Uncertainty with logical, procedural and relational languages

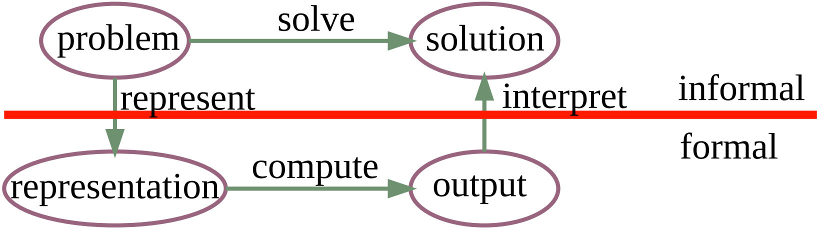
David Poole

Department of Computer Science,
University of British Columbia

UAI 2006 Tutorial

Outline

Knowledge Representation



What do we want in a representation?

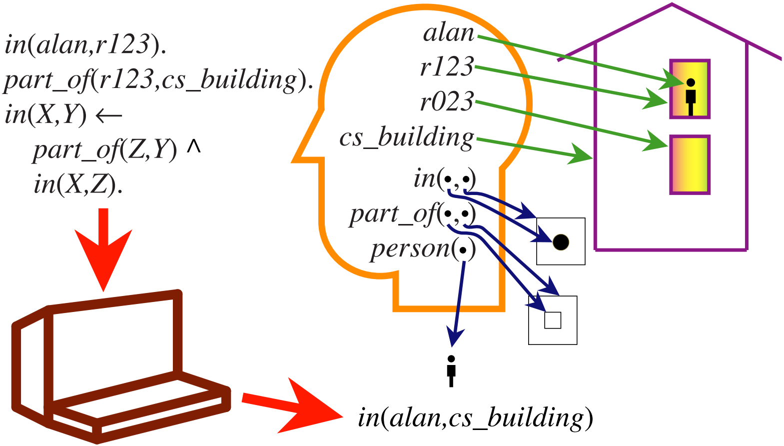
We want a representation to be

- rich enough to express the knowledge needed to solve the problem.
- as close to the problem as possible: compact, natural and maintainable.
- amenable to efficient computation;
  able to express features of the problem we can exploit for computational gain.
- learnable from data and past experiences.
- able to trade off accuracy and computation time

Notational Minefield

- Variable (probability and logic and programming languages)
- Model (probability and logic)
- Parameter (mathematics and statistics)
- Domain (science and logic and probability and mathematics)
- Grounding (logic and cognitive science)
- Object/class (object-oriented programming and ontologies)
- $=$ (probability and logic)
- First-order (logic and dynamical systems)

First-order predicate calculus



Skolemization and Herbrand's Theorem

Skolemization: give a name for an object said to exist

$\forall x \exists y p(x, y)$ becomes $p(x, f(x))$

Herbrand's theorem [1930]:

- If a logical theory has a model it has a model where the domain is made of ground terms, and each term denotes itself.
- If a logical theory T is unsatisfiable, there is a finite set of ground instances of formulas of T which is unsatisfiable.

Logic Programming

definite clauses: $\left\{ \begin{array}{l} part\_of(r123, cs\_building). \\ in(alan, r123). \\ in(X, Y) \leftarrow part\_of(Z, Y) \wedge in(X, Z) \end{array} \right.$

A logic program can be interpreted:

- Logically
- Procedurally: non-deterministic, pattern matching language where predicate symbols are procedures and function symbols give data structures
- As a database language

Unique Names Assumption & Negation as Failure

- Unique Names Assumption:
  different names denote different individuals
  different ground terms denote different individuals
- Negation as Failure:
  — g is false if it can't be proven true
  — Clark's completion:
  $$\begin{aligned}\forall X \forall Y\ in(X, Y) \ \leftrightarrow \ & (X = alan \wedge Y = r123) \vee \\ & (\exists Z\ part\_of(Z, Y) \wedge in(X, Z))\end{aligned}$$
  — stable model is a minimal model M such that an atom g is true in M if and only if there is a rule $g \leftarrow b$ where b is true in M.

Acyclic Logic Programs

In acyclic logic programs

- All recursions are well-founded
- You can't have:

  $a \leftarrow \neg a.$

  $b \leftarrow \neg c,\ c \leftarrow \neg b.$

  $d \leftarrow \neg e,\ e \leftarrow \neg f,\ f \leftarrow \neg d.$

Acyclic Logic Programs

In acyclic logic programs

- All recursions are well-founded
- You can't have:
  $a \leftarrow \neg a.$
  $b \leftarrow \neg c,\ c \leftarrow \neg b.$
  $d \leftarrow \neg e,\ e \leftarrow \neg f,\ f \leftarrow \neg d.$
- With acyclic logic programs:
  —One stable model
  —Clark's completion specifies what is true in that model
  —Can conclude $\neg g$ if g can't be proved
- Cyclic logic programs can have multiple stable models
  —exploited by answer-set programming

Choosing Objects and Relations

How to represent: "Pen #7 is red."

