

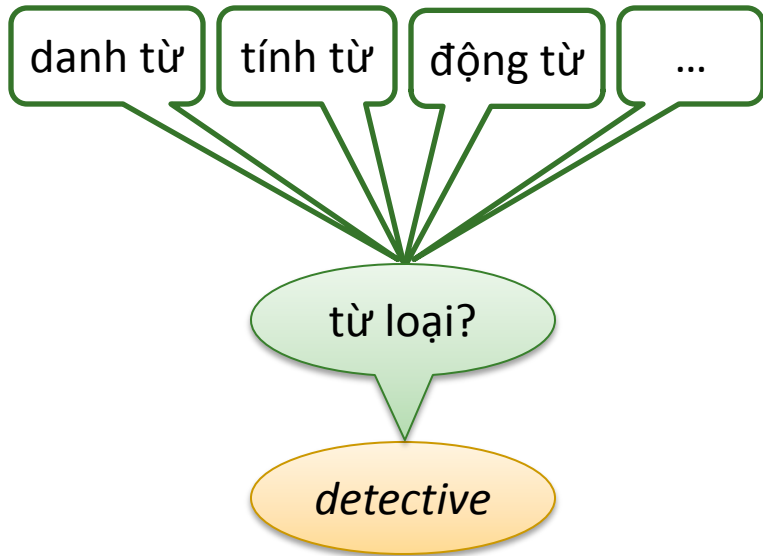
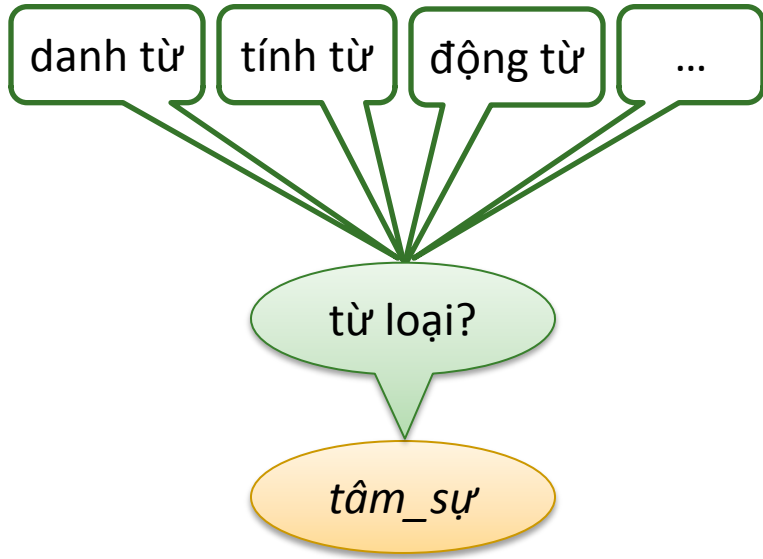


TRƯỜNG HÈ VỀ HỌC MÁY THỐNG KÊ

# Trường ngẫu nhiên có điều kiện và ứng dụng

**Phan Xuân Hiếu**

Phòng thí nghiệm Công nghệ Tri thức,  
Khoa CNTT, Trường ĐH Công nghệ, ĐHQG Hà Nội  
Email: [hieupx@vnu.edu.vn](mailto:hieupx@vnu.edu.vn)



danh từ    tính từ    động từ    ...

từ loại?

tâm\_sự

... có nhiều tâm\_sự trong lòng ...

... mình muốn tâm\_sự với bạn ...

danh từ    tính từ    động từ    ...

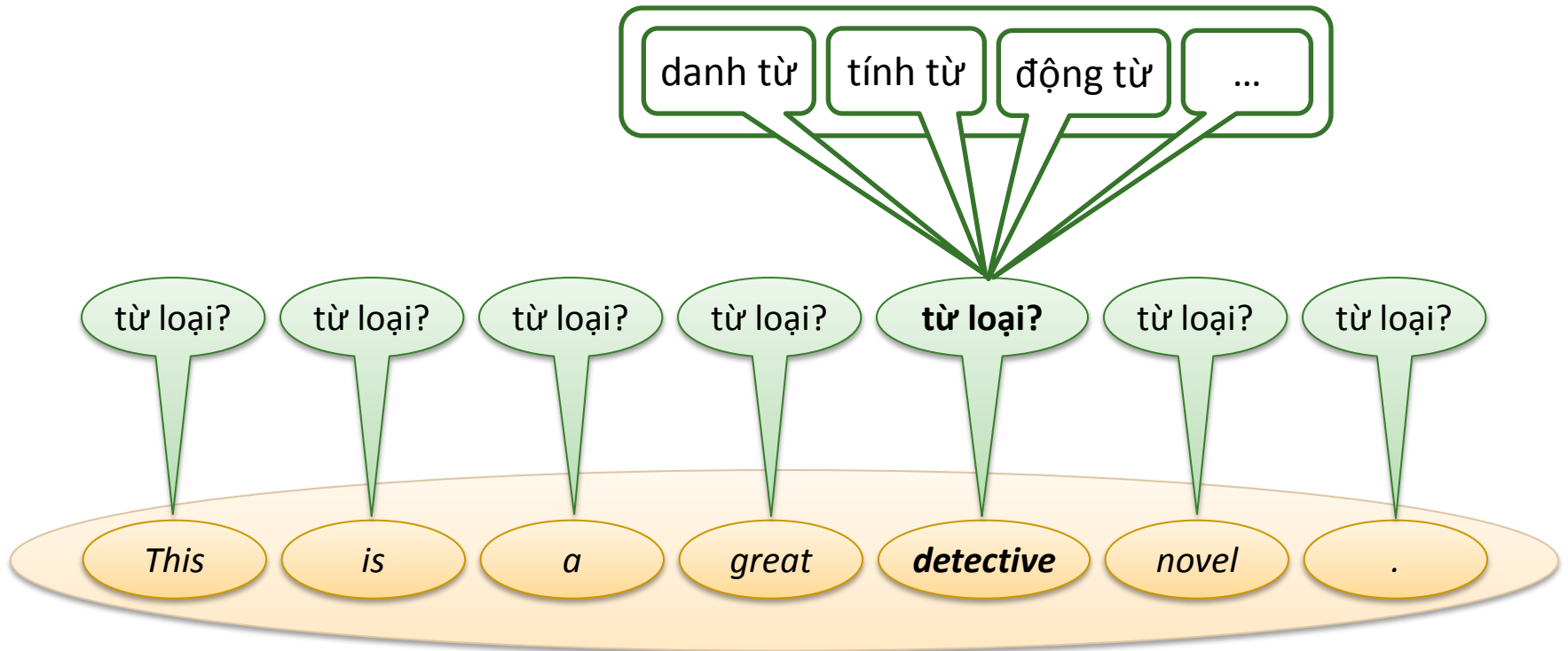
từ loại?

detective

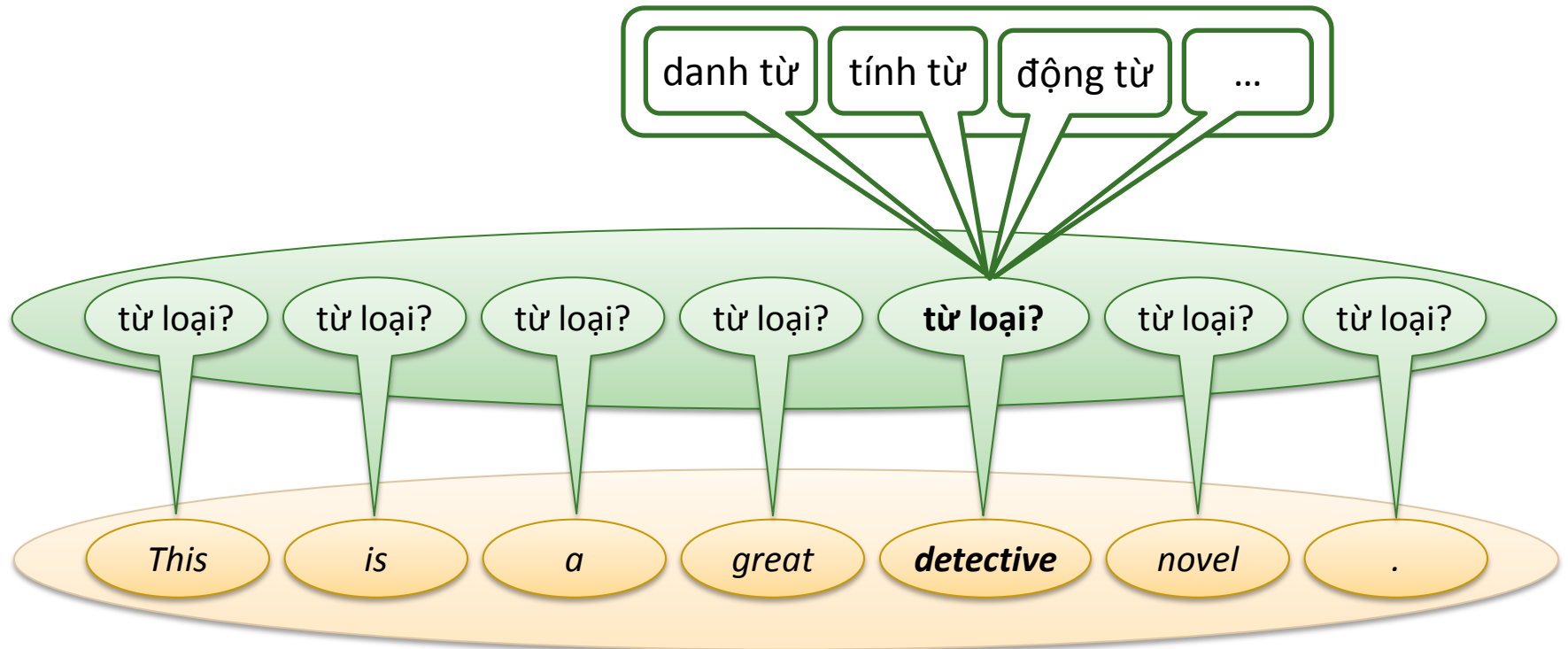
Sherlock Holmes is a great detective .

This is a great detective novel .

# Phân lớp đơn lẻ và độc lập



# Phân lớp dựa trên sự phụ thuộc và tương tác



**Đoán nhận trên dữ liệu có cấu trúc**  
**structured (output) prediction/learning**

$$\mathbf{s} = \{s_1, s_2, \dots, s_T\}$$

Chuỗi trạng thái (state sequence)

Tập nhãn (label set)

$$\mathbf{L} = \{l_1, l_2, \dots, l_q\}$$

danh từ

tính từ

động từ

...

từ loại?

từ loại?

từ loại?

từ loại?

từ loại?

từ loại?

từ loại?

*This*

*is*

*a*

*great*

***detective***

*novel*

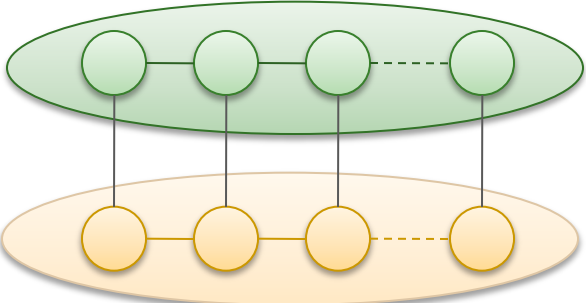
.

