Constraint-based Probabilistic Modeling for Statistical Abduction

Taisuke Sato, Masakazu Ishihata
(Tokyo Institute of Technology)
Katsumi Inoue
(National Institute of Informatics)

Logic and probability

Logic: rules

- Default: nothing connected
- Unless connected by axioms

Probability :uncertainty

- Default: everything connected
- Unless independence assumed

Real world

- Constraint-based
  probabilistic modeling
- Independent random variables
  + constraints

Outline

- We combine
  - probability distribution $P(X_1=x_1,..,X_n=x_n)$ and
  - logical constraints KB over $\{X_1,..,X_n\}$

as a CBPM (constraint-based probabilistic model) P_c(“$X_1=x_1$”,..,“$X_n=x_n$” | KB) where {“$X_1=x_1$”,..,“$X_n=x_n$” } are independent boolean variables w.r.t. P_c s.t. “$X_1=x_1$” is true if-and-only-if $X_1=x_1$

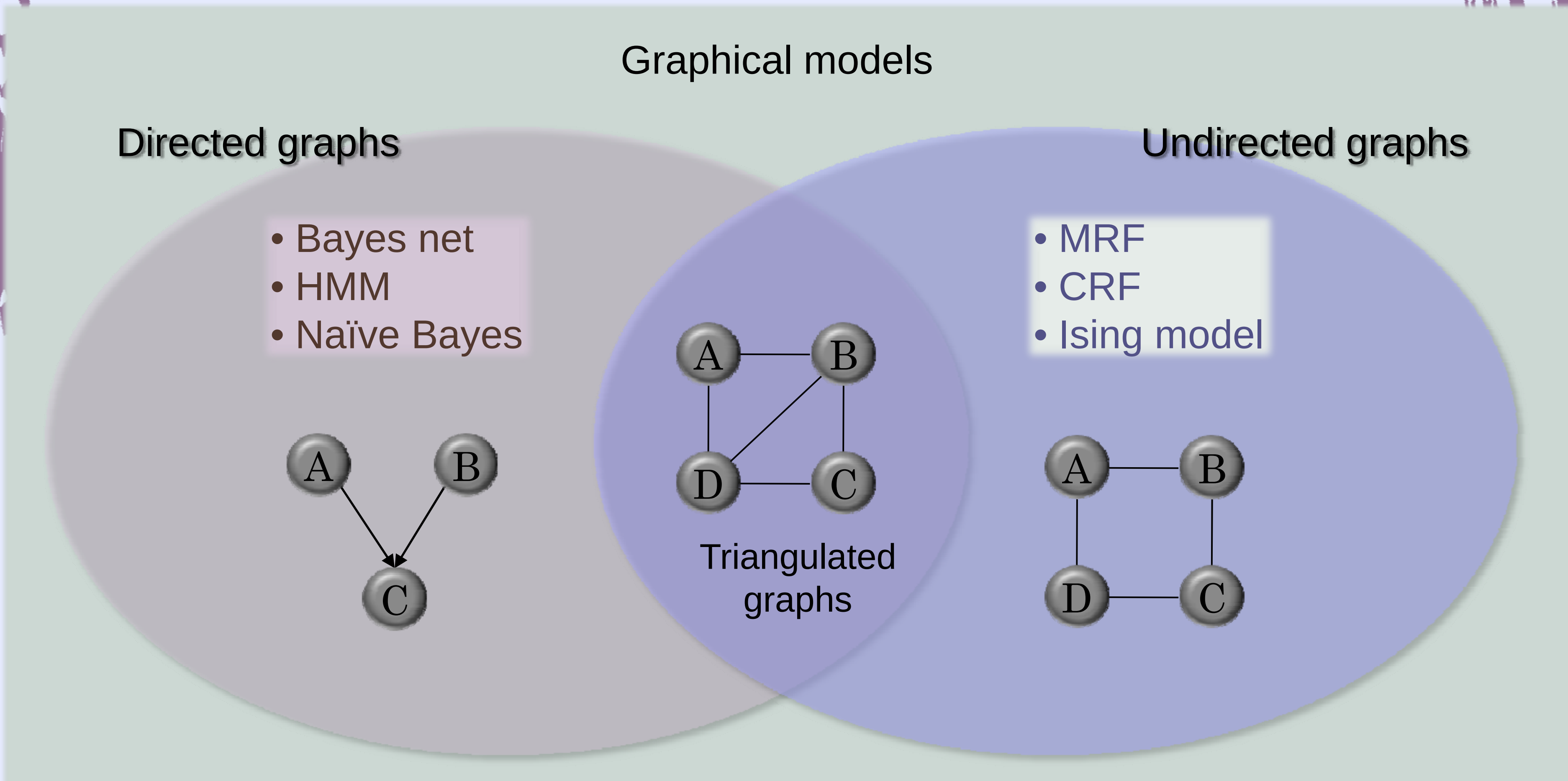
- CBPMs logically unify MRFs, BNs and PCFGs:

