Large Vocabulary Continuous Speech Recognition

Berlin Chen

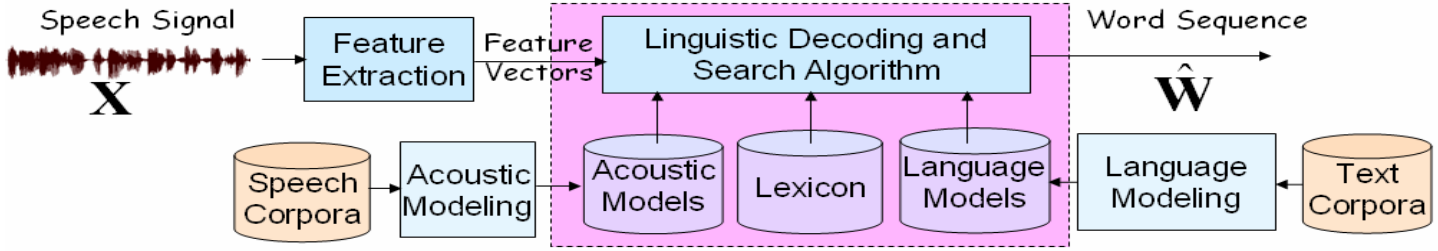
Department of Computer Science & Information Engineering
National Taiwan Normal University

References:

1. X. L., Aubert, “An Overview of Decoding Techniques for Large Vocabulary Continuous Speech Recognition,” *Computer Speech and Language*, vol. 16, 2002
2. H. Ney, “Progress in Dynamic Programming Search for LVCSR,” *Proceedings of the IEEE*, August 2000
3. S.Ortmanns, H. Ney, “The Time-Conditioned Approach in Dynamic Programming Search for LVCSR,” *IEEE Transactions on Speech and Audio Processing*, vol. 8, no. 6, November 2000
4. S.Ortmanns, H. Ney, X. Aubert, “A Word Graph Algorithm for Large Vocabulary Continuous Speech Recognition,” *Computer Speech and Language*, vol. 11, 1997

Why LVCSR Difficult ? (1/2)

- The software complexity of a search algorithm is considerably high
- The effort required to build an efficient decoder is quite large



Bayesian Decision Rule

(Risk Minimization with a Zero-One Risk Function)

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}} P(\mathbf{W}|\mathbf{X})$$

$$= \arg\max_{\mathbf{W}} \frac{p(\mathbf{X}\mid\mathbf{W})P(\mathbf{W})}{P(\mathbf{X})}$$

$$= \arg\max_{\mathbf{W}} p(\mathbf{X}\mid\mathbf{W})P(\mathbf{W})$$

Acoustic Model Probability

Language Model Probability

Why LVCSR Difficult ? (2/2)

- Maximum Approximation of the Decoding Problem

$$\hat{\mathbf{W}} = \arg\max_{\mathbf{W}} p(\mathbf{X} \mid \mathbf{W}) P(\mathbf{W})$$

$$= \arg\max_{\mathbf{W}:W_1^n} \left\{ P(W_1^n) \sum_{s_1^T} p(\mathbf{X}, s_1^T \mid W_1^n) \right\}$$

$$\left(s_1^T : \text{the corresponding state sequence of } W_1^n \right)$$

$$\hat{\mathbf{W}} \approx \arg\max_{\mathbf{W}:W_1^n} \left\{ P(W_1^n)^{\alpha} \max_{s_1^T} p(\mathbf{X}, s_1^T \mid W_1^n) \right\}$$

$$\left(\alpha : \text{heuristic language model factor} \right)$$

Two Major Constituents of LVCSR

- **Front-end Processing** is a spectral analysis that derives feature vectors to capture salient spectral characteristics of speech input
  - Digital signal processing
  - Feature extraction

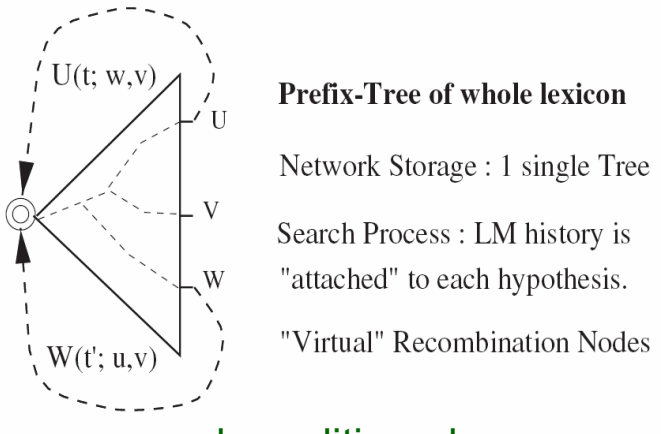
- **Linguistic Decoding** combines word-level matching and sentence-level search to perform an inverse operation to decode the message from the speech waveform
  - Acoustic modeling
  - Language modeling
  - Search

Classification of Approaches to LVCSR Along Two Axes

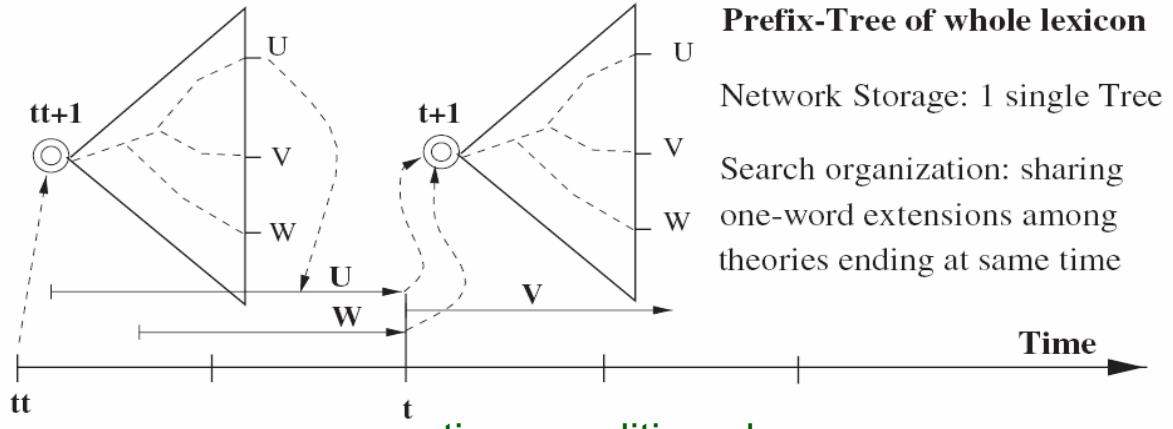
- Network Expansion
  - Dynamic expansion of the search space during the decoding
    - Tree-structured *n*-gram network
    - Re-entrant (word-conditioned) vs. start-synchronous (time-conditioned)
  - (Full) Static expansion of the search space prior to decoding
    - Weighted Finite State Transducers (WFST) - AT&T
      - Composition ($A \circ L \circ G$), Determinization, (Pushing) and Minimization of WFST
    - Static tree-based representations
- Search Strategy
  - Time-synchronous (Viterbi + Beam Pruning) Breadth-first (Parallel)
  - Time-asynchronous (A* or Stack Decoding) Best-first (Sequential)

Word-conditioned vs. Time-conditioned Search (1/3)

- Word-conditioned Search
  - A virtual tree copy is explored for each active LM history
  - More complicated in HMM state manipulation (dependent on the LM complexity)
- Time-conditioned Search
  - A virtual tree copy is being entered at each time by the word end hypotheses of the same given time
  - More complicated in LM-level recombination



word-conditioned



time-conditioned

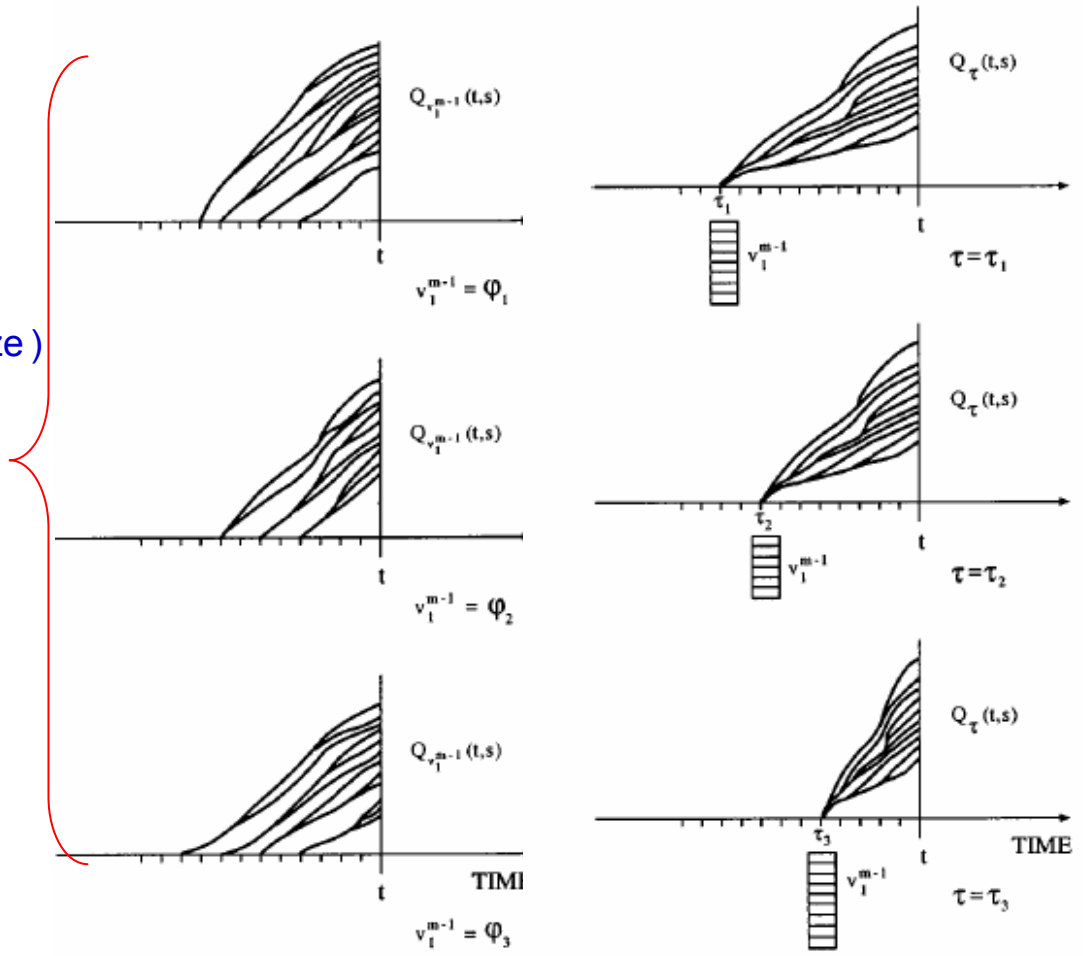
Word-conditioned vs. Time-conditioned Search (2/3)