Chuỗi dữ liệu quan sát (observation sequence)

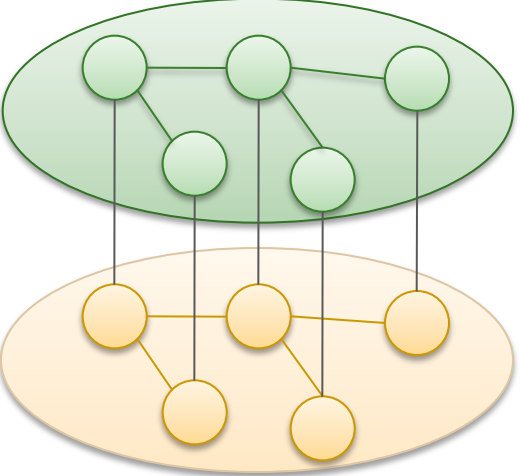
$$\mathbf{o} = \{o_1, o_2, \dots, o_T\}$$

# Một số dạng cấu trúc đồ thị của dữ liệu

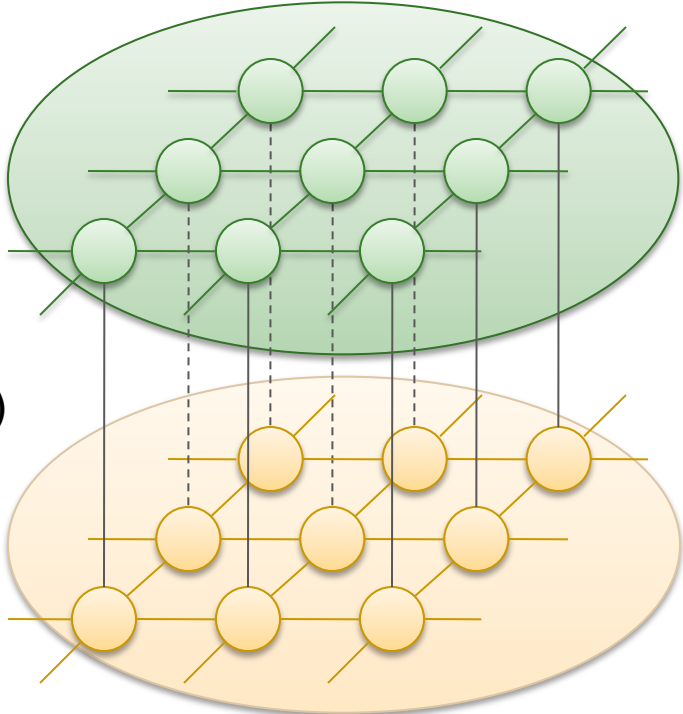
Chuỗi  
(sequence,  
linear-chain)



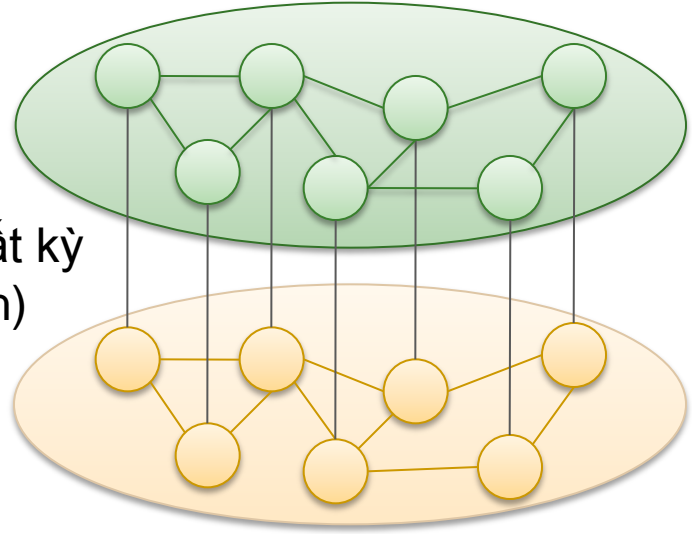
Cây  
(tree)



Lưới  
(grid)



Đồ thị bất kỳ  
(graph)





# Bài toán gắn nhãn và phân đoạn trên dữ liệu chuỗi

- Ký hiệu  $\mathbf{L} = \{l_1, l_2, \dots, l_q\}$  là tập  $q$  nhãn (lớp) được định nghĩa trước.
- Cho  $\mathbf{o} = \{o_1, o_2, \dots, o_T\}$  là một chuỗi dữ liệu đầu vào (input data sequence) bao gồm  $T$  thành phần dữ liệu (data observations)
- Vấn đề gắn nhãn hoặc phân đoạn là đoán nhận (predict) dãy nhãn đầu phù hợp nhất (the most likely output label sequence) cho  $\mathbf{o}$

$$\mathbf{s}^* = \{s_1^*, s_2^*, \dots, s_T^*\} \quad (s_i^* \in \mathbf{L})$$

# Bài toán gán nhãn và phân đoạn trên dữ liệu chuỗi

- Ký hiệu  $\mathbf{L} = \{l_1, l_2, \dots, l_q\}$  là tập  $q$  nhãn (lớp) được định nghĩa trước.
- Cho  $\mathbf{o} = \{o_1, o_2, \dots, o_T\}$  là một chuỗi dữ liệu đầu vào (input data sequence) bao gồm  $T$  thành phần dữ liệu (data observations)
- Vấn đề gán nhãn hoặc phân đoạn là đoán nhận (predict) dãy nhãn đầu phù hợp nhất (the most likely output label sequence) cho  $\mathbf{o}$

$$\mathbf{s}^* = \{s_1^*, s_2^*, \dots, s_T^*\} \quad (s_i^* \in \mathbf{L})$$

Ví dụ về gán nhãn (labeling): gán nhãn từ loại

*Rolls\_NNP Royce\_NNP Motor\_NNP Cars\_NNPS Inc.\_NNP said\_VBD it\_PRP  
expects\_VBZ its\_PRP\$ U.S.\_NNP sales\_NNS to\_TO remain\_VB steady\_JJ ...*

# Bài toán gắn nhãn và phân đoạn trên dữ liệu chuỗi

- Ký hiệu  $\mathbf{L} = \{l_1, l_2, \dots, l_q\}$  là tập  $q$  nhãn (lớp) được định nghĩa trước.
- Cho  $\mathbf{o} = \{o_1, o_2, \dots, o_T\}$  là một chuỗi dữ liệu đầu vào (input data sequence) bao gồm  $T$  thành phần dữ liệu (data observations)
- Vấn đề gắn nhãn hoặc phân đoạn là đoán nhận (predict) dãy nhãn đầu phù hợp nhất (the most likely output label sequence) cho  $\mathbf{o}$

$$\mathbf{s}^* = \{s_1^*, s_2^*, \dots, s_T^*\} \quad (s_i^* \in \mathbf{L})$$

Ví dụ về gắn nhãn (labeling): gắn nhãn từ loại

*Rolls\_NNP Royce\_NNP Motor\_NNP Cars\_NNPS Inc.\_NNP said\_VBD it\_PRP  
expects\_VBZ its\_PRP\$ U.S.\_NNP sales\_NNS to\_TO remain\_VB steady\_JJ ...*

Ví dụ về phân đoạn (segmentation): phân đoạn cụm từ

*[Rolls Royce Motor Cars Inc. NP] [said VP] [it NP] [expects VP]  
[its U.S. sales NP] [to remain VP] [steady ADJP] ...*

**Trường ngẫu nhiên có điều kiện**  
**mô hình hóa, định nghĩa và khái niệm**

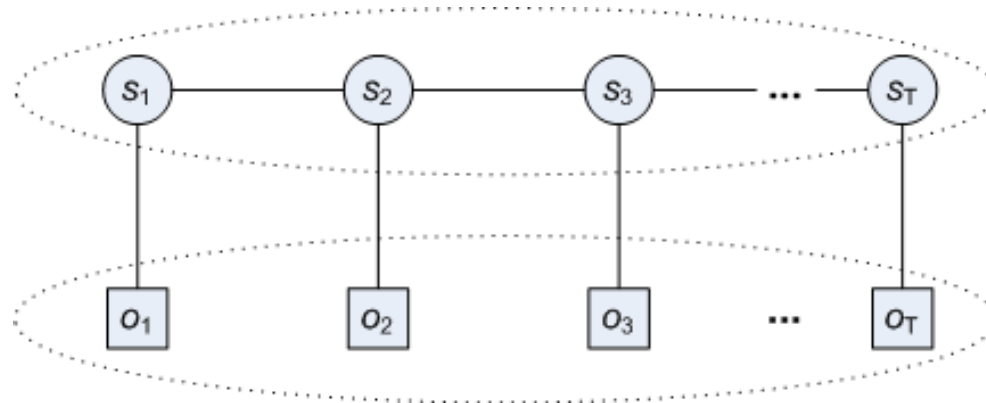
# Lịch sử Conditional Random Fields (CRFs)

John Lafferty, Andrew McCallum, and Fernando Pereira. *Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data*, ICML 2001.



- Test-of-Time Award of ICML 2011.
- Được trích dẫn: hơn 1600 lần trên ACM và hơn 8000 lần trên Google Scholar (08/2015)

# Linear-chain CRFs



- Tập nhãn:  $\mathbf{L} = \{l_1, l_2, \dots, l_q\}$
- Chuỗi dữ liệu quan sát (data observation sequence):  $\mathbf{o} = \{o_1, o_2, \dots, o_T\}$
- Chuỗi trạng thái/nhãn (state/label sequence):  $\mathbf{s} = \{s_1, s_2, \dots, s_T\}$
- Mô hình CRFs:

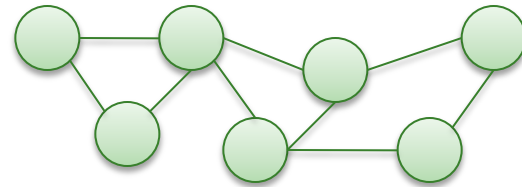
$$p(\mathbf{s}|\mathbf{o})$$

- Đoán nhận (prediction) chuỗi nhãn phù hợp nhất:  $\mathbf{s}^* = \operatorname{argmax}_{\mathbf{s}} p(\mathbf{s}|\mathbf{o})$

# Markov Random Fields (MRFs)



Chuỗi (sequence, linear-chain)

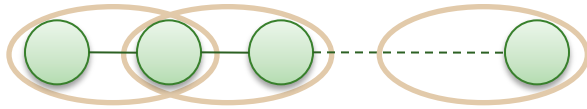


Đồ thị tổng quát (graph)

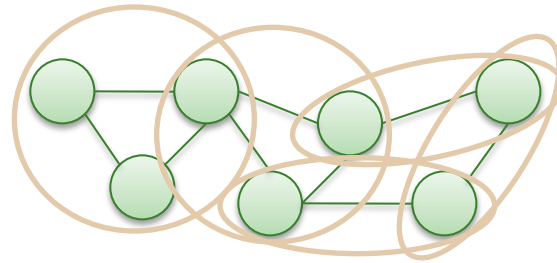
- Ký hiệu đồ thị  $G = (V, E)$  với tập đỉnh  $V = \{v_1, v_2, \dots, v_T\}$  và tập cạnh  $E$
- Mỗi đỉnh nhận giá trị từ một tập hữu hạn cho trước.
- Phân bố cần quan tâm:

$$p(v_1, v_2, \dots, v_T)$$

# Markov Random Fields (cont'd)



Chuỗi (sequence, linear-chain)



Đồ thị tổng quát (general graph)

- Gọi  $\mathcal{C}$  là tập tất cả các đồ thị con đầy đủ cực đại (maximal cliques) trong  $G$
- Theo định lý cơ bản về trường ngẫu nhiên (random fields) bởi Hammersley và Clifford (1971),  $p(v_1, v_2, \dots, v_T)$  có thể được phân rã như sau:

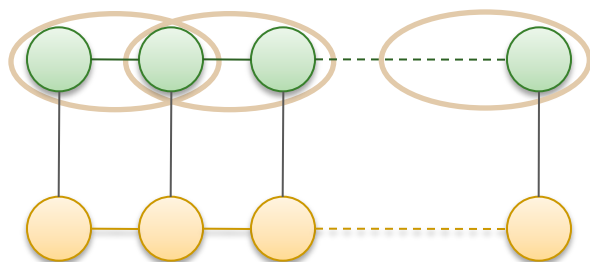
$$p(v_1, v_2, \dots, v_T) = \frac{1}{Z} \prod_{c \in \mathcal{C}} \Psi(V_c)$$

Trong đó:  $\Psi(V_c)$  là hàm địa phương (local/potential function) và

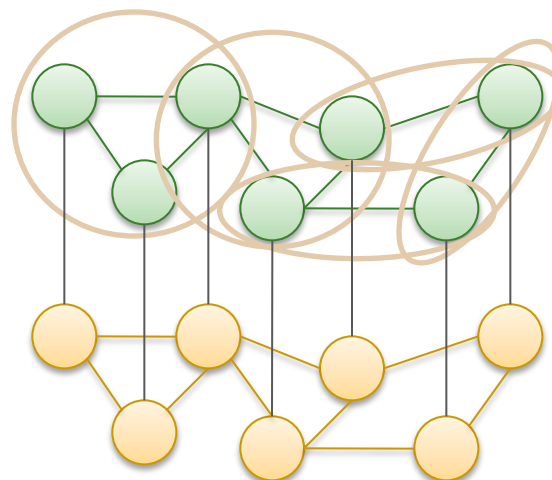
$$Z = \sum_{v_1, v_2, \dots, v_T} \prod_{c \in \mathcal{C}} \Psi(V_c) \quad \text{là hàm chuẩn hóa (normalization function)}$$



# Conditional Random Fields



Chuỗi (sequence, linear-chain)



Đồ thị tổng quát (general graph)

- Mô hình CRFs:

$$p(\mathbf{s}|\mathbf{o}) = p(s_1, s_2, \dots, s_T|\mathbf{o}) = \frac{1}{Z(\mathbf{o})} \prod_{c \in \mathcal{C}} \Psi(\mathbf{s}_c, \mathbf{o})$$

với

$$Z(\mathbf{o}) = \sum_{\mathbf{s}} \prod_{c \in \mathcal{C}} \Psi(\mathbf{s}_c, \mathbf{o})$$

# Conditional Random Fields (cont'd)

- Lafferty et al. 2001 định nghĩa hàm tiềm năng:

$$\Psi(\mathbf{s}_c, \mathbf{o}) = \exp\left(\sum_{i=1}^n \lambda_i f_i(\mathbf{s}_c, \mathbf{o}, c)\right)$$

