WALLENBERG AI, AUTONOMOUS SYSTEMS AND SOFTWARE PROGRAM

digital futures

KTH VETENSKAP OCH KONST

Automated Inference Optimizations in the Probabilistic Programming Language Miking CorePPL

Gizem Caylak
Collaborators: David Broman,Daniel Lundén, Emma Granqvist, Fredrik Ronquist, Viktor Senderov

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation.

Introduction: Probabilistic Programming Languages

- Probabilistic programming languages (PPLs) provide tools to write probabilistic models and run statistical inference over these models.

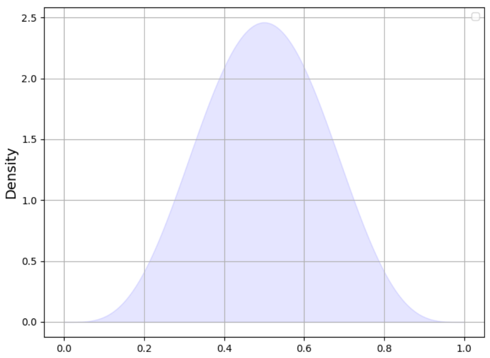
Prior knowledge P(H)

%50

H

%50

T



Probabilistic Model

Observations
P(D|H)

83

H

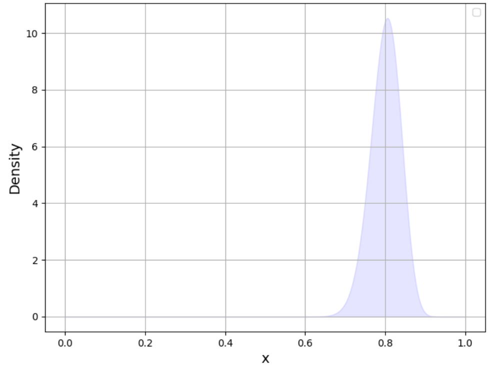
17

T

Inference Algorithm

$$P(H|D) = \frac{P(D|H)P(\mathrm{H})}{P(D)}$$

Posterior distribution P(H|D)



Posterior Distributions

%80

H

%20

T

What is the probability of heads after observing 83 heads

Introduction: Probabilistic Programming Languages

Monte Carlo Methods

Stan

Anglican

Birch

WPPL

WebPPL

Miking CorePPL

Variational Inference

Edward

Pyro

Exact Inference

IBAL

ProbZelus

PERPL

Introduction: Miking CorePPL

- Statically typed **universal** probabilistic programming language
- Calculate posterior distributions and the marginalized log likelihood.

.

sample

```
let a = assume (Beta 5. 5.) in
observe true (Bernoulli a)
```

update the likelihood

$$P(a|D) = \frac{P(D|a)P(\mathrm{a})}{P(D)}$$

Posterior

Marginalized likelihood

$$P(D) = \int P(D|a)P(\mathrm{a})$$

Asymptotically converges to true posterior with samples and their associated likelihood

Introduction: Problem definition

- Trade-off between approximate vs exact methods

$$P(D) = \int P(D|\theta)P(\theta)$$

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$

Approximate Inference

Exact Inference

Expressivess

Accuracy

Dynamic Adaptation

Computational Efficiency

Introduction: Problem definition

- By utilizing exact inference techniques, we can improve the efficiency of approximate methods.

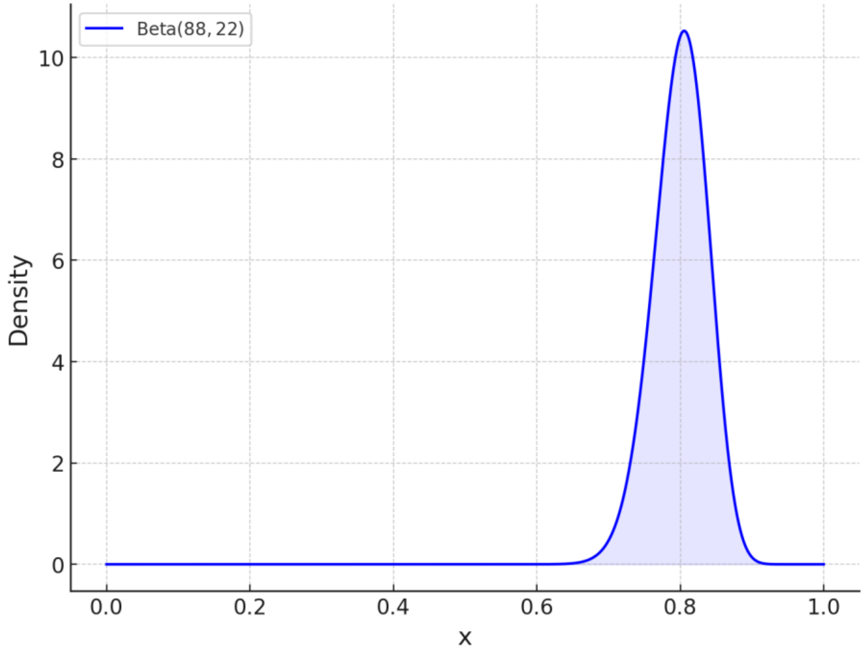
$P(\theta) \sim Beta(5,5)$
$P(D|\theta) \sim Bernoulli(\theta)$
$P(\theta|D) \sim ?$

