Logic meets Probability & Learning

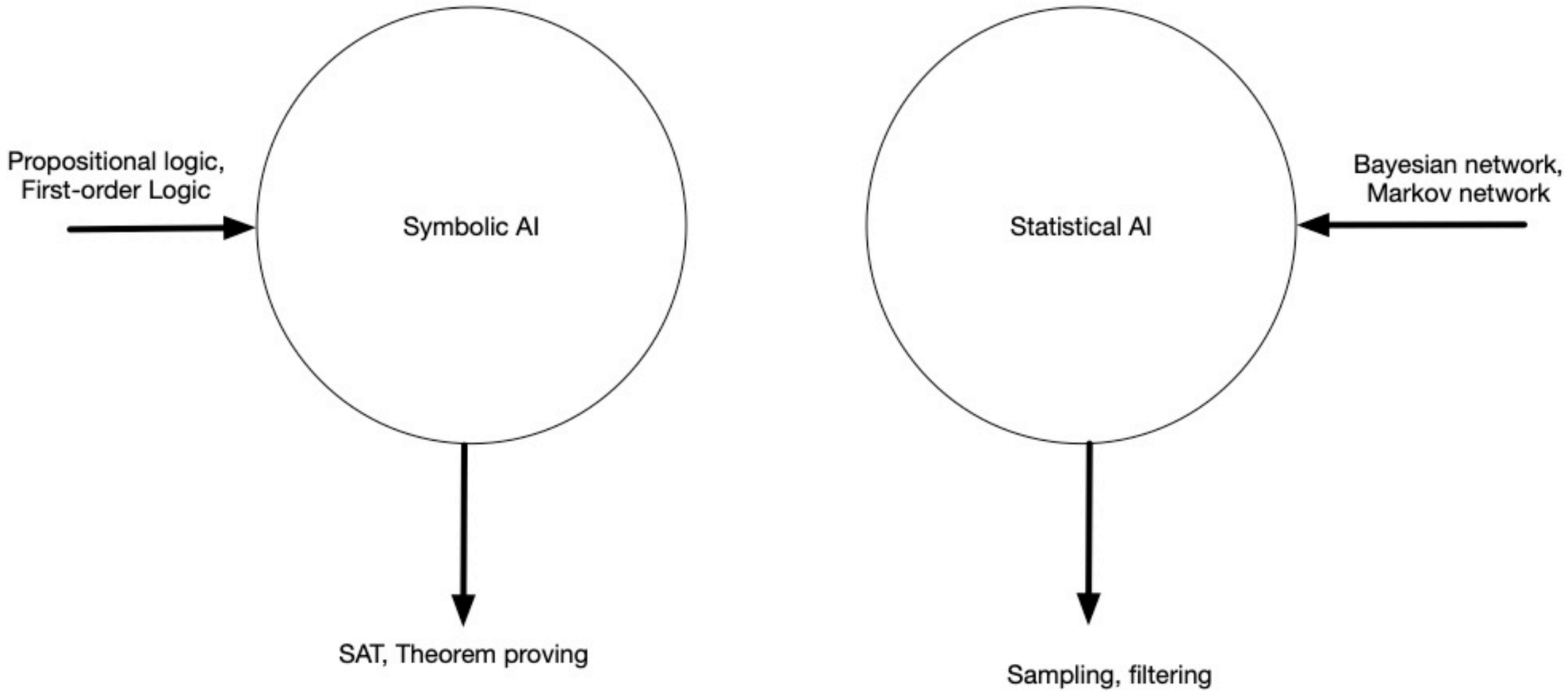
Vaishak Belle
University of Edinburgh & Alan Turing Institute

Scope & Structure

- Logic meets probability
- Probabilistic relational models
  - Effective reasoning
- Effective learning
- ~~More general logics~~
- Conclusions

Motivation

2 disciplines in AI



Complementary views of AI

Logic and probability complementary for representing and reasoning about the world:

Clearly, we often need features of both worlds!

- Symbolic vs numeric
- Qualitative vs quantitative
- Relations + objects vs random variables

The expressivity argument

Encoding of rules of chess:

- 1 page in FOL
- 10^5 pages in PL (~ graphical models)

Encoding of card games: Hand a card $Q\clubsuit$; what remains?

- $1\diamondsuit \vee \ldots \vee K\clubsuit$ in PL
- $\exists x\ (x \neq Q\clubsuit)$ in FOL

The implicit beliefs argument

It is impossible to store the implicit beliefs of an explicit representation in a logically complete manner (e.g., knowledge base completion in NLP)

- E.g., from p, we get $p \vee q, p \vee (q \wedge \neg q), \ldots$, etc.
- E.g., "I saw a black cat today", "... black animal", "... black living thing", "... animal whose color was not red", etc.
- E.g., does my ontology entail that humans and chimpanzees are related in some sense?

May not be a LLM/DB lookup!

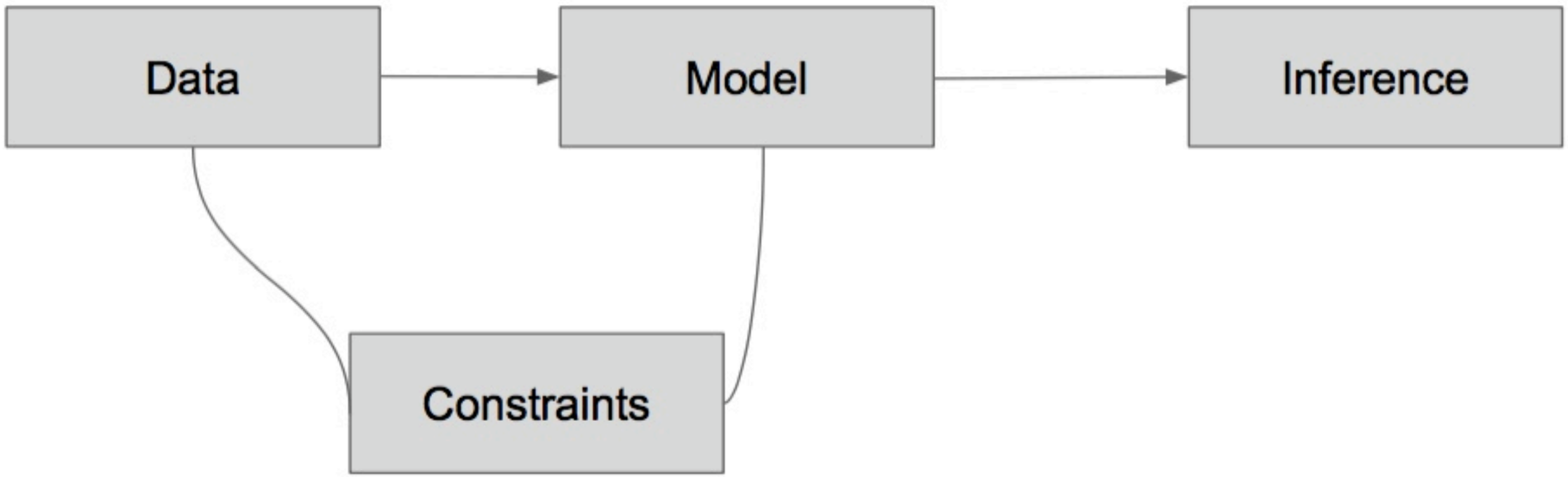
Top-down vs bottom-up

Gary Marcus:

To get computers to think like humans, we need a new A.I. paradigm, one that places "top down" and "bottom up" knowledge on equal footing. Bottom-up knowledge is the kind of raw information we get directly from our senses, like patterns of light falling on our retina. Top-down knowledge comprises cognitive models of the world and how it works.

Low-level, data-intensive, reactive computations needs to be tightly integrated with high-level, deliberative computations: Kahneman's System 1 vs System 2 processing in human cognition

Constraints



Usually data is emphasized, but real world is rich with constraints (e.g., gravity)

Commonsense

- Despite the success of deep learning, need to address *model re-use, transferability, causal understanding, relational abstraction, explainability, data efficiency*
- What might commonsense knowledge look like? Widely acknowledged to involve concepts such as *time, space, abstraction*

So, useful to combine, but how?

- Theorists such as Boole and de Finetti discussed the connections between logic and probability, and considered probability to be part of logic
- Heavy development in the last 70 years, starting with Gaifman and lots of exciting work in AI
- *Key idea:* for $\phi \in \mathcal{L}$, accord weights to $Models(\phi)$

Opportunities

- Advances in cross-over areas such as **statistical relational learning**, **neuro-symbolic systems**, and **high-level control** have illustrated that the *dichotomy is not very constructive*
- Many interesting historical ideas from Aristotle, Hume, Carnap, Pierce, and more recently Plotkin, Muggleton, and others.
- Deeper connection also between logic & probability: **probabilistic logics**, 0-1 laws etc.

