Multi-Modal User Interaction
Fall 2008

Lecture 2: Speech recognition I

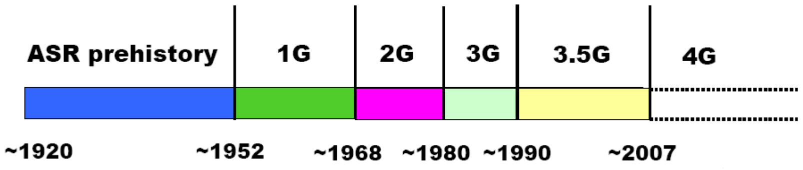
Zheng-Hua Tan

Department of Electronic Systems
Aalborg University, Denmark
zt@es.aau.dk

Part I: Introduction

- Introduction, history and trends
- Speech signal representation
- Template based approach – DTW
- Statistical model based approach – HMM
- Variability

ASR history

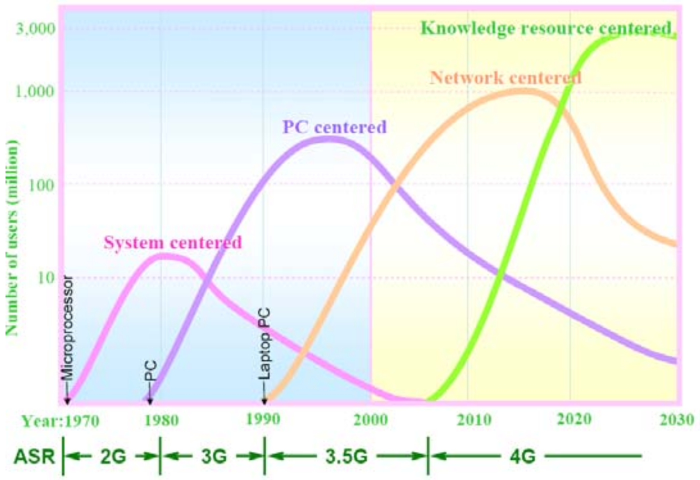


1. Spectral resonances extracted by an analogue filter bank and logic circuits; phoneme, syllable, digit recognition
2. DTW, IWR, IBM LV ASR, Bell Labs SI ASR, CMU CSR, ARPA 5-year project
3. Statistical framework, HMM (CMU, IBM, J.Baker), Δcepstrum (Furui), N-gram (IBM), Neural net, DARPA program (Resource management task)

- Discriminative approach, robust ASR, DARPA program (broadcast news, EARS rich transcription, GALE), spontaneous speech, audio-visual ASR

(S. Furui 2007)

IT Technology progress



(David C. Moschella: "Waves of Power")

1971-1976: The ARPA project

- ARPA launched 5 year Spoken Understanding Research project
- Goal: 1000 word vocabulary, a few speakers, continuous speech, constrained grammar, 90% understanding rate, near real time on a 100 MIPS machine
- 4 Systems built by the end of the program
  - SDC (24%), BBN's HWIM (44%), CMU's Hearsay II (74%), CMU's HARPY (95% -- but 80 times real time!)

Raj Reddy

- HARPY based on engineering approach: search on network of all the possible utterances
- Conclude: Speech Understanding is too early for its time

Lesson learned:

- Hand-built knowledge does not scale up
- Need of a global "optimization" criterion

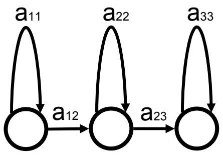
1980s -- The Statistical Approach

- Hidden Markov Models based statistical approach (Fred Jelinek and Jim Baker, IBM)
- Foundations of modern speech recognition engines

$$\hat{W} = \arg\max_{W} P(A \mid W)P(W)$$

Acoustic HMMs

Word Tri-grams



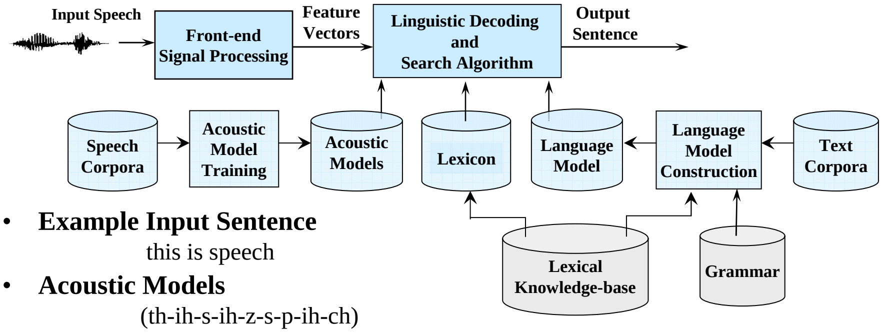
$$P(w_t \mid w_{t-1}, w_{t-2})$$

J Baker

- No Data Like More Data
- Whenever I fire a linguist, our system performance improves (1988)
- Some of my best friends are linguists (2004)

Large vocabulary speech recognition

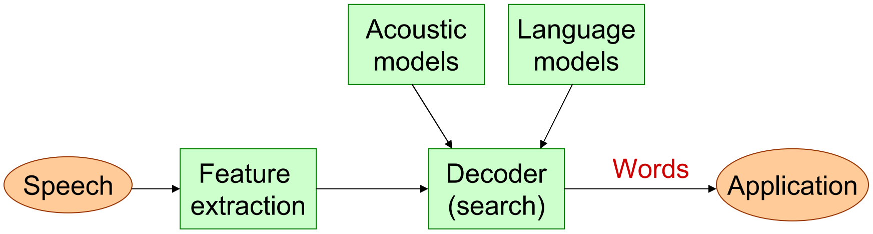
- **A Block Diagram**



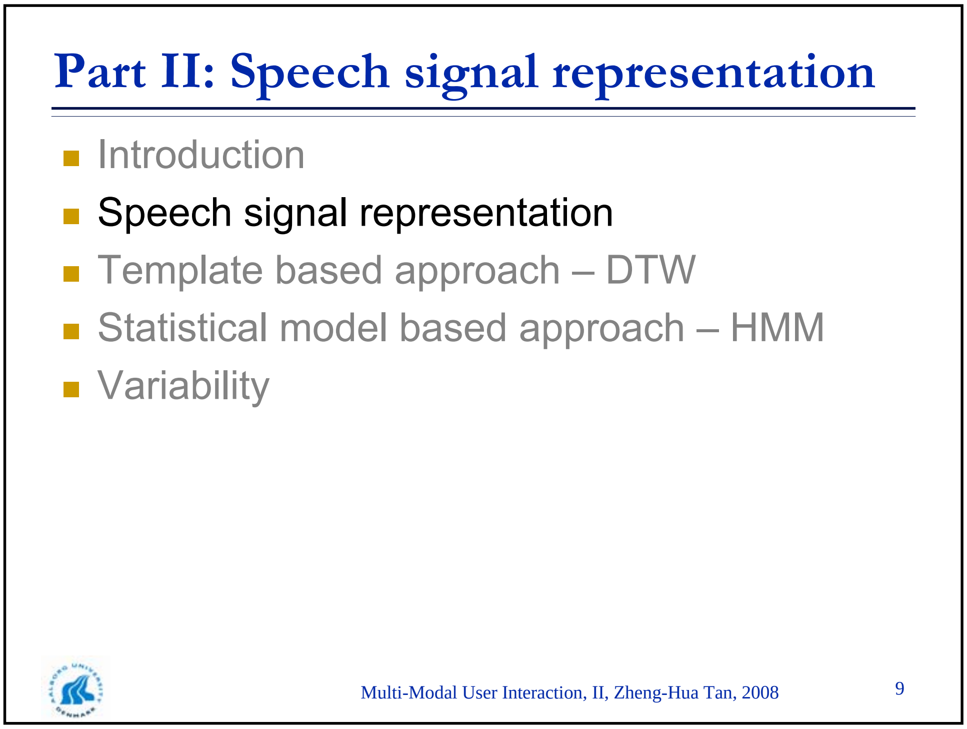
- **Example Input Sentence**
  this is speech
- **Acoustic Models**
  (th-ih-s-ih-z-s-p-ih-ch)
- **Lexicon** (th-ih-s) → this
  (ih-z) → is
  (s-p-iy-ch) → speech
- **Language Model** (this) – (is) – (speech)
  P(this) P(is | this) P(speech | this is)
  $P(w_i|w_{i-1})$ bi-gram language model
  $P(w_i|w_{i-1},w_{i-2})$ tri-gram language model,etc

L.S. Lee, 2007

Key components of LVCSR system

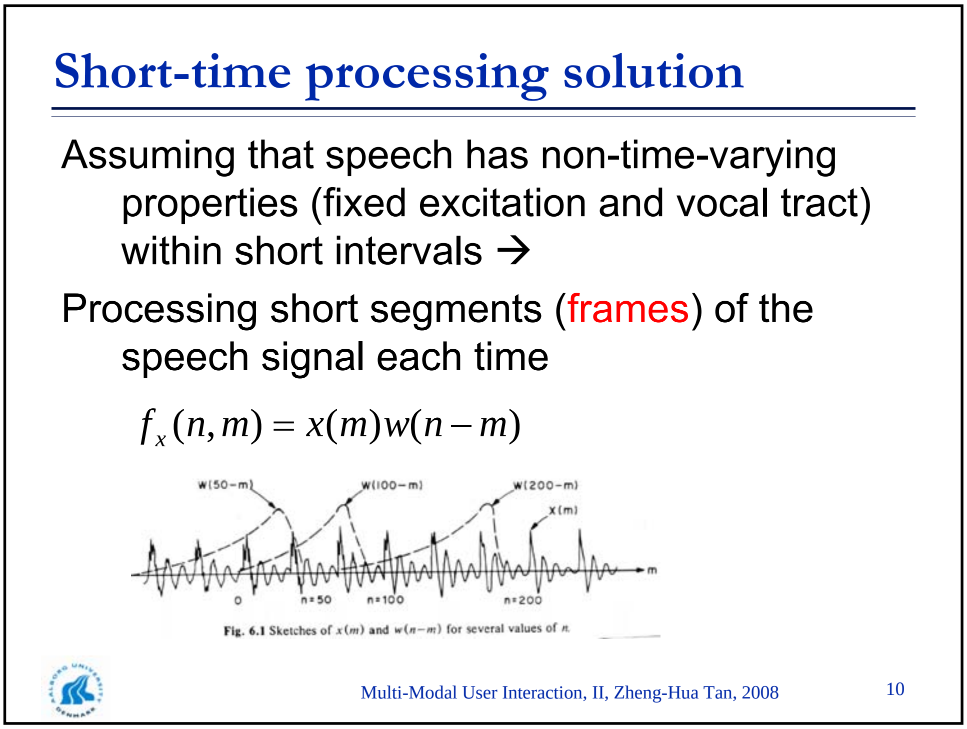


- Speech recognition involves:
  - How to represent the signal
  - How to model both acoustic and language constraints
  - How to search for the optimal answer



