Probabilistic Models for Mining from Labeled Ordered Trees: Application to Glycobiology

Hiroshi Mamitsuka
Bioinformatics Center
Kyoto University

Outline

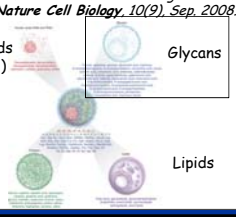
- Introduction
  - Glycobiology: A specific tree application
- Probabilistic model-based approach
  - Preliminary: Hidden Markov model
  - OTMM: Probabilistic model for labeled ordered trees
    - Parameter estimation for OTMM
    - Empirical results
- Summary

Carbohydrate Chains or Glycans

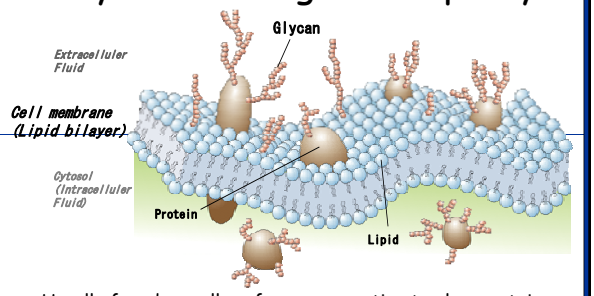
Science special issue:
Carbohydrates and Glycobiology
(*Science*, 291(5512), 2001)

"As indivisible units of life, the cells of all organisms consist of four fundamental macromolecular components: **nucleic acids (including DNA and RNA), proteins, lipids, and glycans.**"

*J.D.Marth, A unified vision of the building blocks of life, **Nature Cell Biology**, 10(9), Sep. 2008.*



Glycans: Biological Property

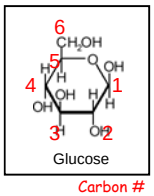


- Usually found on cell surfaces, connecting to glycoproteins
- Crucial to the development and functioning of multicellular organisms

What Does a Glycan Look Like?

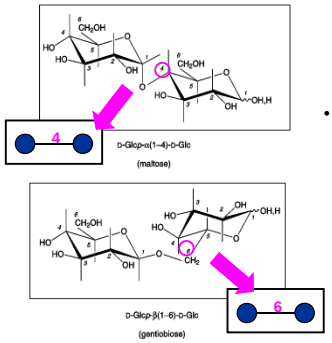
- Building blocks: monosaccharides (sugars)

○	Galp	Galactose
□	GalpNAc	N-acetylgalactosamine
●	Glcp	Glucose
■	GlcpNAc	N-acetylglucosamine
◎	Manp	Mannose
▲	Fucp	Fucose
☆	Xylp	Xylose



Carbon #

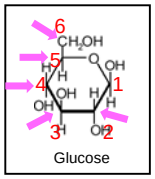
Disaccharides



- Edge, linking two monosaccharides, labeled by carbon numbers

What Does a Glycan Look Like?

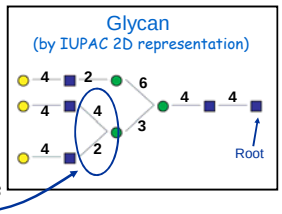
- Monosaccharide connected each other
- No looped connections
- Only one monosaccharide connects to a glycoprotein
  - Resulting in **tree** structures!



What Does a Glycan Look Like (in Summary) ?

- Labeled ordered tree

1. Rooted tree
   - Only one monosaccharide connects to a glycoprotein
2. Labeled tree
   - Nodes labeled by monosaccharides
3. Ordered tree
   - Edges labeled by carbon #s and ordered



Challenges in Glycobiology

- Complex structure and biosynthesis process
- Large range of biological functions, from unimportant to crucial for organism survival
- The same glycan may have different roles
- Experimentally different results based on different environments (i.e., in vitro or in vivo, etc.)

↓

- Informatics must help glycobiology

Glycoinformatics

- Issues:
  - Data collection and database generation
  - Data modeling and glycan structure representation
  - Structure comparison
  - Learning and prediction method/algorithms

etc.

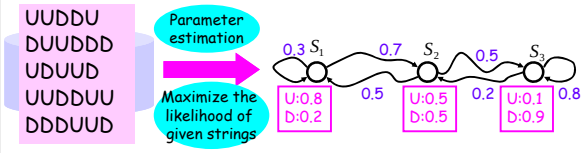
The Objective!