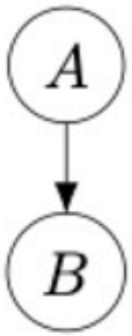
A spectrum (roughly speaking)

Relational BNs, MLNs, WMC	BLOG, Probabilistic logic programming	Languages for logic, probability, action
Symbolic (propositional) languages	*Some first-order features*	*Generalised measures, undecidable in general*

Probabilistic (relational) models

Bayesian networks

Like enumerating all possible propositional interpretations, and according them weights!



a	$\Pr(A = a)$
1	0.5
0	0.5

a	b	$\Pr(B = b \mid A = a)$
1	1	0.6
1	0	0.4
0	1	0.1
0	0	0.9

$$\nu_A = 0.5,$$

and

$$\nu_B = 0.6[\lambda_{B=1}] \cdot [\lambda_{A=1}] + 0.4[\neg\lambda_{B=1}] \cdot [\lambda_{A=1}] + 0.1[\lambda_{B=1}] \cdot [\neg\lambda_{A=1}] + 0.9[\neg\lambda_{B=1}] \cdot [\neg\lambda_{A=1}].$$

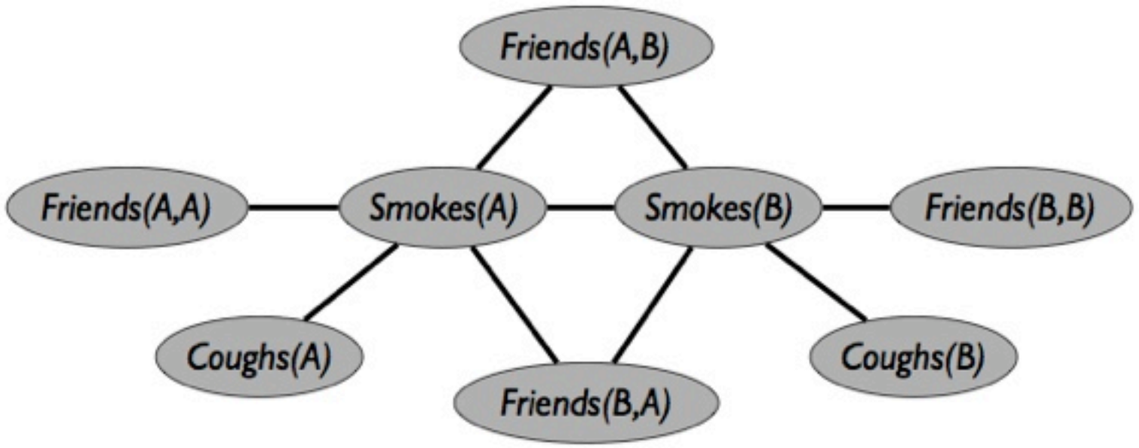
Lifted graphical models

Finite-domain FOL + weights for compact spec of undirected graphical models:

$$.8 \quad \forall x, y{:}\, D \;\; [Smokes(x) \wedge Friends(x, y) \supset Smokes(y)]$$

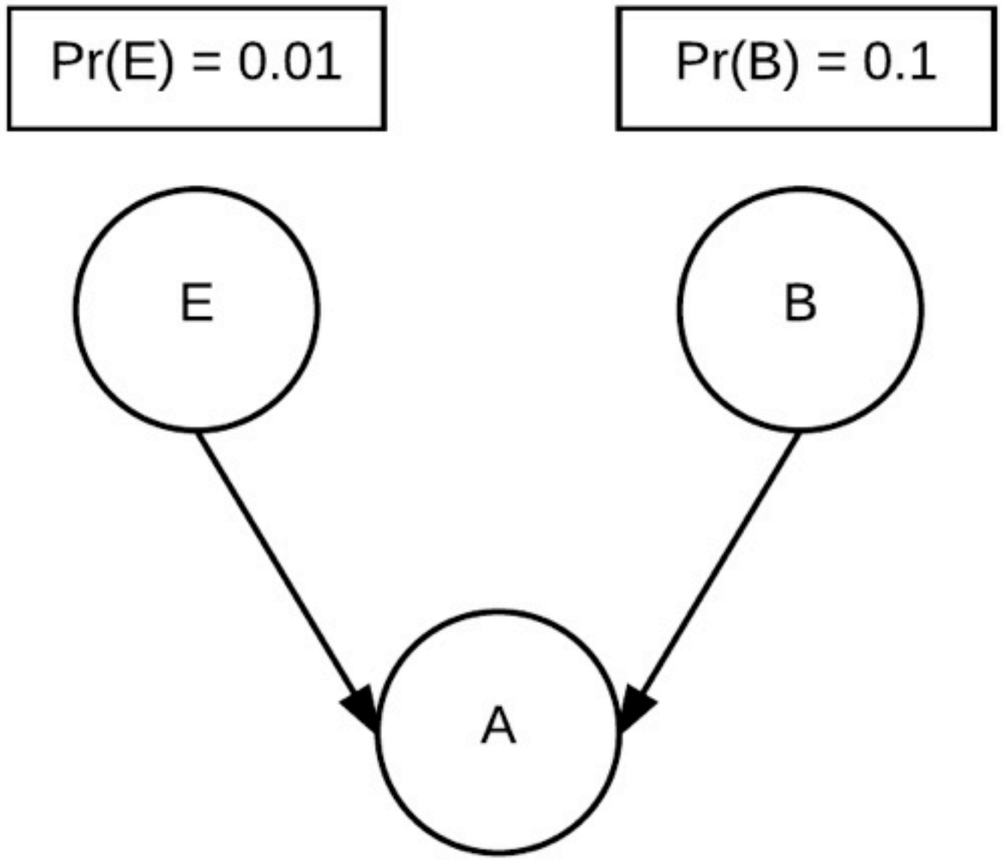
$$.72 \quad \forall x{:}\, D \;\; [Smokes(x) \supset Coughs(x)], D = \{A, B\}$$

- Atoms = random variables
- Weights on formulas = potentials on cliques



Probabilistic logical programs

```
0.1::burglary.
0.01::earthquake.

0.9::alarm :- burglary, earthquake.
0.8::alarm :- burglary, \+earthquake.
0.1::alarm :- \+burglary, earthquake.

evidence(alarm,true).
query(burglary).
query(earthquake).
```



Semantical setup

- Essentially, a finite set of probabilistic atoms, and finitely many models
- Compute probability as a function of these models

But many heterogenous representations, so problematic for general-purpose algorithms?

Effective reasoning

Weighted model counting (WMC)

- Handle hard and soft constraints natively
- General-purpose
- An "assembly language" for different representations (e.g., ProbLog)

- Based on SAT technology, so leverage advances such as *clause learning*
- Clean separation of weight function from logical theory
- State-of-the-art on many benchmark problems

SAT & #SAT

Given propositional CNF $(x \vee y) \wedge (y \vee \neg z)$:

- SAT: find a model $(x = 0, y = 1, z = 0)$
- #SAT: count models; here 5: (0,1,0), (0,1,1), (1,1,0), (1,1,1), (1,0,0)
- Equivalently: satisfying probability = $5/8$

Davis-Putnam-Logemann-Loveland

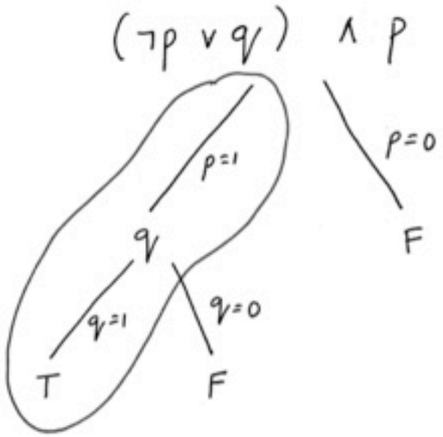
DPLL(F)

if F is true, return 1

if F is false, return 0

else choose a variable *x* to branch:

return (DPLL(F[x=1]) V DPLL(F[x=0]))



#DPLL(F) ... return .5*#DPLL(F[x=1]) + .5*#DPLL(F[x=0])

WMC

Each literal has a weight:

$$WMC(F, w) = \sum_{M \models F} \prod_{l \in M} w(l)$$

WMC(F,w) ... return **w(x=1)WMC(F[x=1])** + **w(x=0)WMC(F[x=0]))**

Example: Markov network

$$\begin{array}{ll} & \neg p \vee \neg q \\ .1 & p \\ .8 & p \vee r \\ .6 & q \Rightarrow r \end{array}$$

Equivalently, $\Delta = (\neg p \vee \neg q) \wedge (f_1 \Leftrightarrow p \vee r) \wedge (f_2 \Leftrightarrow \neg q \vee r)\,.$

Weights: $w(p) = 0.1, w(f_1) = .8, w(f_2) = .6$ and otherwise, the weight is 1.

- Model $M_1 = (p, \neg q, r, f_1, f_2)$ has weight .1 * .8 * .6
- Model $M_2 = (\neg p, \neg q, r, f_1, f_2)$ has weight .9 * .8 * .6

Correctness

Theorem Let *N* be a Markov network over Boolean random variables. Let *(F,w)* be the corresponding WMC encoding. Then for any *q*, *e*, the probability $\Pr_N(q \mid e)$ is given by

$$\frac{WMC(F \wedge q \wedge e, w)}{WMC(F \wedge e, w)}$$

WMC for BNs

Suppose (ϕ, w) encodes a BN. Then:

$$\Pr(q \mid e) = \frac{WMC(\phi \wedge q \wedge e, w)}{WMC(\phi \wedge e, w)}$$

Axioms of probability and BN conditional independence properties carry over to encoding: correctness preserving.