- Khi đó, mô hình CRFs có thể được viết lại:

$$p_{\theta}(\mathbf{s}|\mathbf{o}) = \frac{1}{Z_{\theta}(\mathbf{o})} \exp\left(\sum_{c \in \mathcal{C}} \sum_{i=1}^n \lambda_i f_i(\mathbf{s}_c, \mathbf{o}, c)\right)$$

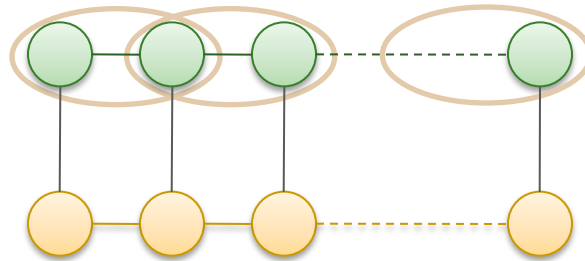
- Trong đó:

$\mathbf{F} = \{f_1, f_2, \dots, f_n\}$  là tập thuộc tính (feature)

$\theta = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$  là tập trọng số tương ứng với các thuộc tính

$$Z_{\theta}(\mathbf{o}) = \sum_{\mathbf{s}} \exp\left(\sum_{c \in \mathcal{C}} \sum_{i=1}^n \lambda_i f_i(\mathbf{s}_c, \mathbf{o}, c)\right)$$

# Linear-chain Conditional Random Fields



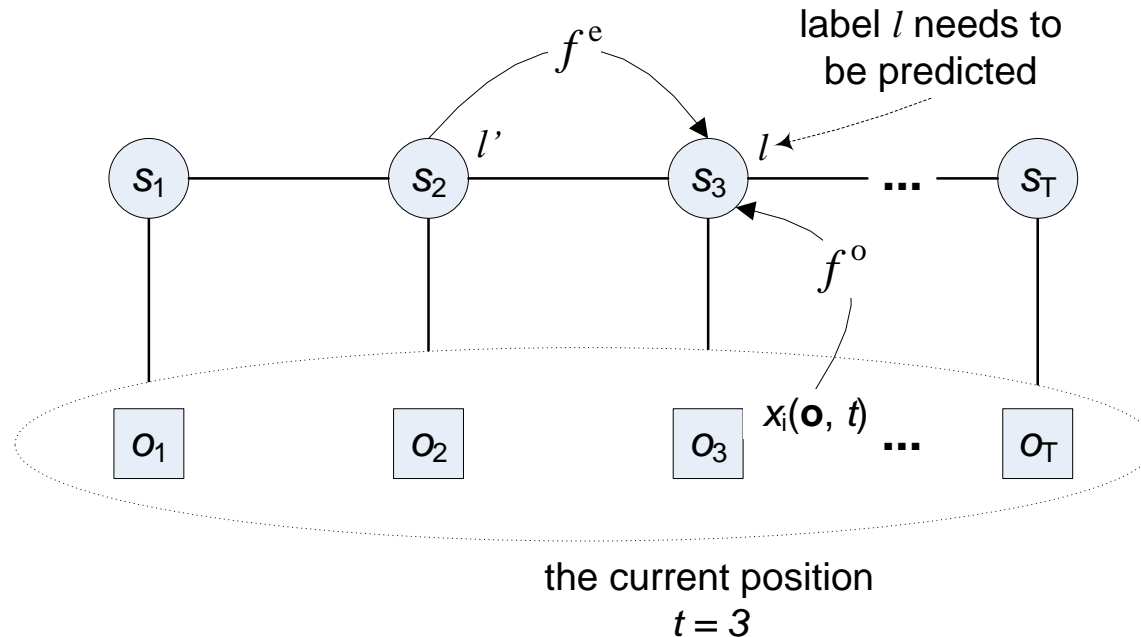
Chuỗi (sequence, linear-chain)

- Các đồ thị con đầy đủ cực đại (maximal cliques) chính là tập các cặp đỉnh/trạng thái liền kề.
- Có thể viết lại mô hình:

$$p_{\theta}(\mathbf{s}|\mathbf{o}) = \frac{1}{Z_{\theta}(\mathbf{o})} \exp\left(\sum_{t=1}^T \sum_{i=1}^n \lambda_i f_i(s_{t-1}, s_t, \mathbf{o}, t)\right)$$

$$Z_{\theta}(\mathbf{o}) = \sum_{\mathbf{s}} \exp\left(\sum_{t=1}^T \sum_{i=1}^n \lambda_i f_i(s_{t-1}, s_t, \mathbf{o}, t)\right)$$

# Các dạng thuộc tính trong CRFs



Hai dạng thuộc tính:

- ❑ Thuộc tính cạnh ( $e$  – edge feature): phụ thuộc Markov giữa các vị trí liền kề
- ❑ Thuộc tính quan sát ( $o$  – observation feature): đặc điểm quan sát được từ chuỗi dữ liệu đầu vào

# Edge feature và observation feature

- Thuộc tính cạnh (edge feature):

$$f_{\langle l', l \rangle}^e(s_{t-1}, s_t, t) = \begin{cases} 1 & \text{nếu } s_{t-1} = l' \text{ và } s_t = l \\ 0 & \text{ngược lại} \end{cases}$$

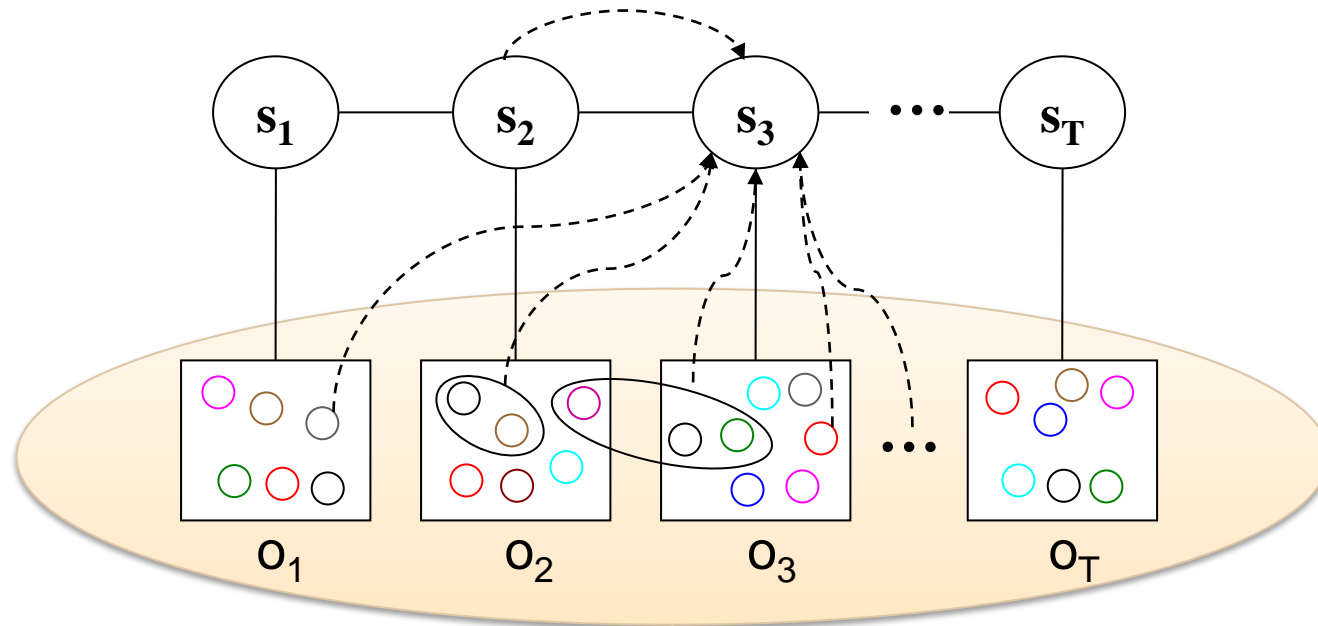
Ví dụ:  $f_{\langle \text{tính\_từ}, \text{danh\_từ} \rangle}^e(s_{t-1}, s_t, t) = \begin{cases} 1 & \text{nếu } s_{t-1} = \text{tính\_từ} \text{ và } s_t = \text{danh\_từ} \\ 0 & \text{ngược lại} \end{cases}$

- Thuộc tính quan sát (observation feature):

$$f_{\langle x_j, l \rangle}^o(s_t, \mathbf{o}, t) = \begin{cases} 1 & \text{nếu } x_j(\mathbf{o}, t) = \text{TRUE} \text{ và } s_t = l \\ 0 & \text{ngược lại} \end{cases}$$

Ví dụ:  $f_{\langle \text{suffix}(o_t) = \text{tive}, \text{tính\_từ} \rangle}^o(s_t, \mathbf{o}, t)$   
 $= \begin{cases} 1 & \text{nếu } [\text{suffix}(o_t) = \text{tive}] = \text{TRUE} \text{ và } s_t = \text{tính\_từ} \\ 0 & \text{ngược lại} \end{cases}$

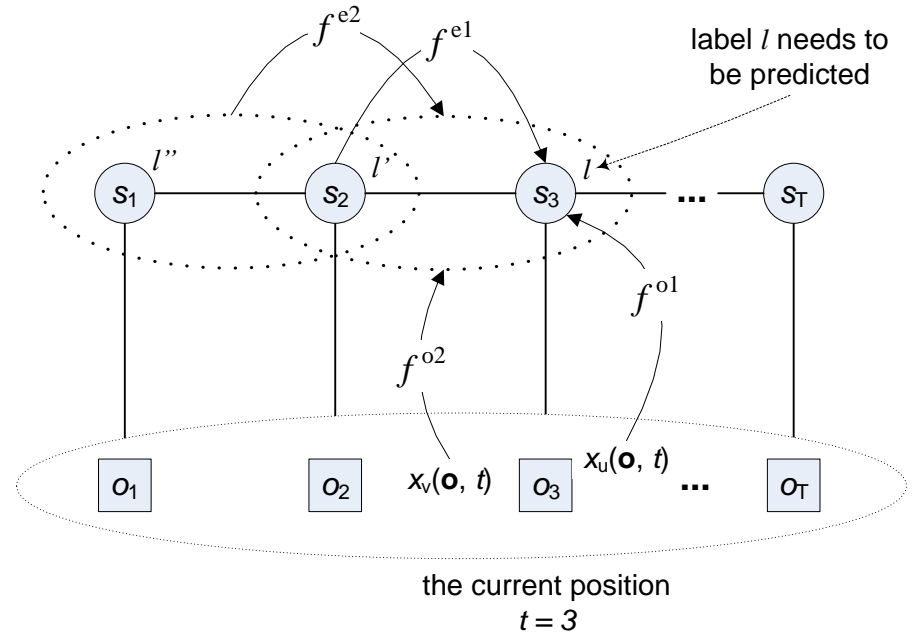
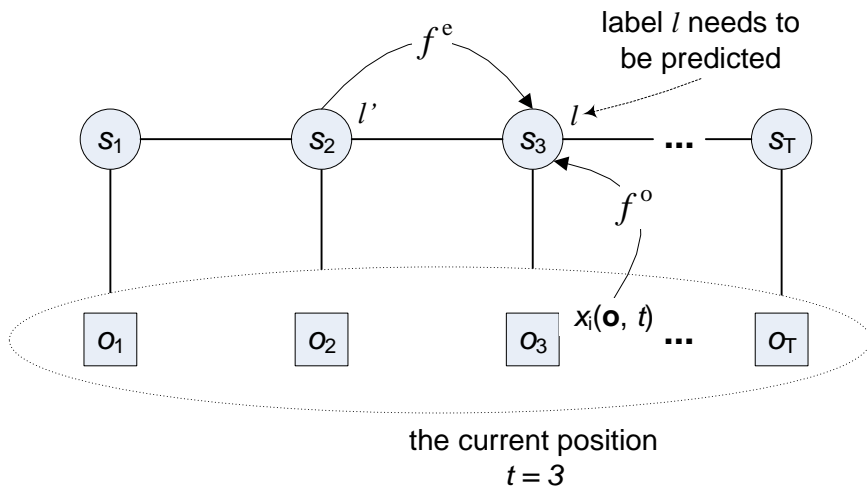
# Trích chọn thuộc tính cho CRFs như thế nào?