  $P(X_1=x_1,..,X_n=x_n) = P_c$(“$X_1=x_1$”,..,“$X_n=x_n$” | KB)

- We apply CBPMs to abduction and propose a new EM algorithm for parameter learning

Graphical models and CBPMs

- Every $P(x) = Z^{-1}\prod_i F_i(x_i)$ has an equivalent CBPM



Simple case

Given $P(X=a, Y=b)$, introduce

- boolean random variables "X=a" and their CNF

$$
\begin{aligned}
XOR(X) &= \left(\bigvee_{a \in V(X)} {}''X=a'' \right) \wedge \bigwedge_{a_1 \neq a_2} \neg({}''X=a_1'' \wedge {}''X=a_2'') \\
EQU &= \bigwedge_{a \in V(X), b \in V(Y)} ({}''X=a'' \wedge {}''Y=b'' \Leftrightarrow \theta_{ab}) \\
KB &= XOR(X) \wedge XOR(Y) \wedge EQU
\end{aligned}
$$

- distribution making variables independent

$$
\begin{aligned}
P_c({}''X=a'', {}''Y=b'', \theta_{ab}) &= P_c({}''X=a'')P_c({}''Y=b'')P_c(\theta_{ab}) \\
P_c({}''X=a'') &= 1/2 \text{ for } \forall a \\
P_c(\theta_{ab}) &= \frac{P({}''X=a'', {}''Y=b'')}{1 + P({}''X=a'', {}''Y=b'')}
\end{aligned}
$$

Simple case (cont'd)

$$KB \Leftrightarrow \bigvee_{a_1 \in V(X), b_1 \in V(Y)} \left\{ \left("X = a_1" \wedge \bigwedge_{a_2 \neq a_1} \neg "X = a_2" \right) \wedge \left("Y = b_1" \wedge \bigwedge_{b_2 \neq b_1} \neg "Y = b_2" \right) \wedge \left(\theta_{a_1,b_1} \wedge \bigwedge_{(a_2,b_2) \neq (a_1,b_1)} \neg \theta_{a_2,b_2} \right) \right\}$$

$$Pc(KB) = \sum_{a \in V(X), b \in V(Y)} K \cdot \left(\frac{P_c("X = a")}{\neg P_c("X = a")} \right) \left(\frac{P_c("Y = b")}{\neg P_c("Y = b")} \right) \left(\frac{P_c(\theta_{ab})}{\neg P_c(\theta_{ab})} \right)$$

Since $P_c("X = a") = .. = 1/2$,

$$\begin{aligned} Pc("X = a", "Y = b" \mid KB) &= \frac{\left(\frac{P_c(\theta_{ab})}{\neg P_c(\theta_{ab})} \right)}{\sum_{a',b'} \left(\frac{P_c(\theta_{a'b'})}{\neg P_c(\theta_{a'b'})} \right)} \\ &= \frac{P(X = a, Y = b)}{\sum_{a',b'} P(X = a', Y = b')} \\ &= P(X = a, Y = b) \end{aligned}$$

Product case

Given product $P(X=a, Y=b, Z=c) = F_1(a,b)F_2(b,c)$, we define

- factor distributions, following the simple case
  $Q^{(1)}(a,b) = \frac{F_1(a,b)}{\sum_{a,b} F_1(a,b)} = P_c^{(1)}(''X=a'', ''Y=b'' \mid KB^{(1)})$
  $Q^{(2)}(b',c) = \frac{F_2(b',c)}{\sum_{b',c} F_2(b',c)} = P_c^{(2)}(''Y'=b''', ''Z=c'' \mid KB^{(2)})$
- product distribution
  $P_c(''X=a'', ''Y=b'', ''Y'=b''', ''Z=c'', \ldots) = P_c^{(1)}(\cdot)P_c^{(2)}(\cdot)$