Via Davis-Putnam-Logemann-Loveland

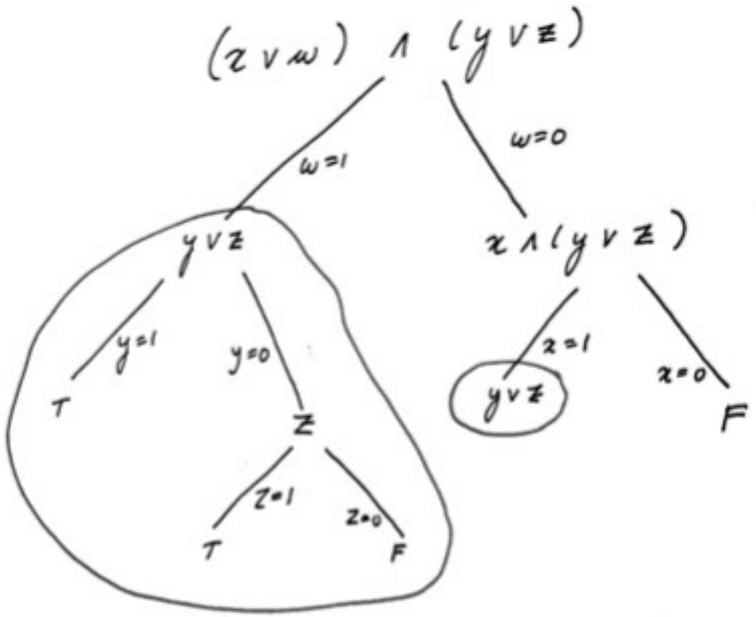
```
DPLL(F)

if F is empty, return 1

if F contains an empty clause, return 0

else choose a variable x to branch:

        return (DPLL(F[x=1]) V DPLL(F[x=0]))
```

```
#DPLL(F) … return .5*#DPLL(F[x=1]) + .5*#DPLL(F[x=0]))
```

$$WMC(F, w) =$$
$$\ldots \text{return } w(x=1) * WMC(F[x=1]) + w(x=0) * WMC(F[x=0]))$$

Exact methods

- #P-hard: exponential in worst case
- Naive implementation clearly looks at all branches
- Instead, cache "components"
  - Let w be treewidth of hypergraph
  - time bounded by $n^{O(1)}2^{O(w)}$

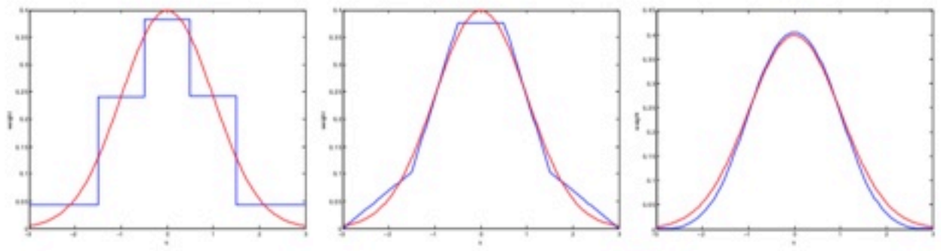


Main limitation

- **Propositional logic** $\Rightarrow$ **Discrete** (finite outcome) distributions only
- Planning, robotics, web mining, vision: **discrete and continuous** (hybrid) random variables
- Literature: **variational** methods with asymptotic guarantees OR analytical results for some families (e.g., Gaussians)

Can we engineer general-purpose methods that can deal with complex logical constraints in continuous domains?

Observation: piecewise representation of continuous distributions



(a) constants (b) degree 1 (c) degree 3

$$
\begin{array}{rl}
0.043 & 1.5 < u \\
(2+u)^3/6 & -2 < u \leq -1 \\
(4-6u^2-3u^3)/6 & -1 < u \leq 0 \\
(4-6u^2+3u^3)/6 & 0 < u \leq 1 \\
(2-u)^3/6 & 1 < u < 2
\end{array}
$$

Define weights on linear constraints!

SAT vs SMT

- Satisfiability modulo theories (SMT): reason about the satisfiability of arithmetic constraints over the rationals
- Becoming as mature as SAT

- E.g., Is $(x > y) \wedge (y \leq 5), x, y \in \mathbb{Z}$ satisfiable?
- Yes, $(y = 1, x = 2), (y = 5, x = 6)$, etc.

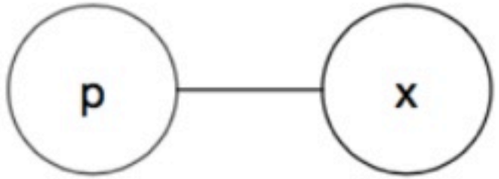
Weighted model integration (WMI)

- A new formulation: model counting on constraints + integration
- Weights may be numbers or polynomials (densities)

$$WMI(\Delta, w) = \sum_{M \models \Delta^{-}} VOL(M, w)$$

$$VOL(M, w) = \int_{\{l^{+} : l \in M\}} \prod_{l \in M} w(l)$$

Example: Markov network



p v (0 <= x<= 10)	w(p) = .1 w(~p) = 2x w(q) = 1 w(~q) = 0
p v q	

- $WMI = VOL(p \wedge q) + VOL(p \wedge \neg q) + VOL(\neg p \wedge q)$
- $VOL(p \wedge q) = \int_{0 \leq x \leq 10} .1 \, dx = 1.$
- $VOL(\neg p \wedge q) = \int_{0 \leq x \leq 10} 2x \, dx = 100.$

Correctness

Theorem WMI is a strict generalization of WMC.

Theorem Let *N* be a Markov network over Boolean and continuous random variables. Let *(F,w)* be the corresponding WMI encoding. Then conditional probabilities can be computed as before.

Remark When the dimensionality is fixed, computing $\int_D f(x_1, \ldots, x_n)$, where *f* is a polynomial of fixed degree is computable in PTIME.

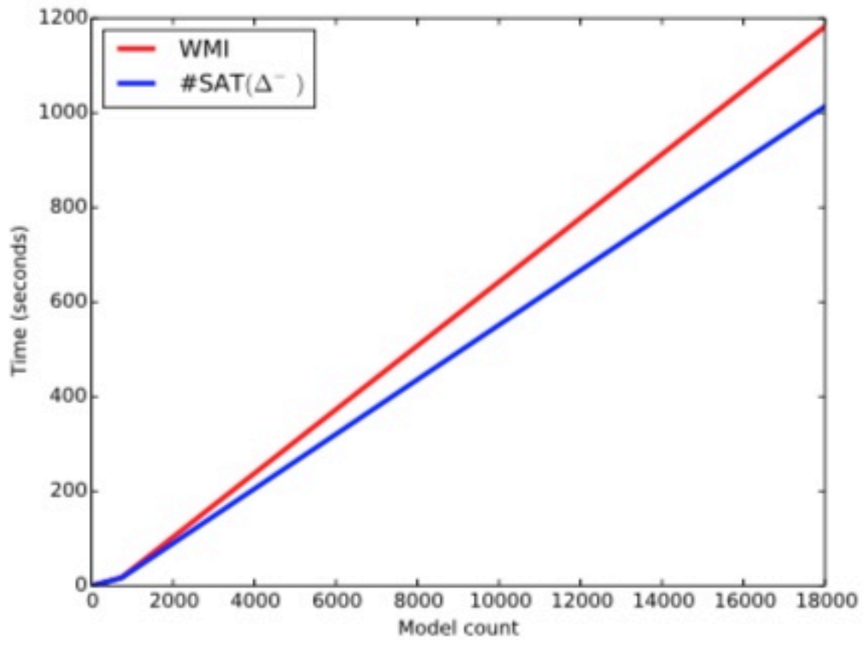
Implementation

Z3 with block-clause strategy + Latte polytope computation tool

Query	Probability
$\Pr(J < 1000 \mid M)$	0.669367
$\Pr(J < 1000 \mid M \wedge F)$	0.558360
$\Pr(J < 1000 \mid M \wedge F \wedge G)$	0.768575

where:

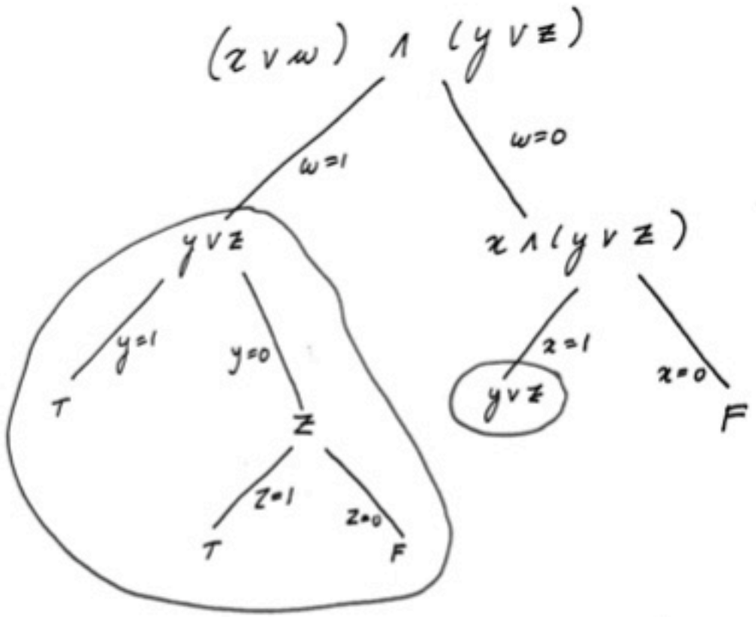
$$J = j_1 + j_2 + j_3;$$
$$M = morn \wedge j_1 > 300;$$
$$F = j_1 > 320 \wedge fri;$$
$$G = (s_3 >= 95)$$



Component caching + WMI?

Can we do the same for WMI?

Perhaps by abstraction ...



Alas, not immediately

$$\Delta = ((p \vee x \leq 10) \wedge (q \vee x \geq 11))$$

Which on abstraction, we get $(p \vee s) \wedge (q \vee t)$

So, two components: $\{(p \vee s), (q \vee t)\}$

But truth not independent, and so cannot model counting independently!

