When Storage Devices Become Computers

Robert Thibadeau, Ph.D.

Seagate Technology, LLC & CMU

Pittsburgh, PA

December 2008

Collaborators

Kevin Gomez Architect, Seagate Research

Tom Mitchell, Professor of AI and Machine Learning and head of Machine Learning Department CMU

Terence Sejnowski, Professor and Head of the Computational Neurobiology Laboratory, Salk Institute

Dave Touretzky, Research Professor, Center for the Neural Basis of Cognition, CMU

Last Summer Interns:

Avleen Bijral, Ph.D. Candidate, Univ of Chicago

Alexander Grubb, Ph.D. Candidate, CMU

Agenda

- Research on how much sense it makes to put processing into non-volatile storage devices like hard disks and flash drives
- Looking at a class of problem we call “Non-Linear Stochastic Processing” or NLSP
- Some basic modelling / optimization results for well-known NLSP problems

Memory Bandwidth

Storage Density and CPU speeds have kept pace with Moore's Law
Interface speed between CPUs and Memory has not.

The disparity has grown 2 orders of magnitude over the last 15 yrs and promises to get worse due to fundamental limits of wire (RC) delays and speed of light.

Current advances in computing are attributable to multi-core architectures with cores/chip predicted by Intel to double annually.

With processing speed increasing at 60%/yr and solid-state memory speed increasing 7%/yr, maintaining a balanced architecture (100s of GB per teraflop of computing) is a memory bandwidth problem which becomes prohibitively expensive to scale.

Seagate is the world's largest producer of non-volatile digital memory
R&D includes hybrid architectures which span the continuum of rotating-media to solid-state memory.

Current storage devices have CPUs and multiple DSPs, and millions of lines of code to manage data access. Should these resources be exploited differently?

Opportunity?

Non-linear stochastic processing (NLSP) problems are an increasingly important class of machine learning and pattern recognition problems characterized by **primarily local**, potentially **probabilistic interactions** between elements.

Exploiting large scale problems in this class could require **~10^{10} units** with **~10^{5} bytes per unit.** Optimizing performance/$ and performance/Watt suggests a new memory centric architecture.

Applications include iterative machine learning algorithms for web-scale data, deep multi-layer neural networks working on large datasets, social networks, and processing of biomedical 3-D CT-scan and 4-D fMRI imaging data.

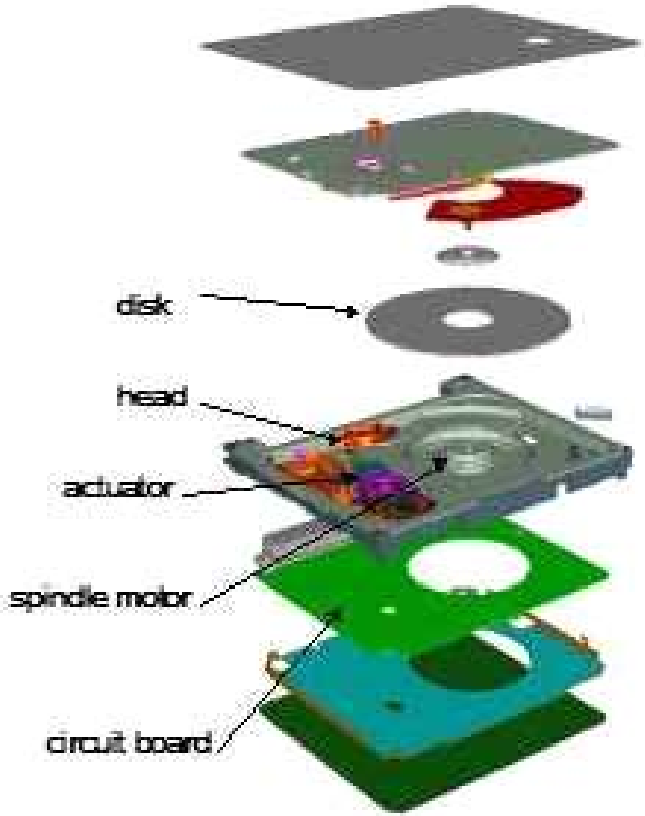
Could specialized storage devices play a significant role in realizing large scale NLSP applications?

