Experiments with Bayesian Inference Accelerators

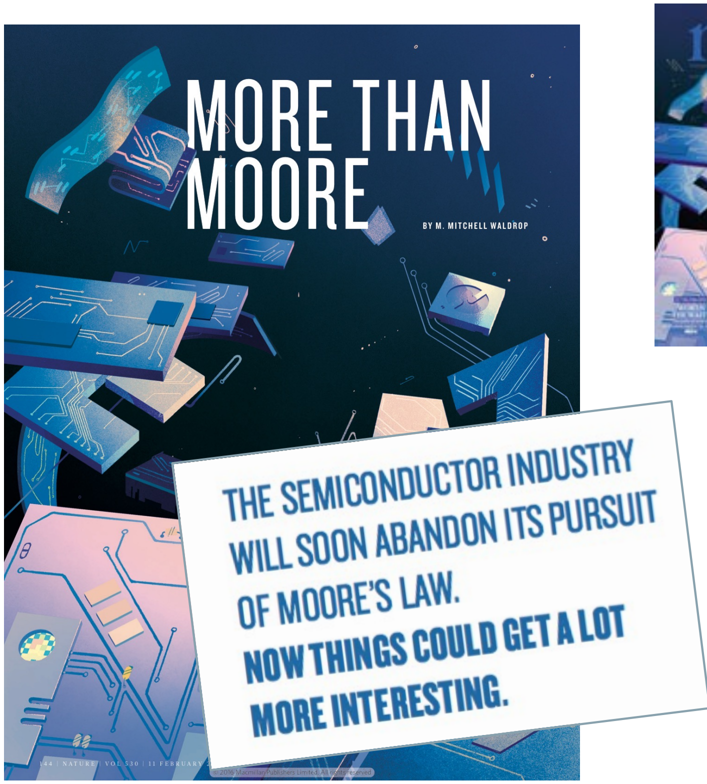
(Why AI Algorithms that are NOT Deep Neural Nets Also Want to be Silicon)

University of Wisconsin-Madison
Virtual Computer Architecture Seminar
October 15, 2020

Rob A. Rutenbar
Senior Vice Provost for Research
Professor, CS & ECE

Pitt Research

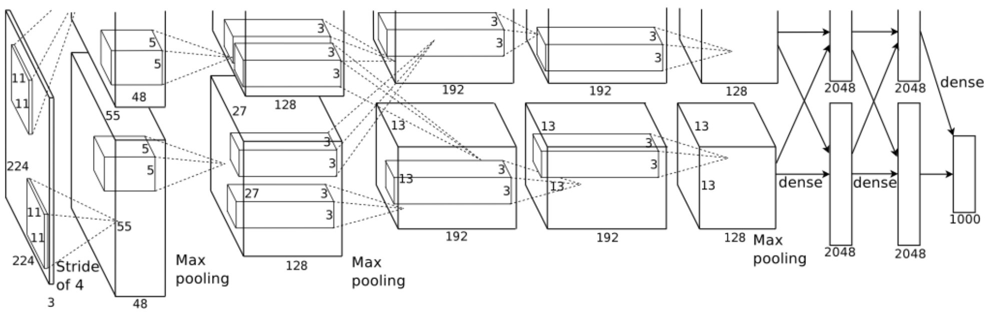
Accelerators: Why Now…?



- **Moore's Law**
  - A great 40-year run
  - Now, running out of gas
- **Apps too slow, or too power-hungry…?**
- **Let's try transistors!**

http://www.nature.com/news/the-chips-are-down-for-moore-s-law-1.19338

Focus: Deep Neural Nets (DNNs)



ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

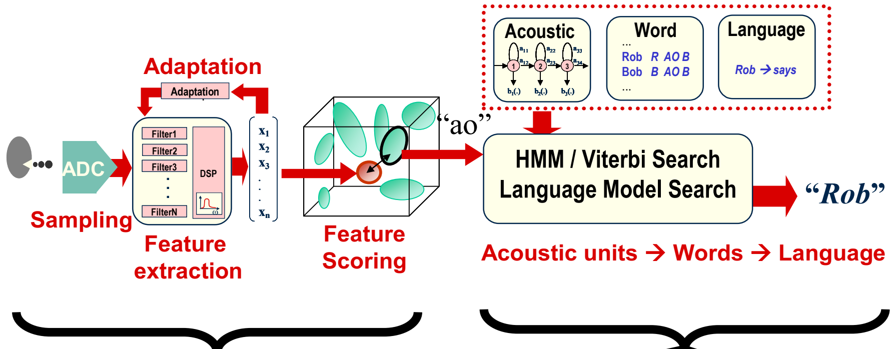
NIPS 2012

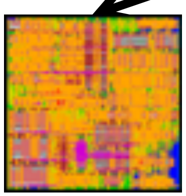
- **In hindsight, hardware is obvious here:**
  - Breakthrough performance; widely useful; too slow on CPU
- **And -- look like (giant) DSP tasks:**
  - Feed-forward (mostly), limited operators, limited precision, etc.
  - Main differences: scale, #weights, data movement

Pitt Research

Aside: Prior to Today’s Bayesian HW

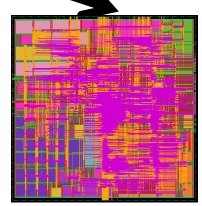
- **Speech recognition in Si:** CMU *In Silico Vox* project





28x Faster than realtime
2000-word vocab
65nm, 200MHz

247x Faster than realtime
60,00-word vocab
65nm, 500MHz, 2W



Low-Power mobile search
20,000-word vocab
65nm, 100MHz, 92 mW

Economist,
March 12, 2005

Pitt Research

Medallia announces $ acquisition of Voci Technologies

Medallia acquires artificial intelligence speech transcription company, Voci Technologies for $59M

> "Voci transcribes 100% of live and recorded calls into text that can be analyzed quickly to determine customer satisfaction, adding a powerful set of signals to the Medallia Experience Cloud. At the same time, Voci enables call analysis moments after each interaction has completed, optimizing every aspect of call center operations securely. Especially important as virtual and remote contact center operations take shape."

CMU accelerators for high-speed recognition went to market as **Voci Technologies**

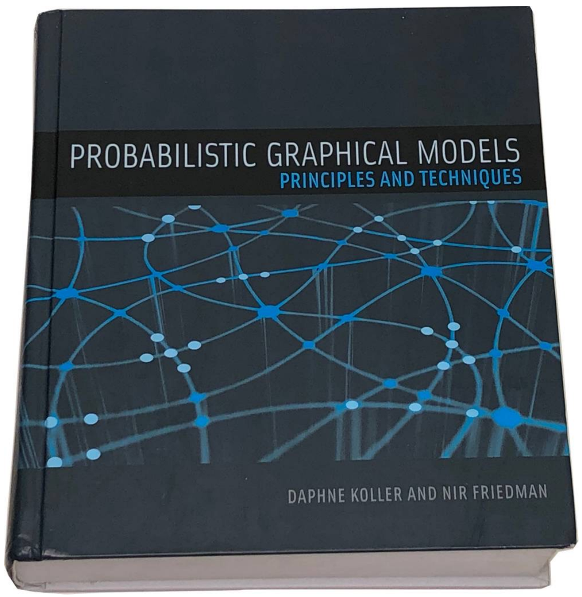
Started life as non-DNN FPGAs…

... but ended up as DNN-GPUs

**Edited from: https://www.mergersight.com/post/medallia-announces-59m-acquisition-of-voci-technologies*

So, DNNs – Is This All This Is…?

- Actually -- **No**



- Focus: **Bayesian inference**
  - **X = hypothesis; Y = evidence**

Likelihood

Prior

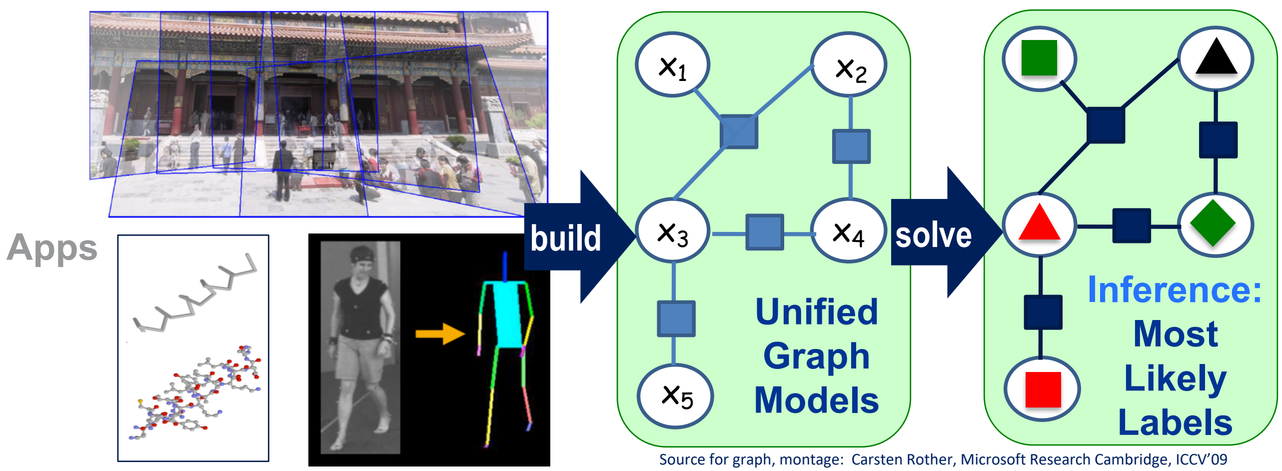
$$P(X|Y) = \frac{P(Y|X)\,P(X)}{P(Y)}$$

Posterior

Marginal Likelihood

Inference on Prob Graphical Models

- **PGMS include:**
  - **Nodes**: encode what we **observe/know**, how much we believe it
  - **Edges**: encode **relationships** (joint dependencies/affinities)
  - **Inference**: solve for **"most likely" labels** @ nodes