1,000,000 particles

$P(\theta|D) \sim Beta(88,22)$

1 particle

Semantically equivalent models



Both models converge to the same posterior with different execution times

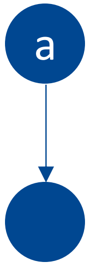
Approaches

When | Compile-time approaches | Runtime approaches

What

- Topology of the model
- Conjugate prior relations between random variables

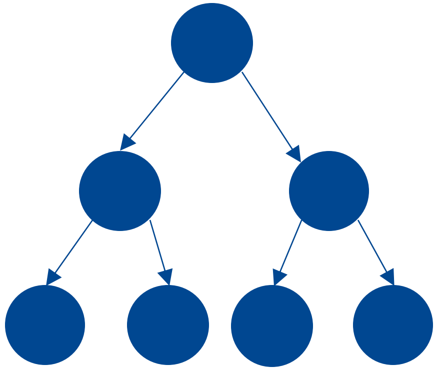
How



Beta(5., 5.)

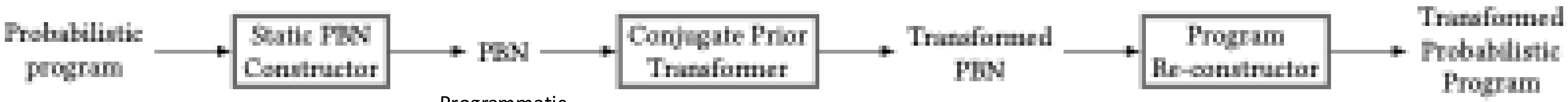
Bernoulli(a)

Delayed sampling by Murray et al. (2018): An approach utilizing conjugate prior relations at runtime



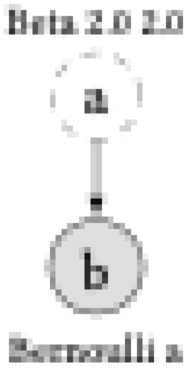
Belief propagation by Pearl (1986) gives exact solution for models forming tree structures.

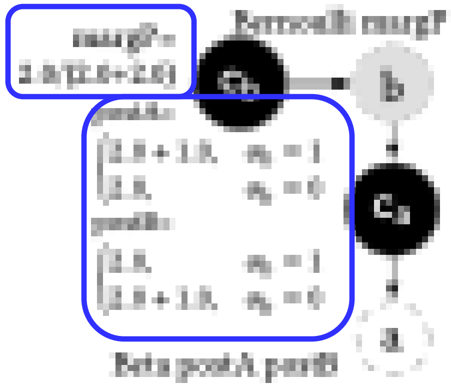
Statically Delayed Sampling Algorithm



Programmatic Bayesian Network

```
let a = assume
  (Beta 2.0 2.0) in
observe b (Bernoulli a)
```






```
let margP=divf 2. (addf 2. 2.) in
observe b (Bernoulli margP);
let postA= if b then addf 2. 1.
     else 2. in
let postB= if b then 2.
     else addf 2. 1. in
let a=assume (Beta postA postB) in ()
```

Marginalize

$$P(b) = \int_a P(b|a)P(\mathrm{a})\, da$$

$$p(\tilde{x} = 1) = \frac{\alpha'}{\alpha' + \beta'}$$

Condition

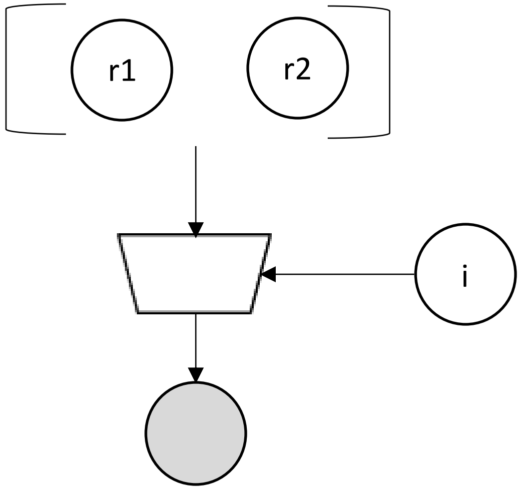
$$P(a|b) = \frac{P(b|a)P(\mathrm{a})}{P(b)}$$

$$\alpha + \sum_{i=1}^{n} x_i,\ \beta + n - \sum_{i=1}^{n} x_i$$

Programmatic Bayesian Network

```
let r1 = assume (Gaussian 0. 1.) in
let r2 = assume (Gaussian 1. 2.) in
let lst = [r1, r2] in
let i = assume (Categorical [0.6, 0.4]) in
let mu = get lst i in
observe 0.9 (Gaussian mu 1.)
```