- Thống kê đơn lẻ (singleton statistics)
- Kết hợp của các thống kê đơn lẻ (conjunction of singleton statistics)
- Overlapping features
- Bất cứ dạng thuộc tính nào cần thiết cho việc đoán nhận

# CRFs phụ thuộc Markov cấp I và cấp II

## First- and second-order Markov CRFs



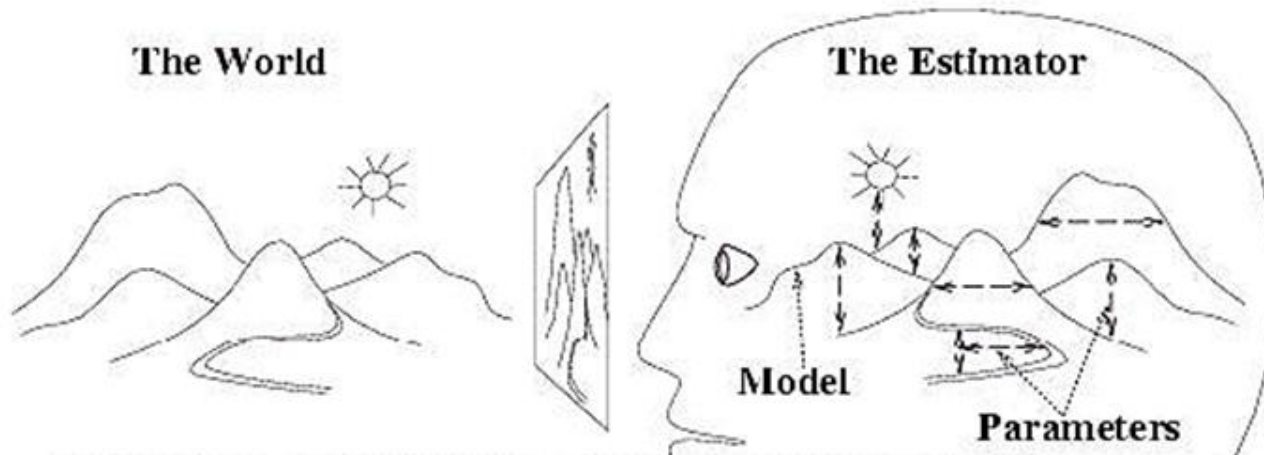
- First-order Markov CRFs
- Two kinds of features:
  - Edge feature  $f^e$
  - Observation feature  $f^o$
- Complexity  $\sim O(|L|^2 T |D| cI)$   
 ( $L$  is the set of class labels)

- Second-order Markov CRFs
- Four kinds of features
  - Edge features:  $f^{e1}, f^{e2}$
  - Observation features:  $f^{o1}, f^{o2}$
- Complexity  $\sim O(|L|^3 T |D| cI)$



**Ước lượng tham số hay huấn luyện mô hình**  
**model training or parameter estimation**

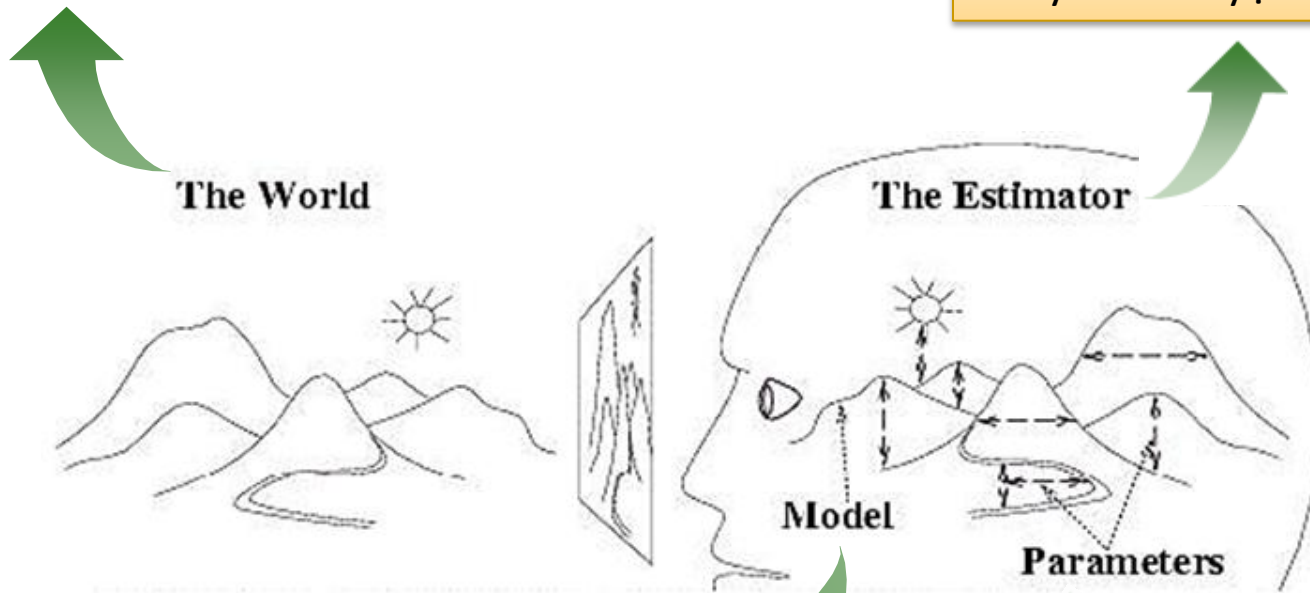
# Mô hình hóa và ước lượng tham số cho các mô hình học máy thống kê



# Mô hình hóa và ước lượng tham số cho CRFs

$$\mathbf{D} = \{(\mathbf{o}^1, \mathbf{s}^1), (\mathbf{o}^2, \mathbf{s}^2), \dots, (\mathbf{o}^m, \mathbf{s}^m)\}$$

Phương pháp ước lượng  
hay huấn luyện mô hình



$$p_{\theta}(\mathbf{s}|\mathbf{o}) = \frac{1}{Z_{\theta}(\mathbf{o})} \exp\left(\sum_{t=1}^T \sum_{i=1}^n \lambda_i f_i(s_{t-1}, s_t, \mathbf{o}, t)\right)$$

$$\theta^* = \{\lambda_1^*, \lambda_2^*, \dots, \lambda_n^*\}$$

# Các phương pháp ước lượng tham số cho CRFs

- Maximum Likelihood
  - Iterative scaling (GIS, IIS)
  - Gradient descent
  - Conjugate gradient
  - Newton's method (limited quasi-Newton method – L-BFGS)
- Stochastic Gradient Methods
- Parallelism

Method	Iterations	LL Evaluations	Time (s)
Algorithm T (IIS)	>150	>150	>188.65
Conjugate gradient (FR)	19	99	124.67
Conjugate gradient (PRP)	27	140	176.55
Limited memory variable metric	22	22	29.72

Efficient Training of Conditional Random Fields (by Hana M. Wallach)

# Ước lượng theo maximum likelihood với L-BFGS

- Hàm likelihood của mô hình CRFs là hàm lồi (convex).
- Thuật toán tối ưu: limited memory quasi-Newton method (L-BFGS):
  - D. Liu and J. Nocedal. On the limited memory BFGS method for large scale optimization, Mathematical Programming, 1989.
  - Là phương pháp tối ưu cấp 2 (second-order), cần ước lượng ma trận Hessian.
  - L-BFGS có thể ước lượng xấp xỉ ma trận Hessian thông qua giá trị hàm mục tiêu và véc tơ đạo hàm cấp 1.
  - Trong nội dung bài giảng: xem L-BFGS như một thủ tục hộp đen (black-box procedure)

$$\mathbf{H} = \begin{bmatrix} \frac{\partial^2 f}{\partial x_1^2} & \frac{\partial^2 f}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_1 \partial x_n} \\ \frac{\partial^2 f}{\partial x_2 \partial x_1} & \frac{\partial^2 f}{\partial x_2^2} & \cdots & \frac{\partial^2 f}{\partial x_2 \partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial x_n \partial x_1} & \frac{\partial^2 f}{\partial x_n \partial x_2} & \cdots & \frac{\partial^2 f}{\partial x_n^2} \end{bmatrix}$$

# Thủ tục huấn luyện mô hình với L-BFGS

---

## Input:

- Training data:  $\mathcal{D} = \{(\mathbf{o}^1, \mathbf{s}^1), (\mathbf{o}^2, \mathbf{s}^2), \dots, (\mathbf{o}^m, \mathbf{s}^m)\}$
- The number of training iterations:  $I$

## Output:

- Optimal feature weights:  $\theta^* = \{\lambda_1^*, \lambda_2^*, \dots, \lambda_n\}$

## Initial Step:

- Generate features with initial weights  $\theta = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$

## Training (each training iteration):

1. compute the log-likelihood function  $L(p_\theta, \mathcal{D})$  and its gradient vector  $\{\frac{\delta L}{\delta \lambda_1}, \frac{\delta L}{\delta \lambda_2}, \dots, \frac{\delta L}{\delta \lambda_n}\}$
  2. perform L-BFGS optimization search to update the new feature weights  $\theta = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$
  3. If #iterations  $< I$  then goto step 1, stop otherwise
-

# Hàm log-likelihood

$$L(p_{\theta}, \mathbf{D}) = \log \prod_{j=1}^m p_{\theta}(\mathbf{s}^j | \mathbf{o}^j) - \sum_{i=1}^n \frac{\lambda_i^2}{2\sigma^2}$$

Euclidean norm regularization:  
giảm overfitting và tránh cân  
bằng các thuộc tính

$$= \sum_{j=1}^m \log(p_{\theta}(\mathbf{s}^j | \mathbf{o}^j)) - \sum_{i=1}^n \frac{\lambda_i^2}{2\sigma^2}$$

$$= \sum_{j=1}^m \left\{ \log \left[ \exp \left( \sum_{t=1}^T \sum_{i=1}^n \lambda_i f_i(s_{t-1}^j, s_t^j, \mathbf{o}^j, t) \right) \right] - \log Z_{\theta}(\mathbf{o}^j) \right\} - \sum_{i=1}^n \frac{\lambda_i^2}{2\sigma^2}$$

$$= \sum_{j=1}^m \sum_{t=1}^T \sum_{i=1}^n \lambda_i f_i(s_{t-1}^j, s_t^j, \mathbf{o}^j, t) - \sum_{j=1}^m \log Z_{\theta}(\mathbf{o}^j) - \sum_{i=1}^n \frac{\lambda_i^2}{2\sigma^2}$$

# Đạo hàm riêng cấp một hàm log-likelihood

$$\begin{aligned}
 & \frac{\vartheta L(p_{\theta}, \mathbf{D})}{\vartheta \lambda_i} \\
 &= \frac{\vartheta \left[ \sum_{j=1}^m \sum_{t=1}^T \sum_{i=1}^n \lambda_i f_i(s_{t-1}^j, s_t^j, \mathbf{o}^j, t) \right]}{\vartheta \lambda_i} - \frac{\vartheta \left[ \sum_{j=1}^m \log Z_{\theta}(\mathbf{o}^j) \right]}{\vartheta \lambda_i} - \frac{\vartheta \left[ \sum_{i=1}^n \frac{\lambda_i^2}{2\sigma^2} \right]}{\vartheta \lambda_i} \\
 &= \sum_{j=1}^m \sum_{t=1}^T f_i(s_{t-1}^j, s_t^j, \mathbf{o}^j, t) - \sum_{j=1}^m \frac{\vartheta \left[ \sum_{\mathbf{s}} \exp \left( \sum_{t=1}^T \sum_{i=1}^n \lambda_i f_i(s_{t-1}, s_t, \mathbf{o}^j, t) \right) \right]}{Z_{\theta}(\mathbf{o}^j)} - \frac{\lambda_i}{\sigma^2} \\
 &= \sum_{j=1}^m \sum_{t=1}^T f_i(s_{t-1}^j, s_t^j, \mathbf{o}^j, t) - \sum_{j=1}^m \sum_{\mathbf{s}} \left[ p_{\theta}(\mathbf{s} | \mathbf{o}^j) \sum_{t=1}^T f_i(s_{t-1}, s_t, \mathbf{o}^j, t) \right] - \frac{\lambda_i}{\sigma^2}
 \end{aligned}$$



# Ước lượng hàm log-likelihood và đạo hàm cấp một

- Để tính:

- Hàm log-likelihood  $L(p_{\theta}, \mathbf{D})$  và

- Véc tơ gradient  $\left\{ \frac{\partial L(p_{\theta}, \mathbf{D})}{\partial \lambda_1}, \frac{\partial L(p_{\theta}, \mathbf{D})}{\partial \lambda_2}, \dots, \frac{\partial L(p_{\theta}, \mathbf{D})}{\partial \lambda_n} \right\}$

- Cần tính:

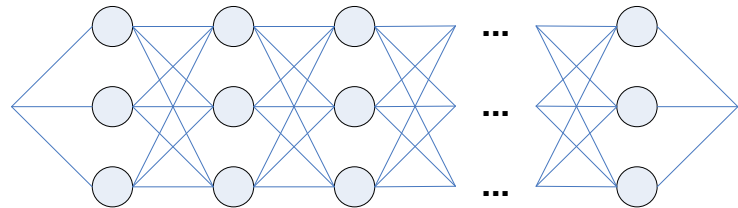
$$Z_{\theta}(\mathbf{o}^j) = \sum_{\mathbf{s}} \exp \left( \sum_{t=1}^T \sum_{i=1}^n \lambda_i f_i(s_{t-1}, s_t, \mathbf{o}^j, t) \right)$$

Dynamic programming

$\mathcal{O}(|L|^2 T)$ : first-order

$\mathcal{O}(|L|^3 T)$ : second-order

( $L$  is the set of labels)