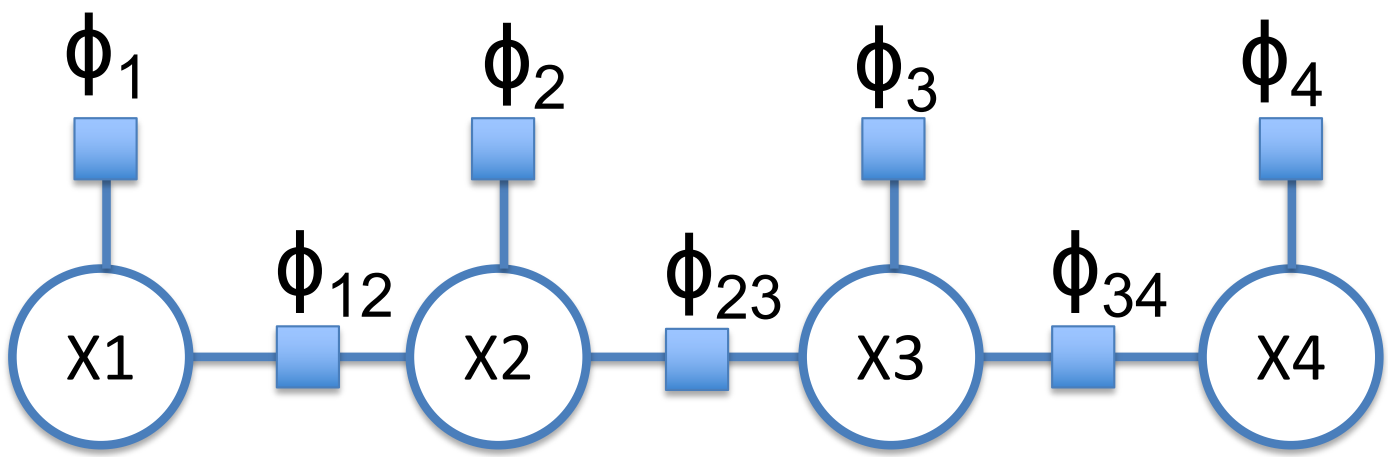


Source for graph, montage: Carsten Rother, Microsoft Research Cambridge, ICCV'09
Source for pose est: Pushmeet Kohil, Microsoft Research Cambridge, ibPRIA'11

Pitt Research

Short Tutorial: Inference on PGMs

- 4 nodes, 3 edges, 3 discrete labels
  - **Markov Random Field (MRF)**, in Factor Graph form



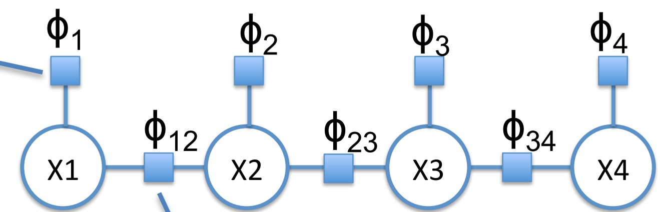
Xi take values in { ○ , ■ , ◆ } -- discrete label set

Pitt Research

PGMs: Factors ϕ

- ϕ_i and ϕ_{ij} are ***affinities***

X1	ϕ_1
○	1
■	10
◆	5

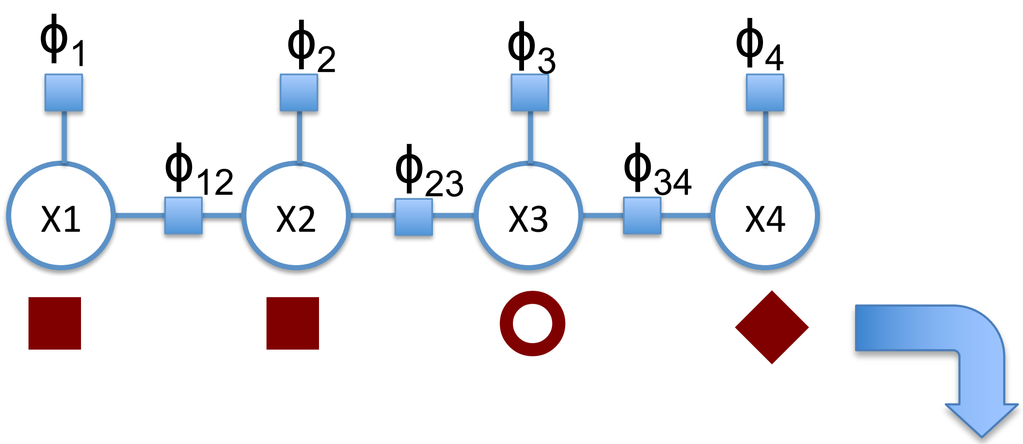


X1	X2	ϕ_{12}
○	○	8
○	■	2
○	◆	1
■	○	3
■	■	9
■	◆	1
◆	○	2
◆	■	5
◆	◆	10

Φ's describe how much the variables "want to be" different label values

PGM: Labeling Entire Graph

- What is “affinity” of whole graph for a set of labels?
- Answer: **Product of the factors ϕ**

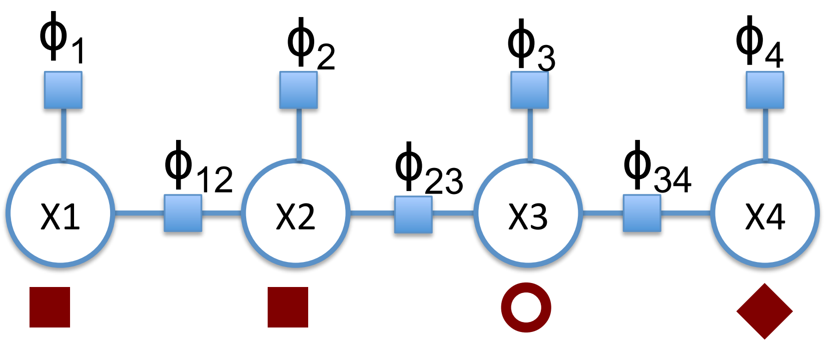


$$\prod \phi = \phi_1(\blacksquare)\ \phi_{12}(\blacksquare,\blacksquare)\ \phi_2(\blacksquare)\ \phi_{23}(\blacksquare,\bigcirc)\ \phi_3(\bigcirc)\ \phi_{34}(\bigcirc,\blacklozenge)\ \phi_4(\blacklozenge)$$

Pitt Research

From Factors to Probabilities

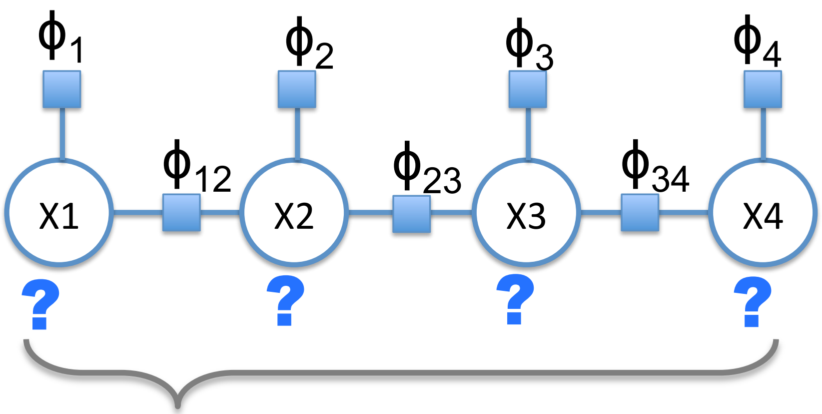
- Affinity != probability.   How to get probabilities?
- Answer: ***Normalize*** **via Z (called 'partition function')**



$$\Pr[X1,X2,X3,X4] = \frac{\phi_1(-)\ \phi_{12}(-,-)\ \phi_2(-)\ \phi_{23}(-,-)\ \phi_3(-)\ \phi_{34}(-,-)\ \phi_4(-)}{\underbrace{\sum_{X1,X2,X3,X4} \phi_1(-)\ \phi_{12}(-,-)\ \phi_2(-)\ \phi_{23}(-,-)\ \phi_3(-)\ \phi_{34}(-,-)\ \phi_4(-)}_{Z}}$$

Focus: MAP Inference Problem

- *Maximum A Posteriori* inference task
- Question: What is **most likely** set of labels for graph?



$$\underset{X1,X2,X3,X4}{\operatorname{argmax}} \left[\frac{\phi_1(-)\ \phi_{12}(-,-)\ \phi_2(-)\ \phi_{23}(-,-)\ \phi_3(-)\ \phi_{34}(-,-)\ \phi_4(-)}{Z} \right]$$

Actual MAP Inference Formulation

$$\underset{X1,X2,X3,X4}{\text{argmax}} \left(\frac{\phi_1(-)\ \phi_{12}(-,-)\ \phi_2(-)\ \phi_{23}(-,-)\ \phi_3(-)\ \phi_{34}(-,-)\ \phi_4(-)}{Z} \right)$$

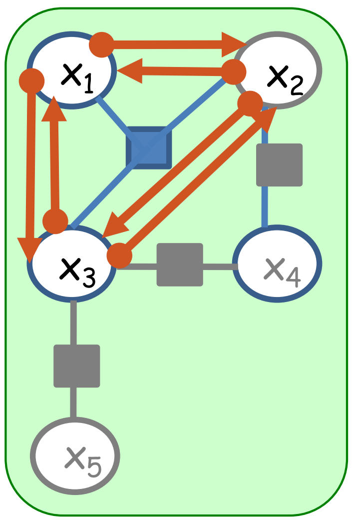
Ignore Z – it's a constant, doesn't matter

Let $\Theta = -\log\phi$ (from stat physics…)

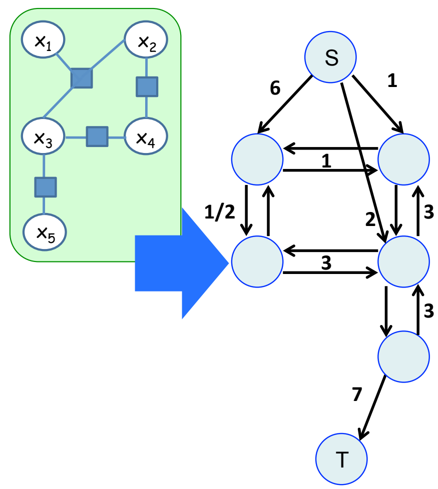
$$= \underset{X1,X2,X3,X4}{\text{argmin}} \underbrace{\left(\Theta_1(-)+\Theta_{12}(-,-)+\Theta_2(-)+\Theta_{23}(-,-)+\Theta_3(-)+\Theta_{34}(-,-)+\Theta_4(-) \right)}_{\text{“Minimum Energy” formulation of MAP inference}}$$

