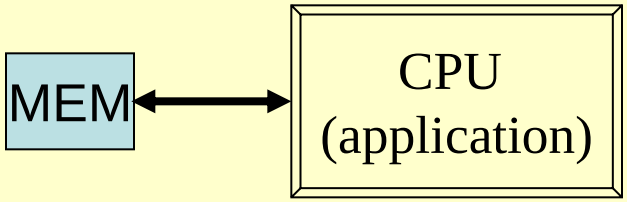
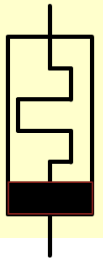




Similarity Calculations in Resistive Memories and Enabling Memory-Intensive Architectures

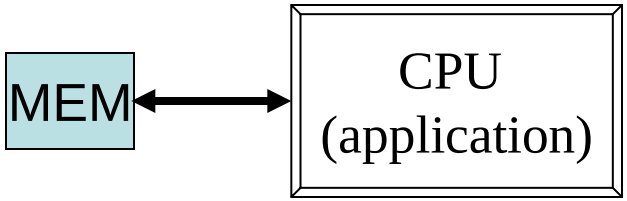
Yuval Cassuto

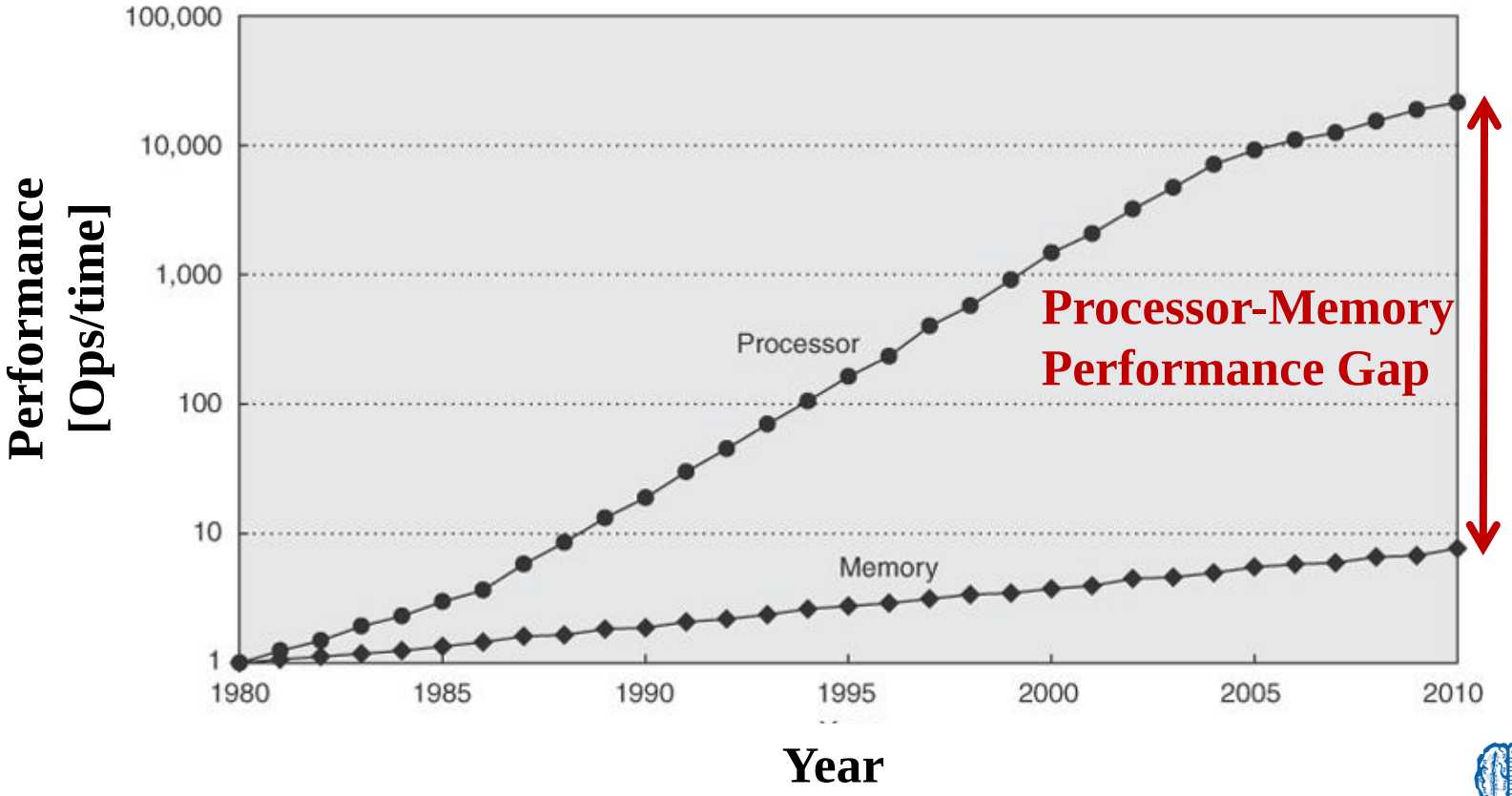
Technion – Viterbi EE Department

Intel ICRI-CI Retreat, May 24 2016

ICRI-CI Intel Collaborative Research Institute Computational Intelligence

The Memory Wall





Complexity

- The old world:

  Complexity = #CPU ops

- The new world:

  Complexity = 200*transfer + #CPU ops

  Energy: 5000*

ICRI-CI Memory-Intensive Architectures

