Scalable NoC-based architecture of neural coding for new efficient associative memories

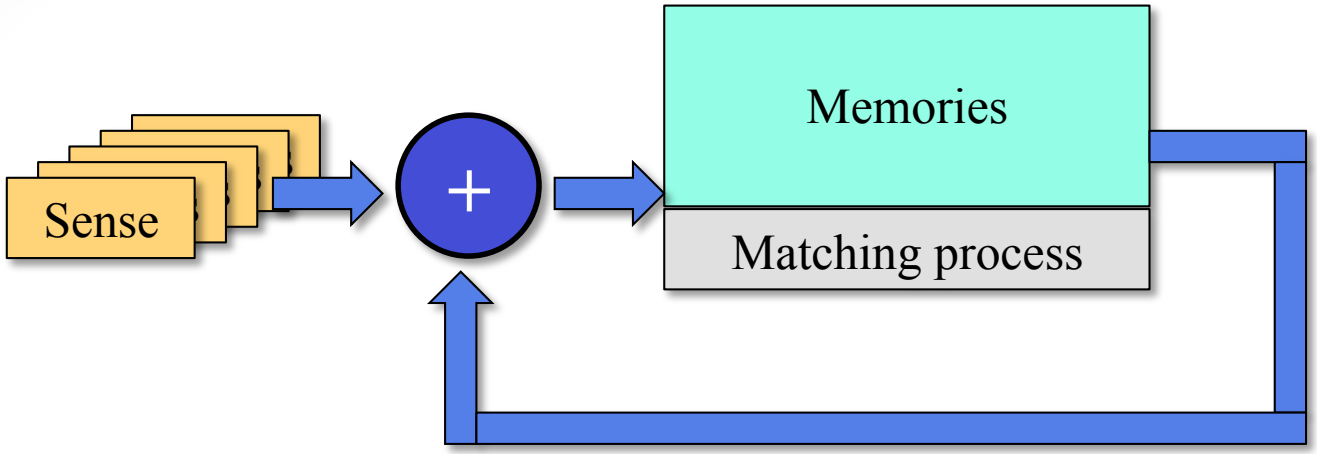
Jean-Philippe DIGUET, Martha J. SEPULVEDA
Lab-STICC, CNRS / Université de Bretagne de Sud
CODES-ISSS'13, Montréal

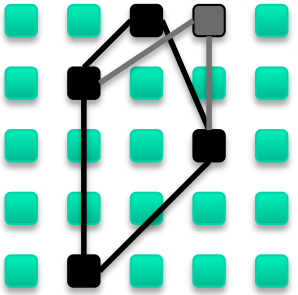
OUTLINE

1. Introduction
2. Neural Coding
3. Architecture
4. NoC Performances
5. SystemC Simulation
6. Conclusion

1.Introduction

- Early days of neurosciences: Bergson, *Le Rêve*, Conf. at International Psychology Institute, Paris, Mai 1901.
  - Consciousness and Dream are similar: a permanent adaptation mechanism to adjust (limited, perturbed) perception and memories.

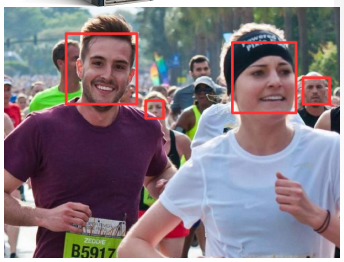




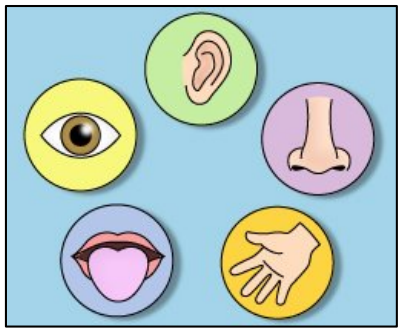
  - Partial perception enough, e.g. Writing Experience, Goldscheider & Müller 1893. *Zur Physiologie und Pathologie des Lesens,* Zeitschr. f. klinische Medicin*, 1893*.

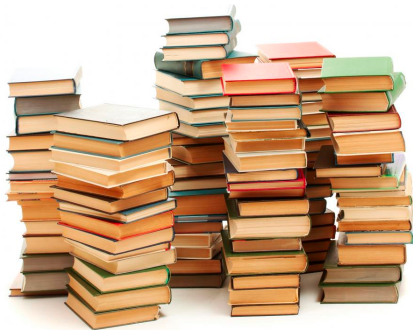
1.Introduction

- Associative Memories
  - Current Applications:
    - Telecom Routers
    - Image Processing : pattern / face /… recognition
    - Compression / Coding



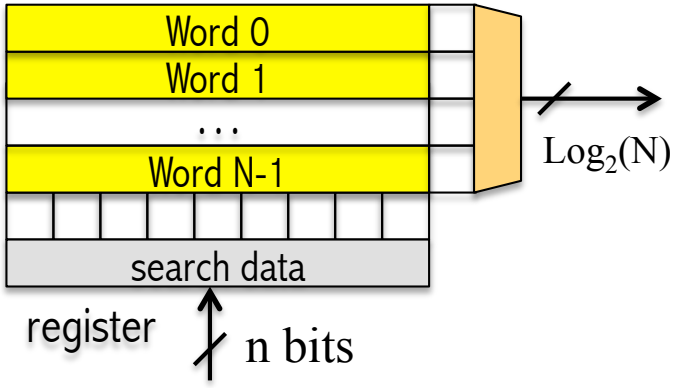
  - Internet increasing demand => Emerging needs:
    - Multi-criterion recognition
    - Cloud Database & Data mining
    - Memory Mimic = partial blur pieces