- Develop a probabilistic model for aligning multiple glycans and capturing conserved patterns as in the case of proteins/DNA by HMM
- Develop a probabilistic model for labeled ordered trees!

Review: Hidden Markov Models (HMMs)

Training HMM

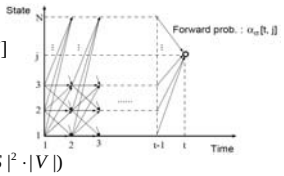
- Probability distributions **trained (estimated) from strings** to present patterns in the strings
- A standard approach (Baum-Welch) maximizes the likelihood to generate given strings



Forward Probability: $\alpha_\sigma[t, j]$

- Given a string, the probability that the current state is j and substring $[1..t]$ is generated, i.e. the probability covering the first part of the string
- Can be computed by dynamic programming over t, due to Markov property
- Updating formula:

$$\alpha_\sigma[t, j] = \sum_i a_{ij} b_j(\sigma_t) \alpha_\sigma[t-1, i]$$

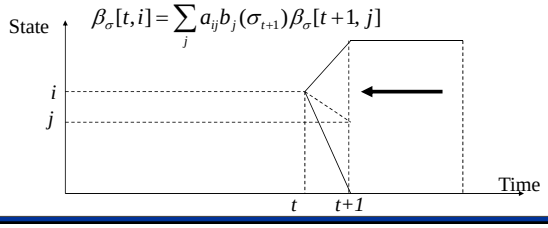


- Can be computed in $O(|S|^2 \cdot |V|)$
  - where S is a set of states and $|V|$ is the string length

Backward Probability: $\beta_\sigma[t, i]$

- Given a string, the probability that the current state is i and substring $[t...n]$ is generated, i.e. the probability covering the last part of the string
- Can be computed by dynamic programming over t in a reverse direction, by the following updating rule:

$$\beta_\sigma[t, i] = \sum_j a_{ij} b_j(\sigma_{t+1}) \beta_\sigma[t+1, j]$$



Baum-Welch Algorithm

- Iterates the following steps until convergence
  - E-step:
    1. Compute forward probabilities: $\alpha_\sigma[t, i]$
    2. Compute backward probabilities: $\beta_\sigma[t, j]$
    3. Compute the expectation value of state transition ij using forward and backward probabilities:

$$E_{P_\sigma}[\#((i, j), \sigma] \propto \sum_t \alpha_\sigma[t, i] a_{ij} b_j(\sigma_{t+1}) \beta_\sigma[t+1, j]$$

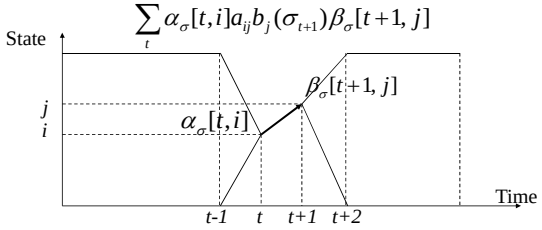
  - M-step:
    1. Update transition probability a_{ij} using expectation values:

$$\hat{a}_{ij} = \frac{E_{P_\sigma}[\#((i, j), \sigma]}{\sum_j E_{P_\sigma}[\#((i, j), \sigma]}$$

Baum-Welch Algorithm Picture

- We can estimate expectation value on the state transition from i to j by using forward and backward probabilities:

$$\sum_t \alpha_\sigma[t, i] a_{ij} b_j(\sigma_{t+1}) \beta_\sigma[t+1, j]$$



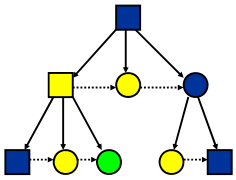
Probabilistic Models for Labeled Ordered Trees

Hidden Tree Markov Model (HTMM)

[Deligenti, 2003]

- Extension of HMM for strings to a model for trees
- Proposed in the literature of image processing
- State depends on that of the parent only

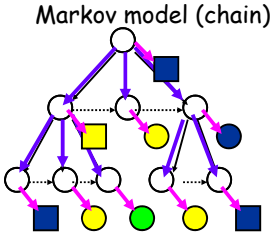
Labeled ordered tree



: Tree edges

: Children ordering

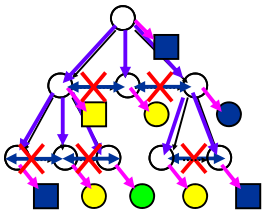
Markov model (chain)



: State transition

: Letter generation

HTMM Cannot Capture Sibling Dependencies!



