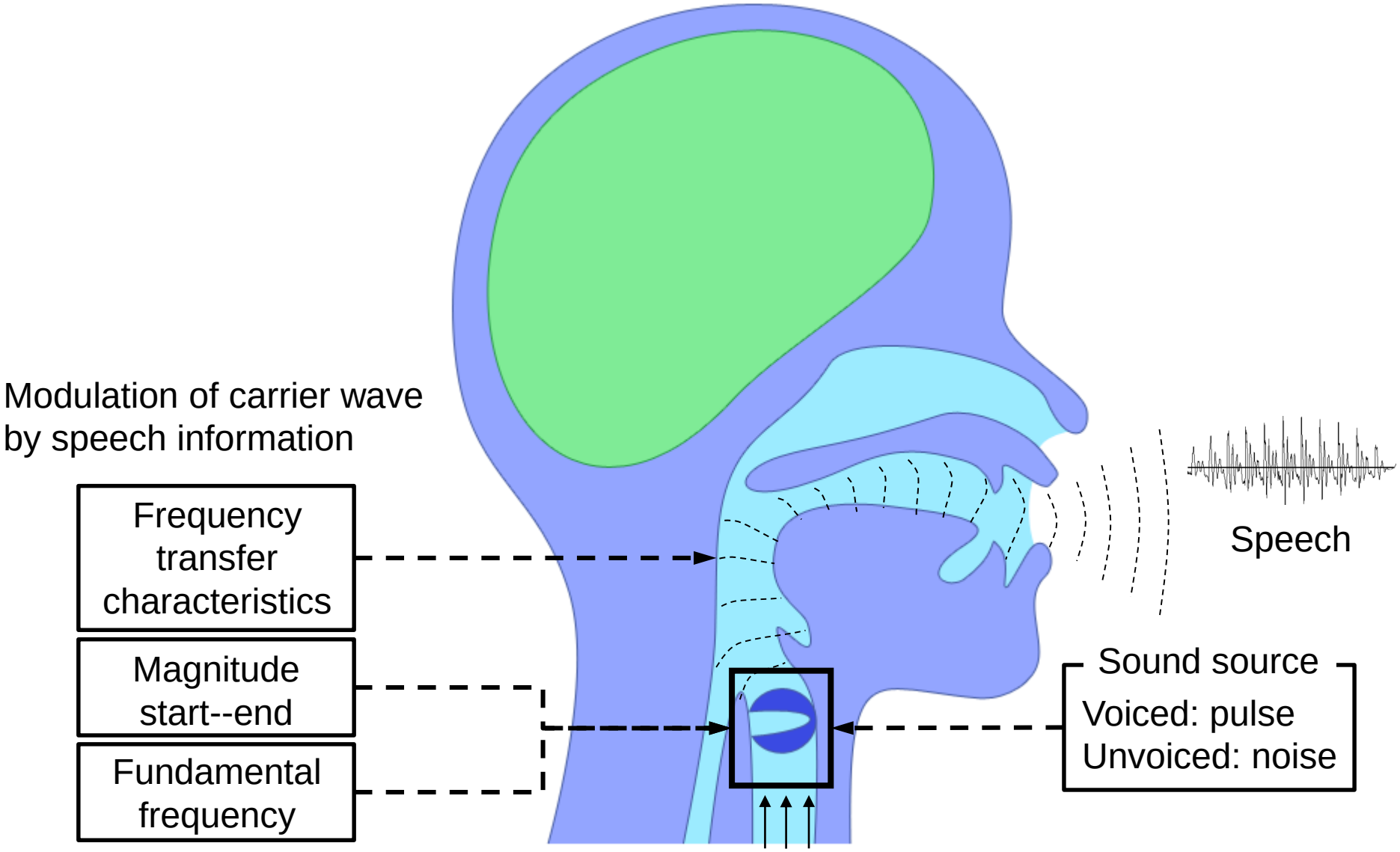
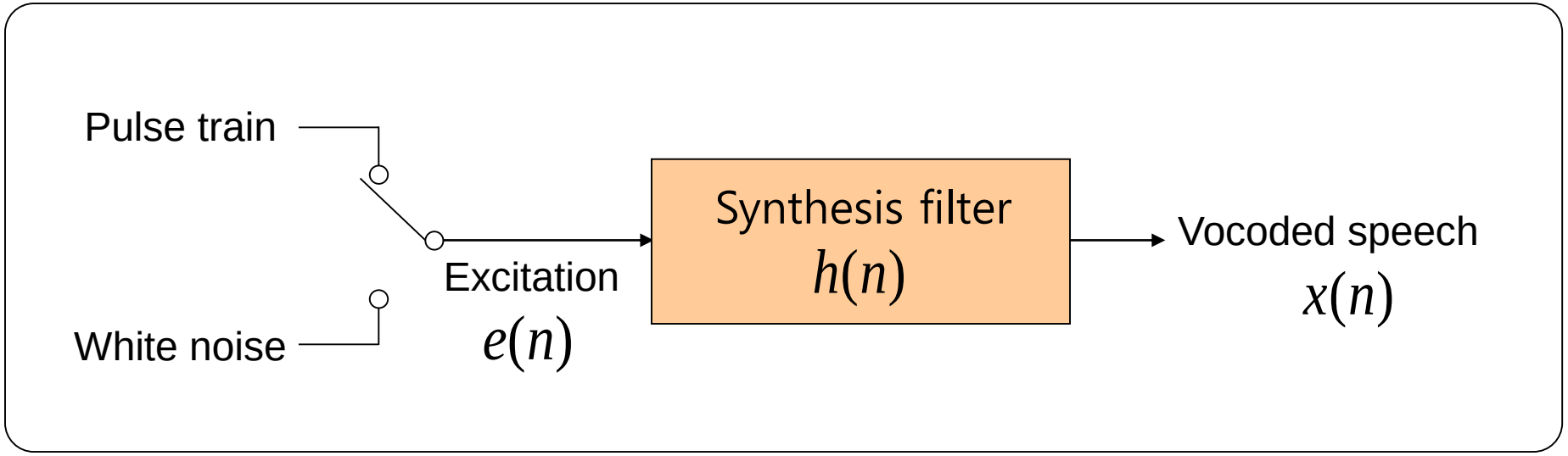
음성 합성 기술 세미나

