PARLab Parallel Boot Camp

PARLab Application: Speech recognition for meetings

Nelson Morgan
International Computer Science Institute (ICSI)
and
Electrical Engineering and Computer Sciences
University of California, Berkeley

PARLab Parallel Boot Camp

PARLab Application:
Speech recognition for meetings

Representing work from a number of people, but primarily:
Adam Janin, Chris Oei, Suman Ravuri, Sherry Zhao (ICSI)

And

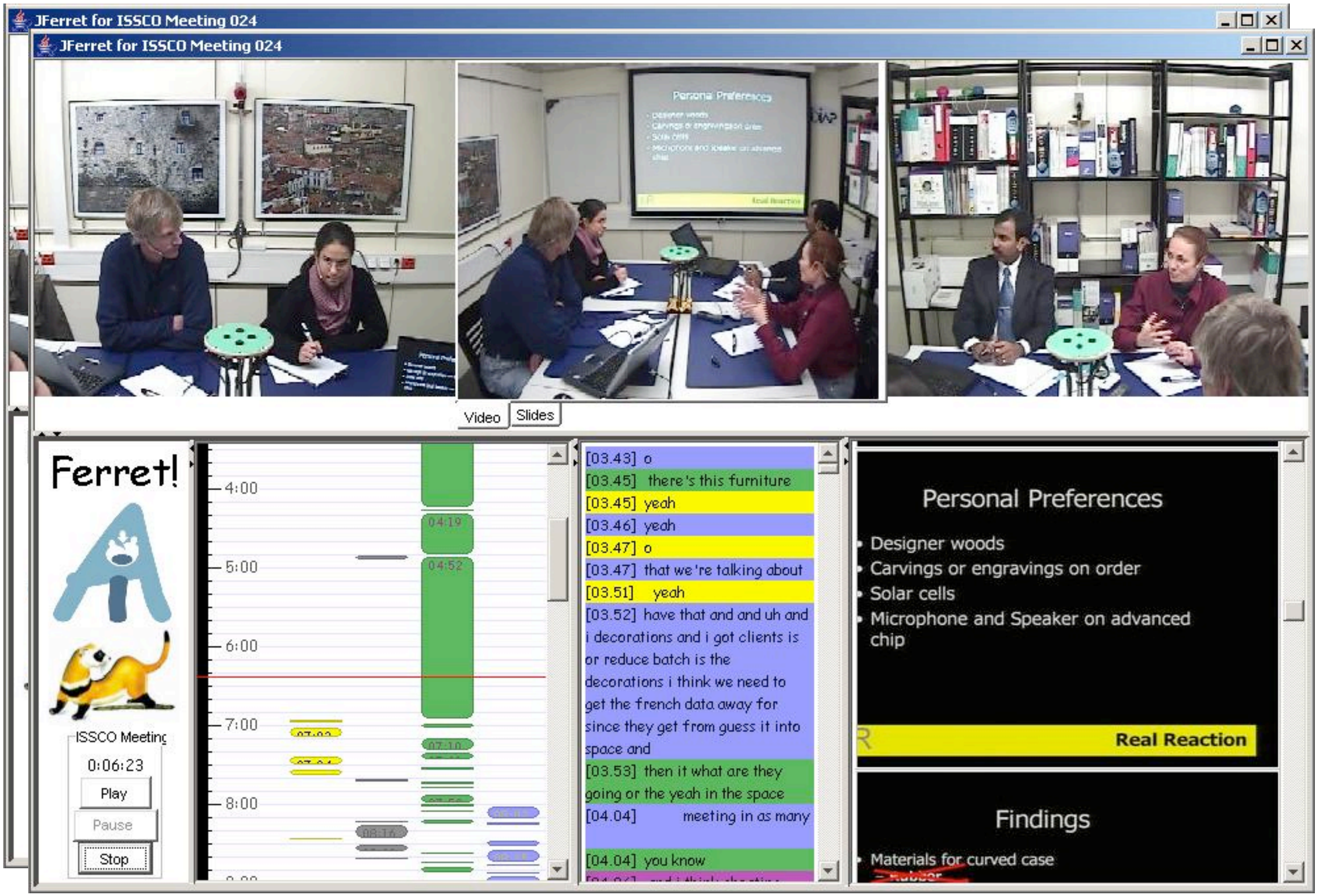
Jike Chong, Youngmin Yi, and Ekaterina Gonina (UCB/EECS)

