Propositionalized Probabilistic Modeling and Bayesian Learning in PRISM

Taisuke Sato
Tokyo Institute of Technology

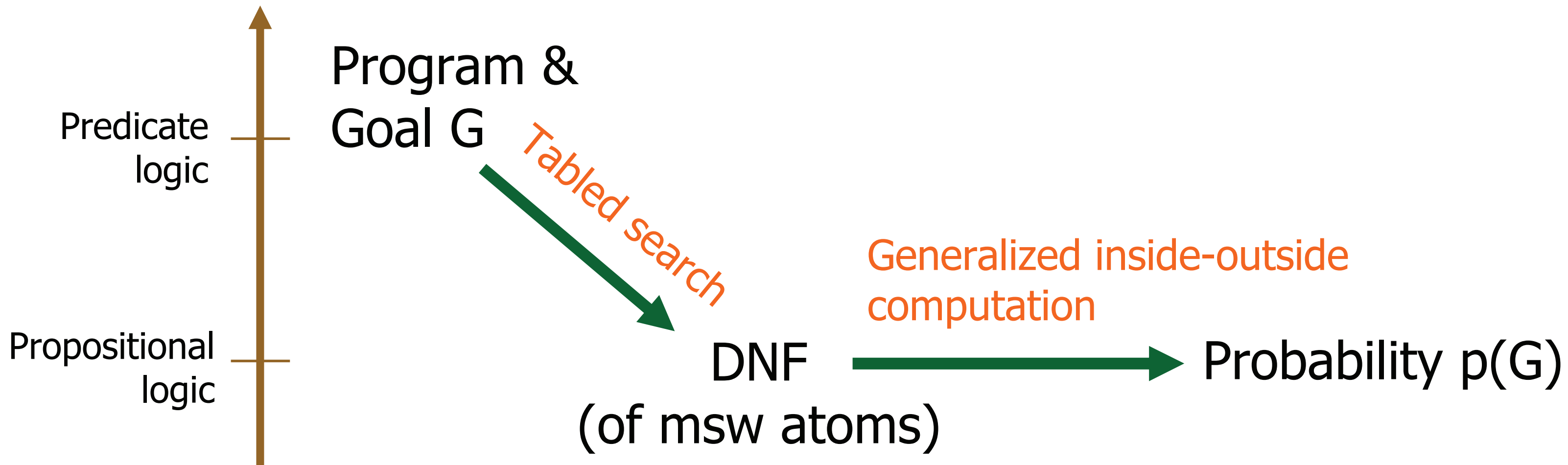
Motivation

- PPC (propositionalized probability computation)
  - Reducing structured models to Boolean formulas
  - Uniformly and efficiently implemented in PRISM
- Bayesian approach
  - Overfitting avoided by penalizing complex models
  - Generally intractable but a good approximation possible by VB (variational Bayes)
- Combining the two in PRISM
  - Generic, robust yet efficient tool for Bayesian modeling of structured objects
  - Covers known (BNs,HMMs,PCFGs) and unknown models

PPC (propositionalized probability computation)

- PPC : computing probability via propositional formulas by considering “X=x” as a probabilistic proposition, i.e. binary random variable
  - Used in various probabilistic models
    - BNs (DNNF by Darwiche ’05, A/O trees by Dechter ’05, ZBDDs by Minato ’07)
    - PCFGs (IO by Baker ‘79, Expl. graphs by Kameya ’00)
    - Log-linear models (CFDs by McAllester ’04)
- Efficient
  - BNs : 0 prob. pruning+CSI (context specific independence)
  - PCFGs : much faster than the traditional IO algorithm (Sato ’01)
- Uniform
  - From “X=x” to DNF (or BDD, ZBDD, CFD, Expl.graph,…)
  - Compute p(DNF) by the sum-product algorithm

PRISM's PPC



Predicate logic

Propositional logic

Program & Goal G

Tabled search

DNF (of msw atoms)

Generalized inside-outside computation

Probability p(G)

