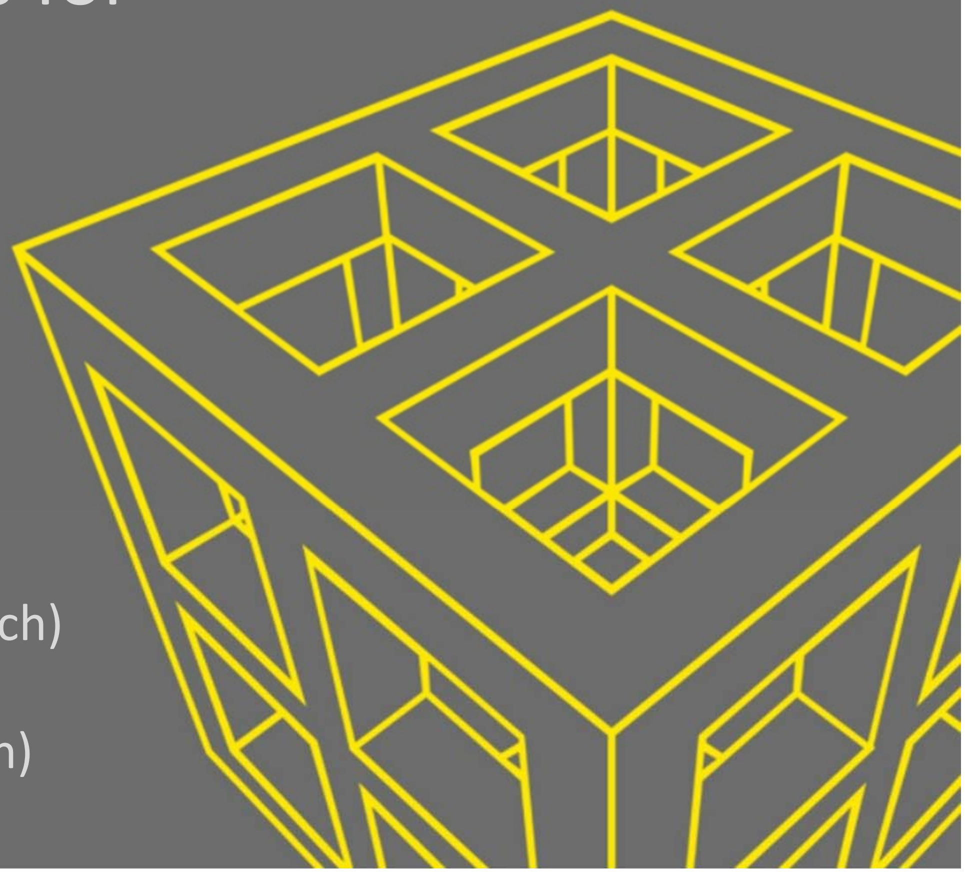
Parallel HMMs

Parallel Implementation of Hidden Markov Models for Wireless Applications



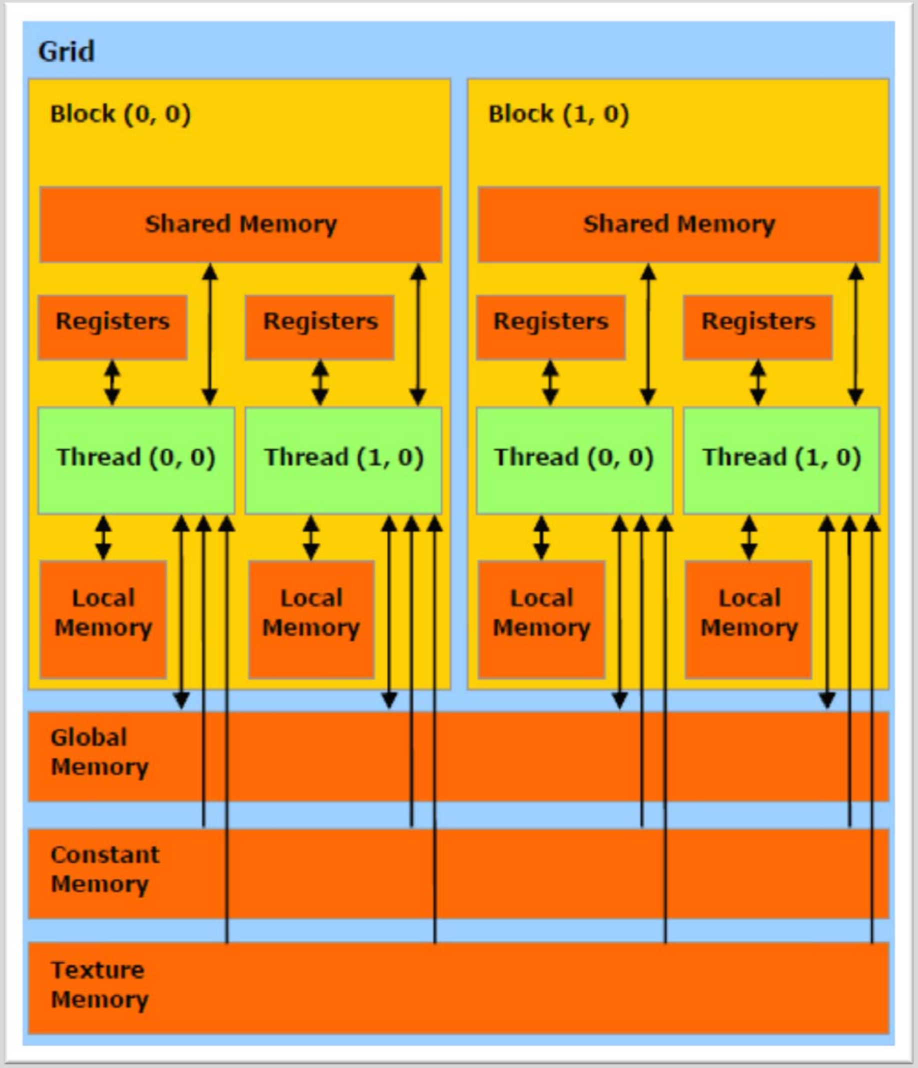
Authors
Shawn Hymel (Wireless@VT, Virginia Tech)
