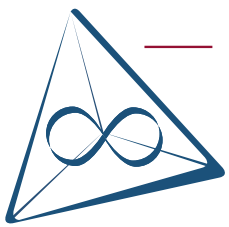
Gibbs sampling for parsimonious Markov models with latent variables

Ralf Eggeling[1], Pierre-Yves Bourguignon[2], André Gohr[1], Ivo Grosse[1]

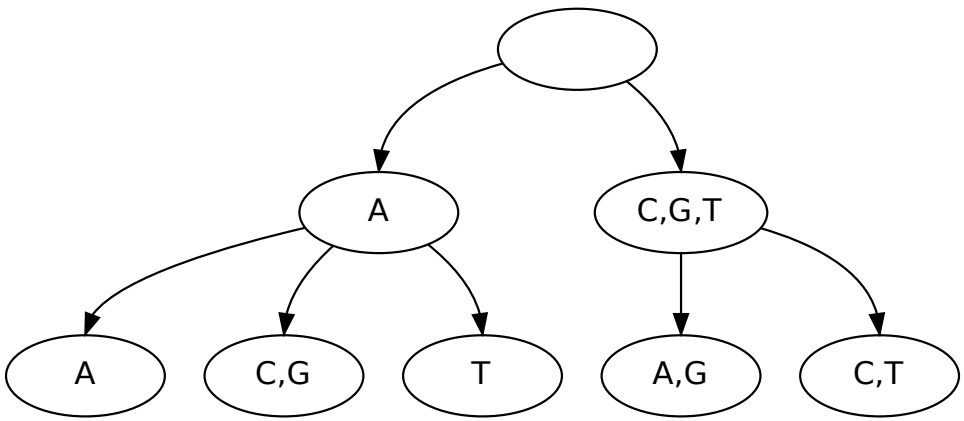
[1] Martin Luther University Halle-Wittenberg
[2] Max Planck Institute for Mathematics in the Sciences



Max Planck Institute for Mathematics in the Sciences

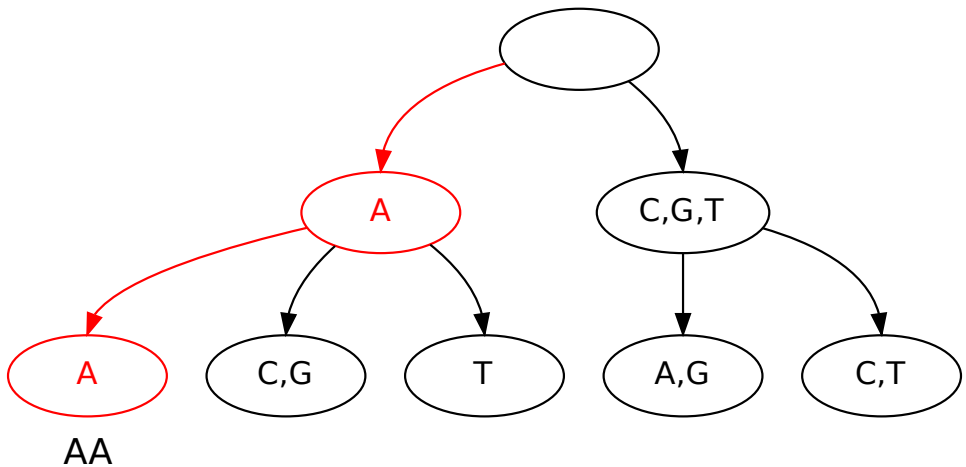
Premise 1: Parsimonious Markov models

- proposed by Bourguignon (2008)
- generalize variable order Markov models
- use parsimonious context trees (PCTs)
- here: inhomogeneous models
  $\rightarrow$ seperate PCTs for each random variable



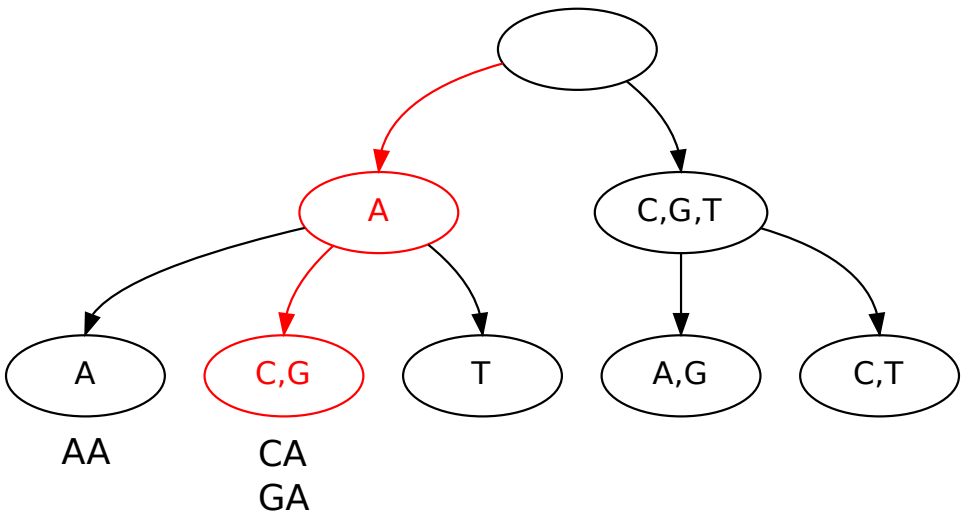
Premise 1: Parsimonious Markov models

- proposed by Bourguignon (2008)
- generalize variable order Markov models
- use parsimonious context trees (PCTs)
- here: inhomogeneous models
  $\rightarrow$ seperate PCTs for each random variable