Example Modifying Conventional Disk Drive

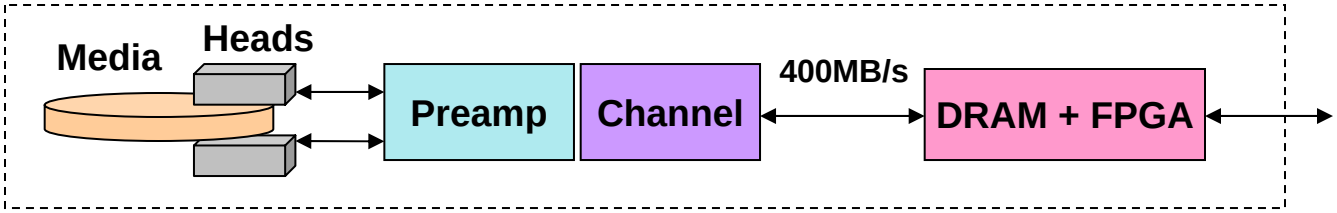
A hard disk drive records data on rotating disks coated with magnetic media. A head suspended on the end of a rotary actuator flies on an air-bearing several angstroms over the surface of the disk. An electromagnet in the head flips magnetic domains in the media to write bits and a magneto-resistive element is used to sense these recorded bits.

Current drives have areal densities in the range of **150×10^9 bits per square inch** (150Gbpsi) with track densities of **150 thousand tracks per inch** and linear densities of **1 million bits per inch**. Areal density has been increasing at a rate of 40% per year.

Increasing performance for NLSP – A current four platter, eight head**,** drive can read or write at 220 MB/s. Only one of the eight heads is accessed at a time. But what if we alter the electronics to access all the heads simultaneously, in parallel? Using FPGA technology for programmable read-modified-write for each head, gives non-volatile internal processing rates exceeding 1.7 GB/s based on existing drive mechanics. More radical, much faster, architectures involving more heads are believed to be feasible.


disk
head
actuator
spindle motor
circuit board

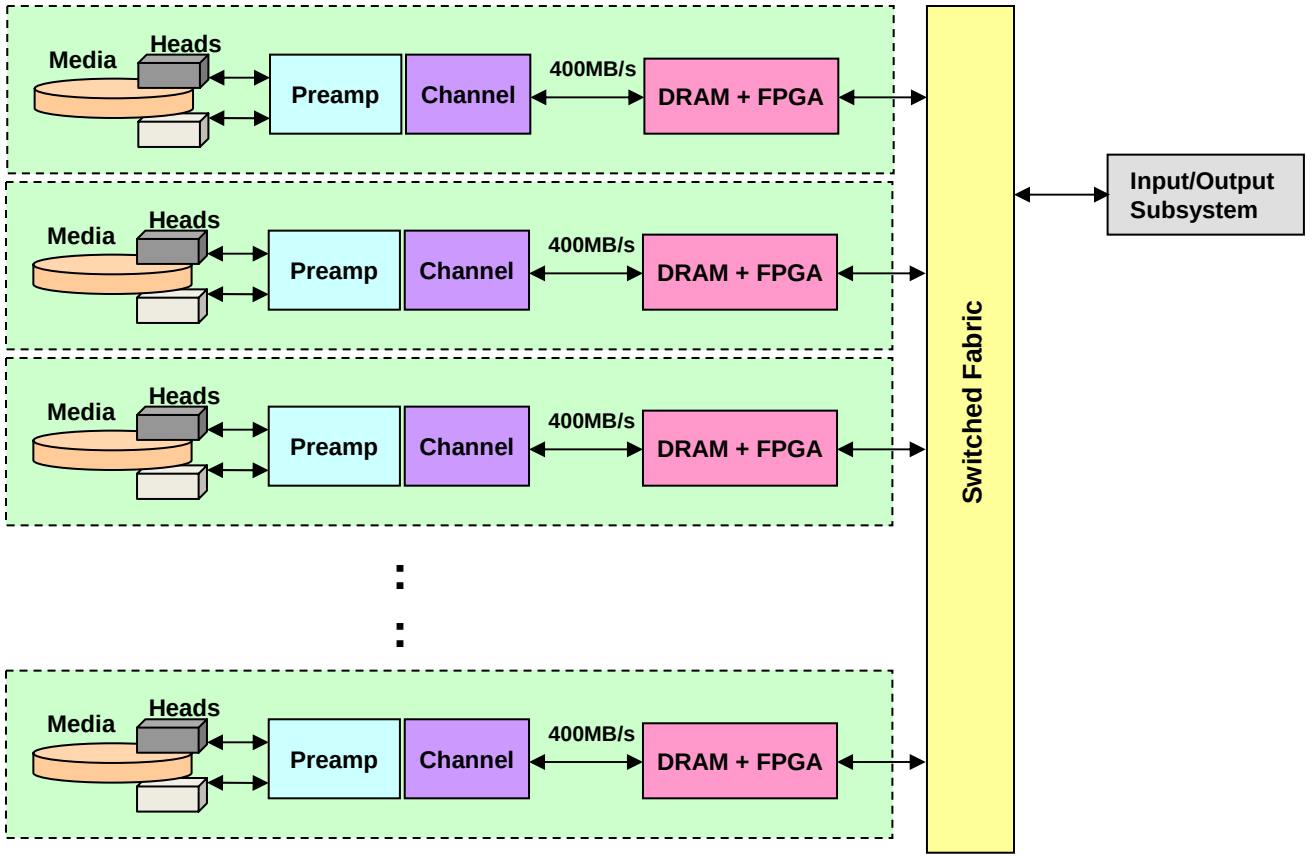
Storage-Compute Element – 'One Lane'