Part II: Speech signal representation

- Introduction
- Speech signal representation
- Template based approach – DTW
- Statistical model based approach – HMM
- Variability



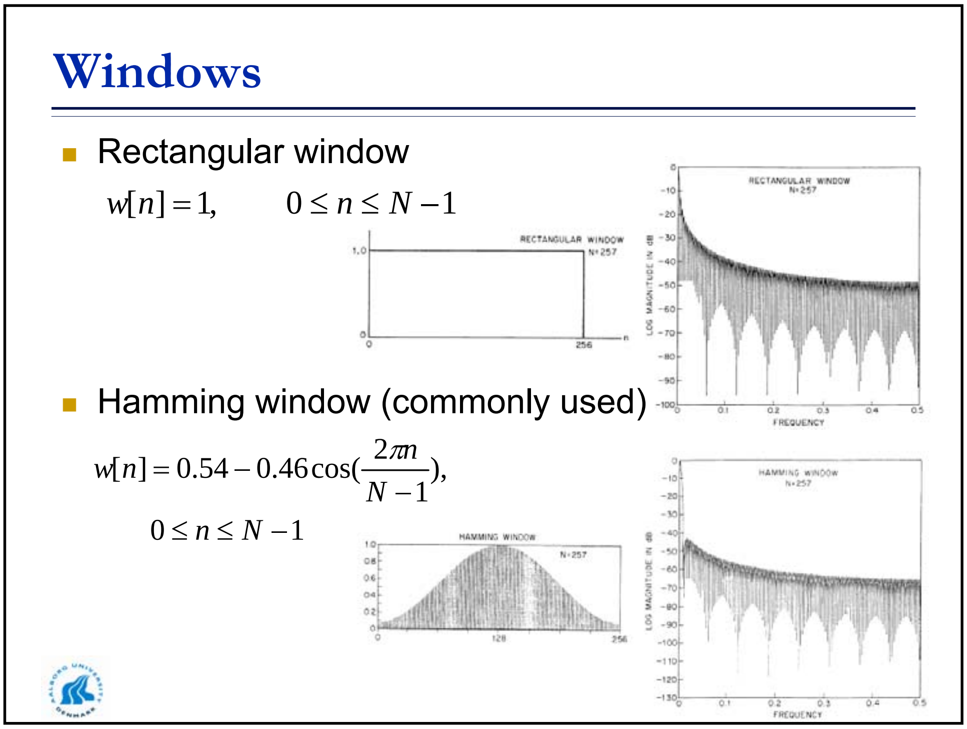
Short-time processing solution

Assuming that speech has non-time-varying properties (fixed excitation and vocal tract) within short intervals →

Processing short segments (frames) of the speech signal each time

$$f_x(n,m) = x(m)w(n-m)$$

Fig. 6.1 Sketches of $x(m)$ and $w(n-m)$ for several values of n.



Windows

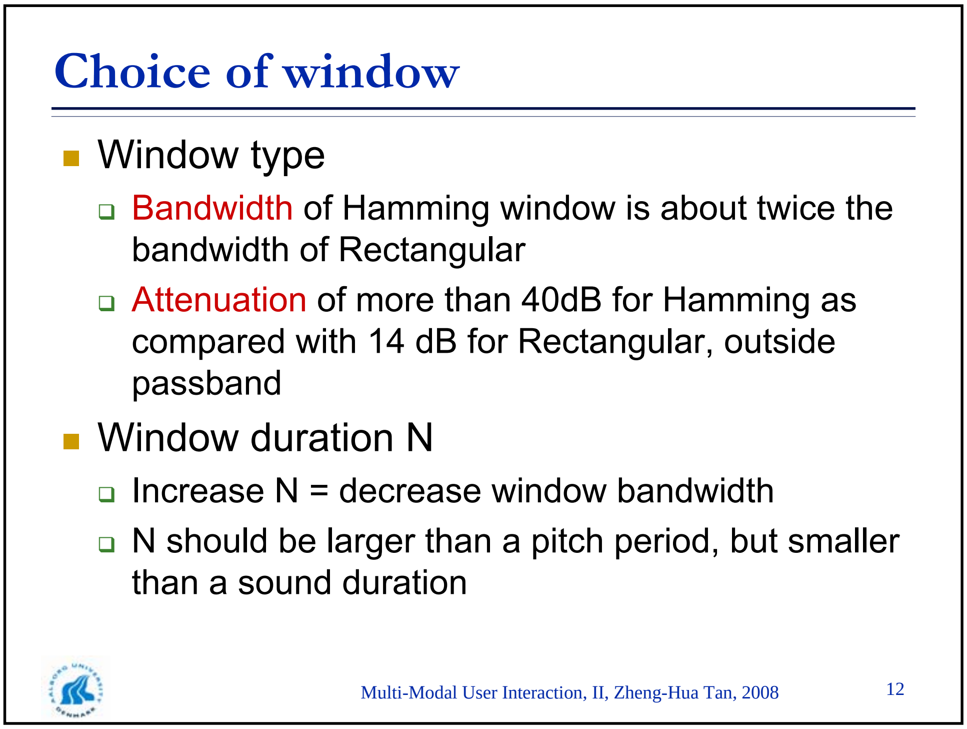
- Rectangular window

$$w[n]=1, \qquad 0 \le n \le N-1$$

- Hamming window (commonly used)

$$w[n]=0.54-0.46\cos(\frac{2\pi n}{N-1}),$$

$$0 \le n \le N-1$$

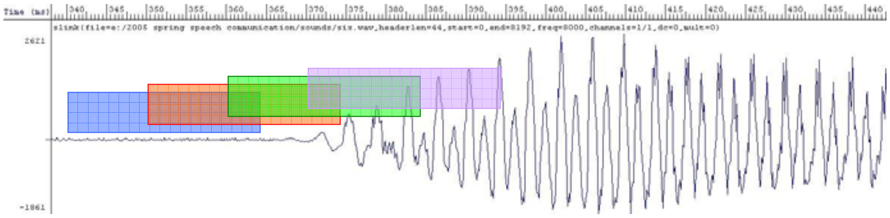


Choice of window

- Window type
  - Bandwidth of Hamming window is about twice the bandwidth of Rectangular
  - Attenuation of more than 40dB for Hamming as compared with 14 dB for Rectangular, outside passband
- Window duration N
  - Increase N = decrease window bandwidth
  - N should be larger than a pitch period, but smaller than a sound duration

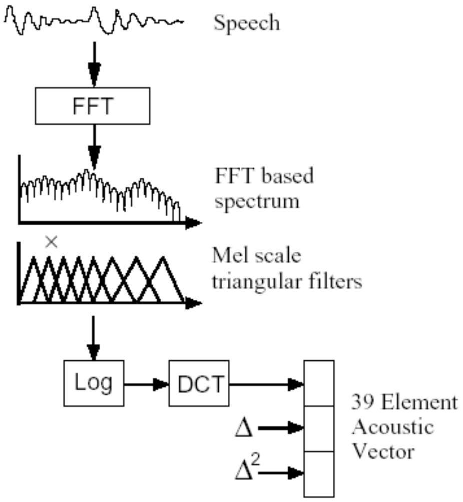
Dimension & speech representation

- The curse of dimension – the computational cost increases exponentially with the dimension of the problem
- The frame-based analysis yields a sequence as a new representation of the speech signal
  - ❑ samples at 8000/sec → vectors at 100/sec