Ihsan Akbar (Harris Corporation)
Jeffrey Reed (Wireless@VT, Virginia Tech)

Agenda

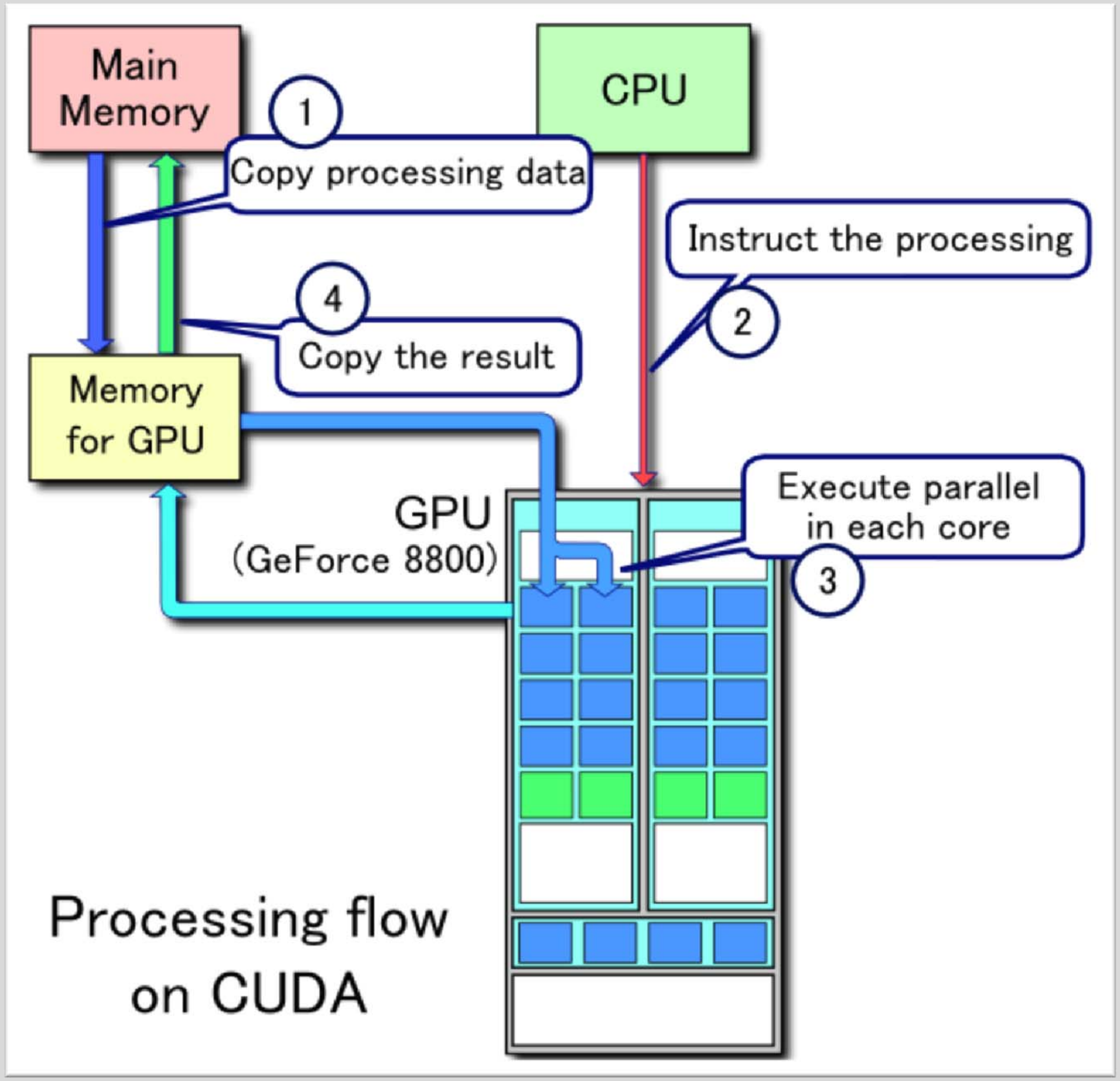
- Overview of GPGPU
- Overview of HMMs
- Parallelization
- Results
- Applications
- Why Is This Useful?