1.Introduction

- Associative Memories
  - CAM-based solutions: Important Complexity and Limited Size



  - Hopfield Neural Networks (HNN)
    - Communications identified as the main bottleneck
      - Bus based solutions + broadcast: good performance but scalability issues
      - Packet-based NoC: high bandwidth and multicast relevant property
    - Neural Coding outperforms HNN in Capacity, Efficiency and Diversity

2.Neural Coding

- Authors

- C.Berrou, V.Gripon

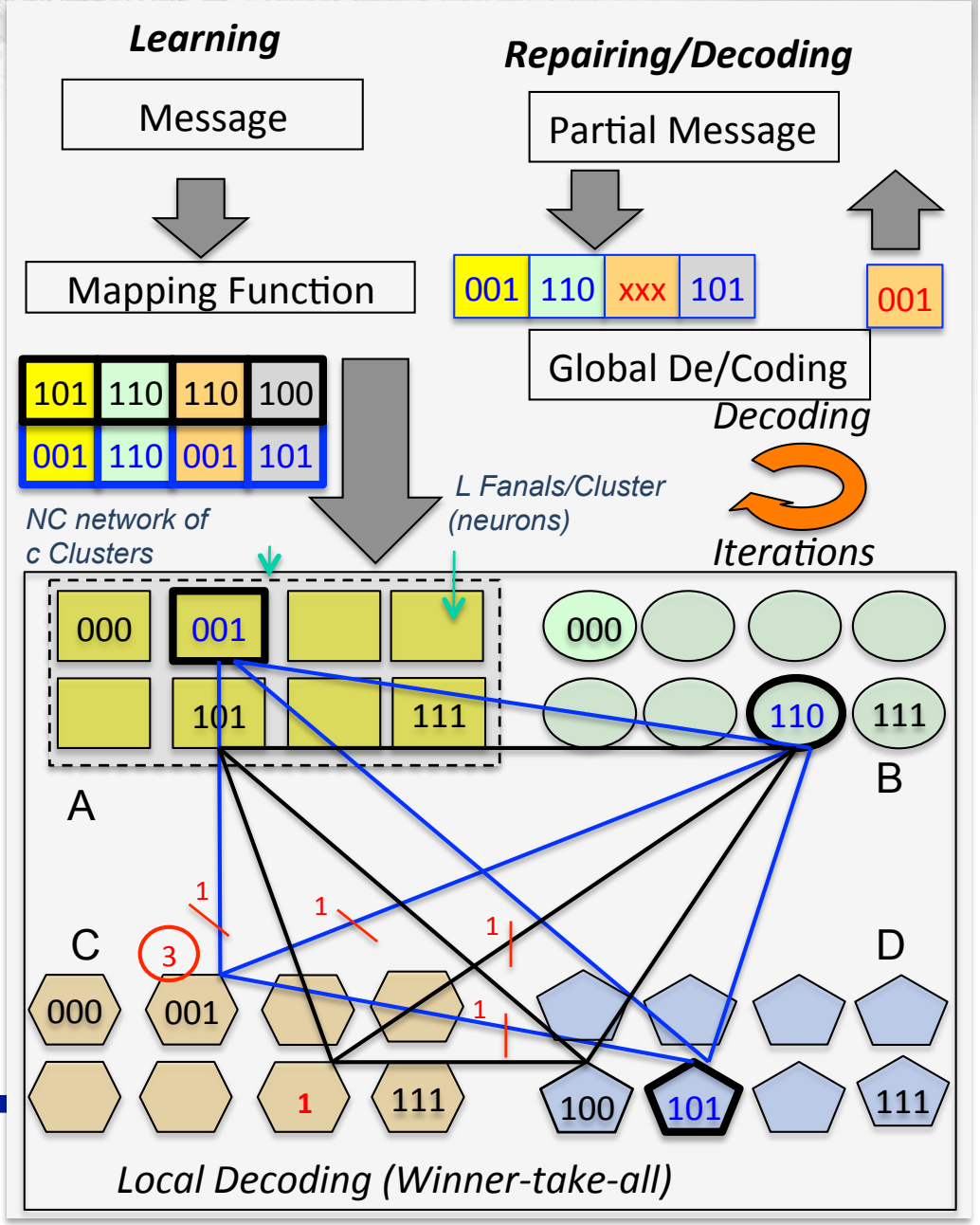
(Telecom Bretagne/Lab-STICC, Brest, France)

- Principle

- Neurons ⇔ Nodes of a graph
- Message ⇔ Clique of C Symbols
- Repairing: decoding
- Decision: Winner-Take -All