Making memory-intensive architectures happen:

- Compute applications
- Data integrity and R/W performance
- Novel memory circuits
- Novel memory devices

The team of PIs:

Avinoam Kolodny, Technion EE

Eby Friedman, U of Rochester

Shahar Kvatinsky, Technion EE

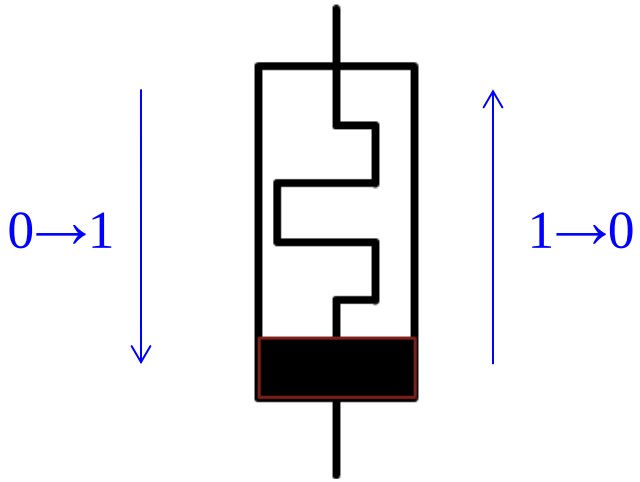
Yuval Cassuto, Technion EE

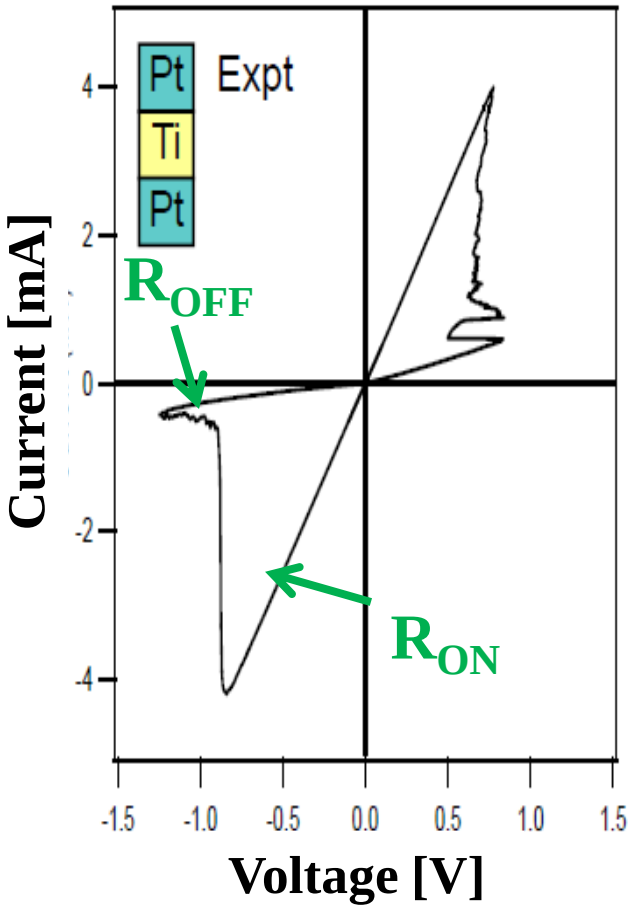
ICRI-CI
Intel Collaborative Research Institute
Computational Intelligence
intel

The Star of the Show

Resistive memory

- Memristor [2008 HP Labs]
- RRAM [industry wide]





What can Resistive Memories Do?