General-Purpose Processing on GPUs

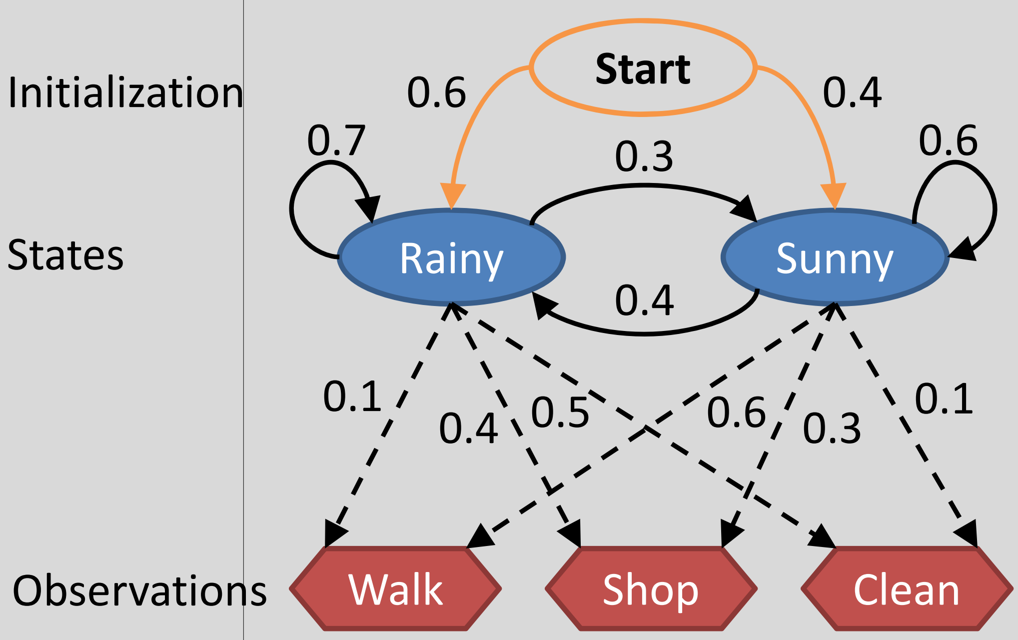
- CUDA-specific
- Important Terms:
  - Threads
  - Blocks
  - Grid