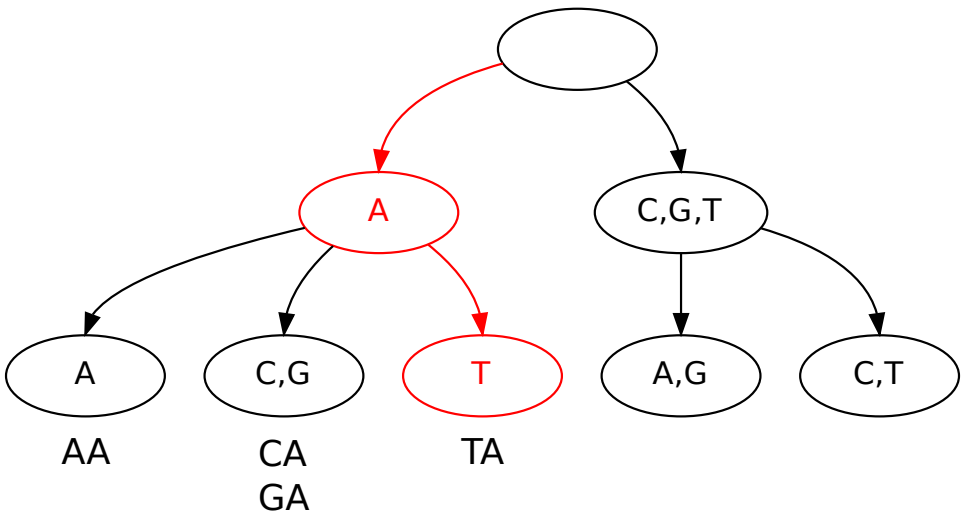
Premise 1: Parsimonious Markov models

- proposed by Bourguignon (2008)
- generalize variable order Markov models
- use parsimonious context trees (PCTs)
- here: inhomogeneous models
  $\rightarrow$ seperate PCTs for each random variable



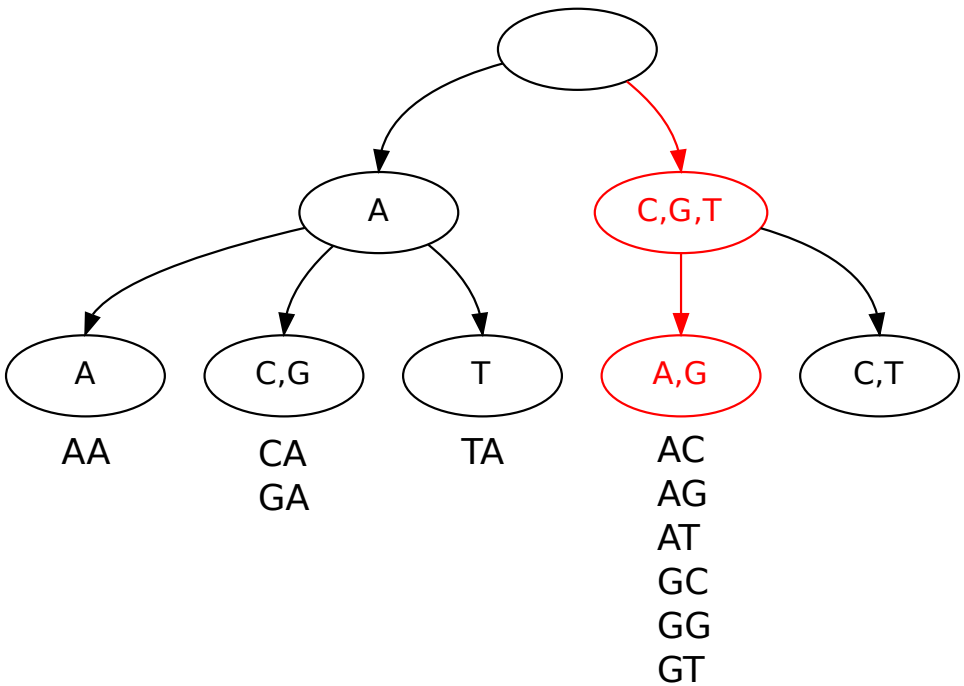
Premise 1: Parsimonious Markov models

- proposed by Bourguignon (2008)
- generalize variable order Markov models
- use parsimonious context trees (PCTs)
- here: inhomogeneous models
  $\rightarrow$ seperate PCTs for each random variable



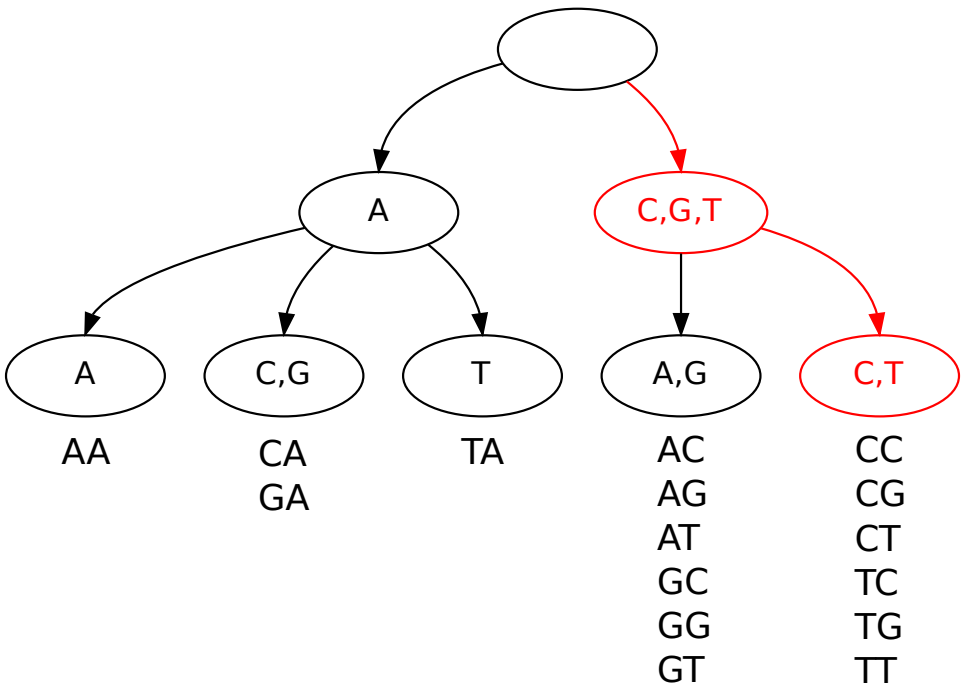
Premise 1: Parsimonious Markov models

- proposed by Bourguignon (2008)
- generalize variable order Markov models
- use parsimonious context trees (PCTs)
- here: inhomogeneous models
  $\rightarrow$ seperate PCTs for each random variable