- Two ideas

- **Sparsity**: message size < Nb Neurons
- **Clique-based codewords**



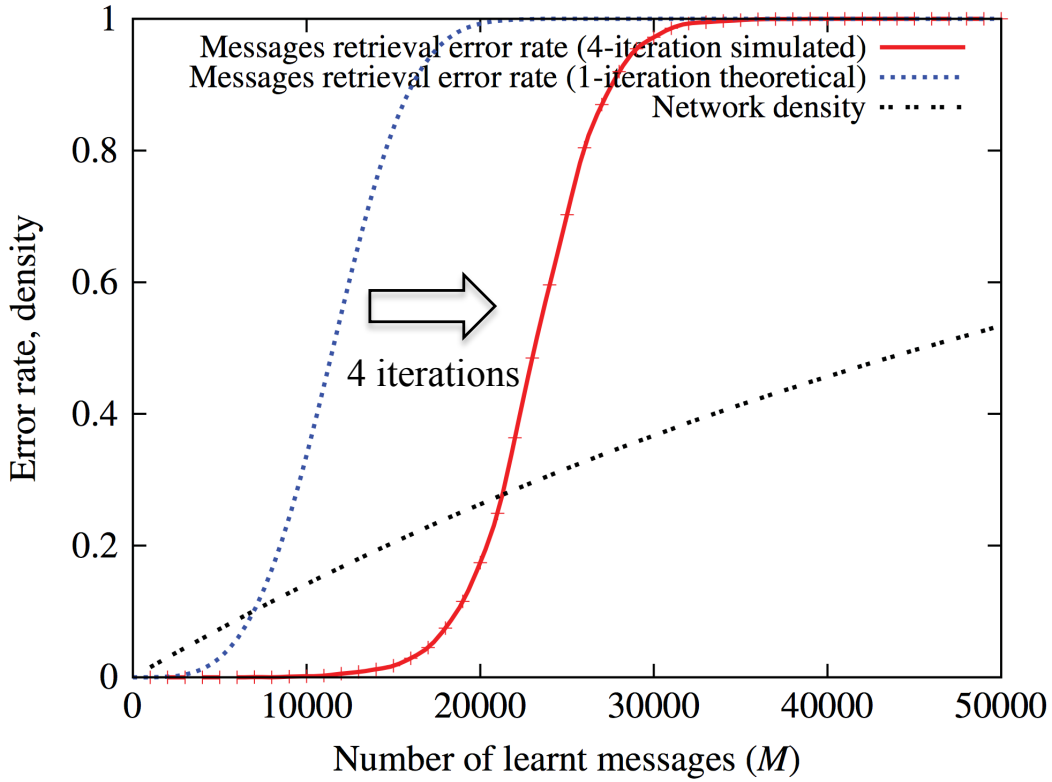
2.Neural Coding

- Properties
  - Factor of Merit F = R.d_{min}=2 > 1 so this is an error-correction code
    - Minimum distance between two codewords: d_{min}=2.(C-1)
    - Coding rate: R=1/(c-1)
  - Simple code comparable to LDPC but easy to decode (Winner-Take-All)

- Theoretical comparison with HNN for associative memories
  - *Gripon, Berrou, IEEE Trans. Neural Networks, Jul. 2012*
  - Same memory size: 1.8Mbits, C=8, L=256
  - Sparsity property:
    - **Capacity: x20**
    - **Diversity: x250**