Then,

$$
\begin{aligned}
& P(X=a, Y=b, Z=c) \\
= \ & \frac{Q^{(1)}(a,b)Q^{(2)}(b,c)}{\sum_{a,b,c} Q^{(1)}(a,b)Q^{(2)}(b,c)} \\
= \ & P_c(''X=a'', ''Y=b'', ''Z=c'' \mid ''Y=Y''' \wedge KB^{(1)} \wedge KB^{(2)})
\end{aligned}
$$

where $''Y=Y''' = \bigvee_b \left(''Y=b'' \wedge ''Y'=b''\right)$

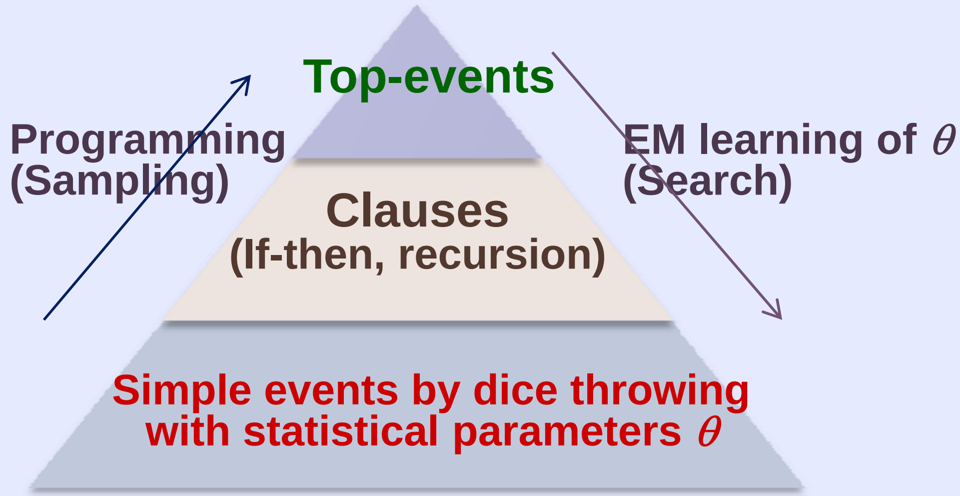
Generative modeling

- Defines a generation process of an output in a sample space
  - Bayesian approach such as LDA
    - prior distribution p(θ|α) ➔ distribution p(D|θ) ➔ data D
    - Given D, predict x by

$$p(\theta \mid D) \propto \int_\theta p(\theta \mid \alpha) p(D \mid \theta) d\theta, \; p(x \mid D) = \int_\theta p(x \mid \theta) p(\theta \mid D) d\theta$$

  - Probabilistic grammars such as PCFGs
    - Rules are chosen probabilistically in the derivation
    - Prob. of sentence s : $p(s) = \sum_{\tau \models s} \overbrace{\prod_{r \in \tau} p(r)^{\phi(r,\tau)}}^{p(\tau)}$
  - Defining distributions by programs
    - PHA[Poole'93], PRISM[Sato et al.'95,97], SLPs[Muggleton'96,Cussens'01], P-log[Baral et al.'04], LPAD[Vennekens et al.'04], ProbLog[De Raedt et al.'07]…

PRISM (PRogramming In Statistical Modeling)



- Tool for generative modeling in machine learning
- Probabilistic extension of Prolog for complex data (*beyond* traditional tabular data)
- has a probabilistic possible worlds semantics named "*distribution semantics*"
- has unified probability computation/parameter learning mechanisms based on PPC
- Equal time complexity guaranteed for BNs, HMMs and PCFGs
- Advanced mechanisms such as VB and DAEM available

Generative model (finite domain)

PRISM programs $DB = F \cup R$ cover generative models

- F: $\mathtt{msw}$ atoms with base distribution $P_{\mathtt{msw}}(\cdot)$

  $\mathtt{msw}(i,t,v)$: throwing dice i at t returns v

  R: set of definite clauses
- $P_{\mathtt{msw}}(\cdot)$ is extended using DB to $P_{DB}(\cdot)$ for all ground atoms
- For goal G, find all E_ks such that $E_k, DB \vdash G$ where $E_k = \mathtt{msw}_1^{(k)} \wedge \cdots \wedge \mathtt{msw}_{m_k}^{(k)}$, giving $P_{DB}(G) = \sum_k P_{\mathtt{msw}}(E_k)$