Bayesian network -- going to NewYork

X_1: Tokyo weather

$x_1 \in \{$ rain, clear $\}$

X_2: departure delay

$x_2 \in \{$ yes, no $\}$

X_3: NY weather

$x_3 \in \{$ rain, clear $\}$

X_4: arrival delay

$x_4 \in \{$ yes, no $\}$

X_5: baseball-cancelled

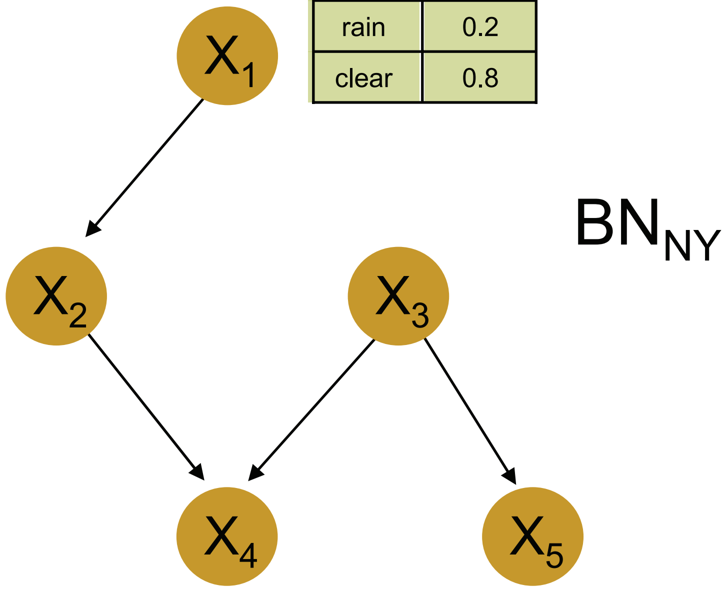
$x_5 \in \{$ yes, no $\}$

CPT

	$P(x_2 \mid x_1)$	$P(x_2 \mid x_1)$
x_2 / x_1	rain	clear
yes	0.3	0.1
no	0.7	0.9

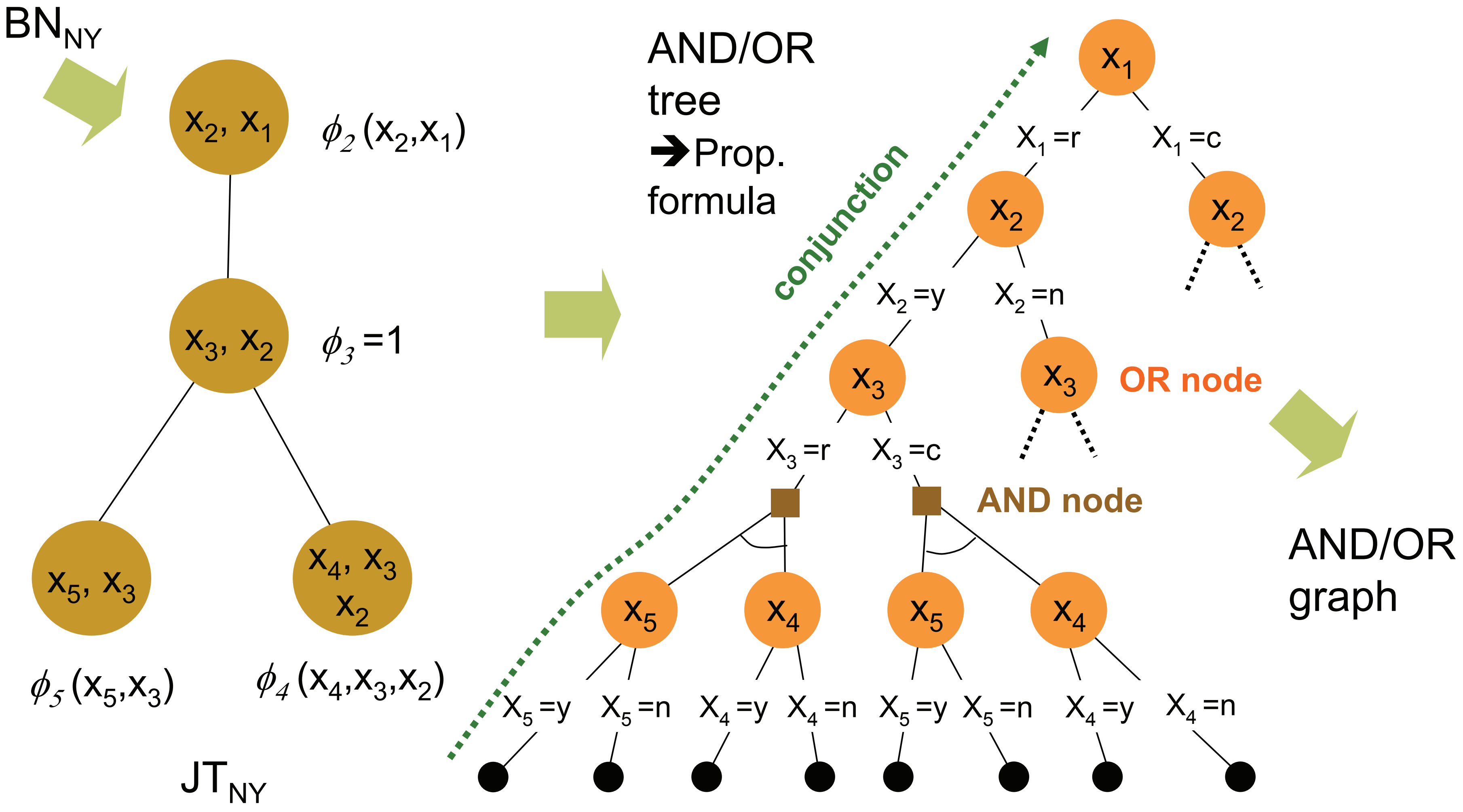
CPT

x_1	$P(x_1)$
rain	0.2
clear	0.8



$$p(X_1=x_1,X_2=x_2,X_3=x_3,X_4=x_4,X_5=x_5)$$
$$= p(x_1)p(x_2 \mid x_1)p(x_4 \mid x_2,x_3)p(x_3)p(x_5 \mid x_3)$$