- Schematic Comparison

Word-conditioned Search

Time-conditioned Search

No. of tree-copies is dependent on the complexity of LM constraints (for bigram constraint, MAX: $|V|^2$, $|V|$ lexicon size)



No. of tree-copies is independent of the complexity of LM constraints, but dependent on the duration of the utterance (however, the number is 30~50 according to the average word duration)

Word-conditioned vs. Time-conditioned Search (3/3)

Word-conditioned Search (with bigram LM)

proceed over time t from left to right

ACOUSTIC LEVEL: process state hypotheses $\{Q_{v_1^{m-1}}(t,s)\}$

- initialization: $Q_{v_1^{m-1}}(t-1,s=0)=H(v_1^{m-1};t-1)$

  $B_{v_1^{m-1}}(t-1,s=0)=t-1$

- time alignment: update $Q_{v_1^{m-1}}(t,s)$ using DP

- propagate back points $B_{v_1^m}(t,s)$

- prune unlikely hypotheses
- purge bookkeeping lists

WORD PAIR LEVEL: process word end hypotheses $\{Q_{v_1^{m-1}}(t,S_{v_m})\}$

for each word w do

$$H(v_2^m;t)=\max_{v_1}\{P(v_m|v_1^{m-1})Q_{v_1^{m-1}}(t,S_{v_m})\}$$

$$\tilde{v}_1(v_2^m;t)=\arg\max_{v_1}\{P(v_m|v_1^{m-1})Q_{v_1^{m-1}}(t,S_{v_m})\}$$

- store best predecessor $\tilde{v}_1=\tilde{v}_1(v_2^m;t)$
- store best boundary $\tilde{\tau}=B_{v_2^{m-1}}(t;S_{v_m})$

Time-conditioned Search (with *m*-gram LM)

proceed over time t from left to right

ACOUSTIC LEVEL: process state hypotheses $\{Q_\tau(t,s)\}$

- initialization:

$$Q_{t-1}(t-1,s)=\begin{cases}H_{\max}(t-1) & \text{if } s=0\\ 0.0 & \text{if } s>0\end{cases}$$

- time alignment: update $Q_\tau(t,s)$ using DP

- prune unlikely hypotheses
- purge bookkeeping lists

WORD LEVEL: process word end hypotheses $\{Q_\tau(t,S_{v_m})\}$

for each word end v_m do

for each boundary τ do

- word score: $h(v_m;\tau,t)=Q_\tau(t,S_{v_m})/H_{\max}(\tau)$

for each word $(m-1)$-gram v_2^m do

$$H(v_2^m;t)=\max_{(v_1,\tau)}\{P(v_m|v_1^{m-1})\cdot H(v_1^{m-1};\tau)\cdot h(v_m;\tau,t)\}$$

$$(\tilde{v}_1,\tilde{\tau})(v_2^m;t)=\arg\max_{(v_1,\tau)}\{P(v_m|v_1^{m-1})\cdot H(v_1^{m-1};\tau)\cdot h(v_m;\tau,t)\}$$

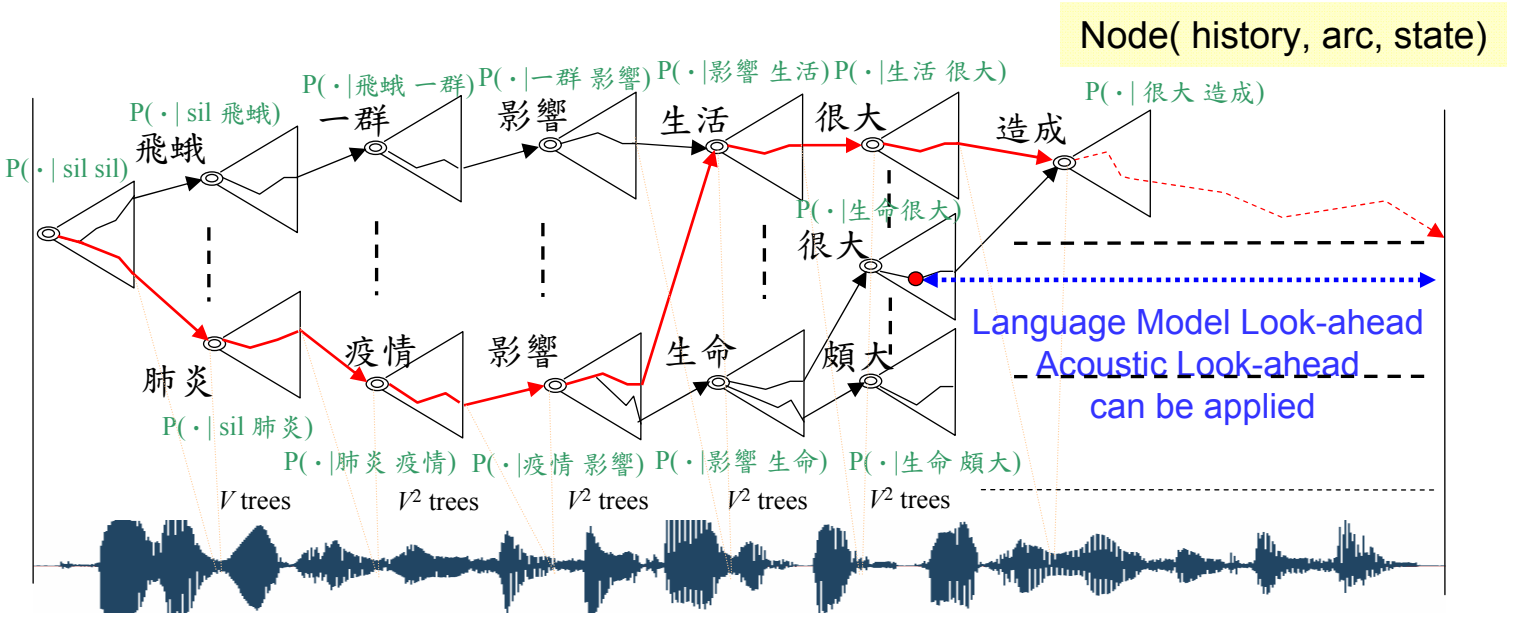
- store best predecessor $\tilde{v}_1(v_2^m;t)$
- store best boundary $\tilde{\tau}(v_2^m;t)$

$$H_{\max}(t)=\max_{v_2^m}\{H(v_2^m;t)\}$$

These two algorithms are stated in a "look-backward" manner.

Word-Conditioned Tree-Copy Search (WC)

- As trigram language modeling is used



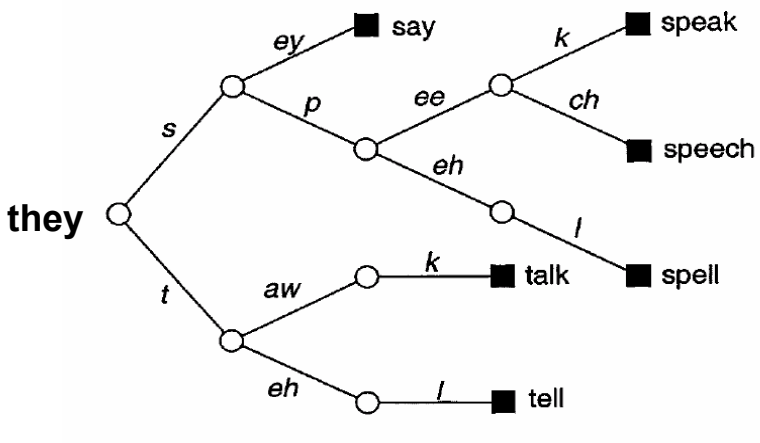
  - The pronunciation lexicon is structured as a tree
  - Due to the constraints of *n*-gram language modeling, a word's occurrence is dependent on the previous *n*-1 words
  - Search through all possible tree copies from the start time to the end time of the utterance to find a best sequence of word hypotheses

Lexical/Phonetic Tree (1/2)

- Each arc stands for a phonetic unit
- The application of language modeling is delayed until leaf nodes are reached
  - Word identities are only defined at leaf nodes (Solution: language model smearing/look-ahead)
  - Each leaf node is shared by words having the same pronunciation

$$P(say \mid they)$$

$$P(tell \mid they)$$



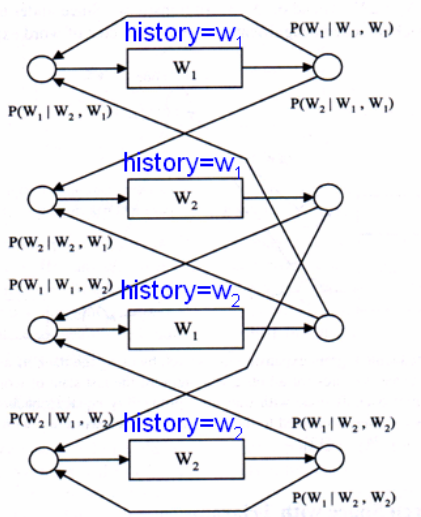
Lexical/Phonetic Tree (2/2)

