra, as in section 10 — which will give us a lot more than modeling the hyper-edges directly.
- We will eventually also want to allow for the possibility of keeping only a relaxed, approximate representation of $\mathcal{V}$ and $\boldsymbol{\mu}$, and in particular, of the ones constructed logically in this way. By specifying them explicitly for now, we will have to do less work to regain manual control in **??**

4.1 Alternate Presentations

4.1.1 Random Variables

If $\mathcal{W} = (W, \mathcal{F}, \mu)$ is a measure space, and $\mathcal{X} = \{X_i : W \to \mathcal{V}(X_i)\}_{i \in I}$ is a collection of measurable random variables on W,[3] and $\mathcal{L} \subseteq I \times I$ is a collection of pairs of variables such that the agent [`todo:` *what is a way of phrasing this that doesn't sound like it's shoehorned in? $\mathcal{L}$ really can represent anything an agent knows. Any subjective conditional probability distribution μ' such that the only measurable subsets are "axis aligned",*

[3] that is: $\mathcal{V}(X_i)$ is a measurable space, taking the form $(D, \mathcal{D})$, and $X_i : W \to D$ is a set function such that for every $B \in \mathcal{D}$, the set $X_i^{-1}(B) \in \mathcal{F}$

in that they involve queries on only one variable, can be represented by $\mathcal{L}$, and for other queries we can simply change variables.], we call $(\mathcal{W}, \mathcal{X})$ an *ensemble*.

Proposition 4.1. *There is a natural correspondence between strict PDGs as defined in definition 4.1, and ensembles such that* [`todo:` *spell this out explicitly to avoid vague categorical intuition*] *. . . μ's are defined on same set and produce same values.*

Proof. /outline: On the one hand, $(\prod_{N\in\mathcal{N}} \mathcal{V}(N).\text{set}, \bigotimes_{N\in\mathcal{N}} \mathcal{V}(N).\text{algebra}, \boldsymbol{\mu})$ is a measure space, with $\{X_N = \pi_N : (\prod \mathcal{V}(N')) \to \mathcal{V}(N)\}_{N\in\mathcal{N}}$ a set of random variables

and on the other, $(I, \mathcal{L}, \mathcal{X}', \mu|_{\mathcal{L}})$ is a strict PDG. □

This is the technical underpinning of our flippant, noncommittal treatment of possible worlds: any time we are thinking in terms of random variables or probability distributions on a fixed set W, we can instead reduce

The complexity of the representation is $O(XV + LV^2)$, compared to $O(XW)$

4.2 Sub-stochastic Transitions

In this section we will see why we called the object in definition 4.1 a *strict* PDG. Sometimes an otherwise very useful variable might not apply in a small percentage of cases; in this case, we want a way of putting all of the extra probability mass in a "something else happened" bucket, giving us effectively a sub-stochastic matrix, or a a lower probability on singletons. For instance, the variable describing whether or not your answer is correct doesn't make sense if you weren't solving problems; the amount of money in your wallet doesn't make sense if you don't own one, and so forth. So now, when you're trying to predict the probability of certain amounts of money in your wallet, some of the probability mass needs to go into the "not applicable / something else" bucket.

There are several closely concepts that we will be able to employ with our framework after integrating them

1. Allowing random variables to be partial, rather than total functions of W.
2. Relaxing the requirement $\int_W \mu \mathrm{d}w = 1$ to $\int_W \mu \mathrm{d}w \leq 1$
3. Requiring that matrices be sub-stochastic, rather than stochastic
4. Replacing probabilities, with the more general class of lower probability measures.

This generalization is useful, but our primary motivation for this generalization is so that we can represent implication, and thus a weakening of knowledge as it travels through our graph, in a way that is not just entropy (which might not be distinguishable from certain knowledge of a high entropy distribution otherwise).