Choosing Objects and Relations

How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"

Choosing Objects and Relations

How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"
- $color(pen_7, red)$. It's easy to ask "What's red?"
  It's easy to ask "What is the color of pen_7?"
  Can't ask "What property of pen_7 has value red?"

Choosing Objects and Relations

How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"
- $color(pen_7, red)$. It's easy to ask "What's red?"
  It's easy to ask "What is the color of pen_7?"
  Can't ask "What property of pen_7 has value red?"
- $prop(pen_7, color, red)$. It's easy to ask all these questions.

Choosing Objects and Relations

How to represent: "Pen #7 is red."

- $red(pen_7)$. It's easy to ask "What's red?"
  Can't ask "what is the color of pen_7?"
- $color(pen_7, red)$. It's easy to ask "What's red?"
  It's easy to ask "What is the color of pen_7?"
  Can't ask "What property of pen_7 has value red?"
- $prop(pen_7, color, red)$. It's easy to ask all these questions.

$prop(Object, Property, Value)$ is the only relation needed:
object-property-value representation, Semantic network, entity relationship model

Universality of *prop*

To represent "a is a parcel"

- $prop(a, type, parcel)$, where *type* is a special property
- $prop(a, parcel, true)$, where *parcel* is a Boolean property

Reification

- To represent $scheduled(cs422, 2, 1030, cc208)$. "section 2 of course $cs422$ is scheduled at 10:30 in room $cc208$."
- Let $b123$ name the booking:

$$prop(b123, course, cs422).$$
$$prop(b123, section, 2).$$
$$prop(b123, time, 1030).$$
$$prop(b123, room, cc208).$$

- We have reified the booking.
- Reify means: to make into an object.

Triples and Semantics Networks

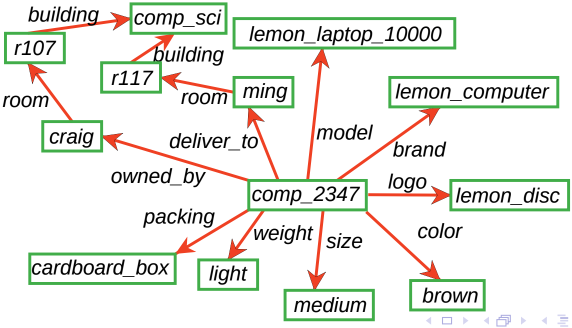
When you only have one relation, *prop*, it can be omitted without loss of information.

$prop(Obj, Att, Value)$ can be depicted as $\langle Obj, Att, Val \rangle$ or

Triples and Semantics Networks

When you only have one relation, *prop*, it can be omitted without loss of information.

$prop(Obj, Att, Value)$ can be depicted as $\langle Obj, Att, Val \rangle$ or



Frames

The properties and values for a single object can be grouped together into a frame.
We can write this as a list of *property* : *value* or *slot* : *filler*.

$$
\begin{aligned}
&[owned\_by : craig,\\
&deliver\_to : ming,\\
&model : lemon\_laptop\_10000,\\
&brand : lemon\_computer,\\
&logo : lemon\_disc,\\
&color : brown,\\
&\cdots]
\end{aligned}
$$

Classes

- A class is a set of individuals. E.g., house, building, officeBuilding
- Objects can be grouped into classes and subclasses
- Property values can be inherited
- Multiple inheritance is a problem if an object can be in multiple classes (no satisfactory solution)
- Need to distinguish class properties from properties of objects in the class

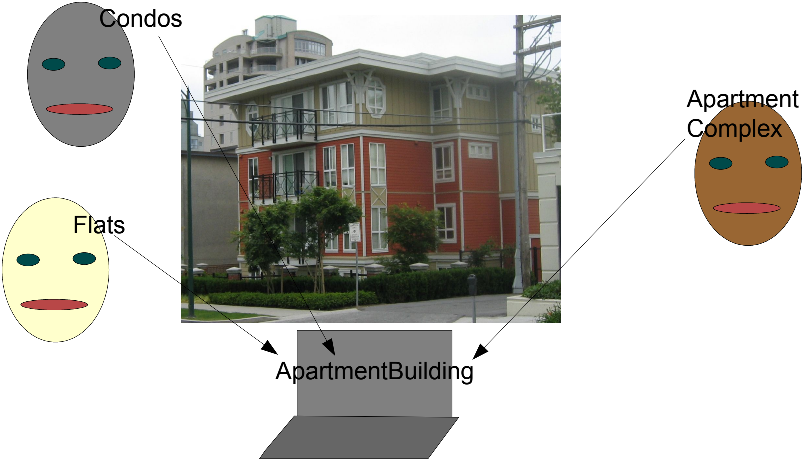
Knowledge Sharing

- If more than one person is building a knowledge base, they must be able to share the conceptualization.
- A conceptualization is a map from the problem domain into the representation. A conceptualization specifies:
  - What sorts of objects are being modeled
  - The vocabulary for specifying objects, relations and properties
  - The meaning or intention of the relations or properties
- An ontology is a specification of a conceptualization.