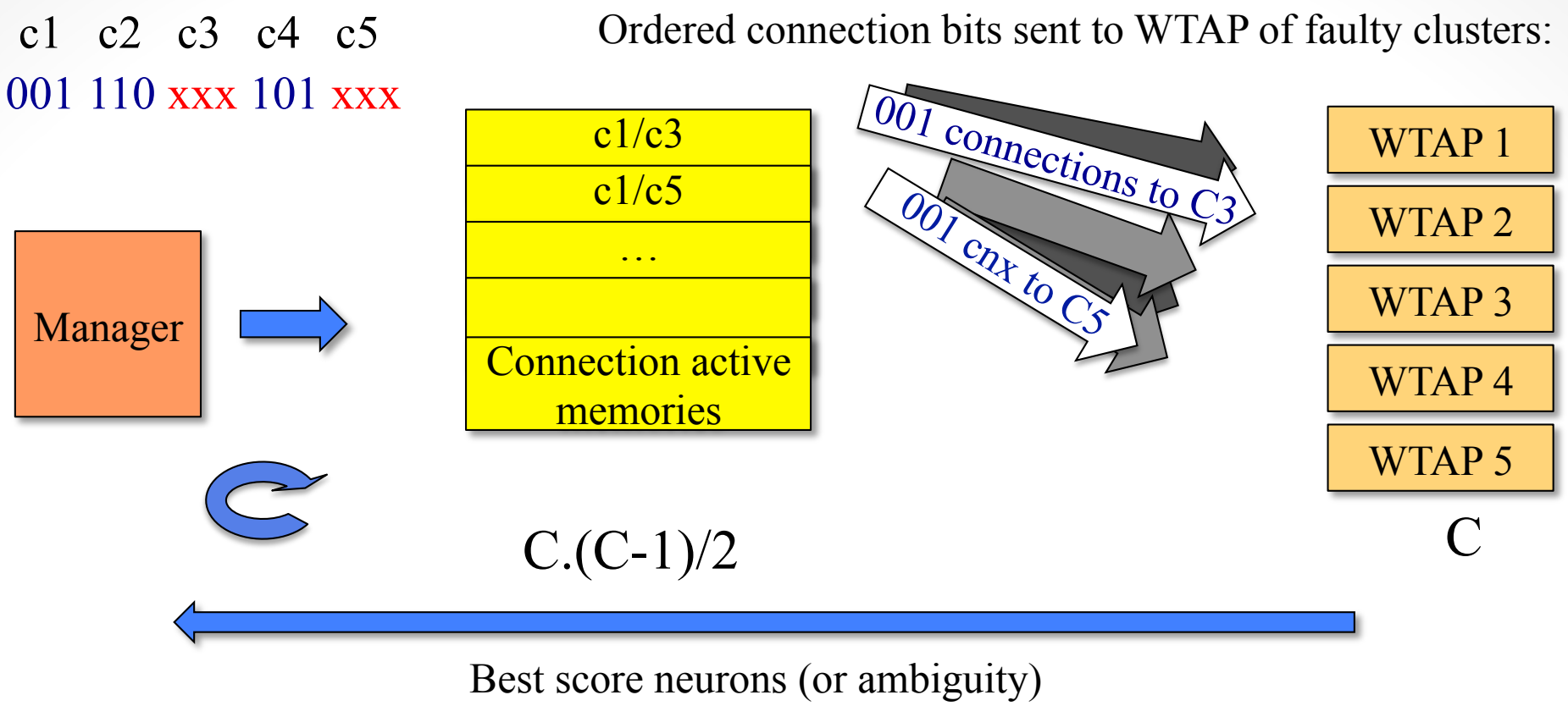
2.Neural Coding

- Error rate vs Density
  - **Density** = Number of learnt messages / Max. number of messages
  - C=8, L=256, Initial Error: 50%



3.Architecture

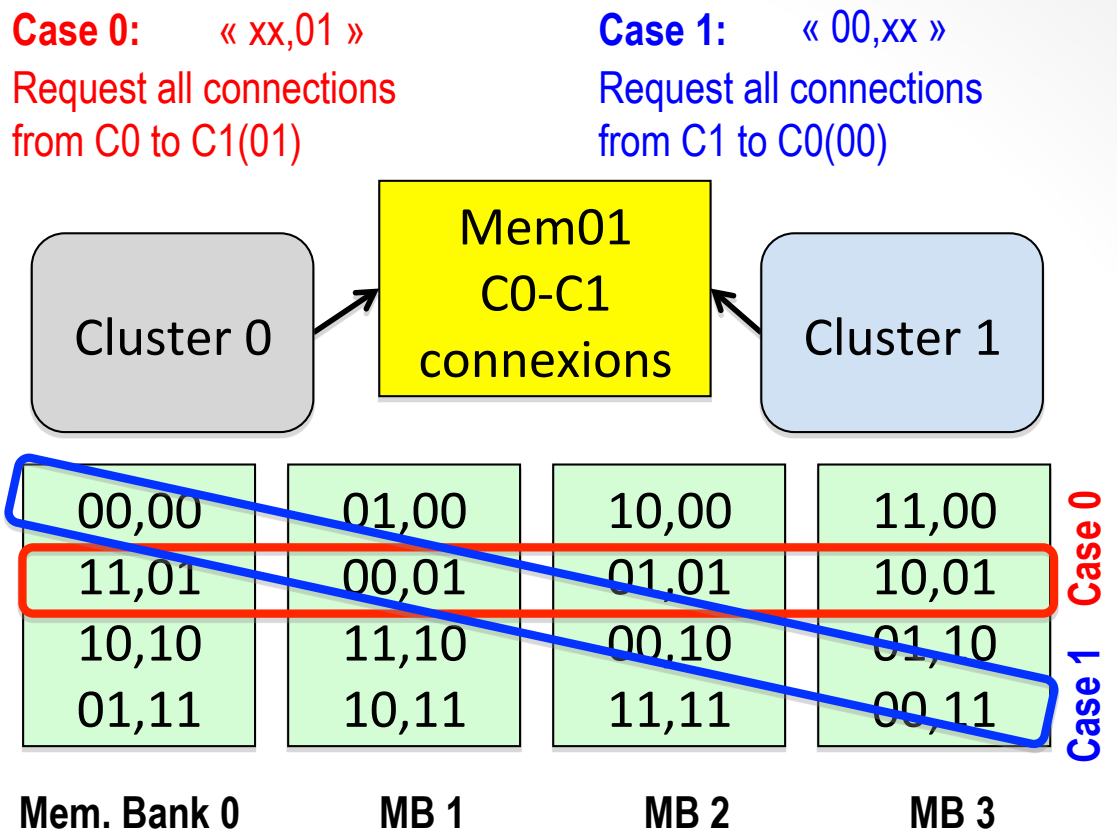
- 3 components + Communication Network
  - Active Memories, Manager, Winner-Take-All Processors