Programmatic Bayesian Network

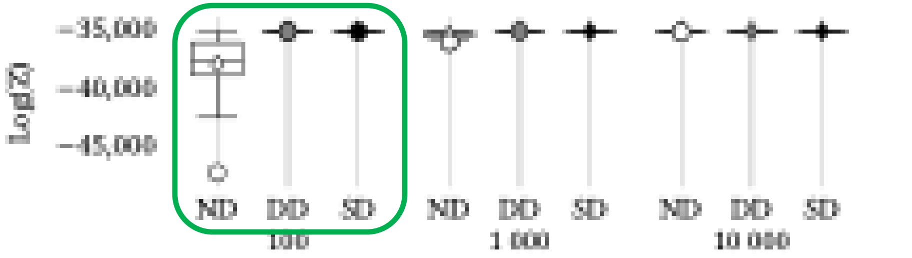


Reconstruct

- **Topologically sort** the graph and then reconstruct each node

Results

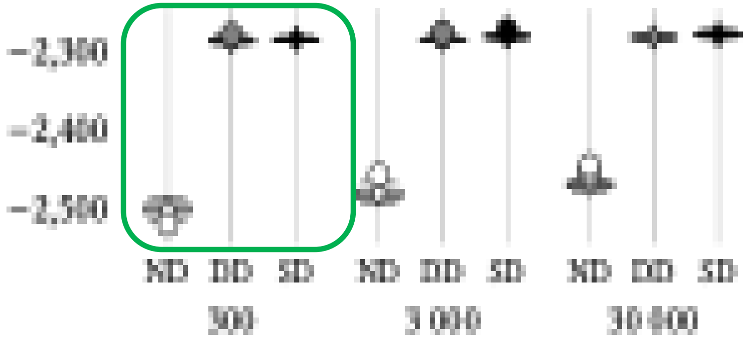
Bayesian linear regression

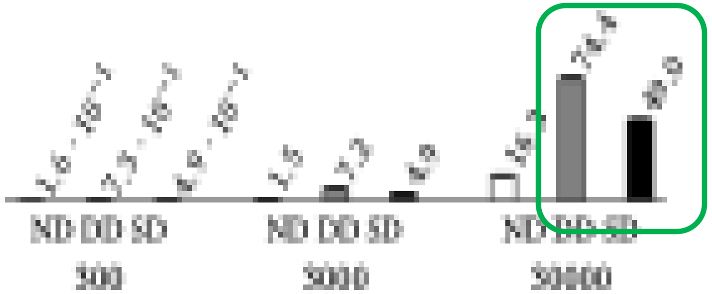


(a) Log normalizing constants.