Front-end feature extraction

MFCC



Speech

FFT

FFT based spectrum

×

Mel scale triangular filters

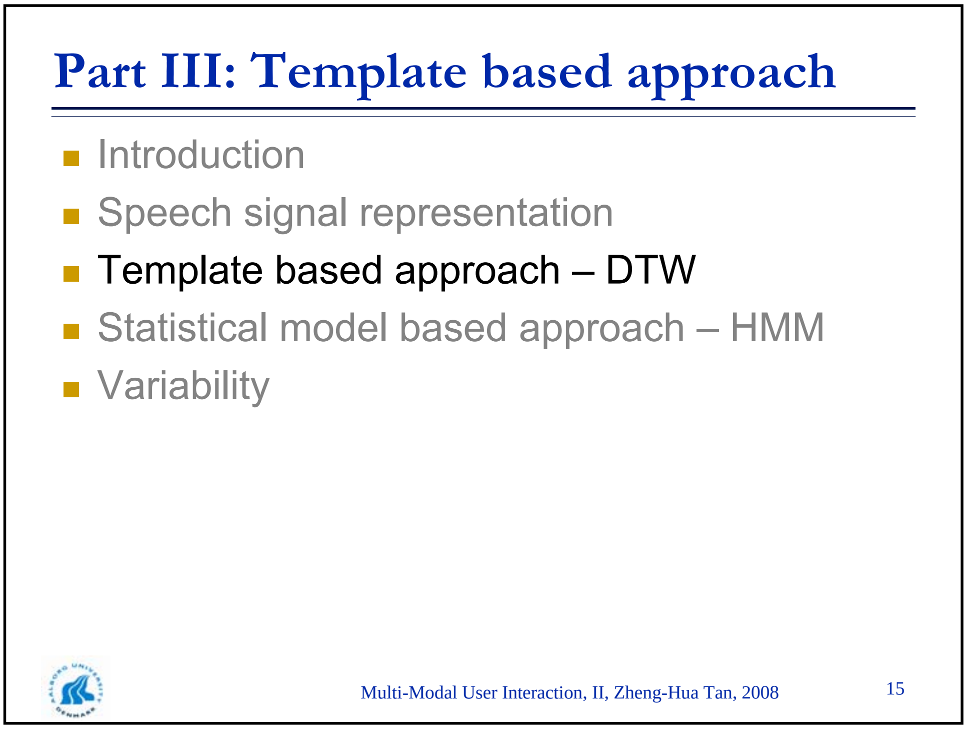
Log

DCT

Δ

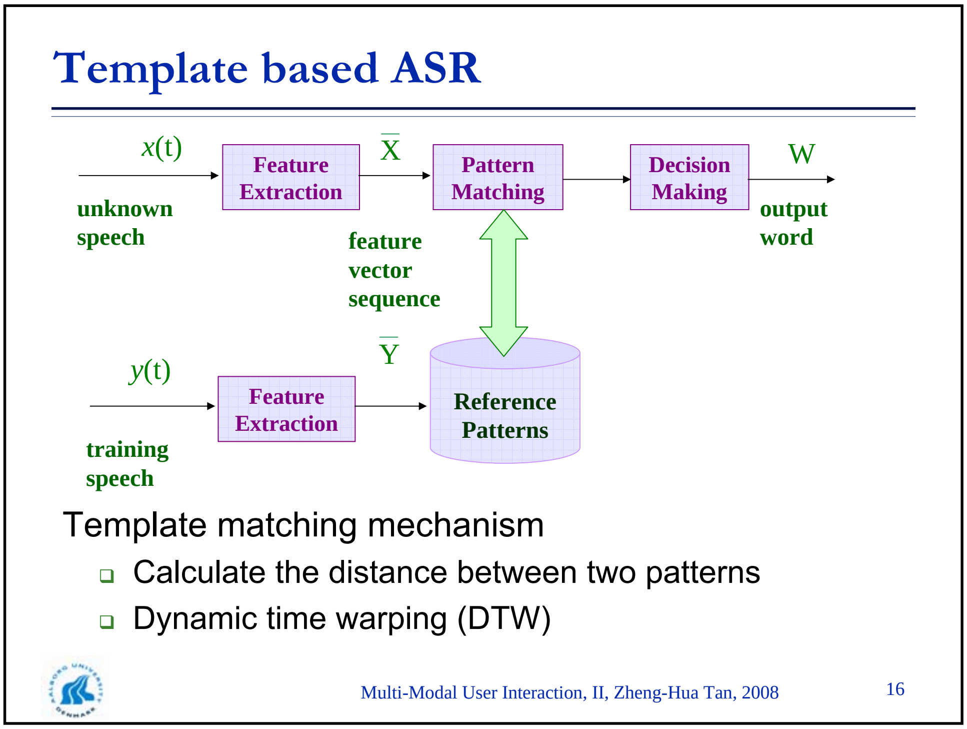
Δ^2

39 Element Acoustic Vector



Part III: Template based approach

- Introduction
- Speech signal representation
- Template based approach – DTW
- Statistical model based approach – HMM
- Variability



Template based ASR

$x(t)$ unknown speech → Feature Extraction → $\overline{X}$ feature vector sequence → Pattern Matching → Decision Making → W output word

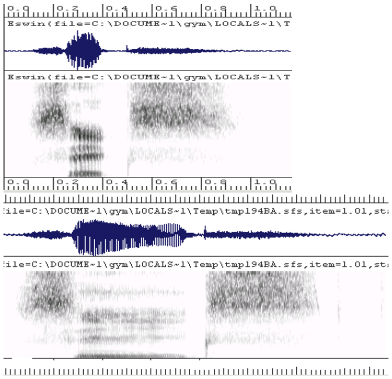
$y(t)$ training speech → Feature Extraction → $\overline{Y}$ → Reference Patterns ↔ Pattern Matching

Template matching mechanism

- Calculate the distance between two patterns
- Dynamic time warping (DTW)

Speaking rate and time-normalization

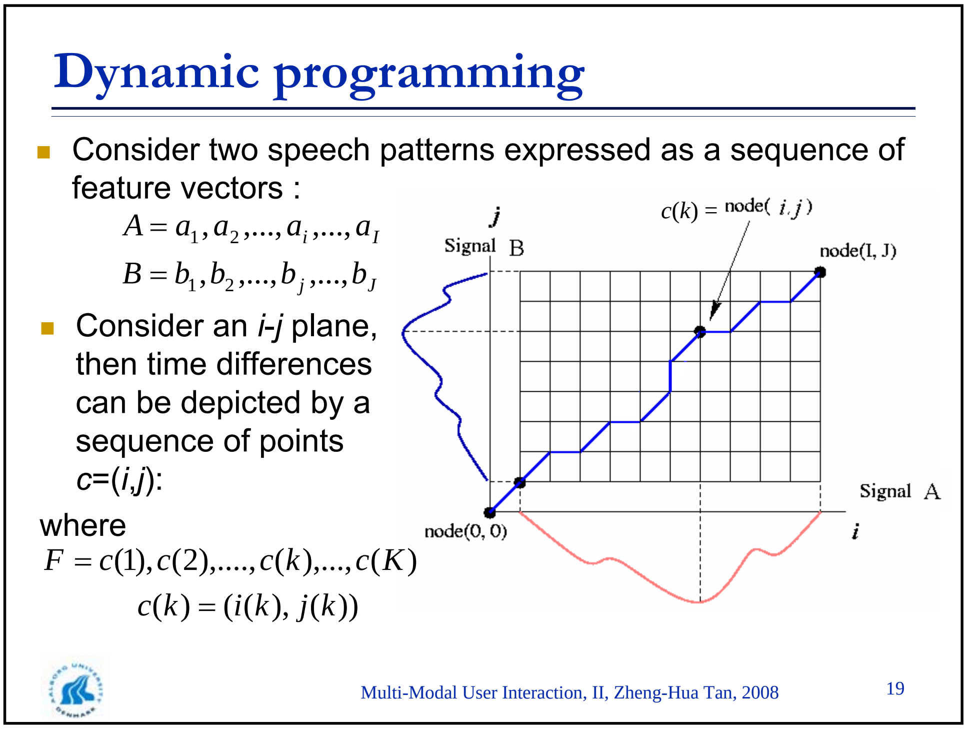
- Speaking rate variation causes nonlinear fluctuation in a speech pattern time axis


0.0 0.2 0.4 0.6 0.8 1.0
Eswin(file=C:\DOCUME~1\gym\LOCALS~1\T
Eswin(file=C:\DOCUME~1\gym\LOCALS~1\T
0.0 0.2 0.4 0.6 0.8 1.0
ile=C:\DOCUME~1\gym\LOCALS~1\Temp\tmp194BA.sfs,item=1.01,st
ile=C:\DOCUME~1\gym\LOCALS~1\Temp\tmp194BA.sfs,item=1.01,st

- Time-normalization is needed.

DP based time-normalization

- Dynamic programming is a pattern matching algorithm with a nonlinear time-normalization effect.
  - Time differences btw two speech patterns are eliminated by warping the time axis of one so that the maximum coincidence is attained with the other, also called dynamic time warping (DTW)
  - The time-normalized distance is calculated as the minimized residual distance between them, remaining still after eliminating the timing differences.



Dynamic programming

- Consider two speech patterns expressed as a sequence of feature vectors :

$$A = a_1, a_2, ..., a_i, ..., a_I$$

$$B = b_1, b_2, ..., b_j, ..., b_J$$

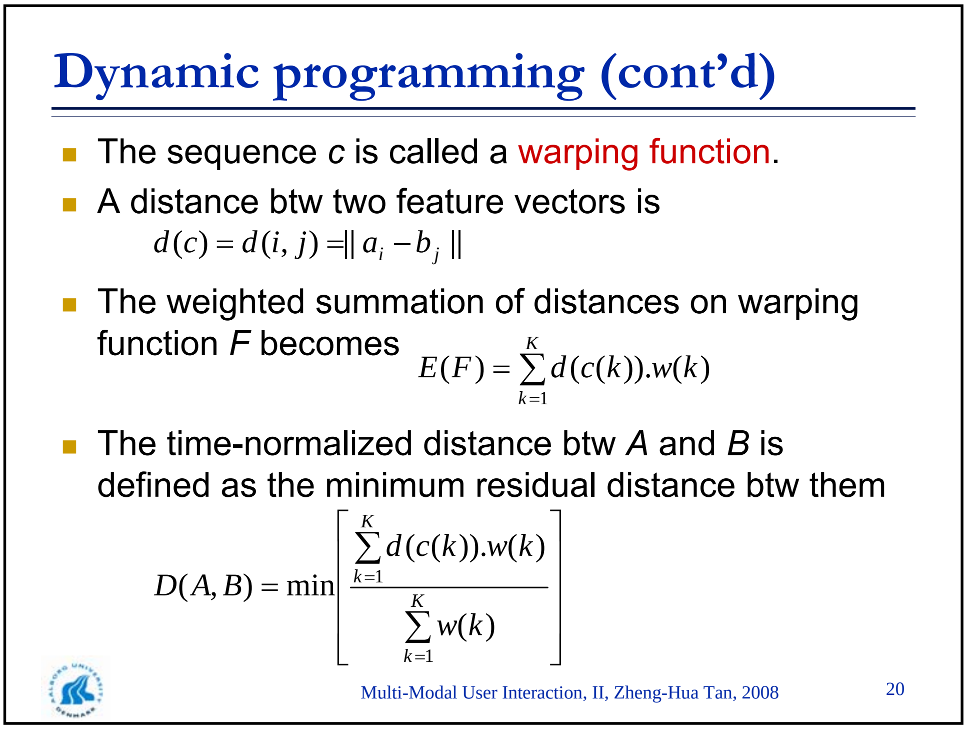
- Consider an *i-j* plane, then time differences can be depicted by a sequence of points *c*=(*i*,*j*):

where

$$F = c(1), c(2), ..., c(k), ..., c(K)$$

$$c(k) = (i(k), j(k))$$

$c(k)$ = node(i, j)
j
Signal B
node(I, J)
Signal A
node(0, 0)
i



Dynamic programming (cont'd)

- The sequence *c* is called a warping function.
- A distance btw two feature vectors is

$$d(c) = d(i, j) = \| a_i - b_j \|$$

- The weighted summation of distances on warping function *F* becomes $E(F) = \sum_{k=1}^{K} d(c(k)).w(k)$
- The time-normalized distance btw *A* and *B* is defined as the minimum residual distance btw them

$$D(A, B) = \min \left[\frac{\sum_{k=1}^{K} d(c(k)).w(k)}{\sum_{k=1}^{K} w(k)} \right]$$

Restrictions on warping function

- Warping function F (or points *c*(*k*)), as a model of time-axis fluctuation in speech, has restrictions:

1) Monotonic conditions :

$i(k-1) \le i(k)$ and $j(k-1) \le j(k)$

2) Continuity conditions :

$i(k) - i(k-1) \le 1$ and $j(k) - j(k-1) \le 1$

3 Boundary conditions :

$i(1) = 1, j(1) = 1$ and $i(K) = I, j(K) = J.$

4) Adjustment window condition

$| i(k) - j(k) | \le r$

5) Slope constraint condition :

A gradient should be neither too steep nor too gentle.