양종열
NCSoft

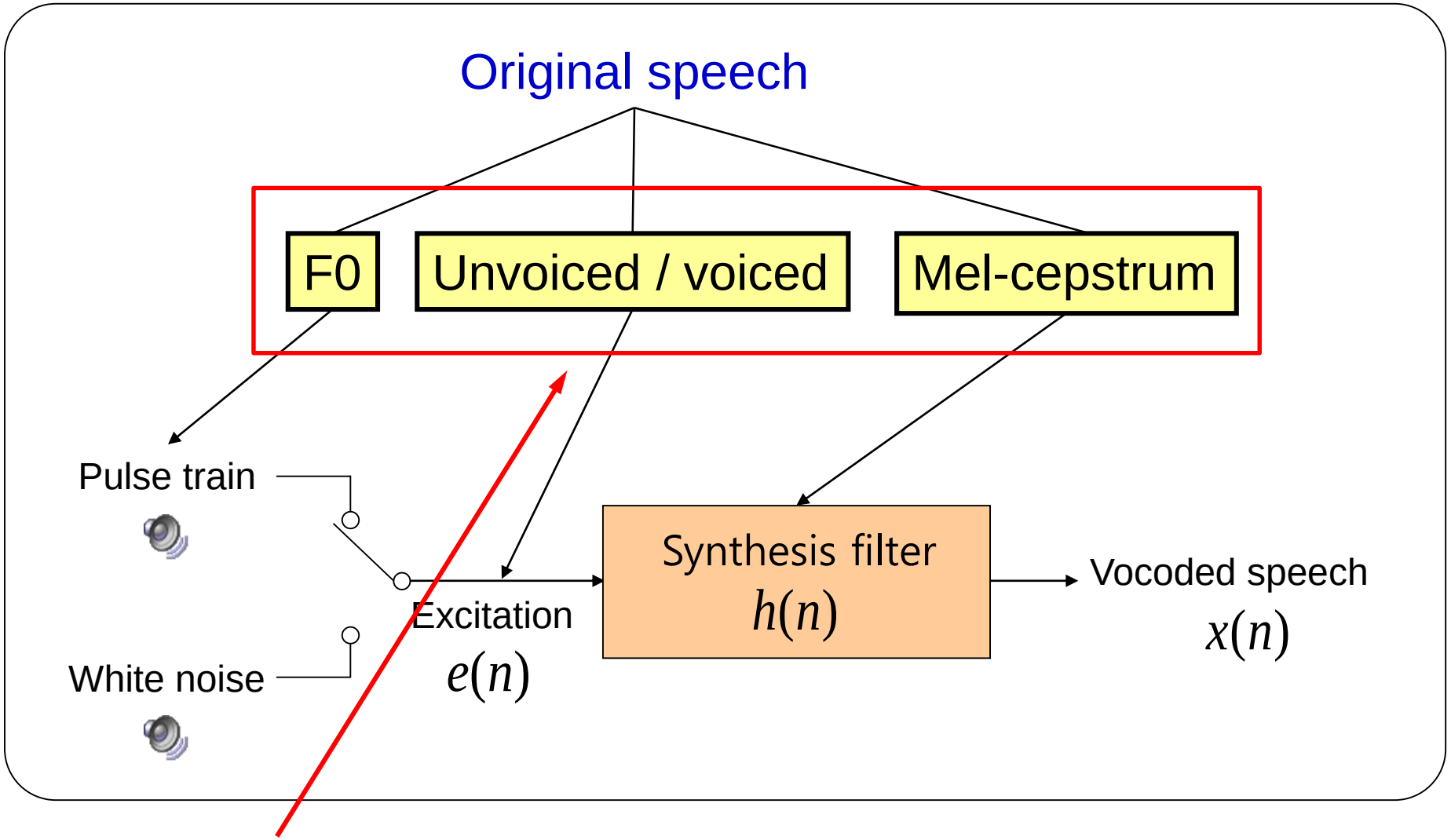
how to make speech?



Very simple structure

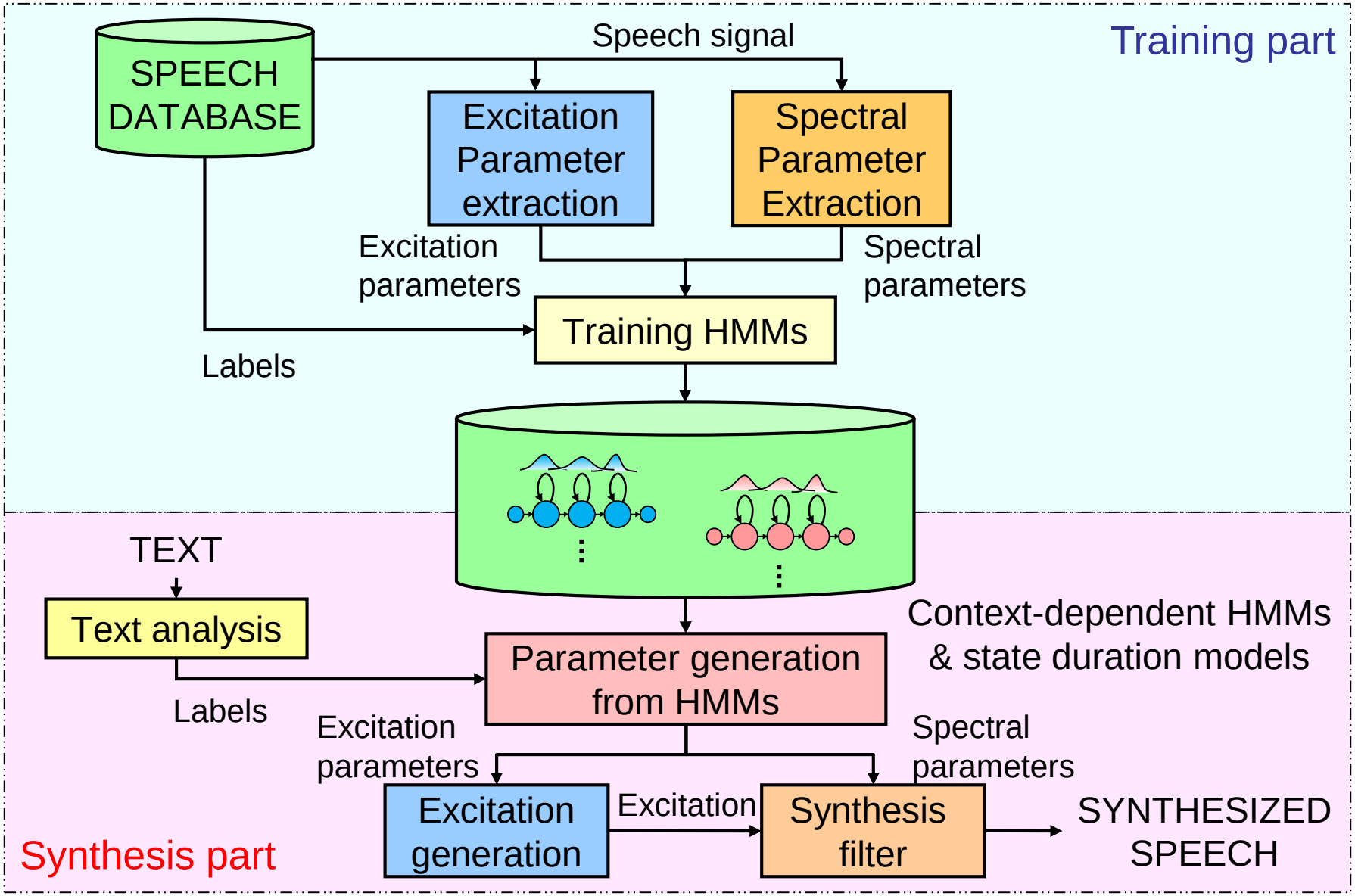


Simple structure



These speech parameters modeled by HMM

Parametric TTS 기본 구조



학습과정 (monophone)

Speech data & labels

↓

Compute variance floor (HCompV)

Calculate minimum variance of speech parameter

↓

Initialize CI-HMMs by segmental k-means (HInit)

Create monophone HMMs for all phonemes and initialize these parameters

↓

Reestimate CI-HMMs by EM algorithm (HRest & HERest)

Estimate parameters with training data (Context-independent : CI)