(b) Execution times

Latent Dirichlet allocation

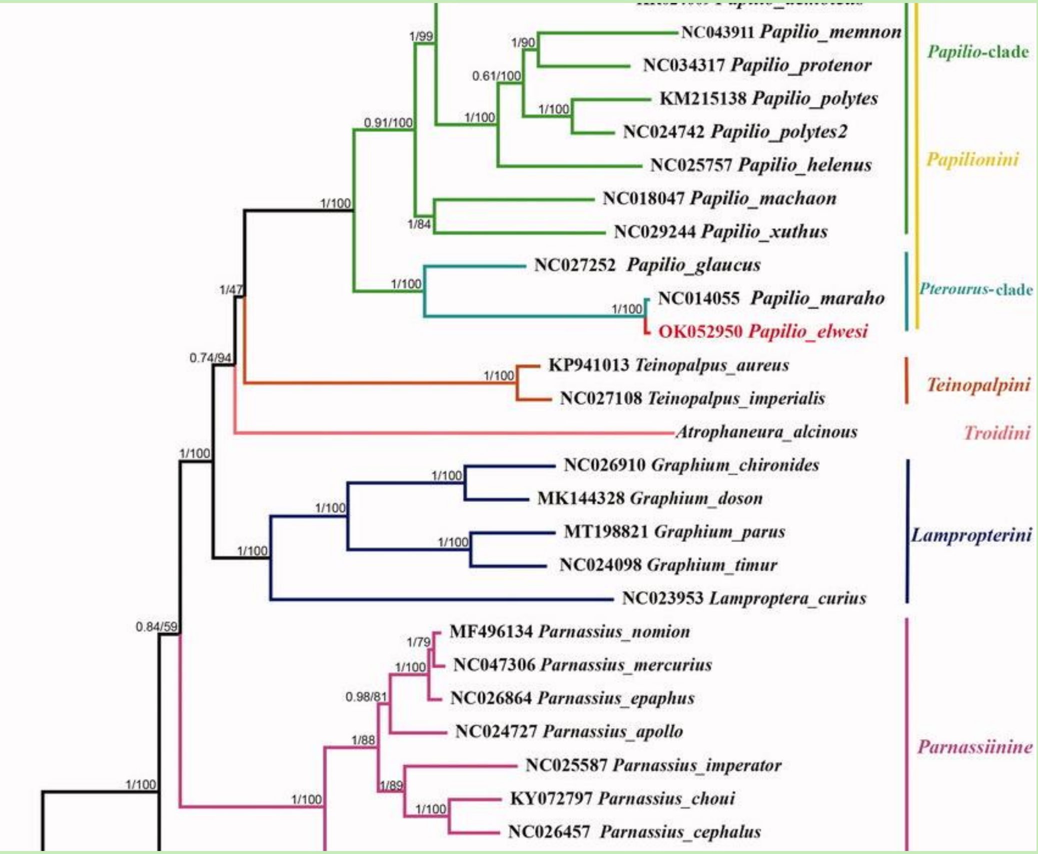




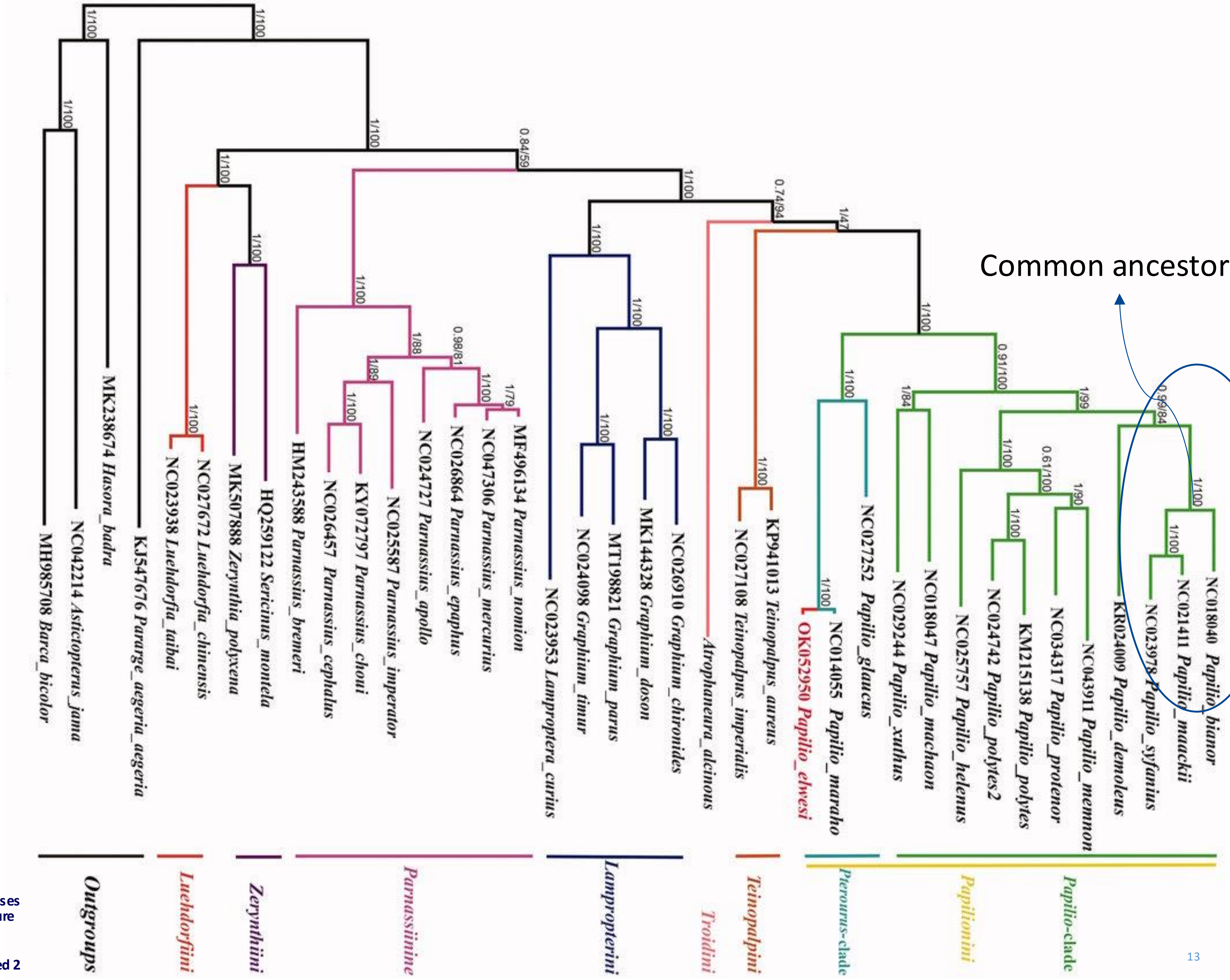
Automating the Forward Pass of Belief Propagation

- Tree inference is a fundamental problem in evolutionary biology
- Given the current species, what is the most likelihood evolutionary tree or distribution of trees?
- Can we answer this question efficiently?