CUDA Code Flow



Hidden Markov Model



$$\lambda = (A, B, \pi)$$

$$A = \begin{bmatrix} 0.7 & 0.3 \\ 0.4 & 0.6 \end{bmatrix}$$

$$B = \begin{bmatrix} 0.1 & 0.6 \\ 0.4 & 0.3 \\ 0.5 & 0.1 \end{bmatrix}$$

$$\pi = \begin{bmatrix} 0.6 & 0.4 \end{bmatrix}$$

HMM Canonical Problems

- Evaluation: P(O|λ)
  - Forward Algorithm
  - Backward Algorithm
- Find the most likely state sequence
  - Viterbi Algorithm
- Training (maximize P(O|λ))
  - Baum-Welch Algorithm

Forward Algorithm

Given a model and an observation sequence, calculate P(O|λ)

- T = number of observations
- N = number of states
- M = number of possible symbols

Initiation:

$$\alpha_1(i) = \pi_i b_i(O_1),\ i = 1,2,\ldots N$$

Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1})$$

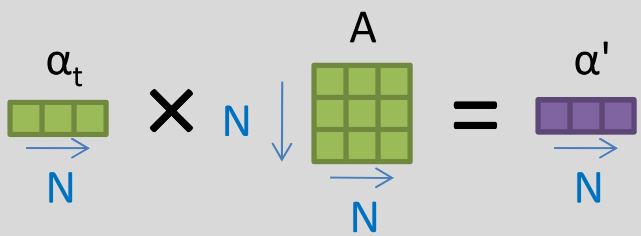
Termination

$$P(O \mid \lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

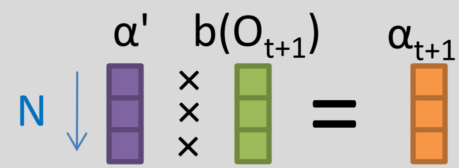
Example of Parallelization

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij} \right] b_j(O_{t+1})$$

For all j, matrix multiplication



For all j, element-by-element multiplication



We can perform this step in parallel!

$O(TN^2) \rightarrow O(T \log N)$

Computational Complexity

	Serial	**Parallel**
Forward Algorithm	$O(TN^2)$	$O(T \log N)$
Viterbi Algorithm	$O(TN^2)$	$O(T \log N)$
Baum-Welch Algorithm	$O(TN^2)$ or $O(TMN)$	$O(T \log N)$

Test Procedures

- Time execution of each algorithm (C vs. CUDA)
  - Vary states
  - Vary symbols
  - Vary sequence length
- Calculate total energy consumption (C vs. CUDA)
  - PowerTOP software