The “meeting” application - goals

For “real” meetings:

- Replacing inconsistent note-taking
- Access to transcriptions
- Indexed information for search
- Query-specific summaries

The "meeting diarizer" application



The "meeting" application - challenges

- Most meeting rooms not heavily instrumented
- Resulting signals have significant noise and reverberation -> poor speech recognition accuracy
- Real time performance necessary for many scenarios
- Some applications require better than real time
- Other components aside from speech recognition also required
- Not just a need for speed: also a need for better performance (accuracy)

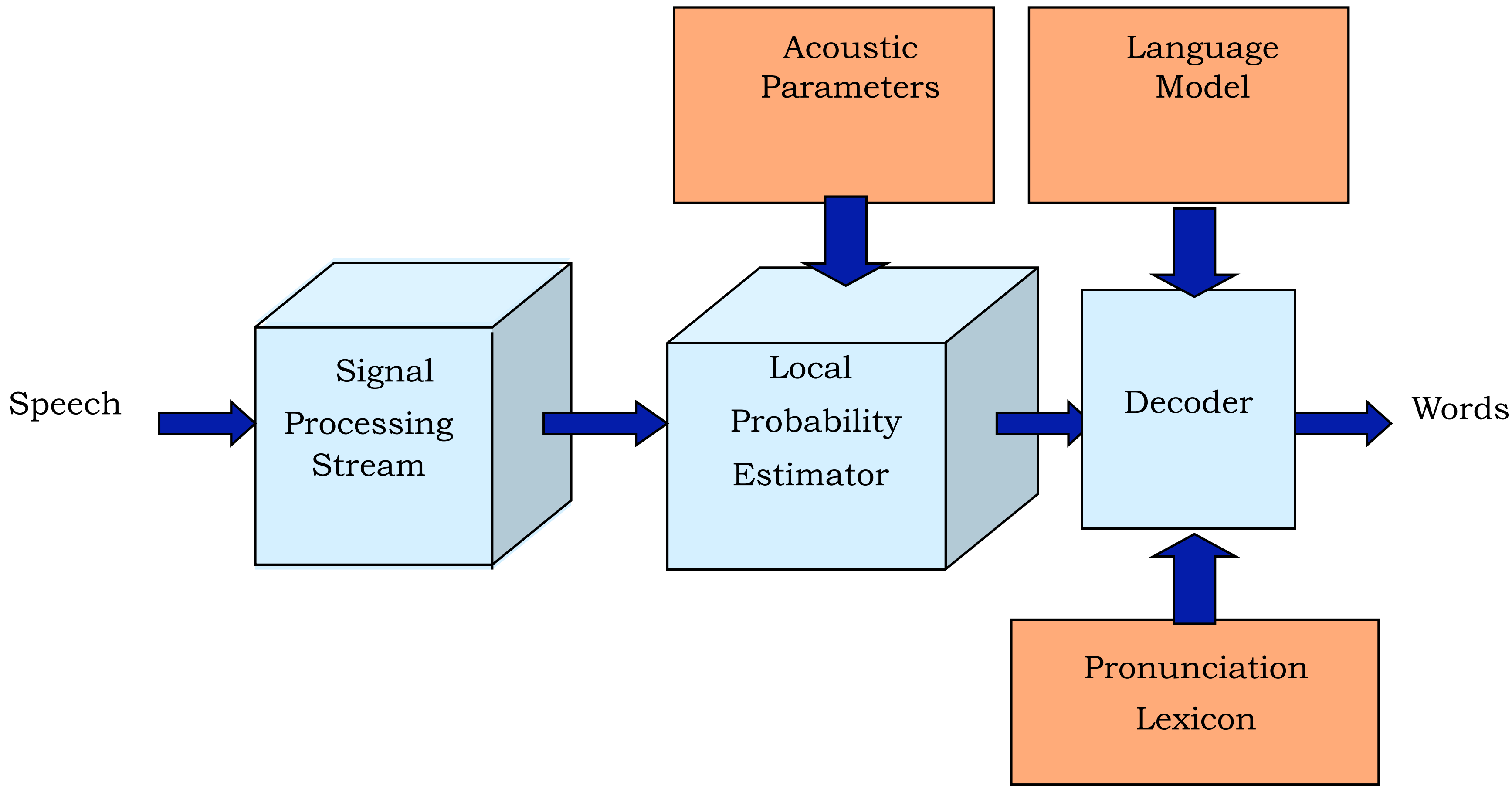
The "meeting" application – primary questions

- Can extreme parallelism be used to improve accuracy?
- Can we make use of PARLab primitives to efficiently represent all of the components of this application?
- Can new approaches to this application be coded by mere mortals?