Capacity : 200 GB ↑ 40% / yr
Bandwidth: 400 MB/s ↑ 18% / yr
Cost ~ Enterprise+ $/GB ↓ 30% / yr

FPGA implements compute array and disk or solid-state memory controller

Compute System



Resources for Spindle, Actuator, Servo & Main Processor shared across Multiple Units

Example

Consider a NLSP system with 10^{10} units, 10^5 bytes per unit

1 PB (10^{15} byte) of memory
Using current technology - 5,000 platters (200GB/platter)
Assume 2 heads per platter in parallel - 400MB/s data rate
Memory bandwidth – **2 TB/s (2 x 10^{12} byte/s)**
Total cost of memory ~ ***$200,000***

Compare with ~ *$200,000,000* for 1 PB of DRAM

First cut NLSP Drive Model Results

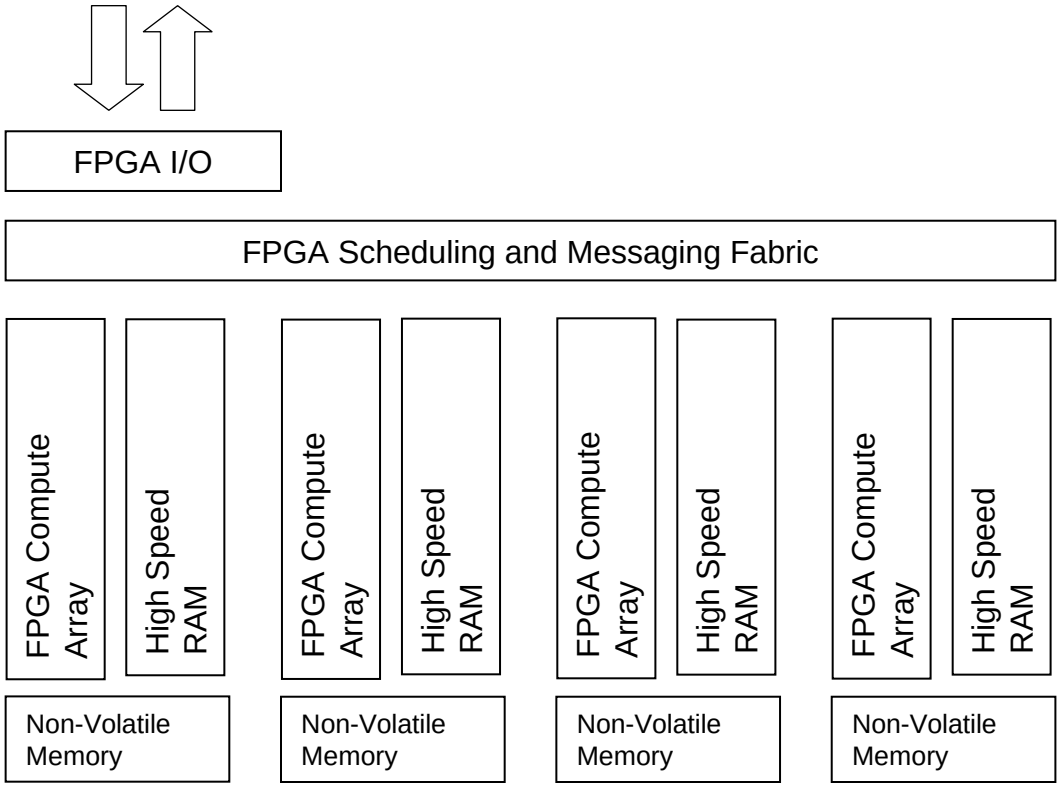
			Recognition Drive Design Parameters		
		Today's Conventional Drive	More RDrives	Less RDrives	100 RDrives
Number of Neurons		10,000,000,000	10,000,000,000	10,000,000,000	10,000,000,000
R/W Bits per Second Per Head		1,000,000,000	2,000,000,000	2,000,000,000	2,000,000,000
Size of Neuron in Bits		1,000,000	1,000,000	1,000,000	1,000,000