↓

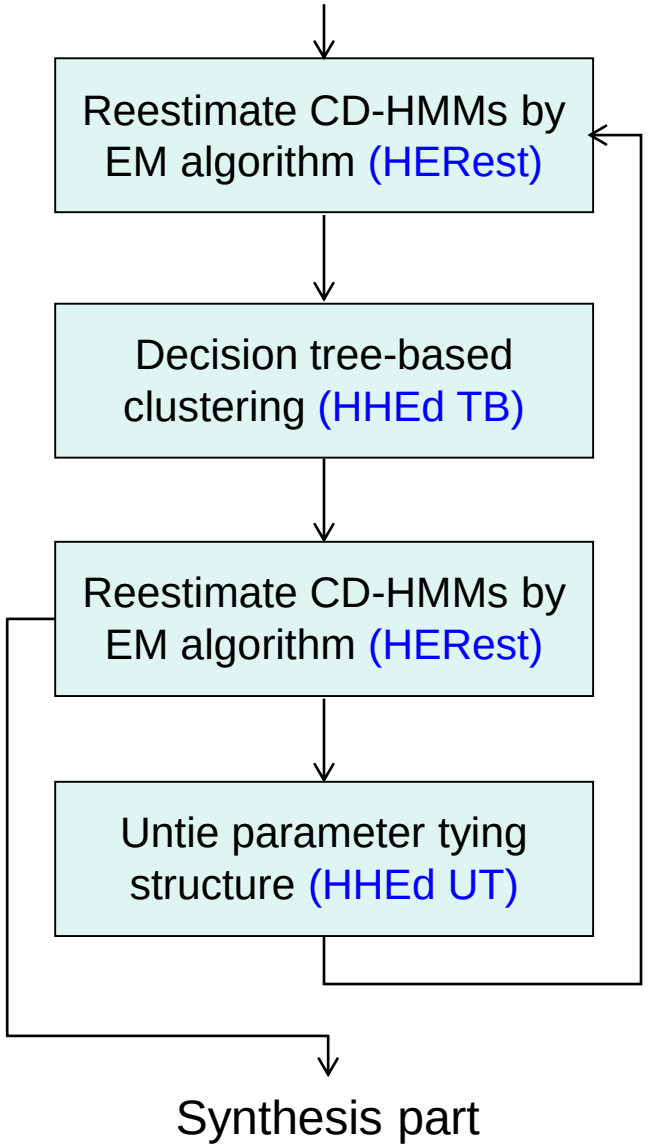
Copy CI-HMMs to CD-HMMs (HHEd CL)

Concatenate monophone HMMs according to fullcontext labels (Context-dependent : CD)

↓

Training fullcontext HMMs

학습과정 (fullcontext)



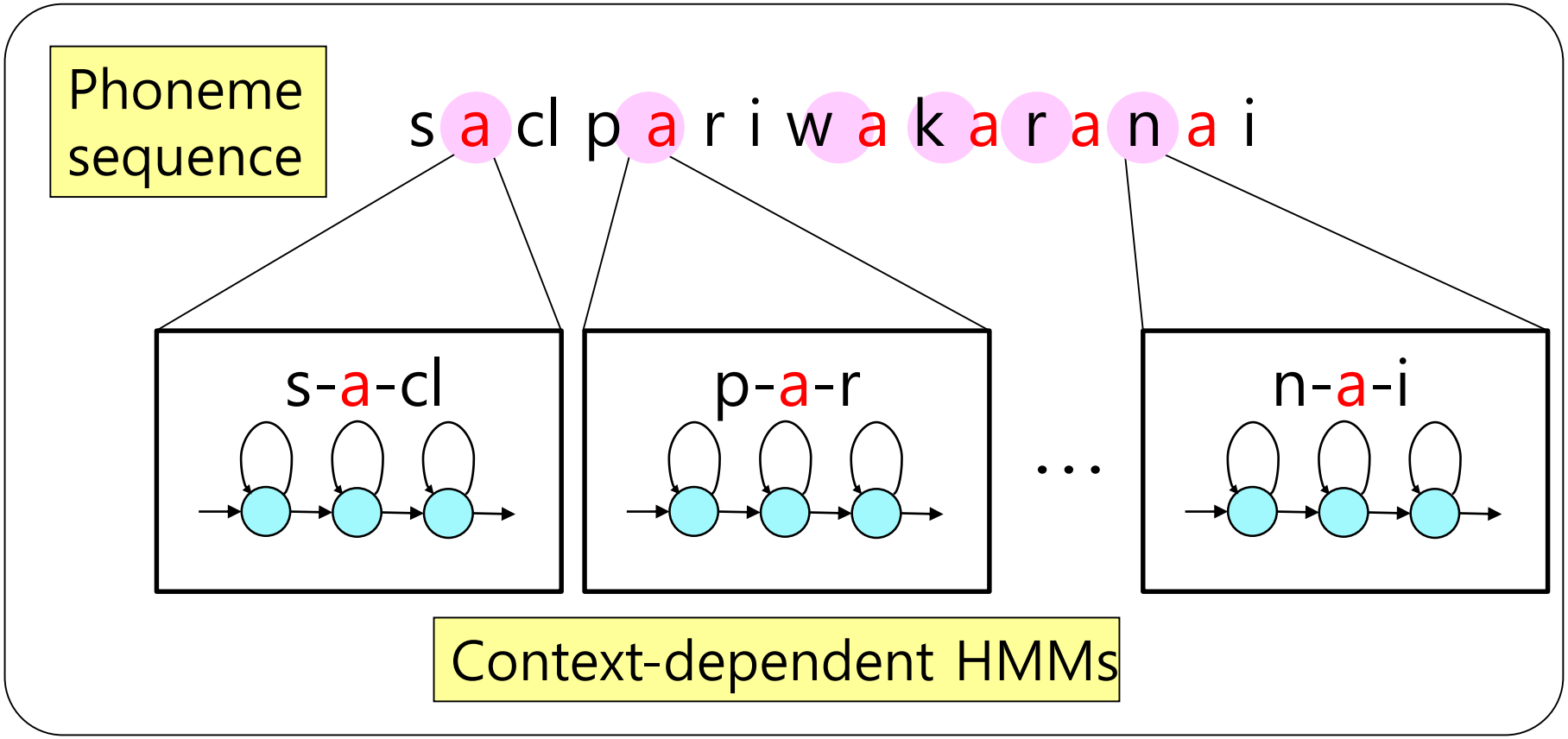
Estimate parameters with training data
(parameters are not shared)

Construct a decision tree by
the context clustering technique
(Use TB command)

Estimate parameters with training data
(parameters are shared)

Untying shared structure
(Use UT command)

Context-dependent model



- Considering relations between phonemes
  - Context ⇒ factor of speech variations
  - Improving model accuracy

Context-dependent modeling

Phoneme

- {preceding, succeeding} two phonemes
- current phoneme

Syllable

- # of phonemes at {preceding, current, succeeding} syllable
- {accent, stress} of {preceding, current, succeeding} syllable
- Position of current syllable in current word
- # of {preceding, succeeding} {accented, stressed} syllable in current phrase
- # of syllables {from previous, to next} {accented, stressed} syllable
- Vowel within current syllable

Word

- Part of speech of {preceding, current, succeeding} word
- # of syllables in {preceding, current, succeeding} word
- Position of current word in current phrase
- # of {preceding, succeeding} content words in current phrase
- # of words {from previous, to next} content word