Solution: eager encoding, add $s \equiv \neg t$ to Δ

What about computing integrals?

Previously,

$$WMC(F, w) \ldots \text{ return } w(x = 1) * WMC(F[x = 1]) + \ldots$$

Now pass around information about densities and intervals:

$$\text{return } w(x = 1) * interval(x^+) * density(F[x = 1]) + \ldots$$

Theorem Suppose (Δ, w) is an eager encoding of a WMI theory, mentioning *n* propositions. Then there is an execution of WMI bounded by $n^{O(1)} 2^{O(w)}$.

On hybrid benchmarks

Dataset	#vars	#clauses	#literals
ALARM	596	865	4167
CHILD	279	419	1655
INSURANCE	787	1380	6591
WATER	3661	3072	18145

Dataset	CONST	POLY
ALARM	3.06	26.94
CHILD	0.02	1.64
INSURANCE	20.77	135.49
WATER	0.01	0.33

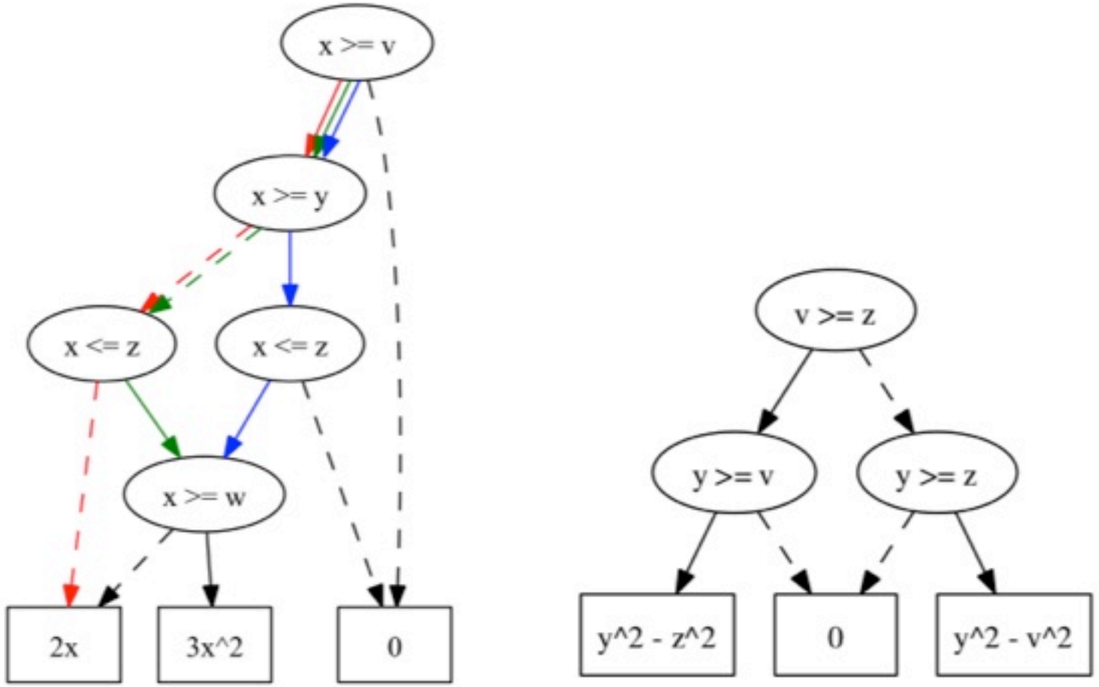
Missing	5	10	15	20	25	30	50	75	100	150	200
					ALARM						
BC	1.09	261.74	X	X	X	X	X	X	X	X	X
ALL	0.48	4.88	162.89	2842.69	X	X	X	X	X	X	X
CC	0.00	0.00	0.00	0.01	0.00	0.00	0.00	0.00	0.00	0.00	0.01
					CHILD						
BC	0.19	1.13	478.25	X	X	X	X	X	X	X	X
ALL	0.18	0.31	4.54	139.98	X	X	X	X	X	X	X
CC	0.00	0.00	0.00	0.01	0.00	0.01	0.00	0.00	0.00	0.00	0.01
					INSURANCE						
BC	1.47	344.41	X	X	X	X	X	X	X	X	X
ALL	0.67	6.71	238.33	X	X	X	X	X	X	X	X
CC	0.01	0.00	0.00	0.00	0.00	0.01	0.01	0.00	0.00	0.00	0.00
					WATER						
BC	6.08	1700.70	X	X	X	X	X	X	X	X	X
ALL	2.73	34.54	2282.18	X	X	X	X	X	X	X	X
CC	0.00	0.01	0.01	0.00	0.00	0.00	0.01	0.01	0.01	2.03	0.01

BC = (standard) block clause, ALL = allsat + WMI, CC = component caching with eager encoding

The search for more effective solvers is still on

$\Delta = x \geq v \wedge (x < y \vee x \leq z)$ with
$w(a) = 3x^2, w(\neg a) = 2x, a = (x \leq z \wedge x \geq v)$

By knowledge compilation: XADDs extend ADDs, support variable elimination (e.g., eliminate x), etc.



Problems with exact techniques

Ultimately, worst case exponential

What about approximation methods?

- Variational methods, sampling etc. very fast, but not robust, weak (asymptotic) guarantees

Can we do better?

(ϵ, δ)-algorithms by "cutting" up the space of solutions with hash functions

Belle et al. (UAI, 2015); Morettin et al. (AAAI, 2020); Abboud et al. (AIJ, 2022)

Logic & learning

Key questions

Aristotle: *learn the general from the particular*

- What knowledge is **provided** by the modeler versus what can be acquired by observations?
- What is the language for **representing** and reasoning with the background knowledge and observations?

- What kind of **semantics** governs the **updating** of a priori knowledge given new and possibly conflicting observations (e.g., unknown ground truth)?
- How does the system **generalize** from low-level observations to high-level structured knowledge?

Key dimensions

- **Logic vs. Machine Learning**, including the study of problems that can be solved using either logic-based techniques or via machine learning, ...

- **Machine Learning for Logic**, including the learning of logical artefacts, such as formulas, programs, ...

- **Logic for Machine Learning**, including the role of logics in delineating the boundary between tractable and intractable learning problems, …, and the use of logic as a declarative framework for expressing machine learning constructs

- Logic vs. Machine Learning: *Part 1*
- Machine Learning for Logic: *Part 2*
- Logic for Machine Learning: *Part 2* (briefly)

Learning for Logic

Acquiring new knowledge

Entailment Vs likelihood

Given: $\mathcal{H} \in \{\text{Logic programs}, \ldots\}$

1. **Find** $\mathcal{H}$ s.t. $\mathcal{B} \cup \mathcal{H} \models \mathcal{D}$; or
2. **Find** $\mathcal{H}$ s.t. $\delta(\mathcal{H}, \mathcal{D}) > \delta(\mathcal{H}', \mathcal{D})$ where $\delta(\mathcal{H}, \mathcal{D}) \propto \log \Pr(\mathcal{D} \mid \mathcal{H}) - |\mathcal{H}|$

Parameter learning

Maximum likelihood estimation: $\max_{w} L(w : D)$

Achieve by minimizing $E_D(\alpha) - E_w(\alpha)$ for all formulas α

Where, $E_w(\alpha) = WMI(\alpha \wedge \Delta, w)/WMI(\Delta, w)$

$$E_D(\alpha) = \text{size}(\{d \mid d \models \alpha \text{ and } d \in D\}) \ / \ \text{size}(D)$$

E.g., $D = \{x = 6, x = 7.1, y = 5, \ldots\}, \Delta = (x \leq 8) \vee \ldots,$ then $E_D(x \leq 8) = 2/|D|$

Limitations

Does counting, so only learns constant weights! What about polynomials?

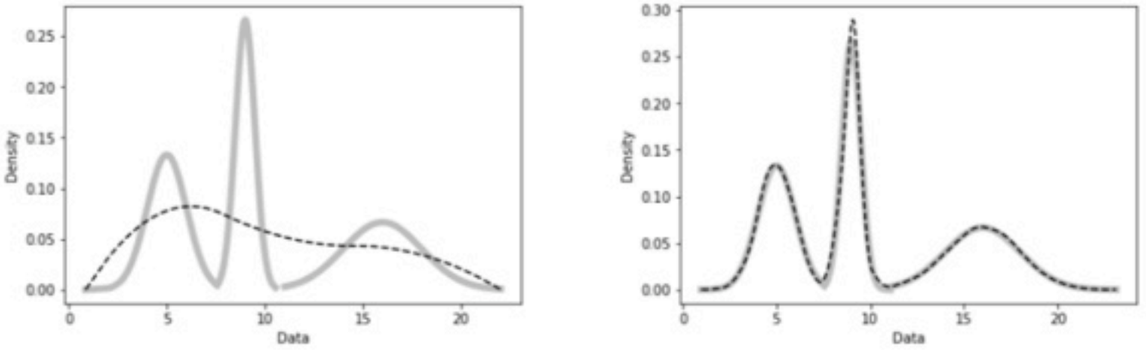
B-splines: suppose we had interval cut-points $t_0, \ldots, t_n$ for data *D*

$$B_{i,1}(x) \doteq \begin{matrix} 1 & \text{if} & t_i \leq x < t_{i+1} \\ 0 & & \text{otherwise} \end{matrix}$$

$$B_{i,k+1}(x) \doteq \frac{x - t_i}{t_{i+k} - t_i} B_{i,k}(x) + \frac{t_{i+k+1} - x}{t_{i+k+1} - t_{i+1}} B_{i+1,k}(x)$$

Polynomial parameter learning

- Choose discretization scheme (get $t_0, \ldots, t_n$)
- Upper bound: maximum polynomial degree
- Loop from degree 0: use maximum likelihood estimation for finding good splines
- Choose criterion (e.g., degree vs likelihood fit) to stop



(a) Interval=2,order=4 (b) Interval=12,order=4

What about programs?