Number of Drives		4,166,667	10,000	1,000	100
T in seconds		0.05	0.05	0.05	0.05
R/W Bits per T		50,000,000	100,000,000	100,000,000	100,000,000
Percent of Neurons in each T		1%	1%	1%	1%
Heads Per Drive		24	100	1,000	10,000
Neurons Per Drive		2,400	1,000,000	10,000,000	100,000,000
Drive Capacity (Bytes)		300,000,000	125,000,000,000	1,250,000,000,000	12,500,000,000,000
External 16 Way Switch Depth		6	4	3	3
External Number Switches		277,781	667	67	7
Internal 16 Way Switch Depth		2	3	3	4
Internal Number Switches		2	7	67	667
Notes		Impractical:Over a Million Disk Drives Required	Drive would likely have more space than this	Probably OK in Space, still 1000 Drives	Aggressive in Space and Heads, but now 100 Drives

Petabyte, 1-2 Terabit/s System Evolution

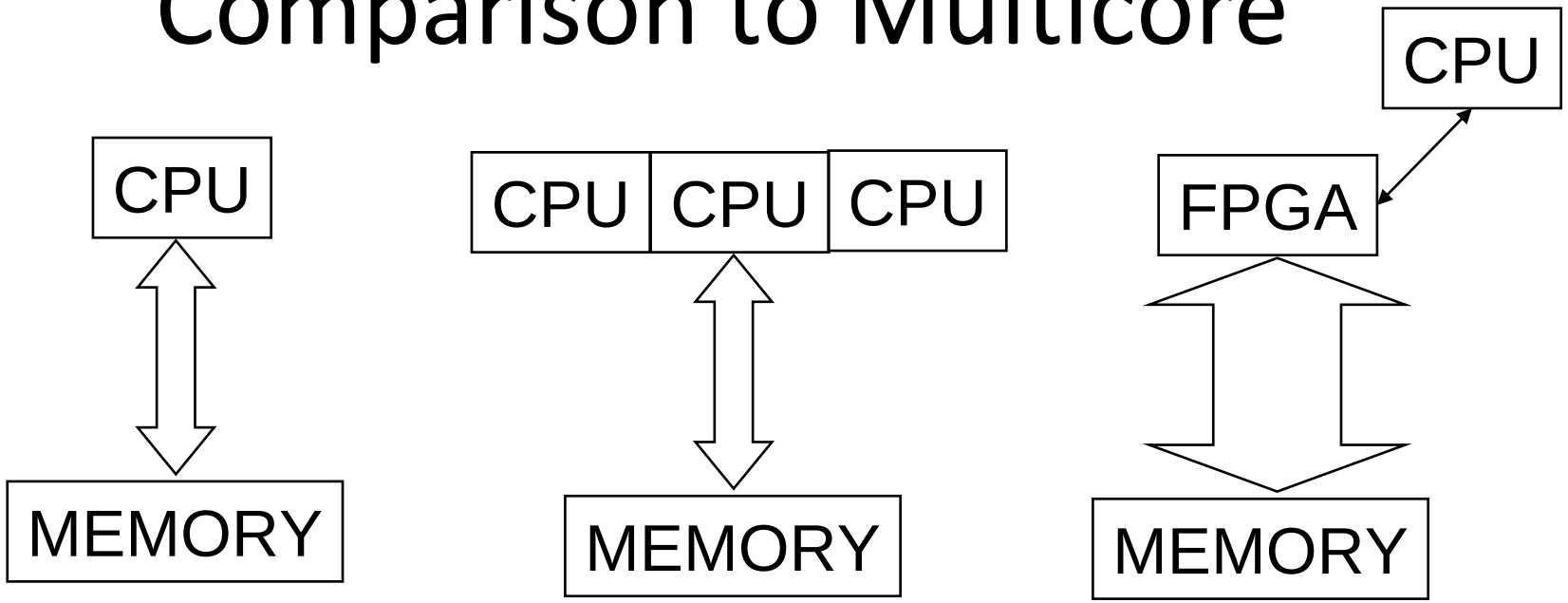
Year	GB/disk	# of disks	Heads/ Surface	#Hd	MB/s/Hd	TB/s
2008	280	3571	1	7143	237	1.7
2010	549	1822	1	3644	331	1.2
2012	1076	930	2	3719	464	1.7
2014	2108	474	3	2846	649	1.8
2016	4132	242	4	1936	909	1.8
2017	5785	173	4	1383	1076	1.5

