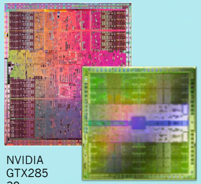
# Exploring Recognition Network Representations for Efficient Speech Inference on the GPU

Jike Chong, Ekaterina Gonina, Kisun You, Kurt Keutzer, Department of Electrical Engineering and Computer Science, University of California, Berkeley
jike@eecs.berkeley.edu, egonina@eecs.berkeley.edu, ksyou@dsp.snu.ac.kr, keutzer@eecs.berkeley.edu

Parallel Computing Lab

## Maturing Highly Parallel Platforms



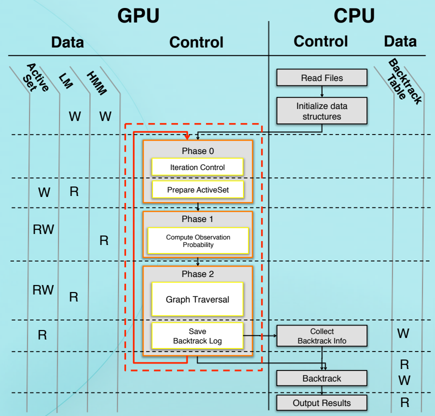
NVIDIA GTX285
30 cores,
8-wide SIMD

NVIDIA GTX480
(Fermi)
15 cores, 16-wide SIMD

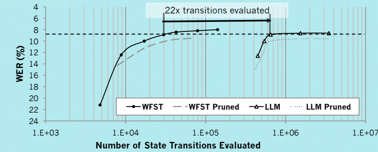
- Architecture trend:
  - Increasing vector unit width
  - Increasing numbers of cores per die
- Maturing HW architecture:
  - Including caches as well as local stores that benefit irregular accesses

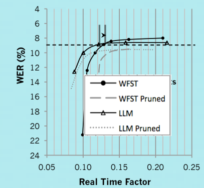
Ongoing work investigates performance of alternative approaches to speech recognition on these highly parallel platforms

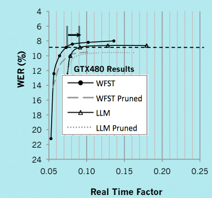
## Implementation Architecture



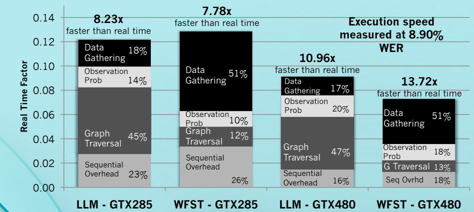

## Evaluation of the Recognition Network Representations




- To achieve the same accuracy:
  - LLM traverses 22x more state transitions than WFST
  - On GTX285, LLM is faster
  - On GTX480, WFST is faster
- Looking at detailed timing:
  - LLM takes 3-5x more time in Graph Traversal, but evaluates 22x more transitions
  - Regularity of LLM reduces cost of Data Gathering (18%)
  - 51% of the execution time in WFST is spent in gathering data from its irregular data structure

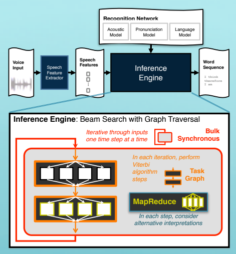



- Per state transition LLM is 53-65x faster in data gather and 4.7-6.4x faster in graph traversal
- GTX480 improves sequential overhead by 85% and 159% for LLM and WFST respectively
- WFST becomes faster on GTX480 due to the reduction in overhead and caching

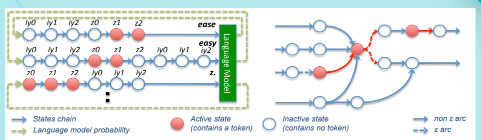
## Speech Recognition Inference Engine Characteristics

- Parallel graph traversal through Recognition network
  - Guided by a sequence of input audio vectors
  - Computing on continuously changing data working set
- Implementation challenges
  - Define a scalable software architecture to expose fine-grained application concurrency
  - Efficiently synchronize between an increasing number of concurrent tasks
  - Effectively utilize the SIMD-level parallelism

Want to learn more about this topic?
**Session 2046 - Efficient Automatic Speech Recognition on the GPU**
Thursday, September, 23rd, 15:00 - 15:50




## Two Recognition Network Representations




A sample Linear Lexical Model Network — A section of a Weighed Finite State Transducer Network

- LLM Network
  - Chain of triphone states for each pronunciation
  - Each chain constructed using a separate copy of triphone states – many duplications
  - Evaluate possibility of transition from one word to all other words at the end of each triphone chain
- WFST Network
  - FSM of composed pronunciation and language models
  - Across-word transitions explicitly represented
  - Encapsulates large amount of information with little redundancy
  - Fewer tokens required to be maintained for target accuracy

| | LLM Pruned | LLM | WFST Pruned | WFST |
|---|---|---|---|---|
| # States | 123,246 | 123,246 | 1,091,295 | 3,925,931 |
| # Arcs | 537,608 | 1,596,884 | 2,955,145 | 11,394,956 |

- Wall Street Journal 1 Corpus
  - Based on a 5,000 word vocabulary, 1,350,392 bigrams (291,116 pruned)
  - 3000 16-mixture acoustic models, 39 dim features based on 13 dim MFCC
    - WFST network is an HCLG model compiled and optimized offline

## Conclusions

- Simpler LLM network representation performs competitively with highly optimized WFST representation
- WFST representation is a more concise representation requiring traversal of 1/22th number of state transition to achieve the same accuracy
- Per state transition LLM gathers data 53-65x faster and evaluates transition 4.7-6.4x faster than WFST
- Uncoalesced memory accesses is still a major bottleneck in implementations using the WFST representation

Emergence of highly parallel platforms brings forth an opportunity to reevaluate computational efficiency of speech recognition approaches.

Thanks to Nelson Morgan, Andreas Stolcke, and Adam Janin at ICSI for insightful discussions and continued support in the infrastructure used in this research. This research is supported in part by an Intel Ph.D. Fellowship. This research is also supported in part by Microsoft (Award #024263 ) and Intel (Award #024894) funding and by matching funding by U.C. Discovery (Award #DIG07-10227).