The simplest DP of symmetric form

- Step 1: Initialisation:

$$g(1,1) = 2d(1,1)$$

- Step 2: Iteration (DP equation):

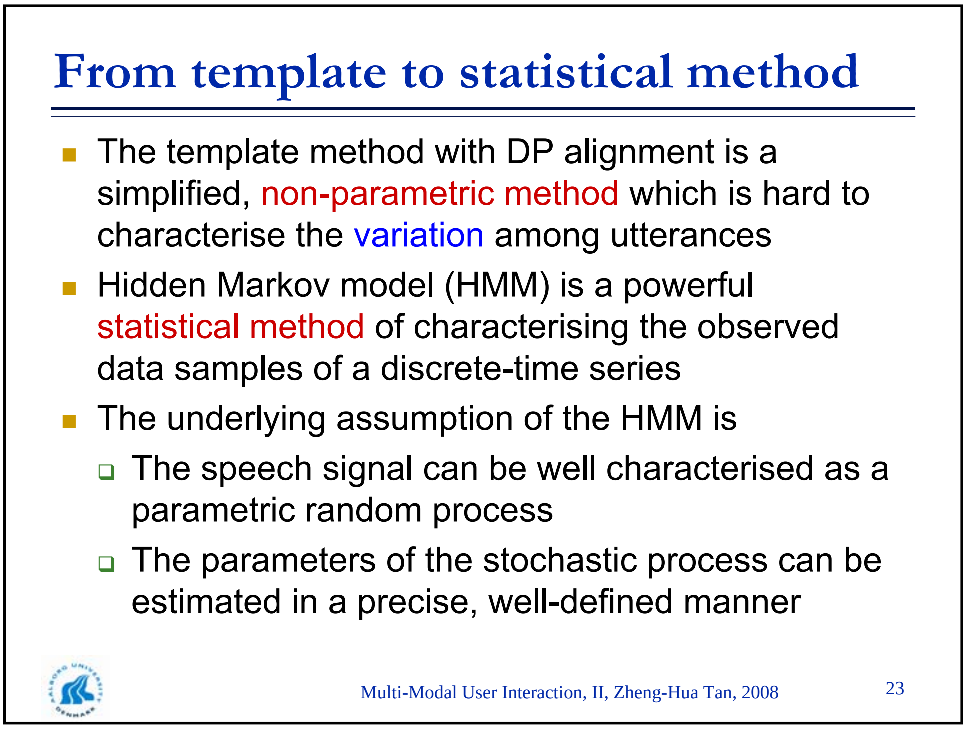
$$g(i,j) = \min \begin{bmatrix} g(i, j-1) + d(i,j) \\ g(i-1, j-1) + 2d(i,j) \\ g(i-1, j) + d(i,j) \end{bmatrix}$$

Adjustment window: $j - r \le i \le j + r$

- Step 3: Termination:

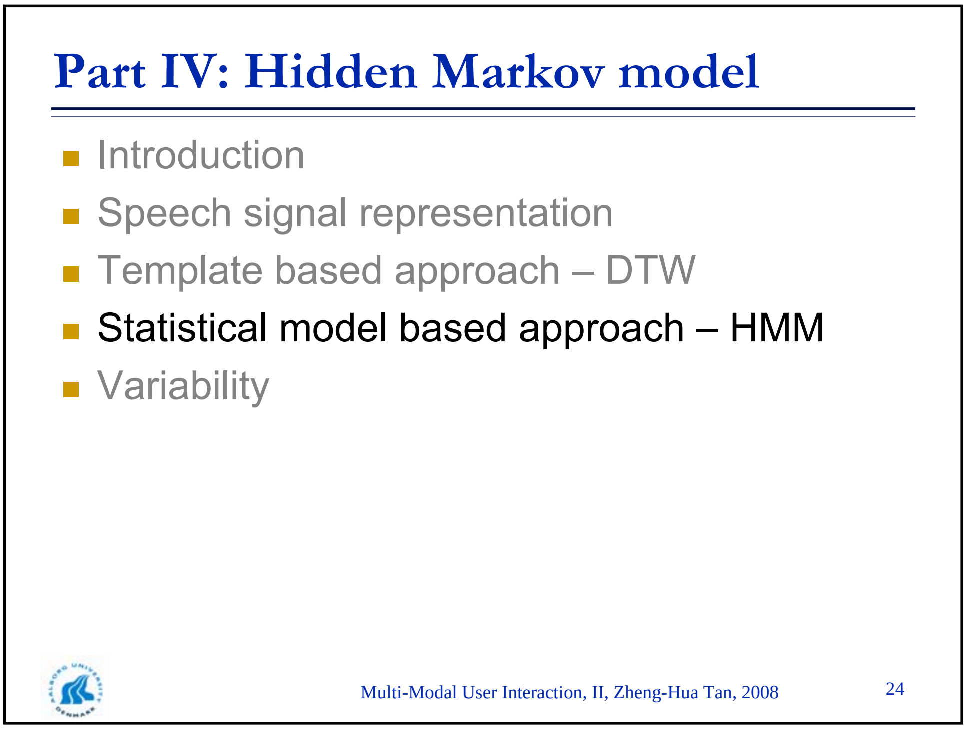
Time-normalised distance

$$D(A,B) = \frac{1}{N} g(I,J), \quad \text{where } N = I + J$$



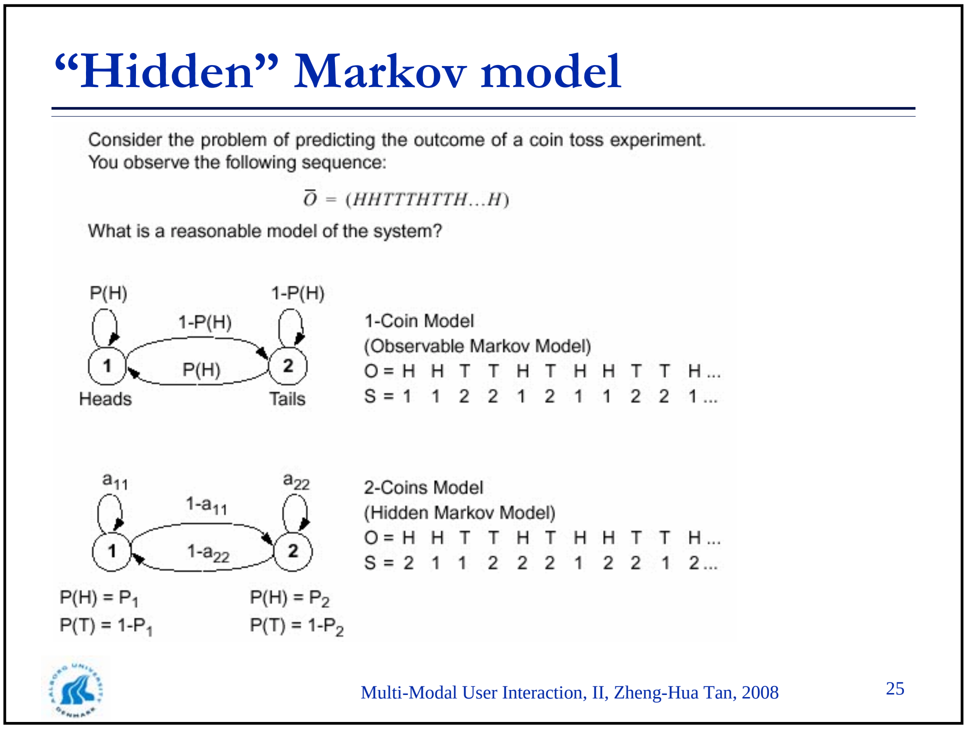
From template to statistical method

- The template method with DP alignment is a simplified, non-parametric method which is hard to characterise the variation among utterances
- Hidden Markov model (HMM) is a powerful statistical method of characterising the observed data samples of a discrete-time series
- The underlying assumption of the HMM is
  - The speech signal can be well characterised as a parametric random process
  - The parameters of the stochastic process can be estimated in a precise, well-defined manner



Part IV: Hidden Markov model

- Introduction
- Speech signal representation
- Template based approach – DTW
- Statistical model based approach – HMM
- Variability