Ontologies

- Philosophy:
  — Study of existence
- AI:
  — "Specification of a Conceptualization"
  — Map: Concepts in head $\leftrightarrow$ symbols in computer
  — Allow some inference and consistency checking

Shared Conceptualization



Semantic Web Ontology Languages

- RDF — language for triples in XML. Everything is a resource (with URI)
- RDF Schema — define resources in terms of each other: type, subClassOf, subPropertyOf
- OWL — allows for equality statements, restricting domains and ranges of properties, transitivity, cardinality...
- OWL-Lite, OWL-DL, OWL-Full

Three views of KR

- KR as semantics We want to devise logics in which you can state whatever you want, and derive their logical conclusions.
  Examples: Logics of Bacchus and Halpern
- KR as common-sense reasoning We want something where you can throw in any knowledge and get out 'reasonable' answers.
  Examples: non-monotonic reasoning, maximum entropy.
- KR as modelling We want a symbolic modelling language for 'natural' modelling of domains.
  Examples: logic programming, Bayesian networks.

Logic and Uncertainty

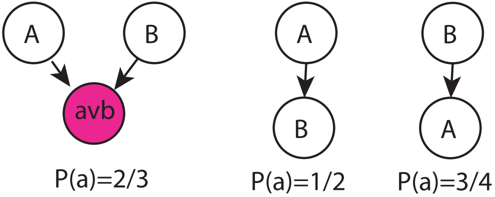
Choice:

- Rich logic including all of first-order predicate logic — use both probability and disjunction to represent uncertainty.
- Weaker logic where all uncertainty is handled by Bayesian decision theory. The underlying logic is weaker. You need to make assumptions explicit.

Logic and Uncertainty

tell $a \vee b$
ask $P(a)$

- Rich logics try to give an answer:
  $P(a) = 2/3$
  $P(a) \in [0.5, 0.75]$
- Weaker logics: you have not specified the model enough.



Probability over possible worlds or individuals

To mix probability and logic, two main approaches:

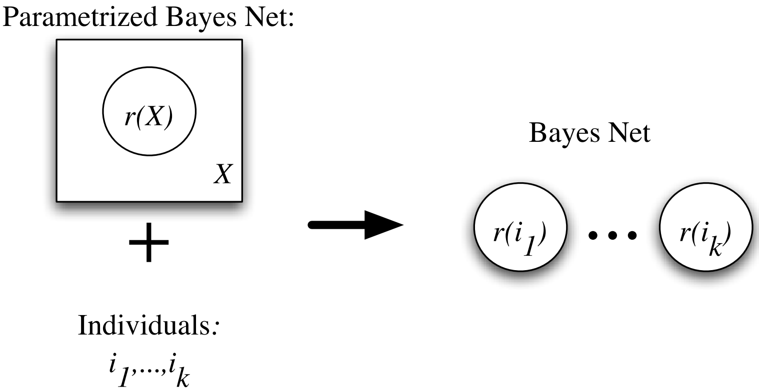
- a probability distribution over possible worlds
  — a possible world is like an interpretation but can have other properties.
  — measure over sets of possible worlds where the sets are described by finite logical formulae

Probability over possible worlds or individuals

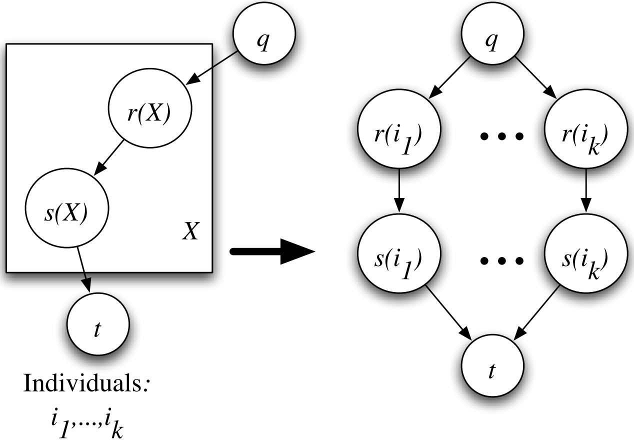
To mix probability and logic, two main approaches:

- a probability distribution over possible worlds
  — a possible world is like an interpretation but can have other properties.
  — measure over sets of possible worlds where the sets are described by finite logical formulae
- a probability distribution over individuals
  — proportion of individuals obeys the axioms of probability.