Assuming density growth of %40/yr
Bit ratio of length/width stays constant
Maintain >1TB/s bandwidth by increasing # of heads/surface
What architecture will enable system cost to reduce ~20%/yr to < $20K by 2017?

NV Drive Architecture includes Flash



Comparison to Multicore



Workstation

Multicore / Multiprocessor

NLSP Storage Device

NLSP Device Model Researched

- Processing elements are assigned ***separate*** DRAM and NVM partitions (Lanes)
- Communication between partitions is "messaged"
- There are two levels of communication speed analogous to within device and between device (if you put many devices in a system).

Applications Examined

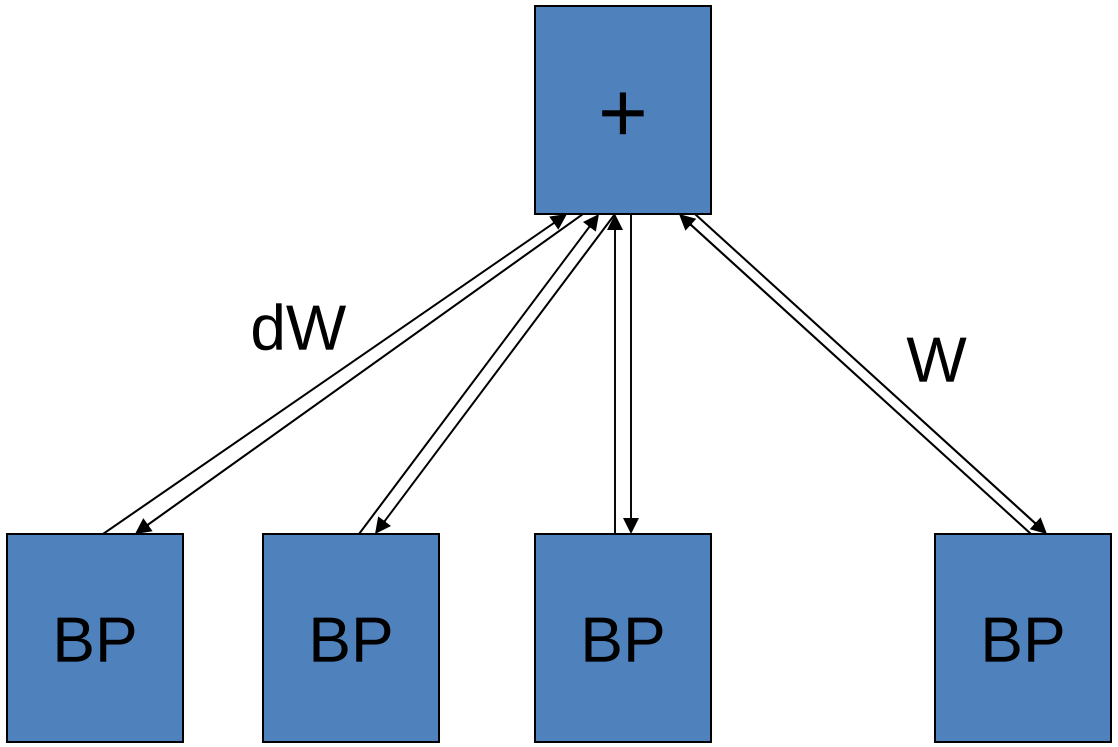
- *Backprop (BP), textbook*
- *Restricted Boltzman Machine (RBM) Deep Neural Networks, Geoff Hinton*
- *Bayesian network (BN), Judea Pearl*
- *Mitchell Co-Training (CT), also Bayesian*
- *Google Pagerank (PR), simple Pagerank*
- *Maximum Likelihood (ML), textbook*