“Hidden” Markov model

Consider the problem of predicting the outcome of a coin toss experiment. You observe the following sequence:

$$\bar{O} = (HHTTTHTTH\ldots H)$$

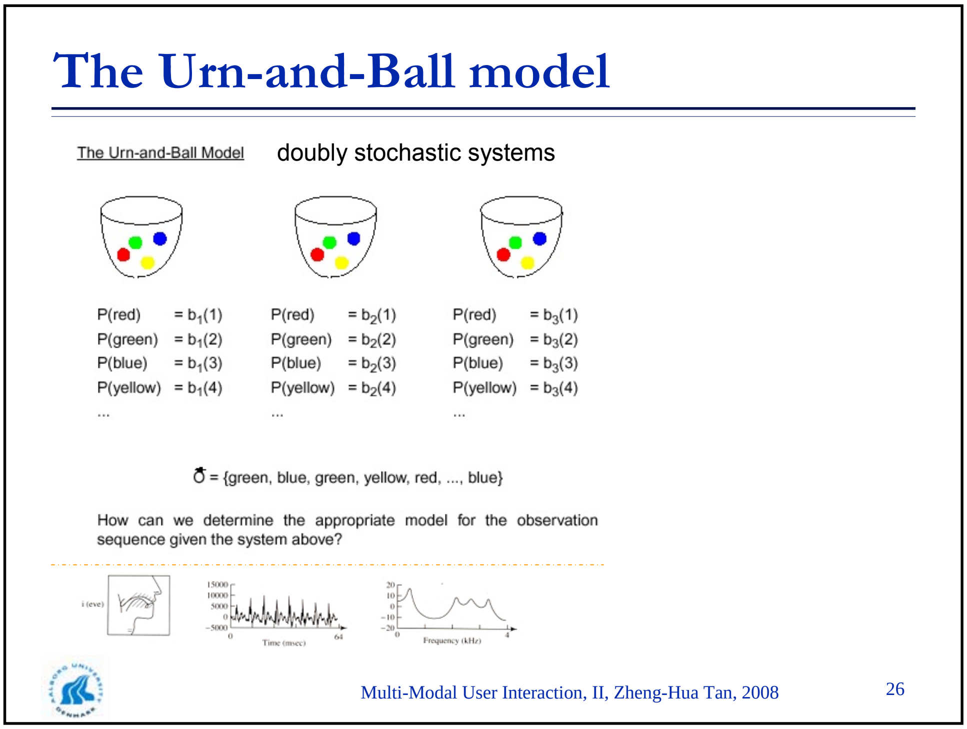
What is a reasonable model of the system?

1-Coin Model
(Observable Markov Model)
O = H H T T H T H H T T H ...
S = 1 1 2 2 1 2 1 1 2 2 1 ...

2-Coins Model
(Hidden Markov Model)
O = H H T T H T H H T T H ...
S = 2 1 1 2 2 2 1 2 2 1 2 ...

$P(H) = P_1$
$P(T) = 1-P_1$

$P(H) = P_2$
$P(T) = 1-P_2$



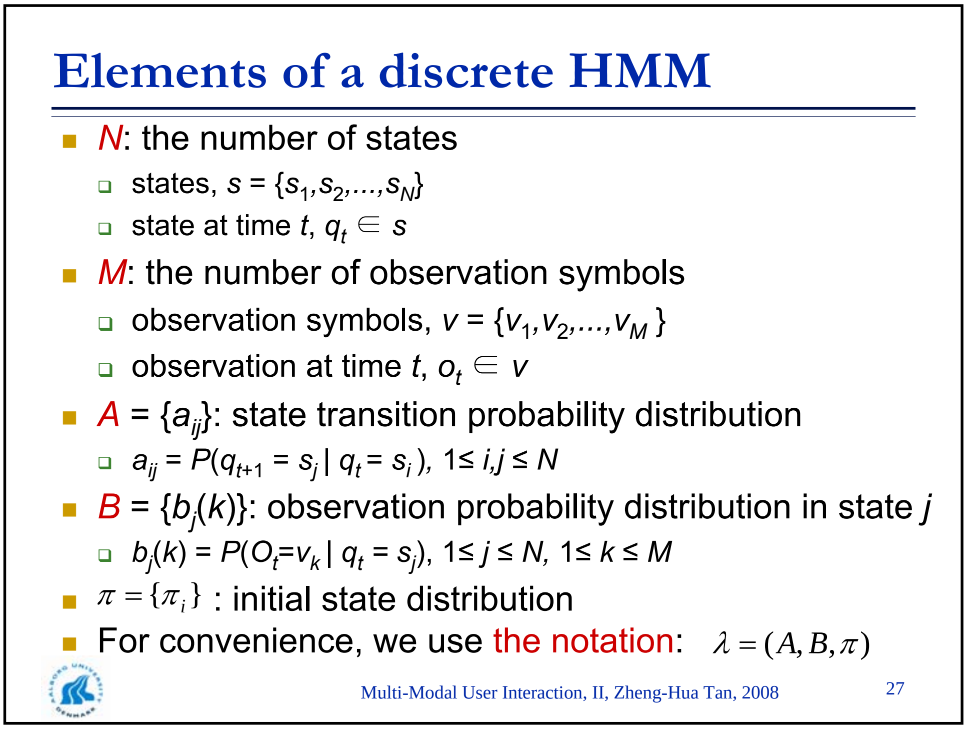
The Urn-and-Ball model

The Urn-and-Ball Model doubly stochastic systems

P(red)	$= b_1(1)$	P(red)	$= b_2(1)$	P(red)	$= b_3(1)$
P(green)	$= b_1(2)$	P(green)	$= b_2(2)$	P(green)	$= b_3(2)$
P(blue)	$= b_1(3)$	P(blue)	$= b_2(3)$	P(blue)	$= b_3(3)$
P(yellow)	$= b_1(4)$	P(yellow)	$= b_2(4)$	P(yellow)	$= b_3(4)$
...		...		...	

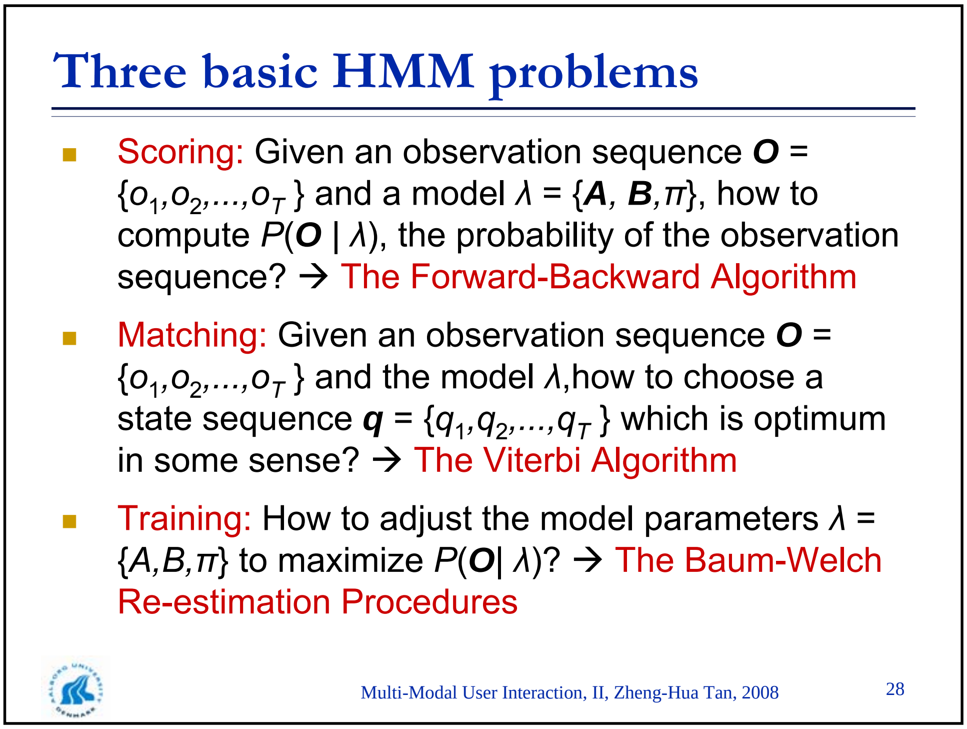
$\bar{O}$ = {green, blue, green, yellow, red, ..., blue}

How can we determine the appropriate model for the observation sequence given the system above?



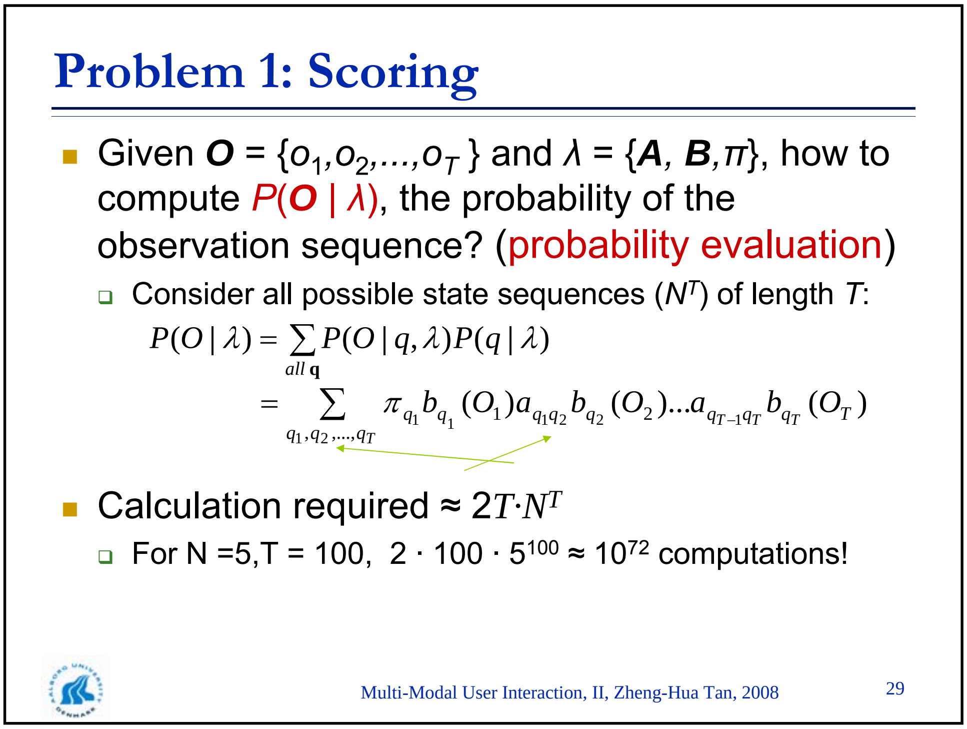
Elements of a discrete HMM

- N: the number of states
  - states, $s = \{s_1, s_2, \ldots, s_N\}$
  - state at time t, $q_t \in s$
- M: the number of observation symbols
  - observation symbols, $v = \{v_1, v_2, \ldots, v_M\}$
  - observation at time t, $o_t \in v$
- $A = \{a_{ij}\}$: state transition probability distribution
  - $a_{ij} = P(q_{t+1} = s_j \mid q_t = s_i)$, $1 \le i,j \le N$
- $B = \{b_j(k)\}$: observation probability distribution in state j
  - $b_j(k) = P(O_t = v_k \mid q_t = s_j)$, $1 \le j \le N$, $1 \le k \le M$
- $\pi = \{\pi_i\}$: initial state distribution
- For convenience, we use the notation: $\lambda = (A, B, \pi)$