- Reasons for using the lexical/phonetic tree
  - States according to phones that are common to different words are shared by different hypotheses
    - A compact representation of the acoustic-phonetic search space
  - The uncertainty about the word identity is much higher at its beginning than its ending
    - More computations is required at the beginning of a word than toward its end

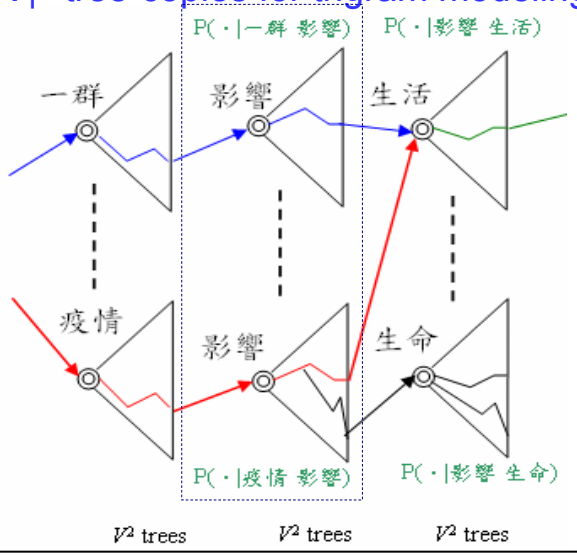
Linear Lexicon vs. Tree Lexicon
(for Word-Conditioned Search)

- In the general *n*-gram case ($n \geqq 1$), the total number of prefix tree copies is equal to $|V|^{n-1}$, where $|V|$ is the lexicon size and *n* the LM order
- When using a (flat) linear lexicon and in the general *n* -gram case ($n \geqq 2$), the total number of copies of the linear lexicon would be $|V|^{n-2}$

$|V|$ linear lexicons for trigram modeling

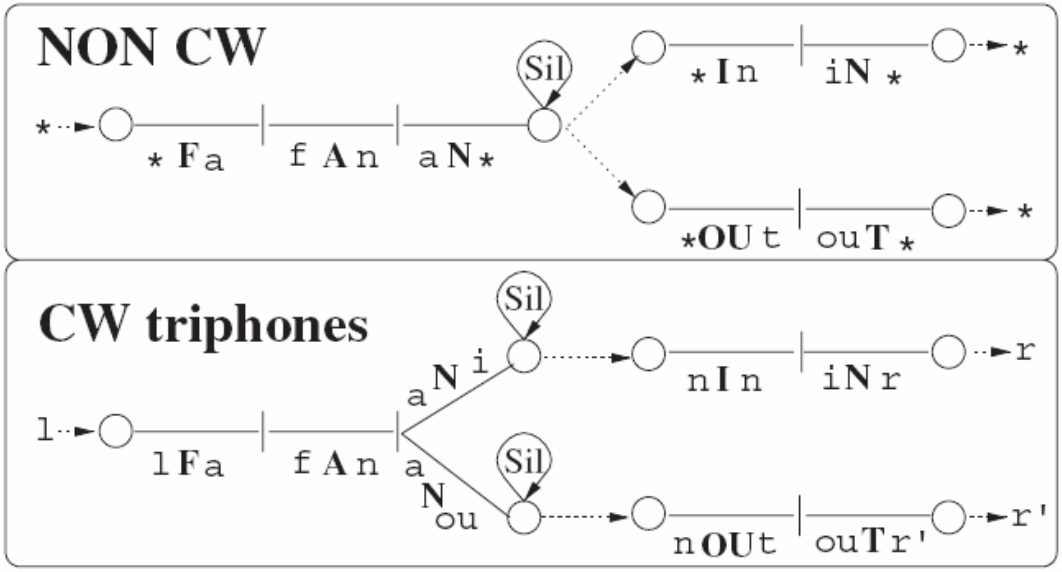


$|V|^2$ tree-copies for trigram modeling



Context-Dependent Phonetic Constraints

- Context-Dependent modeling (e.g., cross-word, CW, triphone modeling) is a crucial factor regarding the search space at the junction between successive words



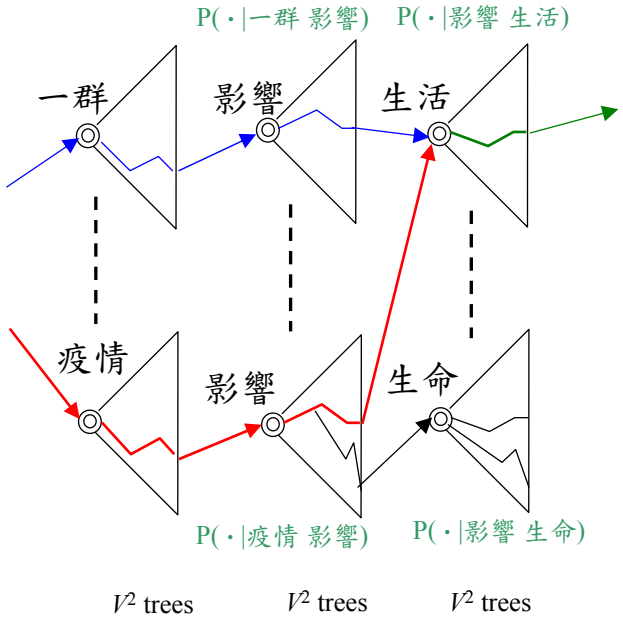
Implementation of CW would be much more difficult for time-conditioned than word-conditioned search (why ?)

- Current NTNU system is implemented with intra-word INITIAL/FINAL or triphone modeling

More Details on Word-Conditioned Search

Word-Conditioned Search (1/3)

- Word (history)-conditioned Search
  - **A virtual/imaginary tree copy** explored for linguistic context of active search hypotheses
  - Search hypotheses recombined at tree root nodes according to language modeling (or the history)



For *n*-gram language modeling:

- Retain distinct *n*-1-gram word histories

  - Time is the independent variable; expansion proceeds in parallel in a "breadth-first" manner

Word-Conditioned Search (2/3)

- Integration of **acoustic** and **linguistic** knowledge by means of tree-dimensional coordinates
  - A network (dynamically) built to describe sentences in terms of words
    - Language models for network transition probabilities
  - A network (statically) built to describe words in terms of phone
    - The pronunciation dictionary (organized as a phonetic tree)
    - Transition penalties are applied
  - A network (statically) built to describe a phone unit in terms of sequences of HMM states
    - Spectral vectors derived from the speech signal are consumed

Word-Conditioned Search (3/3)

- Three basic operations performed
  - Acoustic-level re-combinations within tree arcs
    - Viterbi search

$$Q_{v_1^{n-1}}(t,s;arc)=\max_{s'}\left[Q_{v_1^{n-1}}(t-1,s';arc)P(s|s';arc)\right]P(x_t|s;arc)$$

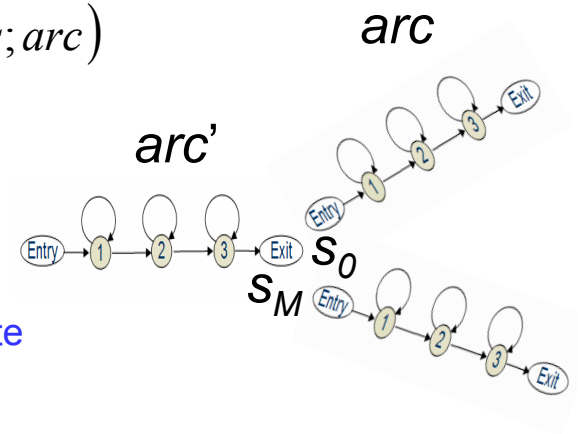
  - Tree arc extensions

$$Q_{v_1^{n-1}}(t,S_0;arc)=Q_{v_1^{n-1}}(t-1,S_M;arc')$$

The beginning state

The ending state

Backtracking Information should be manipulated

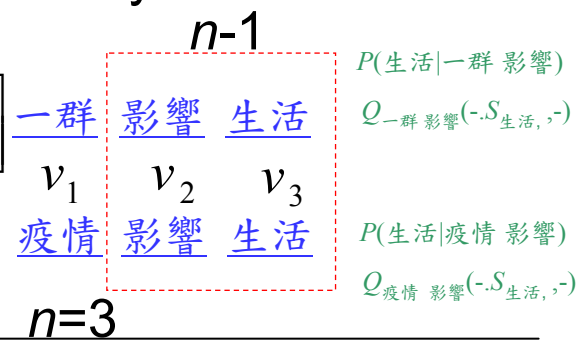


  - Language-model-level recombination
    - Word end hypotheses sharing the same history were recombined

$$H(v_2^n;t)=\max_{v_1}\left[Q_{v_1^{n-1}}(t,S_{v_n};arc_E)\cdot P(v_n|v_1^{n-1})^{\alpha}\right]$$