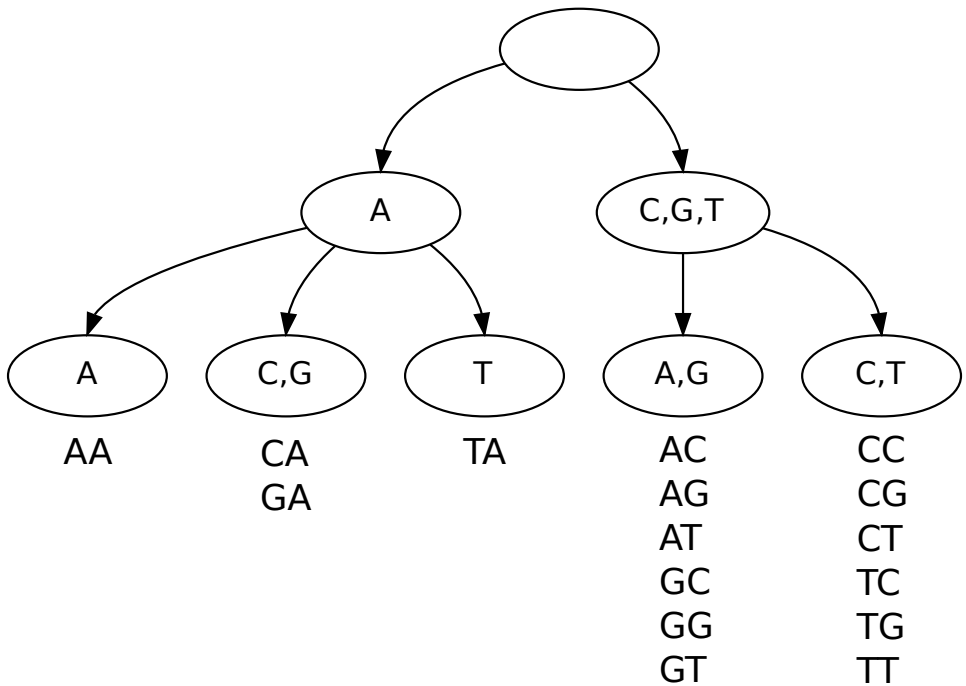
Premise 1: Parsimonious Markov models

- proposed by Bourguignon (2008)
- generalize variable order Markov models
- use parsimonious context trees (PCTs)
- here: inhomogeneous models
  $\rightarrow$ seperate PCTs for each random variable



Premise 1: Parsimonious Markov models

- proposed by Bourguignon (2008)
- generalize variable order Markov models
- use parsimonious context trees (PCTs)
- here: inhomogeneous models
  $\rightarrow$ seperate PCTs for each random variable



Premise 2: Latent variable models

- many practical applications: latent variables, unobserved/missing data

Premise 2: Latent variable models

- many practical applications: latent variables, unobserved/missing data
- examples:
  - Naive Bayes
  - Hidden Markov models
  - **Mixture models**

Premise 2: Latent variable models

- many practical applications: latent variables, unobserved/missing data
- examples:
  - Naive Bayes
  - Hidden Markov models
  - **Mixture models**

Mixture models

- model assumption: data point i generated from one out of C component models
  $\rightarrow$ latent variable $u_i \in \{1, \ldots, C\}$
- analytical learning infeasible
- approximative algorithms:
  - EM algorithm
  - Gibbs sampling