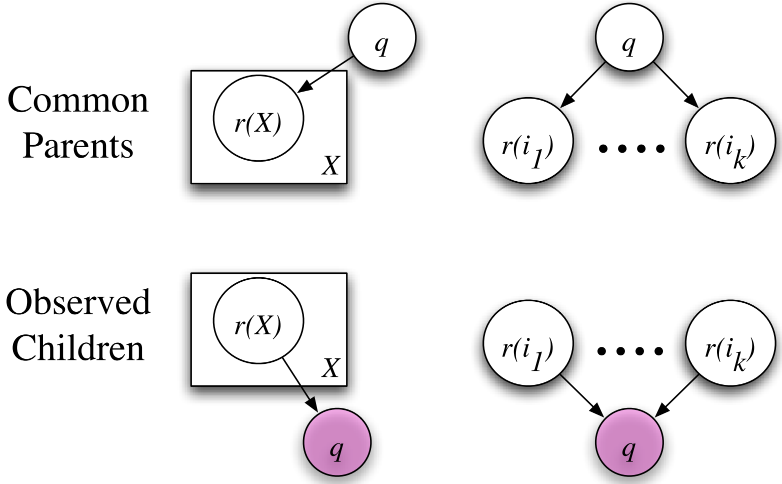
Parametrized Bayesian networks / Plates



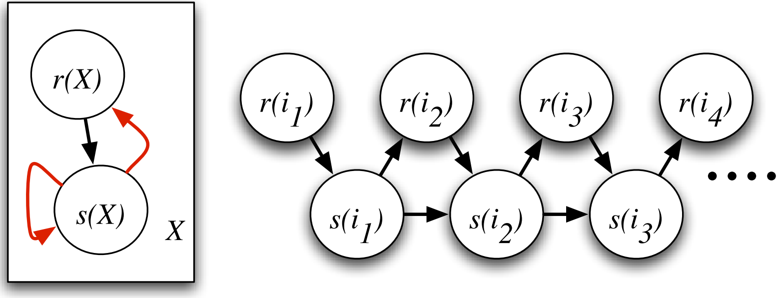
Parametrized Bayesian networks / Plates (2)



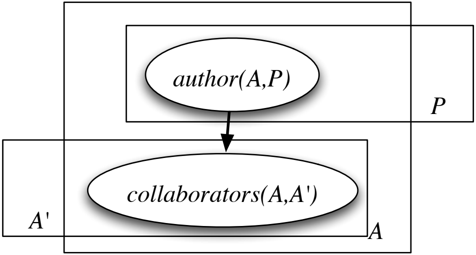
Creating Dependencies

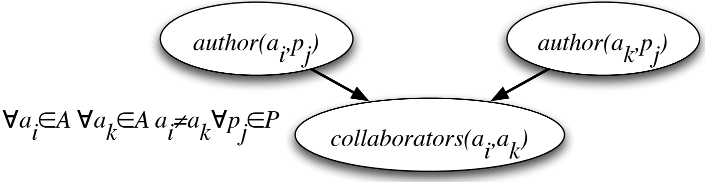


Creating Dependencies: Exploit Domain Structure



Creating Dependencies: Relational Structure



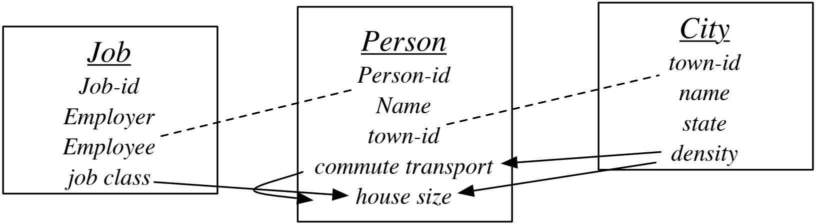


Probabilistic Relational Models

- In the object-property-value representation, there is a random variable:
  — for each object-property pair for each functional property
  - The range of the property is the domain of the variable.

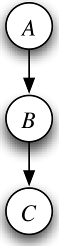
  — for each object-property-value there is a Boolean random variable for non-functional properties
- Plate for each class.

Probabilistic Relational Model Example



Procedural and Relational Probabilistic Languages

A Bayesian network can be represented as a deterministic system with (independent) stochastic inputs.



Independent Inputs	Deterministic System
a	
$bifa$ $bifna$	$b \leftrightarrow (a \wedge bifa)$ $\vee (\neg a \wedge bifna)$
$cifb$ $cifnb$	$c \leftrightarrow (b \wedge cifb)$ $\vee (\neg b \wedge cifnb)$

Procedural and Relational Probabilistic Languages

- A choice space is a set of random variables.
  Each random variable has a domain.
  [A set of the exclusive propositions corresponding to a random variable is an alternative.]
- There is a possible world for each assignment of a value to each random variable.
  [or from each selection of one proposition from each alternative.]
- The deterministic system specifies what is true in the possible world.
- You can also represent decision/game theory by having multiple agents making choices.