Learning logic programs

```
parent(bart,stijn).
parent(bart,pieter).
parent(luc,soetkin).

female(alice).
female(an).
female(esther).

male(bart).
male(etienne).
male(leon).

grandmother(esther,soetkin).
grandmother(esther,stijn).
```

```
$ probfoil family.settings family.data
...
Candidates for iteration 4:
===================================
grandmother(A,B) :- parent(C,B), parent(A,C), parent(D,A) 0.503462603878
grandmother(A,B) :- parent(C,B), parent(A,C), parent(B,D) 0.457063711911
grandmother(A,B) :- parent(C,B), parent(A,C), male(C) 0.432528690146
grandmother(A,B) :- parent(C,B), parent(A,C), male(B) 0.432528690146
===================================
RULE LEARNED: grandmother(A,B) :- parent(C,B), parent(A,C), \+male(A) 1.0
```

Learning with probabilities

1. *A set of examples E, consisting of pairs (x_i, p_i) where x_i is a ground fact for the unknown target predicate t and p_i is the target probability.*
2. *A background theory B containing information about the examples in the form of a ProbLog program;*
3. *A loss function $loss(H, B, E)$, measuring the loss of a hypothesis (set of clauses) H w.r.t B and E;*
4. *A space of possible clauses L_h specified using a declarative bias;*

Find: A hypothesis $H \subseteq L_H$ such that $H = \arg\min_{\mathcal{H}} loss(H, B, E)$.

What are we after?

Learn the (discrete) distribution on entities (a)

Country	Happiness	Region	GDP (T)	...
UK	5.6	Europe	2.619	...
India	4.9	Asia	2.2	...
Germany	6.8	Europe	1.53	...
...	...	...	...	

Learn correlations between relational instances (b)

Learn the (continuous) spread of values (a)

Inducing logic programs + WMI

- Learn univariate distributions (parameter learning)
- Input: **weighted atoms** to program learner (structure learning)
- Output: Hybrid ProbLog program

```
-0.024719432823743857 + 0.000517156689054617 1 I  :: int_low(I).
int_low(I) :- intelligence(I), below(I,70).
int_mid(I) :- intelligence(I), ininterval(I,70,90).
```

A sample induced program

```
grade_high(A,B) :- sat_low(A,B), \+diff_hard(B), \+iq_1(A,X).
grade_high(A,B) :- takes(A,B), \+diff_hard(B).
```

```
grade_high(A,B) :- nrhours_4(B,Y), iq_1(A,X).
grade_high(A,B) :- nrhours_4(B,Y), course(B).
grade_high(A,B) :- sat_low(A,B)
```

```
-0.099723+0.0015473 * Y :: nrhours_4(B,Y):-
 course(B), nrhours(Y),between(64.45,78.73,Y)
0.056366359324390373 -0.0015982833846236427* X
+ 1.257392797387837e-05 * X ^ 2 ::iq_1(A,X):-
student(A),iq(X),between(52.6,103.4,X)
```

Not continuous but countable

- Humans generalize, and make statements about infinite sets of objects (e.g., object = day: sun rises and sets for every day)
- We might also encounter objects never seen before!
- Open-universe models capture this flavor

Example

Let Δ be the union of:

- $\forall(Smoker(x) \wedge Friends(x, y) \supset Smoker(y))$
- $Smoker(john)$
- $\forall y(Friends(john, y))$

Where quantifier is over an infinite set

Query: $Smoker(jane)$?

Types of open-world statements

- Unknown atoms: $\forall x\ (x \neq john \supset Smoker(x))$
- Unknown values: $\forall x(Canary(x) \supset \neg Color(x, black))$

Where quantifier is over an infinite set

- Varying sets of objects: $\forall x(Thing(x) \supset Color(x, black))$
- Closed-world assumption:
  $\forall x((x = john) \supset Smoker(x)) \wedge \forall x((x \neq john) \supset \neg Smoker(x))$

Where quantifier is over an infinite set

How can we solve this?

Define $GND(\Delta)$ as grounding clauses wrt entire domain

- $\forall x\colon D\ Smoker(x)$ with $D = \{A, B\}$ yields 2 atoms
- If $D = \mathbb{N}$, then countably infinite set of atoms

Define $OUGND(\Delta)$ as grounding wrt only constants mentioned + rank

- For $D = \mathbb{N}$, then only one atom (arbitrarily chosen constant)

Open-world (W)MC

Theorem For a large class of queries, it suffices to only consider $OUGND(\Delta)$, where KB is *universally quantified*

Why? All unseen constants behave identically, and exact instantiation irrelevant (exploit symmetry over unknowns)

Example revisited

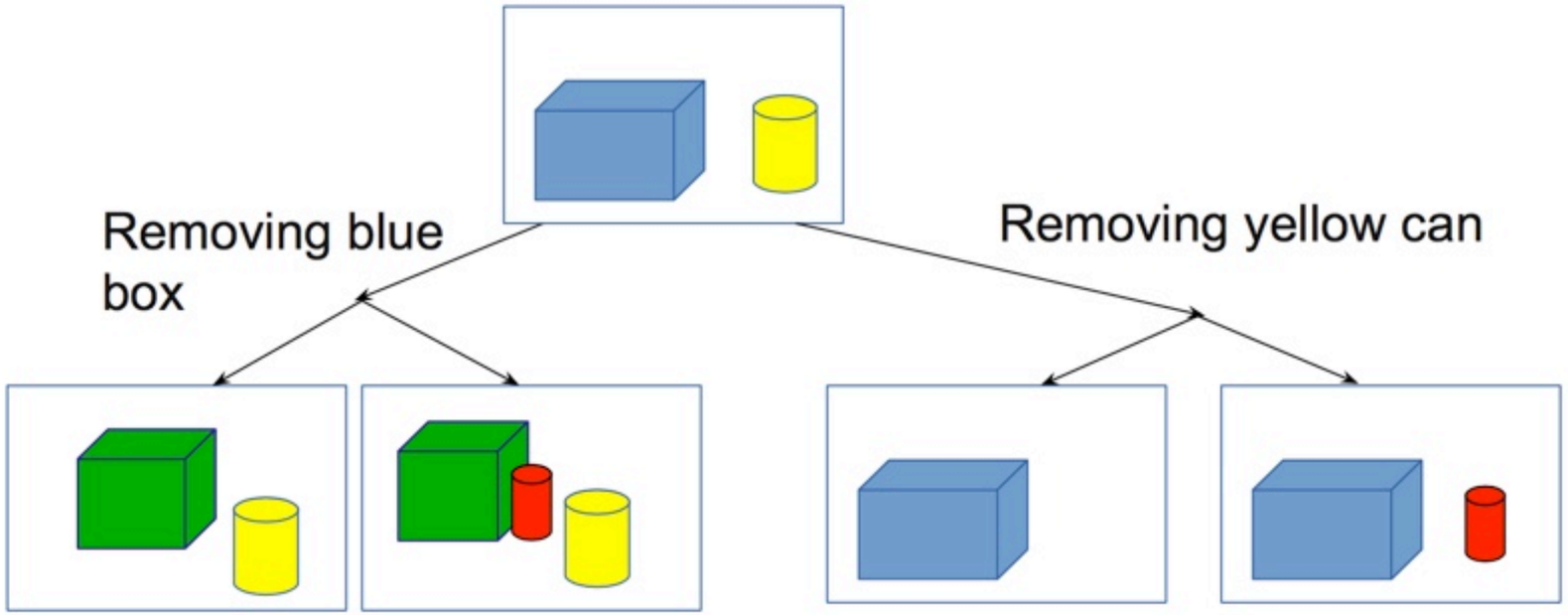
Let Δ be the union of:

- $\forall(Smoker(x) \wedge Friends(x, y) \supset Smoker(y))$
- $Smoker(john)$
- $\forall y(Friends(john, y))$

Query Q: $Smoker(jane)$

- $OUGND(\Delta \wedge \neg Q)$, then ground wrt {*john,jane,bob,mary*}
- Which means *Friends(john,jane)*, and *Smoker(jane)*; so, inconsistent with $\neg Q$
- UNSAT

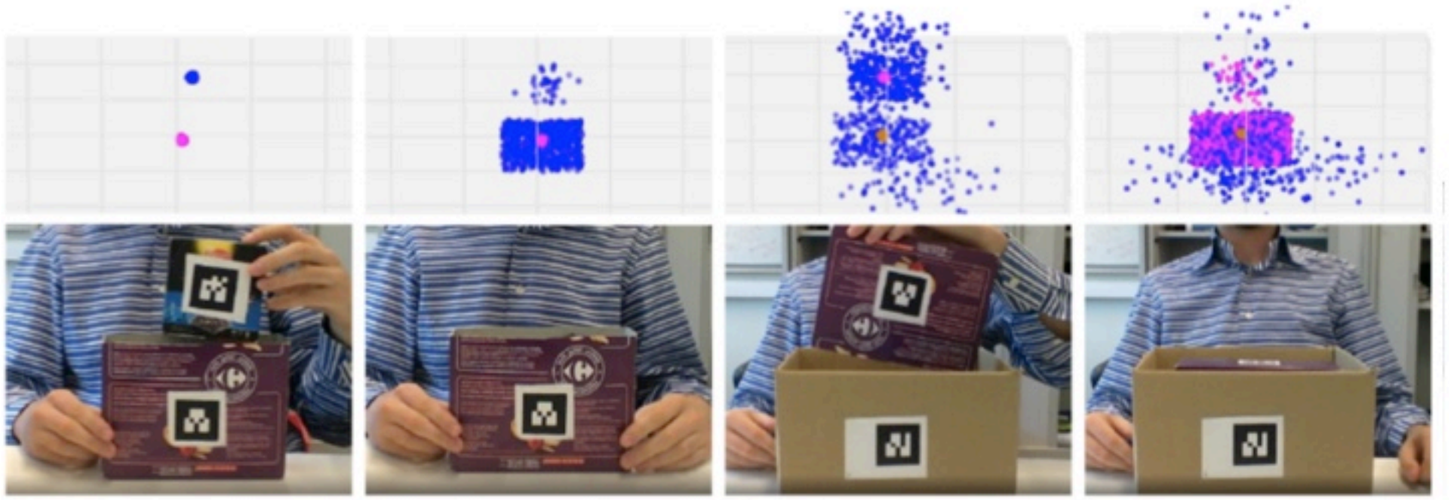
Planning in open-world domains



Nitti, Belle, De Laet, De Raedt (2018)

Where is the occluded object?