**Summing over all possible label paths**

# Parallelism – Huấn luyện CRFs song song

---

## Input:

- Training data:  $\mathcal{D} = \{(\mathbf{o}^1, \mathbf{s}^1), (\mathbf{o}^2, \mathbf{s}^2), \dots, (\mathbf{o}^m, \mathbf{s}^m)\}$
- The number of parallel processes:  $P$ ;
- The number of training iterations:  $I$

## Output:

- Optimal feature weights:  $\theta^* = \{\lambda_1^*, \lambda_2^*, \dots, \lambda_n^*\}$

## Initial Step:

- Generate features with initial weights  $\theta = \{\lambda_1, \lambda_2, \dots, \lambda_n\}$
- Each process loads its own data partition  $\mathcal{D}_i$

## Parallel Training (each training iteration):

1. The root process broadcasts  $\theta$  to all parallel processes
  2. Each process  $P_i$  computes the local log-likelihood  $L_i$  and local gradient vector  $\{\frac{\delta L}{\delta \lambda_1}, \frac{\delta L}{\delta \lambda_2}, \dots, \frac{\delta L}{\delta \lambda_n}\}_i$  on  $\mathcal{D}_i$
  3. The root process gathers and sums all  $L_i$  and  $\{\frac{\delta L}{\delta \lambda_1}, \frac{\delta L}{\delta \lambda_2}, \dots\}_i$  to obtain the global  $L$  and  $\{\frac{\delta L}{\delta \lambda_1}, \frac{\delta L}{\delta \lambda_2}, \dots, \frac{\delta L}{\delta \lambda_n}\}$
  4. The root process performs L-BFGS optimization search to update the new feature weights  $\theta$
  5. If #iterations  $< I$  then goto step 1, stop otherwise
-

**Suy diễn hay đoán nhận với mô hình CRFs**  
**inference or prediction in linear-chain CRFs**

# Đoán nhận chuỗi nhãn phù hợp với mô hình CRFs

- Bộ tham số tối ưu và mô hình tương ứng:

$$\boldsymbol{\theta}^* = \{\lambda_1^*, \lambda_2^*, \dots, \lambda_n^*\}$$

$$p_{\boldsymbol{\theta}^*}(\mathbf{s}|\mathbf{o}) = \frac{1}{Z_{\boldsymbol{\theta}^*}(\mathbf{o})} \exp\left(\sum_{t=1}^T \sum_{i=1}^n \lambda_i^* f_i(s_{t-1}, s_t, \mathbf{o}, t)\right)$$

- Sử dụng mô hình này để đoán nhận (predict) chuỗi nhãn phù hợp nhất (most likely label sequence)  $\mathbf{s}^*$  cho mỗi chuỗi dữ liệu đầu vào (input data observation sequence)  $\mathbf{o}$  như sau:

$$\begin{aligned} \mathbf{s}^* &= \operatorname{argmax}_{\mathbf{s}} p_{\boldsymbol{\theta}^*}(\mathbf{s}|\mathbf{o}) = \operatorname{argmax}_{\mathbf{s}} \frac{1}{Z_{\boldsymbol{\theta}^*}(\mathbf{o})} \exp\left(\sum_{t=1}^T \sum_{i=1}^n \lambda_i^* f_i(s_{t-1}, s_t, \mathbf{o}, t)\right) \\ &= \operatorname{argmax}_{\mathbf{s}} \left[ \exp\left(\sum_{t=1}^T \sum_{i=1}^n \lambda_i^* f_i(s_{t-1}, s_t, \mathbf{o}, t)\right) \right] \end{aligned}$$

# Tìm chuỗi nhãn phù hợp với thuật toán Viterbi

- Thuật toán quy hoạch động, sử dụng rộng rãi với mô hình HMMs
- Lưu lại xác suất của *chuỗi nhãn phù hợp nhất* từ đầu cho đến vị trí  $t$  nào đó trong chuỗi và kết thúc với trạng thái  $s_t^u$  (hàm ý  $s_t^u = l_u \in \mathbf{L}$ )  
Ký hiệu xác suất đó là  $\varphi_t(s_t^u)$  ( $0 \leq t \leq T - 1$ ) và  $\varphi_0(s_0^u)$  là xác suất xuất phát của chuỗi.
- Chúng ta có công thức đệ quy như sau:

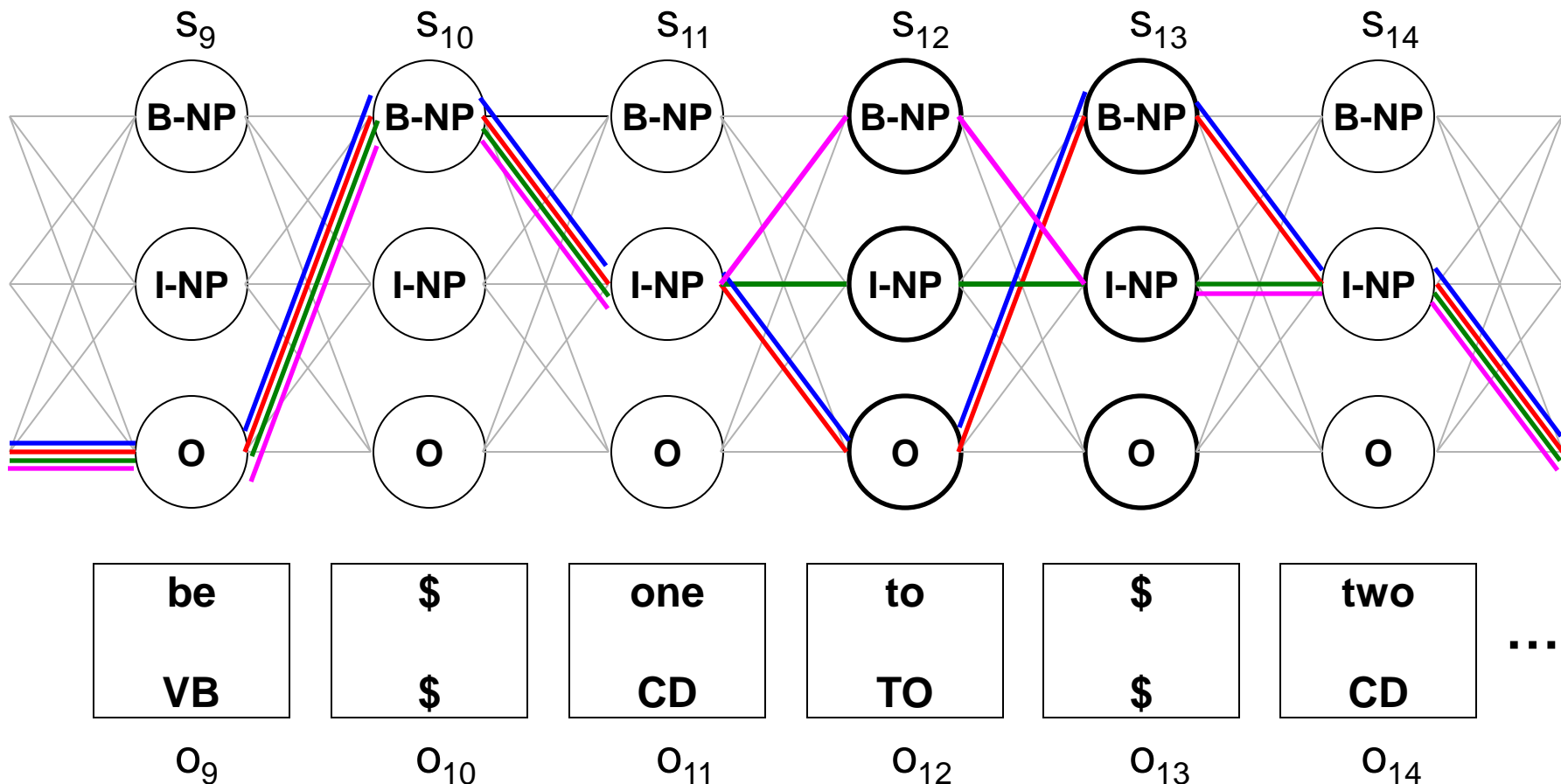
$$\varphi_{t+1}(s_{t+1}^v) = \max_{s_t^u} \left[ \varphi_t(s_t^u) \exp \left( \sum_{i=1}^n \lambda_i^* f_i(s_t^u, s_{t+1}^v, \mathbf{o}, t) \right) \right]$$

- Việc tính toán đệ quy kết thúc khi  $t = T - 1$  và xác suất lớn nhất (chưa chuẩn hóa) là

$$p^* = \max_u [\varphi_T(s_T^u)]$$
$$l_u^* = \operatorname{argmax}_u [\varphi_T(s_T^u)]$$

Backtracking dựa trên các thông tin đã lưu để tìm chuỗi nhãn phù hợp nhất

# Tìm $k$ chuỗi nhãn tốt nhất ( $k$ -best label sequences)

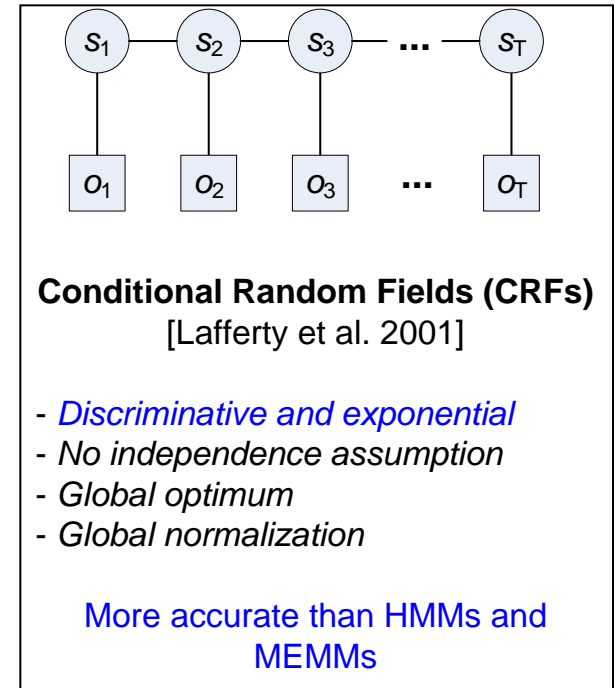
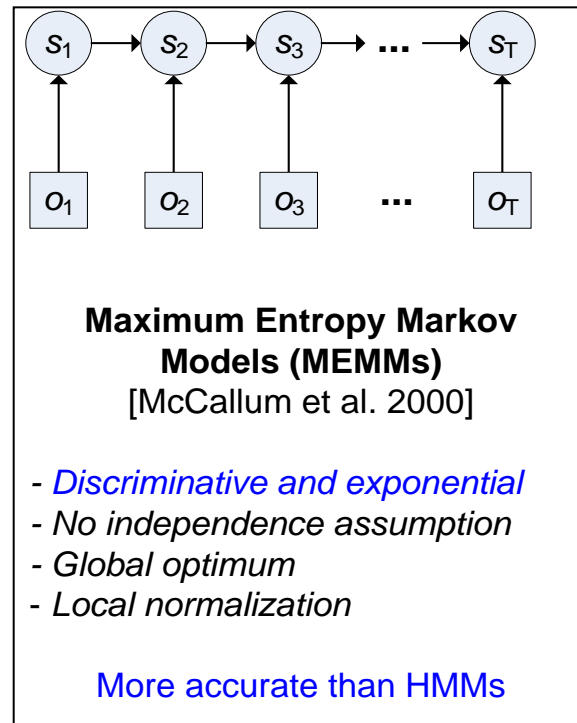
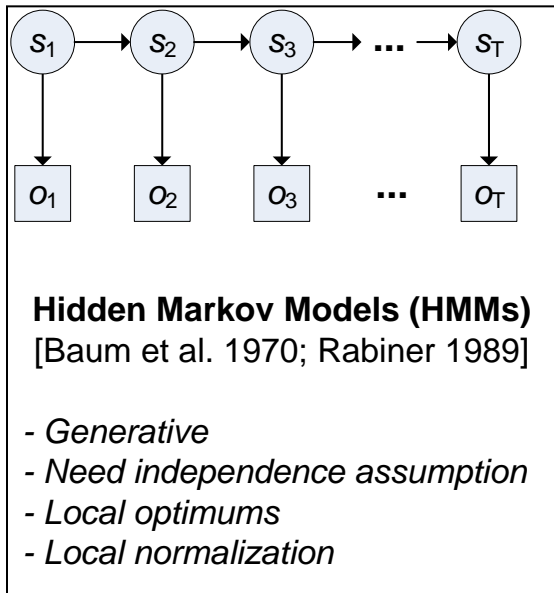


— Path<sub>1</sub> (the best path) — Path<sub>2</sub> — Path<sub>3</sub> ... — Path<sub>n</sub>