Pitt Research

“Big 3” Inference Methods for PGMs

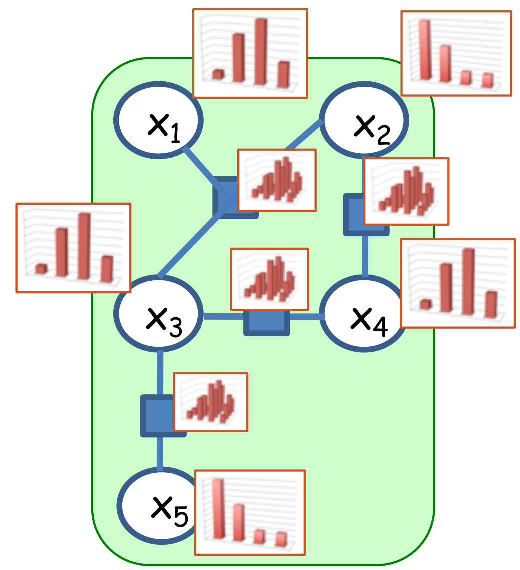


Belief Propagation



Graph Cuts

(→ Network Flow)



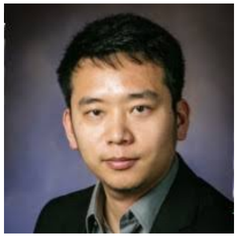
Sampling

(Gibbs/MCMC)

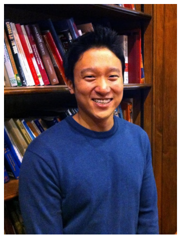
Pitt Research

Key Collaborators

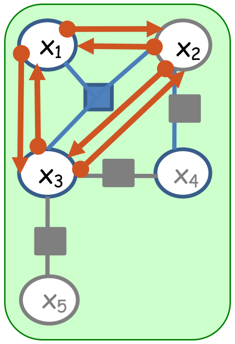
Jungwook Choi
PhD Illinois ‘15
Hanyang University



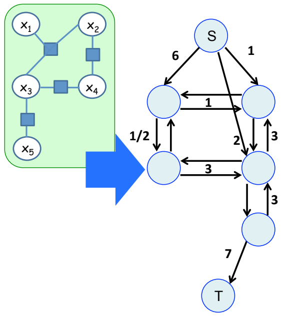
Tianqi Gao
PhD Illinois ‘20
Apple SEG



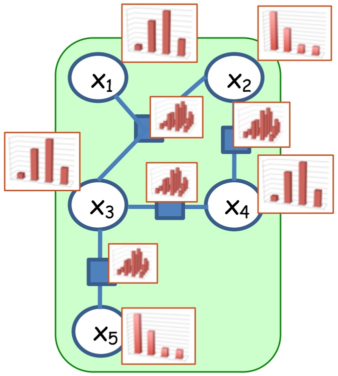
Glenn Ko
PhD Illinois ‘17
Harvard



Belief Propagation



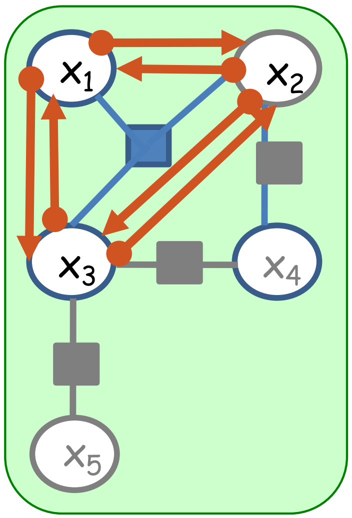
Graph Cuts
(→ Network Flow)



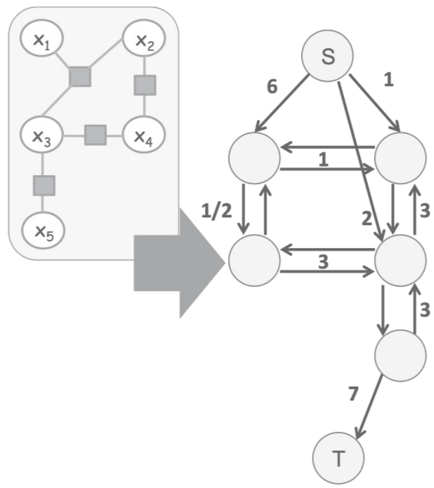
Sampling
(Gibbs/MCMC)

Pitt Research

"Big 3" Inference Methods for PGMs



Belief Propagation



Graph Cuts

(→ Network Flow)

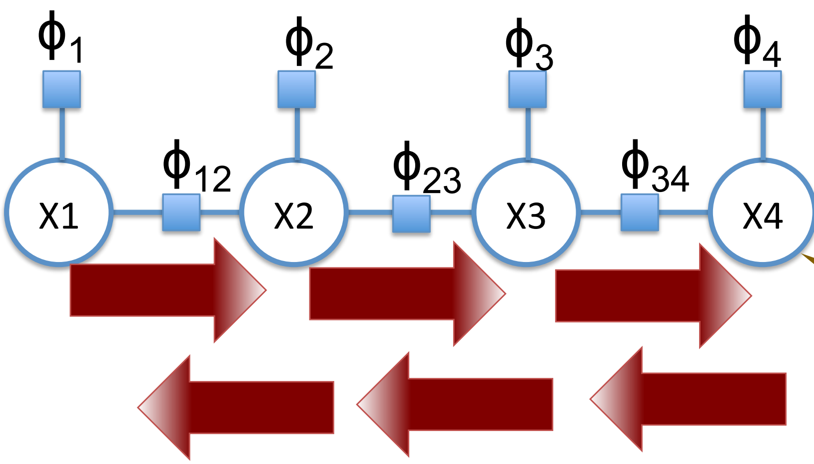
Sampling

(Gibbs/MCMC)

Pitt Research

Belief Propagation: Iterative & Local

- Smart order of **local, message passing** computations (like Viterbi!) that calculate a **“belief”** per label, per node

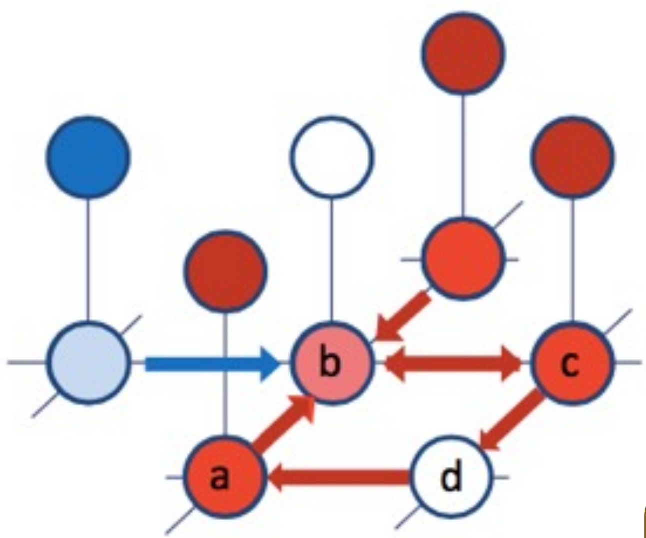


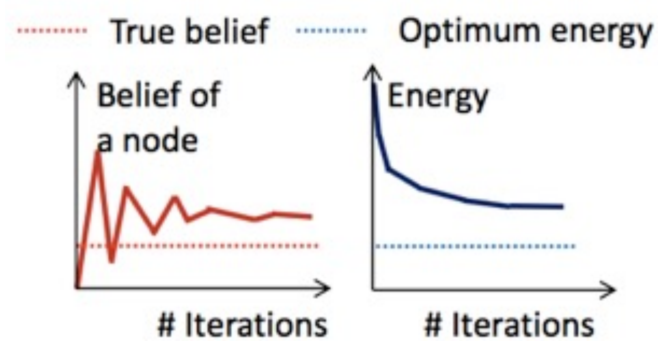
If graph is a chain/tree, BP converges to **optimum in 2 passes** of messaging

Pitt Research

Belief Propagate: Iterative & Local

- But if graph has **loops** – **no guarantee** of convergence!



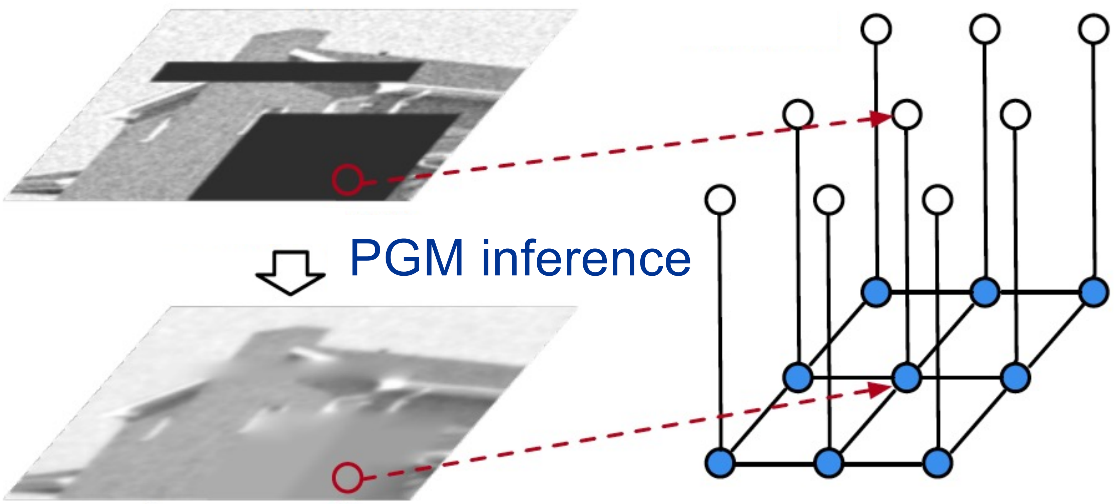


If graph is "**loopy**" -- **not** a chain/tree, simple BP message passing may **diverge**