Back Propagation and RBM Optimization

Parallelizing Small Networks

- For small network (e.g. Hinton's digit auto encoder) parallelization is easy
- Make copies of the network, run each copy on different chunks of data
- Need to merge weight updates later

In Pictures



Hinton's Network

(784-1000-500-250-30)

Necessary Storage	~11.4 MB
Weight Update Size	~11.3 MB
Ops / iteration	~8.3 MFLOPS

- 1000-1000-500-250-30 & 10M Training Examples
- On an 8 core, 2.8GHz machine:
  - Can do ~100 iterations/s
  - **40 Days**
- **Theoretically** on 1 FPGA:
  - **50 Lanes with** 5GB, 5GB/s RAM, 1 GB/s Interlane
  - Assuming 73 GFLOP/s
  - Can do 6400 iterations/s
  - **<16 Hours**
  - Above 50-100 Lanes, Performance Degrades due to interlane speeds

- Number of FPGAs (lanes) limited only by weight update overhead
  - For Hinton example, can handle up to 20 lanes assuming ~250 MB/s transfer between lanes
  - Additionally, those lanes could store 400-1000 GB of training data (10^9 example digits)
  - Given 20 lanes @ 10000 iter/s, should be able to process ~200 MB/s of data

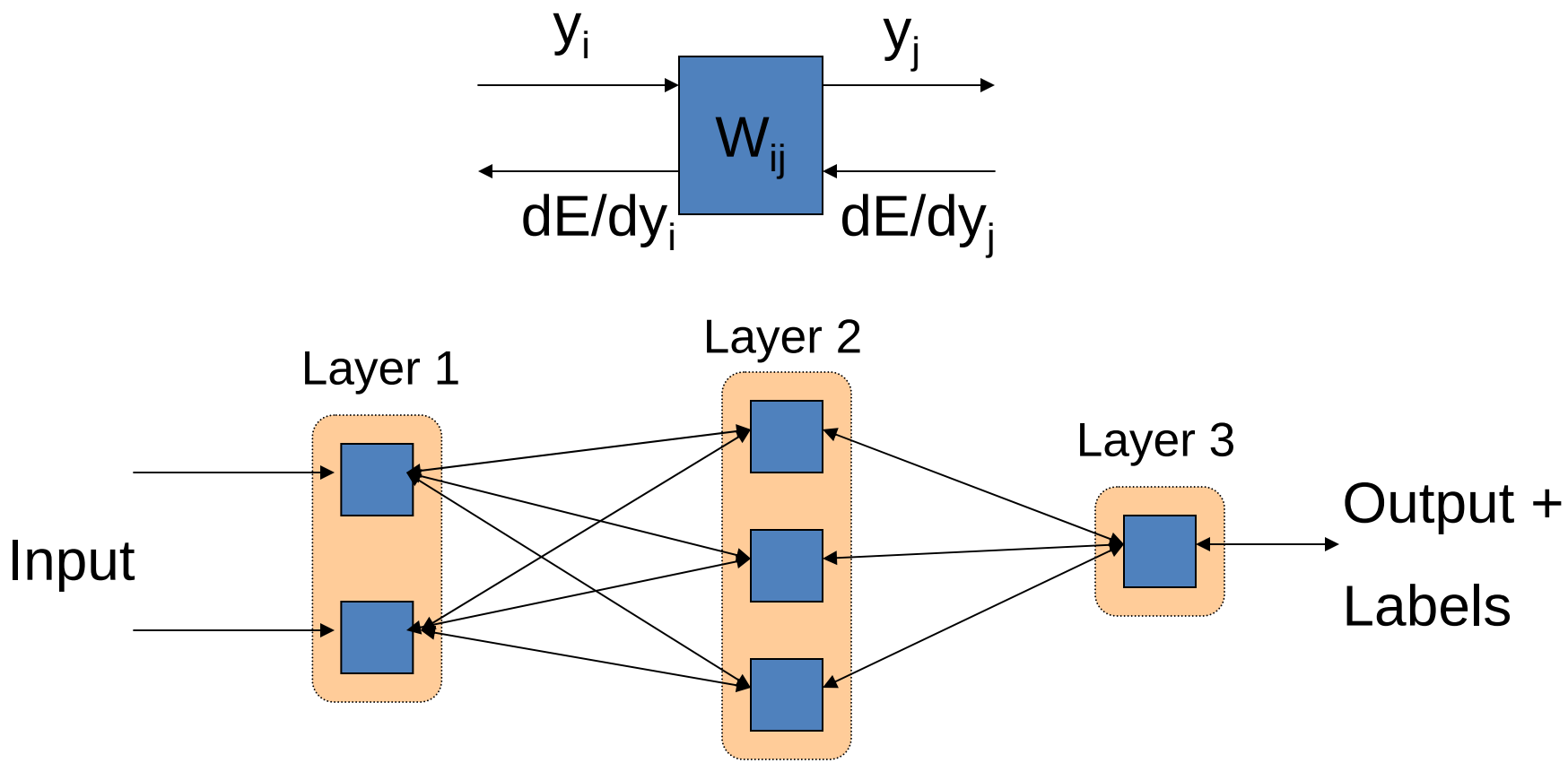
Scaling Up The Network

- What if we need a larger network?
- Example: scale up Hinton's network to 100000-100000-50000-25000-1000, or factor of 100
- Can no longer store whole network on one lane (~130 GB weight data)
- Need to parallelize network itself

Parallelizing Large Networks

- First parallelize by layer, then parallelize within layers by only considering chunks of units