Introduce $P_c(\cdot)$ (as before) and constratins;

- $\mathrm{iff}^g(R)$: iff-form of some ground instantiations of R

  $\mathrm{iff}^g(R) \models_H G \Leftrightarrow E_1 \vee \cdots \vee E_h$
- $XOR_{\mathtt{msw}}$: boolean formula stating $\mathtt{msw}$ atoms are exclusive

$$\begin{aligned} P_c(G \mid \mathrm{iff}^g(R) \wedge XOR_{\mathtt{msw}}) &= P_c(E_1 \vee \cdots \vee E_h \mid XOR_{\mathtt{msw}}) \\ &= \sum_k P_c(E_k \mid XOR_{\mathtt{msw}}) \\ &= \sum_k P_{\mathtt{msw}}(E_k) = P_{DB}(G) \end{aligned}$$

Generative model (infinite domain)

Assume PRISM program $DB = F \cup R$ and $P_c(\cdot)$ as before

- iff(R): iff-form of R in DB e.g.

  $$\forall x, y\,(mem(x,y) \Leftrightarrow \exists z, w\;(y = [x|z] \vee (y = [w|z] \wedge mem(x,z))))$$

- We also assume for a goal G and its explanations E_i, iff$(R) \models_H G \Leftrightarrow E_1 \vee \cdots \vee E_h$ in the H-universe of DB
- $XOR_{\mathtt{msw}}$: infinite set of boolean formulas stating msw atoms are exclusive
- We can prove that prob. measure $P_c^{\infty}(\cdot \mid \text{iff}(R) \wedge XOR_{\mathtt{msw}})$ over H-interepretations for DB exists for some class of PRISM programs (including PCFGs) such that

$$P_c^{\infty}(G \mid \text{iff}(R) \wedge XOR_{\mathtt{msw}}) \;=\; \sum_k P_{\mathtt{msw}}(E_k) = P_{DB}(G)$$

Infinite domain (1)

$\{\phi_1, \phi_2, \ldots\}$: satisfiable set of boolean formulas in "$X = x$"'s

- Define
$$\begin{aligned} P_c^{k,n}(X_1 = x_1, \ldots, X_k = x_k) \\ = \ & P_c(X_1 = x_1, \ldots, X_k = x_k \mid \phi_1, \ldots, \phi_n) \\ n_{0,i} = \ & i \ \ (i = 1, 2, \ldots) \end{aligned}$$
- Repeat k: Given distribution sequence $\left(P_c^{k,n_{k-1,i}}(\cdot)\right)_i$, choose a convergent subsequnce $n_{k,i}$ of $n_{k-1,i}$ s.t. $\lim_i P_c^{k,n_{k,i}}(X_1 = x_1, \ldots, X_k = x_k)$ exists for any $x_1, \ldots, x_k$
- $n_{i,i}$ is a subsequence of every $n_{k,i}$ $(k \geq 1)$ s.t.

$$\begin{aligned} & P_c^{k,\infty}(X_1 = x_1, \ldots, X_k = x_k) \\ & = \lim_i P_c(X_1 = x_1, \ldots, X_k = x_k \mid \phi_1, \ldots, \phi_{n_{i,i}}) \end{aligned}$$

exists for any $x_1, \ldots, x_k$ and $k (k \geq 1)$

Infinite domain (2)

- By construction,

$$\sum_{x_{k+1}} P_c^{k+1,\infty}(X_1 = x_1, \ldots, X_k = x_k, X_{k+1} = x_{k+1})$$
$$= \sum_{x_{k+1}} \lim_{i \to \infty} P_c^{k+1,n_{i,i}}(X_1 = x_1, \ldots, X_{k+1} = x_{k+1})$$
$$= P_c^{k,\infty}(X_1 = x_1, \ldots, X_k = x_k)$$

- Kolmogorov's extension theorem applied to $\{P_c^{k,\infty}(\cdot) \mid k \geq 1\}$ guarantees a prob. measure $P_c^{\infty}(\cdot \mid \bigwedge_i^{\infty} \phi_i)$ exists on the set of assignments (H-interpretations) s.t.