Meaningless Example

Alternatives: $\{c_1, c_2, c_3\}$, $\{b_1, b_2\}$

$$P_0(c_1) = 0.5 \quad P_0(c_2) = 0.3 \quad P_0(c_3) = 0.2$$
$$P_0(b_1) = 0.9 \quad P_0(b_2) = 0.1$$

$f \leftrightarrow (c_1 \wedge b_1) \vee (c_3 \wedge b_2)$, $d \leftrightarrow c_1 \vee (\neg c_2 \wedge b_1)$, $e \leftrightarrow f \vee \neg d$

Possible Worlds:

$$\begin{array}{llllllll}
w_1 & \models & c_1 & b_1 & f & d & e & P(w_1) = 0.45 \\
w_2 & \models & c_2 & b_1 & \neg f & \neg d & e & P(w_2) = 0.27 \\
w_3 & \models & c_3 & b_1 & \neg f & d & \neg e & P(w_3) = 0.18 \\
w_4 & \models & c_1 & b_2 & \neg f & d & \neg e & P(w_4) = 0.05 \\
w_5 & \models & c_2 & b_2 & \neg f & \neg d & e & P(w_5) = 0.03 \\
w_6 & \models & c_3 & b_2 & f & \neg d & e & P(w_6) = 0.02
\end{array}$$

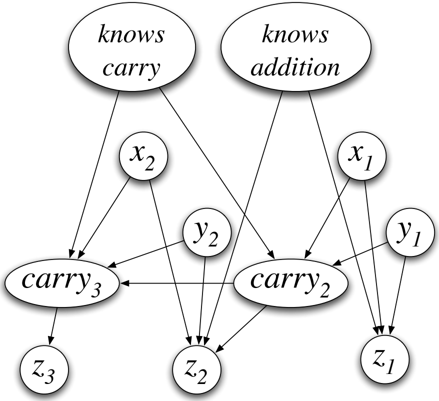
$$P(e) = 0.45 + 0.27 + 0.03 + 0.02 = 0.77$$

Some Representation Languages

- Independent Choice Logic (ICL): deterministic system is given by an acyclic logic program
- IBAL: deterministic system is given by a ML-like functional programming language
- A-Lisp: deterministic system is given in Lisp
- CES: deterministic system is given in a C-like language

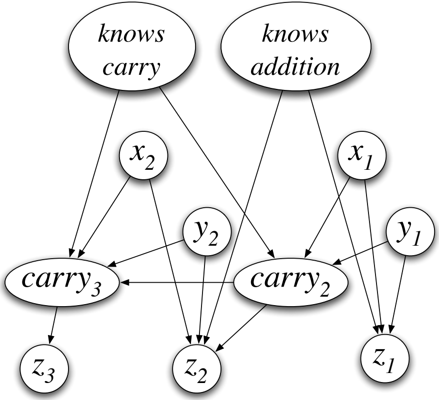
Diagnosing students errors

$$\begin{array}{cccc} & & x_2 & x_1 \\ + & & y_2 & y_1 \\ \hline & z_3 & z_2 & z_1 \end{array}$$



Diagnosing students errors

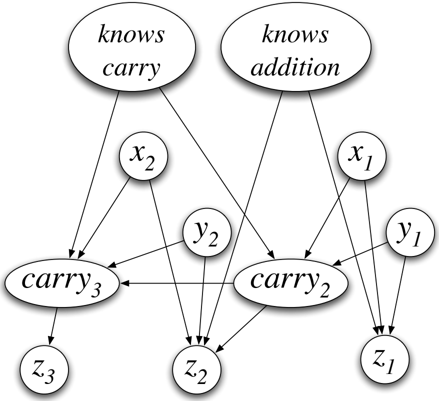
$$\begin{array}{cccc} & & x_2 & x_1 \\ + & & y_2 & y_1 \\ \hline & z_3 & z_2 & z_1 \end{array}$$



What if there were multiple digits

Diagnosing students errors

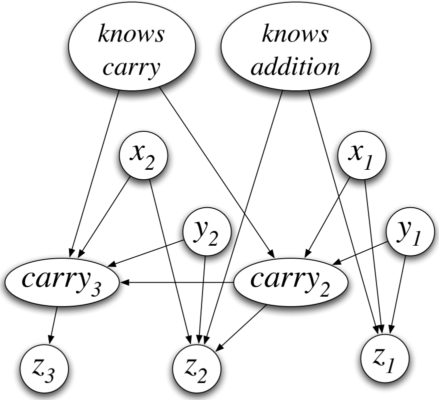
$$\begin{array}{rccc} & & x_2 & x_1 \\ + & & y_2 & y_1 \\ \hline & z_3 & z_2 & z_1 \end{array}$$