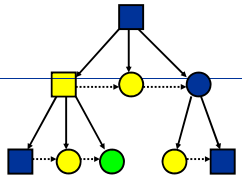
: State transition

: Letter generation

Our First Model: Probabilistic Sibling-dependent Tree Markov Model (PSTMM)

[SDM04, ISMB04, TKDE05, ISMB06]

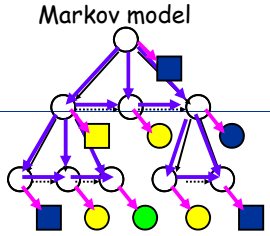
Labeled ordered tree



: Tree edges

: Children ordering

Markov model



: State transition

: Letter generation

Note: State of a node depends upon the states of both elder sibling and parent, except the case that the node is the eldest sibling

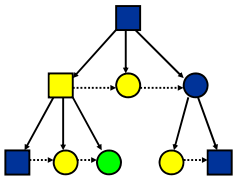
Properties of PSTMM

- Advantage
  - Empirically outperformed simpler models, such as HTMM, in classifying labeled ordered trees in synthetic as well as real data
- Drawbacks
  - Computational complexity in learning is equivalent to that of context free grammars for strings: maximum practical bound
    - Time: $O(|T| \cdot |S|^3 \cdot |V| \cdot |C|)$, Space: $O(|S|^3)$
      - where T is a set of given trees, S is a set of states, V is a set of nodes and C is the maximum size of children
  - Overfitting problems!
  - Difficult to retrieve patterns from learned states

Our New Model: Ordered Tree Markov Model (OTMM)

[KDD06, TKDD08]

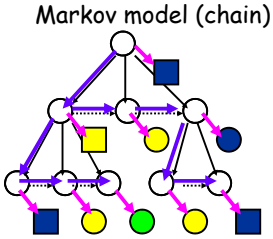
Labeled ordered tree



: Tree edges

: Children ordering

Markov model (chain)



: State transition

: Letter generation

Note: State of a node depends upon the states of only elder sibling, except the eldest sibling, where the state depends upon that of parent

Define OTMM Parameters

- Two main probability parameters
  - State transition probability (two cases):

    $a[q,m] = a[s_q, s_m] (= P(z_j^u = s_m \mid z_p^u = s_q))$

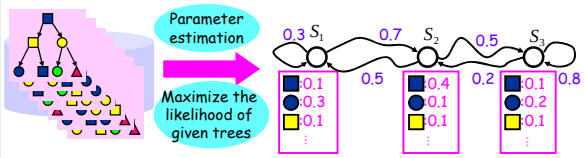
    - Probability that the state of j is s_m given that the state of the parent is s_q (eldest siblings)

    $a[l,m] = a[s_l, s_m] (= P(z_j^h = s_m \mid z_i^u = s_l))$

    - Probability that the state of j is s_m given that the state of the immediately elder sibling is s_l (otherwise)
  - Label output probability: $b[s_l, \sigma_h]$
    - Probability that the state s_l outputs σ_h

Training OTMM

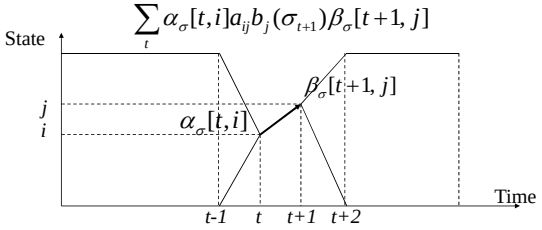
- Probability distributions **trained (estimated) from "labeled ordered trees"** to present patterns/rules in them
- A standard approach of HMMs extended
  - Computational complexities kept at the same level !