Three basic HMM problems

- Scoring: Given an observation sequence $\boldsymbol{O} = \{o_1, o_2, \ldots, o_T\}$ and a model $\lambda = \{\boldsymbol{A}, \boldsymbol{B}, \pi\}$, how to compute $P(\boldsymbol{O} \mid \lambda)$, the probability of the observation sequence? → The Forward-Backward Algorithm
- Matching: Given an observation sequence $\boldsymbol{O} = \{o_1, o_2, \ldots, o_T\}$ and the model λ,how to choose a state sequence $\boldsymbol{q} = \{q_1, q_2, \ldots, q_T\}$ which is optimum in some sense? → The Viterbi Algorithm
- Training: How to adjust the model parameters $\lambda = \{A, B, \pi\}$ to maximize $P(\boldsymbol{O} \mid \lambda)$? → The Baum-Welch Re-estimation Procedures



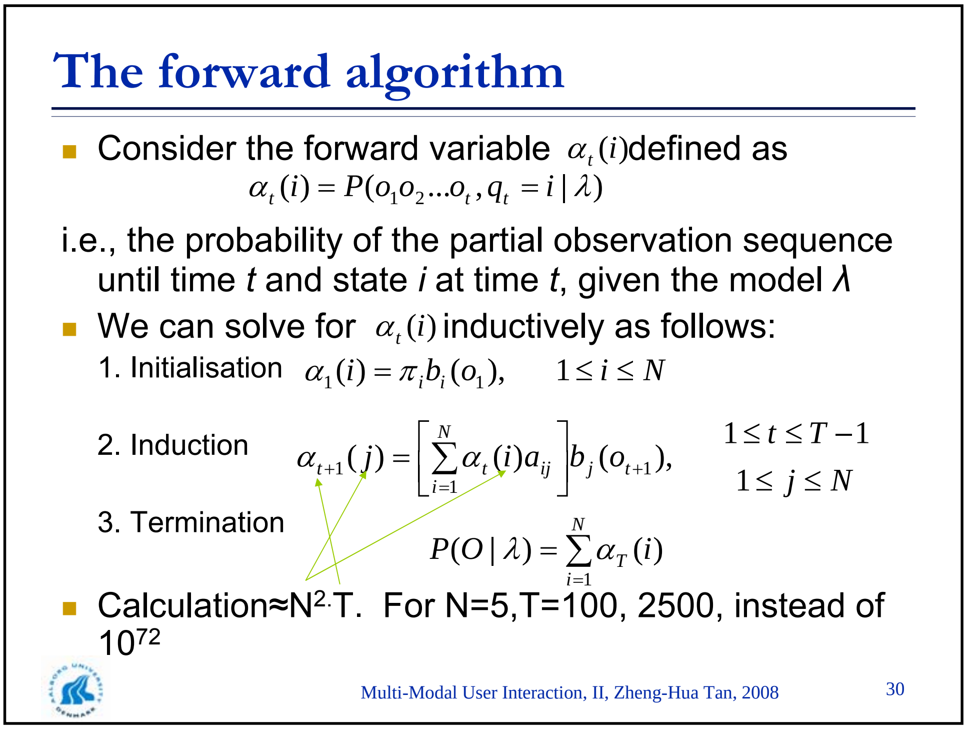
Problem 1: Scoring

- Given ***O*** = {$o_1,o_2,\ldots,o_T$ } and *λ* = {***A***, ***B***,*π*}, how to compute *P*(***O*** | *λ*), the probability of the observation sequence? (probability evaluation)
  - Consider all possible state sequences (N^T) of length *T*:

$$P(O\,|\,\lambda)=\sum_{all\ \mathbf{q}} P(O\,|\,q,\lambda)P(q\,|\,\lambda)$$

$$=\sum_{q_1,q_2,\ldots,q_T}\pi_{q_1}b_{q_1}(O_1)a_{q_1q_2}b_{q_2}(O_2)\ldots a_{q_{T-1}q_T}b_{q_T}(O_T)$$

- Calculation required ≈ $2T{\cdot}N^T$
  - For N =5,T = 100, $2 \cdot 100 \cdot 5^{100} \approx 10^{72}$ computations!



The forward algorithm

- Consider the forward variable $\alpha_t(i)$defined as

$$\alpha_t(i)=P(o_1o_2\ldots o_t,q_t=i\,|\,\lambda)$$

i.e., the probability of the partial observation sequence until time *t* and state *i* at time *t*, given the model *λ*

- We can solve for $\alpha_t(i)$ inductively as follows:
  1. Initialisation $\alpha_1(i)=\pi_i b_i(o_1),\qquad 1\le i\le N$
  2. Induction $\alpha_{t+1}(j)=\left[\sum_{i=1}^{N}\alpha_t(i)a_{ij}\right]b_j(o_{t+1}),\qquad \begin{array}{c}1\le t\le T-1\\ 1\le j\le N\end{array}$
  3. Termination $P(O\,|\,\lambda)=\sum_{i=1}^{N}\alpha_T(i)$
- Calculation≈$N^{2\cdot}$T. For N=5,T=100, 2500, instead of 10^{72}

Illustration of forward algorithm

(Rabiner, 1989)

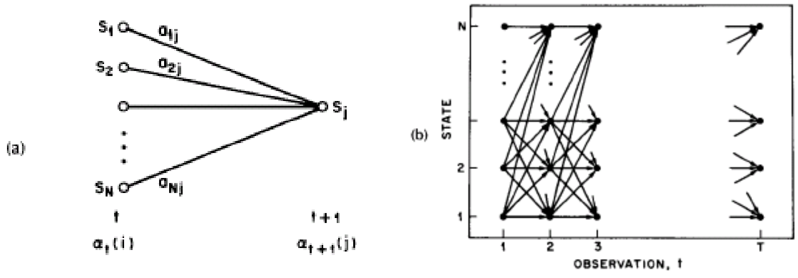


Fig. 4. (a) Illustration of the sequence of operations required for the computation of the forward variable $\alpha_{t+1}(j)$. (b) Implementation of the computation of $\alpha_t(i)$ in terms of a lattice of observations t, and states i.

The backward algorithm

- Similarly, consider the backward variable $\beta_t(i)$ defined as $\beta_t(i) = P(o_{t+1}o_{t+2}...o_T \mid q_t = i, \lambda)$

i.e., the probability of the partial observation sequence from time *t* +1 to the end, given state *i* at time *t* and model *λ*

- We can solve for $\beta_t(i)$ inductively as follows:

1. Initialisation $\beta_T(i) = 1, \qquad 1 \le i \le N$
2. Induction $\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(o_{t+1}) \beta_{t+1}(j), \qquad t = T-1, T-2, ..., 1 \quad 1 \le i \le N$
3. Termination $P(O \mid \lambda) = \sum_{i=1}^{N} \pi_i b_i(o_1) \beta_1(i)$

- Again, calculation≈$N^2 \cdot T$.

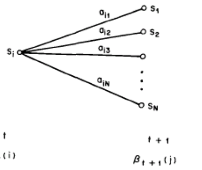
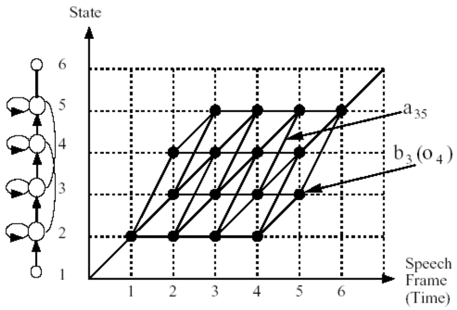


Fig. 5. Illustration of the sequence of operations required for the computation of the backward variable $\beta_t(i)$.

Problem 2: Matching

- Given ***O*** = {$o_1, o_2, \ldots, o_T$},how to choose a state sequence ***q*** = {$q_1, q_2, \ldots, q_T$} which is optimum in some sense? ("Optimal" state sequence)



Trellis diagram for an Isolated Word Recognition task.

Finding optimal state sequence

- One optimality criterion is to choose the states q_i that are individually most likely at each time *t*
  - Define the probability of being in state *i* at time *t*, given the observation sequence O, and the model λ

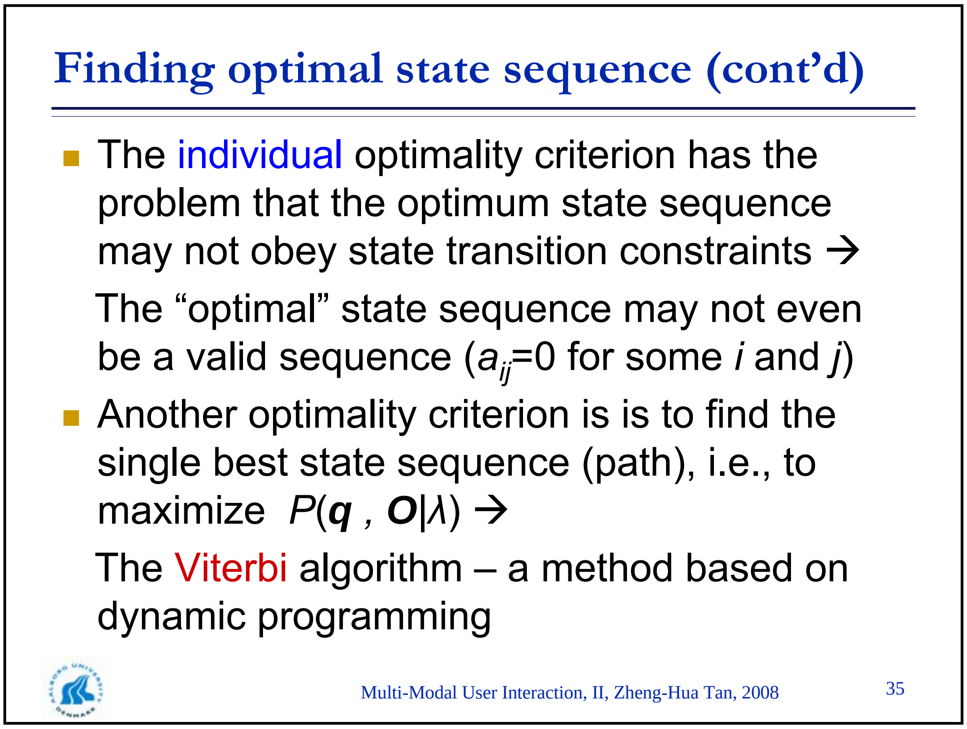
$$\gamma_t(i) = P(q_t = i \mid O, \lambda) = \frac{P(O, q_t = i \mid \lambda)}{P(O \mid \lambda)} = \frac{P(O, q_t = i \mid \lambda)}{\sum_{i=1}^{N} P(O, q_t = i \mid \lambda)}$$