What if there were multiple digits, problems

Diagnosing students errors

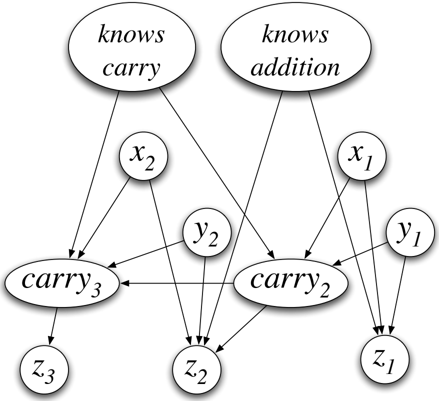
$$\begin{array}{cccc} & & x_2 & x_1 \\ + & & y_2 & y_1 \\ \hline & z_3 & z_2 & z_1 \end{array}$$



What if there were multiple digits, problems, students

Diagnosing students errors

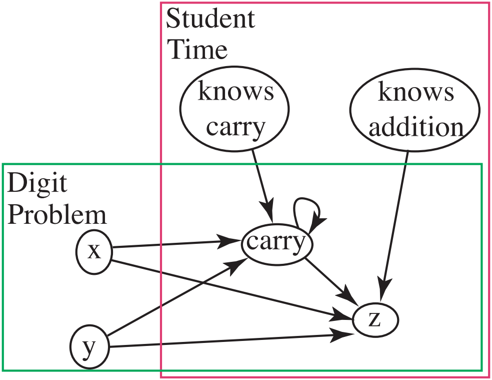
$$\begin{array}{cccc} & & x_2 & x_1 \\ + & & y_2 & y_1 \\ \hline & z_3 & z_2 & z_1 \end{array}$$



What if there were multiple digits, problems, students, times?

Example: Multi-digit addition

$$
\begin{array}{cccccc}
 & x_{j_x} & \cdots & x_2 & x_1 \\
+ & y_{j_z} & \cdots & y_2 & y_1 \\
\hline
 & z_{j_z} & \cdots & z_2 & z_1
\end{array}
$$



ICL rules for multi-digit addition

$$
\begin{aligned}
z(D,P,S,T) = V \leftarrow\ & \\
& x(D,P) = Vx \wedge \\
& y(D,P) = Vy \wedge \\
& carry(D,P,S,T) = Vc \wedge \\
& knowsAddition(S,T) \wedge \\
& \neg mistake(D,P,S,T) \wedge \\
& V \text{ is } (Vx + Vy + Vc) \text{ div } 10.
\end{aligned}
$$

$$
\begin{aligned}
z(D,P,S,T) = V \leftarrow\ & \\
& knowsAddition(S,T) \wedge \\
& mistake(D,P,S,T) \wedge \\
& selectDig(D,P,S,T) = V. \\
z(D,P,S,T) = V \leftarrow\ & \\
& \neg knowsAddition(S,T) \wedge \\
& selectDig(D,P,S,T) = V.
\end{aligned}
$$

Alternatives:

$\forall DPST \{noMistake(D,P,S,T), mistake(D,P,S,T)\}$

$\forall DPST \{selectDig(D,P,S,T) = V \mid V \in \{0..9\}\}$

First-order Probabilistic Inference

- Ground the representation to a ground Bayes net
- Carry out inference in the lifted representation (without grounding unless necessary)
- Compile to secondary structure, where first-order representations lead to structure sharing.

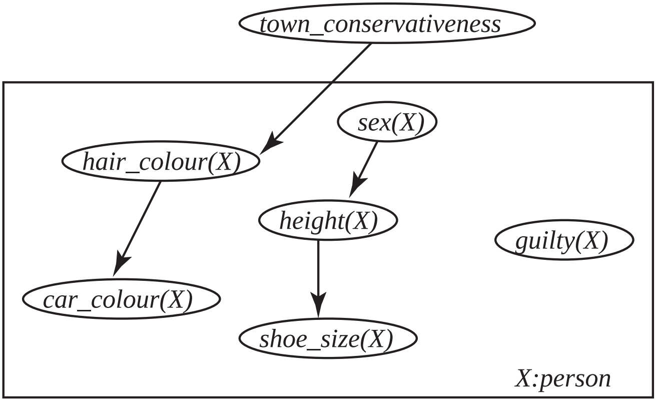
Lifted Inference Example

Suppose we observe:

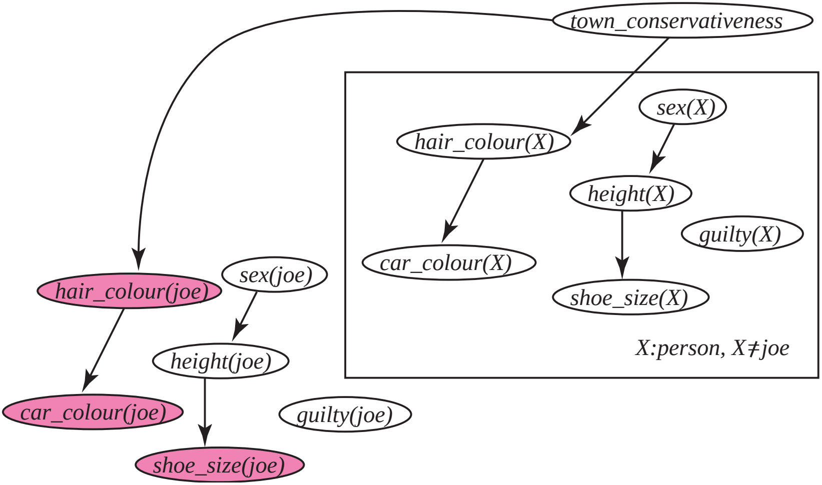
- Joe has purple hair, a purple car, and has big feet.
- A person with purple hair, a purple car, and who is very tall was seen committing a crime.