3.Architecture

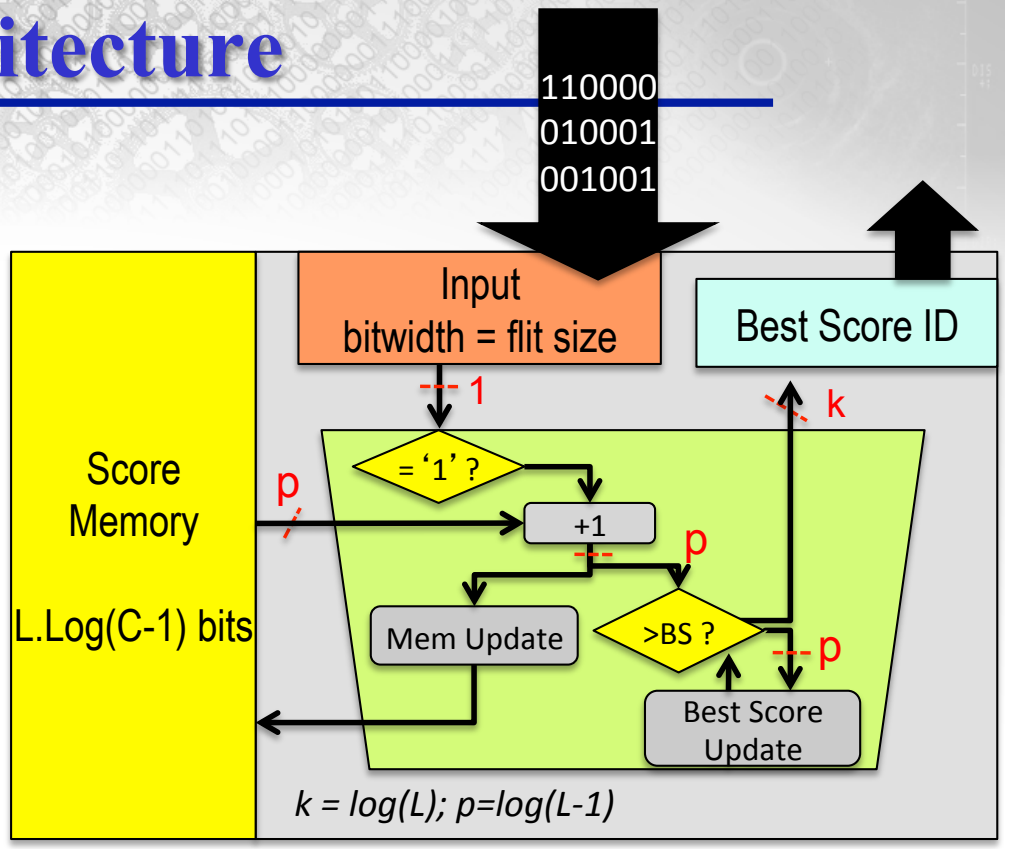
- Connection active Memory
  - Minimum bound: **1 bit / connection**
  - C.(C-1)/2 Mem. of L^2 bits
  - L banks of L bits
  - 2 possible responses
    - Full connection vector

    (ordered diagonal or line)
    - ID of '1' connection only



3.Architecture

- Winner-Take-All Processor
  - Connection vector
  - Best score sorting
  - L cycles if serial implementation
  - Alternatives for performances:
    - Send only “1”
    - Parallel sorting
- Manager
  - Simple FSM
  - Mapping function
  - Iteration control

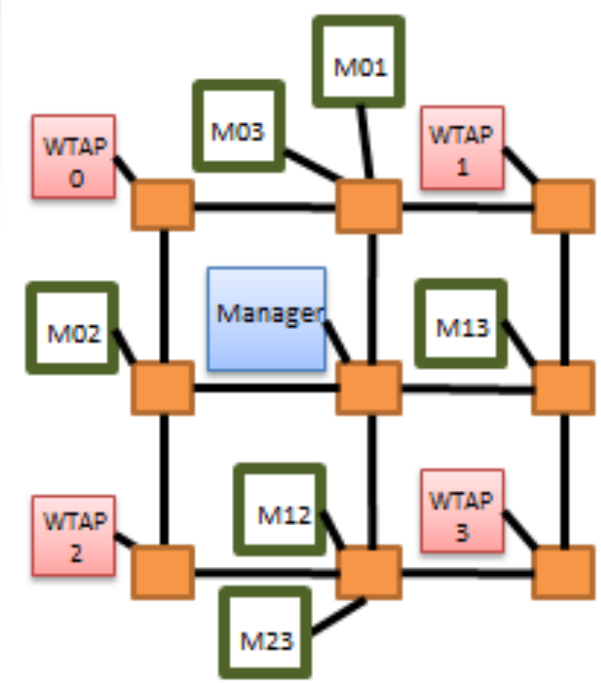


3.Architecture

- Communication
  - **Learning**: Store connection between clusters
    - Data to Manager.
    - Manager analyzes and distributes the data among the memories.
  - **Repairing**: Recovery of data
    - Stage 1: From Manager to memories
      - Which store links of clusters with unknown neurons
    - Stage 2: From memories to WTAP
      - Retrieve links
    - Stage 3: From WTAP to Manager
      - Send best neuron candidate

3.Architecture

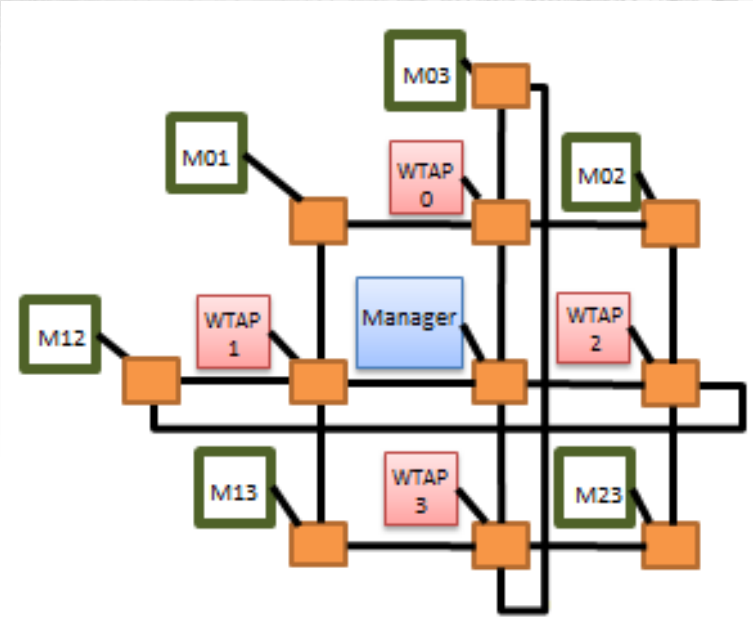
- Network On Chip
  - Four topologies



1. Hybrid interconnection (Bus + NoC)
   - **Bus**: Broadcast data (*manager-memories*)
     - Learning
     - Repairing (*stage 1*)
   - **NoC:** Parallel communication
     - Repairing (*stages 2 and 3*)