Phrase

- # of syllables in {preceding, current, succeeding} phrase

Huge # of combinations ⇒ Difficult to have all possible models

An example of context-dependent label format for HMM-based speech synthesis in English

HTS Working Group

December 25, 2015

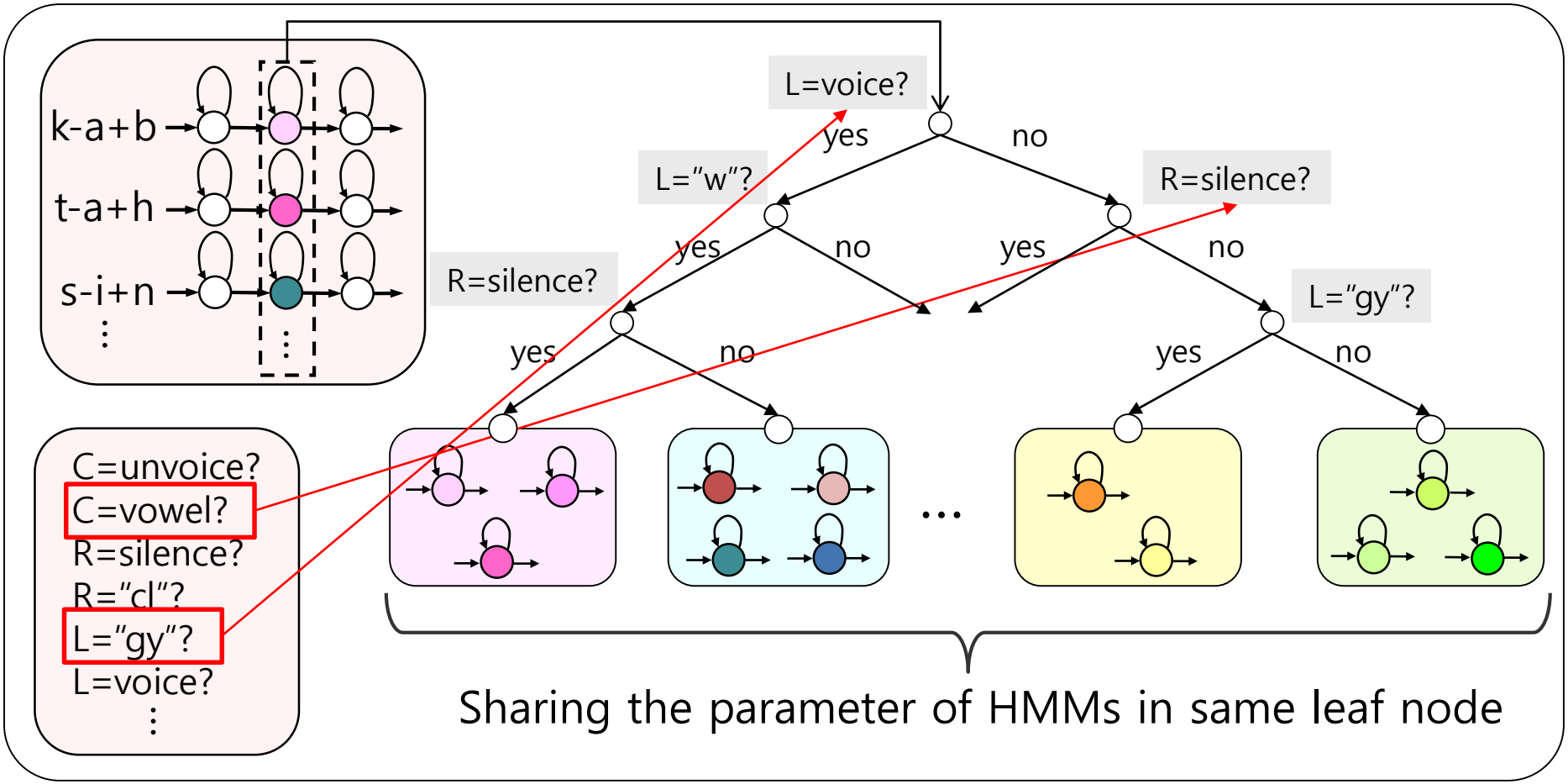
p_1^p_2-p_3+p_4=p_5@p_6_p_7
/A:a_1_a_2_a_3 /B:b_1-b_2-b_3@b_4-b_5&b_6-b_7#b_8-$b_9$$$b_{10}$-$b_{11}$!$b_{12}$-$b_{13}$;$b_{14}$-$b_{15}$|$b_{16}$ /C:c_1+c_2+c_3
/D:d_1_d_2 /E:e_1+e_2@e_3+e_4&e_5+e_6#e_7+e_8 /F:f_1_f_2
/G:g_1_g_2 /H:h_1=h_2^h_3=h_4|h_5 /I:i_1=i_2
/J:j_1+j_2-j_3

p_1	the phoneme identity before the previous phoneme
p_2	the previous phoneme identity
p_3	the current phoneme identity
p_4	the next phoneme identity
p_5	the phoneme after the next phoneme identity
p_6	position of the current phoneme identity in the current syllable (forward)
p_7	position of the current phoneme identity in the current syllable (backward)
a_1	whether the previous syllable stressed or not (0: not stressed, 1: stressed)
a_2	whether the previous syllable accented or not (0: not accented, 1: accented)
a_3	the number of phonemes in the previous syllable
b_1	whether the current syllable stressed or not (0: not stressed, 1: stressed)
b_2	whether the current syllable accented or not (0: not accented, 1: accented)
b_3	the number of phonemes in the current syllable
b_4	position of the current syllable in the current word (forward)
b_5	position of the current syllable in the current word (backward)