What is the probability that Joe is guilty?

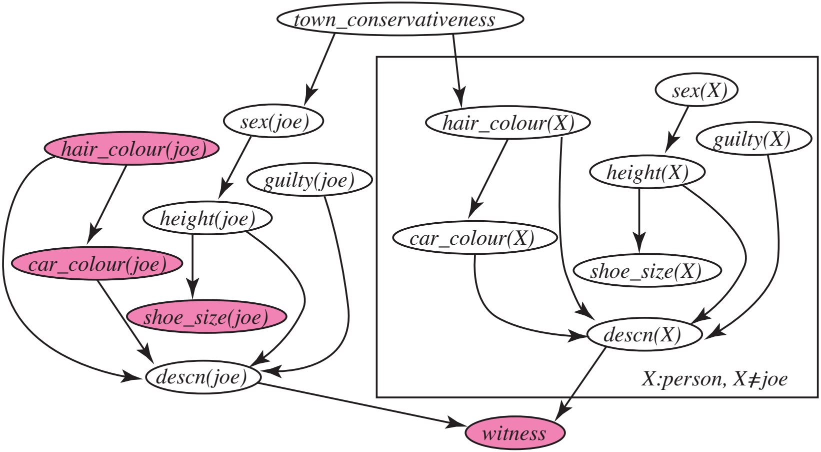
Background parametrized belief network



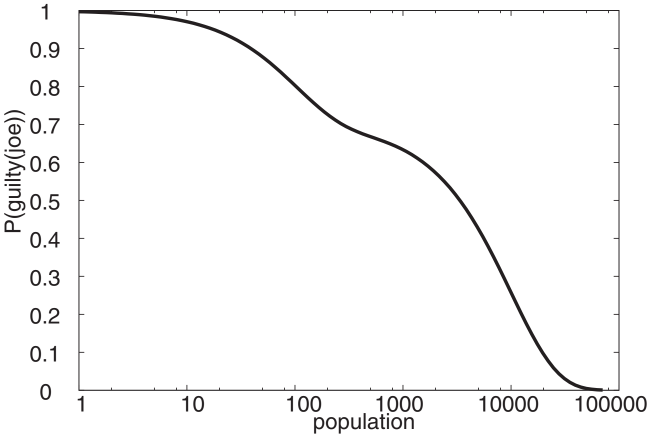
Observing information about Joe



Observing Joe and the crime



Guilty as a function of population



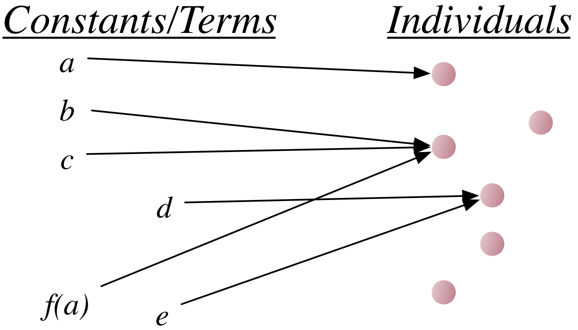
Learning

- Although there can be an unbounded number of variables, parameter sharing means that are only a finite number of distribution parameters to learn.
- You can also define a score on structure and search for the optimal structure.

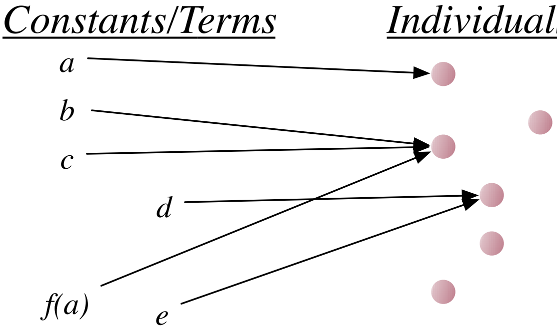
Identity Uncertainty

- Is this reference to the same paper as another reference?
- Is this the person who committed the crime?
- Is this patient the same as the patient who was here last week?
- Is this car the same car that was identified 3km ago?