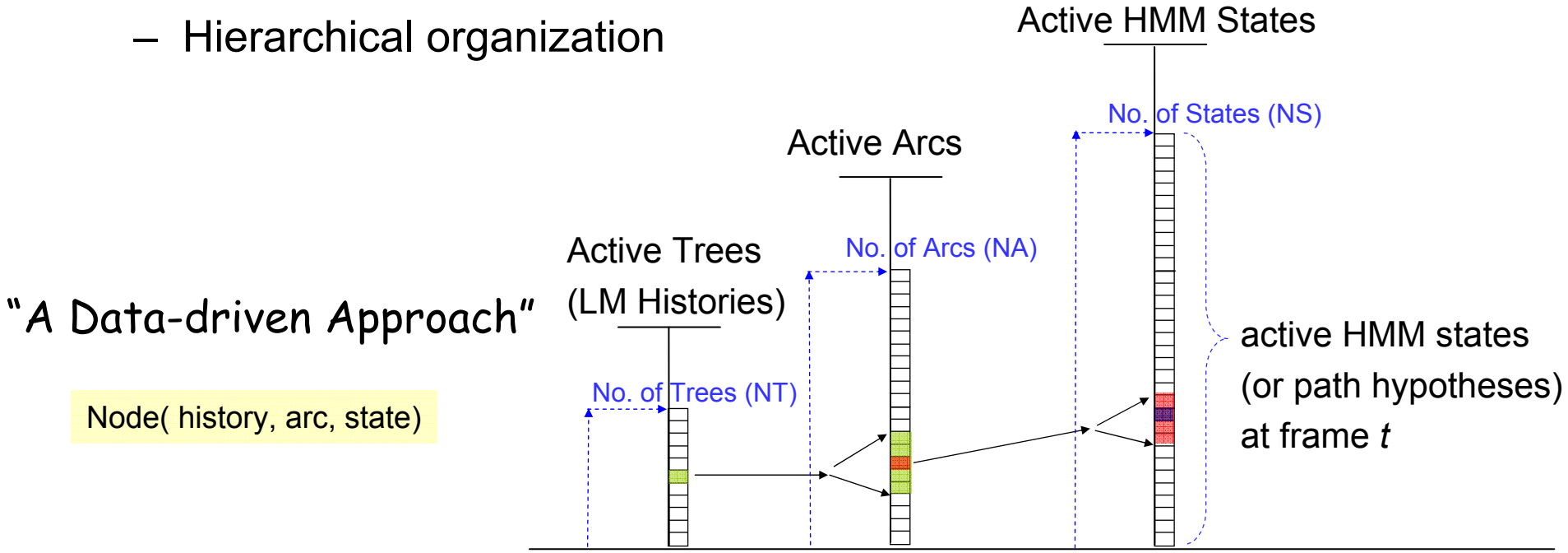
$$Q_{v_2^n}(t,S_0;arc_B)=H(v_2^n;t)$$

$$Q_{v_1^{n'-1}}(t,S_0;arc_B)$$



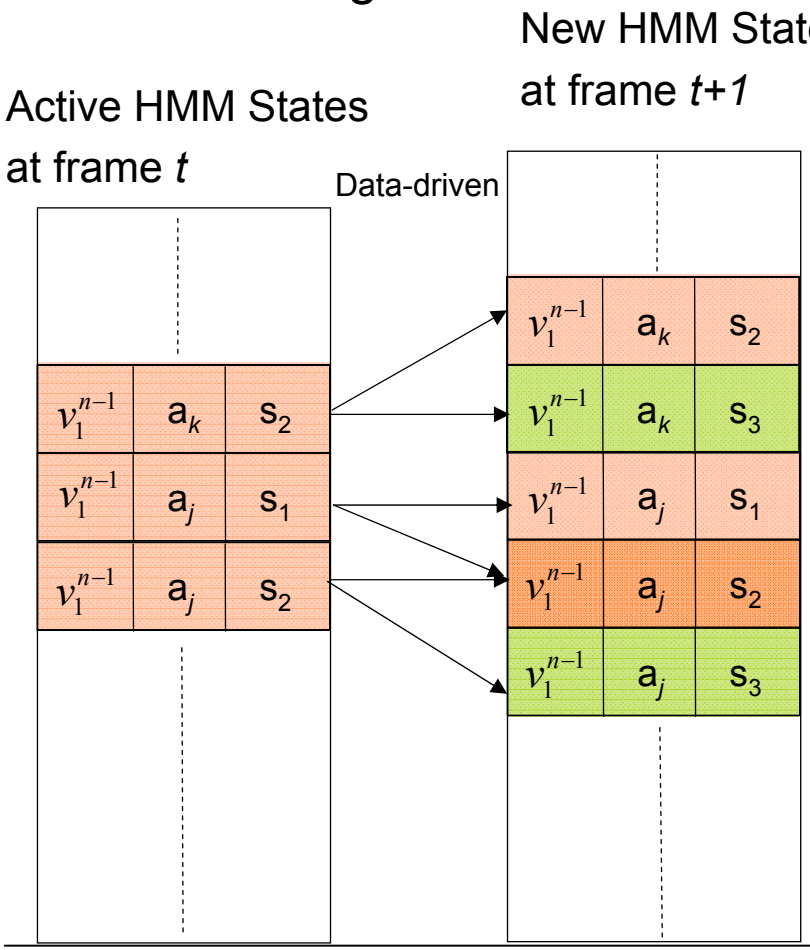
WC: Implementation Issues (1/3)

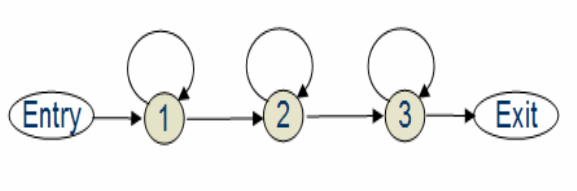
- Path hypotheses at each time frame are differentiated based on
  - The *n*-1 word history (for the *n*-gram LM)
  - The phone unit (or the tree arc)
  - The HMM state
- Organization of active path hypotheses (states)
  - Hierarchical organization



WC: Implementation Issues (2/3)

- Organization of active path hypotheses (states)
  - Flat organization





Acoustic level recombination

$$Q_{v_1^{n-1}}(t, s; arc) = \max_{s'} \left[Q_{v_1^{n-1}}(t-1, s'; arc) P(s|s'; arc) \right] P(x_t|s; arc)$$

C++ STL (Standard Template Libraries) is suitable for such an access

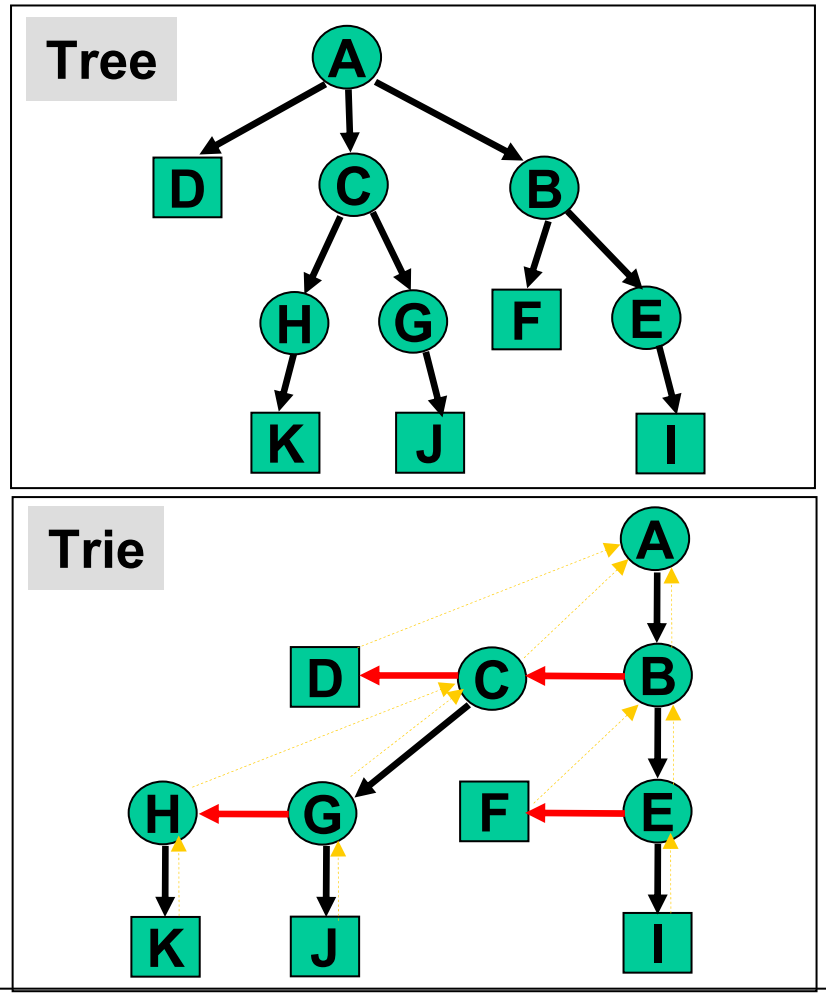
$\log(NS)$ for the access of any HMM state

NS: the number of HMM states

WC: Implementation Issues (3/3)

- The pronunciation lexicon is organized as a “trie” (tree) structure

```
struct DEF_LEXICON_TREE
{
        short    Model_ID;
        short    WD_NO;
        int      *WD_ID;
        int       Leaf;
        double   Unigram;
        struct   Tree *Child;
        struct   Tree *Brother;
        struct   Tree *Father;
};
```



WC: Pruning of Path Hypotheses (1/5)

- Viterbi search
  - Belong to a class of breadth-first search
    - *Time-synchronous*
    - Hypotheses terminate at the same point in time
      - Therefore, hypotheses can be compared
  - The search hypotheses will grow *exponentially*
  - Pruning away unlikely (incorrect) paths is needed
    - Viterbi *beam* search
    - Hypotheses with likelihood falling within a fixed radius (or beam) of the most likely hypothesis are retained
    - The beam size determined empirically or adaptively

WC: Pruning of Path Hypotheses (2/5)

- Pruning Techniques
  1. Standard Beam Pruning (Acoustic-level Pruning)
     - Retain only hypotheses with a score close to the best hypothesis

$$Thr_{AC}(t) = \left[\max_{(v_1^{n-1}, s, arc)} Q_{v_1^{n-1}}(t, s; arc) \right] \times f_{AC}$$

$$Q_{v_1^{n-1}}(t, s; arc) < Thr_{AC}(t) \Rightarrow \text{pruned!}$$

  2. Language Model Pruning (Word-level Pruning)
     - Applied to word-end or tree start-up hypotheses

$$Thr_{LM}(t) = \left[\max_{(v_1^{n-1}, S_0, arc_B)} Q_{v_1^{n-1}}(t, S_0; arc_B) \right] \times f_{LM}$$

$$Q_{v_1^{n-1}}(t, S_0; arc_B) < Thr_{LM}(t) \Rightarrow \text{pruned!}$$

  3. Histogram Pruning
     - Limit the number of surviving state hypotheses to a maximum number (Need some kind of sorting!)
     - Not recommended!