Premise 3: Bayesian prediction

Classical prediction

- estimate optimal parameters $\hat{\Theta}(X)$ from training data X

$$P_{\text{classic}}(Y|X) = P(Y|\hat{\Theta}(X))$$

Premise 3: Bayesian prediction

Classical prediction

- estimate optimal parameters $\hat{\Theta}(X)$ from training data X

$$P_{\text{classic}}(Y|X) = P(Y|\hat{\Theta}(X))$$

Bayesian prediction

- do not estimate optimal parameters

$$P_{\text{Bayes}}(Y|X) = \int P(Y|\Theta)P(\Theta|X)d\Theta$$

Premise 3: Bayesian prediction

Classical prediction

- estimate optimal parameters $\hat{\Theta}(X)$ from training data X

$$P_{\text{classic}}(Y|X) = P(Y|\hat{\Theta}(X))$$

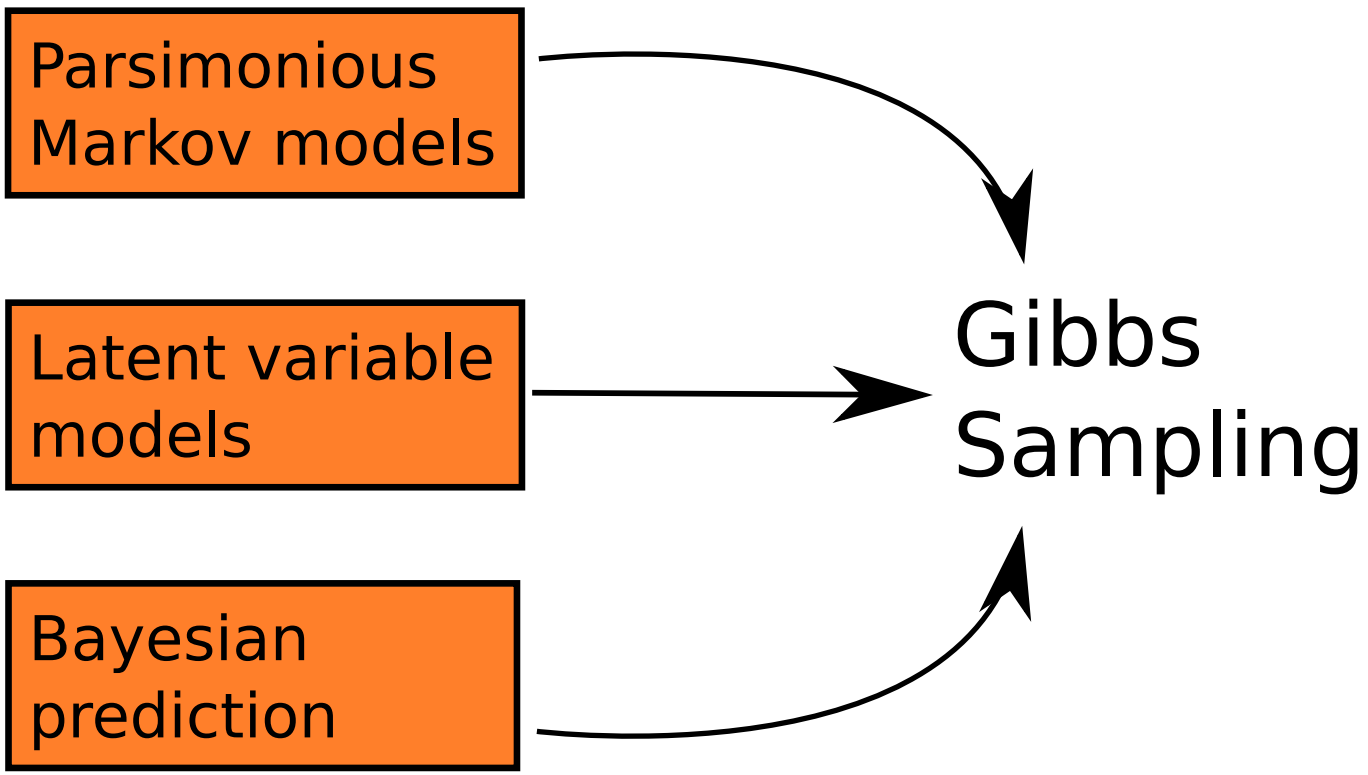
Bayesian prediction

- do not estimate optimal parameters

$$P_{\text{Bayes}}(Y|X) = \int P(Y|\Theta)P(\Theta|X)d\Theta$$

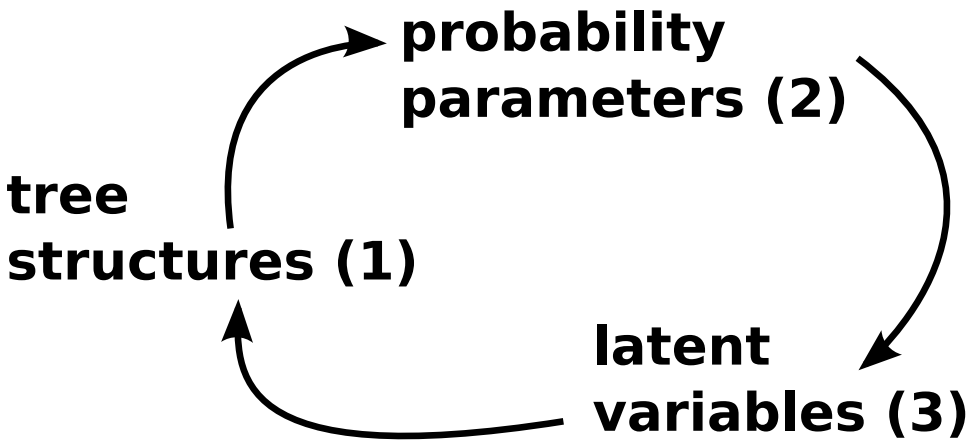
- classical prediction approximates Bayesian prediction
- posterior concentrated around $\hat{\Theta} \rightarrow$ good approximation
- posterior diverse $\rightarrow$ bad approximation

Putting premises together



Gibbs sampling algorithm

- goal: sample from posterior distribution
- Gibbs sampling: sample iteratively from conditional probability distributions of each variable/parameter



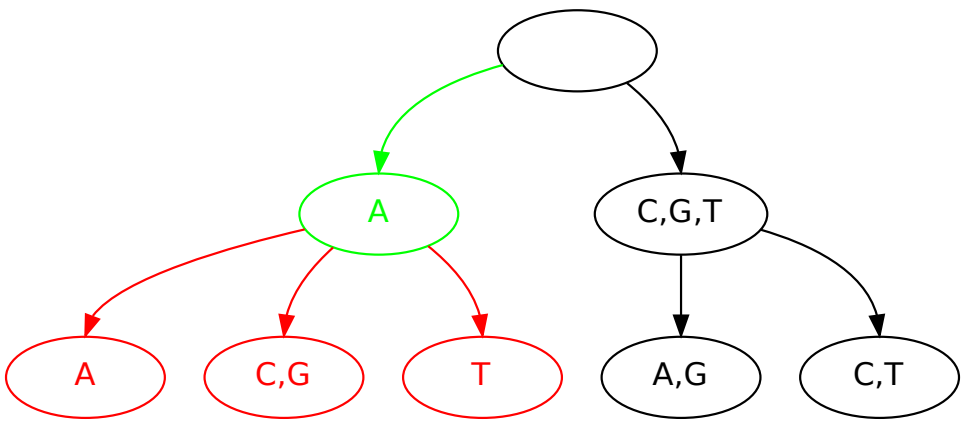
- probability parameters $\rightarrow$ simple
- latent variables $\rightarrow$ simple
- structure $\rightarrow$ difficult

Structure sampling

- probability of a PCT structure:

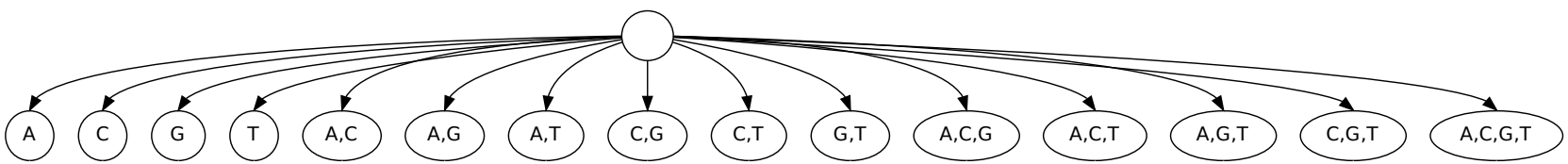
$$P(\tau|\mathbf{X}) \propto \prod_{\mathbf{w}\in\mathcal{C}_\tau} \kappa \frac{\mathcal{B}(\vec{N}_\mathbf{w} + \vec{\alpha}_\mathbf{w})}{\mathcal{B}(\vec{\alpha}_\mathbf{w})}$$

- product of leaf scores
- observation: subtree (red) probability independent of sibling subtree(s) given subtree root (green)