Test Hardware

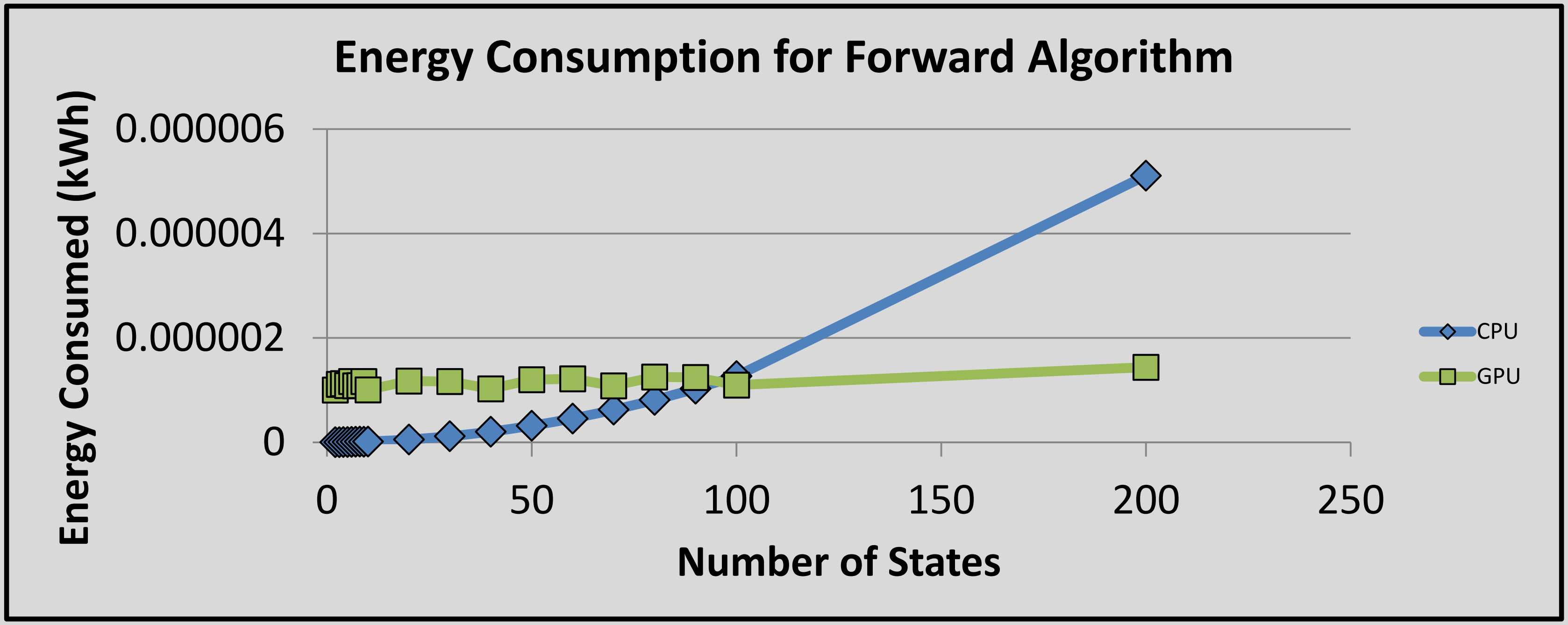
Component	Specification
CPU	Intel Core 2 Duo U7300 @ 1.30GHz
GPU	NVIDIA GeForce GT 335M
GPU Core Speed	450 MHz
GPU Shader Speed	1080 MHz
GPU Memory Speed	1066 MHz
CUDA Cores	72

Speed Results

Number of States	CPU Runtime (s)	GPU Runtime (s)	Speed Increase
	Forward Algorithm		
4	0.001	0.1531	**0.007x**
40	0.04	0.1393	**0.287x**
400	4.2816	0.2379	**17.99x**
4000	534.2028	2.9495	**181.12 x**
	Viterbi Algorithm		
4	0.0033	0.1605	**0.021x**
40	0.0436	0.1801	**0.242x**
400	4.2684	1.6595	**2.57x**
4000	534.5543	116.2531	**4.60 x**
	Baum-Welch Algorithm		
4	0.0021	0.4142	**0.005x**
40	0.1946	0.4299	**0.453x**
400	17.6719	0.7502	**23.56x**
4000	1834.672	28.1271	**65.23 x**

Energy Consumption

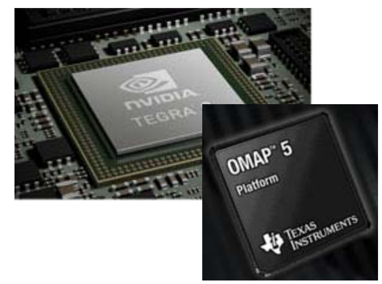
Algorithm	Power (W)		States to Break Even
	C	CUDA	
Forward	18.5	26.5	~100
Viterbi	18.5	29.1	~120
BWA	18.3	26.1	~70



Applications

- Pattern Recognition
  - Spectrum Sensing
  - Signal Classification
  - Specific Emitter Identification
  - Geolocation
- Modeling
  - Channel Fading
  - Call Drop Prediction

Why Is This Useful?



- Evolution of GPUs and multi-core processors
  - Smart phones, tablets, SDR
  - Co-processor
- Utilize existing hardware for HMM applications
  - Large number of states
  - 2D/3D HMMs
- Uses in other fields (speech recognition, computer vision)
- Extrapolation to other algorithms (pattern recognition)

Questions?