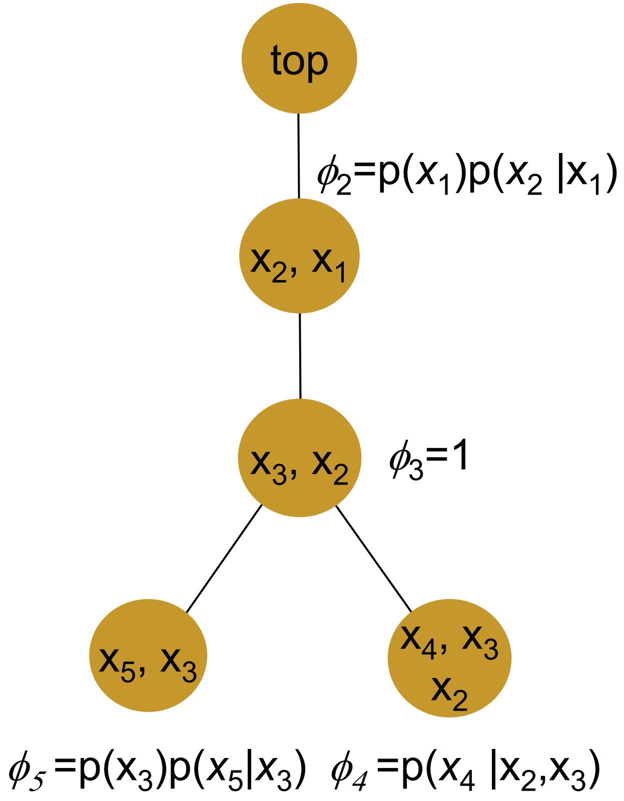
PPC using junction trees



AND/OR tree
➔Prop. formula

AND/OR graph

JT program in PRISM


top
ϕ_2=p(x_1)p(x_2 |x_1)
x_2, x_1
x_3, x_2
ϕ_3=1
x_5, x_3
x_4, x_3 x_2
ϕ_5 =p(x_3)p(x_5|x_3)
ϕ_4 =p(x_4 |x_2,x_3)

JT_{NY}

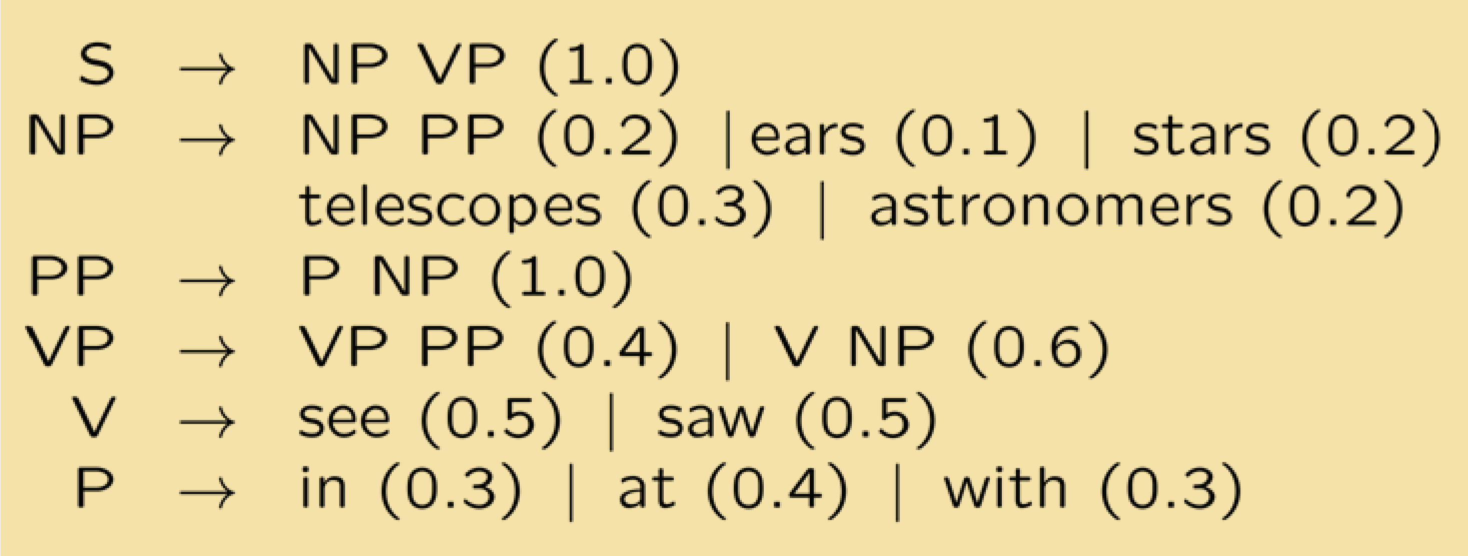
Logical description:

top:- $node_{\{2,1\}}(X_2,X_1)$.
$node_{\{2,1\}}(X_2,X_1)$:- $cpt_{\{2,1\}}([X_2],X_1)$,$cpt_{\{1\}}([],X_1)$,
$\quad$ $node_{\{3,2\}}(X_3,X_2)$.
$node_{\{3,2\}}$ (X_3,X_2):-
$\quad$ $node_{\{5,3\}}(X_5,X_3)$,$node_{\{4,3,2\}}(X_4,X_3,X_2)$.
$node_{\{5,3\}}$ (X_5,X_3)- $cpt_{\{5,3\}}([X_3],X_5)$,$cpt_{\{3\}}([],X_3)$.
$node_{\{4,3,2\}}(X_4,X_3,X_2)$:- $cpt4_{\{3,2\}}([X_2,X_3],X_4)$.

- PRISM's PPC applied to this program simulates two phases in BP
  - collecting evidence of BP in JT_{NY}
  - distributing evidence of BP in JT_{NY}

PCFGs(probabilistic context free grammars)