# **Các ưu và nhược điểm của mô hình CRFs**

## **advantages and limitations of CRFs**

# Ưu điểm của CRFs đối với HMMs và MEMMs

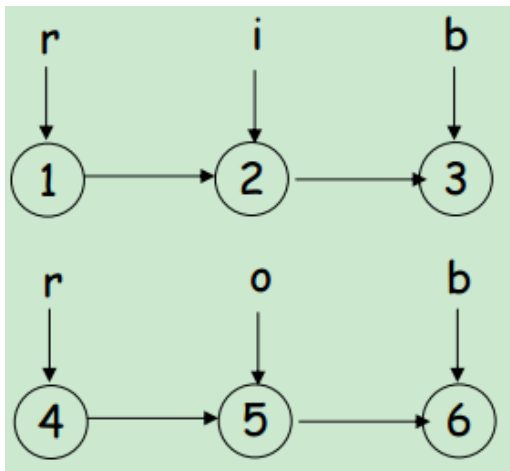




# Giải quyết được vấn đề “label bias”

- Maximum Entropy Markov Models (MEMMs)

$$p(\mathbf{s}|\mathbf{o}) = p(s_1|o_1) \prod_{t=2}^T p(s_t|s_{t-1}, o_t)$$



## Training Data

**X:Y**

rib:123

rib:123

rib:123

rob:456

rob:456

## Parameters

$$P(1|r) = 0.6, P(4|r) = 0.4,$$

$$P(2|i,1) = P(2|o,1) = 1,$$

$$P(5|i,4) = P(5|o,4) = 1,$$

$$P(3|b,2) = P(6|b,5) = 1$$

$$\begin{aligned} P(123|\mathbf{rob}) &= P(1|r)P(2|o,1)P(3|b,2) \\ &= 0.6 \times 1 \times 1 = 0.6 \end{aligned}$$

$$\begin{aligned} P(456|\mathbf{rob}) &= P(4|r)P(5|o,4)P(6|b,5) \\ &= 0.4 \times 1 \times 1 = 0.4 \end{aligned}$$

# Nhược điểm

- Tính phụ thuộc Markov trong dữ liệu (bản chất dữ liệu)
- Thời gian tính toán (huấn luyện) tăng khi số nhãn nhiều.
- Mô hình có thể lớn (tốn bộ nhớ)

# **CRFs với các bài toán ứng dụng thực tế**

## **CRFs and its real-world applications**

# CRFs for Image Processing and Computer Vision

- K. Murphy et al. *Using The Forest to See The Trees: A Graphical Model Relating Features, Objects, and Scenes*, NIPS 2003.
- S. Kumar and M. Hebert. *Discriminative Fields for Modeling Spatial Dependencies in Natural Images*, NIPS 2003.
- X. He et al. *Multiscale Conditional Random Fields for Image Labeling*, CVPR 2004.
- C. Smimchisescu et al. *Conditional Models for Contextual Human Motion Recognition*, ICCV 2005.
- A. Quattoni et al. *Conditional Random Fields for Object Recognition*, NIPS 2005.
- A. Torralba et al. *Contextual Models for Object Detection using Boosted Random Fields*, NIPS 2005.
- Y. Wang and Q. Ji. *A Dynamic Conditional Random Field Model for Object Segmentation in Image Sequences*, CVPR 2005.

# CRFs for Bioinformatics & Computational Biology

- K. Sato and Y. Sakakibara. *RNA secondary structural alignment with conditional random fields*, Bioinformatics 2005.
- Y. Liu et al. *Protein fold recognition using segmentation conditional random fields*, Journal of Computational Biology 2006.
- M. Li et al. *Protein-protein interaction site prediction based on conditional random fields*, Bioinformatics 2007.
- F. Zhao et al. *A probabilistic graphical model for ab initio folding*, *Research in Computational Molecular Biology*, 2009.
- X. Geng et al. *Protein backbone dihedral angle prediction based on probabilistic models*, iCBBE 2010.
- T. Gehrman et al. *Conditional random fields for protein function prediction*, PRIB 2013.

# CRFs for NLP and Information Extraction

- Word segmentation
- Part-of-speech tagging
- Phrase chunking (shallow parsing)
- Named entity recognition (NER)
  - Both general text and biomedical text
- Information extraction from text/web
  - Text: table, author-affiliation (research papers), for form filling, ...
  - Web: product description, transforming semi-structured into structured data

# Một số bài toán ứng dụng CRFs

- Gắn nhãn (labeling):
  - Gắn nhãn từ loại (part-of-speech tagging) cho tiếng Anh
- Phân đoạn (segmentation):
  - Xác định cụm danh từ (noun phrase chunking) cho tiếng Anh
  - Xác định cụm từ (phrase chunking/shallow parsing) cho tiếng Anh
  - Tách từ (word segmentation) cho tiếng Việt

# Flexible Conditional Random Fields

## FlexCRFs: Flexible Conditional Random Fields

Copyright © 2004-2005 by [Xuan-Hieu Phan](#), [Le-Minh Nguyen](#), and [Cam-Tu Nguyen](#)

[About](#)[Download](#)[Documents](#)[Casestudy](#)[Citations](#)[References](#)

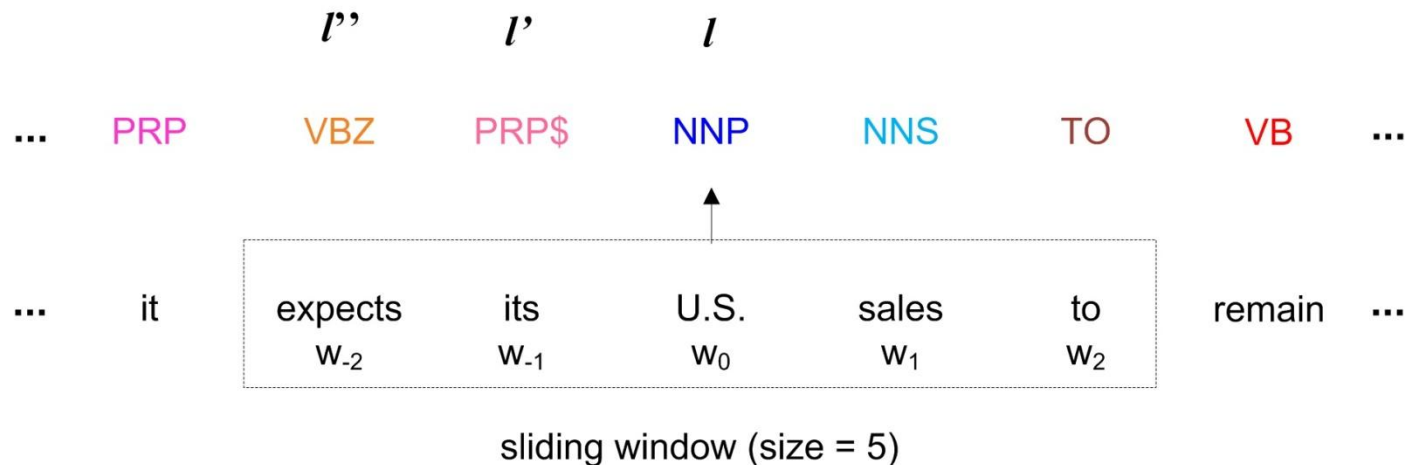
**FlexCRFs** is a conditional random field toolkit for segmenting and labeling sequence data written in C/C++ using STL library. It was implemented based on the theoretic model presented in (Lafferty et al. 2001) and (Sha and Pereira 2003). The toolkit uses L-BFGS (Liu and Nocedal 1989) - an advanced convex optimization procedure - to train CRF models. FlexCRFs was designed to deal with hundreds of thousand data sequences and millions of features. FlexCRFs supports both first-order and second-order Markov CRFs. We have tested FlexCRFs on Linux (Red Hat, Fedora, Ubuntu), Sun Solaris, and MS Windows with MS Visual C++.

**PCRFS** is a parallel version of FlexCRFs that allows us to train conditional random fields on massively parallel processing systems supporting Message Passing Interface (MPI). PCRFS helps to train conditional random fields on large-scale datasets containing up to millions of data sequences. We have tested PCRFS on large parallel systems, such as Cray XT3, SGI Altix, and IBM SP.



# Part-of-Speech Tagging on WSJ Corpus

*Rolls\_NNP Royce\_NNP Motor\_NNP Cars\_NNPS Inc.\_NNP said\_VBD it\_PRP  
expects\_VBZ its\_PRP\$ U.S.\_NNP sales\_NNS to\_TO remain\_VB steady\_JJ ...*



- WSJ Corpus:
  - Training set: sections 00-18
  - Development test set: sections 19-21
  - Final test set: sections 22-24
- First and second-order Markov CRFs

# Feature Templates for Part-of-Speech Tagging

Template for edge feature type 1 (i.e., $f^{e1}$ )		
	state $s_{t-1}$	current state $s_t$
	$l'$	$l$
Template for edge feature type 2 (i.e., $f^{e2}$ )		
state $s_{t-2}$	state $s_{t-1}$	current state $s_t$
$l''$	$l'$	$l$
Template for observation feature type 1 (i.e., $f^{o1}$ )		
context predicate templates $x_u(\mathbf{o}, t)$		current state $s_t$
$w_{-2}, w_{-1}, w_0, w_1, w_2, w_{-1}w_0, w_0w_1$		$l$
1, 2, 3, 4-character prefixes and suffixes of $w_0$		$l$
$w_0$ is initially capitalized		$l$
$w_0$ is all capitalized		$l$
$w_0$ has number; $w_0$ has hyphen		$l$
Template for observation feature type 2 (i.e., $f^{o2}$ )		
context predicate templates $x_v(\mathbf{o}, t)$	state $s_{t-1}$	current state $s_t$
$w_{-1}, w_0, w_{-1}w_0$	$l'$	$l$
$w_0$ is initially capitalized	$l'$	$l$
$w_0$ is all capitalized	$l'$	$l$
$w_0$ has number; $w_0$ has hyphen	$l'$	$l$

# Part-of-Speech Tagging Comparison on WSJ

Methods	Devel. Acc.%	Final Acc.%
Toutanova et al. 2003 (Dependency Network, 4-order Markov dependencies, 2-forward, 2-backward)	97.15	97.24
<b>Ours</b> (second-order Markov CRFs)	<b>97.05</b>	<b>97.16</b>
Collins 2002 (Discriminative HMMs)	97.07	97.11
<b>Ours</b> (first-order Markov CRFs)	<b>96.92</b>	<b>96.92</b>

SOURCEFORGE.NET®

## CRFTagger: CRF English POS Tagger

(built upon [FlexCRFs](#))

URL: <http://crftagger.sourceforge.net/>

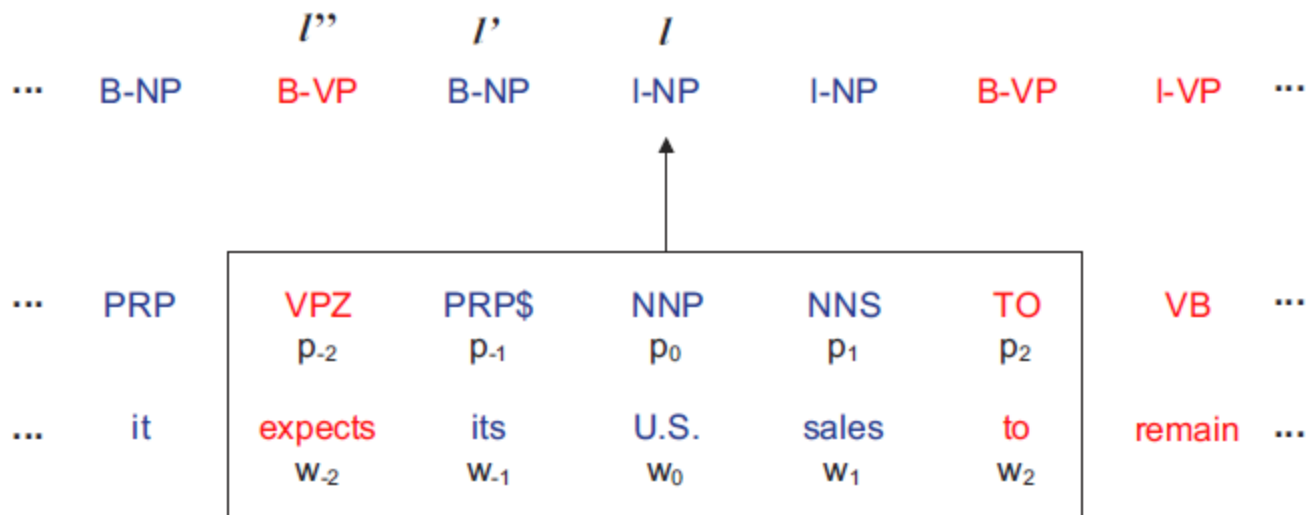
Copyright (c) 2006 by

[Xuan-Hieu Phan](#) (pxhieu at gmail dot com), Graduate School of Information Sciences, Tohoku University

# Phrase Chunking on Wall Street Journal Corpus

Rolls\_NNP Royce\_NNP Motor\_NNP Cars\_NNPS Inc.\_NNP said\_VBD it\_PRP  
 expects\_VBZ its\_PRP\$ U.S.\_NNP sales\_NNS to\_TO remain\_VB steady\_JJ ...