Contact Information

Email: hymelsr@vt.edu
Blog: http://sgmustadio.wordpress.com/
Code: http://code.google.com/p/hmm-cuda/

Other Good Resources

cuHMM: http://code.google.com/p/chmm/
MATLAB: http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html
HTK: http://htk.eng.cam.ac.uk/

Supporting Slide:
Reductions

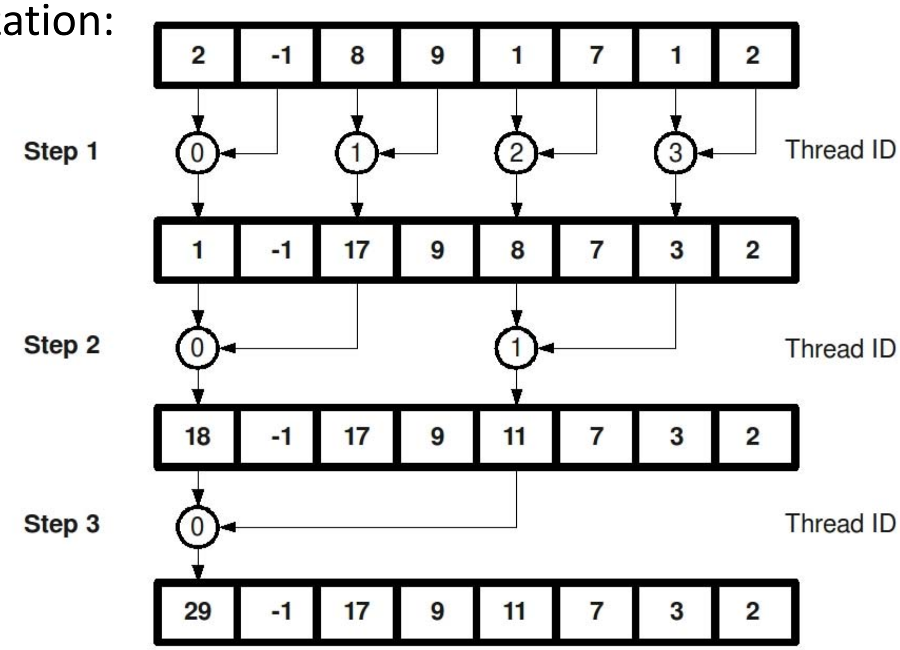
MATLAB example:

```
>> sum(A)
```

C Implementation:

```
sum = 0;
for (i = 0; i < length; i++) {
    sum = sum + A[i];
}
```

Parallelization:

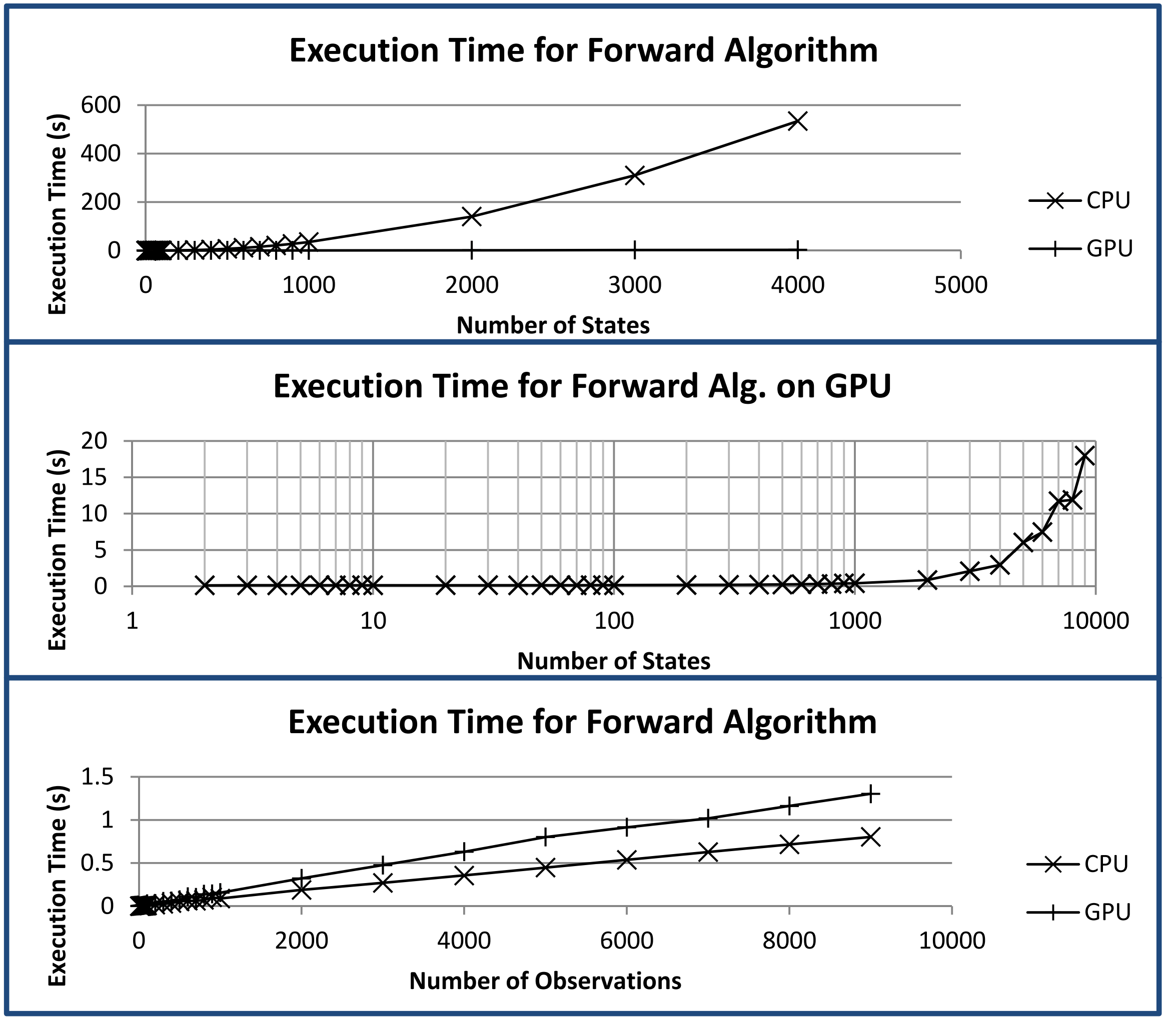


Reducing arrays to a single value (e.g. sum) go from O(N) to O(log N)

Supporting Slide:
Timing Results (Forward)