Label File

```
x^x-pau+ae=l@x_x/A:0_0_0/B:x-x-x@x-x&x-x#x-x$x-x!x-x;x-x|x/C:1+1+2/D:0_0/E:x+x@x+x&x+x#x+x/F:content_2/G:0_0/H:x=x^1=10|0/I:19=12/J:79+57-10
x^pau-ae+l=ax@1_2/A:0_0_0/B:1-1-2@1-2&1-19#1-10$1-5!0-2;0-8|ae/C:0+0+2/D:0_0/E:content+2@1+12&1+6#0+2/F:aux_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
pau^ae-l+ax=s@2_1/A:0_0_0/B:1-1-2@1-2&1-19#1-10$1-5!0-2;0-8|ae/C:0+0+2/D:0_0/E:content+2@1+12&1+6#0+2/F:aux_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
ae^l-ax+s=w@1_2/A:1_1_2/B:0-0-2@2-1&2-18#1-10$1-5!1-1;1-7|ax/C:1+0+3/D:0_0/E:content+2@1+12&1+6#0+2/F:aux_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
l^ax-s+w=aa@2_1/A:1_1_2/B:0-0-2@2-1&2-18#1-10$1-5!1-1;1-7|ax/C:1+0+3/D:0_0/E:content+2@1+12&1+6#0+2/F:aux_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
ax^s-w+aa=z@1_3/A:0_0_2/B:1-0-3@1-1&3-17#1-9$1-5!2-2;2-6|aa/C:0+0+3/D:content_2/E:aux+1@2+11&2+6#1+1/F:content_3/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
s^w-aa+z=b@2_2/A:0_0_2/B:1-0-3@1-1&3-17#1-9$1-5!2-2;2-6|aa/C:0+0+3/D:content_2/E:aux+1@2+11&2+6#1+1/F:content_3/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
w^aa-z+b=ih@3_1/A:0_0_2/B:1-0-3@1-1&3-17#1-9$1-5!2-2;2-6|aa/C:0+0+3/D:content_2/E:aux+1@2+11&2+6#1+1/F:content_3/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
aa^z-b+ih=g@1_3/A:1_0_3/B:0-0-3@1-3&4-16#2-9$1-5!1-1;3-5|ih/C:1+0+2/D:aux_1/E:content+3@3+10&2+5#2+2/F:to_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
z^b-ih+g=ih@2_2/A:1_0_3/B:0-0-3@1-3&4-16#2-9$1-5!1-1;3-5|ih/C:1+0+2/D:aux_1/E:content+3@3+10&2+5#2+2/F:to_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
b^ih-g+ih=n@3_1/A:1_0_3/B:0-0-3@1-3&4-16#2-9$1-5!1-1;3-5|ih/C:1+0+2/D:aux_1/E:content+3@3+10&2+5#2+2/F:to_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
ih^g-ih+n=ih@1_2/A:0_0_3/B:1-0-2@2-2&5-15#2-8$1-5!2-3;4-4|ih/C:0+0+2/D:aux_1/E:content+3@3+10&2+5#2+2/F:to_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
g^ih-n+ih=ng@2_1/A:0_0_3/B:1-0-2@2-2&5-15#2-8$1-5!2-3;4-4|ih/C:0+0+2/D:aux_1/E:content+3@3+10&2+5#2+2/F:to_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
ih^n-ih+ng=t@1_2/A:1_0_2/B:0-0-2@3-1&6-14#3-8$1-5!1-2;5-3|ih/C:0+0+2/D:aux_1/E:content+3@3+10&2+5#2+2/F:to_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
n^ih-ng+t=ax@2_1/A:1_0_2/B:0-0-2@3-1&6-14#3-8$1-5!1-2;5-3|ih/C:0+0+2/D:aux_1/E:content+3@3+10&2+5#2+2/F:to_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
ih^ng-t+ax=g@1_2/A:0_0_2/B:0-0-2@1-1&7-13#3-8$1-5!2-1;6-2|ax/C:1+0+3/D:content_3/E:to+1@4+9&3+5#1+1/F:content_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
ng^t-ax+g=eh@2_1/A:0_0_2/B:0-0-2@1-1&7-13#3-8$1-5!2-1;6-2|ax/C:1+0+3/D:content_3/E:to+1@4+9&3+5#1+1/F:content_1/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
t^ax-g+eh=t@1_3/A:0_0_2/B:1-0-3@1-1&8-12#3-7$1-5!3-1;7-1|eh/C:1+1+3/D:to_1/E:content+1@5+8&3+4#2+1/F:content_2/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
ax^g-eh+t=v@2_2/A:0_0_2/B:1-0-3@1-1&8-12#3-7$1-5!3-1;7-1|eh/C:1+1+3/D:to_1/E:content+1@5+8&3+4#2+1/F:content_2/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
g^eh-t+v=eh@3_1/A:0_0_2/B:1-0-3@1-1&8-12#3-7$1-5!3-1;7-1|eh/C:1+1+3/D:to_1/E:content+1@5+8&3+4#2+1/F:content_2/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
eh^t-v+eh=r@1_3/A:1_0_3/B:1-1-3@1-2&9-11#4-6$1-4!1-2;8-5|eh/C:0+0+1/D:content_1/E:content+2@6+7&4+3#1+1/F:content_2/G:0_0/H:19=12^1=10|L-H%/I:3=3/J:79+57-10
```

Decision tree-based state clustering

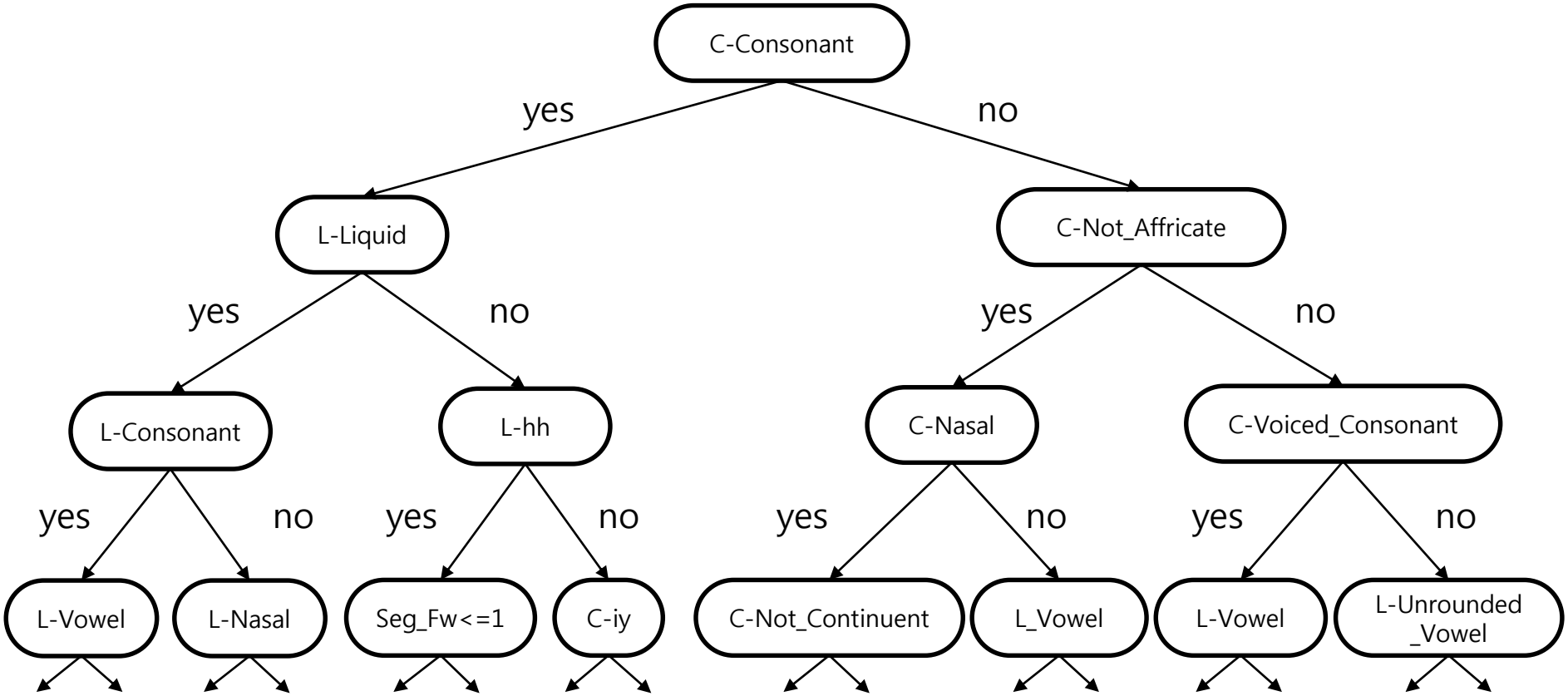


– Each state separated automatically by the optimum question
– The optimum question determined for increasing likelihood