At first glance, though, it might not be clear why this particular weakening buys us anything at all, because we can always just add the "something else" bucket $\bullet$, to $\mathcal{V}(X)$ for each X, and come up with a new strict PDG. A variable which might not make sense can always take a `null` value, and so now the set of possible is once again exhaustive. From the perspective of providing conditional distributions, however, this resolution poses a problem: our marginals now require us to estimate distributions from a null value— this is problematic, as a big part of the reason we've been using links to avoid assigning probabilities to everything. Suppose you are trying to represent the belief that you're happier when you get the right answer as a marginal link $L[\text{RightAns} \to ☺]$. We now need a distribution on happiness when you get the right answer, when you get the wrong answer, and also for when $\bullet$. Why might it not be applicable? Are you not solving problems because you're skiing? Because you've been injured? Maybe you are solving problems but there are multiple right answers? You can't just answer with a prior over happiness if you want to have consistent beliefs, because solving problems and happiness might be correlated. One *could* have such a thing but it seems unreasonable not to be able to express a belief about "does the right answer make you happy?" without also answering the much more difficult question, "how happy are you when 'the right answer' is not applicable to your current situation?"

To see how this increases our expressive power, suppose A, B are binary variables (taking values $a, \bar{a}$ and $b, \bar{b}$ respectively). While we can easily easily represent $A = B$, $A = \neg B$ as stochastic matrices,

$$p(B \mid A) = \begin{matrix} \\ a \\ \bar{a} \end{matrix}\!\!\begin{matrix} \begin{matrix} b & \bar{b} \end{matrix} \\ \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \end{matrix} \quad \text{and} \quad p(B \mid A) = \begin{matrix} \\ a \\ \bar{a} \end{matrix}\!\!\begin{matrix} \begin{matrix} b & \bar{b} \end{matrix} \\ \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \end{matrix}$$

we cannot (via stochastic matrices) represent an assertion that $A \Rightarrow B$ without also giving a distribution over B given $\bar{a}$. One strategy is a uniform prior (used in [**logicalinduction**]), but this can easily lead to avoidable inconsistencies — perhaps for totally different reasons you have very good reason to believe that the true distribution of B is true in 90% of cases; you don't want an arbitrary assumption of a prior competing with actual knowledge.

For this reason, we drop the requirement that our null element, $\bullet$, indexes a distribution in marginals. Below is an example of transition matrix $A \to B$ including the extra element. As mentioned, the last row is not something we are keeping track of.

$$\begin{matrix} \\ a_0 \\ a_1 \\ a_2 \\ \bullet \end{matrix}\!\!\begin{matrix} \begin{matrix} b_0 & b_1 & \bullet \end{matrix} \\ \left[\begin{array}{ccc} .2 & .1 & 0.7 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \\ \hline .2 & .6 & 0 \end{array}\right] \end{matrix}$$

Furthermore, because the final column is just whatever is necessary to make the rows sum to 1, we don't need to keep that either; as a result, it is sufficient to keep a smaller matrix without any $\bullet$-indices; the only price that we pay is that this matrix is *sub*-stochastic rather than stochastic: its row entries sum to at most 1, rather than exactly 1. Composition works just as before; the product of sub-stochastic matrices is sub-stochastic. A probability distribution alone, and by extension a standard Bayesian network cannot do this — because we require the look-up tables to exactly match all possible values, we can't drop any without totally giving up on any world which looks like that.

4.2.1 Relation to Partial Functions of W

4.2.2 Reduction to Lower Probability Measures

4.3 Categorical Definition

[note: *I will not put any time into this, as it's not going in the paper, but it's here as a placeholder, and I'll list some reasons why this is worth thinking about.*] One reason this works out so nicely is every construction is universal. We can in fact give a simpler categorical presentation of PDGs for those who already know category theory. The highlights are as follows:

1. A PDG is an attention-shaped diagram in the Markov category. That is, functor from the free category generated by the graph $(\mathcal{N}, \mathcal{L})$ representing attention, to the Markov category. Indeed $\mathcal{V}$ is the action on objects, assigning each $\mathcal{N}$ to a measurable set, $\boldsymbol{\mu}$ is the action on morphisms, sending edges in $\mathcal{L}$ to Markov kernels between their associated objects.
   (a) Composition works out in general as we place no restrictions on anything, but
   (b) If every edge in $\mathcal{L}$ represents the causal structure of their relationship, then the image of the resulting diagram will be flat, and so effectively there will only be at most one, belief, and no possibility of conflict.
   (c) Interpreting with a different model of uncertainty (such as the powerset, giving us non-deterministic possibility) is simply an exchange of interpretation. However, for nice interaction with deterministic functions and logic, this notion of uncertainty must be a monad.