Pitt Research

Why We Care: Images are PGMs

Original Image



Processed Image

Input pixel Yi = constant

Output pixel Xi = TBD

Image MRF

A pixel = 5-connected node
4 compass neighbors
+ original pixel

Bayesian

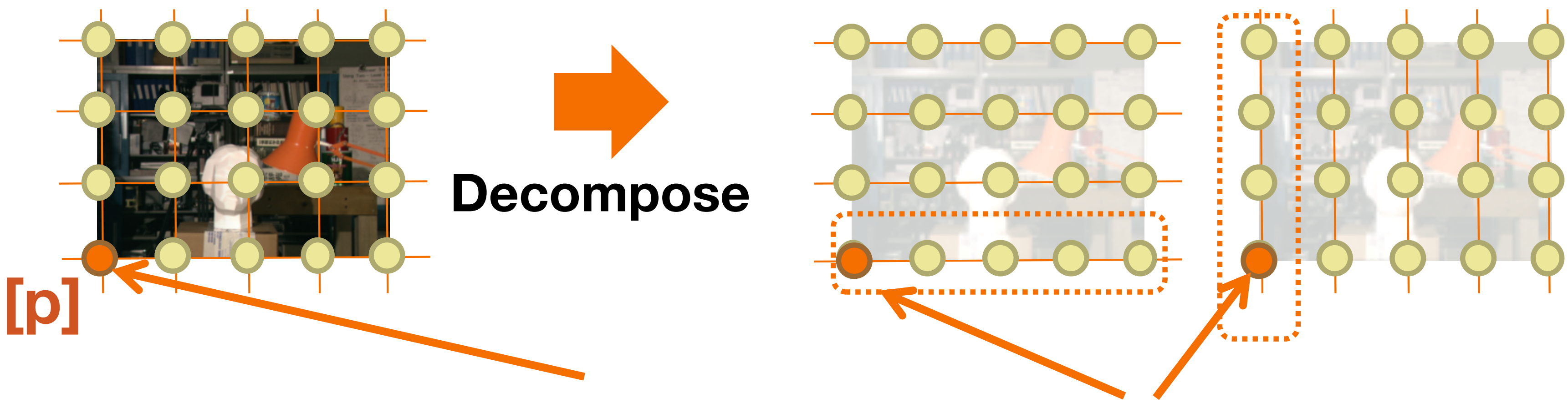
$$P(X|Y) = \frac{P(Y|X)\,P(X)}{P(Y)}$$

$$= (1/Z)\,\Pi\phi \sim -\Sigma\theta$$

Pitt Research

Our BP: Sequential Tree-Reweighting

- **Idea**: Decompose a loopy graph to a set of **trees**, do inference sequentially across trees, recombine “**smart**”
  - [Kolmogorov PAMI’06]: Empirically good on loopy case; **slow**

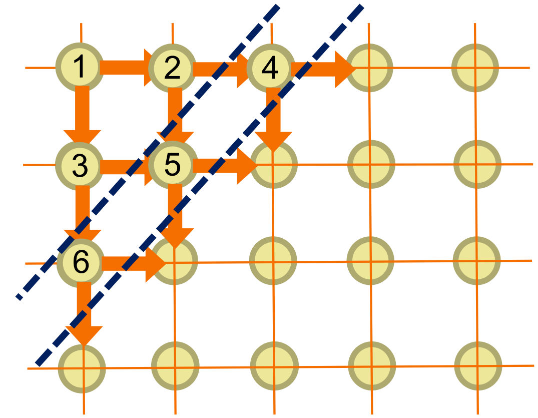


Recombine “smart”: Energy[p] = weighted sum from decomp

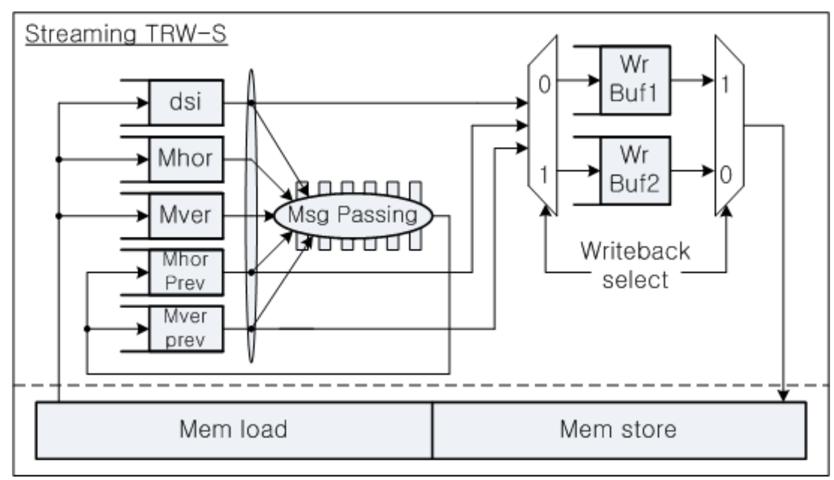
Pitt Research

HW: First, Streaming "Diagonal Order" Arch

Key: Diagonal ordering **of all message pass → parallelism**



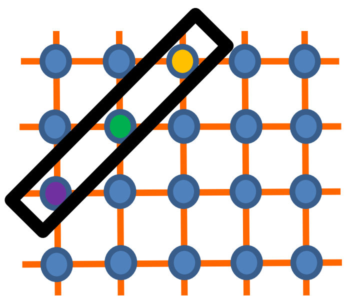
- **Decoupled, streaming arch**
- **Launch/retire 1 pixel/clock**
  - **Complete** label-set likelihood updates (~1Kb) for all labels
- **14-stage pixel pipeline**
  - So: **14 pixels** "in flight" / clock



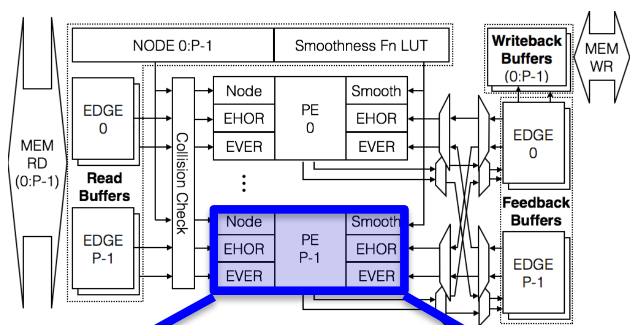
Pitt Research

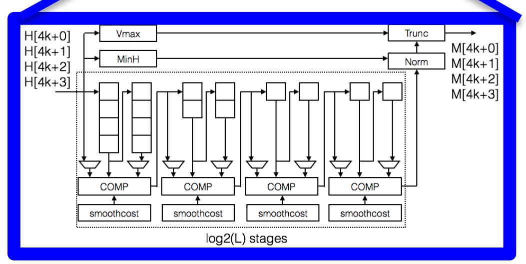
Next: Parallel/Configurable Pipes

- Not just one pipeline any longer: ***more* parallel**…



P Parallel
processor
elements
(pixel streams)





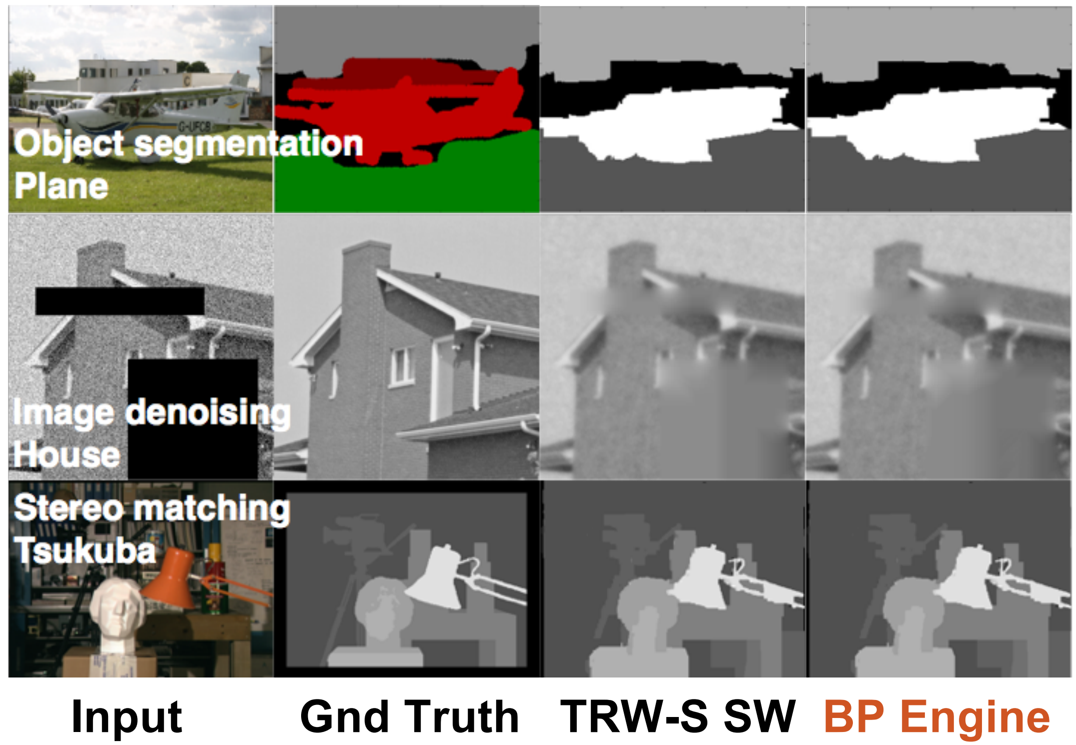
Efficient memory subsystem overlaps BW and computation, checks for data conflicts

Novel, configurable Factor-Eval Units removes O(|labels|^2) complexity (FFT tricks)

Pitt Research

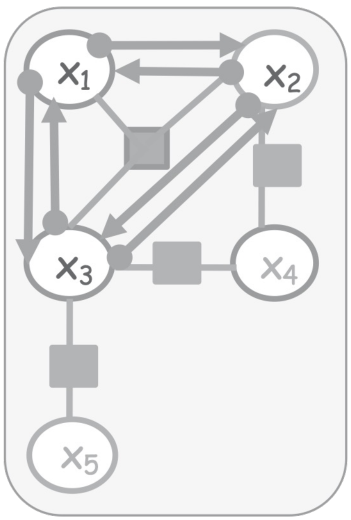
Results: Configurable BP Engine

- Xilinx **Virtex5** FPGA
- **12-40X** faster than SW (PE = 4, ~2015)
- No loss of quality
- First custom HW to **run >1** Middlebury ML benchmark

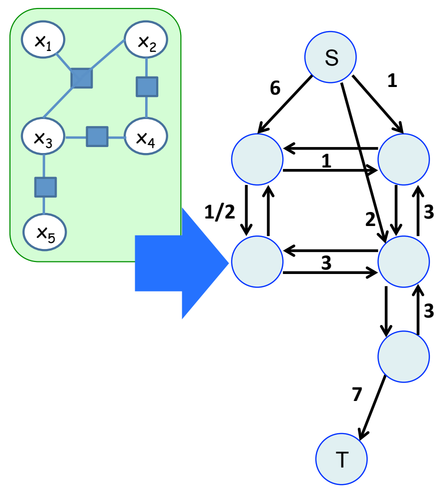


Pitt Research

“Big 3” Inference Methods for PGMs



Belief Propagation



Graph Cuts

(→ Network Flow)