Since $P(O, q_t = i \mid \lambda) = \alpha_t(i)\beta_t(i)$

We have $\gamma_t(i) = \dfrac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)}$

  - The individually most likely state q_t^* at time *t* is

$$q_t^* = \arg\max_{1 \le i \le N} [\gamma_t(i)]$$

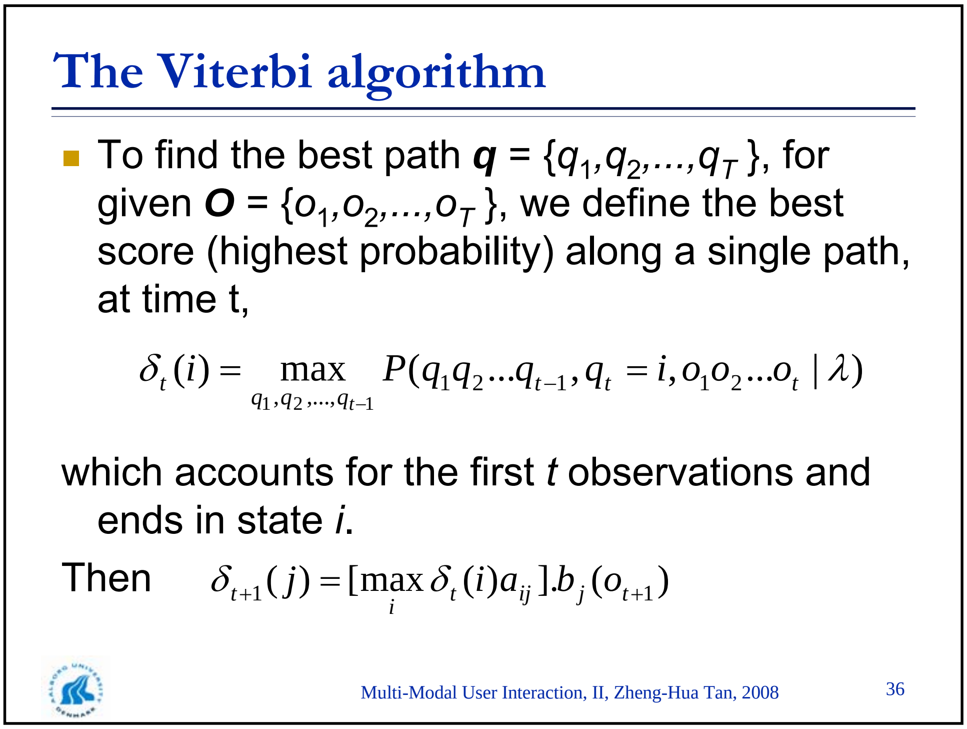


Finding optimal state sequence (cont'd)

- The individual optimality criterion has the problem that the optimum state sequence may not obey state transition constraints →

  The "optimal" state sequence may not even be a valid sequence (a_{ij}=0 for some *i* and *j*)
- Another optimality criterion is is to find the single best state sequence (path), i.e., to maximize $P(\boldsymbol{q}\ ,\ \boldsymbol{O}|\lambda)$ →

  The Viterbi algorithm – a method based on dynamic programming



The Viterbi algorithm

- To find the best path $\boldsymbol{q} = \{q_1,q_2,\ldots,q_T\}$, for given $\boldsymbol{O} = \{o_1,o_2,\ldots,o_T\}$, we define the best score (highest probability) along a single path, at time t,

$$\delta_t(i) = \max_{q_1,q_2,\ldots,q_{t-1}} P(q_1q_2\ldots q_{t-1}, q_t = i, o_1o_2\ldots o_t \mid \lambda)$$

which accounts for the first *t* observations and ends in state *i*.

Then $\delta_{t+1}(j) = [\max_i \delta_t(i)a_{ij}].b_j(o_{t+1})$

The Viterbi algorithm (cont'd)

1. Initialisation $\delta_1(i) = \pi_i b_i(o_1), \qquad 1 \le i \le N$

   $\psi_1(i) = 0$

2. Recursion

   $$\delta_t(j) = \max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}]b_j(o_t), \qquad 2 \le t \le T \quad 1 \le j \le N$$

   $$\psi_t(j) = \arg\max_{1\le i\le N}[\delta_{t-1}(i)a_{ij}], \qquad 2 \le t \le T \quad 1 \le j \le N$$

3. Termination

   $$P^* = \max_{1\le i\le N}[\delta_T(i)]$$

   $$q_T^* = \arg\max_{1\le i\le N}[\delta_T(i)]$$

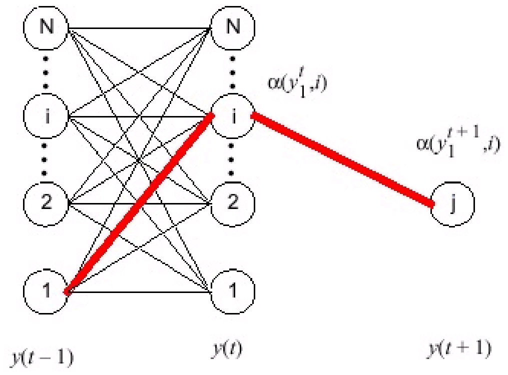
4. Path (state sequence) backtracking

   $$q_t^* = \psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, \ldots, 1$$

The Viterbi algorithm (cont'd)

(Joseph Picone)

The *Viterbi algorithm* can also be used to find the best state sequence. Note that the principal difference is that we model the overall sequence probability by the probability of the single best path:



The power of recursive equation

Computing factorials *n!*

Method 1. simply caculate $n!$ for each n

Method 2. use $n! = n(n-1)!$

$$\text{if } F(n) = n! \text{ then}$$

$$F(n) = nF(n-1) \text{ for } n \geq 1$$

(Recursive Equation)

Problem 3: Training

- How to tune the model parameters *λ* = {*A*,*B*,*π*} to maximize *P*(***O***| *λ*)? - a learning problem
  - No efficient algorithm for global optimisation
  - Effective iterative algorithm for local optimisation: the Baum-Welch re-estimation
- Baum-Welch
  - = forward-backward algorithm (Baum, 1972)
  - is a special case of EM (expectation-maximization) algorithm
  - computes probabilities using current model *λ*;
  - refines *λ* *to* $\bar{\lambda}$ such that *P*(***O***| *λ*) is locally maximised
  - uses α and β from forward-backward algorithm

Baum-Welch re-estimation

Define $\xi_t(i,j)$, the probability of being in state *i* at time *t*, and state *j* at time *t*+1, given *λ* and ***O***, i.e.

$$\begin{aligned}\xi_t(i,j) &= P(q_t = i, q_{t+1} = j \mid \mathbf{O}, \lambda)\\ &= \frac{P(q_t = i, q_{t+1} = j, \mathbf{O} \mid \lambda)}{P(\mathbf{O} \mid \lambda)}\\ &= \frac{\alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}{P(\mathbf{O} \mid \lambda)}\\ &= \frac{\alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(\mathbf{o}_{t+1}) \beta_{t+1}(j)}\end{aligned}$$

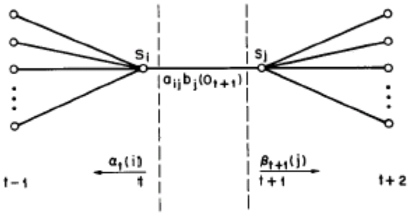


Fig. 6. Illustration of the sequence of operations required for the computation of the joint event that the system is in state S_i at time *t* and state S_j at time *t* + 1.

Baum-Welch Re-estimation (cont'd)

- Recall that $\gamma_t(i)$ is defined as the probability of being in state i at time t, given the entire observation sequence and the model, so

$$\gamma_t(i) = P(q_t = i \mid O, \lambda) = \sum_{j=1}^{N} P(q_t = i, q_{t+1} = j \mid O, \lambda) = \sum_{j=1}^{N} \xi_t(i,j)$$

- Sum $\gamma_t(i)$ and $\xi_t(i,j)$ over *t*, we have

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{expected number of transitions from state } i \text{ in } \mathbf{O}$$

$$(\sum_{t=1}^{T} \gamma_t(i) = \text{the expected number of times that state } i \text{ is visitied.})$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \text{expected number of transitions from state } i \text{ to state } j \text{ in } \mathbf{O}$$

Baum-Welch re-estimation formulas

$$\bar{\pi}_i = \text{expected frequency (number of times) in state } i$$
$$\text{at time } (t=1) = \gamma_1(i)$$

$$\bar{a}_{ij} = \frac{\text{expected number of transitions from state } i \text{ to state } j}{\text{expected number of transitions from state } i}$$