Symbol Denotations



Symbol Denotations



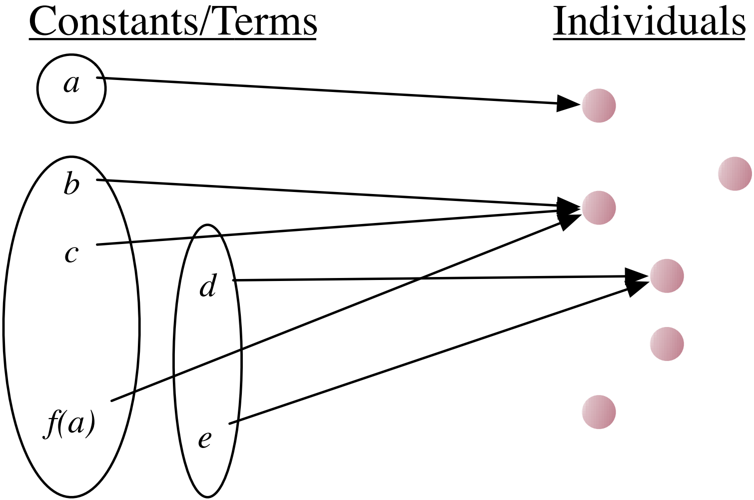
In logic, $x = y$ is true if x and y refer to the same individual.
$a \neq b$, $b = c$, $b = f(a)$, $d = e$, $d \neq b, \ldots$

Equality

Equality can be axiomatized with:

- $x = x$
- $x = y \Rightarrow y = x$
- $x = y \land y = z \Rightarrow x = z$
- $y = z \Rightarrow f(x_1, \ldots, y, \ldots, x_n) = f(x_1, \ldots, z, \ldots, x_n)$
- $y = z \land p(x_1, \ldots, y, \ldots, x_n) \Rightarrow p(x_1, \ldots, z, \ldots, x_n)$

Symbol Partitioning



Probability and Identity

- Have a probability distribution over partitions of the terms
- The number of partitions grows faster than any exponential (Bell number)
- The most common method is to use MCMC: one step is to move a term to a new or different partition.

Existence Uncertainty

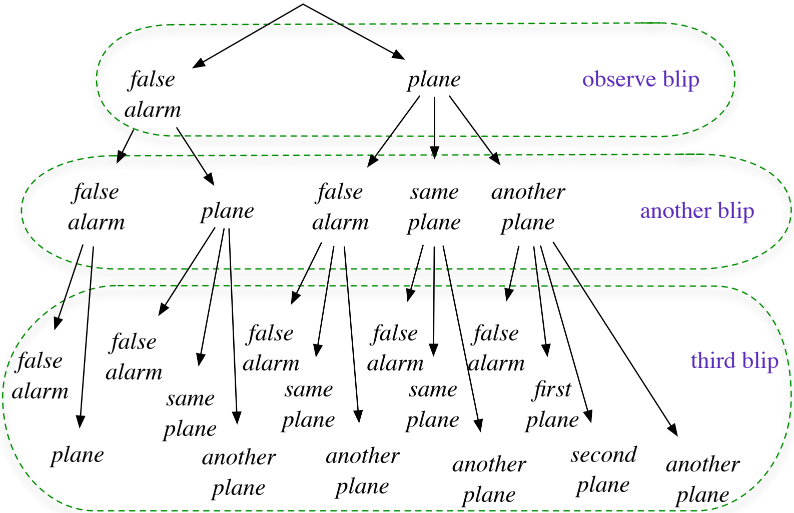
- What is the probability there is a plane in this area?
- What is the probability there is a large gold reserve in some region?
- What is the probability that there is a third bathroom given there are two bedrooms?
- What is the probability that there are three bathrooms given there are two bedrooms?

Existence Uncertainty

Two approaches:

- BLOG: you have a distribution over the number of objects, then for each number you can reason about the correspondence.
- NP-BLOG: keep asking: is there one more?
  e.g., if you observe a radar blip, there are three hypotheses:
  - the blip was produced by plane you already hypothesized
  - the blip was produced by another plane
  - the blip wasn't produced by a plane

Existence Example



Uncertainty and Ontologies

- We need to share conceptualizations.
  — People providing models and observations need to have common vocabulary.
- We need hierarchical type systems.
  — Probabilistic models may be at different levels of detail and abstraction than observations.
- . . . therefore we need ontologies.

Potential Confusions

- Object-oriented programming provides valuable tools for data/code sharing, abstraction and organization.
- Use the notion of class and object:

```
class person {
   int height;
}
```

  An instance of this is not a person!
- You cannot be uncertain about your own data structures!
- The notion of class and instance means something different in ontologies
  — this difference matters when you have uncertainty.

Ontologies and Uncertainty

- A community develops an ontology to allow semantic interoperability.
- People build probabilistic and/or preference models using this ontology.
- People describe the world using the ontology.

e.g., models of apartments, geohazards (e.g., where is it possible that there will be a toxic spill?),...

Conclusions

- There has been much progress over 20 years.
- We don't yet have the "Prolog" of first-order probabilistic reasoning.
- We need more experience with real applications to see what we really need.