Sampling

(Gibbs/MCMC)

GC: Transform from MRF to Network Flow

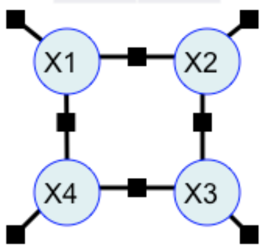
Start: Binary Label MRF

X_1 X_2	$\log\phi_{12}$
0 0	0
0 1	6
1 0	6
1 1	0

X_4	$\log\phi_1$
0	7
1	0

X_2	$\log\phi_2$
0	0
1	2

X_1 X_4	$\log\phi_{14}$
0 0	0
0 1	1
1 0	1
1 1	0



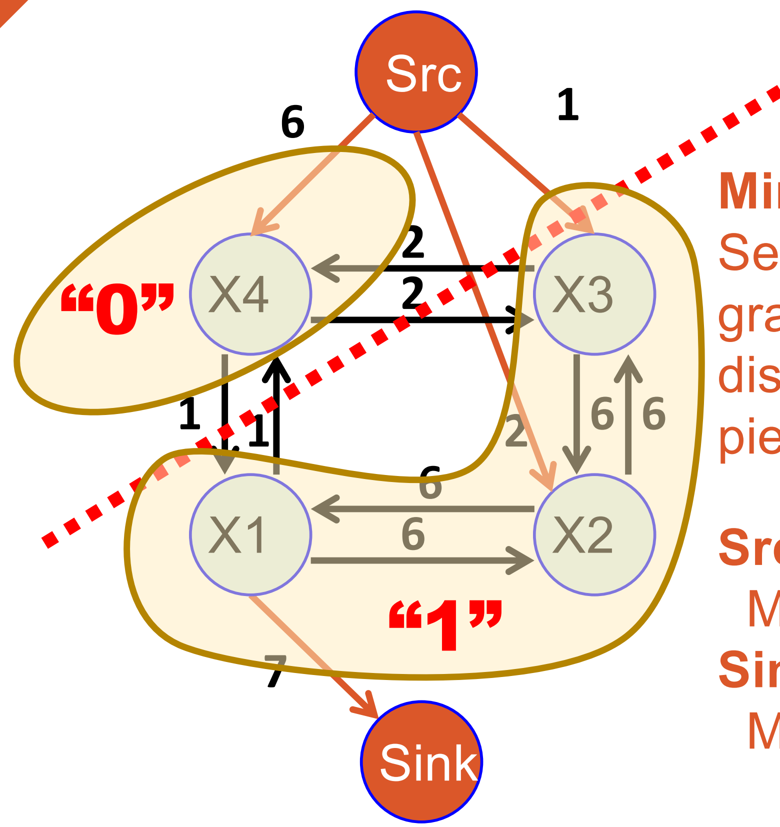
X_2 X_3	$\log\phi_{23}$
0 0	0
0 1	6
1 0	6
1 1	0

X_4	$\log\phi_4$
0	0
1	6

X_3 X_4	$\log\phi_{34}$
0 0	0
0 1	2
1 0	2
1 1	0

X_3	$\log\phi_2$
0	0
1	1

Build: Network Flow Graph



Min Edge Cut
Separates graph into 2 disconnected pieces

Src-side:
MAP value **0**

Sink-Side:
MAP value **1**

Pitt Research

GC: Why Hardware

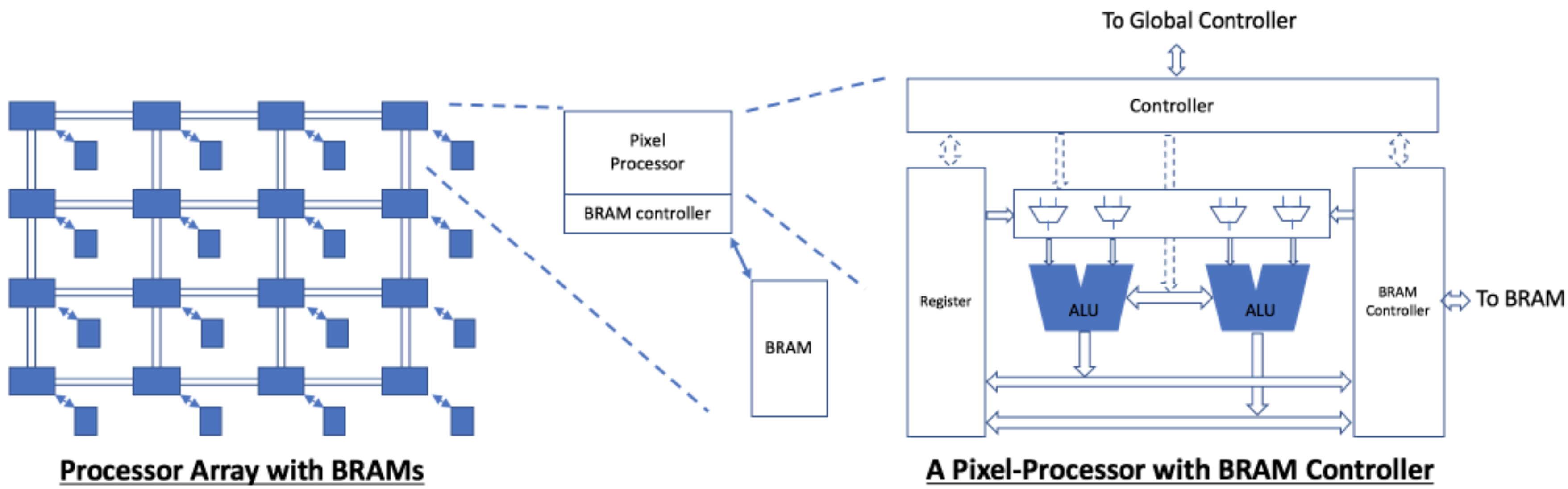
- **Push-Relabel Network Flow:** a ”min cut” algorithm can be executed (almost) entirely with just neighbor values
- **Neighbor**: Nodes that share an edge in PGM (N-E-W-S)
- **Iterative and Convergent:** a “well behaved” algorithm

→Perfect for **large images**, modeled as **grid-MRFs**

→There are tricks for doing **gray-scale/color** images

GC: Pixel-Parallel Array Processor

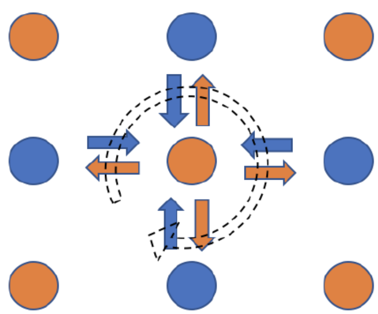
- FPGA target, one processor per pixel



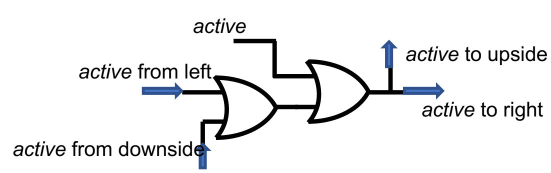
Pitt Research

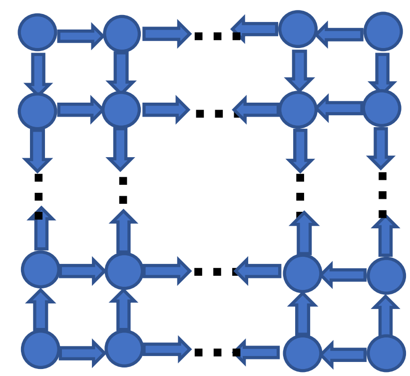
Pixel Processor: Key Tricks

- **Serial** bottlenecks
  - Cannot push flow to a node that is already “pushing” out
- **Solution**: Checkerboard scheduling + ordering



- **Not** all local: Global convergence detection
- **Solution**: O(rows+cols) shift register to array center to check activity



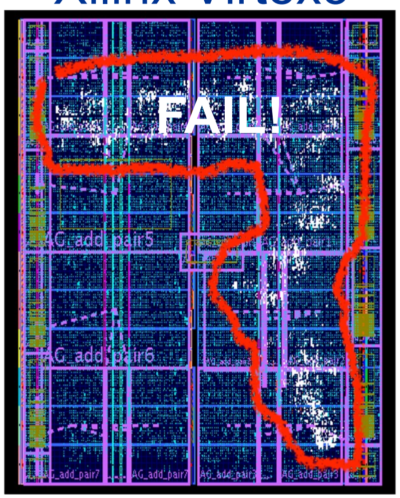


Pitt Research

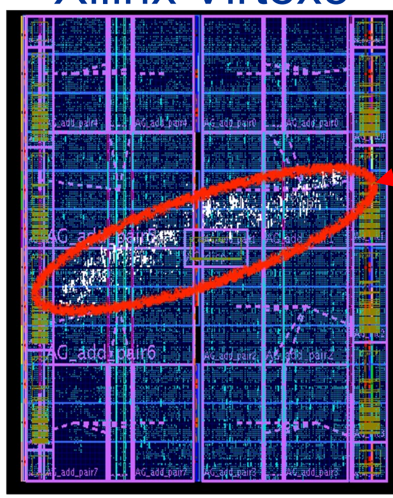
Array "Tile": How Big?