Components of the application

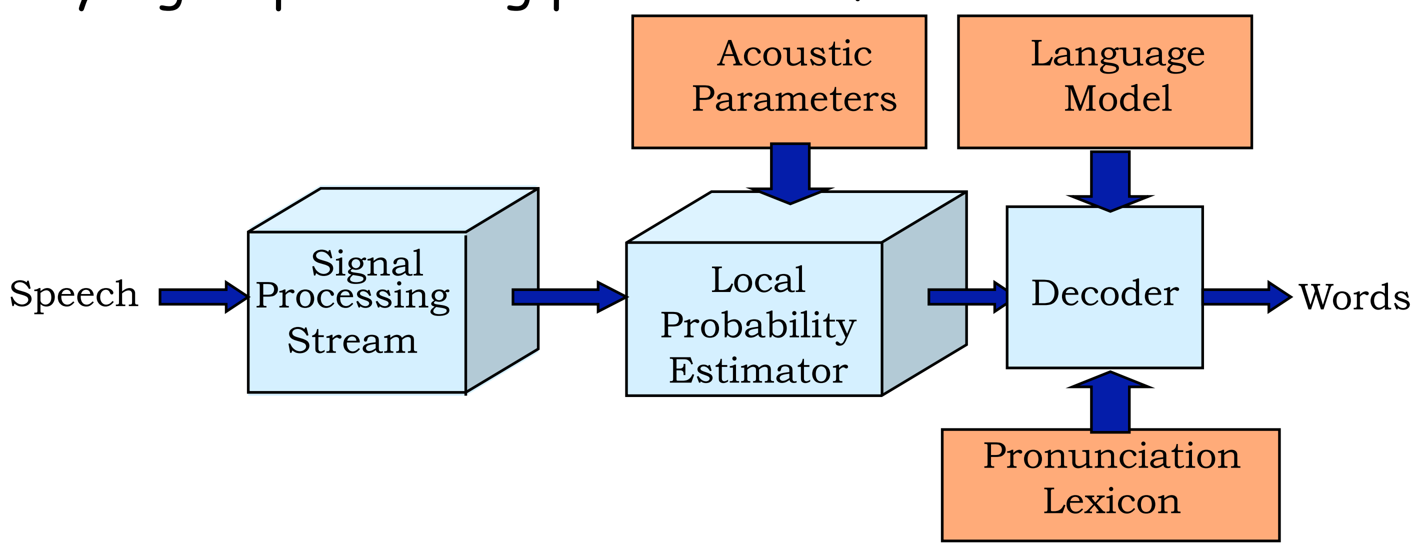
- Automatic speech recognition
- Speaker diarization
- Speaker recognition
- Question answering/summarization
- Topic clustering
- …