[Rolls Royce Motor Cars Inc. NP] [said VP] [it NP] [expects VP]  
 [its U.S. sales NP] [to remain VP] [steady ADJP] ...



sliding window (size = 5)

# Label Sequence Representation for Segmentation

Input observation sequence		Output label sequence			
Sentence	POS tag	IOB2	IOB1	IOE2	IOE1
Confidence	NN	B-NP	I-NP	E-NP	I-NP
in	IN	B-PP	I-PP	E-PP	I-PP
the	DT	B-NP	I-NP	I-NP	I-NP
pound	NN	I-NP	I-NP	E-NP	I-NP
is	VBZ	B-VP	I-VP	I-VP	I-VP
widely	RB	I-VP	I-VP	I-VP	I-VP
expected	VBN	I-VP	I-VP	I-VP	I-VP
to	TO	I-VP	I-VP	I-VP	I-VP
take	VB	I-VP	I-VP	E-VP	I-VP
another	DT	B-NP	I-NP	I-NP	I-NP
sharp	JJ	I-NP	I-NP	I-NP	I-NP
dive	NN	I-NP	I-NP	E-NP	I-NP
if	IN	B-SBAR	I-SBAR	E-SBAR	I-SBAR
trade	NN	B-NP	I-NP	I-NP	I-NP
figures	NNS	I-NP	I-NP	E-NP	I-NP
for	IN	B-PP	I-PP	E-PP	I-PP
September	NNP	B-NP	I-NP	E-NP	I-NP
,	,	O	O	O	O
due	JJ	B-ADJP	I-ADJP	E-ADJP	I-ADJP
for	IN	B-PP	I-PP	E-PP	I-PP
release	NN	B-NP	I-NP	E-NP	E-NP
tomorrow	NN	B-NP	B-NP	E-NP	I-NP
,	,	O	O	O	O
fail	VB	B-VP	I-VP	I-VP	I-VP
to	TO	I-VP	I-VP	I-VP	I-VP
show	VB	I-VP	I-VP	E-VP	I-VP
a	DT	B-NP	I-NP	I-NP	I-NP
substantial	JJ	I-NP	I-NP	I-NP	I-NP
improvement	NN	I-NP	I-NP	E-NP	I-NP
from	IN	B-PP	I-PP	E-PP	I-PP
July	NNP	B-NP	I-NP	I-NP	I-NP
and	CC	I-NP	I-NP	I-NP	I-NP
August	NNP	I-NP	I-NP	E-NP	E-NP
's	POS	B-NP	B-NP	I-NP	I-NP
near-record	JJ	I-NP	I-NP	I-NP	I-NP
deficits	NNS	I-NP	I-NP	E-NP	I-NP
.	.	O	O	O	O

# Feature Templates for Phrase Chunking

Template for edge feature type 1 (i.e., $f^{e1}$ )		
	state $s_{t-1}$	current state $s_t$
	$l'$	$l$
Template for edge feature type 2 (i.e., $f^{e2}$ )		
state $s_{t-2}$	state $s_{t-1}$	current state $s_t$
$l''$	$l'$	$l$
Template for observation feature type 1 (i.e., $f^{o1}$ )		
context predicate templates $x_u(\mathbf{o}, t)$		current state $s_t$
$w_{-2}, w_{-1}, w_0, w_1, w_2, w_{-1}w_0, w_0w_1$		$l$
$p_{-2}, p_{-1}, p_0, p_1, p_2$		$l$
$p_{-2}p_{-1}, p_{-1}p_0, p_0p_1, p_1p_2$		$l$
$p_{-2}p_{-1}p_0, p_{-1}p_0p_1, p_0p_1p_2, p_{-1}w_{-1}, p_0w_0$		$l$
$p_{-1}p_0w_{-1}, p_{-1}p_0w_0, p_{-1}w_{-1}w_0, p_0w_{-1}w_0, p_{-1}p_0p_1w_0$		$l$
Template for observation feature type 2 (i.e., $f^{o2}$ )		
context predicate templates $x_v(\mathbf{o}, t)$	state $s_{t-1}$	current state $s_t$
$w_{-1}, w_0, w_{-1}w_0$	$l'$	$l$
$p_{-1}, p_0, p_{-1}p_0, p_{-1}w_{-1}, p_0w_0$	$l'$	$l$
$p_{-1}p_0w_{-1}, p_{-1}p_0w_0, p_{-1}w_{-1}w_0, p_0w_{-1}w_0$	$l'$	$l$

# Chunking Results on the CoNLL2000 Shared Task

	NP chunking						All-phrase chunking					
Init	IOB2, #feat: 417,831			IOE2, #feat: 416,262			IOB2, #feat: 451,300			IOE2, #feat: 450,063		
$\theta$	Pre.	Rec.	$F_{\beta=1}$	Pre.	Rec.	$F_{\beta=1}$	Pre.	Rec.	$F_{\beta=1}$	Pre.	Rec.	$F_{\beta=1}$
.00	94.57	94.26	94.42	94.49	94.45	94.47	93.94	93.90	93.92	93.94	93.98	93.96
.01	94.60	94.39	94.50	94.50	94.29	94.39	93.95	93.92	93.93	94.03	93.94	93.98
.02	94.48	94.33	94.40	94.52	94.35	94.44	93.91	93.90	93.91	93.91	94.05	93.98
.03	94.70	94.41	94.56	94.48	94.34	94.41	93.93	93.99	93.96	93.96	93.94	93.95
.04	94.65	94.22	94.43	94.37	94.39	94.38	93.98	93.87	93.92	93.99	93.98	93.98
.05	94.70	94.44	<b>94.57</b>	94.43	94.36	94.39	93.93	93.96	93.95	94.11	93.98	<b>94.05</b>
.06	94.70	94.21	94.46	94.39	94.41	94.40	94.05	93.83	93.94	94.08	93.94	94.01
.07	94.73	94.28	94.50	94.53	94.37	94.45	94.06	93.85	93.96	94.02	94.04	94.03

Table 5.5: Results of NP chunking and chunking with different initial values ( $\theta$ ) of feature weights on the CoNLL2000 shared task (training: sections 15-18, testing: section 20 of WSJ)

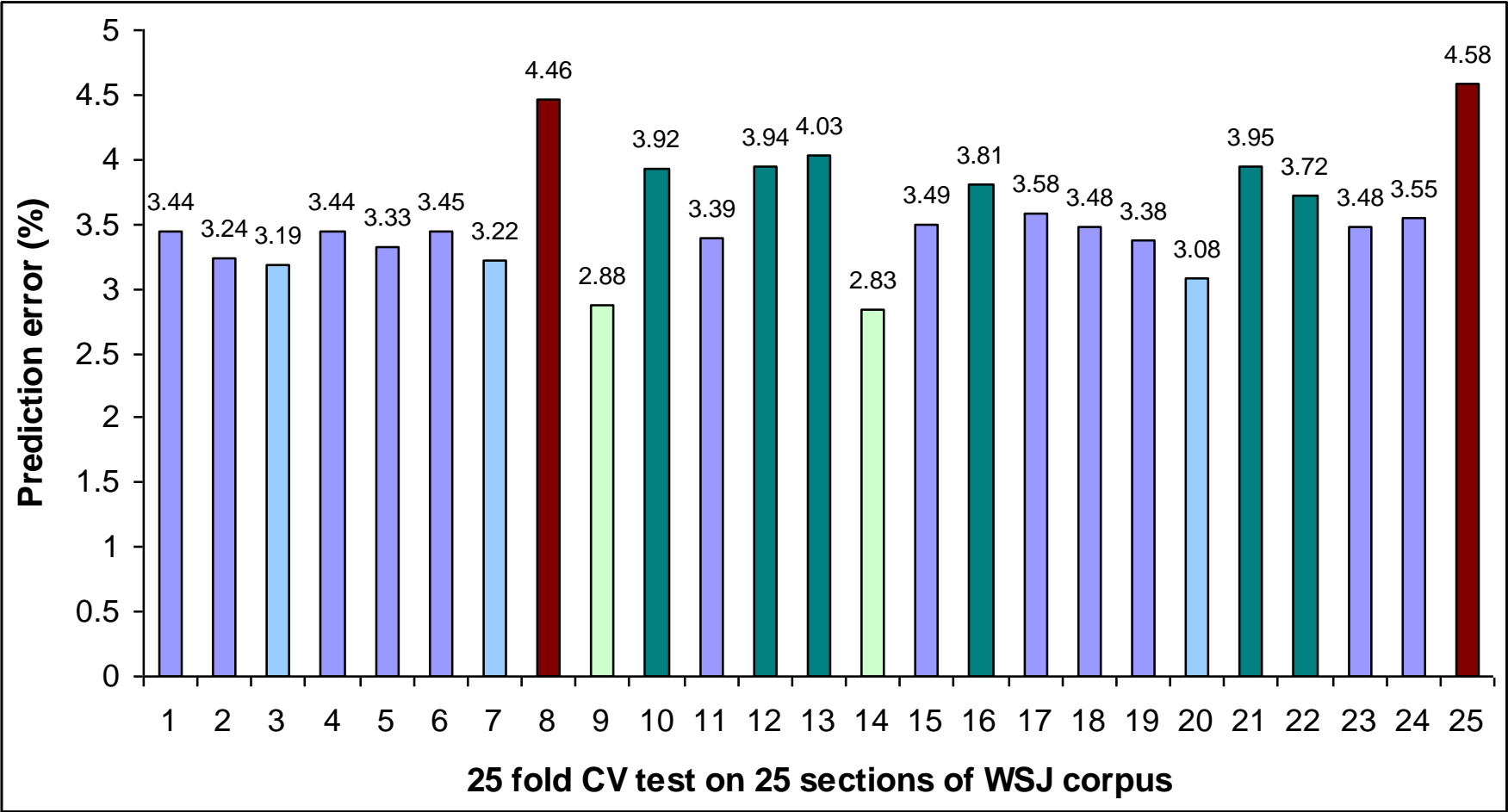
# Chunking Results on the CoNLL2000-L

Init $\theta$	NP chunking						All-phrase chunking					
	IOB2, #feat: 1,351,627			IOE2, #feat: 1,350,514			IOB2, #feat: 1,471,004			IOE2, #feat: 1,466,312		
	Pre.	Rec.	$F_{\beta=1}$	Pre.	Rec.	$F_{\beta=1}$	Pre.	Rec.	$F_{\beta=1}$	Pre.	Rec.	$F_{\beta=1}$
.00	96.54	96.37	96.45	96.49	96.37	96.43	96.09	96.04	96.06	96.10	96.10	96.10
.01	96.50	96.32	96.41	96.51	96.44	96.48	96.09	96.04	96.06	96.12	96.09	96.11
.02	96.63	96.31	96.47	96.59	96.36	96.47	96.11	96.10	96.10	96.19	96.09	96.14
.03	96.53	96.31	96.42	96.50	96.44	96.47	96.09	96.01	96.05	96.13	96.08	96.11
.04	96.67	96.35	96.51	96.57	96.33	96.45	96.07	95.98	96.03	96.16	96.04	96.10
.05	96.59	96.29	96.44	96.63	96.55	<b>96.59</b>	96.12	96.01	96.07	96.13	96.04	96.09
.06	96.54	96.40	96.47	96.72	96.43	96.58	96.10	96.00	96.05	96.20	97.17	<b>96.18</b>
.07	96.59	96.33	96.46	96.49	96.54	96.51	96.03	96.07	96.05	96.12	96.17	96.15

Table 5.6: Results of NP chunking and chunking with different initial values ( $\theta$ ) of feature weights on the CoNLL2000-L (training: sections 02-21, testing: section 00 of WSJ)