- Each lane will be responsible for computing weight updates for a subset of weights
  - Will update the weights between a subset of nodes at layer j and all nodes in layer below, layer i

In Pictures


y_i
y_j
W_{ij}
dE/dy_i
dE/dy_j
Layer 1
Layer 2
Layer 3
Input
Output +
Labels

Large Network

(100000-100000-50000-25000-1000, splits 8-6-4-1)

Necessary Storage	~10 GB
Network Transfer (Aggregate)	~1 MB (13.2 MB)
Ops / iteration	~6.2 GFLOPS

Splitting Further

- 10000-10000-5000-2500-300 and 1M Training
- Improvements still >100 lanes
- 200 Lanes with 5GB, 5GB/s, 1GB/s inter, 291 GFLOP/s, **9.3 Days**

Basic Results

- Some NLSP problems lend themselves well to performance optimization by NLSP devices (BP, RBM, BN, CT, PR), others do not (ugly one is ML when requiring matrix inversion)
- There seem to be four optimization strategies
  - The network and data are small enough to fit in a single processor so don't bother with NLSP Devices
  - The network is small but the data is large, perform batch learning on whole network in parallel, and merge later
  - The network is large and the data is small or large, distribute the network and pass changes (e.g., weights), and, if data is distributed, merge later
  - There are cases (e.g., ML) which would not be improved because the PU-DRAM-NLM partitioning doesn't help
- For good NLSP problems, even with reduced processing power (FPGA), the modeling suggests the DRAM wall is the limiting factor to performance optimization

NLSP Design Fits

Scale up to Large number of Small networks….