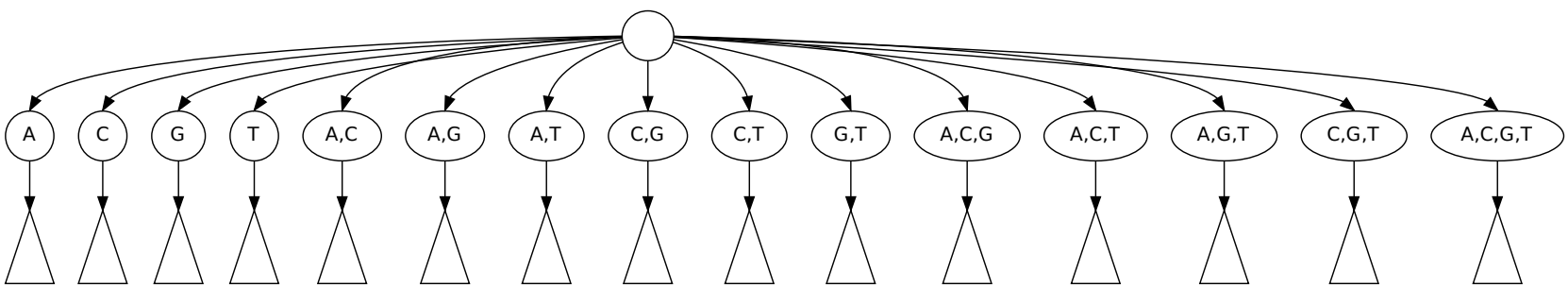
Structure sampling

- dynamic programming on extended PCT
  $\rightarrow$ sibling nodes form $\mathcal{P}(\mathcal{A}) \setminus \{\varnothing\}$



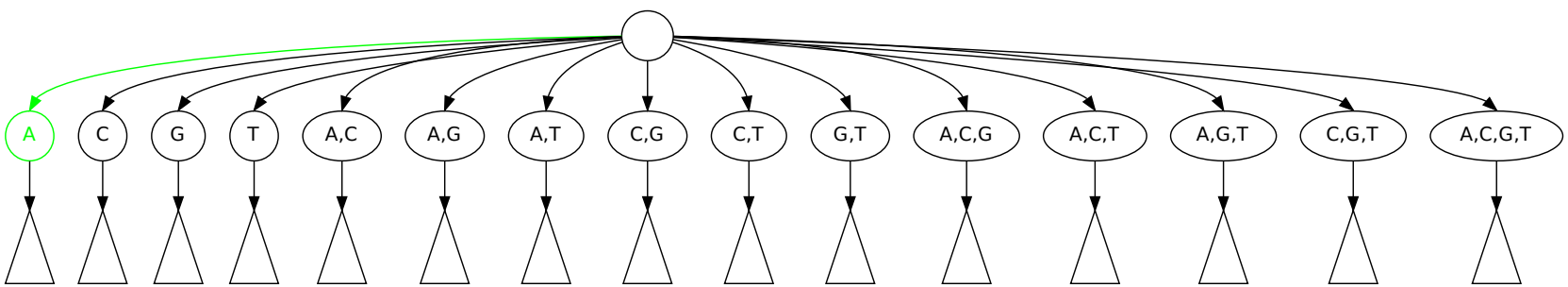
Structure sampling

- dynamic programming on extended PCT
  $\rightarrow$ sibling nodes form $\mathcal{P}(\mathcal{A}) \setminus \{\varnothing\}$
- depth identical to that of PCT



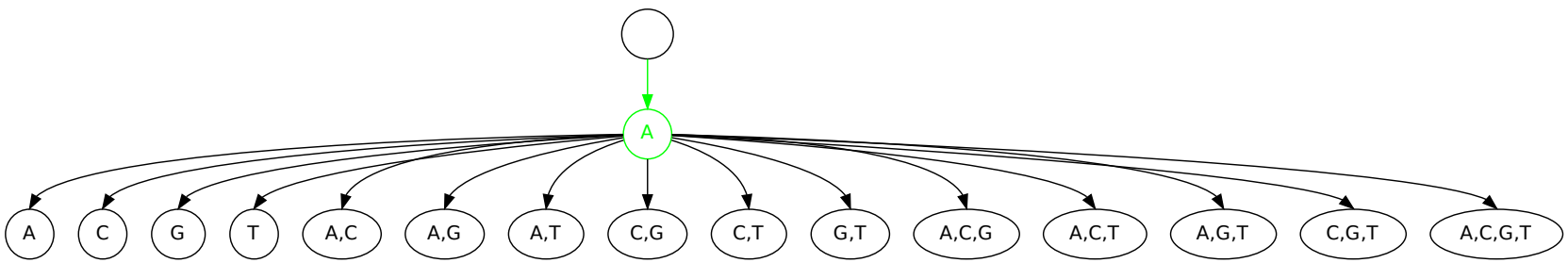
Structure sampling

- dynamic programming on extended PCT
  $\rightarrow$ sibling nodes form $\mathcal{P}(\mathcal{A}) \setminus \{\varnothing\}$
- depth identical to that of PCT
- traverse tree top-down



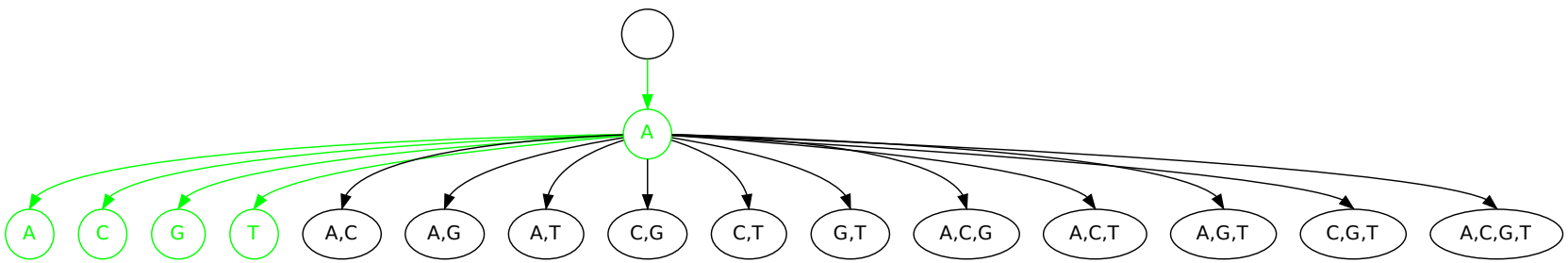
Structure sampling

- sample subtrees bottom-up
- child nodes are
  a) leaves
  b) roots of valid PCT subtrees