Basic uni-stream speech recognition



High Level Parallel Pattern

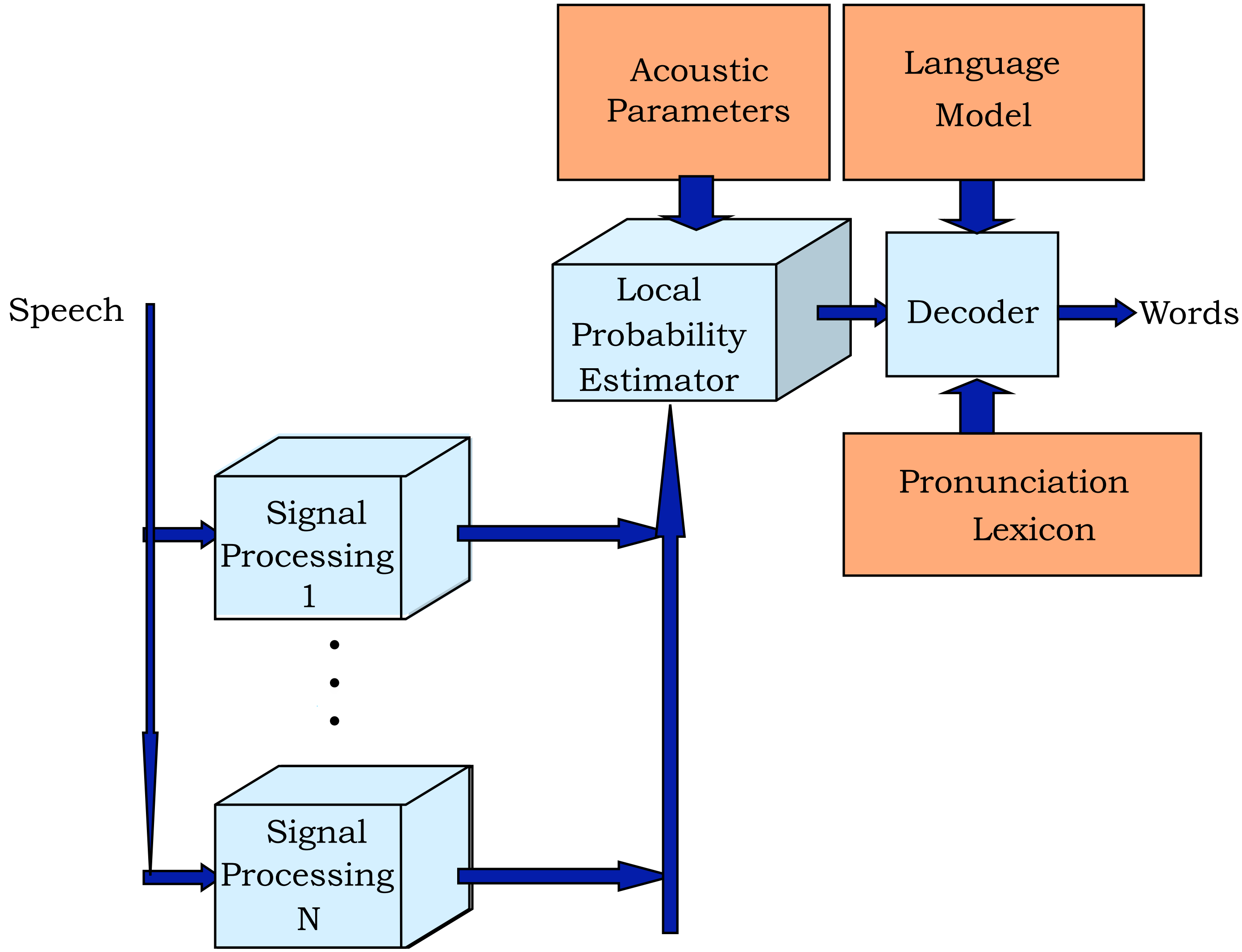
- System level parallelism is determined by “decoding” strategy. Current state-of-the-art decoders are time synchronous, but this is not the only option.
- With time synchronous decoding, the system-level pattern is pipe-and-filter with task parallelism.
- Most systems integrate the local probability estimator and the decoder.
- Currently signal processing part is small; but should it be?