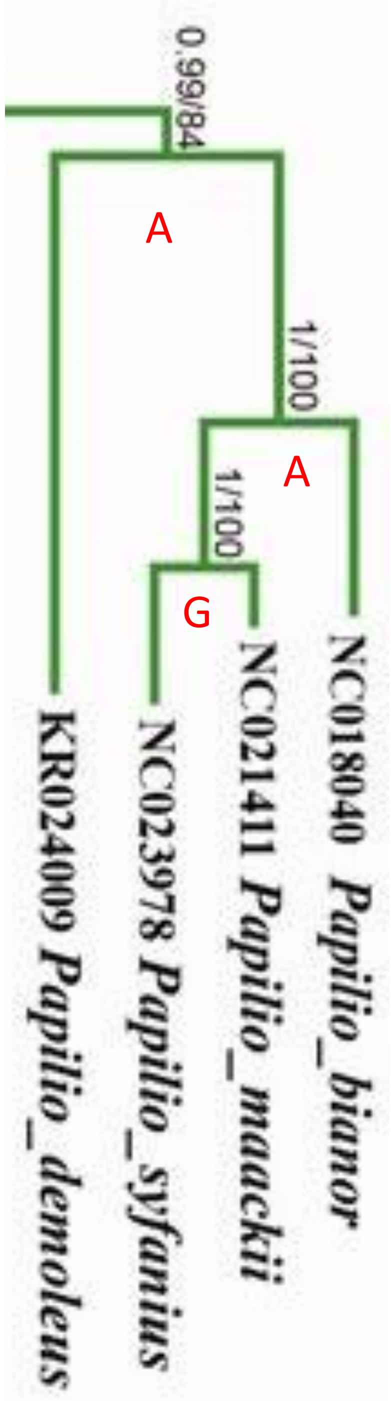
Complete mitochondrial genome of Papilio elwesi and its phylogenetic analyses with other swallowtail butterflies (Lepidoptera, Papilionidae) - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Phylogenetic-tree-using-Bayesian-inference-BI-and-maximum-likelihood-ML-analysis_fig1_359347159 [accessed 2 Dec 2024]

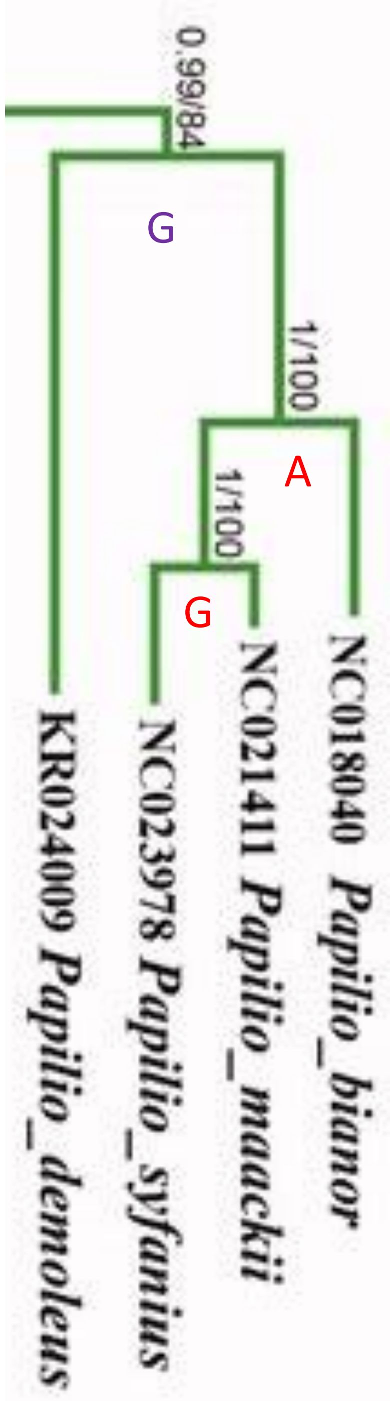


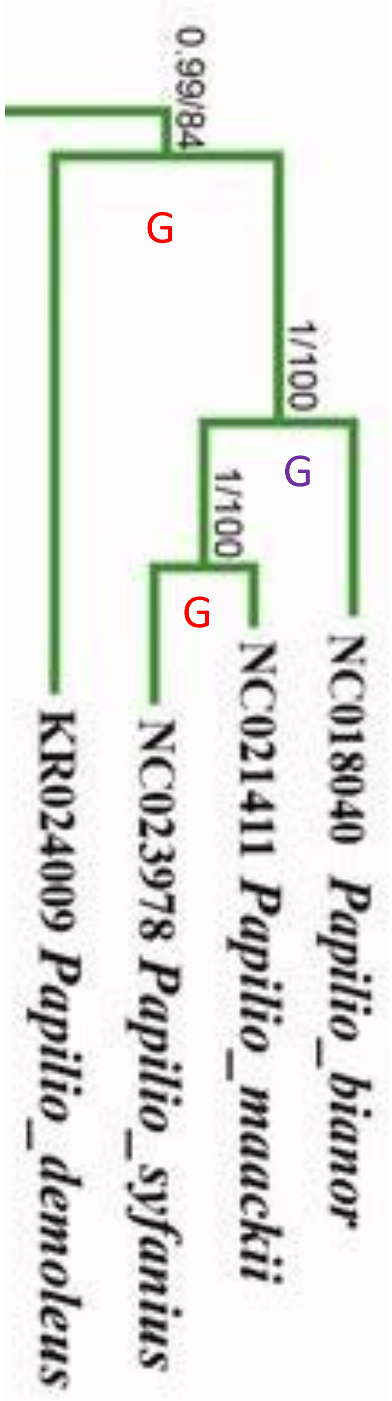
The purpose is to calculate the likelihood of this tree