Question File

```
QS "LL-Vowel"                {aa^*,ae^*,ah^*,ao^*,aw^*,ax^*,axr^*,
QS "LL-Consonant"            {b^*,ch^*,d^*,dh^*,dx^*,f^*,g^*,hh^*,
QS "LL-Stop"                 {b^*,d^*,dx^*,g^*,k^*,p^*,t^*}
QS "LL-Nasal"                {m^*,n^*,en^*,ng^*}
QS "LL-Fricative"            {ch^*,dh^*,f^*,hh^*,hv^*,s^*,sh^*,th^
QS "LL-Liquid"               {el^*,hh^*,l^*,r^*,w^*,y^*}
QS "LL-Front"                {ae^*,b^*,eh^*,em^*,f^*,ih^*,ix^*,iy^
QS "LL-Central"              {ah^*,ao^*,axr^*,d^*,dh^*,dx^*,el^*,e
QS "LL-Back"                 {aa^*,ax^*,ch^*,g^*,hh^*,jh^*,k^*,ng^
QS "LL-Front_Vowel"          {ae^*,eh^*,ey^*,ih^*,iy^*}
QS "LL-Central_Vowel"        {aa^*,ah^*,ao^*,axr^*,er^*}
QS "LL-Back_Vowel"           {ax^*,ow^*,uh^*,uw^*}
QS "LL-Long_Vowel"           {ao^*,aw^*,el^*,em^*,en^*,en^*,iy^*,o
QS "LL-Short_Vowel"          {aa^*,ah^*,ax^*,ay^*,eh^*,ey^*,ih^*,i
QS "LL-Dipthong_Vowel"       {aw^*,axr^*,ay^*,el^*,em^*,en^*,er^*,
QS "LL-Front_Start_Vowel"    {aw^*,axr^*,er^*,ey^*}
QS "LL-Fronting_Vowel"       {ay^*,ey^*,oy^*}
QS "LL-High_Vowel"           {ih^*,ix^*,iy^*,uh^*,uw^*}
QS "LL-Medium_Vowel"         {ae^*,ah^*,ax^*,axr^*,eh^*,el^*,em^*,
QS "LL-Low_Vowel"            {aa^*,ae^*,ah^*,ao^*,aw^*,ay^*,oy^*}
QS "LL-Rounded_Vowel"        {ao^*,ow^*,oy^*,uh^*,uw^*,w^*}
QS "LL-Unrounded_Vowel"      {aa^*,ae^*,ah^*,aw^*,ax^*,axr^*,ay^*,
QS "LL-Reduced_Vowel"        {ax^*,axr^*,ix^*}
QS "LL-IVowel"               {ih^*,ix^*,iy^*}
QS "LL-EVowel"               {eh^*,ey^*}
QS "LL-AVowel"               {aa^*,ao^*,aw^*,axr^*,ay^*,er^*}
```

Tree for Spectrum (1st state)