Particles and theories



Unknown color

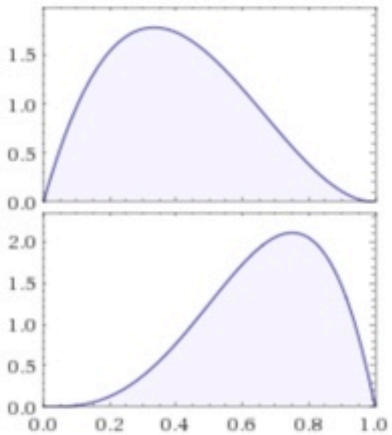
$$\overbrace{\texttt{color}(\texttt{X})}^{\text{Name random variable}} \sim \overbrace{\texttt{uniform}([\texttt{black}, \texttt{brown}])}^{\text{distribution}} \leftarrow \overbrace{\texttt{material}(\texttt{X}) \sim= \texttt{wood}.}^{\text{conditions (body)}}$$

Unknown size

$$\texttt{material}(\texttt{X}) \sim \texttt{finite}([0.3:\texttt{wood}, 0.7:\texttt{metal}]) \leftarrow \texttt{between}(1, \texttt{N}, \texttt{X}).$$

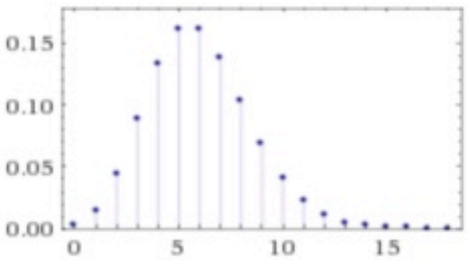
$$\texttt{size}(\texttt{X}) \sim \texttt{beta}(2,3) \leftarrow \texttt{material}(\texttt{X}) \sim= \texttt{metal}.$$

$$\texttt{size}(\texttt{X}) \sim \texttt{beta}(4,2) \leftarrow \texttt{material}(\texttt{X}) \sim= \texttt{wood}.$$



Unknown numbers

$\boxed{\texttt{n} \sim \texttt{poisson}(6).}$



(Infinite valued discrete distribution)

$$\texttt{material}(\texttt{X}) \sim \texttt{finite}([0.3:\texttt{wood}, 0.7:\texttt{metal}]) \leftarrow \texttt{n} \sim= \texttt{N}, \texttt{between}(1, \texttt{N}, \texttt{X}).$$

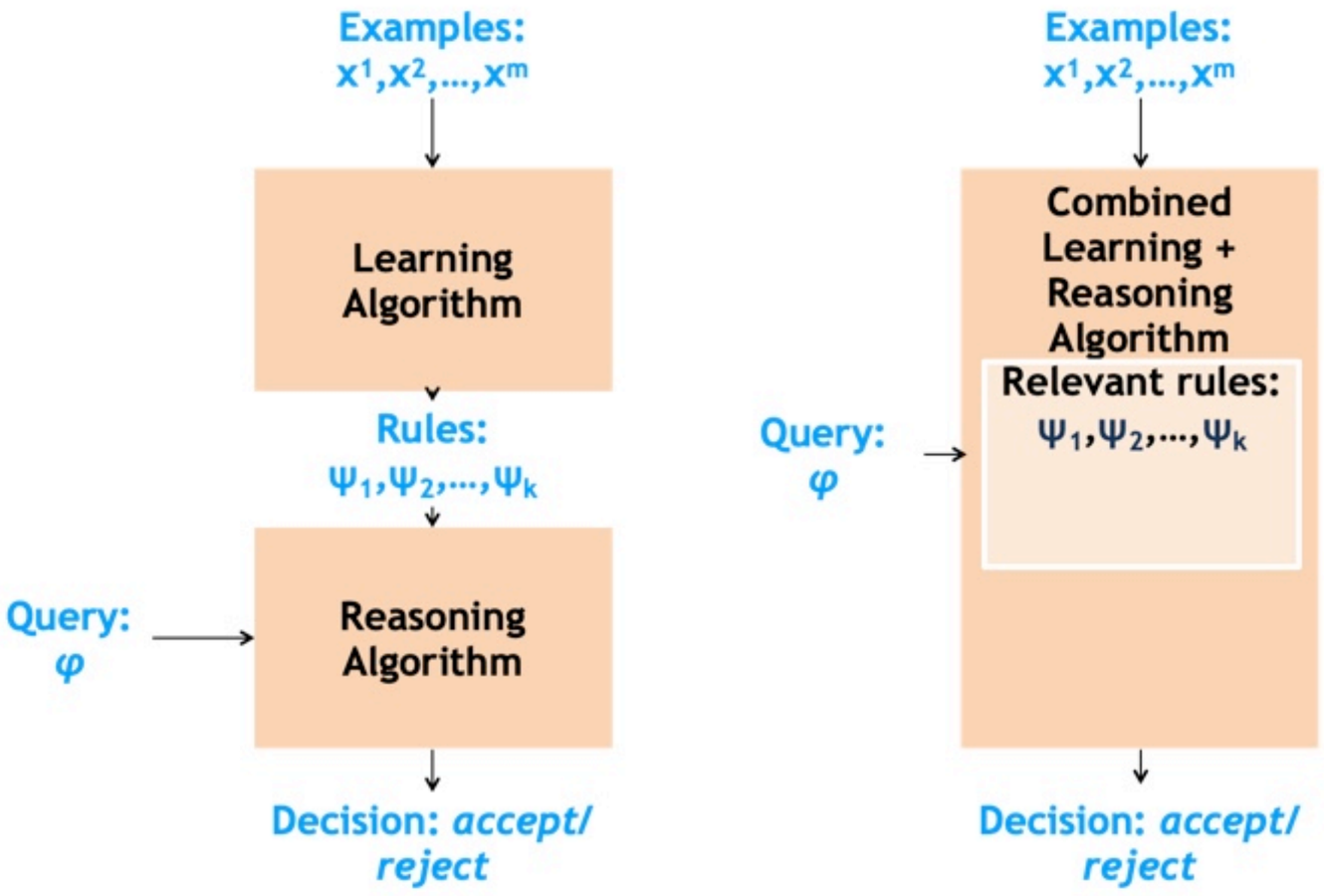
Back to the object

$$\begin{aligned}&\mathtt{inside}_{\mathtt{t+1}}(\mathtt{ID},\mathtt{B}) \sim \mathtt{finite}([0.8{:}\mathtt{true}, 0.2{:}\mathtt{false}]) \leftarrow \\ &\quad \mathtt{not}(\simeq(\mathtt{inside}_{\mathtt{t}}(\mathtt{ID},\mathtt{C})) = \mathtt{true}),\ \mathtt{on}_{\mathtt{t}}(\mathtt{ID},\mathtt{B}), \\ &\quad \mathtt{type}(\mathtt{B},\mathtt{box}),\ \simeq(\mathtt{yaw}_{\mathtt{t}}(\mathtt{B})) > 0,\ \mathtt{smaller}(\mathtt{ID},\mathtt{B}).\end{aligned}$$

Use transitivity:

$$inside(A, B) \wedge inside(B, C) \supset inside(A, C)$$

Avoid explicitly learning



Implicit learning in Belle & Juba/FOL (2019) + Mocanu et al./LRA (2021)

Rather than constructing a hypothesis s.t. $B \cup H \models \bigwedge_i e_i$,

Check instead whether $B \wedge e_i \models q, \forall i$.

Using the probably approximately correct (PAC) semantics, you can deal with **noise + partial examples**.