2. Regular mesh (Unicast-multicast)
   - **Unicast:** Single destination
   - **Multicast:** Multiple destination.
     - Destination extract the proper parts of message.
     - Stage 1

3.Architecture

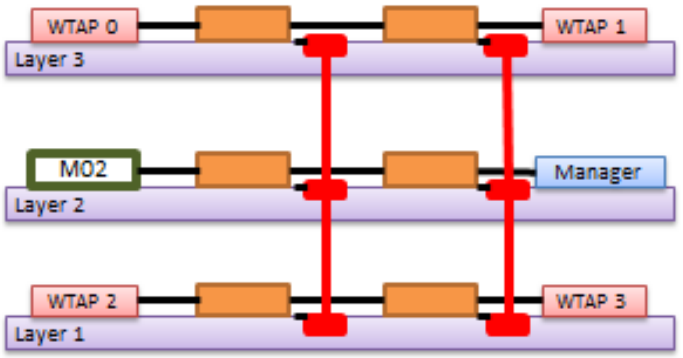


3. Semi-Torus NoC (Unicast-multicast)

- Decrease hop distance
- Memories WTAP
- XY min routing

4. NoC 3D (Bus + NoCs)

- Vertical integration
- Layered: Storage-processing
- Increase bandwidth

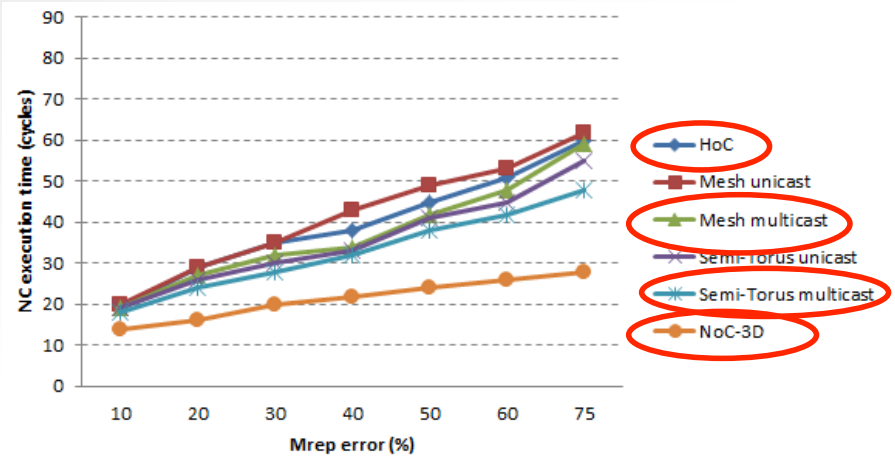


4.Performances Analysis

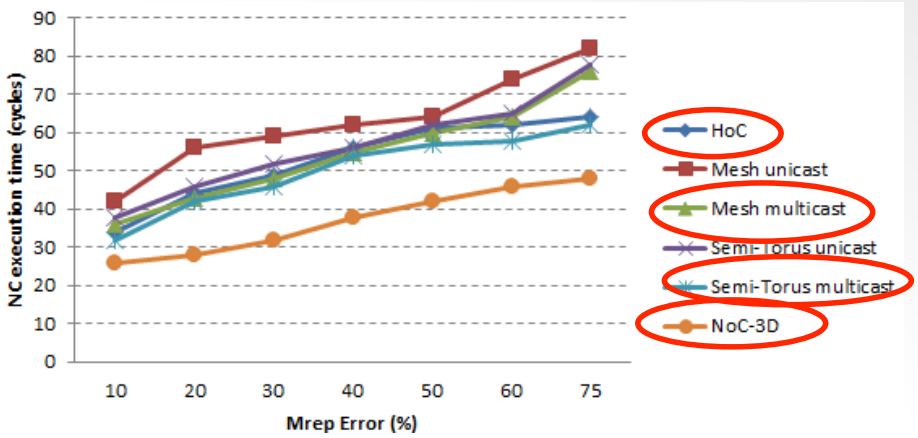
- Experimental setup
  - VHDL-RTL implemented models:
    - Case 1 : C=4 L= 4 (6 memories, 11 IPs)
    - Case 2 : C=16 L=128 (120 memories,137 IPs)
    - Components synthesized on a 65 nm Virtex 5
    - Six NoC Topologies for C=4, L=16, 32 and 64
  - Traffic simulation for 6 NoC topologies:
      - Hybrid interconnection
      - Regular mesh unicast
      - Regular mesh multicast
      - Semi-torus unicast
      - Semi-torus multicast
      - NoC 3D
    - Traffic model: 25% Learning, 75% repairing
    - Percentage of errors: 10% to 75%