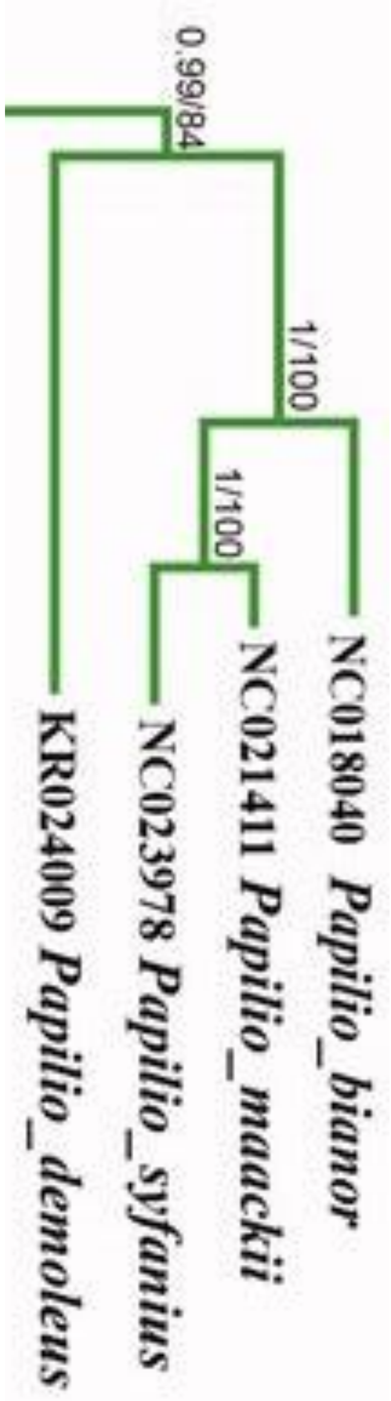
Complete mitochondrial genome of Papilio elwesi and its phylogenetic analyses with other swallowtail butterflies (Lepidoptera, Papilionidae) - Scientific Figure on ResearchGate. Available from: https://www.researchgate.net/figure/Phylogenetic-tree-using-Bayesian-inference-BI-and-maximum-likelihood-ML-analysis_fig1_359347159 [accessed 2 Dec 2024]


0.99/84
A
1/100
A
1/100
G
KR024009 Papilio_demoleus
NC023978 Papilio_syfanius
NC021411 Papilio_maackii
NC018040 Papilio_bianor


0.99/84
G
1/100
A
1/100
G
KR024009 Papilio_demoleus
NC023978 Papilio_syfanius
NC021411 Papilio_maackii
NC018040 Papilio_bianor


0.99/84
G
1/100
G
1/100
G
KR024009 Papilio_demoleus
NC023978 Papilio_syfanius
NC021411 Papilio_maackii
NC018040 Papilio_bianor

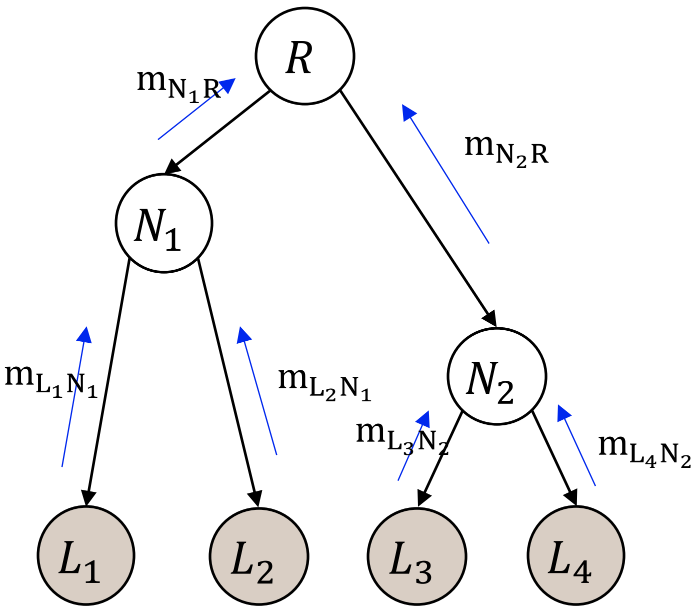
…



- The important information is the topology of the tree not specific information about internal nodes such as their genetic code
- Can we summarize this information:

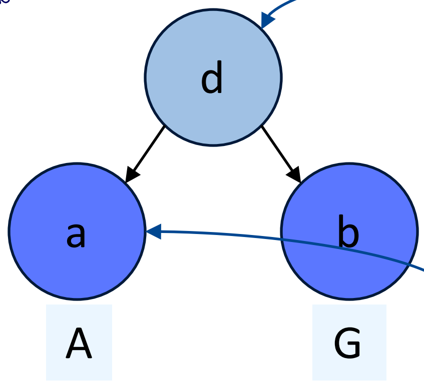
For all the possible values internal nodes can take, what is the likelihood of this tree?