Logic for Learning

Structured Meta-Representations for Distributions and Learning Regimes

Tractability & Meta-theory

1. Tractable probabilistic representations
2. Abstraction, programming, etc.
3. Regularisation

Tractable models

logic-based data structures to encodes a joint distribution

Distribution as a polynomial

$X \sim$ Bernoulli(0.6)

$$F(X, \bar{X}) = pX + (1-p)\bar{X};$$
$$\Pr(X = 1) = F(1,0) = p;$$
$$\Pr(X = 0) = F(0,1) = 1 - p$$

With 2 variables

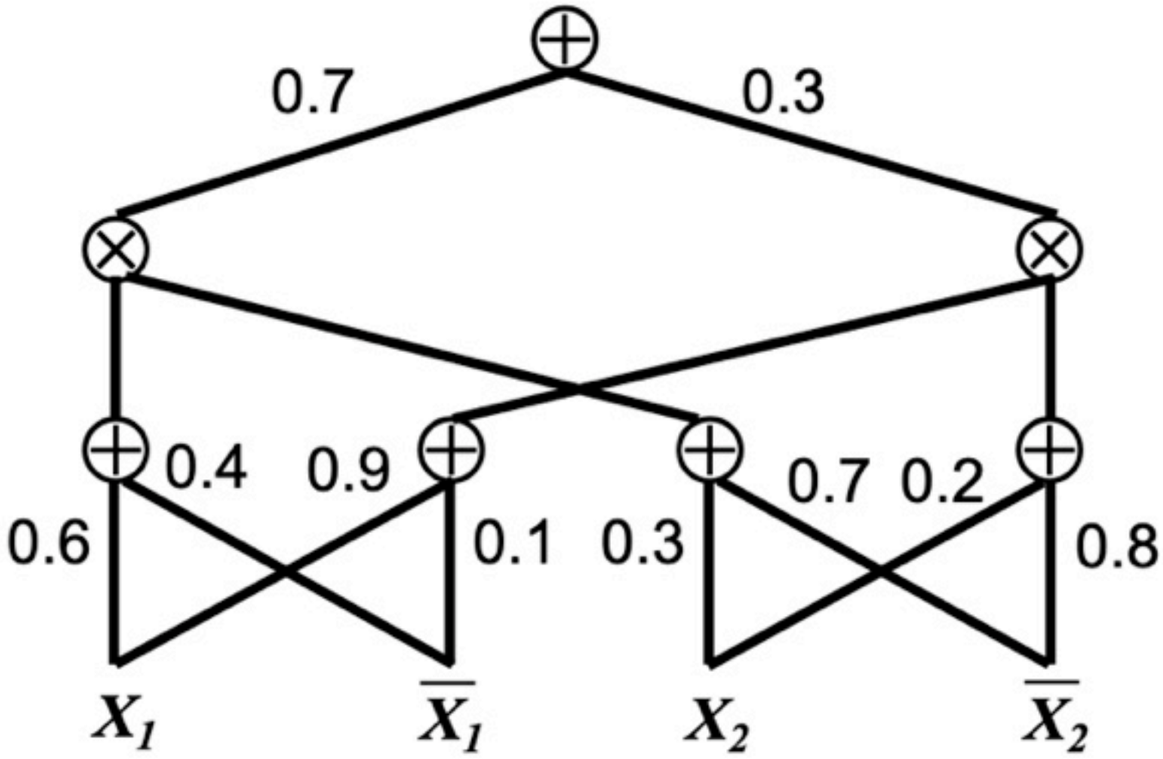
$$F(X_1, \bar{X}_1, X_2, \bar{X}_2) = \Pr(X_1, X_2)X_1X_2 + \Pr(\bar{X}_1, X_2)\bar{X}_1X_2 + \Pr(X_1, \bar{X}_2)X1\bar{X}_2 + \Pr(\bar{X}_1, \bar{X}_2)\bar{X}_1\bar{X}_2$$

Marginalising X_1: $F(1,1,X_2, \bar{X}_2) = \Pr(X_2)X_2 + \Pr(\bar{X}_2)\bar{X}_2$

Conditioning: $\Pr(X_2 \,|\, X_1 = 1) = F(1,0,X_2, \bar{X}_2)/F(1,0,1,1)$

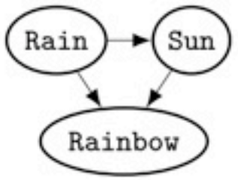
Sum product networks

- Leaf nodes: tractable univariate distributions, weighted sums of products (i.e., weighted mixtures)
- Certain computations linear in size of circuit (WMC & knowledge compilation)



standard deep models $P(y \mid x)$ vs SPNs: $P(x, y)$

BNs as PSDDs



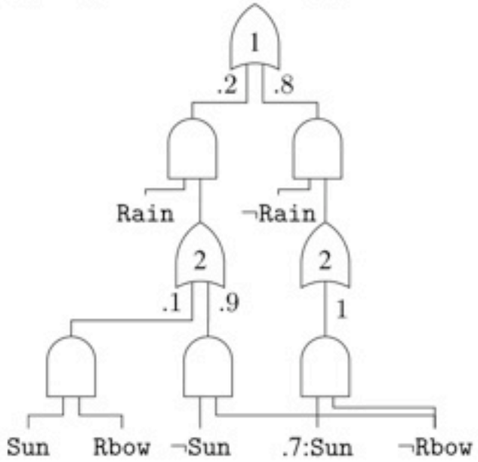
(a) Bayes net

$$\Pr(\texttt{Rain}) = 0.2,$$

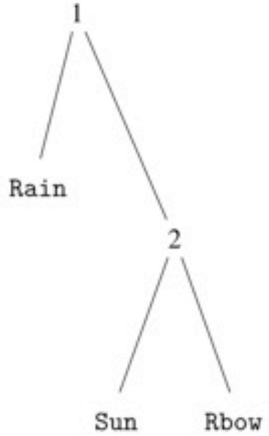
$$\Pr(\texttt{Sun} \mid \texttt{Rain}) = \begin{cases} 0.1 \text{ if } \texttt{Rain} \\ 0.7 \text{ if } \neg\texttt{Rain} \end{cases}$$

$$\Pr(\texttt{Rbow} \mid \texttt{R}, \texttt{S}) = \begin{cases} 1 \text{ if } \texttt{Rain} \wedge \texttt{Sun} \\ 0 \text{ otherwise} \end{cases}$$

(b) Conditional probabilities



(c) Equivalent PSDD circuit



(d) PSDD's vtree

Kisa et al (2014)

Applications

- BN inference (exponential in compilation, polytime inference in size)
- learn tractable models (e.g., fair models)
- explanations with tractable models

Knowledge compilation: Chavira & Darwiche, 2008; fairness: Varley & Belle, 2021; counterfactuals: Papantonis & Belle (2024)

Diverse Counterfactuals with Multilinear models

Diversity constraint

Given d,

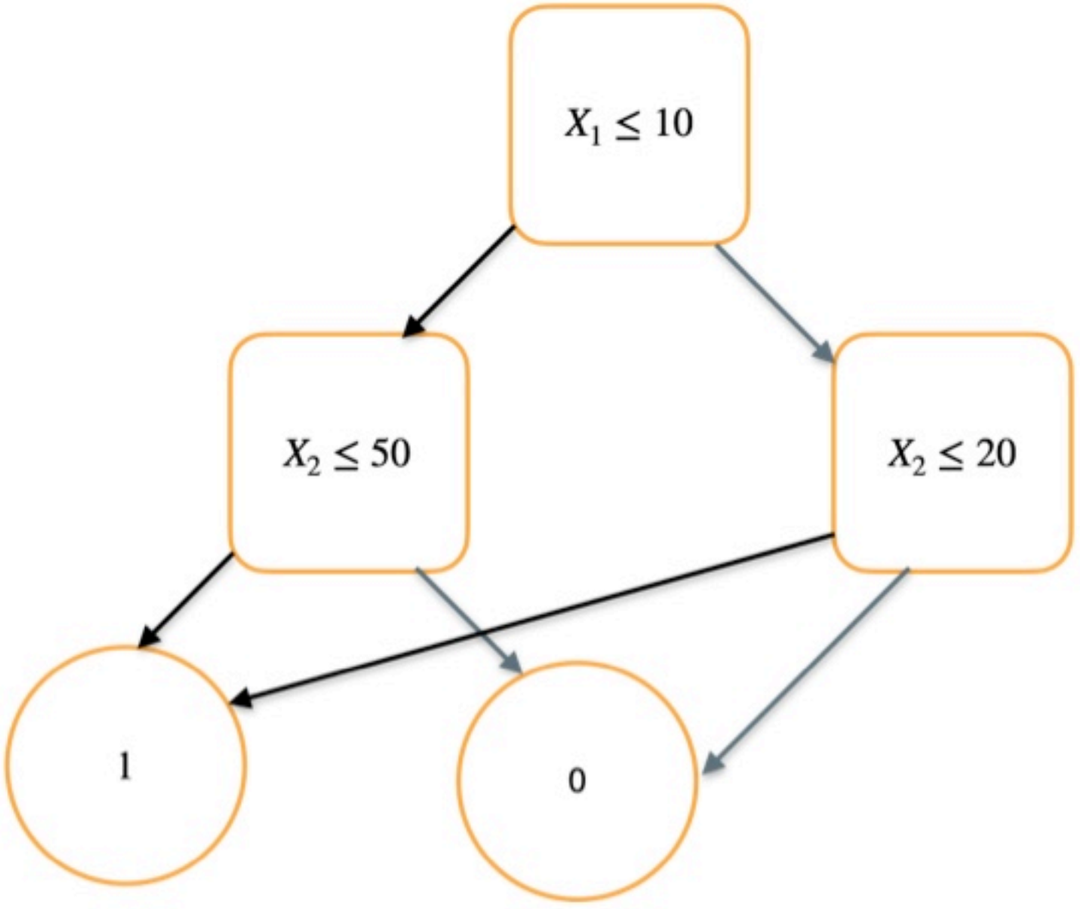
Find $d', f(d') \neq f(d)$.