2. This highlights the role of the "qualitative" and "quantitative" versions of this framework (which work out much more cleanly than for BNs in a categorical sense)
3. A limit of this diagram is a space of worlds and all of the random variables as functions. A colimit is a the strongest thing that must be true according to the model (suspicion: this is somehow related to common knowledge). There is some strangeness about how samples work that I have not yet figured out.

5 Semantics

These graphs admit multiple semantics. As discussed in section **??**, we think of Probabilistic Dependency Graphs as being a representation of knowledge in and of themselves, rather than a compression of something more fundamental such as a probability distribution. Still, we will find it useful to interpret them in various ways: doing so will make it possible to compare them more directly with existing graphical models, which one thinks of as really just being compressed distributions. In this section, we would like to highlight three important semantics.

5.1 As Sets of Distributions

If the focus is on under-constrained models, then just as a BN represents a distribution on joint space, a PDG might might thought of as representing the set of all distributions that marginalize out to it exactly.

Definition 5.1. If $M = (\mathcal{N}, \mathcal{L}, \mathcal{V}, \boldsymbol{\mu})$ is a PDG, let $[\![M]\!]_{\mathbf{Set}}$ be the set of distributions over the variables in M consistent with $\boldsymbol{\mu}$ on every marginal. Formally,

$$[\![M]\!]_{\mathbf{Set}} := \left\{ \mu \in \Delta\left[\prod_{N \in \mathcal{N}} \mathcal{V}(N)\right] \;\middle|\; \begin{array}{r} \mu(B = b \mid A = a) = \boldsymbol{\mu}[A, B](b \mid a) \\ \text{for all } A, B \in \mathcal{L},\ a \in \mathcal{V}(A), \text{ and } b \in \mathcal{V}(B) \end{array} \right\}$$

5.2 As Distributions

To satisfy any lingering desire to compress all of the information to a single distribution, we also offer a way of interpreting a PDG as a single distribution.

Definition 5.2. If ℓ is a scoring function for probabilities, let

$$[\![M]\!]^{\ell}_{\mathbf{S}} = \left\{ \mu \in [\![M]\!]_{\mathbf{Set}} \;\middle|\; \forall \mu' \in [\![M]\!]_{\mathbf{Set}}.\ \ell(\mu) \leq \ell(\mu') \right\}$$

One particularly useful one for emulating Bayesians is the following one, maximizing entropy:

$$[\![M]\!]_{\substack{\mathbf{Max} \\ \mathbf{Ent}}} := [\![M]\!]^{-H(\cdot)}_{\mathbf{Set}}$$

6 Relations to Other Graphical Models

6.1 Bayesian Networks

6.2 Factor Graphs

7 Relations to Other Representations of Uncertainty

Probabilistic Dependency Graphs are far from the first formalism to provide a weaker notion of uncertainty than probability. Belief functions, inner measures, sets of probabilities, lower probabilities, weighted sets of probabilities, and plausibility measures have all been studied extensively in the past. One feature that each of these has in common is that they are under-specified, from the perspective of wanting probabilities for everything.

The natural question now becomes: to what do these under-constrained representations of belief correspond to under-constrained bits of a Probabilistic Dependency Graph?

7.1 Sets of Probability Measures

As we discuss in section 5.1

8 Using Inconsistency

Given a distribution μ, and a

9 Algorithms

9.1 Belief Propagation

10 Algebra

Definition 10.1. If σ is a signature, a σ-PDG M' on a PDG $M = (\mathcal{N}, \mathcal{L}, \mathcal{V}, \mu)$ is a Probabilistic Dependency Graph $(\mathcal{N}', \mathcal{L}', \mathcal{V}', \mu')$ such that

- $\mathcal{N}' := T_\sigma(\mathcal{N})$ is the term algebra for the signature σ over the alphabet $\Sigma = \mathcal{N}$.
- $\mathcal{L}' = \mathcal{L} \cup \mathcal{L}^\sigma$ is $\mathcal{L}$ extended with extra links for operations that are

Example 10.1. content △

11 Conclusions