WC: Pruning of Path Hypotheses (3/5)

- Pruning Techniques (cont.)
  - Stricter pruning applied at word ends
    - The threshold is tightly compared to the acoustic-level one
    - **Reasons**
      - A single path hypothesis is propagated into multiple word ends (words with the same pronunciation)
      - A large number of arcs (models) of the new generated tree copies are about to be activated

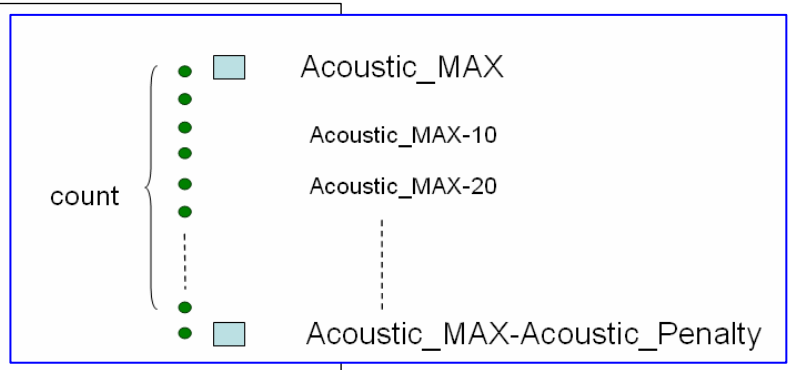
Pose severe requirements on the system memory

WC: Pruning of Path Hypotheses (4/5)

- A simple dynamic pruning mechanism in the NTNU system
  - Acoustic-Level Pruning

```
Acoustic_Penalty=800;
Acoustic_MAX=(float) Min_Delta;
count=0;
for(state_no=0;state_no<NewTreeState;state_no++)
{
  cur_HMM=LEX_STATE[PT2][state_no].TPTR->Model_ID;
  cur_state=LEX_STATE[PT2][state_no].HMM_state;
  if(LEX_STATE[PT2][state_no].Score>Acoustic_MAX)
        Acoustic_MAX=LEX_STATE[PT2][state_no].Score;

}
for(state_no=0;state_no<NewTreeState;state_no++)
{
  cur_HMM=LEX_STATE[PT2][state_no].TPTR->Model_ID;
  cur_state=LEX_STATE[PT2][state_no].HMM_state;
  if(LEX_STATE[PT2][state_no].Score>(Acoustic_MAX-Acoustic_Penalty))
      count++;
}
//20020522
if(count>100000)     Acoustic_MAX=Acoustic_MAX-40;
else if(count>50000) Acoustic_MAX=Acoustic_MAX-80;
else if(count>10000) Acoustic_MAX=Acoustic_MAX-100;
else if(count>5000)  Acoustic_MAX=Acoustic_MAX-150;
else if(count>2000)  Acoustic_MAX=Acoustic_MAX-200;
else if(count>1000)  Acoustic_MAX=Acoustic_MAX-300;
else if(count>400)   Acoustic_MAX=Acoustic_MAX-400;
else                 Acoustic_MAX=Acoustic_MAX-500;
ATreeState=0;
for(state_no=0;state_no<NewTreeState;state_no++)
{
   if(LEX_STATE[PT2][state_no].Score>Acoustic_MAX)
   {
     LEX_STATE[PT1][ATreeState]=LEX_STATE[PT2][state_no];
     ATreeState++;
   }
}
```



WC: Pruning of Path Hypotheses (5/5)

- A simple dynamic pruning mechanism in the NTNU system
  - Word-Level (language-model-level) Pruning

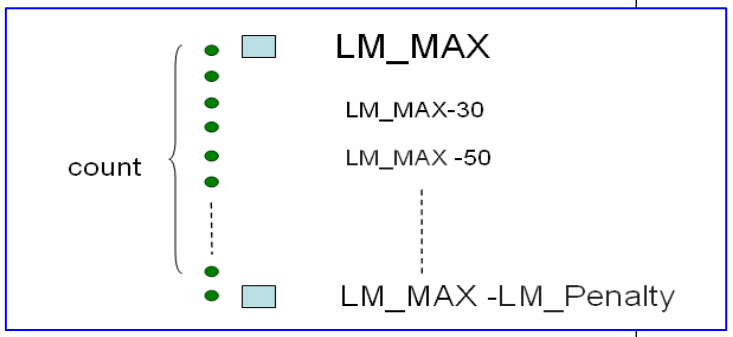
```
LM_Penalty=200;
LM_MAX=(float) Min_Delta;
for(j=0;j<LOCAL_ACTIVE_WORD_NO;j++)
  if(LOCAL_ACTIVE_TREE[j].Score>LM_MAX)
       LM_MAX=LOCAL_ACTIVE_TREE[j].Score;

count=0;
for(j=0;j<LOCAL_ACTIVE_WORD_NO;j++)
  if(LOCAL_ACTIVE_TREE[j].Score>(LM_MAX-LM_Penalty))
       count++;
 //berfore 20020522
if     (count>200) LM_MAX=LM_MAX-30;
else if(count>100) LM_MAX=LM_MAX-50;
else if(count>50)  LM_MAX=LM_MAX-70;
else               LM_MAX=LM_MAX-80;

count=0;
for(j=0;j<LOCAL_ACTIVE_WORD_NO;j++)
  if(LOCAL_ACTIVE_TREE[j].Score>=LM_MAX)
       count++;

if((ACTIVE_TREE_WORD[Frame_Num]
      =( struct DEF_ACTIVE_TREE_WORD *)malloc((count+1)*sizeof( struct DEF_ACTIVE_TREE_WORD)))==NULL)
  {
    printf("ACTIVE_TREE_WORD allocation error at FRAME %d!\n",Frame_Num);
    exit(1);
  }

ACTIVE_TREE_WORD_NO[Frame_Num]=0;
for(j=0;j<LOCAL_ACTIVE_WORD_NO;j++)
  if(LOCAL_ACTIVE_TREE[j].Score>=LM_MAX)
    {
       ACTIVE_TREE_WORD[Frame_Num][ACTIVE_TREE_WORD_NO[Frame_Num]]=LOCAL_ACTIVE_TREE[j];
       ACTIVE_TREE_WORD_NO[Frame_Num]++;
    }
```



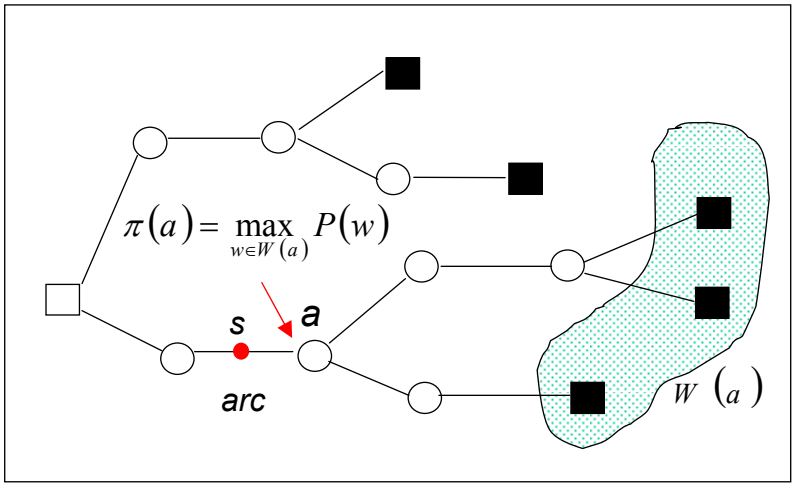
WC: Language Model Look-ahead (1/2)

- Language model probabilities incorporated as early in the search as possible
  - Language model probability incorporated for computing of $Q_{v_1^{n-1}}(t,s;arc)$
  - **Unigram Look-ahead**

$$\pi(a) = \max_{w \in W(a)} P(w)$$

  - **Bigram Look-ahead**

$$\pi_v(a) = \max_{w \in W(a)} P(w|v)$$



- Anticipate the language model probabilities with the state hypothesis

$$\tilde{Q}_{v_1^{n-1}}(t,s;arc) = \pi(a_{s,arc}) Q_{v_1^{n-1}}(t,s;arc)$$

$$Thr_{AC}(t) = \left[\max_{(v_1^{n-1},s,arc)} \pi(a_{s,arc}) Q_{v_1^{n-1}}(t,s;arc) \right] \times f_{AC}$$

$$\tilde{Q}_{v_1^{n-1}}(t,s;arc) < Thr_{AC}(t) \Rightarrow \text{pruned!}$$

Implementation of bigram look-ahead would be much difficult for time-conditioned than word-conditioned search (why ?)

WC: Language Model Look-ahead (2/2)

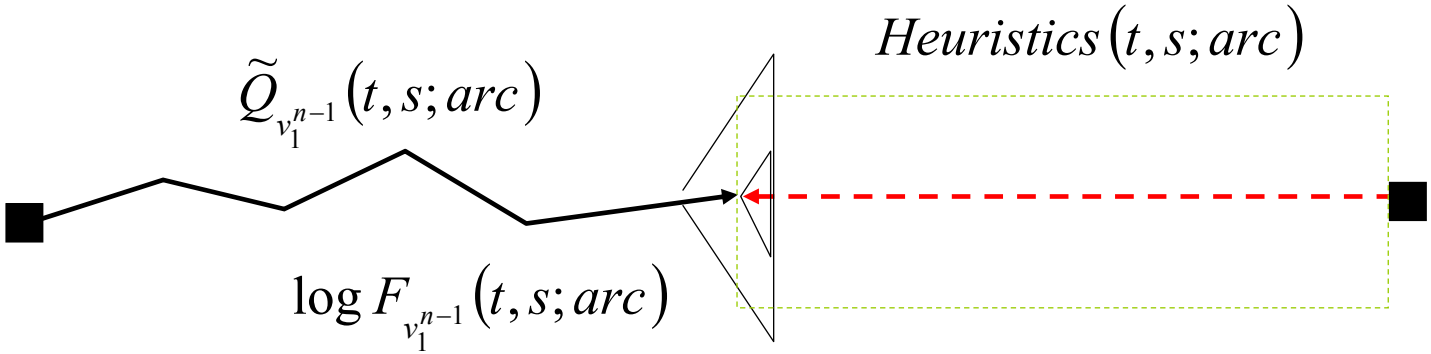
- A Simple recursive function for calculating unigram LM look-ahead

```
void SpeechClass::Calculate_Word_Tree_Unigram()
{
      if(Root==(struct Tree *) NULL) return;
      Do_Calculate_Word_Tree_Unigram(Root);
}

void SpeechClass::Do_Calculate_Word_Tree_Unigram(struct Tree *ptrNow)
{
  if(ptrNow==(struct Tree *) NULL) return;
  Do_Calculate_Word_Tree_Unigram(ptrNow->Brother);
  Do_Calculate_Word_Tree_Unigram(ptrNow->Child);
  if(ptrNow->Father!=(struct Tree *) NULL)
   if(ptrNow->Unigram > ptrNow->Father->Unigram)
     ptrNow->Father->Unigram=ptrNow->Unigram;
}
```