PCFG = CFG + probability



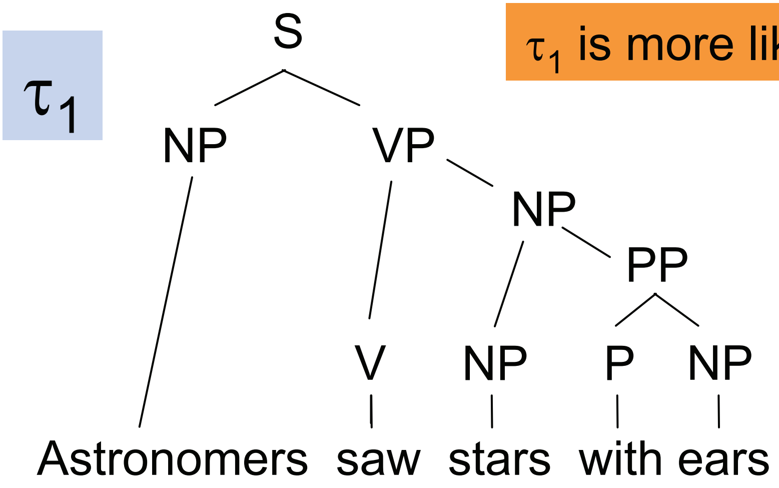
$$
\begin{array}{rcl}
S & \rightarrow & NP\ VP\ (1.0) \\
NP & \rightarrow & NP\ PP\ (0.2)\ |\ \text{ears}\ (0.1)\ |\ \text{stars}\ (0.2) \\
 & & \text{telescopes}\ (0.3)\ |\ \text{astronomers}\ (0.2) \\
PP & \rightarrow & P\ NP\ (1.0) \\
VP & \rightarrow & VP\ PP\ (0.4)\ |\ V\ NP\ (0.6) \\
V & \rightarrow & \text{see}\ (0.5)\ |\ \text{saw}\ (0.5) \\
P & \rightarrow & \text{in}\ (0.3)\ |\ \text{at}\ (0.4)\ |\ \text{with}\ (0.3)
\end{array}
$$

$S \rightarrow^{*}$ Astronomers saw stars with ears

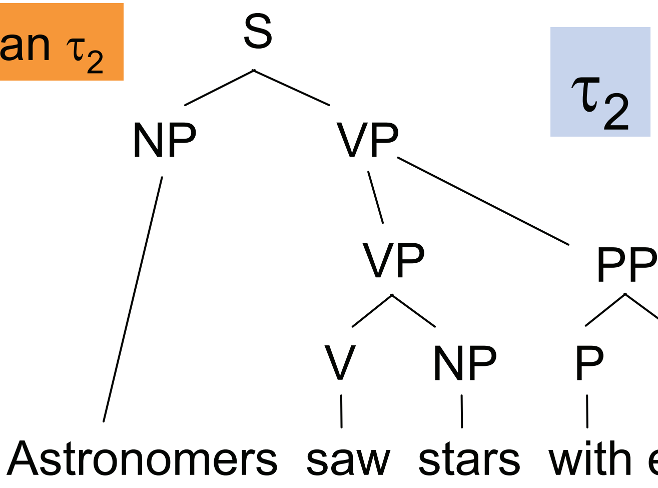
Disambiguation

```
  S  →  NP VP (1.0)
 NP  →  NP PP (0.2) |ears (0.1) | stars (0.2)
        telescopes (0.3) | astronomers (0.2)
 PP  →  P NP (1.0)
 VP  →  VP PP (0.4) | V NP (0.6)
  V  →  see (0.5) | saw (0.5)
  P  →  in (0.3) | at (0.4) | with (0.3)
```

τ_1 is more likely than τ_2



$P(\tau_1)=1.0*0.2*0.6*0.5*0.4*0.2*1.0*0.3*0.1$
$=0.00192$



$P(\tau_2)=1.0*0.2*0.4*0.6*0.5*0.2*1.0*0.3*0.1$
$=0.00144$

PCFGs (contd.)

- Probabilistic derivation of sentences
  - Rules have choice probabilities (parameters $\theta = \theta_{A\to\alpha_1}, \dots, \theta_{A\to\alpha_n}$)

$$\overbrace{P(A \to \alpha_1)}^{\theta_{A\to\alpha_1}} + \cdots + \overbrace{P(A \to \alpha_n)}^{\theta_{A\to\alpha_n}} = 1$$

  - Probability of a sentence is the sum of products of parameters associated with rules in a derivation

$$P(s \mid \theta) = \sum_{\tau \in parse(s)} \prod_{occ(A\to\alpha)\in\tau} \theta_{A\to\alpha}$$

- Parameters are estimated by ML estimation
  - The IO (inside-outside) algorithm [Baker'79] used