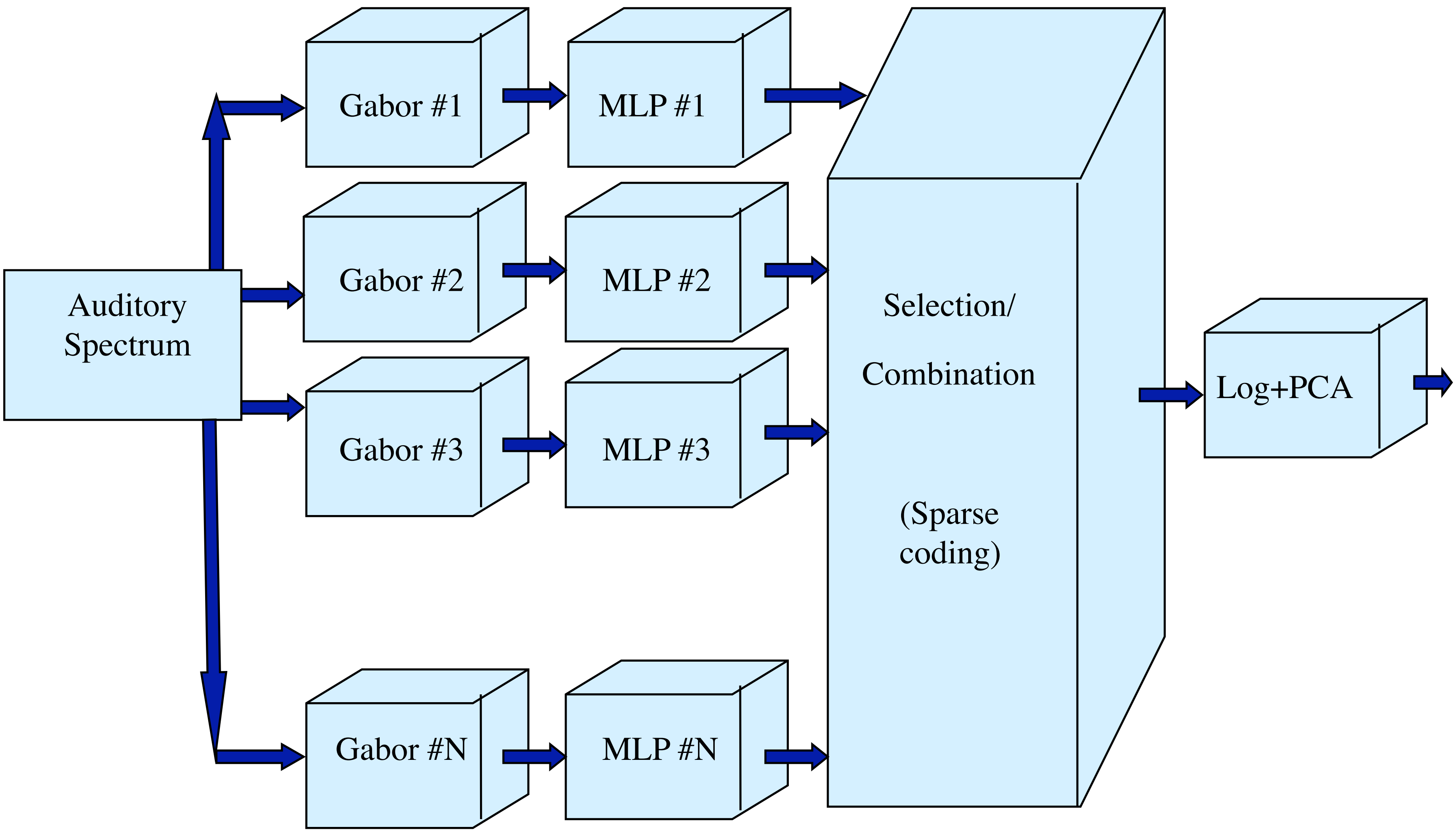
Speech recognition: one stream to multi to many

- Speech recognition works well under good conditions given plentiful resources (e.g., training) [<10% word error rate (WER)]
- Poor performance for common conditions [>30% WER] (noise, reverb, + casual/conversational speech)
- Multiple and diverse signal processing methods help, e.g., several "streams" of features
- An open question: can a large (>100) number of streams provide much greater robustness?
- Preliminary results suggest yes (15% WER -> 8%)