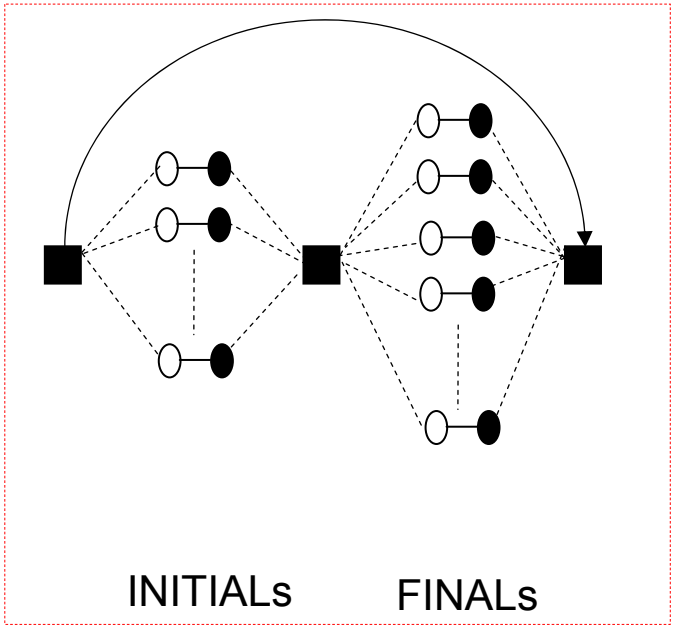
WC: Acoustic Look-ahead (1/3)

- The Chinese language is well known for its monosyllabic structure, in which each Chinese word is composed of one or more syllables
  - Utilize syllable-level heuristics to enhance search efficiency
    - Help make the right decision when pruning
  - How to design the a suitable structure in order to estimate the heuristics for the unexplored portion of a speech utterance?

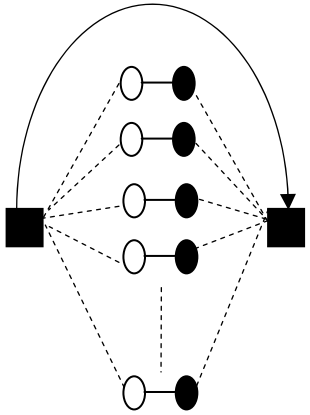


WC: Acoustic Look-ahead (2/3)

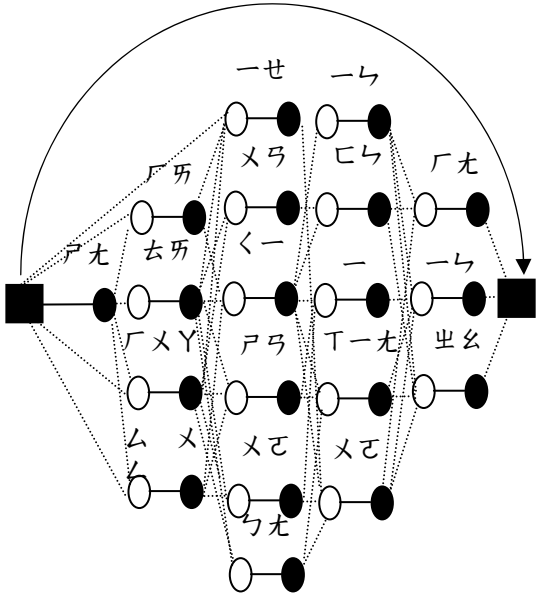
- Possible structures for heuristic estimation for the Chinese language



INITIALs FINALs



Syllables



A lattice based on the lexical tree

- Incorporated with the actual decoding score and language model look-ahead score

$$\log \widetilde{D}\left(t,h,arc_k,s_q\right)=m_1'\cdot\log D\left(t,h,arc_k,s_q\right)+m_2'\cdot\log L_{LM}\left(arc_k\right)+m_3'\cdot\log L_{AC}\left(t,arc_k,s_q\right)$$

WC: Acoustic Look-ahead (3/3)

- Tested on 4-hour radio broadcast news (Chen et al., ICASSP 2004)
  - Use Kneser-Ney backoff smoothing for language modeling

Table 1. The baseline character error rates (%) achieved using different feature extraction approaches.

	Character Error Rate (%)	
	TS	WG
MFCC	26.34	22.55
LDA-1	23.10	19.90
LDA-2	23.13	19.97
LDA-2+Acoustic Look-ahead	23.24	20.12

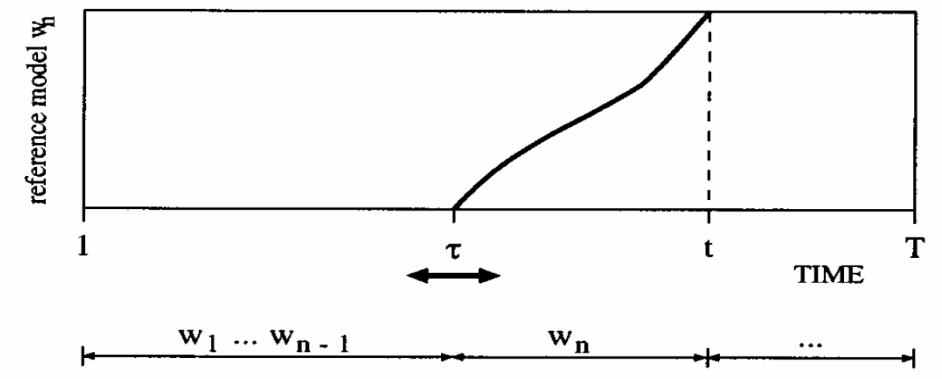
Table 2. Recognition efficiency achieved as acoustic model look-ahead was further applied. The recognition efficiency is expressed in terms of the real time factor.

	FE	L_{AC}	TS	WG	Total
Without Acoustic Look-ahead	0.323	0.000	1.264	0.196	1.783
With Acoustic Look-ahead	0.323	0.004	0.738 (41.61%)	0.149 (23.98%)	1.214 (31.91%)

- The recognition efficiency for TC improves significantly (a relative improvement of 41.61% was obtained) while the time spent on acoustic look-ahead (0.004 real time factor) was almost negligible

More Details on Time-Conditioned Search

TC: Decomposition of Search History (1/3)



$$\underbrace{x_1, \cdots, \cdots, x_\tau}_{G(w_1^{n-1};\tau)} \underbrace{x_{\tau+1}, \cdots, x_t}_{h(w_n;\tau,t)} \underbrace{x_{t+1}, \cdots, \cdots, x_T}_{\cdots}$$

$$h(w;\tau,t) = \max_{s_{\tau+1}^t} p\left(x_{\tau+1}^t, s_{\tau+1}^t \middle| w\right)$$

$$= \text{conditional prob. that word } w \text{ produces } x_{\tau+1}^t$$

$$G(w_1^n;t) = P(w_1^n) \max_{s_1^t} p\left(x_1^t, s_1^t \middle| w_1^n\right)$$

$$= \text{joint prob. of observing } x_1^t \text{ and } w_1^n \text{ ending at } t$$

If the whole word history is kept:

$$G(w_1^n;t) = \max_{\tau} \left\{P\left(w_n \middle| w_1^{n-1}\right) G(w_1^{n-1};\tau) h(w;\tau,t)\right\}$$

$$= P\left(w_n \middle| w_1^{n-1}\right) \cdot \max_{\tau} \left\{G(w_1^{n-1};\tau) h(w;\tau,t)\right\}$$

If only the latest *m*-1 word history is instead kept:

$$G(w_1^n;t) \approx H(v_2^m;t)$$

$$H(v_2^m;t)$$

$$= \max_{w_1^n : w_{n-m+2}^n = v_2^m} \left\{P(w_1^n) \cdot \max_{s_1^t} p\left(x_1^t, s_1^t \middle| w_1^n\right)\right\}$$

(with DP recursion)

$$= \max_{v_1} \left\{P\left(v_m \middle| v_1^{m-1}\right) \cdot \max_{\tau} \left\{H(v_1^{m-1};\tau) h(w;\tau,t)\right\}\right\}$$

TC: Decomposition of Search History (2/3)

- Tree-internal search hypotheses (for an internal node, not word-end node, *s*)

If the whole word history is kept:

$$Q_\tau(t;s) = \max_{w_1^n} P(w_1^n) \cdot \left\{ \max_{s_1^t : s_\tau = S_{w_n}, s_t = s} p(x_1^t, s_1^t | w_1^n) \right\}$$

$$= \max_{w_1^n} P(w_1^n) \cdot \left\{ \max_{s_1^t : s_\tau = S_{w_n}} p(x_1^\tau, s_1^\tau | w_1^n) \right\} \cdot \max_{s_{\tau+1}^t : s_t = s} p(x_{\tau+1}^t, s_{\tau+1}^t | -)$$

$$= \max_{w_1^n} G(w_1^n; \tau) \cdot \max_{s_{\tau+1}^t : s_t = s} p(x_{\tau+1}^t, s_{\tau+1}^t | -)$$

Word identity is still unknown

If only the latest *m*-1 word history is instead kept:

$$Q_\tau(t;s) = \max_{w_1^n} G(w_1^n; \tau) \cdot \max_{s_{\tau+1}^t : s_t = s} p(x_{\tau+1}^t, s_{\tau+1}^t | -)$$