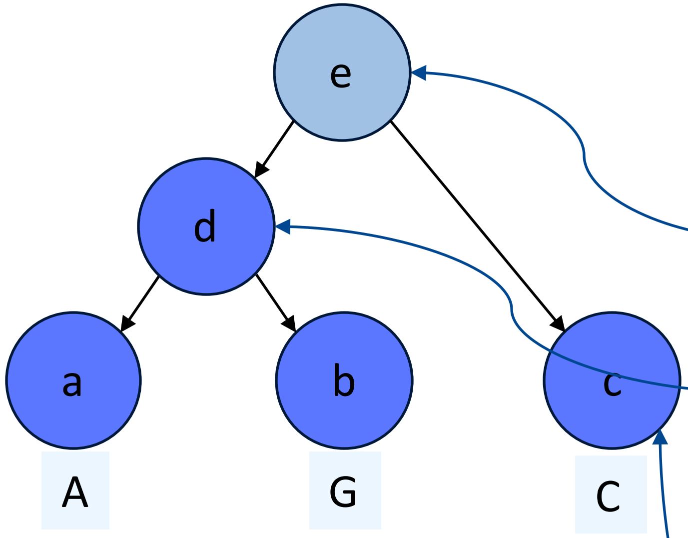
Belief Propagation on Tree Inference Problem