Review Again: Baum-Welch Algorithm Picture

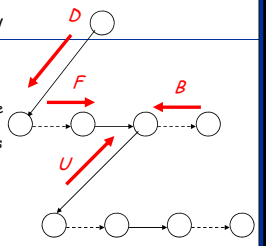
- We can estimate expectation value on the state transition from *i* to *j* by using forward and backward probabilities:

$$\sum_t \alpha_\sigma[t,i]a_{ij}b_j(\sigma_{t+1})\beta_\sigma[t+1,j]$$



Four Marginalized Probabilities

- Four directions: Parent-child & Siblings
  - Child to Parent
  - Parent to Child
  - Younger sibling to elder sibling
  - Elder sibling to younger sibling
- Define four probabilities for a tree u with nodes $x_1, ..., x_n$
  - Upward $U_u(s_i, x_n)$
    - Covers subtree
  - Downward $D_u(s_i, x_1)$
    - Covers wholel tree except the subtree
  - Backward $B_u(s_m, x_j)$
    - Covers subtrees of all younger siblings
  - Forward $F_u(s_i, x_j)$
    - Covers whole tree except that by Backward
- Each computed by dynamic programming (DP) efficiently

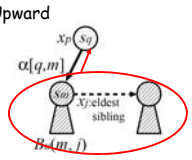


Upward Probabilities

- Cover subtree
- Can be computed from backward probabilities (covering younger siblings and their children) of the eldest child

$$U_u(q,p) = \begin{cases} \text{If } C_u(p) = \emptyset \text{ then } b[q, o_p^u] \\ \text{otherwise} \\ b[q, o_p^u] \sum_{m=1}^{|S|} \alpha[q,m] B_u(m,j) \\ (\text{s.t. } x_j^u = x_{\leftarrow}^u(p)) \end{cases}$$

Upward

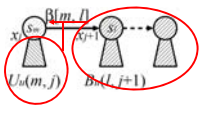


Backward Probabilities

- Cover younger siblings and their subtrees
- Can be computed from the upward probability of the eldest child and backward probabilities of the younger child

$$B_u(m,j) = \begin{cases} \text{If } x_j^u = x_{\rightarrow}^u(p) \text{ then } U_u(m,j), \\ \text{otherwise} \\ U_u(m,j) \sum_{l=1}^{|S|} \beta[m,l] B_u(l, j+1). \end{cases}$$

Backward

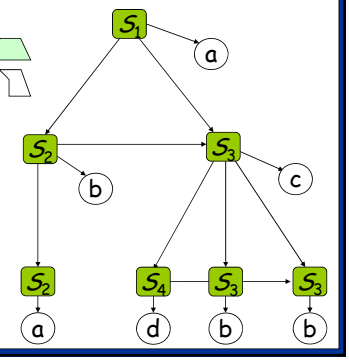


Computing Backward and Upward Probabilities with Dynamic Programming

- Example

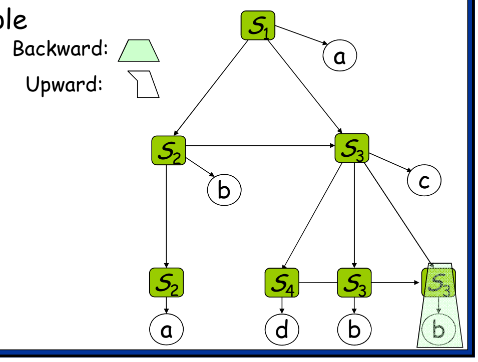
Backward:

Upward:

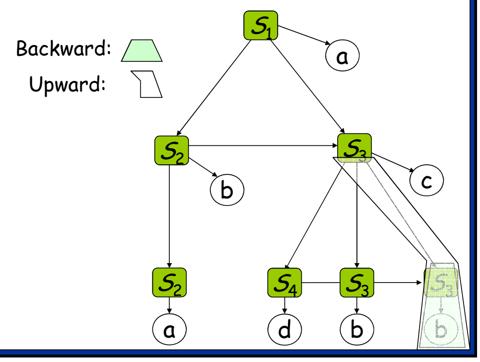


Computing Backward and Upward Probabilities with Dynamic Programming

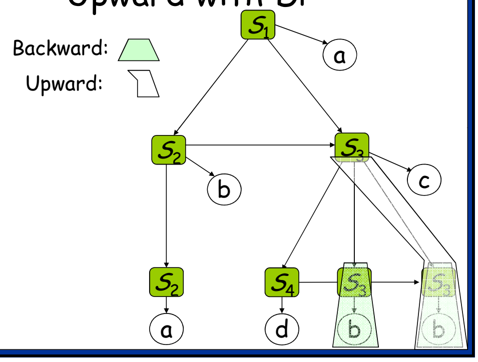
- Example



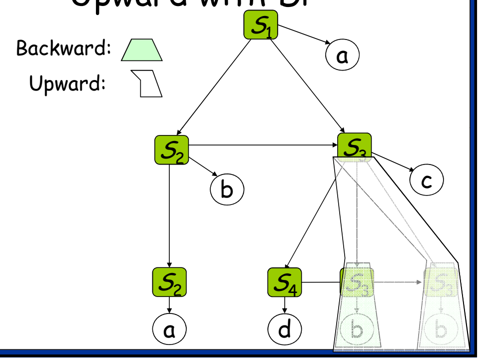
Computing Backward and Upward Probabilities with Dynamic Programming