PCFG program in PRISM

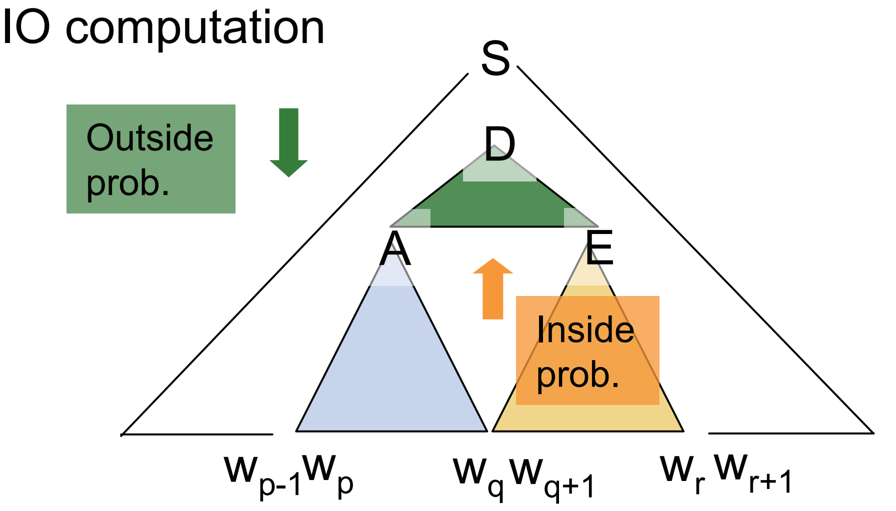
- PCFG = { A -> RHS_1,..,RHS_k,… }

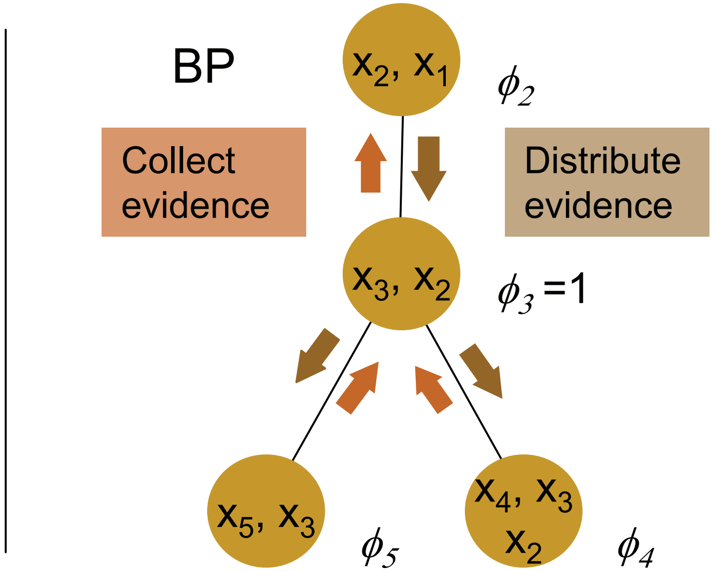
```
pdcg(L):- start_symbol(A), pdcg2(A,L).  % L : sentence
pdcg2(A,L):-                          % nonterminal A derives L
   ( terminal(A) → L=[A]
   ; msw(A,RHS),                      % msw = built-in for
     pdcg3(RHS,L)  ).                 % probabilistic choice
pdcg3([],[],_).
pdcg3([A|R],L3):-
   pdcg2(A,L1), pdcg3(R,L2), append(L1,L2,L3).
```

- PPC in PRISM for this program exactly coincides with IO probability computation for PCFGs

IO and BP in PRISM

- Different
  - Inside-Outside computation for PCFGs (recursion), SNLP community
  - Belief propagation for BNs (no recursion), UAI community
- PRISM’s PPC subsumes both algorithms
  - Goals are reduced to DNFs (explanation graphs)
  - Generalized IO algorithm is applied to explanation graphs





Why is PPC efficient?

- Because PPC realizes value-wise computation
  - Variables are interdependent syntactically but their values are not necessarily so
  - But variable-wise computation such as BP ignores this fact
  - Express value dependencies as propositional formulas by a graph, and apply dynamic programming to compute probabilities (value-wise computation)
- Value-wise computation is
  - effective when 0 prob. and CSI (context specific independence) are abundant (BNs)
  - a must, o.w. exponential time required (PCFGs)

Bayesian approach

- Model parameters θ have a prior distribution $p(\theta)$

  $p(y) = \int p(y|\theta)p(\theta)d\theta$ (for model selection by data y)

  $p(x|y) = \int p(x|\theta)p(\theta|y)d\theta$ (for filtering)

- Bayesian inference
  - ❑ MAP : $\theta_{MAP} = \mathrm{argmax}_\theta\, p(y|\,\theta)p(\theta)$,

    $\ln p(y) \approx \ln p(y|\theta_{MAP}) + \ln p(\theta_{MAP})$
  - ❑ BIC : $\ln p(y) \approx \ln p(y|\theta_{MAP}) – ½\, |\theta_{MAP}| \ln |y|$
  - ❑ Cheeseman-Stutz approximation : use MAP, fast
  - ❑ MCMC sampling : exact but slow
  - ❑ Variational Bayes : approx., fast, better-than CS

Variational Bayes

- Suppose our model p(x,y|θ) has x (hidden) and y (observable) with a prior distribution p(θ)
- We wish to approximate p(y) = ∫ p(x,y|θ)p(θ)dxdθ
- Note for any distribution q(x,θ), we have

$$
\begin{aligned}
\ln p(y) \quad &= \quad \int \left(\ln \frac{p(x,y,\theta)}{q(x,\theta)} \right) q(x,\theta) dx d\theta + J \\
&\qquad \text{where } J = \int \ln \left(\frac{q(x,\theta)}{p(x,\theta \mid y)} \right) q(x,\theta) dx d\theta \quad (\geq 0) \\
&\geq \quad \int \left(\ln \frac{p(x,y,\theta)}{q(x,\theta)} \right) q(x,\theta) dx d\theta \quad (= -F(q))
\end{aligned}
$$

- So, let's maximize the lower bound – F(q) of ln p(y) by minimizing J(q) (F(q) : variational free energy)

The VB-EM scheme

- To simplify the problem, we restrict q(x,θ) = q(x)q(θ)

  and minimize

$$J(q(x), q(\theta)) = \int \ln\left(\frac{q(x)q(\theta)}{p(x,\theta \mid y)}\right) q(x)q(\theta)dxd\theta$$

  under constraints $\int q(\theta)d\theta = 1, \int q(x)dx = 1$

- By applying the calculus of variation, we reach VB-EM scheme

$$q(x) \approx p(x \mid y), q(\theta) \approx p(\theta \mid y)$$

$$\left\{ \begin{array}{lll} q(x) & \propto & \exp \int q(\theta) \ln p(x, y \mid \theta) d\theta \\ q(\theta) & \propto & p(\theta) \exp \int q(x) \ln p(x, y \mid \theta) dx \end{array} \right.$$