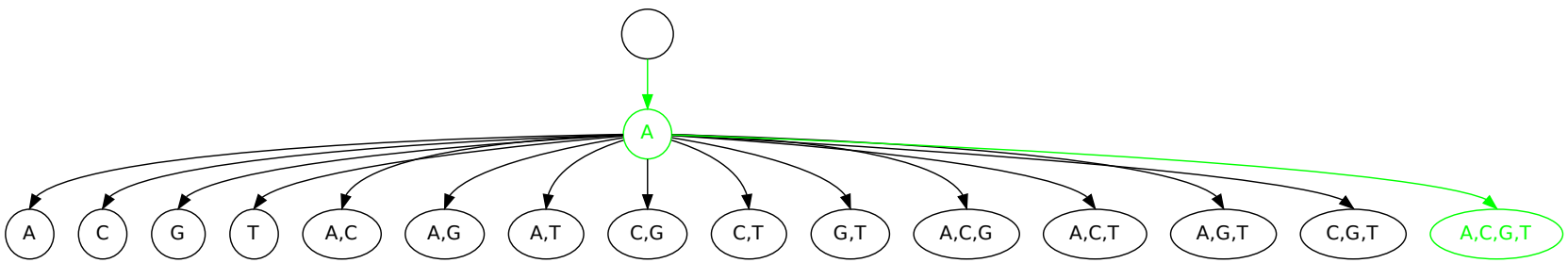
Structure sampling

- sample subtrees bottom-up
- child nodes are
  a) leaves
  b) roots of valid PCT subtrees
- compute score of all valid child combinations (15)



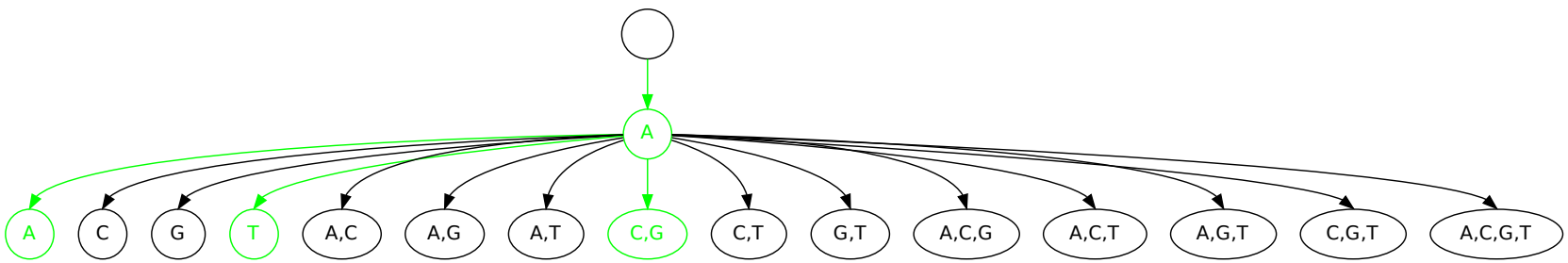
Structure sampling

- sample subtrees bottom-up
- child nodes are
  a) leaves
  b) roots of valid PCT subtrees
- compute score of all valid child combinations (15)



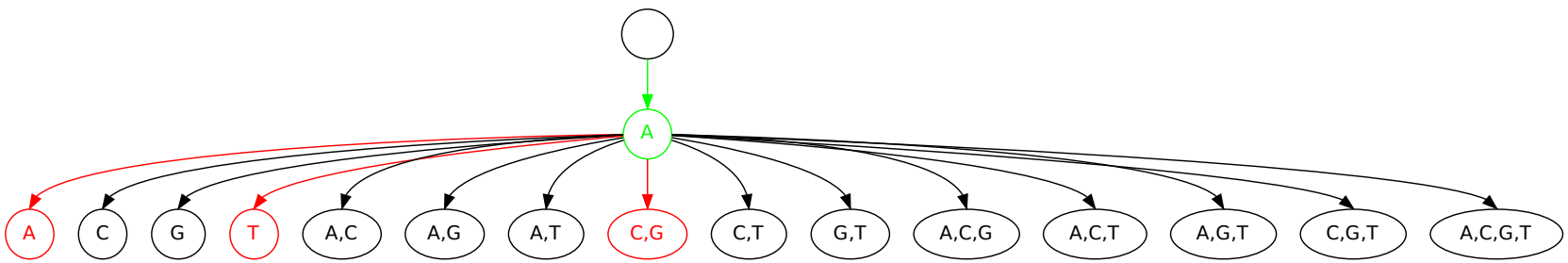
Structure sampling

- sample subtrees bottom-up
- child nodes are
  a) leaves
  b) roots of valid PCT subtrees
- compute score of all valid child combinations (15)



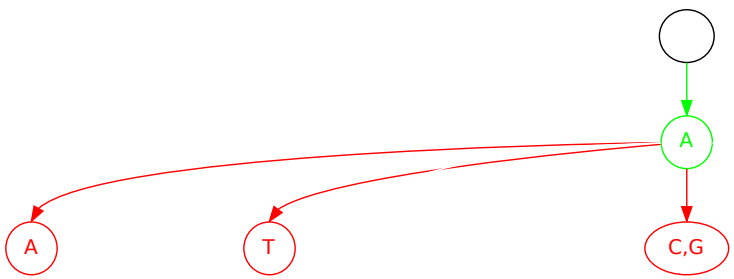
Structure sampling

- sample subtrees bottom-up
- child nodes are
  a) leaves
  b) roots of valid PCT subtrees
- compute score of all valid child combinations (15)
- sample from that distribution



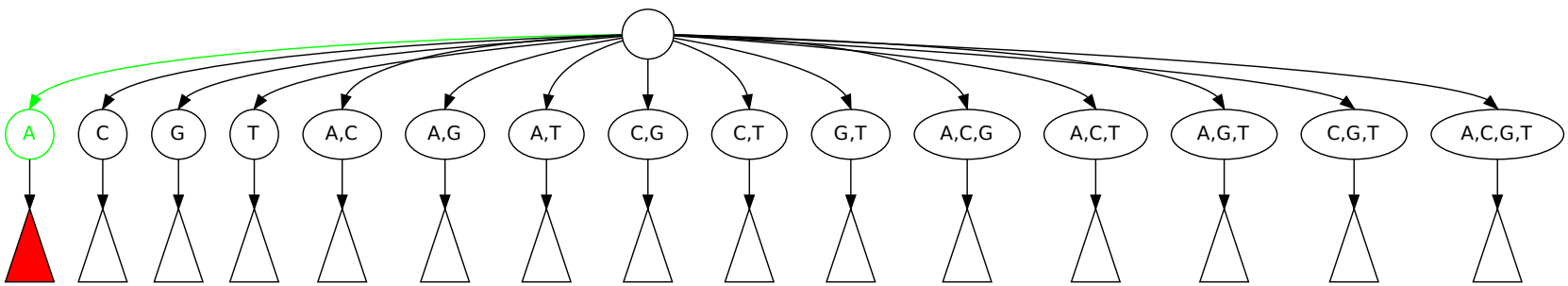
Structure sampling

- sample subtrees bottom-up
- child nodes are
  a) leaves
  b) roots of valid PCT subtrees
- compute score of all valid child combinations (15)
- sample from that distribution
- discard rest