Diversity constraint e.g., $age > 30, sex \neq male, \ldots$

From differential models to multilinear (SPNs, DTs, BNs, RFs)

$$|d - d'| \text{ small}$$

Decision tree



0-decision polynomial

$$[X_1 \leq 10] \times (1 - [X_2 \leq 50]) + (1 - [X_1 \leq 10]) \times (1 - [X_2 \leq 20])$$

If $P_0(X) = 0$ then every term is 0

Remark: $P_0(X) = 0 \Rightarrow P_1(1) = 1$

Counterfactual

Given a 0-class instance X, to get corresponding 1-class instance, find optimal point where:

- 0-DP should be 0
- $|d - d'|_w = w_1 \times ([X_1 \leq 10] - [X'_1 \leq 10]) + \ldots$
- constraints, such as $X'_1 \geq 5$, could be further added

Such an X' makes 0-DP = 0, so it makes 1-DP =1, so 1-class instance

	Sex	LSAT	Race	UGPA	Outcome
Factual	Male	34	White	3	Pass
Counterfactual	Male	$< \mathbf{19.25}$	White	3	Fail
Diverse counterfactual	Male	($\underline{> 25}$) 34	**Black**	$< \mathbf{1.95}$	Fail
Factual	Male	36.5	White	3.2	Pass
Counterfactual	Male	**20.75**	**Black**	$\leq \mathbf{1.95}$	Fail
Diverse counterfactual	Male	**19.25**	(<u>White</u>) White	**2.15**	Fail
Factual	Female	43	White	2.8	Pass
Counterfactual	Female	**26.75**	White	2.8	Fail
Diverse counterfactual	Female	$\leq \mathbf{19.25}$	(<u>White</u>) White	$\leq \mathbf{2.15}$	Fail
Factual	Male	35	White	2.7	Pass
Counterfactual	Male	35	**Black**	**1.95**	Fail
Diverse counterfactual	Male	$\leq \mathbf{19.25}$	(<u>White</u>) White	2.7	Fail
Factual	Male	33	White	3	Pass
Counterfactual	Male	33	**Black**	**1.85**	Fail
Diverse counterfactual	Male	$\leq \mathbf{19.25}$	(<u>White</u>) White	3	Fail

Programming models: beyond WMC

$$AMC(\phi, w) = \oplus_{M \models \phi} \otimes_{l \in M} w(l)$$

Generalizes WMC wrt commutative semirings: allowing soft constraints, gradients etc. + composition?

(semiring) program = logic + semiring + solver

Kimmig et al (2012); Belle & De Raedt (2020)

```
(set-logic FOL)
(set-algebra [NAT,+,0])
(set-type COLOR={r,b,g})
(set-type NODE={1,2,3})
(declare-predicate node (NODE))
(declare-predicate edge (NODE,NODE))
(declare-predicate color (NODE,COLOR))
N = (node(1) and node(2) and node(3))
E = (edge(1,2) and edge(2,3) and edge(3,1))
DATA = (N and E)
CONS = /* coloring constraints (omitted) */
(declare-weight TRUE 1)
(count (DATA and CONS))
```

Counting graph coloring

```
(set-logic LRA)
(set-algebra [REAL,inf,0])
(declare-function input (INT,INT) REAL)
...  /* declare left, right, app */
(declare-function err () REAL)
DATA = /* entries in input matrix (omitted) */
F = app(x,y) == sum{e} left(x,e)*right(e,y)
G = err == norm(sum{x,y} input(x,y) - app(x,y))
(declare-weight TRUE err)
(count (DATA and F and G))
```

Matrix factorization

```
(set-logic PL)
(set-algebra [NAT,max,*,0,1])
(declare-predicate p ())
(declare-predicate q ())
F = (p or q)
(declare-weight (p 1))
(declare-weight ((neg p) 2))
(declare-weight (q 3))
(declare-weight ((neg q) 4))
(count F)
```

MPE

```
(set-logic QF_LIA;PL)
(set-algebra [NAT,max,0])
(set-type INT={1,...,10})
(declare-function x1 () INT)
(declare-function x2 () INT)
(declare-predicate p1 ())
(declare-predicate p2 ())
F = ((p1 or p2) => 3*x1 <= 4)
G = (p2 => (2*x2 <= 5))
H = ((F and G) and p2)
(declare-weight TRUE x1*x2)
(count H)
```

Logical-integer programming

	FT	FN	IT	IN	C	S	R
APROBLOG	✓	×	×	×	✓	✓	✓•
CHURCH	✓	×	✓	×	✓•	×	×
ESSENCE	×	✓	×	×	✓•	×	×
CSP	×	✓•	×	✓•	✓•	×	×
SEMIRING CSP	×	✓•	×	×	✓•	✓	×
AMPL	×	✓	×	✓	✓•	×	×
ASP	✓	✓•	×	×	✓	×	✓
(O)SMT	×	✓	×	✓	✓	×	×
#SMT	✓	×	✓	×	✓	×	×
SP	✓	✓	✓	✓	✓	✓	✓

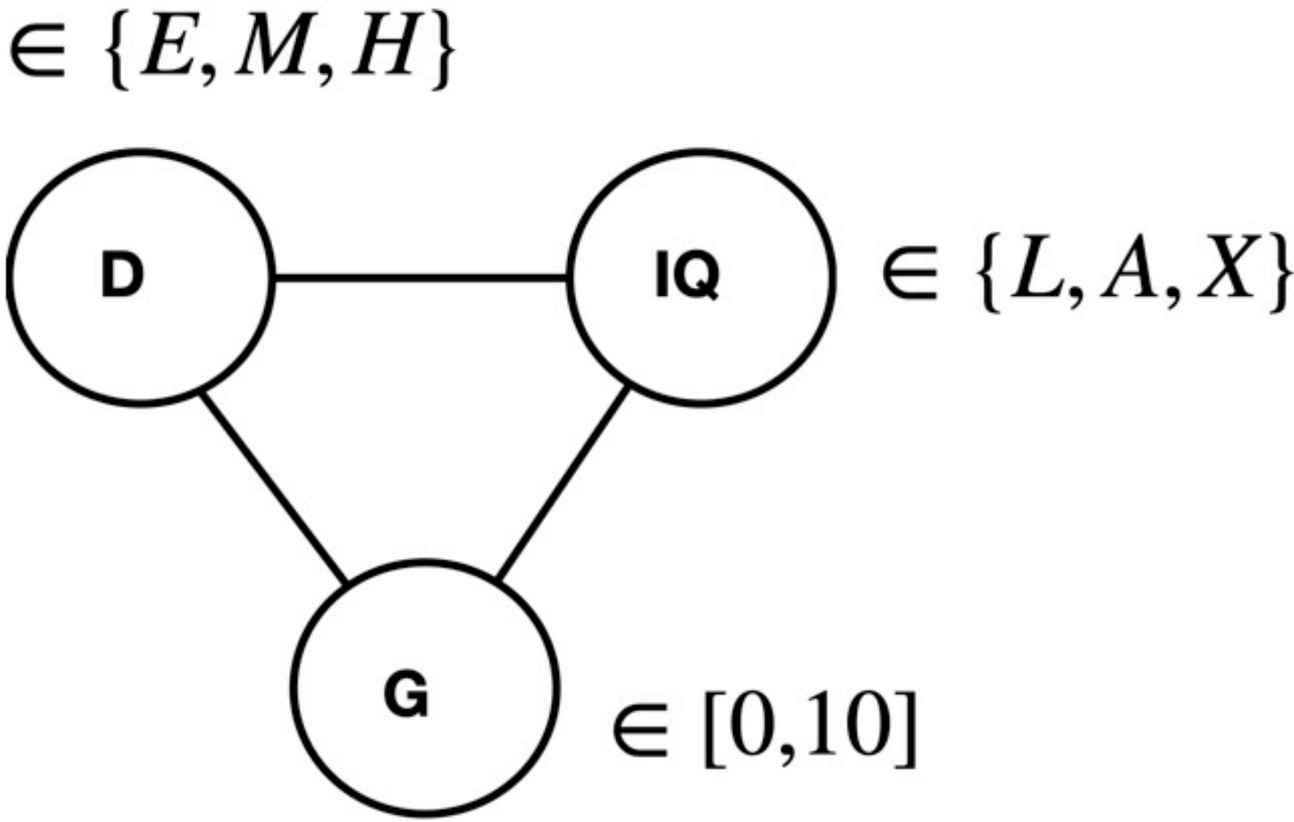
A comparison, where F = finite, T = factorized, N = non-factorized, I = infinite, C = logical connectives and quantifiers, R = non-standard, S = semiring apparatus, and • denotes that a feature is available in a restricted sense.

Abstraction

- How will relate **abstract, high-level concepts to** low-level sensory data and feature learning?
- What is a **faithful alignment** between two representations?

- Structure knowledge, hierarchically, for explanations, commonsense reasoning, etc.
- Abstracting problem domain to a smaller search space (even with tractable representations)

- Collapse domain (e.g., **D**'s domain is *easy* or *not-easy)*
- Make **G** a discrete variable
- Obtain a **qualitative** abstraction (e.g., alert admin if low-IQ students fail easy course)
- Bi-simulation between models via WMC



Outlook

- **Alternative computational schemes:** logic-based solvers for constrained probabilistic reasoning and learning
- **Representation and expressiveness:** logic can help us model the relational, continuous & countably infinite

- **Contexualising:** verify, add domain knowledge, enable modularity and re-usability, combine declared and learned knowledge, meta-theory

On other hand, learning can address the fundamental question of how to arrive at symbolic knowledge (from data)