Vary States

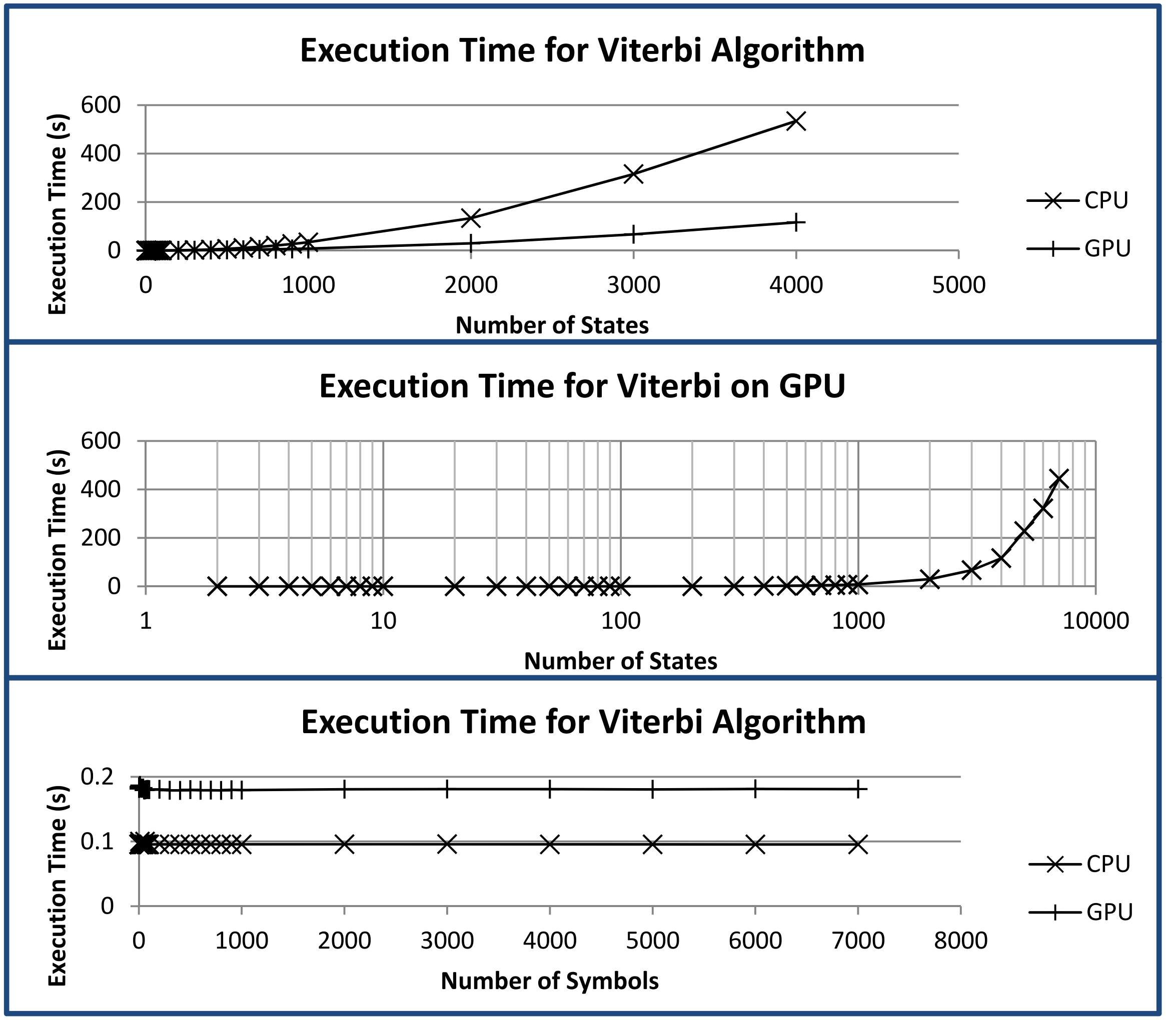
Vary Symbols



Supporting Slide:
Timing Results (Viterbi)

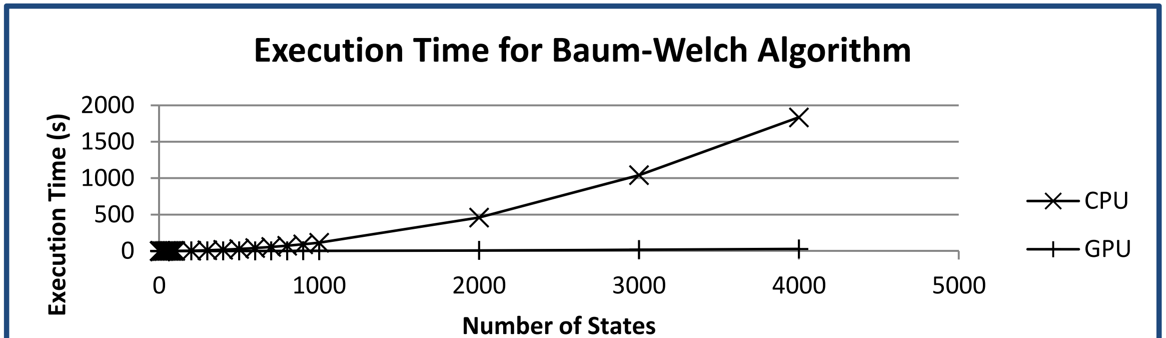
Vary States

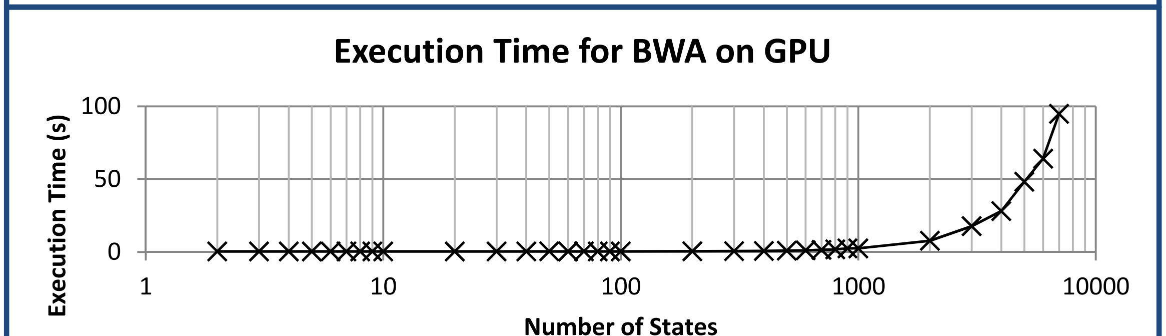
Vary Symbols



Supporting Slide:
Timing Results (BWA)

Vary States





Vary Symbols