- Note q(x) and p(θ) are interdepedent

  ➔ solve them iteratively like EM ➔ VB-EM algorithm

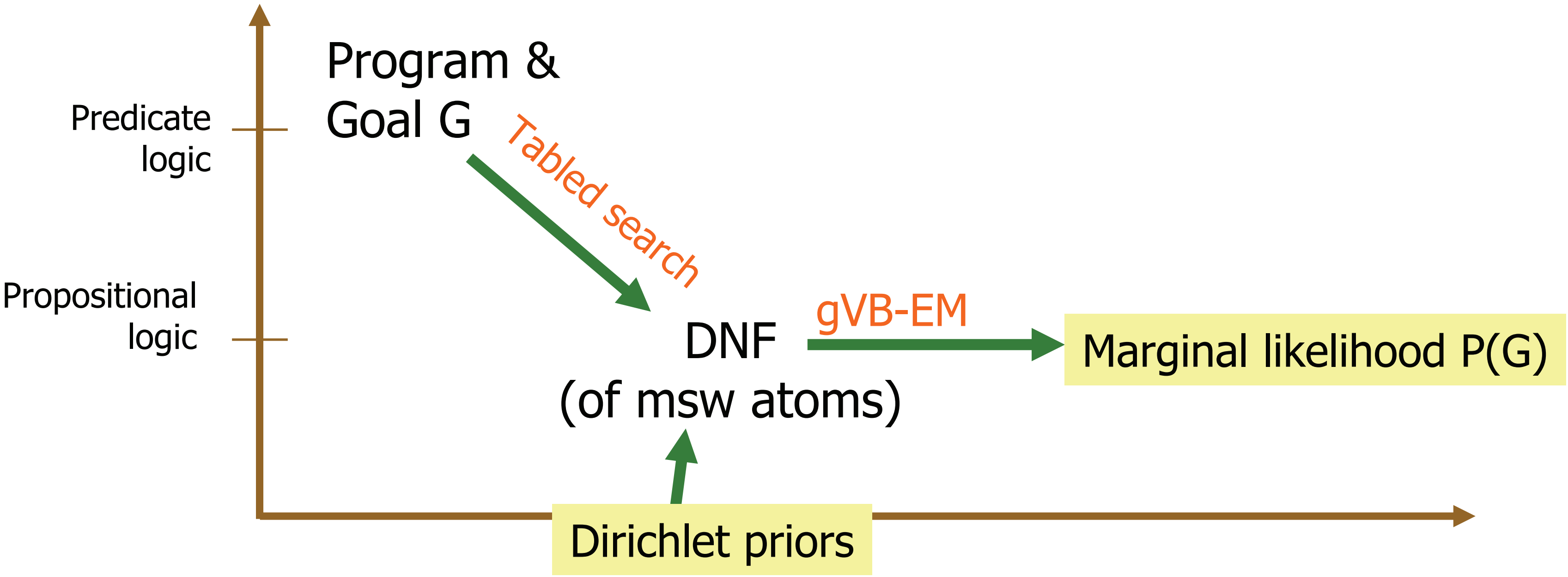
Combining VB and PRISM's PPC

- VB is much more complex to implement than MLE
  - Determine priors and derive a VB-EM algorithm for a model
  - Determine data structure and implement (and debug)
  - Only a few VB-EM algorithms (BNs,PCFGs) have been derived
- We can introduce VB to propositionalized probability computation in PRISM, which will
  - eliminate the need for deriving a VB-EM algorithm specific to each model from scratch, and
  - allow us to explore arbitrarily complex Bayesian models only by writing programs at predicate level (as it happens for EM learning in case of MLE)

Msw atoms and Dirichlet priors

- PRISM has the gEM (graphical EM) algorithm for probability learning
  - Observation G is reduced to DNFs made up of msw atoms (see below) representing the generation paths for G
- "X_i = v" is represented propositionally by a ground atom msw(i,v)
  - msw(i,v) says a probabilistic choice named i yields v
  - If X_i takes one of $\{v_1,..,v_k\}$, msw(i,v_1),..,msw(i,v_k) are exclusively true
  - gEM estimates $\theta_{i,v}$ = p(msw(i,v))
- To derive the gVB-EM algorithm for PRISM,
  - We place a Dirichlet distribution on $\{\theta_{i,v1},..,\theta_{i,vk}\}$ and apply the VB-EM scheme to PRISM's semantics

PPC and VB-PRISM



Predicate logic

Propositional logic

Program & Goal G

Tabled search

DNF (of msw atoms)

gVB-EM

Marginal likelihood P(G)

Dirichlet priors

VB-EM for PRISM

- Suppose $y = (G_1, G_2, \ldots, G_T)$ observed (G_t: observed atom)
- Obtain by SLD search the set $\varphi(G_t)$ of explanations for G_t

  $\varphi(G_t) = \{E_{t1}, \ldots, E_{tk}\}$ such that $G_t \Leftrightarrow E_{t1} \vee \ldots \vee E_{tk}$

  where E_{ti} is a conjunction of msw atoms and E_is are exclusive
- Put $x = (E_1, E_2, \ldots, E_T)$ for $y = (G_1, G_2, \ldots, G_T)$ $(E_t \in \varphi(G_t))$

  x can be considered as (a value of) some hidden variable
- Place a Dirichlet distribution $p(\theta)$
  - I = set of ids for msws, θ_i = set of parameters for msw(i,v_1),..,msw(i,v_k)

  $$p(\theta) = \prod_{i\in I} p(\theta_i) \quad \text{where} \quad p(\theta_i) = \frac{1}{Z_i}\prod_{v\in V_i} \theta_{i,v}^{\alpha_{i,v}-1} \quad \text{and} \quad Z_i = \frac{\prod_{v\in V_i}\Gamma(\alpha_{i,v})}{\Gamma(\sum_{v\in V_i}\alpha_{i,v})}$$
- (x,y) is a complete data whose distribution is

  $$p(x, y \mid \theta) = \prod_{t=1}^{T} p(E_t, G_t) = \prod_{t=1}^{T}\prod_{i\in I}\prod_{v\in V_i} \theta_{i,v}^{\sigma_{i,v}(E_t)}$$

  where $\sigma_{(i,v)}(E_t)$ is #msw(i,v) in a conjunction E_t

MSA(multiple sequnece alignment)