$$P_c^{\infty}(X_1 = x_1, \ldots, X_k = x_k \mid \bigwedge_i^{\infty} \phi_i)$$
$$= P_c^{k,\infty}(X_1 = x_1, \ldots, X_k = x_k) \quad \text{for } \forall k \geq 1$$
$$P_c^{\infty}(\varphi \mid \bigwedge_i^{\infty} \phi_i) = 1 \text{ if } \bigwedge_i^{\infty} \phi_i \vdash \varphi$$

So, CBPMs are …

- General: graphical/rule-based models are CBPMs
- Uniform: every variable is boolean
  - all variables are independent
    - ➔ $P_c()$ exists for infinitely many variables
    - ➔ sum-product computation (on BDDs) possible
- Expressive:
  - ϕ, KB in $P_c(\phi \mid KB)$ can be first-order (infinite domain)
  - value-wise dependency (CSI in BNs, etc)
  - Maybe less efficient in known models (BNs,PCFGs,..)
- We apply CBPMs to statistical abduction, which statistically infers the best explanation E for observation O, using KB such that $E \wedge KB$ is consistent and $E \wedge KB \vdash O$

Abduction in a metabolic network [Tamaddoni-Nezhad et al.'06, Inoue et al.'09]

- Observation: metabolite concentration (20 observations)

  concentration('1-2-aminoadipate',up,8),
  concentration('succinate',down,8) ,…
- KB: what we know about the network

  reaction('1-2-aminoadipate',2.6.1.39','2-oxo-glutarate'),…
  concentration(Y,up,T) ←
  reaction(X,Enz,Y) ∧ ¬ inhibited(Enz,X,Y,T),…
  - Cyclic dependencies (loops in the network)
- Explanation: conjunction of abducibles (inhibition states)
  - 24 abducibles (inhibited atoms) assumed, 66 explanations found (each is a conjunction of 15 abducibles) by SOLAR such that $E_t \wedge KB \ \vdash O_1 \wedge \ldots \wedge O_{20}$

    E_1 = inhibited(2.6.1.39,'l-2-aminoadipate','2-oxo-glutarate',8) ∧…
    E_2 ,…,E_{66}

A key problem

- We wish to select the best explanation from multiple explanations $E_1,...,E_N$: $E_h \wedge KB \vdash O$
  - Assume $P_c(\cdot|\theta)$ that makes all atoms independent with probabilities θ
  - Select $E_{best} = \mathrm{argmax}_E P_c(E|KB \wedge O,\theta)$
- Learning θ from O in machine learning
  - When correct answer E_h is known (supervised):
    $\theta^* = \mathrm{argmax}_\theta P_c(E_h|KB \wedge O,\theta)$
  - Correct answer unknown (unsupervised):
    $\theta^* = \mathrm{argmax}_\theta P_c(O|KB,\theta)$

Parameter learning for abduction

- Unsupervised learning :
  - $O^{(t)}$: observations ($1 \leq t \leq T$)
  - $E^{(t)}$: disjunction of explanations abduced from $KB^{(t)}$ and $O^{(t)}$, i.e. $E^{(t)} = E_1^{(t)} \vee \ldots \vee E_N^{(t)}$ s.t. for h ($1 \leq h \leq N$)

$$E_h^{(t)} \wedge KB^{(t)} \;|\!-\; O^{(t)}$$

  - $KB^{(t)}$: set of clauses in KB used in the proof above
- We learn parameters θ by MLE applied to the $O^{(t)}$'s

$$L(\theta) = \Pi_t P_c(E^{(t)} | KB^{(t)}, \theta)$$
$$\theta^* = \mathrm{argmax}_\theta L(\theta)$$

using the EMC (EM with constraints) algorithm

The EMC algorithm

$$\boldsymbol{\theta} = \mathsf{argmax}_{\boldsymbol{\theta}} \prod_{t=1}^{T} P(O^{(t)} \mid KB^{(t)}, \boldsymbol{\theta})$$

$$\mathcal{W}_1^{(t)} = \{u_t \mid u_t \models \neg KB^{(t)}\},\ \mathcal{W}_2^{(t)} = \{x_t \mid x_t \models O^{(t)} \wedge KB^{(t)}\}$$