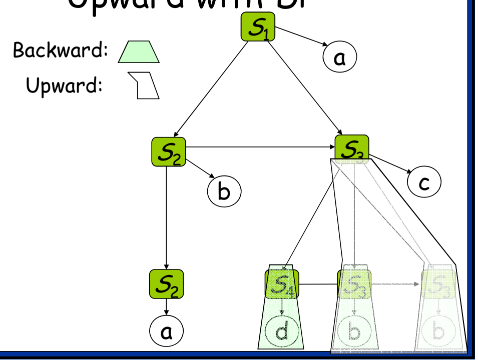
Computing Backward and Upward with DP



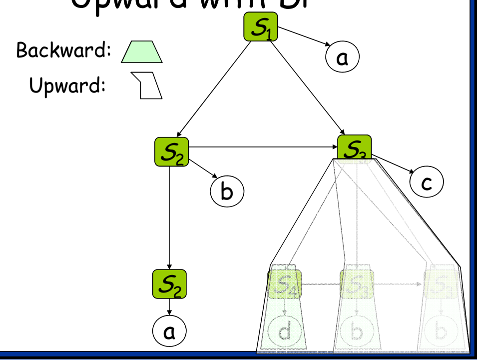
Computing Backward and Upward with DP



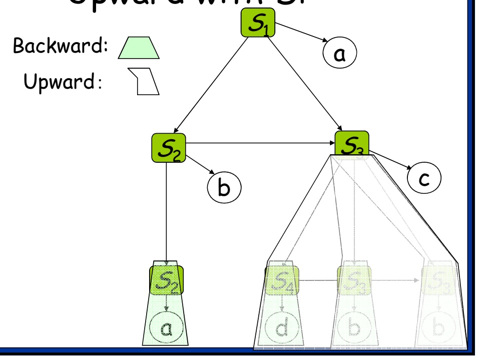
Computing Backward and Upward with DP



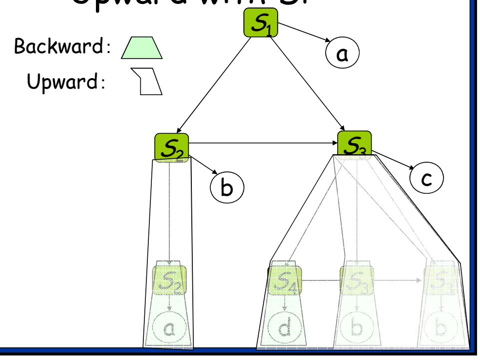
Computing Backward and Upward with DP



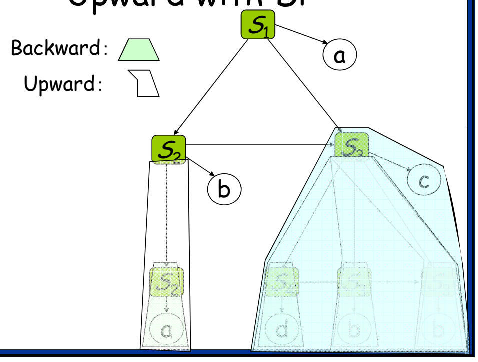
Computing Backward and Upward with DP



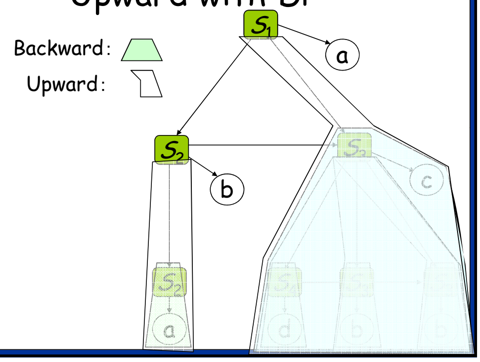
Computing Backward and Upward with DP



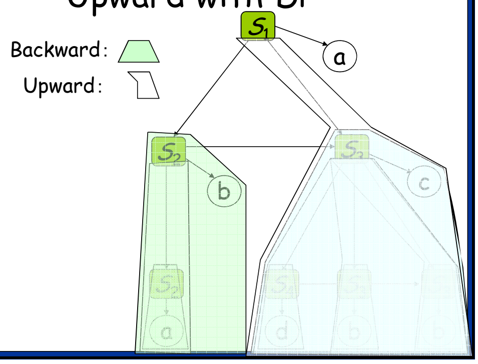
Computing Backward and Upward with DP



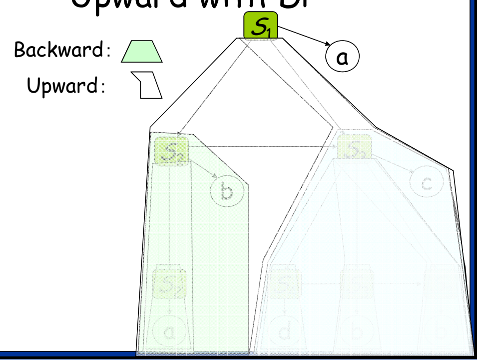
Computing Backward and Upward with DP



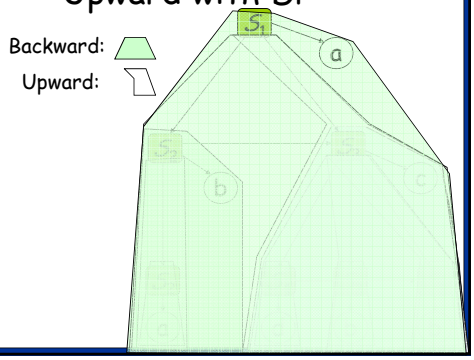
Computing Backward and Upward with DP



Computing Backward and Upward with DP



Computing Backward and Upward with DP



Computing Likelihood

- Possible by just using forward probabilities only, since they can cover all nodes of the input tree:

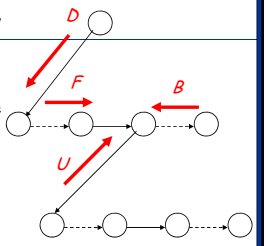
$$L(T_u) = \sum_{l=1}^{|S|} \pi[l] U_u(l, 1).$$

- Can be computed for a given set of trees:

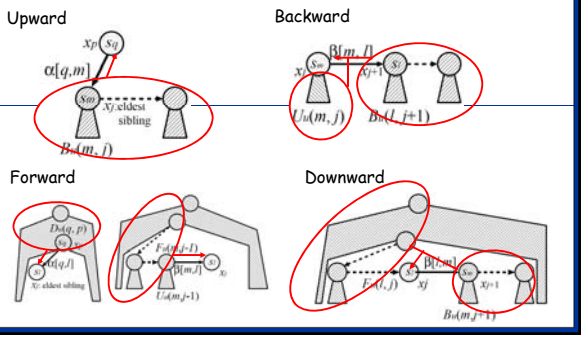
$$L(\mathbf{T}) = \prod_{u=1}^{|\mathbf{T}|} L(T_u) = \prod_{u=1}^{|\mathbf{T}|} \sum_{l=1}^{|S|} \pi[l] U_u(l, 1).$$

Four Marginalized Probabilities

- Four directions: Parent-child & Siblings
  - Child to Parent
  - Parent to Child
  - Younger sibling to elder sibling
  - Elder sibling to younger sibling
- Define four probabilities for a tree u with nodes $x_1, ..., x_n$
  - Upward $U_u(s_q, x_p)$
    - Covers subtree
  - Downward $D_u(s_l, x_i)$
    - Covers wholel tree except the subtree
  - Backward $B_u(s_m, x_j)$
    - Covers subtrees of all younger siblings
  - Forward $F_u(s_l, x_j)$
    - Covers whole tree except that by Backward
- Each computed by dynamic programming (DP) efficiently