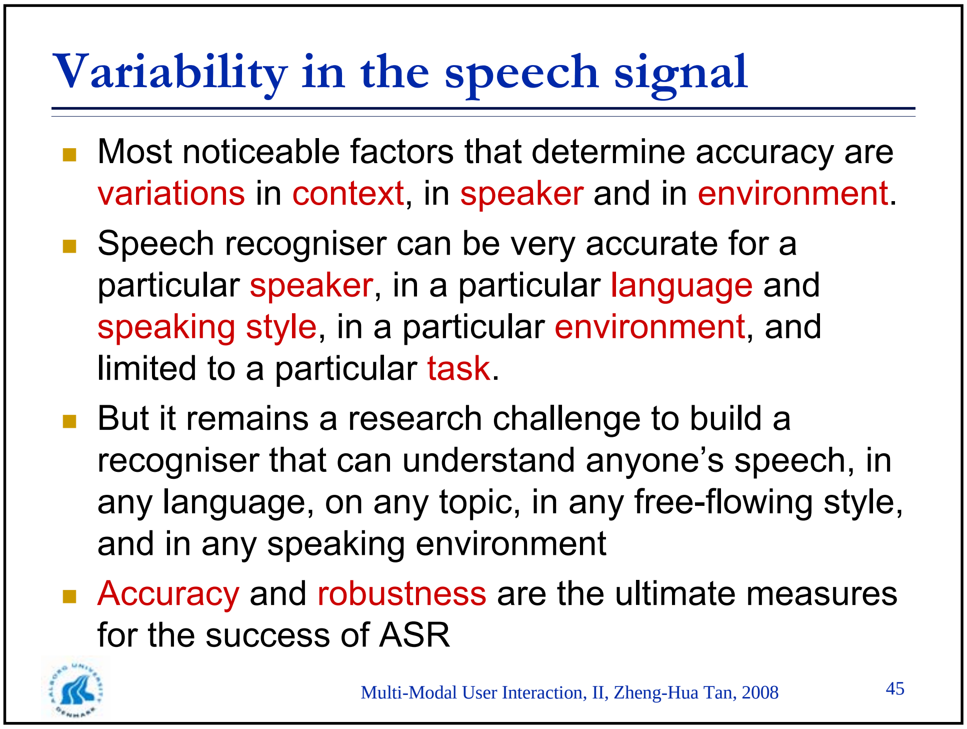
$$= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)}$$

$$\bar{b}_j(k) = \frac{\text{expected number of times in state } j \text{ and observing symbol } v_k}{\text{expected number of times in state } j}$$

$$= \frac{\sum_{\substack{t=1 \\ s.t. o_t = v_k}}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} = \frac{\sum_{t=1}^{T} P(O, q_t = i \mid \lambda) \cdot \delta(o_t, v_k)}{\sum_{t=1}^{T} P(O, q_t = i \mid \lambda)} \qquad \delta(o_t, v_k) = \begin{cases} 1 & o_t = v_k \\ 0 & \text{otherwise} \end{cases}$$

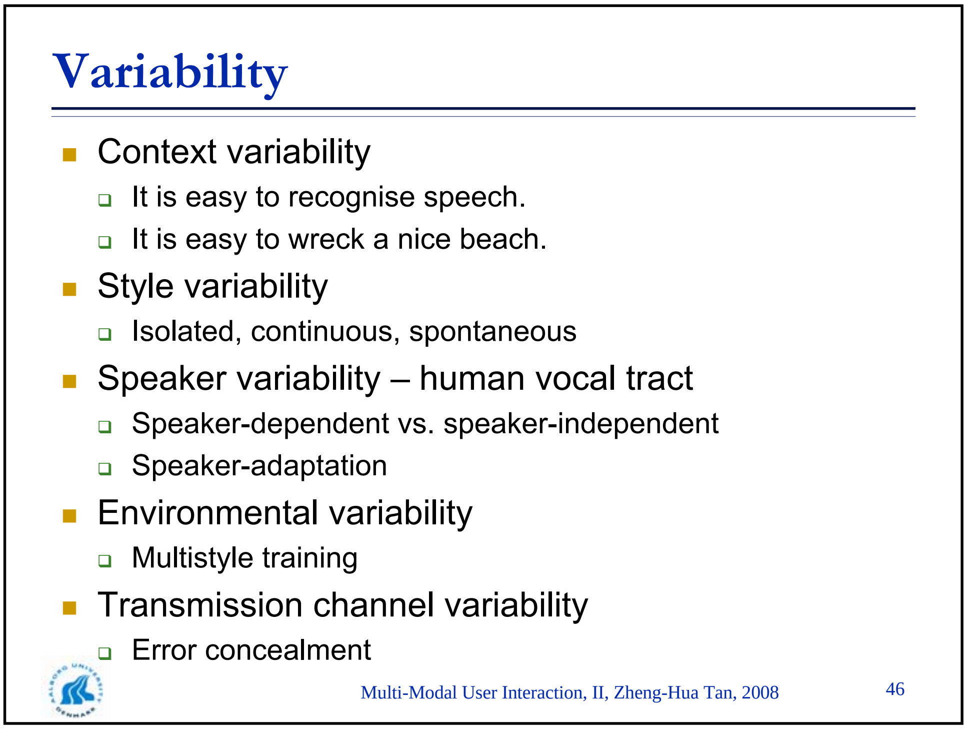
Part VI: Variability

- Introduction
- Speech signal representation
- Template based approach – DTW
- Statistical model based approach – HMM
- Variability



Variability in the speech signal

- Most noticeable factors that determine accuracy are variations in context, in speaker and in environment.
- Speech recogniser can be very accurate for a particular speaker, in a particular language and speaking style, in a particular environment, and limited to a particular task.
- But it remains a research challenge to build a recogniser that can understand anyone’s speech, in any language, on any topic, in any free-flowing style, and in any speaking environment
- Accuracy and robustness are the ultimate measures for the success of ASR



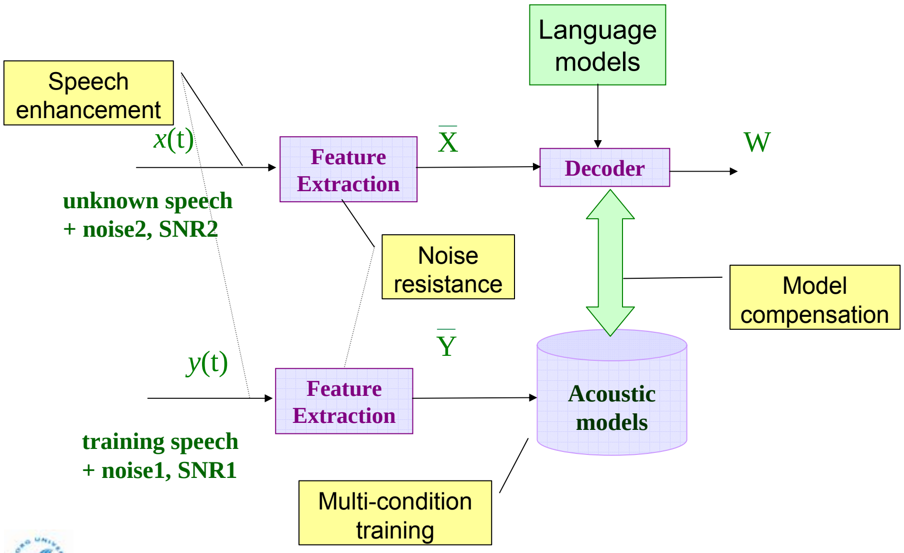
Variability

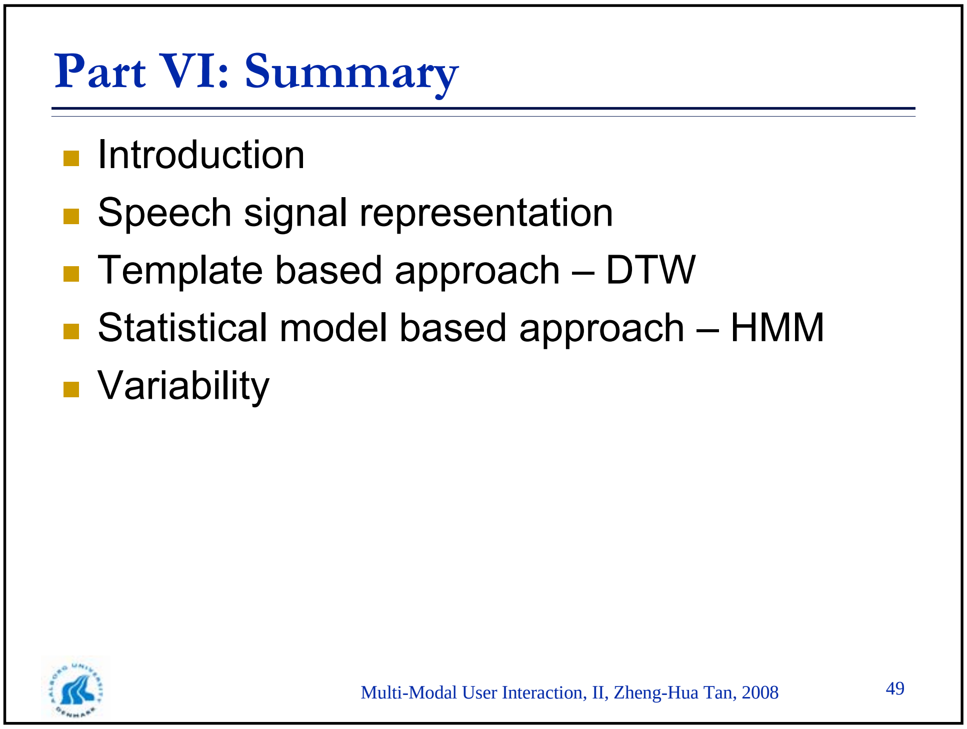
- Context variability
  - It is easy to recognise speech.
  - It is easy to wreck a nice beach.
- Style variability
  - Isolated, continuous, spontaneous
- Speaker variability – human vocal tract
  - Speaker-dependent vs. speaker-independent
  - Speaker-adaptation
- Environmental variability
  - Multistyle training
- Transmission channel variability
  - Error concealment

Problems with noise

- High-level performance in controlled environments
- Degradation in noisy situations
  - 100% to 30% accuracy in a car with 90km/h
  - 99% to 50% in a cafeteria
- Key issue: mismatch in training and operating environments

Noise robustness





Part VI: Summary

- Introduction
- Speech signal representation
- Template based approach – DTW
- Statistical model based approach – HMM
- Variability