- Alignment of multiple (>2) biological sequences

```
HLKIANRKDKHHNKEFGGHHLA
HLKATHRKDQHHNREFGGHHLA
VLKFANRKSKHHNKEMGAHHLA
HKKGATPVNVS
HKKGATATGNPKHVC
QFKVAAAVGKHQDASRGVHHID
SFKGQGAVSKHQDPEWGVHHID
SFKGQGAVSVPQAPAWGINHID
HFKSQAEVNKHDRPEWGLNQID
HFRSQAEVNQRQFNHHRPQWSFNQIG
SFNVVKGASKRENGGMGAEPVD
KFKKVDGLGKKEHPALGVH
KFMVGGKDGKNRKDAHAHRKVE
KYKVPEKDGKKRTNAHSHRKVE
RYKIPESDGKKRTNSHRHRKVE
RYKIASMDGKKRYAEHKHKKLE
```

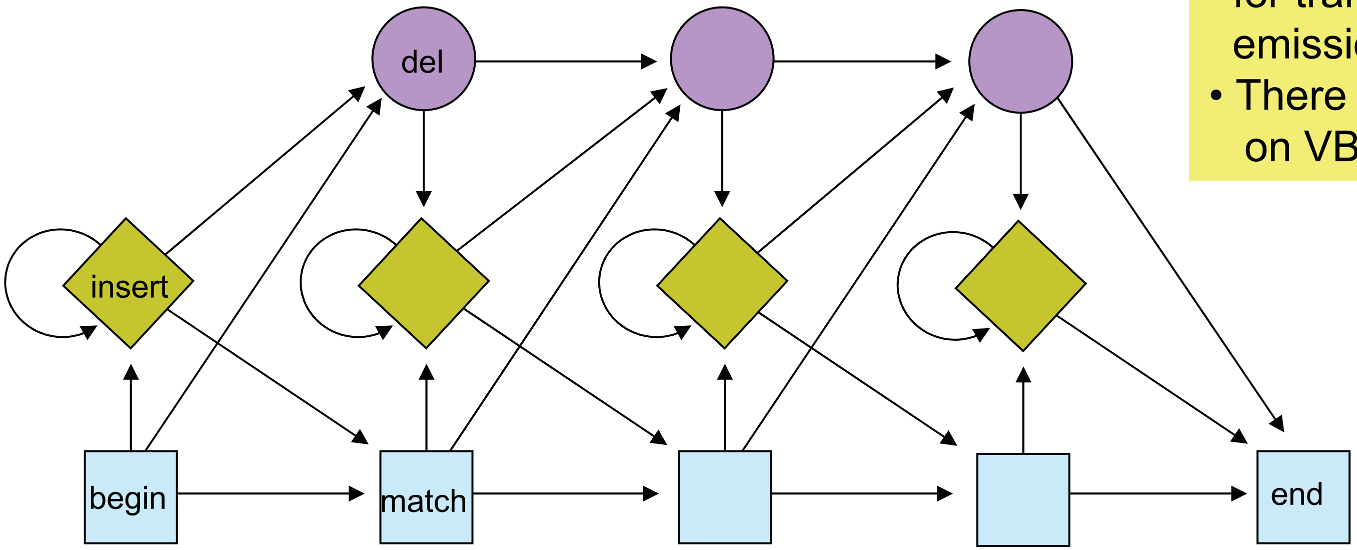
(artificail data)

```
HLKIANRKDK----HHNKEFGGHHLA
HLKATHRKDQ----HHNREFGGHHLA
VLKFANRKSK----HHNKEMGAHHLA
HKKGAT---------------PVNVS
HKKGATATG-----------NPKHVC
QFKVAAAVGK----HQDASRGVHHID
SFKGQGAVSK----HQDPEWGVHHID
SFKGQGAVSV----PQAPAWGINHID
HFKSQAEVNK----HDRPEWGLNQID
HFRSQAEVNQRQFNHHRPQWSFNQIG
SFNVVKGASK----RENGGMGAEPVD
KFKKVDGLGK----KEHPALGVH---
KFMVGGKDGK----NRKDAHAHRKVE
KYKVPEKDGK----KRTNAHSHRKVE
RYKIPESDGK----KRTNSHRHRKVE
RYKIASMDGK----KRYAEHKHKKLE
```

(correct alignment)

Profile-HMMs

- Variant of HMMs
  - del : no output symbol (ε symbol)
  - insert : any symbol, self-loop possible
  - match : any symbol, next state



- Many parameters for transition and emission to estimate
- There seems no literature on VB + profile-HMMs

Profile-HMMs in PRISM

```
target(observe,1).
data(‘phmm.dat’).

observe(Sequence) :- hmm(Sequence, start).
 % State = insert(X), match(X), delete(X), start, end
 % X = 0,1,..,|Sequence|
hmm([], end ).
hmm(Sequence, State) :- State ¥== end,
       msw(move_from(State), NextState),
       msw(emit_at(State), Symbol),
       (Symbol = epsilon
      ->  hmm(Sequence, NextState)
       ;  Sequence = [Symbol|TailSeq],
           hmm(TailSeq, NextState)
       ).
amino_acids(['A','C','D','E','F','G','H','I','K','L','M',
            'N','P','Q', 'R','S','T','V','W','X','Y']).
```

```
observe( ['H','L','K','I','A','N','
R','K','D','K','H','H','N','K','E','
F','G','G','H','H','L','A'] ).
observe( ['H','L','K','A','T','H',
'R','K','D','Q','H','H','N','R','E',
'F','G','G','H','H','L','A'] ).
….
```

```
msw(emit_at(match(3)),Symbol)
 ➔ Symbol = any amino acid
```