Repeat

$$\eta_\theta^v[s] = \sum_{t=1}^{T} \frac{1}{1 - P(\neg KB^{(t)})} \sum_{u_t \in \mathcal{W}_1^{(t)}} \sigma_{s,v}(u_t) \prod_{s' \in \mathcal{S}} \prod_{v' \in \{1,0\}} \theta_{s',v'}^{\sigma_{s',v'}(u_t)} + \frac{1}{P(O^{(t)} \wedge KB^{(t)})} \sum_{x_t \in \mathcal{W}_2^{(t)}} \sigma_{s,v}(x_t) \prod_{s' \in \mathcal{S}} \prod_{v' \in \{1,0\}} \theta_{s',v'}^{\sigma_{s',v'}(x_t)}$$

$$P(\neg KB^{(t)} \mid \boldsymbol{\theta}) = \sum_{u_t \in \mathcal{W}_1^{(t)}} \prod_{s \in \mathcal{S}} \prod_{v \in \{1,0\}} \theta_{s,v}^{\sigma_{s,v}(u_t)}$$

$$P(O^{(t)} \wedge KB^{(t)} \mid \boldsymbol{\theta}) = \sum_{x_t \in \mathcal{W}_2^{(t)}} \prod_{s \in \mathcal{S}} \prod_{v \in \{1,0\}} \theta_{s,v}^{\sigma_{s,v}(x_t)}$$

$$\widehat{\theta}_{s,v} = \frac{\eta_\theta^v[s]}{\eta_\theta^1[s] + \eta_\theta^0[s]}$$

EMC for abduction

- EMC is applicable to log-linear models in general
- Abduction infers the best explanation E for observation O, using KB s.t. E ∧ KB $\vdash^*$ O and E ∧ KB consistent
  - EMC provides a generic parameter learning algorithm for statistical abduction

$\vdash^*$	*KB*	formalism	parameter learning (EM)
$\vdash$	Horn	PHA, SLP, PRISM	gEM,FAM
$\models_s$	acyclic	ICL	
$\vdash$	any	SOLAR	

EMC applicable

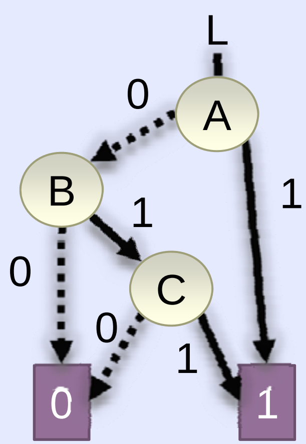
The BDD-EMC Algorithm

Table 1: Definitions of sympols

Δ	set of BDDs
δ_X	BDD for X
$N(\delta)$	set of nodes consisting of δ
$V(\delta)$	set of variables consisting of δ
Π	set of parameters
$Var(n)$	variable correspond to node n
$Par(v)$	parameter correspond to variable v
$Ch_0(n)$	1-child of node n
$Ch_1(n)$	0-child of node n
θ_π	probability of variables correspond to parameter π being true
$\theta_{\bar\pi}$	$1-\theta_\pi$
$\mathrm{B}[n]$	backward probability of node n
$\mathrm{F}[n]$	forward probability of node n
$\eta_1[\pi]$	expectation of variables correspond to parameter π being true
$\eta_0[\pi]$	expectation of variables correspond to parameter π being false

$$\Delta = \{\delta_{KB}, \delta_{E^{(1)}\wedge KB}, \ldots, \delta_{E^{(T)}\wedge KB}\}$$
$$N(\delta) = \{n_1, \ldots, n_{|N(\delta)|}\}$$
$$V(\delta) = \{v_1, \ldots, v_{|N(\delta)|}\}$$