Geoff Hinton approach with small RBMs and BP

A Connectionist Model of Sentence Comprehension and Production

Ph.D. Thesis : Douglas L. T. Rohde

http://tedlab.mit.edu/~dr/Thesis/

Multimodal, Multifaceted, over many distinct generalization networks

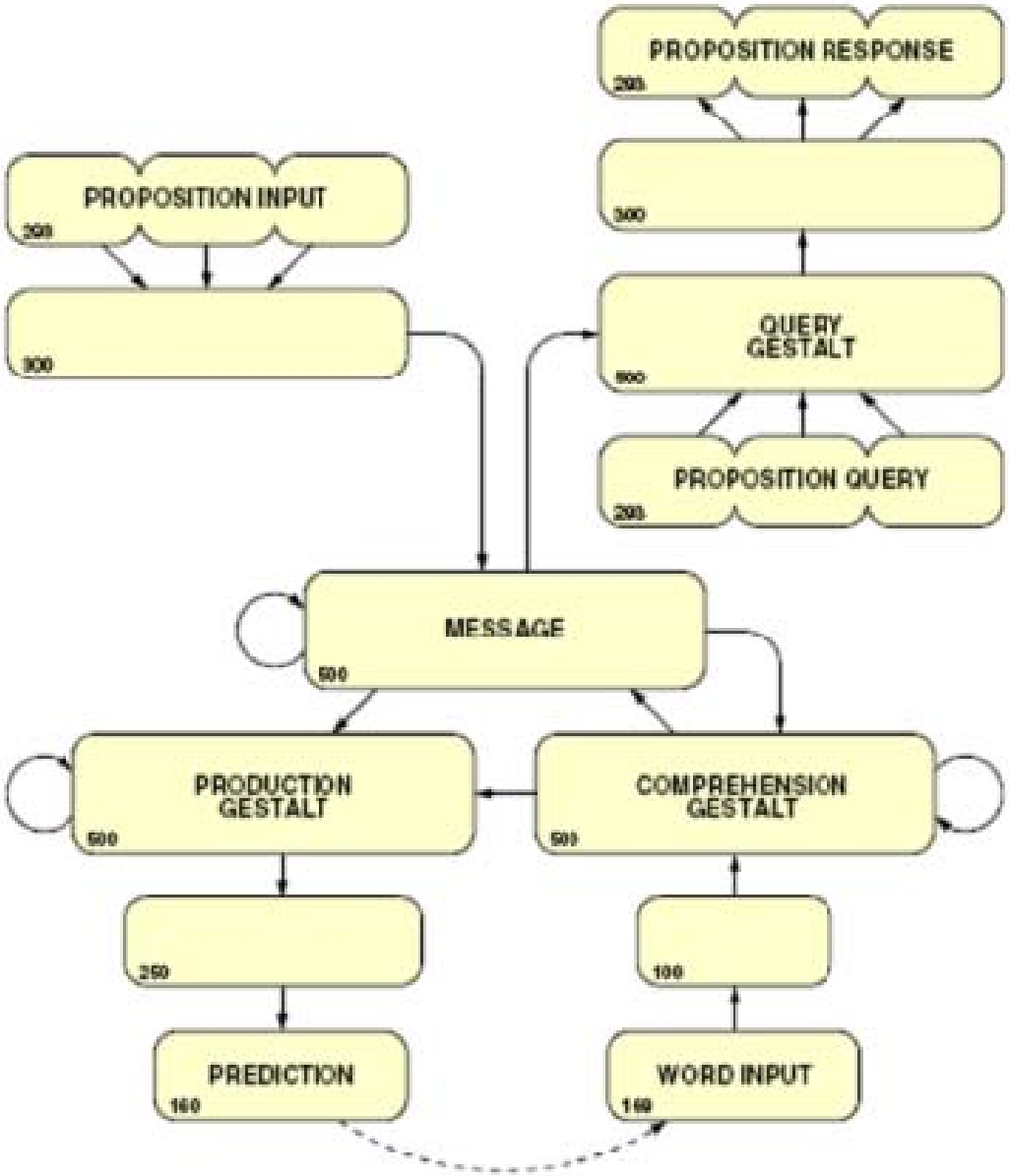


Figure 6.1: The full CSCP model.

From Rohde