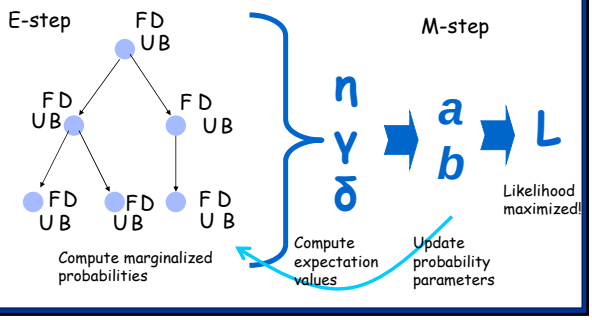
Dynamic Programming for Marginalized Probabilities



Learning OTMM

- Extension of Baum-Welch of HMM to OTMM
- Made possible by using four auxiliary probabilities: forward, backward, upward and downward
- Dynamic programming: polynomial (cubic!) order computation time and space

EM Iteration Procedure



OTMM: Computational Complexity

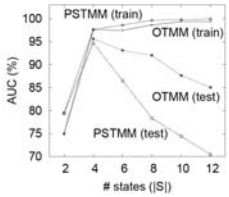
- Notation
  - |T|:#trees, |S|:#states, |V|:string length
- Comparison with PSTMM (and HTMM)
  - Efficiency always increased by |S|

	Time
OTMM	$O(\|\mathbf{T}\| \cdot \|S\|^2 \cdot \|V\|)$
HTMM	$O(\|\mathbf{T}\| \cdot \|S\|^2 \cdot \|V\|)$
PSTMM	$O(\|\mathbf{T}\| \cdot \|S\|^3 \cdot \|V\| \cdot \|C\|)$

	Space
OTMM	$\max\{O(\|S\| \cdot \|V\|), O(\|S\|^2), O(\|S\| \cdot \|\sum\|)\}$
HTMM	$\max\{O(\|S\| \cdot \|V\|), O(\|S\|^2), O(\|S\| \cdot \|\sum\|)\}$
PSTMM	$\max\{O(\|S\|^2 \cdot \|V\|), O(\|S\|^3), O(\|S\|^2 \cdot \|\sum\|)\}$

Synthetic Data Experiment

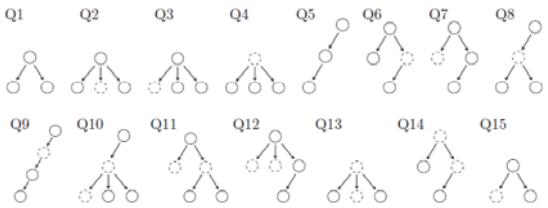
- Evaluated by classification
  - Data (Pos vs Neg): Trees with vs. without fixed patterns
  - # random training and test trees: 100
  - AUC (Area under the ROC curve) for both training and test



Overfitting caused by PSTMM, relieved by OTMM

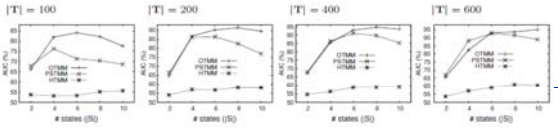
Synthetic Data Experiment

- Fifteen patterns of tree fragments embedded in positives

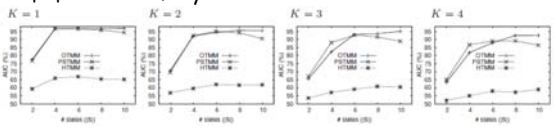


Synthetic Data Experiment Performance (AUC)

- Increase #training trees (|T|) and #states (|S|) with Q=1, #patterns=3 and fully-connected models

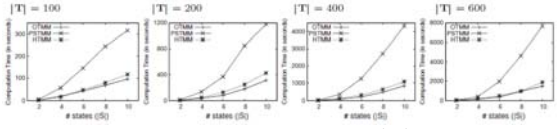


- Increase #patterns and #states (|S|) with Q=1, |T|=600 and fully-connected models

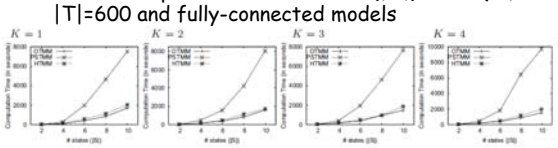


Synthetic Data Experiment Computation Time

- Increase #training trees (|T|) and #states (|S|) with Q=1, #patterns=3 and fully-connected models



- Increase #patterns and #states (|S|) with Q=1, |T|=600 and fully-connected models