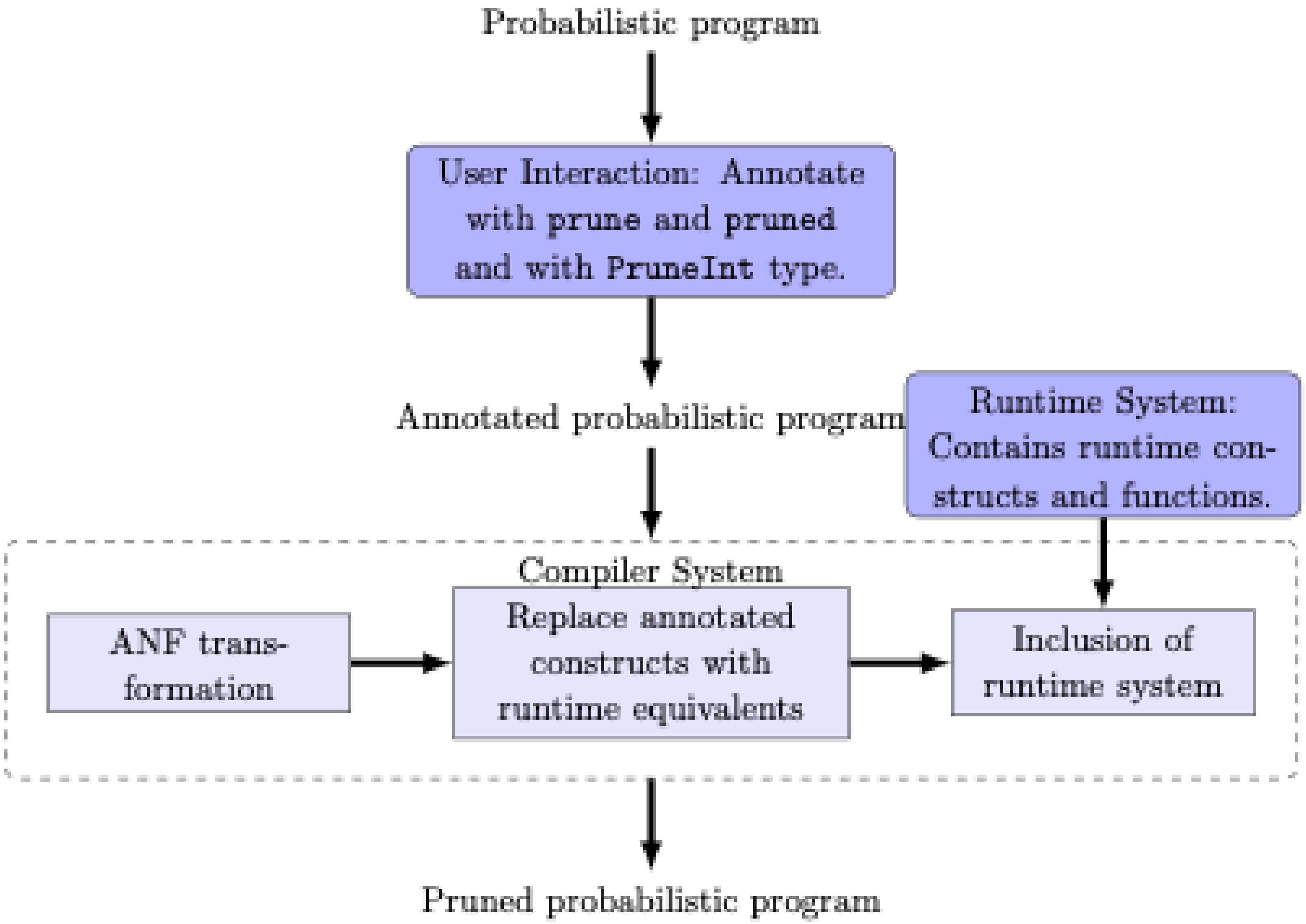
Marginalize out the internal nodes

Propagate the messages to the root to calculate the likelihood of observations given the tree topology






```
let d_seq = prune (Categorical [0.25,0.25,0.25,0.25]) in

let p1 = ctmc (pruned d_seq) q (subf age (getAge a)) in
observe (getSeq a) (Categorical p1);

let p2 = ctmc (pruned d_seq) q (subf age (getAge b)) in

observe (getSeq b) (Categorical p2);

let e_seq = prune (Categorical [0.25,0.25,0.25,0.25]) in

let p3 = ctmc (pruned e_seq) q (subf age (getAge d)) in

observe (pruned (getNodeSeq d)) (Categorical p3);

cancel (observe (pruned(getNodeSeq d))
    (Categorical [0.25,0.25,0.25,0.25]));

let p4 =  ctmc (pruned e_seq) q (subf age (getAge c)) in

observe (getSeq c) (Categorical p4)
```

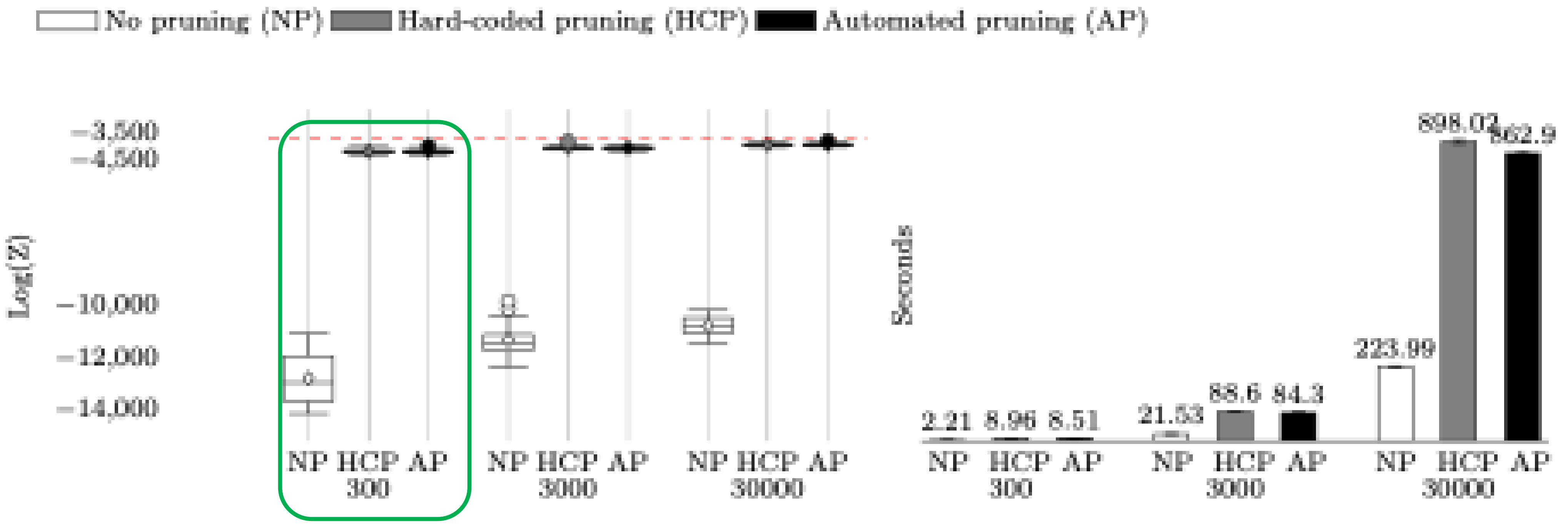

Probabilistic program
User Interaction: Annotate with prune and pruned and with PruneInt type.
Annotated probabilistic program
Runtime System: Contains runtime constructs and functions.
Compiler System
ANF transformation
Replace annotated constructs with runtime equivalents
Inclusion of runtime system
Pruned probabilistic program

Results

Without pruning, GTR model is encoded in 110 lines

The number of lines that need to be changed for
Automated pruning: 18
Hard-coded pruning: 107

Generalized time reversible model– Primates data



(a) Log normalizing constants. (b) Execution times

Conclusion