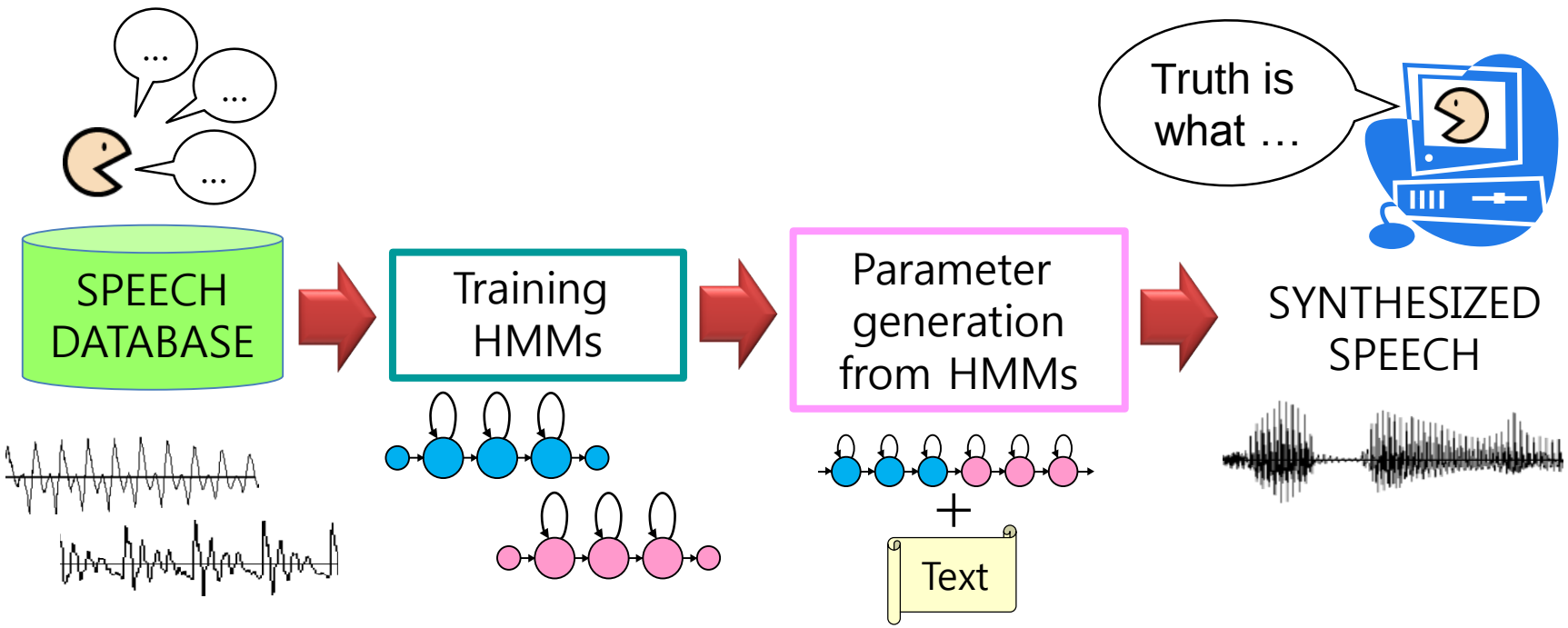


–Questions about phonetic attributes

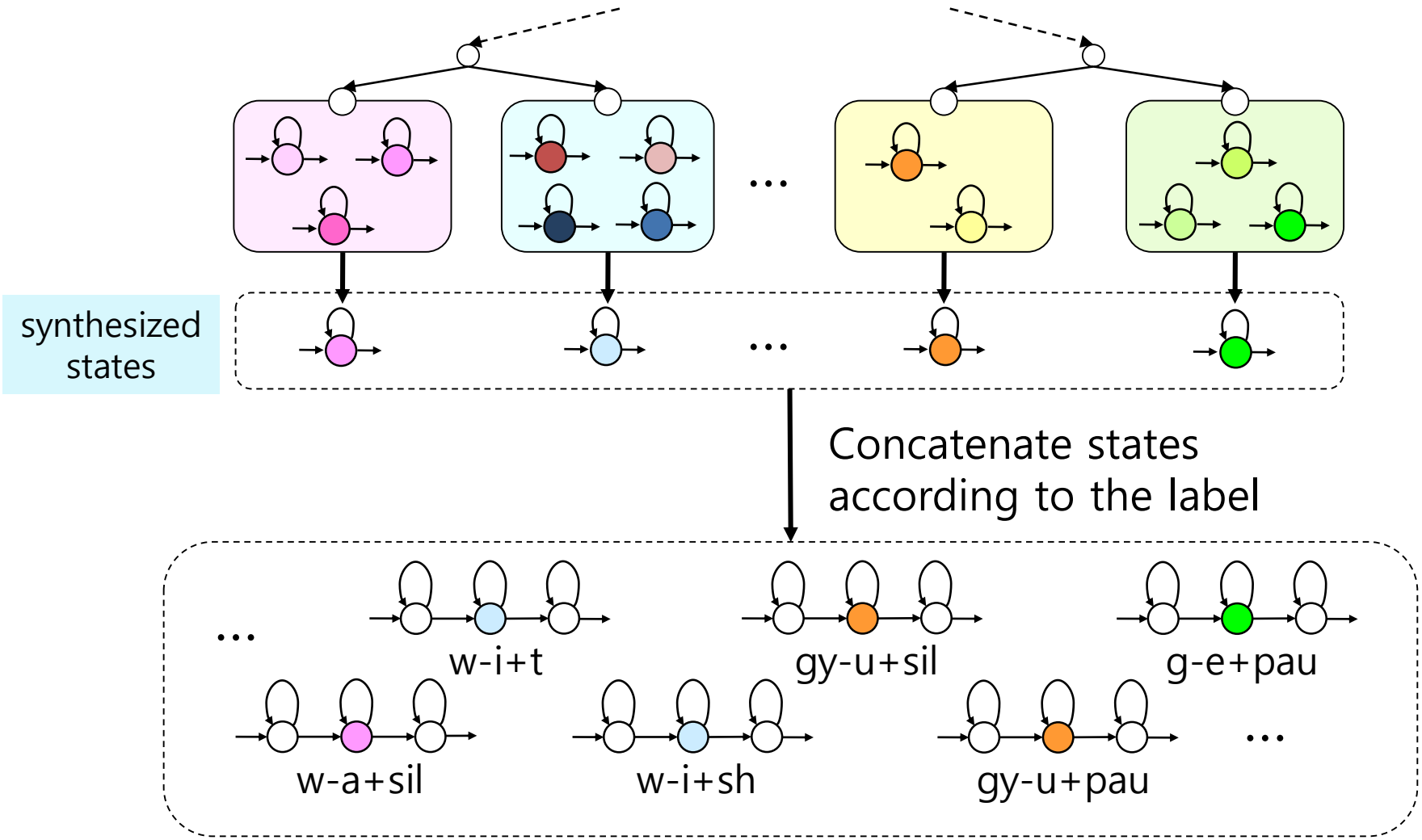
Synthesis part

- Speech parameters are generated from HMMs
  - Spectrum parameters
  - Excitation parameters (F0)
- Vocoding parameters to synthesized speech

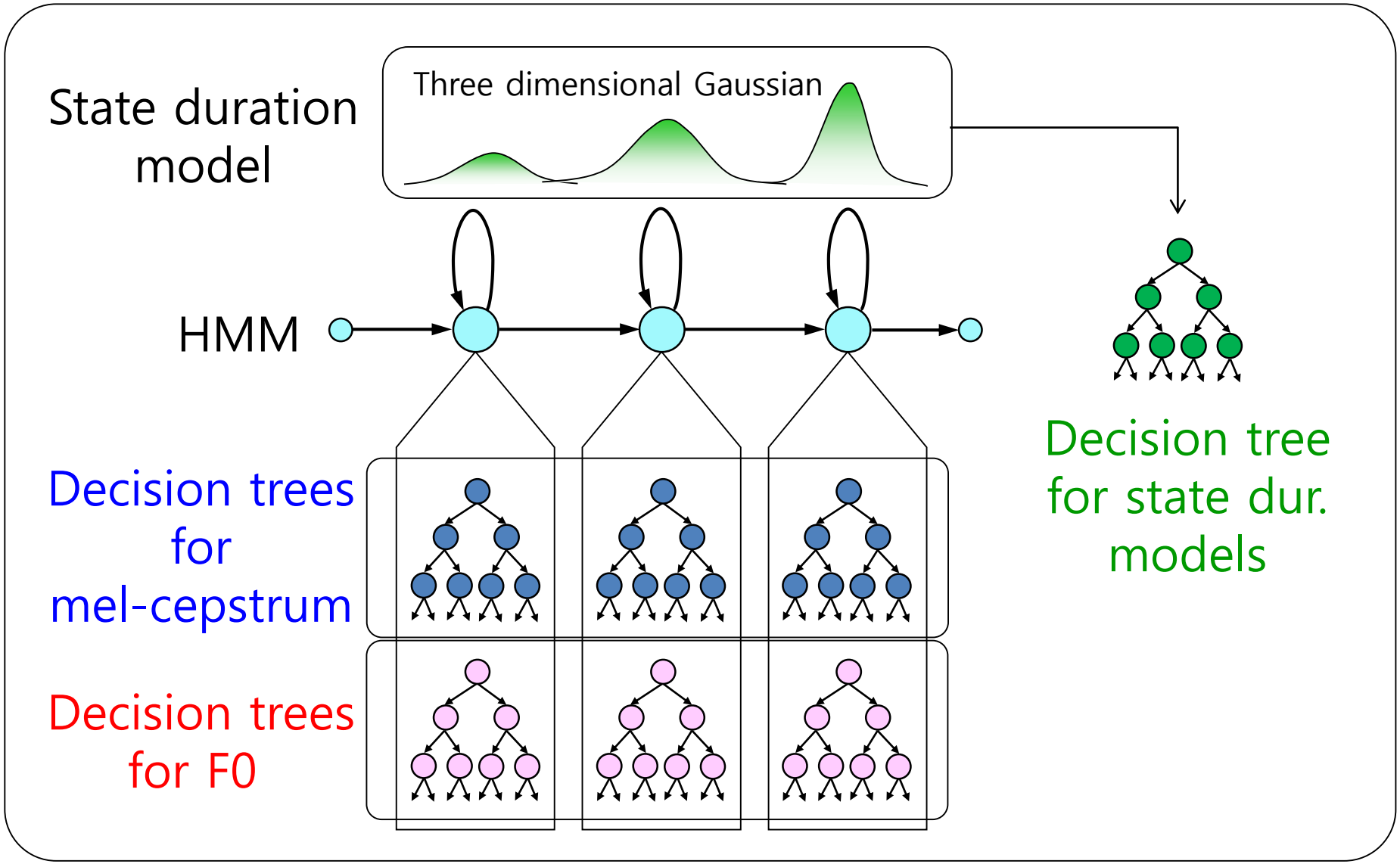
⇒ Obtain high-quality synthesized speech



Synthesize from leaf nodes



Stream-dependent tree-based clustering



HMM-based Speech Synthesis System (HTS) - Home

Total: 391481
Today: 201

Welcome! †

The HMM-based Speech Synthesis System (HTS) has been developed by the HTS working group and others (see Who we are and Acknowledgments) HTK and released as a form of patch code to HTK. The patch code is released under a free software license. However, it should be noted that once y publications about the techniques and algorithms used in HTS can be found here.

HTS version 2.3 includes VBLR speaker adaptation, DAEM-based parameter generation algorithm, and other minor new features. Many bugs in HTS the Festival Speech Synthesis System (English, Spanish, etc.), DFKI MARI Text-to-Speech System (German, English, etc.), Flite+hts_engine (English) HTS slides are also released as a tutorial of HMM-based speech synthesis.