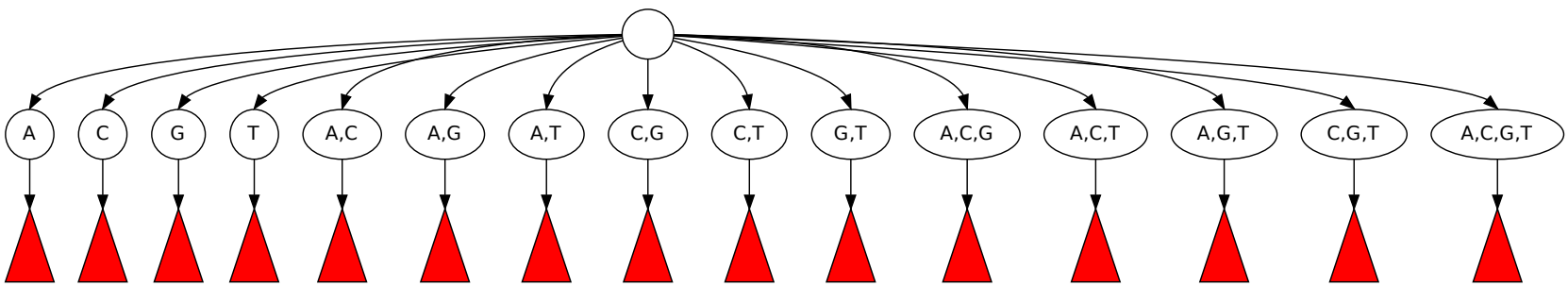
Structure sampling

- assign score of sampled children to subtree root



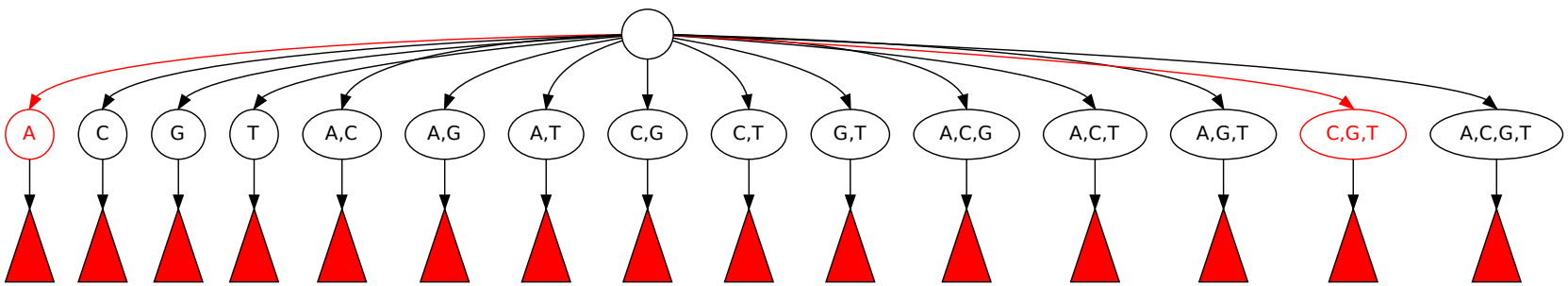
Structure sampling

- assign score of sampled children to subtree root
- repeat procedure for all siblings



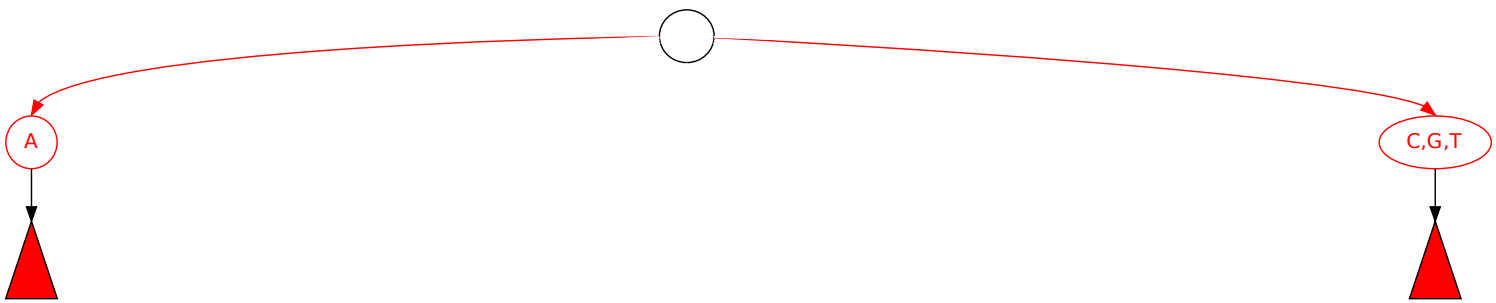
Structure sampling

- assign score of sampled children to subtree root
- repeat procedure for all siblings
- sample a valid selection



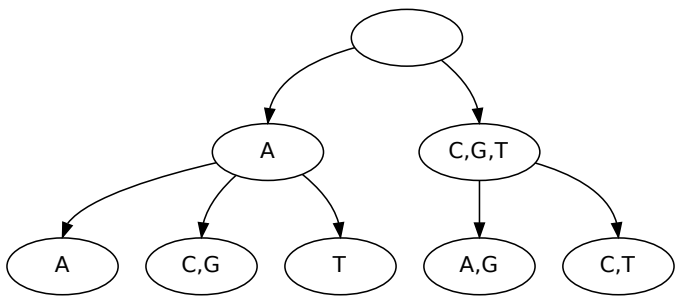
Structure sampling

- assign score of sampled children to subtree root
- repeat procedure for all siblings
- sample a valid selection
- discard rest



Structure sampling

- assign score of sampled children to subtree root
- repeat procedure for all siblings
- sample a valid selection
- discard rest



Case studies

- $C = 2$ mixture components
- each component: parsMM(2)

- two questions:

Case studies

- $C = 2$ mixture components
- each component: parsMM(2)

- two questions:
  - Does the algorithm converge?
    $\rightarrow$ see paper/poster

Case studies

- $C = 2$ mixture components
- each component: parsMM(2)