$L \Leftrightarrow A \vee (B \wedge C)$




```
Procedure: BDD-EMC(Δ,Π)
  repeat
    initialize(Δ,Π);
    Estep(Δ);
    Mstep(Π);
  until parameters converge
end
```

```
Procedure: initialize(Δ,Π)
  for all δ_X ∈ Δ do
    for all n ∈ N(δ_X) do
      B[n] = 0; F[n] = 0;
    end for
  end for
  for all π ∈ Π do
    η1[π] = 0; η0[π] = 0;
  end for
end
```

```
Procedure: Estep(Δ)
  for all δ_X ∈ Δ do
    backward(δ_X);
    forward(δ_X);
  end for
  expectationKB(δ_KB);
  for all δ_X ∈ Δ \ δ_KB do
    expectation(δ_X);
  end for
end
```

```
Procedure: Mstep(Π)
  for all π ∈ Π do
    θ_π := η1[π]/(η1[π] + η0[π]);
  end for
end
```

```
Procedure: backward(δ_X)
  B[⊤] = 1; B[⊥] = 0;
  for i = |N(δ_X)| to 1 do
    π = Par(Var(n_i));
    B[n_i] = θ_π B[Ch0(n_i)]
             +θ_π̄ B[Ch1(n_i)];
  end for
end
```

```
Procedure: forward(δ_X)
  F[n_1] = 1;
  for i = 1 to |N(δ_X)| do
    π = Par(Var(n_i));
    F[Ch0(n_i)] += θ_π F[n_i];
    F[Ch1(n_i)] += θ_π̄ F[n_i];
  end for
end
```

```
Procedure: expectation(δ_X)
  for i = 1 to |N(δ_X)| do
    π = Par(Var(n_i));
    η1[n_i] += F[n_i] B[Ch0(n_i)] θ_π/B[n_1];
    η0[n_i] += F[n_i] B[Ch1(n_i)] θ_π̄/B[n_1];
  end for
end
```

```
Procedure: expectationKB(δ_KB)
  for i = 1 to |N(δ_KB)| do
    π = Par(Var(n_i));
    η1[n_i] +=
      F[n_i] (1−B[Ch0(n_i)])θ_π/B[n_1];
    η0[n_i] +=
      F[n_i] (1−B[Ch1(n_i)])θ_π̄/B[n_1];
  end for
end
```

Rich friends (1)
-- cyclic dependencies

KB:

rich(X) ⇔ smart(X) ∨
∃Y (friend(X,Y) ∧ rich(Y) ∧ generous(Y))
friend(a,b)
friend(b,c)
generous(b)
friend(X,Y) 🡐 friend(Y,X)

- Smart people are rich
- People are rich iff they have a rich and generous friend
- “b” is generous, “a” and “b” are friends and so are “b” and “c” but we don’t know about “a” and “c”
- The state of rich(a) and rich(c) observed several times like a(yes:20 / no:10), c(yes:1 0/ no:20) from which we wish to know if “b” is rich

Rich friends (2)
-- parameter learning by EMC

Atoms	Observations		
	a(30/0)c(30/0)	a(20/10)c(10/20)	a(0/30)c(0/30)
friend(a,b)	1.00000	1.00000	1.00000
friend(b,c)	1.00000	1.00000	1.00000
friend(c,a)	0.61979	0.58768	0.59516
generous(a)	0.49553	0.30898	0.43071
generous(b)	1.00000	1.00000	1.00000
generous(c)	0.49909	0.47435	0.34752
smart(a)	0.53169	0.54293	0.00000
smart(b)	1.00000	0.00190	0.00000
smart(c)	0.56106	0.05930	0.00000
rich(a)	1.00000	0.66666	0.00000
rich(b)	1.00000	0.31058	0.00000
rich(c)	1.00000	0.33334	0.00000

Rich friends (3)
-- logical-probabilistic inference

- KB: $rich(X) \Leftrightarrow smart(X) \vee$
  $\exists Y\ (friend(X,Y) \wedge rich(Y) \wedge generous(Y))$
  friend(a,b)
  friend(b,c)
  generous(b)
  $friend(X,Y) \leftarrow friend(Y,X)$

- Notice KB $\vdash rich(a) \leftarrow rich(b)$
  Hence, $P_c(rich(a) \leftarrow rich(b) \mid KB) = 1$
  So $P_c(rich(a) \mid KB) >= P_c(rich(b) \mid KB)$
  Similarly $P_c(rich(c) \mid KB) >= P_c(rich(b) \mid KB)$

Concluding remarks

- CBPMs (constraint-based probabilistic models) $P_c(\phi \mid KB)$ are proposed in which independent boolean variables are constrained by KB
    - They cover both graphical models (BNs,MRFs) and generative models (PCFGs,PRISM)
- The EMC (EM with constraints) algorithm is proposed for the parameter learning of CBPMs
    - EMC works for log-linear models in general
    - Unlike the double-loop IM algorithm for PCLP [Riezler'98], EMC is a single-loop algorithm
    - For efficiency, it is implemented on BDDs (binary decision diagrams) as the BDD-EMC algorithm