Multi/many stream speech recognition



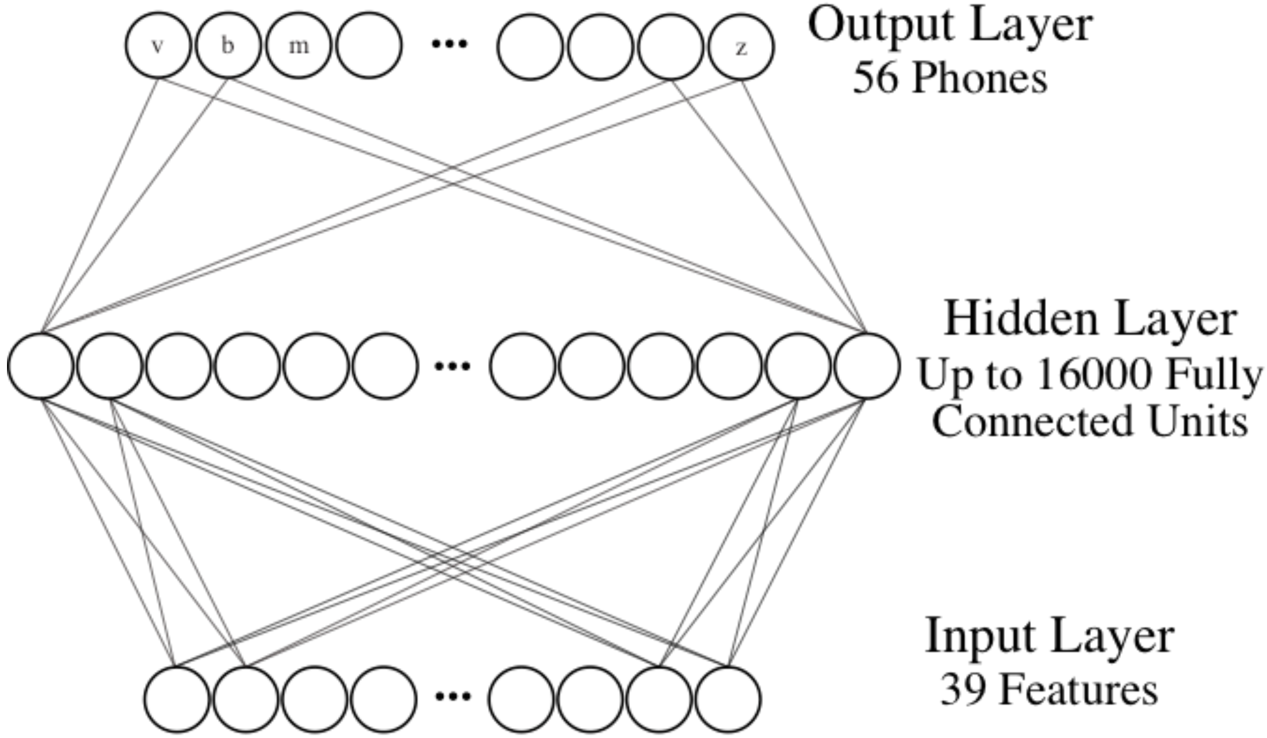
Multi/many stream feature extraction



Multi/many stream parallel pattern

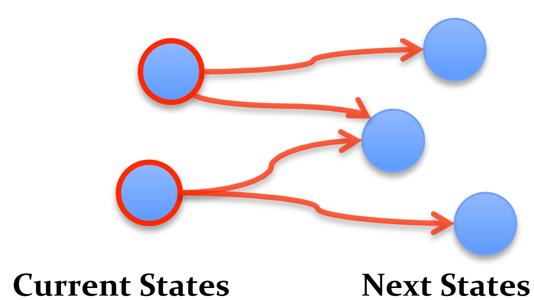
- Multi/many stream computation
  - Map Reduce pattern
  - Task parallelism
- Gabor filters
  - Dense linear algebra, SIMD
- MLPs
  - Dense linear algebra, SIMD
- If the filters are similar enough, one could instead use SIMD across all the filters.

Multilayer Perceptron
(a.k.a Neural Network)



Decoder

- The "decoder" outputs the most likely word sequence given the data.
- Implemented as a Weighted Finite State Transducer
- Complex graph traversal algorithm
- Innermost loop is state (node) update
  - Parallel over states OR arcs
  - SIMD



Parallelizing the parts

- Explicitly parallel parts: multiple feature streams, including MLPs -> task parallel
- Embarassingly parallel parts: MLPs, Gabor filter, and Gaussian computations -> dense linear algebra, SIMD.
- Tricky stuff: speech "decoding" -> graph traversal

  (currently done with weighted finite state transducers)

Summary

- Application person's point of view: improving the application performance
- Parallelization is a means to that end
- For some applications, faster than real-time is useful
- To run meeting app on future handheld devices, parallelism will be required
- Each of the meeting diarizer components needs to be parallelized
- For the speech recognition part, we have done this in a painstaking way
- Given the identification of parallel motifs, we hope to be able to build the full application with ParLab tools