This distribution includes demo scripts for training speaker-dependent and speaker-adaptive systems using CMU ARCTIC database (English). For trair Japanese, and Japanese song) are also released.

In addition, HTS version 2.3.1 demo scripts support frame-by-frame modeling option using DNN (deep neural network) based on HMM state alignme

News! †

- **December 25, 2016**

  HTS version 2.3.1 was released.
  Its new features are

  - Demo scripts:
    - Add frame-by-frame modeling option using DNN (deep neural network) based on HMM state alignment.

- **December 25, 2015**

  HTS version 2.3 was released.
  Its new features are

  - HERest:
    - Add VBLR adaptation.
  - HMGenS:
    - Add DAEM-based parameter generation.
    - Support DP search to determine state duration when the model alignments are given.
  - HInit, HRest, HRest:
    - Support parallel mode.

1. festival (2.4)

A general framework for building speech synthesis systems

- 여기서는 English text를 label 포맷으로 변경해주는 역할을 합니다.
- British, English, Spanish 지원

http://festvox.org/packed/festival/2.4/festival-2.4-release.tar.gz

- Dependencies

a. speech_tools

http://festvox.org/packed/festival/2.4/speech_tools-2.4-release.tar.gz

- Dependencies

1. libncurses5-dev (sudo apt-get install libncurses5-dev)

- gcc-4.2~gcc4.8로 빌드해야함

- install

ubuntu:~/festival$ ./configuration

ubuntu:~/festival$ ./make

2. SPTK (3.9)
Speech Signal Processing Toolkit

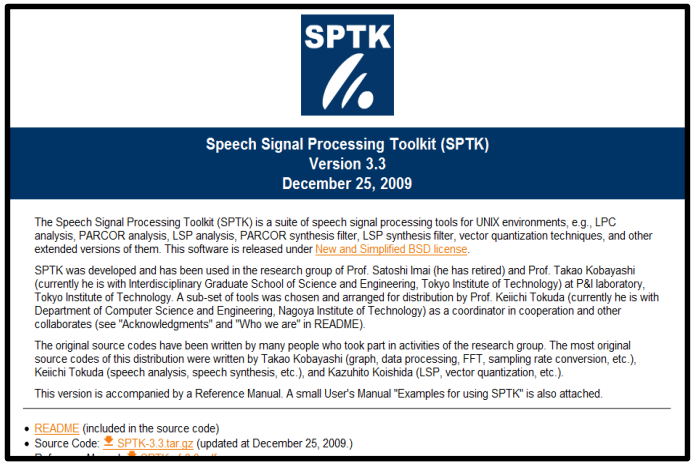
- wave 파일로부터 다양한 feature를 추출하는데 사용합니다.
- 그 외에도 신호처리를 위한 다양한 프로그램이 제공됩니다.

https://sourceforge.net/projects/sp-tk/?source=typ_redirect

- Dependencies
  1. csh

- install

```
ubuntu:~/festival$ ./configure
ubuntu:~/festival$ ./make
```



SPTK

Speech Signal Processing Toolkit (SPTK)
Version 3.3
December 25, 2009

The Speech Signal Processing Toolkit (SPTK) is a suite of speech signal processing tools for UNIX environments, e.g., LPC analysis, PARCOR analysis, LSP analysis, PARCOR synthesis filter, LSP synthesis filter, vector quantization techniques, and other extended versions of them. This software is released under New and Simplified BSD license.

SPTK was developed and has been used in the research group of Prof. Satoshi Imai (he has retired) and Prof. Takao Kobayashi (currently he is with Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology) at P&I laboratory, Tokyo Institute of Technology. A sub-set of tools was chosen and arranged for distribution by Prof. Keiichi Tokuda (currently he is with Department of Computer Science and Engineering, Nagoya Institute of Technology) as a coordinator in cooperation and other collaborates (see "Acknowledgments" and "Who we are" in README).

The original source codes have been written by many people who took part in activities of the research group. The most original source codes of this distribution were written by Takao Kobayashi (graph, data processing, FFT, sampling rate conversion, etc.), Keiichi Tokuda (speech analysis, speech synthesis, etc.), and Kazuhito Koishida (LSP, vector quantization, etc.).

This version is accompanied by a Reference Manual. A small User's Manual "Examples for using SPTK" is also attached.

- README (included in the source code)
- Source Code: SPTK-3.3.tar.gz (updated at December 25, 2009.)

3. HTS

Toolkit for HMM-based speech synthesis

- acoustic model을 학습하는데 사용됩니다.
- Research platform for HMM-based speech synthesis
- Released as a patch code for HTK
- Speaker dependent (SD) / adaptation (SA) demo scripts

http://hts.sp.nitech.ac.jp/archives/2.3/HTS-2.3_for_HTK-3.4.1.tar.bz2

- Dependencies

1. HTK (http://htk.eng.cam.ac.uk/ftp/software/HTK-3.4.1.tar.gz)
2. HDecode (http://htk.eng.cam.ac.uk/ftp/software/hdecode/HDecode-3.4.1.tar.gz)
3. libx11-dev (sudo apt-get install libx11-dev)

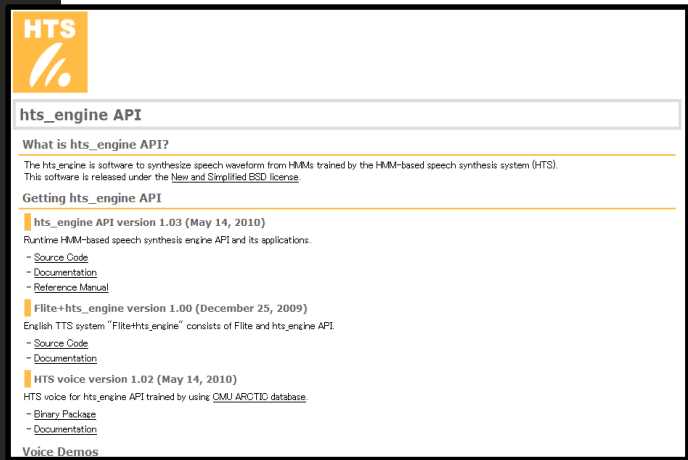
4. hts_engine

Runtime HMM-based speech synthesis engine

- Software to synthesize speech waveform from HMMs
  (HMMs trained by the HTS)
- 현재 여러 device에 embedded되어 사용되고 있습니다.

http://downloads.sourceforge.net/hts-engine/hts_engine_API-1.10.tar.gz

```
Installation Instructions
*************************

1. After unpacking the tar.gz file, cd to the hts_engine API directory.

2. Run configure script with appropriate options.

   % ./configure

   For detail, please see.

   % ./configure --help

3. Run make.

   % make

4. Install library and binary.

   % make install
```



1. Database
   - 음성, text, label
   - 새로운 언어라면 label 포맷을 정의해야 합니다.
   - question file(decision tree에 사용)
   - 좋은 성우를 선정하는 것과 음성과 text로부터 오류를 제거하는 것이 중요합니다.

2. Text로부터 Label을 만들어줄 프로그램 (G2P 포함)
   - 입력 Text를 label포맷으로 변경해줄 때 필요합니다.
   - DB에 label이 포함되어있지 않다면 DB label을 만들 때 사용될 수도 있습니다.
   - POS는 critical하지 않을 수 있지만 Phoneme은 정확하게 변환할 수 있어야 합니다.

Q & A