- Interesting example of logical physical co-design

256 pixel = 16x16
Xilinx Virtex5

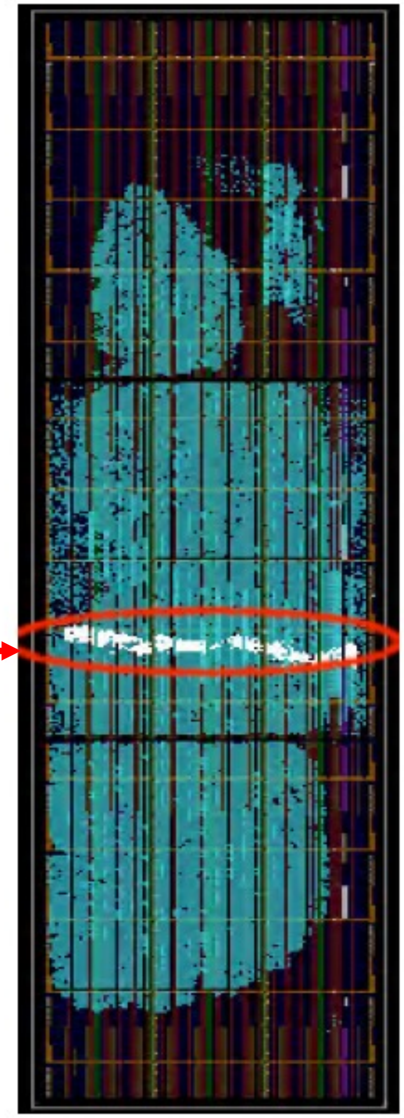


256 pixel = 8x32
Xilinx Virtex5



1 row of processors

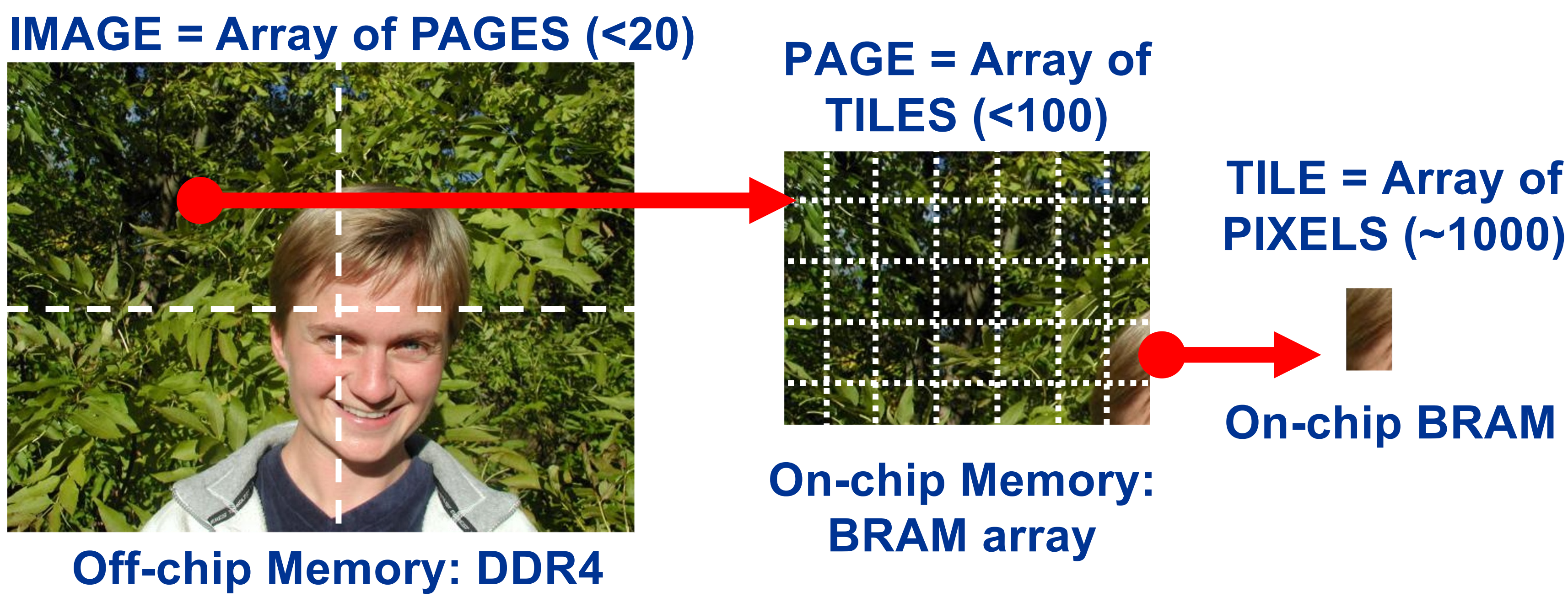
1024 pixel = 16x64
Xilinx Virtex
Ultra-Scale



Pitt Research

Next: What About “Big” Images?

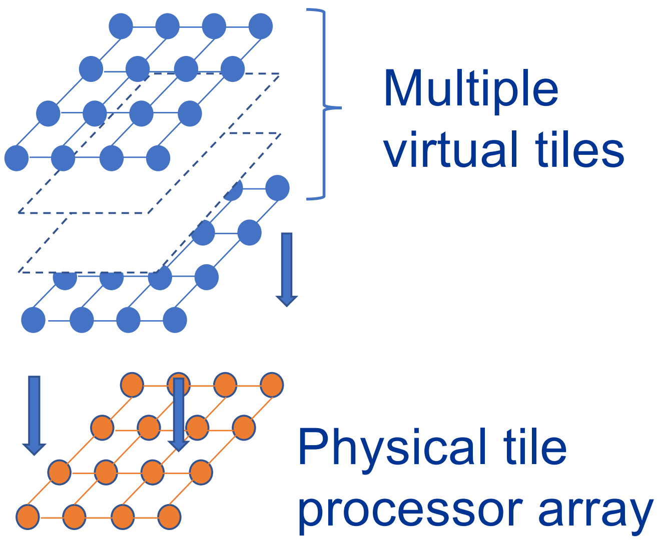
- We built a full **virtual tile (memory)** system on array



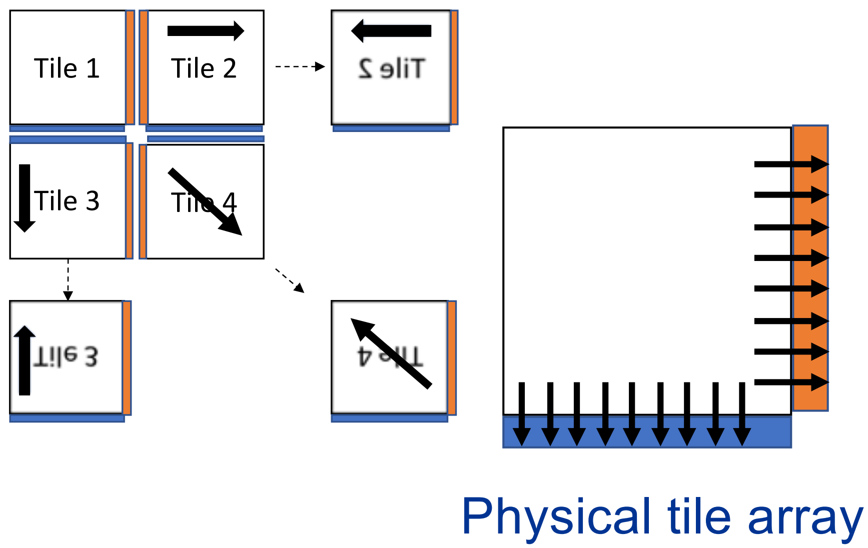
Pitt Research

Virtual Tile Architecture

- Virtual tiles "**stack**" on the physical tile array on-chip



- Geometric nuances (lots!) at tile (and page) **edges**



GC Virtual Tile Engine: Results

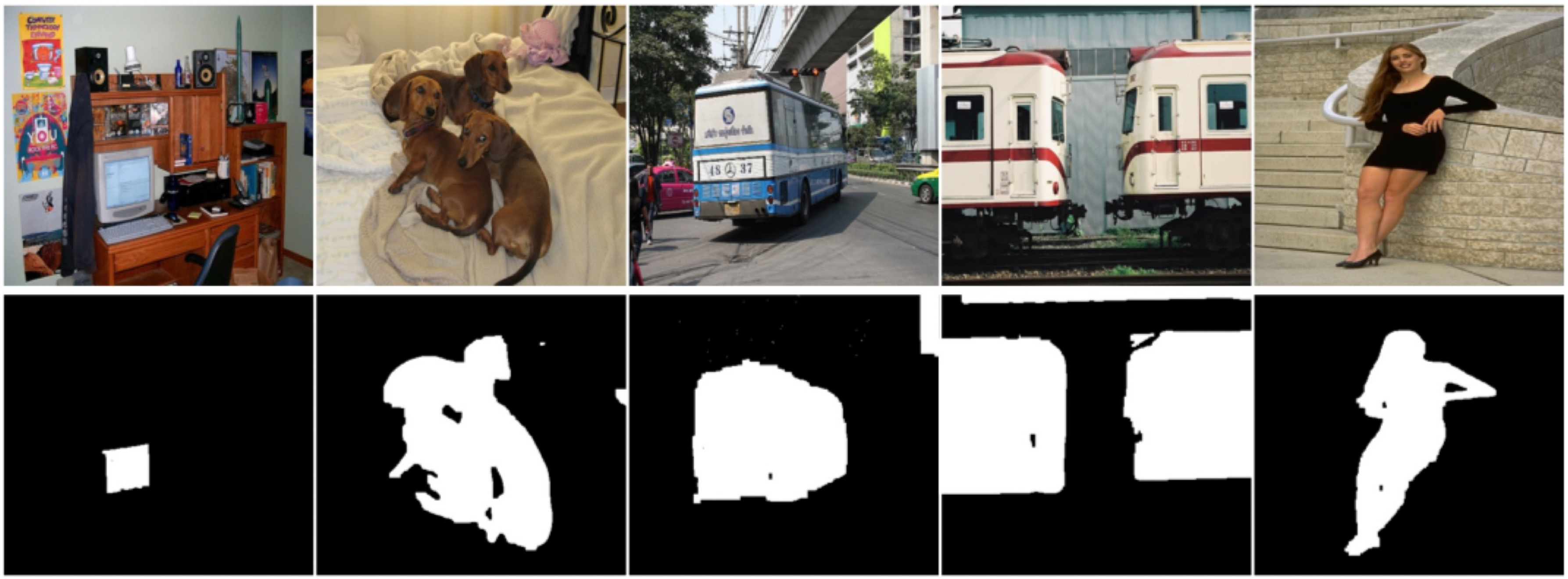
1536-pixel tile array
AWS F-node
(Xilinx UltraScale)

Frequency	Array Size	Slice LUTs	BRAM	BRAM per Pixel Processor	AXI Memory Protocol	Page Size	Ultra RAM
125Mhz 0.018ns Slack	16x96 = 1536 pixel processor 2*(16+96)=224 shadow processor	86.6%	66.5%	18Kb	512b wide 192 Burst length	60 Virtual Tiles 11.25Mb	16%