- two questions:
  - Does the algorithm converge?
    $\rightarrow$ see paper/poster
  - Does the algorithm work?
    $\rightarrow$ classify splice donor sites vs non splices sites
    $\rightarrow$ compare:
    Bayesian prediction (using Gibbs sampling) with classical prediction (using EM algorithm)

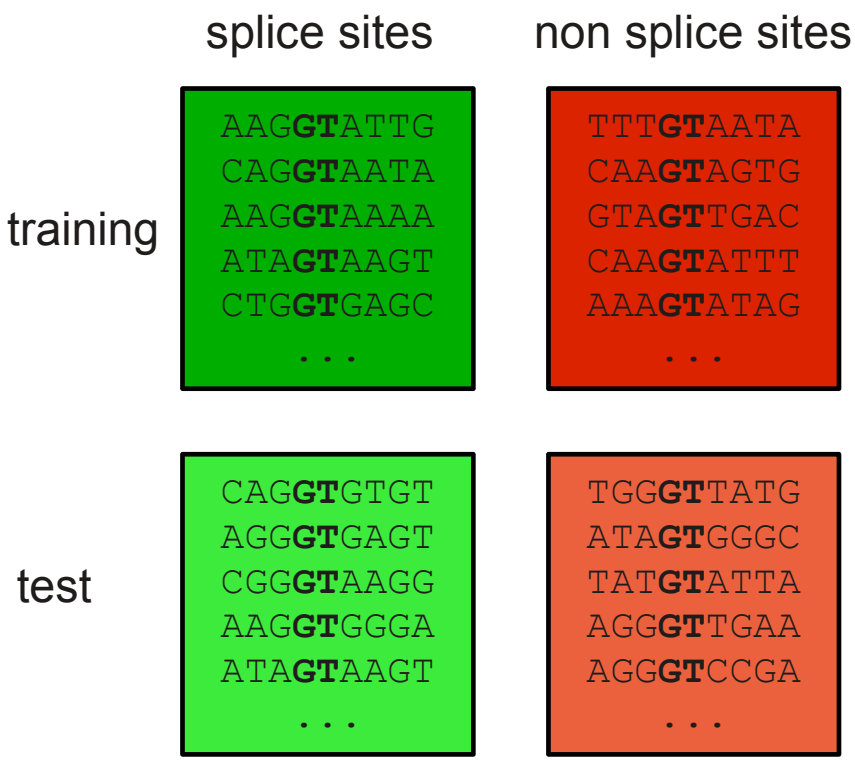
Data

Splice sites

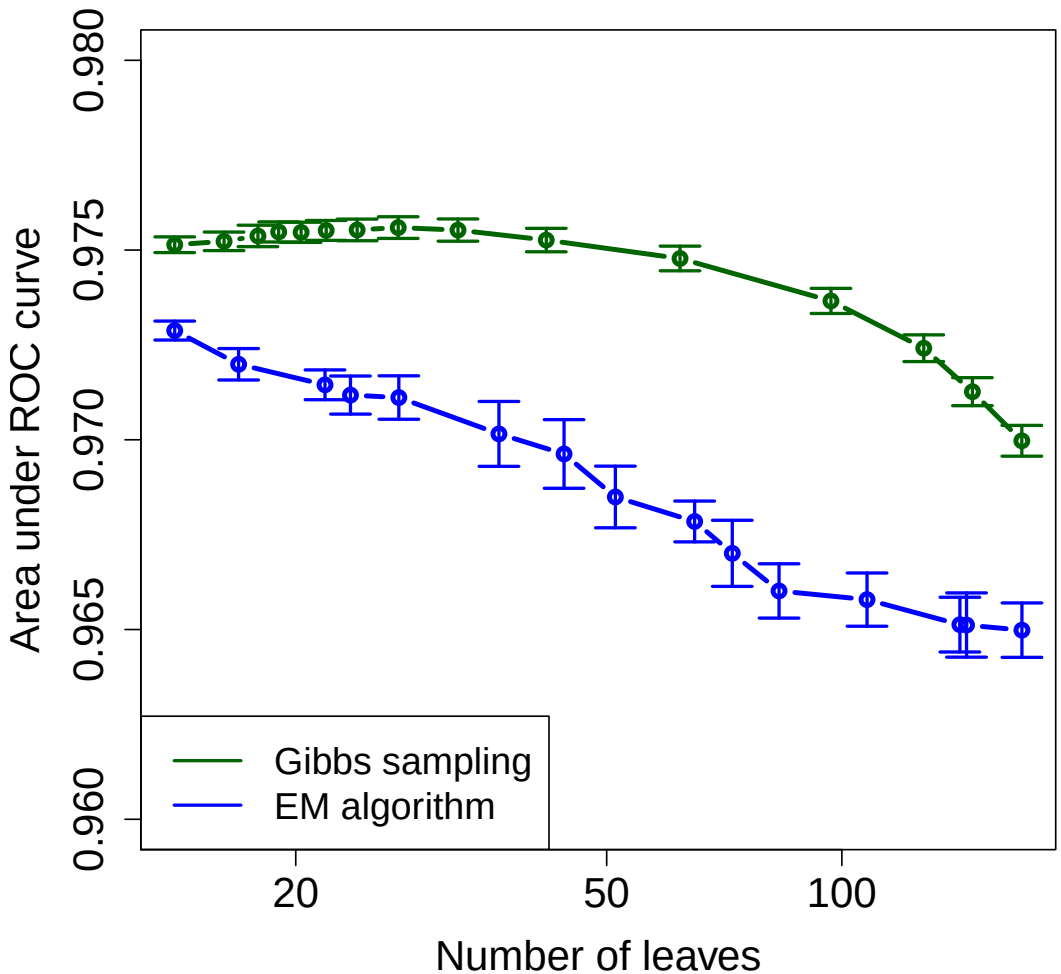
- exon-intron boundaries in genes of higher organisms
- length $= 9$ (7)
- alphabet $\mathcal{A} = \{A, C, G, T\}$

Classification problem

- data from Yeo/Burge 2004
- repeated holdout, samplesize = 500



Classification results



Summary

- premises:
  - parsimonious Markov models
  - latent variables
  - Bayesian prediction

  $\rightarrow$ Gibbs sampling
- key step: structure sampling
  $\rightarrow$ Dynamic programming
- convergence: autocorrelations okay
- classification: Bayesian prediction/Gibbs sampling outperforms classical prediction/EM algorithm
- future: application to other problems involving latent variables