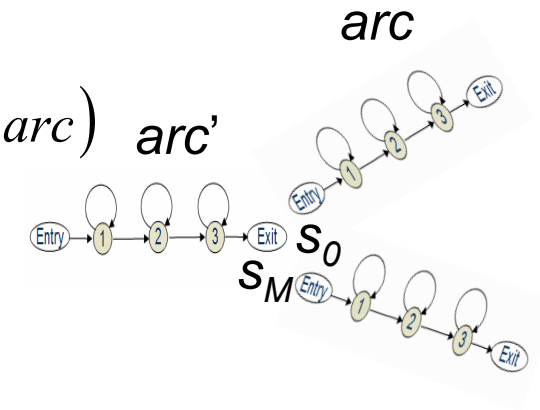
$$\approx \max_{v_2^m : v_2^m = w_{n-m+2}^n} H(v_2^m; \tau) \cdot \max_{s_{\tau+1}^t : s_t = s} p(x_{\tau+1}^t, s_{\tau+1}^t | -)$$

$$= H_{\max}(\tau) \cdot \max_{s_{\tau+1}^t : s_t = s} p(x_{\tau+1}^t, s_{\tau+1}^t | -)$$

TC: Decomposition of Search History (3/3)

- Three basic operations performed
  - Acoustic-level re-combinations within tree arcs
    - Viterbi search

$$Q_\tau(t,s;arc)=\max_{s'}\left[Q_\tau(t-1,s';arc)P(s|s';arc)\right]P(x_t|s;arc)$$



  - Tree arc extensions

$$Q_\tau(t,S_0;arc)=Q_\tau(t-1,S_M;arc')$$

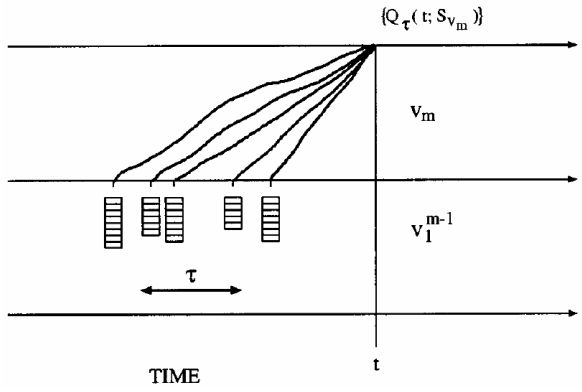
The beginning state — The ending state

  - Word-level recombination

$$H(v_2^m;t)=\max_{v_1}\left\{P(v_m|v_1^{m-1})\cdot\max_{\tau}\left\{H(v_1^{m-1};\tau)h(v_m;\tau,t)\right\}\right\}$$

$$=\max_{v_1}\left\{P(v_m|v_1^{m-1})\cdot\max_{\tau}\left\{H(v_1^{m-1};\tau)\frac{Q_\tau(t,S_{v_m};arc_E)}{H_{\max}(\tau)}\right\}\right\}$$

$$Q_t(t,S_0;arc_B)=H_{\max}(t)=\max_{v_2^m}H(v_2^m;t)$$



TC vs. WC: Conflicting Expectations

- Word-conditioned hypotheses seem to be more condense
  - Since we have got rid of the word boundary information when performing path recombination
- The number of time-conditioned hypotheses might be smaller because the number of possible start times (1,…,*T*) is always smaller than the number of word histories in the word-conditioned search
  - E.g., a recognition task consisting of 10,000 words and a sentence consisting of 2000 frames

$$2 \times 10^{3} \ll 1 \times 10^{8}$$

(TC) (WC)

characteristic features	search method	
	WC	TC
time synchronous processing of most promising hypotheses	yes	yes
optimization over word boundaries	integrated	separate
caching of intermediate results for LM recombination	no	yes
total number of state hyp. for more complex LMs	increases	constant

TC vs. A* search (1/2)

- A* for LVCSR
  - Performed with the maximum approximation
  - The partial word sequence hypotheses $\left(w_1^n;\tau\right)$ associated with the scores $G\left(w_1^n;\tau\right)$ play the central role
    - Multistack: time-specific priority queues used to rank the partial word sequence hypotheses $\left(w_1^n;\tau\right)$ at each time τ
    - Need a conservative estimate of the prob. Score (LM + Acoustic) extending from $\left(w_1^n;\tau\right)$ to the end of the speech signal (difficult to achieve!)
      - Instead approximately obtained normalized $G\left(w_1^n;\tau\right)$ with respect to $\max_{w_1^n} G\left(w_1^n;\tau\right)$

TC vs. A* search (2/2)

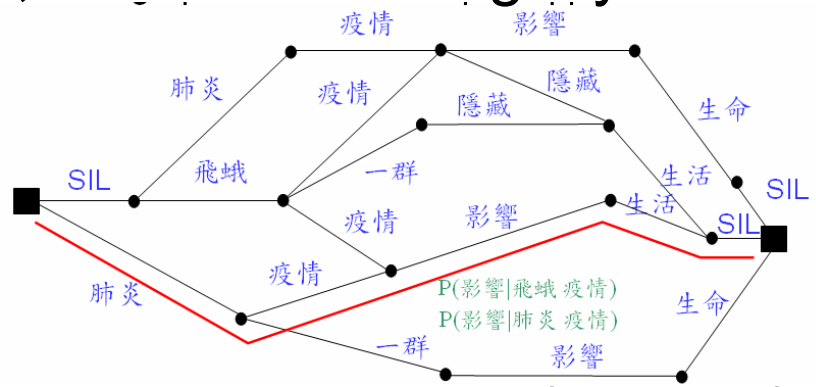
- Operations (time-asynchronous and look-forward)
  - Select the most promising hypothesis $\left(w_1^n;\tau\right)$
    - The best hypotheses with the shortest end time τ are extend first
      - » When reaching a word-end state at time t , incorporate the LM prob. And update the relevant queue
  - Pruning at
    - The level of partial word sequence hypotheses $\left(w_1^n;\tau\right)$
    - The level of acoustic word hypotheses $\left(w_{n+1};\tau,t\right)$
    - The level of DP recursion on the tree-internal state hypotheses

- TC for LVCSR
  - Operations (or algorithm) performed in time-synchronous and look-backward manner

Word Graph Rescoring

Word Graph Rescoring (1/6)

- Tree-copy search with a higher order (trigram, fourgram, etc.) language model (LM) will be very time-consuming and impractical
- A word graph provides word alternatives in regions of the speech signal, where the ambiguity about the actual spoken words in high

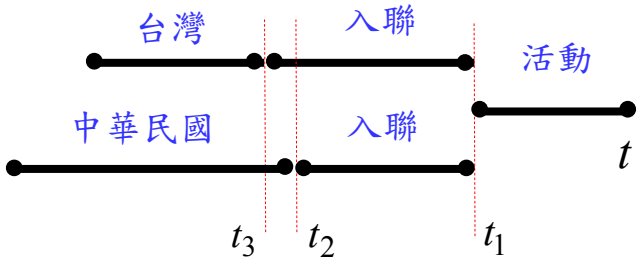


  - Decouple the acoustic recognition (matching) and language model application
    - Such that more complicated long-span language models (such as PLSA, LSA, Trigger-based LMs, etc.) can be applied in the word graph rescoring process

Word Graph Rescoring (2/6)

- If word hypotheses ending at each time frame have scores higher than a predefined threshold
  - Their associated decoding information, such as the word start and end speech frames, the identities of current and predecessor words, and the acoustic score, can be kept in order to build a word graph for further higher-order language model rescoring
    - E.g., a bigram LM was used in the tree search procedure, while a trigram LM in the word graph rescoring procedure
- Keep track of word hypotheses whose scores are very close to the locally optimal hypothesis, but that do not survive due to the recombination process

Bigram History Assumed



P(活動|台灣 入聯)

P(活動|中華民國 入聯)

Word Graph Rescoring (3/6)

- If **bigram LM** is employed in the tree-copy search
  - For a word hypothesis w ending at time t, information about its beginning time and its immediate predecessor word should be retained