# 25-fold CV Tests of NP Chunking (25 Sections WSJ)

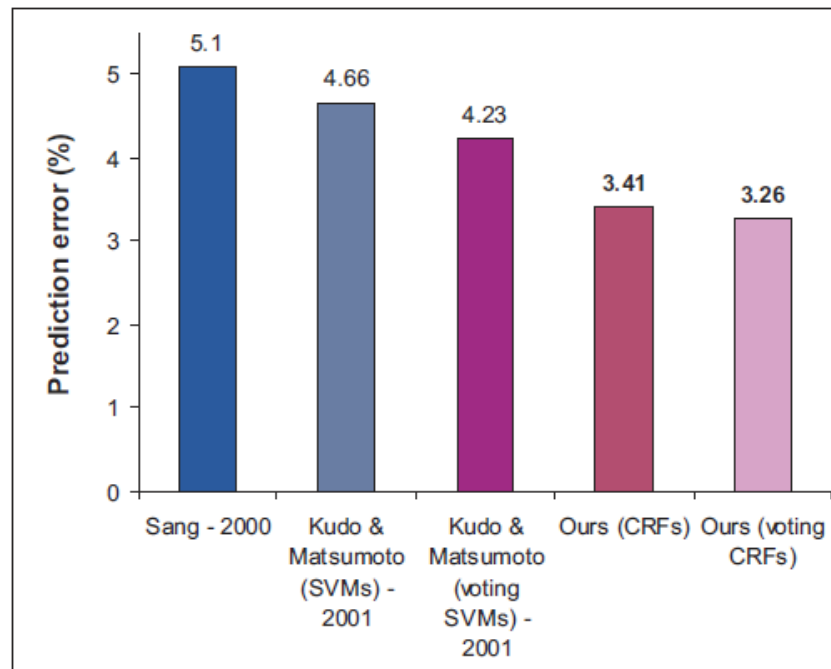


# Accuracy Comparison on CoNLL2000

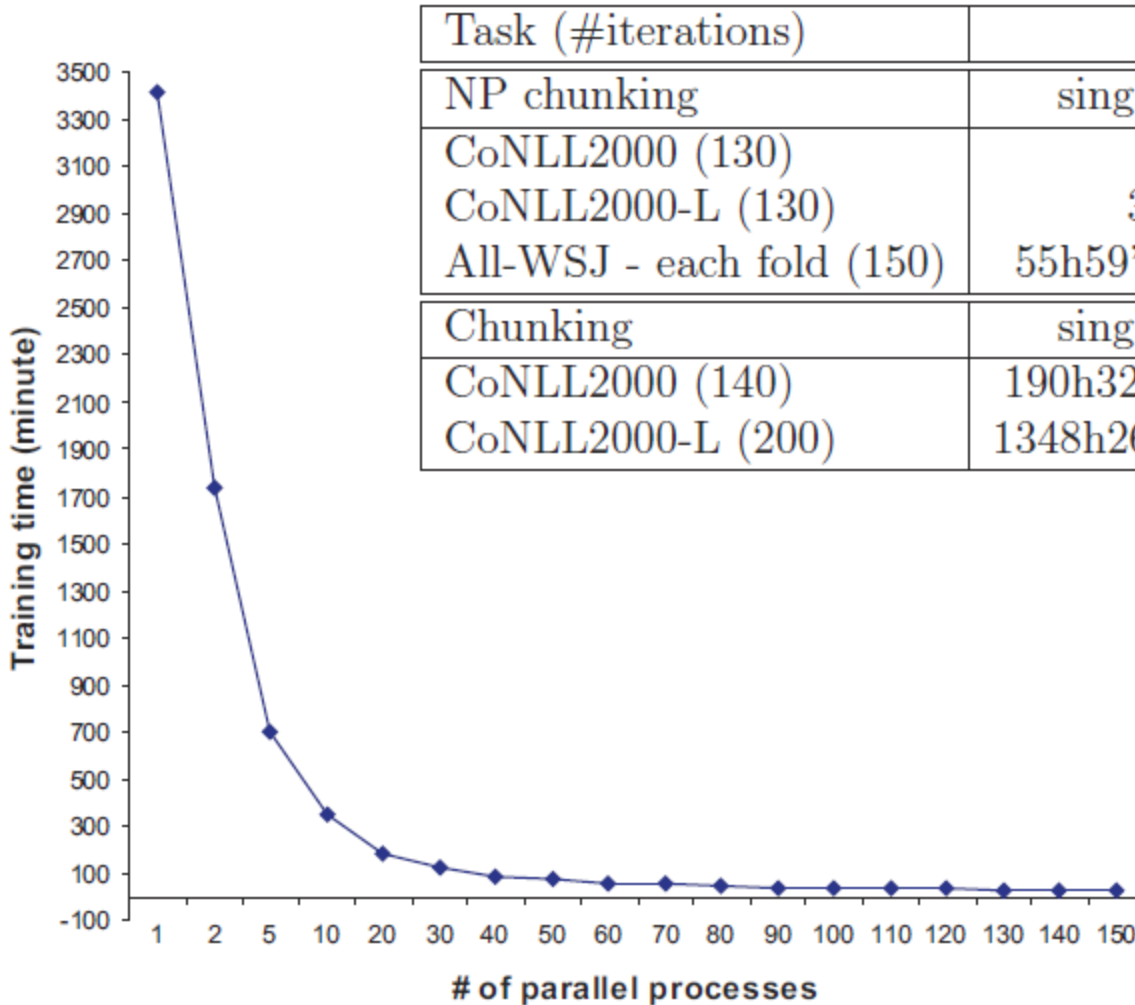
Methods	F <sub>1</sub> (NP)	F <sub>1</sub> (All)
Daumé III & Marcu 2005 (LaSo + external lists of named entities, etc.)	-	94.4x
Ando & Zhang (2005) + 15 millions unlabeled words	94.70	94.39
<b>Ours</b> (simple majority voting among 16 CRFs)	<b>94.73</b>	<b>94.15</b>
<b>Ours</b> (single CRF with 417,831 features)	<b>94.57</b>	<b>94.05</b>
Carreras & Marquez (2003) (perceptron with two layers)	94.41	93.74
Kudo & Matsumoto (2001) (voting among 8 SVMs)	94.39	93.91
Kudo & Matsumoto (2001) (SVMs)	94.11	93.85
Sha & Pereira (2003) (single CRF with 3 million features)	94.38	-
Zhang et al. (2002) (generalized winnow + full parser)	94.38	94.17
Zhang et al. (2002) (generalized winnow)	93.89	93.57

# Accuracy Comparison on CoNLL2000-L

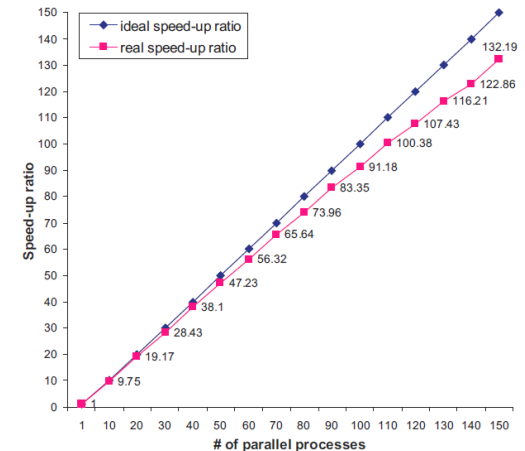
Methods	NP $F_{\beta=1}$	All $F_{\beta=1}$
<b>Ours (majority voting among 16 CRFs)</b>	<b>96.74</b>	<b>96.33</b>
<b>Ours (CRFs, about 1.3M - 1.5M features)</b>	<b>96.59</b>	<b>96.18</b>
Kudo & Matsumoto 2001 (voting SVMs)	95.77	–
Kudo & Matsumoto 2001 (SVMs)	95.34	–
Sang 2000 (system combination)	94.90	–



# Computational Time of Parallel Training



Task (#iterations)	Training time	
NP chunking	single process	45 processes
CoNLL2000 (130)	4h50'	6'52"
CoNLL2000-L (130)	38h57'	56'
All-WSJ - each fold (150)	55h59'(estimated)	1h21'
Chunking	single process	90 processes
CoNLL2000 (140)	190h32'(estimated)	2h29'
CoNLL2000-L (200)	1348h26'(estimated)	17h46'



# Phrase Chunker for English

SOURCEFORGE.NET®

## CRFChunker: CRF English Phrase Chunker

(built upon [FlexCRFs](#))

URL: <http://crfchunker.sourceforge.net/>

Copyright (c) 2006 by

[Xuan-Hieu Phan](#) (pxhieu at gmail dot com), Graduate School of Information Sciences, Tohoku University

---

**CRFChunker:** A Java-based Conditional Random Fields Phrase Chunker (Phrase Chunking Tool) for English that was built upon [FlexCRFs](#). The model was trained on sections 01..24 of WSJ corpus and using section 00 as the development test set (F1-score of 95.77). Chunking speed: 700 sentences / second.

# Tách từ cho tiếng Việt

- Evaluation data: VLSP Corpus ++
- Feature templates: very rich, a lot of regular expressions for time, email, url, currency, number, name, etc.
- Parameters:
  - First-order Markov CRFs
  - Training sentences: 46160
  - Test sentences: 11533
  - #Context predicates: 567149
  - #Features: 659933 (after pruning)
  - Feature rare threshold: 1
  - Context predicate rare threshold: 2
  - #Training iterations: 200
  - Sigma square: 10.0
  - Number of approximated Hessian matrixes: 7

Iteration: 194

Log-likelihood = -10308.109632  
Norm(log-likelihood gradient vector) = 2197.104237  
Norm(lambda vector) = 282.873801

Iteration elapsed: 233 seconds

Label-based performance evaluation:

Label	Manual	Model	Match	Pre. (%)	Rec. (%)	F1-Measure (%)
B	247080	247202	246115	99.56	99.61	99.58
I	71670	71542	70583	98.66	98.48	98.57
O	41704	41710	41704	99.99	100.00	99.99
Avg1.			99.40	99.36	99.38	
Avg2.	360454	360454	358402	99.43	99.43	99.43

Chunk-based performance evaluation:

Chunk	Manual	Model	Match	Pre. (%)	Rec. (%)	F1-Measure (%)
Word	247080	247202	244316	98.83	98.88	98.86
Avg1.			98.83	98.88	98.86	
Avg2.	247080	247202	244316	98.83	98.88	98.86

Current ~~max~~ chunk-based F1: 98.86 (iteration 194)







Training iteration elapsed (including testing & evaluation time): 237 seconds

# Một số cài đặt của mô hình CRFs

## implementation of CRFs



This is a partial list of software that implement generic CRF tools.

- [RNNSharp](#)  CRFs based on recurrent neural networks ([C#](#), [.NET](#))
- [CRF-ADF](#)  Linear-chain CRFs with fast online ADF training ([C#](#), [.NET](#))
- [CRFSharp](#)  Linear-chain CRFs ([C#](#), [.NET](#))
- [GCO](#)  CRFs with submodular energy functions ([C++](#), [Matlab](#))
- [DGM](#)  General CRFs ([C++](#))
- [GRMM](#)  General CRFs ([Java](#))
- [factorie](#)  General CRFs ([Scala](#))
- [CRFall](#)  General CRFs ([Matlab](#))
- [Sarawagi's CRF](#)  Linear-chain CRFs ([Java](#))
- [HCRF library](#)  Hidden-state CRFs ([C++](#), [Matlab](#))
- [Accord.NET](#)  Linear-chain CRF, HCRF and HMMs ([C#](#), [.NET](#))
- [Wapiti](#)  Fast linear-chain CRFs ([C](#))<sup>[10]</sup>
- [CRFSuite](#)  Fast restricted linear-chain CRFs ([C](#))
- [CRF++](#)  Linear-chain CRFs ([C++](#))
- [FlexCRFs](#)  First-order and second-order Markov CRFs ([C++](#))
- [crf-chain1](#)  First-order, linear-chain CRFs ([Haskell](#))
- [imageCRF](#)  CRF for segmenting images and image volumes ([C++](#))
- [MALLET](#)  Linear-chain for sequence tagging ([Java](#))
- [PyStruct](#)  Structured Learning in Python ([Python](#))

By Xuan-Hieu Phan, Le-Minh Nguyen, and Cam-Tu Nguyen (2004):

- + first C/C++ implementation
- + second-order Markov
- + parallelism (MPI)

# **Những kinh nghiệm xây dựng ứng dụng với CRFs**

**building applications with CRFs: experience and lessons learnt**

- Trước khi quyết định dùng CRFs:
  - Xem xét bản chất của dữ liệu (nature of data)
  - Sequential dependencies phổ biến mức nào? mạnh mức nào?
  - Long-range dependencies?
  
- Huấn luyện mô hình CRFs:
  - Chọn window size phù hợp
  - Nhiều thuộc tính chưa chắc tốt: overfitting, tăng thời gian tính toán, mô hình bị phình to
  - Trích chọn mẫu thuộc tính (feature templates) tỉ mỉ và kỹ lưỡng, cần nhiều thuộc tính có tính phân biệt cao (highly discriminative)
  - Phân tích lỗi, điều chỉnh thuộc tính, điều chỉnh dữ liệu huấn luyện
  - Nên có development test set
  
- Hiệu quả và chi phí khi sử dụng CRFs:
  - Mang lại kết quả vượt trội so với các mô hình phân lớp đơn lẻ
  - Triển khai dưới dạng web services hay mobile services?

# **Kết luận bài giảng**

## **Summary**

# Những nội dung chính đã học

- Đoán nhận trên dữ liệu có cấu trúc
- Mô hình hóa CRFs
  - Markov random fields
  - Conditional random fields: linear-chain and general CRFs
  - Feature types of CRFs
- CRFs phụ thuộc cấp I và cấp II (first- and second-order Markov CRFs)
- Ước lượng tham số cho mô hình CRFs
  - Maximum likelihood estimation với phương pháp tối ưu L-BFGS
  - Huấn luyện CRFs song song
- Suy diễn/đoán nhận với CRFs
- Các ưu/nhược điểm của CRFs
- CRFs với các bài toán ứng dụng thực tế
- Những kinh nghiệm khi xây dựng ứng dụng với CRFs

**Thank you.**

**Q&A time**