	Our Hardware	CUDA Cuts 2	Kobori et al [1] FPGA 1	Nikitakis et al [2] FPGA 2	Szeliski et al [15] CPU 1	CPU 2	CPU 3
Device	Virtex UltraScale+	Nividia Titan	Virtex 6	Virtex 7	NA	Intel Xeon E5	Intel i5
Frequency	125 MHz	1405 MHz	201 MHz	260 MHz	NA	3.4GHz	3.1GHz
Image Flower (600x450)	9.93 ms	11.67ms	30.7 ms	NA	188 ms	1.03 s	2.553 s
Image Person (600x450)	12.27 ms	16.09 ms	36.7 ms	NA	140 ms	1.9 s	4.716 s
Image Sponge (640x480)	7.88 ms	12..99 ms	45.8 ms	NA	142 ms	1.29 s	2.67 s
Synthesis (128x128)	0.13 ms	NA	NA	0.95 ms	NA	NA	NA
50 Images Average	8.20 ms	17.23 ms	NA	NA	NA	NA	NA
Avg speedup	1X	2.10X	5.77X	7.28X	23.44X	240X	506X

Pitt Research

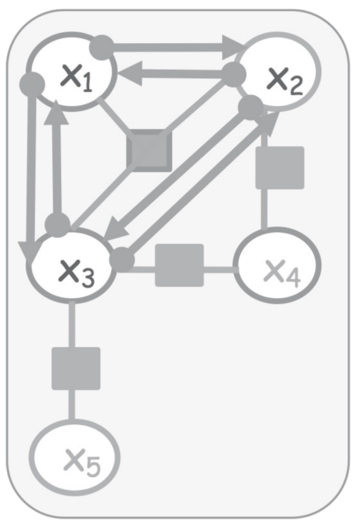
And – It Really Works on Real Images



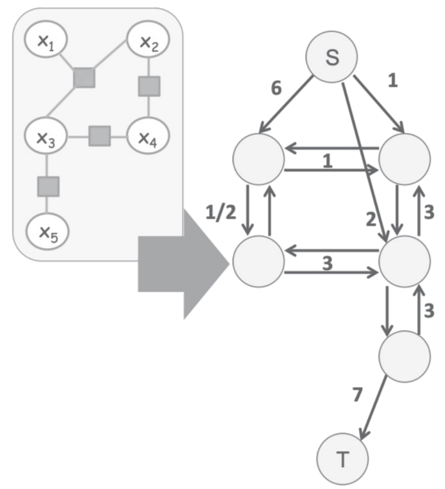
[1] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother, "A comparative study of en- ergy minimization methods for markov random fields with smoothness- based priors," IEEE transactions on pattern analysis and machine in- telligence, vol. 30, no. 6, pp. 1068–1080, 2008.
[2] A. Nikitakis and I. Papaefstathiou, "Highly efficient reconfigurable par- allel graph cuts for embedded vision," in Proceedings of the 2016 Con- ference on Design, Automation & Test in Europe. EDA Consortium, 2016, pp. 1405–1410.
[3] V. Gulshan, C. Rother, A. Criminisi, A. Blake, and A. Zisserman, "Geodesic star convexity for interactive image segmentation," in Pro- ceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2010.

Pitt Research

“Big 3” Inference Methods for PGMs

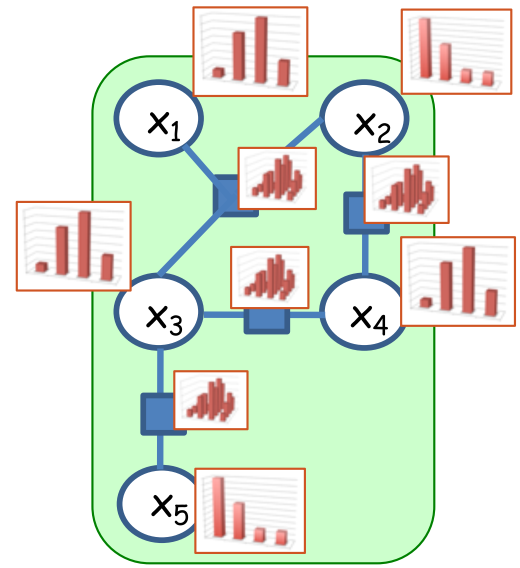


Belief Propagation



Graph Cuts

(→ Network Flow)

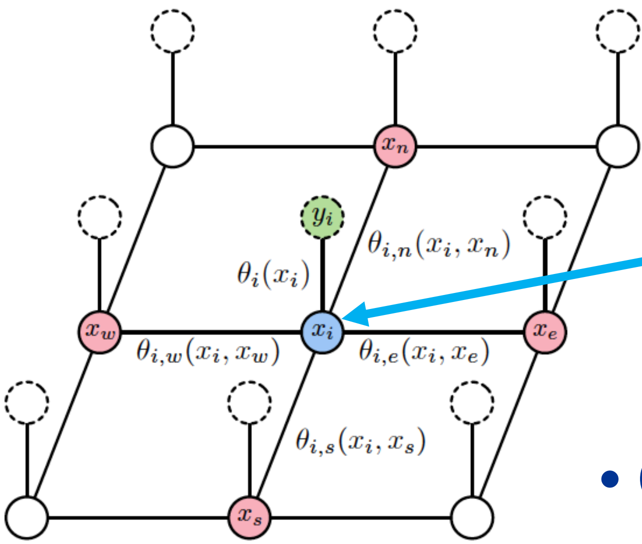


Sampling

(Gibbs/MCMC)

Pitt Research

Gibbs Sampling: Serial Baseline



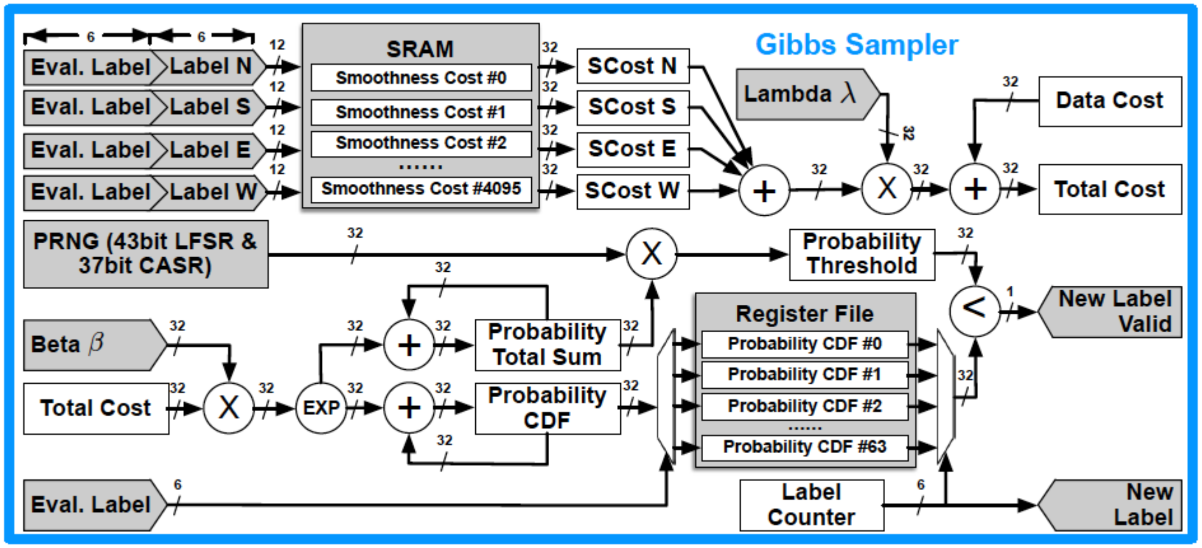
Gibbs sampling on MRF

Initialize x^0
for $t = 0$ to T **do**
  for $i = 0$ to N **do**
    $\boxed{x_i^{(t+1)} \sim P(x_i | x_{north}^{(t)}, x_{south}^{(t)}, x_{west}^{(t)}, x_{east}^{(t)})}$
  end for
end for
return x

- Generate **samples of xi**, from right prob distribution, **based on neighbors**
- Lets us compute **Pr(xi = Label[k])** **∀ k**
- Like GC: **iterate** to convergence

Pitt Research

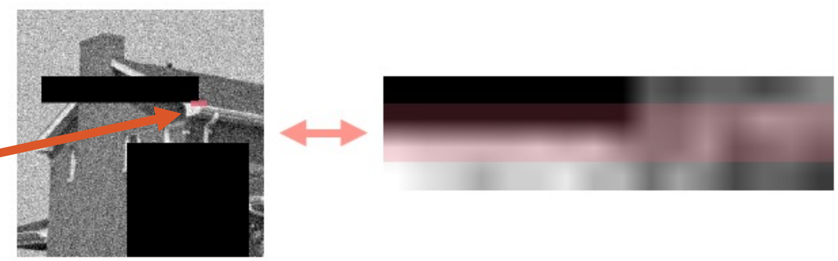
Gibbs Sampler (GS) Core



- Up to 64 labels/node
- 32b variable fixed-pt
- Tightly coupled PRNG
- Iterative architecture for small footprint

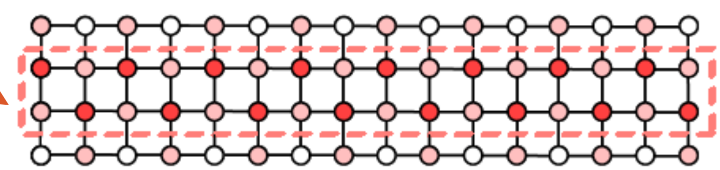
Two Levels of Parallelism

Sample **independent tiles** in parallel – treat as if they were separate images



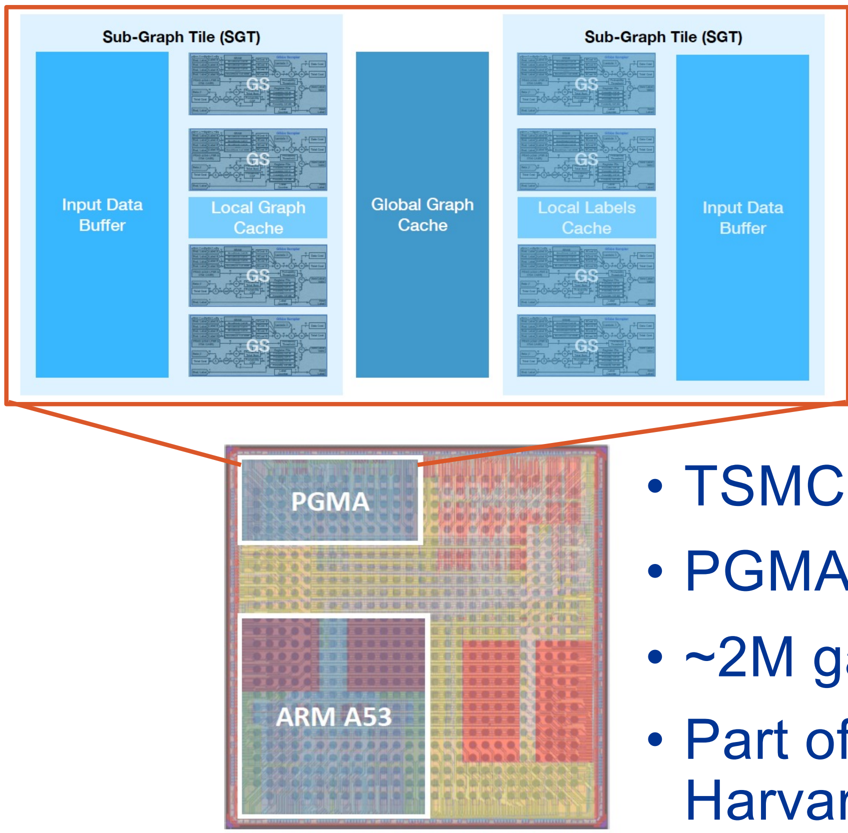
while (< max Gibbs sampling iterations)
  foreach (**tile in an image**)
    while (< max tile sampling iterations)
      foreach (**node in a tile**)
        sample (*)