4.Performances Analysis

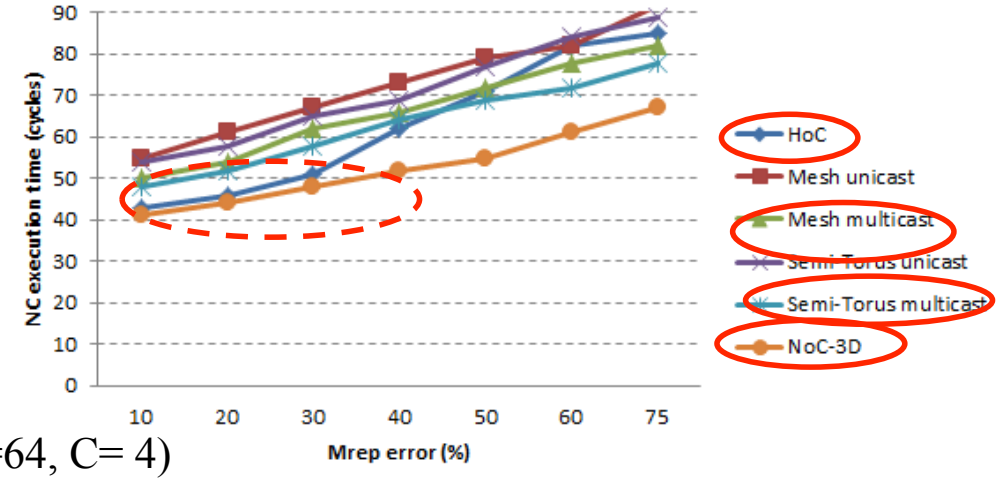
- Latency vs error percentage, 6 NoC configurations
  - R: #Neuron/#Cluster ratio



R= 4 (L=16, C=4)



R= 8 (L=32, C=4)



R= 16 (L=64, C= 4)

4.Cost: Resources

- Resources
  - 16 bits links
  - Cost : Mainly Memory: $L^2.C.(C-1)/2 + L.C.Log(C-1)$ (C_{ij}Mem / WTAP)
  - But NoC to be chosen carefully: Mesh+Multicast: good tradeoff

NC	Communication structure		
	Configuration	**FFs**	**LUTs**
{4,4,6} (11 IPs)	HoC	514	721
	Regular mesh (unicast)	452	614
	Regular mesh (multicast)	502	746
	Semi-Torus (unicast)	622	936
	Semi-Torus (multicast)	734	1021
	NoC 3D	1248	1972
{16,128,120} (137 IPs)	HoC	5672	8435
	Regular mesh (unicast)	4956	7122
	Regular mesh (multicast)	5482	8504
	Semi-Torus (unicast)	6702	10483
	Semi-Torus (multicast)	8212	11537
	NoC 3D	14222	22086

NC	Manager		WTAP		Memories	
	FFs	**LUTs**	**FFs**	**LUTs**	**FFs**	**LUTs**
{4,4,6} (11 IPs)	5	52	28	36	23	30
	-		+1 BRAM		-	
{16,128,120} (137 IPs)	12	57	44	77	16384	28563
	-		+4 BRAM			

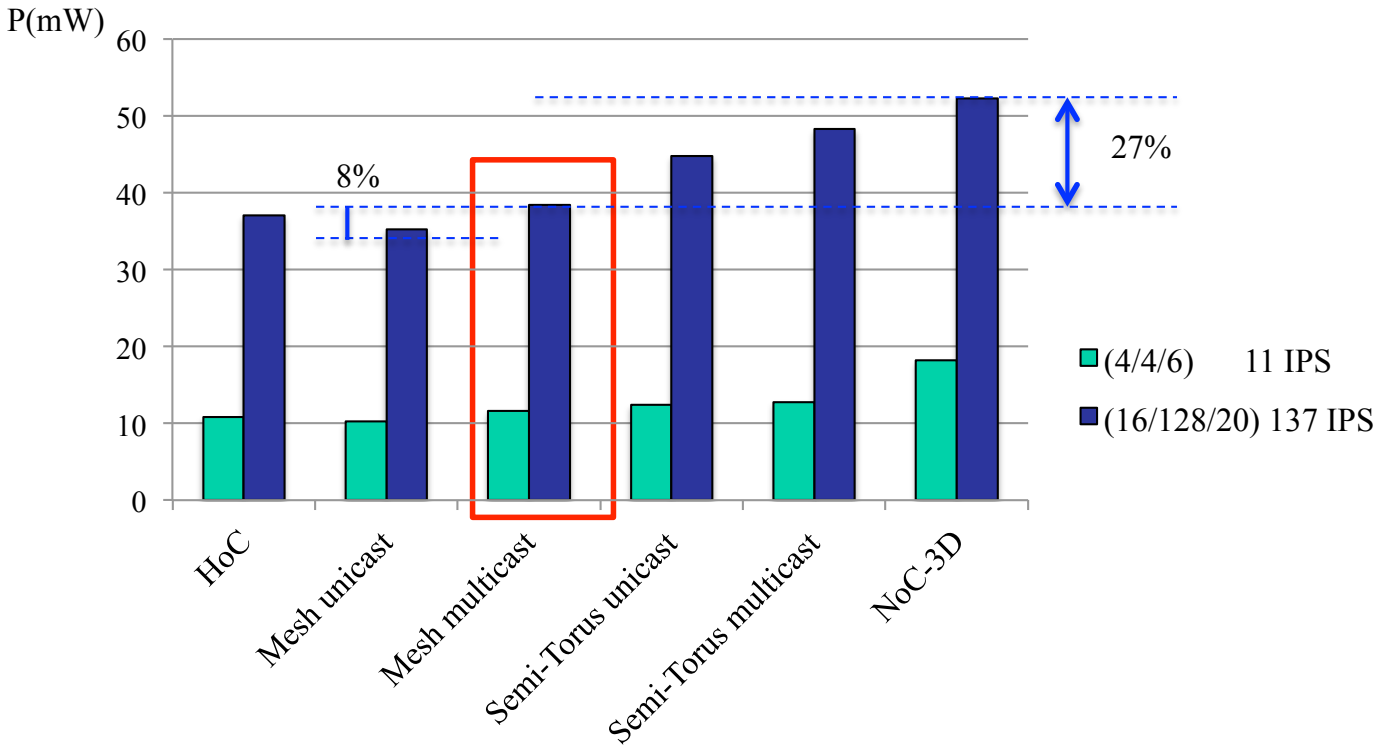
X 3.4

X 1.3

C=16, L=128, 120 Mem.