$$\tau(t; v, w) = B_v(t, S_w; arc_E)$$

beginning time of w

v : immediate predecessor word of w

  - The acoustic score of a word hypothesis w is also retained

$$AC_v(w; \tau, t) = Q_v(t, S_{v_n}; arc_E) / H(v, \tau)$$

acoustic score of w

$AC_{v_0}(w; \tau_0, t)$

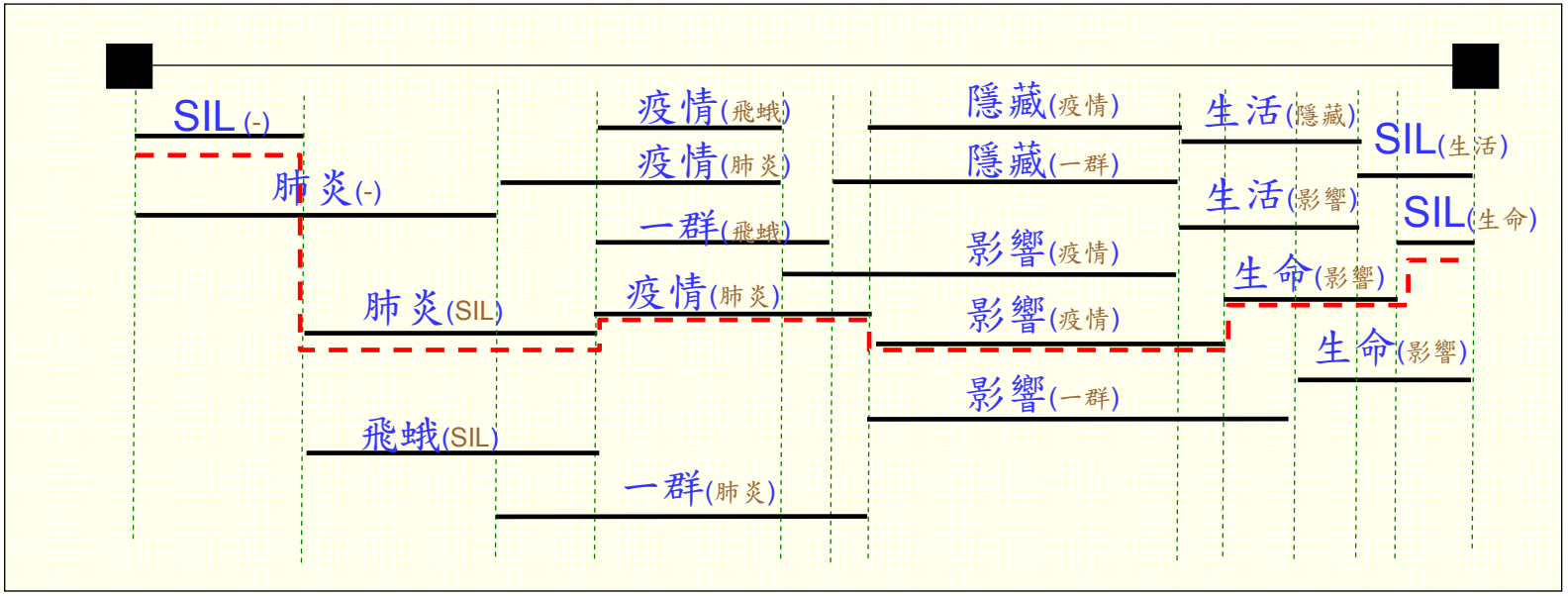
$AC_{v_1}(w; \tau_1, t)$

$AC_{v_2}(w; \tau_2, t)$

For each possible word hypothesis w, not only the word segment with the best predecessor word were recorded (those do not survive due to re-combination are also kept)

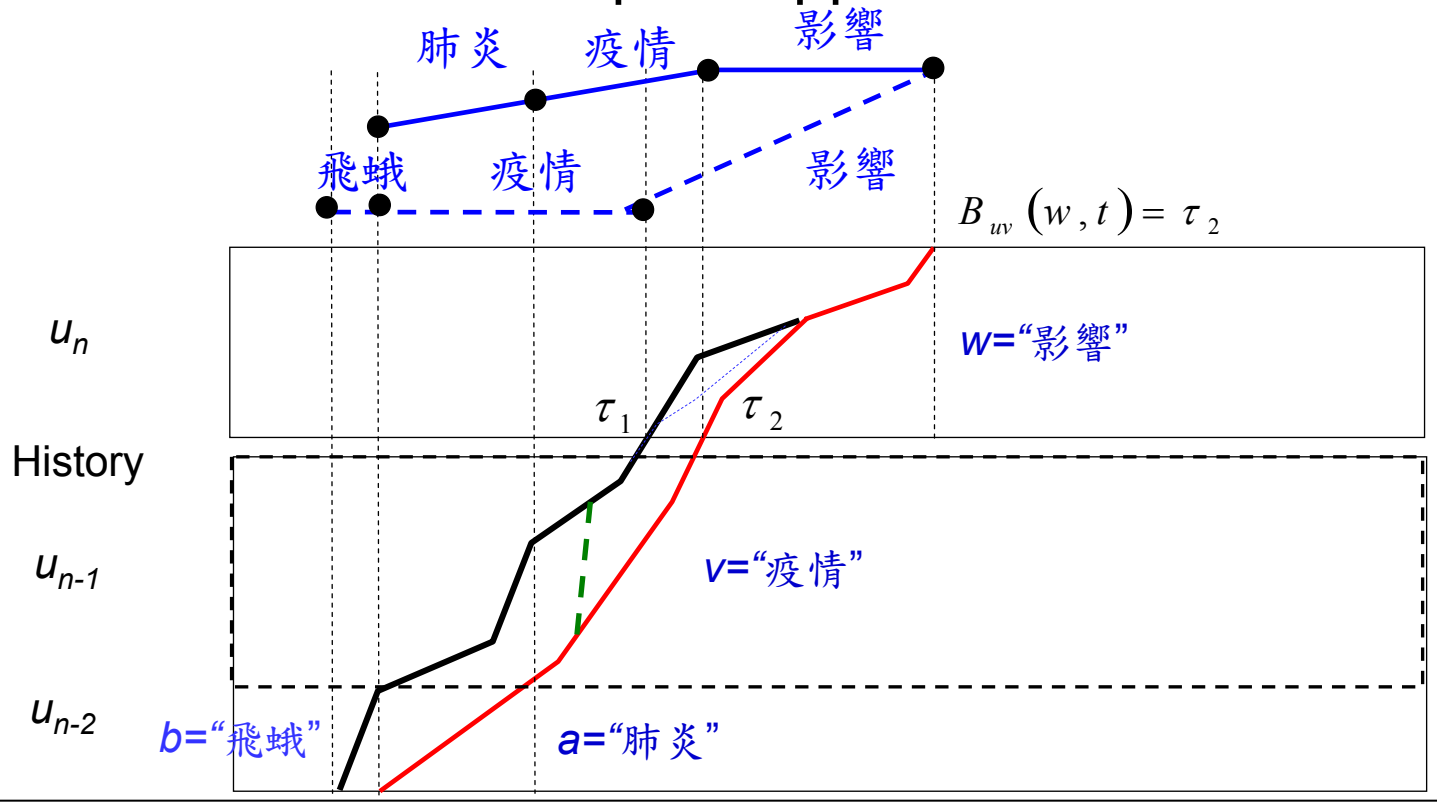
Word Graph Rescoring (4/6)

- Bookkeeping at the word level
  - When word hypotheses were recombined into one hypothesis to start up the next tree
    - Not only the word segments (arcs) with the best predecessor word were recorded
    - But for the hypotheses that have the same LM history, only the best one was kept ("word-pair" approximation)



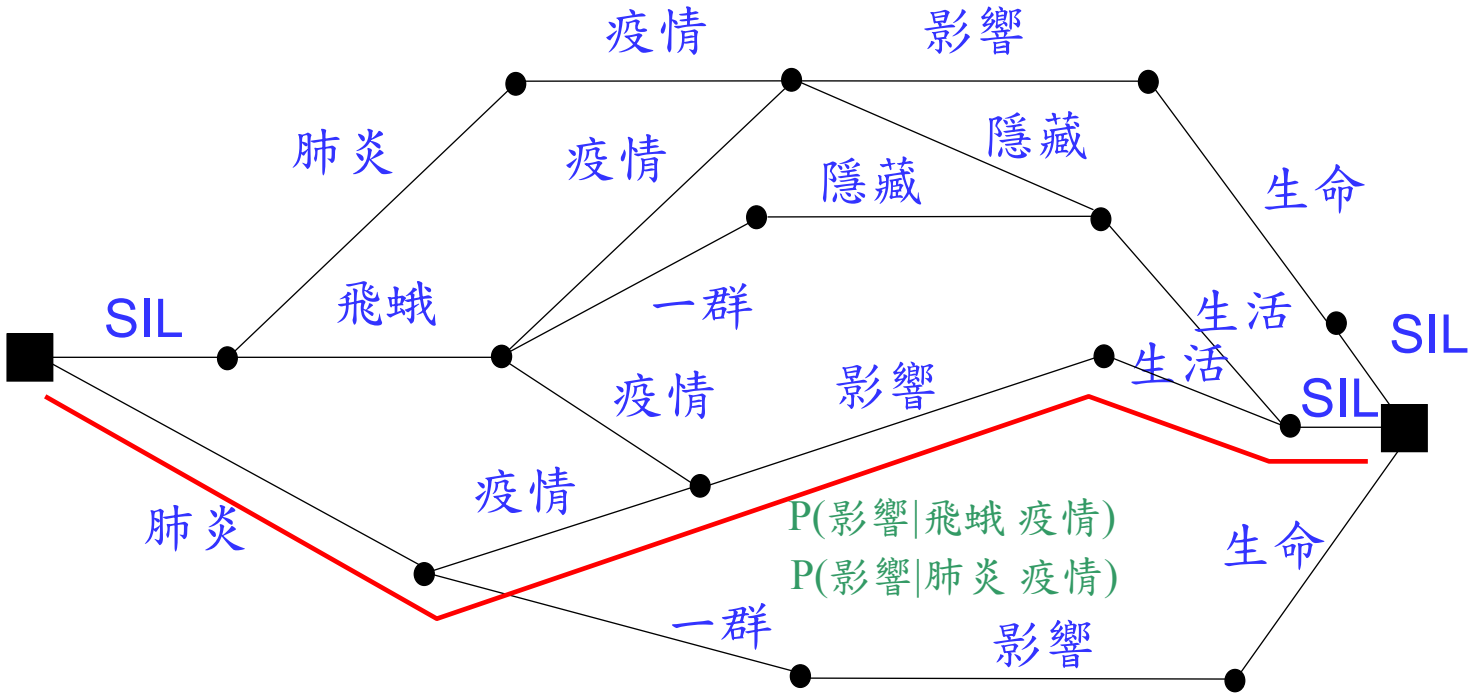
Word Graph Rescoring (5/6)

- "Word-Pair" Approximation: for each path hypothesis, the position of word boundary between the last two words is independent of the other words of this path hypothesis
- An illustration for "word-pair" approximation



Word Graph Rescoring (6/6)

- A word graph built with word-pair approximation



- Each edge stands for a word hypothesis
- The node at the right side of an edge denotes the **word end**
  - There is a maximum of incoming word edges for a node
  - There is no maximum of the num. of outgoing edges for a node

Configuration of NTNU System

- Feature Extraction
  - HLDA (Heteroscedastic Linear Discriminant Analysis) + MLLT (Maximum Likelihood Linear Transformation) + MVN (Mean and Variance Normalization)
- Language Modeling
  - Bigram/Trigram trained with the SRI toolkit
- Acoustic Modeling
  - 151 RCD INITIAL/FINAL models trained with the HTK toolkit
  - Intra-word triphone modeling is currently under development
- Look-ahead Schemes
  - Unigram Language model look-ahead
  - Utterance-level acoustic look-ahead