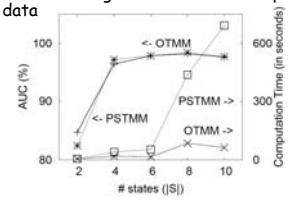
Synthetic Data Experiment Summary of AUC

- AUC for fifteen patterns of tree fragments

Fragment	OTMM	PSTMM	HTMM
Q1	91.2	**93.1**	60.2
Q2	86.3	**90.8**	57.6
Q3	**91.7**	91.3	58.2
Q4	**95.5**	95.2	63.7
Q5	**91.0**	89.9	60.9
Q6	**88.7**	87.8	60.4
Q7	87.1	**88.0**	60.2
Q8	**91.9**	91.1	64.8
Q9	**71.2**	70.2	55.2
Q10	83.3	**86.7**	61.2
Q11	**88.7**	88.3	61.2
Q12	83.0	**85.2**	58.1
Q13	82.6	**83.0**	53.9
Q14	**87.2**	85.6	54.4
Q15	73.9	**75.1**	54.9

Glycan Data Experiment

- Evaluated in classification
  - 10-fold Cross-validation
  - Pos vs. Neg: N-glycans vs. O-glycans
  - Used parameter settings achieved the best performance in synthetic data

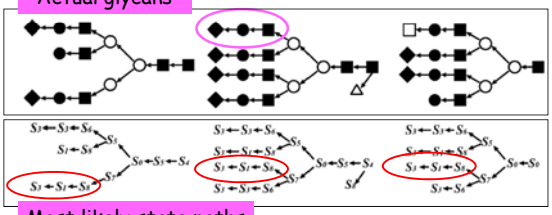


OTMM computed efficiently,
keeping the same predictive performance

Multiple Tree Alignment

- Most likely state paths for three glycans allow multiple tree alignment:

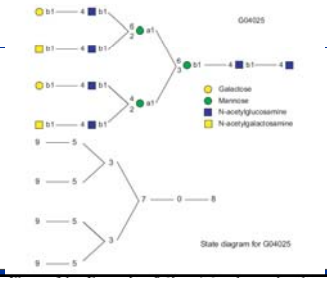
Actual glycans



Most likely state paths

Glycan Pattern Mining

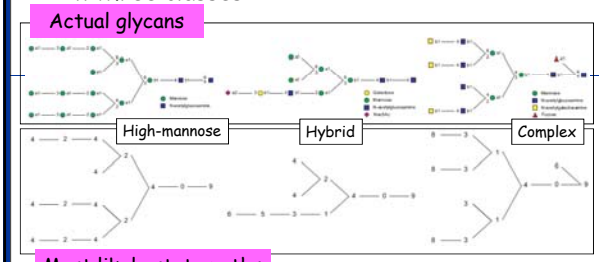
- Assigning states for node labels by most likely state path (by Viterbi algorithm)



Findings from Glycans

- Actual glycans and the most likely state paths in three classes

Actual glycans



Most likely state paths

Summary

- New Markov model for "labeled ordered trees" presented
  - Has moderate complexity, keeping high representation power
  - Is an extension of hidden Markov model
- Advantages:
  - Noise robustness
  - Biological comprehensibility
- Drawback: patterns (probabilities) are dependencies between neighboring tree nodes, being unable to show a larger pattern

Acknowledgements

Kyoto University
Bioinformatics Center
Institute for Chemical Research

- Kosuke Hashimoto
- Nobuhisa Ueda
- Kiyoko Flora Aoki-Kinoshita (Currently Soka University)
- Minoru Kanehisa

Reference Information

- [TKDD08]
  - A New Efficient Probabilistic Model for Mining Labeled Ordered Trees Applied to Glycobiology., *ACM Transactions on Knowledge Discovery from Data*, 2 (1), Article No. 6, 2008.
- [DDT08]
  - Informatic Innovations in Glycobiology: Relevance to Drug Discovery., *Drug Discovery Today*, 13(3/4), 118-123 (2008).

Funding Acknowledgements

- BIRD, JST
- Research fellowship for Young Scientists, JSPS
- Grant-in-Aid for Young Scientists, JSPS
- Kyoto University 21st Century COE program, Knowledge Information Infrastructure for Genome Science, JSPS
- Education and Research Organization for Genome Information Science, JST