4.Cost: Power

- Power Consumption
  - 65nm 3D NoC Power model from *3D Integration for NoC-based SoC Architectures*, Springer, 2011, Heibanyrad, Pétrot, Jantsch eds.
  - Function of : #Hops, Link and Router activities, Vertical and Horizontal links

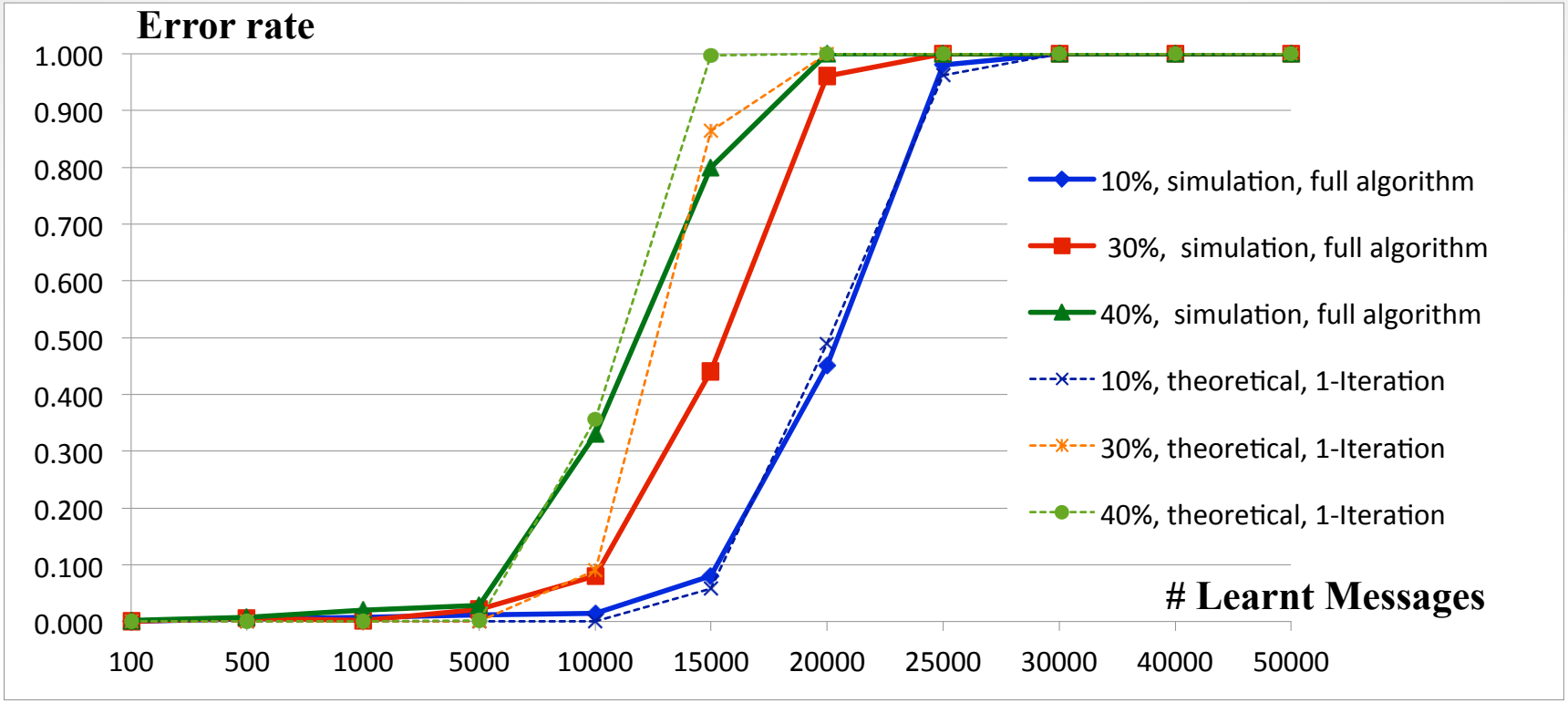


5. SystemC Simulation

- Experimental Setup
  - Memory, WTAP, Manager: TLM-Level, cycle budget from implementation
  - NoC: cycle accurate, required for routing
  - C=16 clusters, L=128 neurons
  - Messages: randomly generated with a given error ratio (e)
  - 4 points generated for each case, then average (28-120 min / point)
  - 3 experiments e =10%, 30% and 40%

5. SystemC Simulation

- Results



- 10% very close to 1-iteration theory (1 iteration enough most of the time)
- 30-40% : Gap = corrections (#iterations) & random process approximation side effects

6. Conclusion

- Promising solution: cost & performance
  - Cost vs CAM
  - Efficiency vs HNN

- Architecture
  - Successfully demonstrated
  - NoC based architectures fit with NC requirements
  - MESH + Multicast NoC = good tradeoff.
  - Open to ongoing neural coding evolutions

6. Conclusion

- Perspectives: Optimization tracks to be explored:
  - Links Bitwidth (1, 8, ..) => NoC Cost reduction
  - Fewer WTAP: dynamic allocations (reduced sorting memory cost)
  - More active memories: send only '1' (sparsity benefit, reduced WTA delay)
  - Hierarchy of NoC for multi concept associative memory