Sample **independent nodes** in parallel – checkerboard / graph coloring schedule



Pitt Research

PGMA: Prototype PGM Accelerator



A 3mm^2 Programmable Bayesian Inference Accelerator for Unsupervised Machine Perception using Parallel Gibbs Sampling in 16nm

Glenn G. Ko[1], Yuji Chai[1], Marco Donato[1], Paul N. Whatmough[1,2], Thierry Tambe[1], Rob A. Rutenbar[3], David Brooks[1], Gu-Yeon Wei[1]

[1]Harvard University, MA, [2]Arm Research, MA, [3]University of Pittsburgh, PA

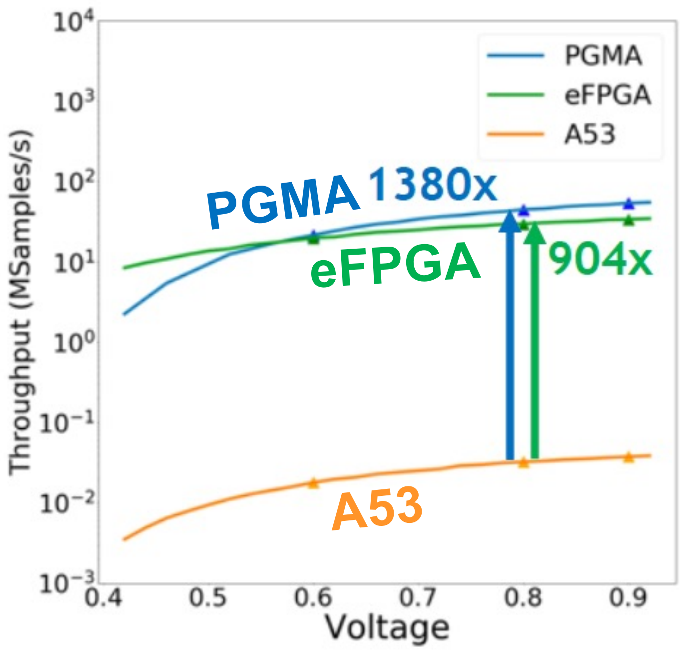
VLSI 2020

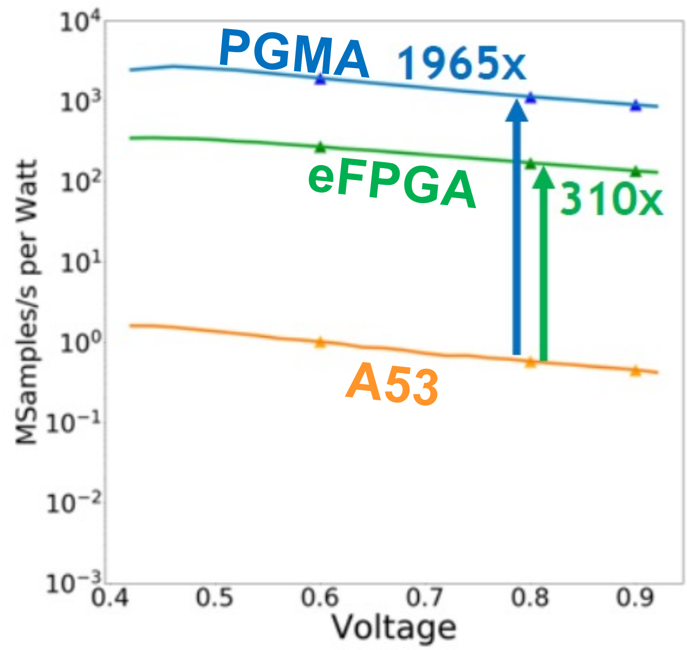
- TSMC 16nm FFC
- PGMA area: 2.3 x 1.3mm^2
- ~2M gates, 450MHz
- Part of a larger SOC experiment at Harvard called **SM5: ML for IOT**

Pitt Research

Refs: Ko et al., VLSI 2020. Whatmough et al. VLSI, 2020.

PGMA vs A53 vs eFPGA (on SOC)

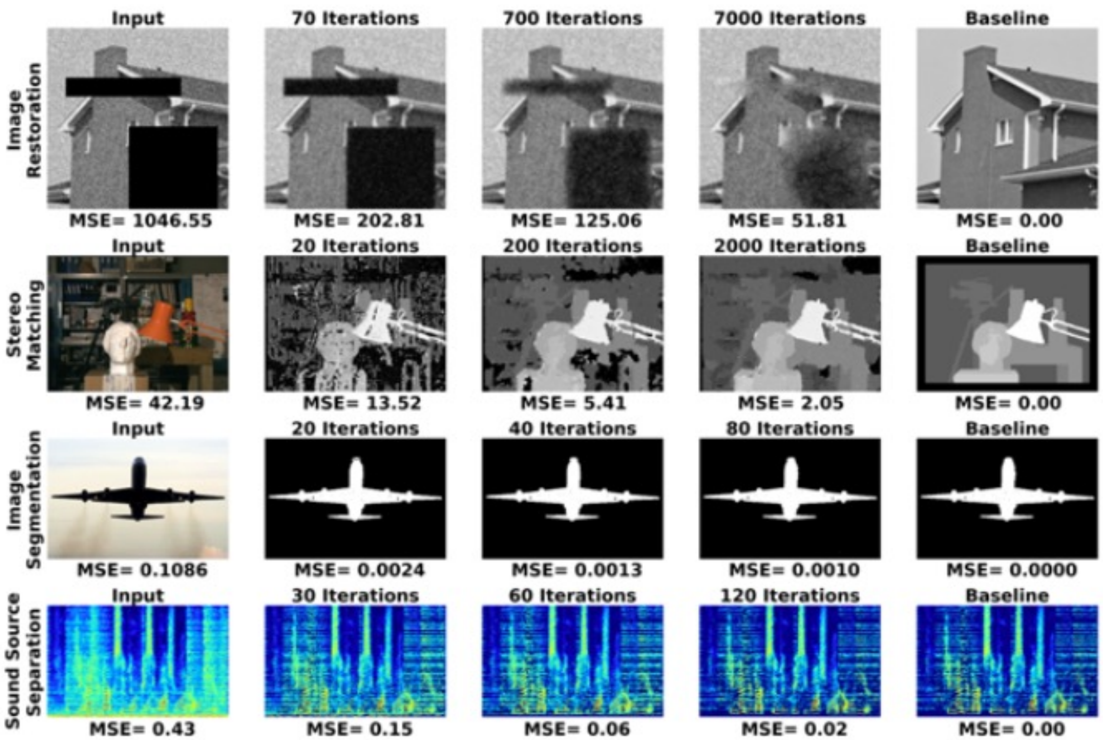




- **PGMA**: **1000X+** throughput vs CPU; **6X+** ops/W vs eFPGA

Pitt Research

PGMA ML Results



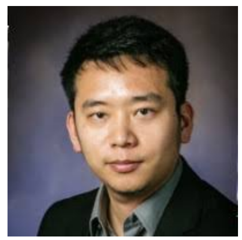
Four example applications:
- Image restoration
- Stereo matching
- Image segmentation
- Sound source separation

Features:
- No labeled dataset
- Completely unsupervised
- Both training and inference on-the-fly

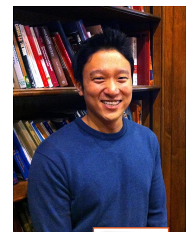
Conclusions

∃ (AI apps **X**) [interesting(**X**) ∧ ¬DNN(**X**) ∧ hardwareworthy(**X**)]

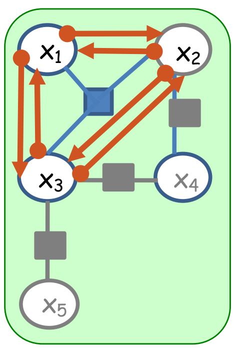
Jungwook Choi
PhD Illinois '16
Hanyang University



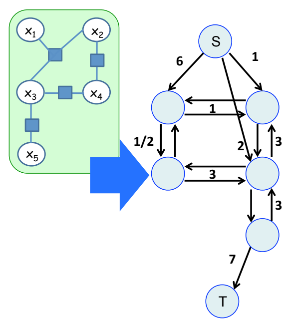
Tianqi Gao
PhD Illinois '20
Apple SEG



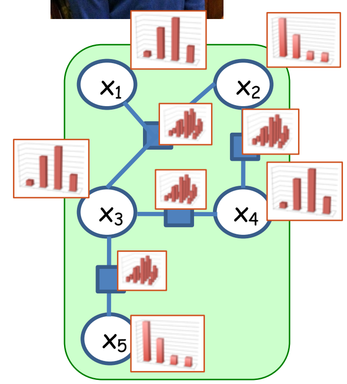
Glenn Ko
PhD Illinois '17
Harvard



Belief Prop



Graph Cuts



Sampling

Pitt Research

Acknowledgements

- Contributors:
  - **Harvard**: Yuji Chai, Marco Donato, Paul N. Whatmough, Thierry Tambe, David Brooks and Gu-Yeon Wei
  - **Illinois**: Paris Smaragdis, Minje Kim, Shang-nien Tsai

- Sponsors:
  - DARPA/SRC: FCRP C2S2, SONIC, JUMP ADA
  - DARPA CRAFT
  - Intel and ARM

Pitt Research