VB makes a differece

```
HLKIANRKDK----HHNKEFGGHHLA
HLKATHRKDQ----HHNREFGGHHLA
VLKFANRKSK----HHNKEMGAHHLA
HKKGAT---------------PVNVS
HKKGATATG-----------NPKHVC
QFKVAAAVGK----HQDASRGVHHID
SFKGQGAVSK----HQDPEWGVHHID
SFKGQGAVSV----PQAPAWGINHID
HFKSQAEVNK----HDRPEWGLNQID
HFRSQAEVNQRQFNHHRPQWSFNQIG
SFNVVKGASK----RENGGMGAEPVD
KFKKVDGLGK----KEHPALGVH---
KFMVGGKDGK----NRKDAHAHRKVE
KYKVPEKDGK----KRTNAHSHRKVE
RYKIPESDGK----KRTNSHRHRKVE
RYKIASMDGK----KRYAEHKHKKLE
```

```
HLK---I-A--NRKDKHH-N-K-EFG---G-HH-LA-
HLK-----AT-HRKDQHH-N-R-EFG---G-HH-LA-
VLK---F-A--NRKSKHH-N-K-EMG---A-HH-LA-
HKK---G-A--T---------P-------V-NV-S--
HKKG----ATAT--G----N-P-------K-HV-C--
QFK---VAAA-VGKHQD--ASR---G---V-HHID--
SFKGQG--AVSK--HQD----P-EWG---V-HHID--
SFKGQG--AVSV--PQA----P-AWG---I-NHID--
HFK---SQAE-VNKH----D-RPEWG---L-NQID--
HFR---SQAE-VNQRQFNHH-RPQWS---F-NQIG--
SFN---V-V--K--G-A--SKR-ENGGMGAEPV-D--
KFK-----K--VDGLGK--KEHPALG---VH------
KFM---V-G---GKDGK--N-RKD-A---H-AHRKVE
KYK---V-PE-K--DGK--K-R-T-N---AHSHRKVE
RYK---I-PES---DGK--K-R-T-N---SHRHRKVE
RYK---I-AS-M--DGK--K-R-Y-A---EHKHKKLE
```

Graph size: 148558
# iterations: 51
log likelihood: -544.33
Total learning time: 1.75 sec

- EM ➔ Viterbi

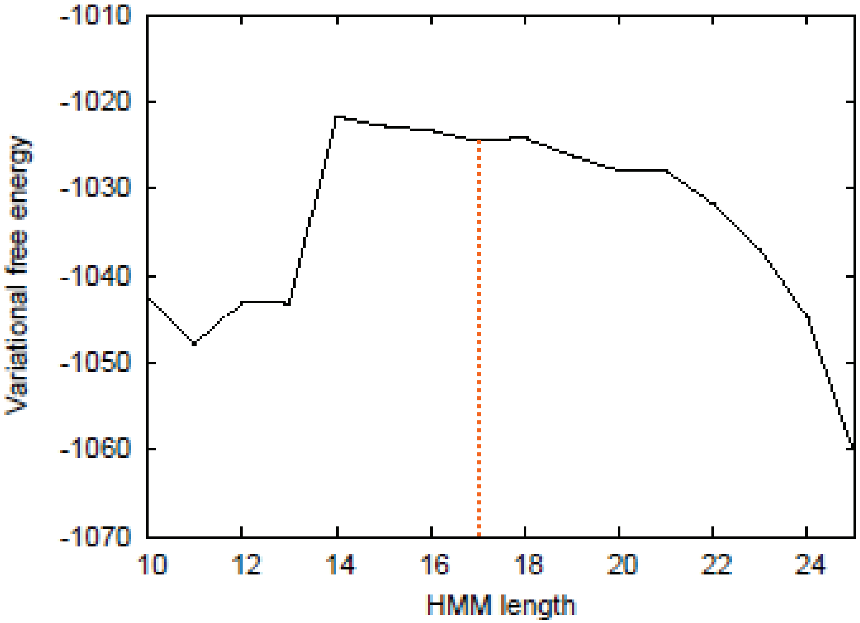
```
HLKIANRKDK----HHNKEFGGHHLA
HLKATHRKDQ----HHNREFGGHHLA
VLKFANRKSK----HHNKEMGAHHLA
HKKGATPVN---------------VS
HKKGATATG-------NP---K-HVC
QFKVAAAVGK----HQDASRGVHHID
SFKGQGAVSK----HQDPEWGVHHID
SFKGQGAVSV----PQAPAWGINHID
HFKSQAEVNK----HDRPEWGLNQID
HFRSQAEVNQRQFNHHRPQWSFNQIG
SFNVVKGASK----RENGGMGAEPVD
KFKKVDGLGK----KEHPALGVH---
KFMVGGKDGK----NRKDAHAHRKVE
KYKVPEKDGK----KRTNAHSHRKVE
RYKIPESDGK----KRTNSHRHRKVE
RYKIASMDGK----KRYAEHKHKKLE
```

Graph size: 148558
# iterations (VB-EM): 47
# iterations (MAP-EM): 17
log likelihood: -534.56
Total learning time: 2.1 sec

- VB ➔ MAP ➔Viterbi

Model selection
--- determining the length of P-HMM

- We determine the length L of the p-HMM so that it maximized the log marginal likelihood of input data ➔ Choose L that maximizes the variational free energy



True L = 17

Conclusion

- We argued the universality of PPC (propositionalized probability computation).
- We proposed to combine the PPC in PRISM with variational Bayes and derived the graphical VB-EM algorithm for PRISM.
- VB-EM is implemented in PRISM 1.11
- It will make Bayesian inference much easier such as the one for profile-HMMs.