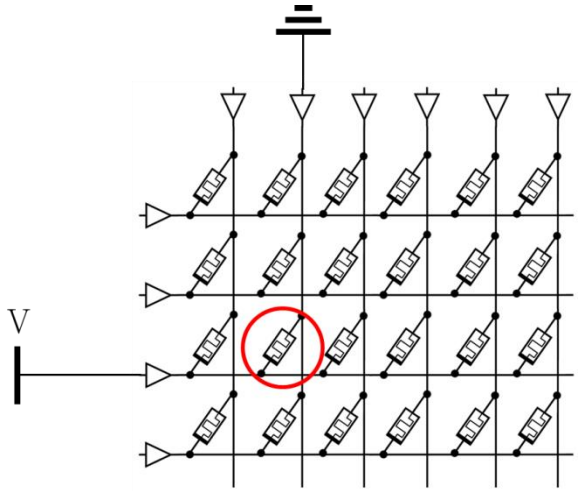
Shahar Kvatinsky:

- Logic within memory (MPU)
- Integrated memristor-CMOS logic
- Multithreading with multistate registers (CFMT)
- Hardware for neural networks
- Coding and circuit design for memristors
- Resistive Processing in Memory (ReAP, ReGP-SIMD)

ICRI-CI
Intel Collaborative Research Institute
Computational Intelligence
intel

The Primal Challenge

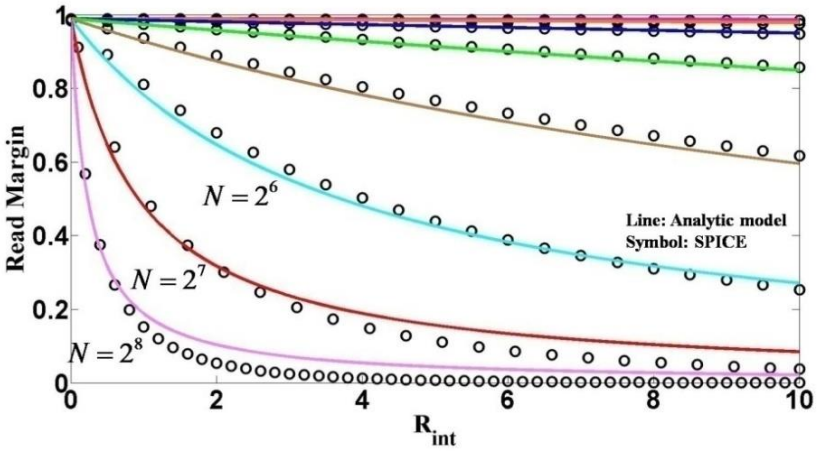
Crossbar array



- Read margins
- Sneak paths
- Non-additive noise
- Write disturbs
- Interconnect resistance
- …

A Mix of Approaches

- Device/circuit theory [A. Ciprut and E. G. Friedman]
  - Analytical crossbar model



- Information/detection theory [YC et. al]
  - Encode the bits to reduce errors
  - Optimal bit decisions given measurements

Mem. Arch. for Machine Learning

??

In-Memory Computing

ICRI-CI
Intel Collaborative Research Institute
Computational Intelligence
intel

Mem. Arch. for Machine Learning

Sparse (binary) vector

In-Memory Computing

ICRI-CI
Intel Collaborative Research Institute
Computational Intelligence
intel

Hamming Similarity

joint work with K. Crammer, in ISIT 2015 Semi-Plenary Session

feature vectors

$\longrightarrow$ 0100110110 10

$\longrightarrow$ 000011011011

$d_H = 2$: similar

$\longrightarrow$ 110101011001

$d_H = 5$: dissimilar

ICRI-CI Intel Collaborative Research Institute Computational Intelligence

Efficient Hamming Similarity

Recall that:

Cost is the transfer, not the computation

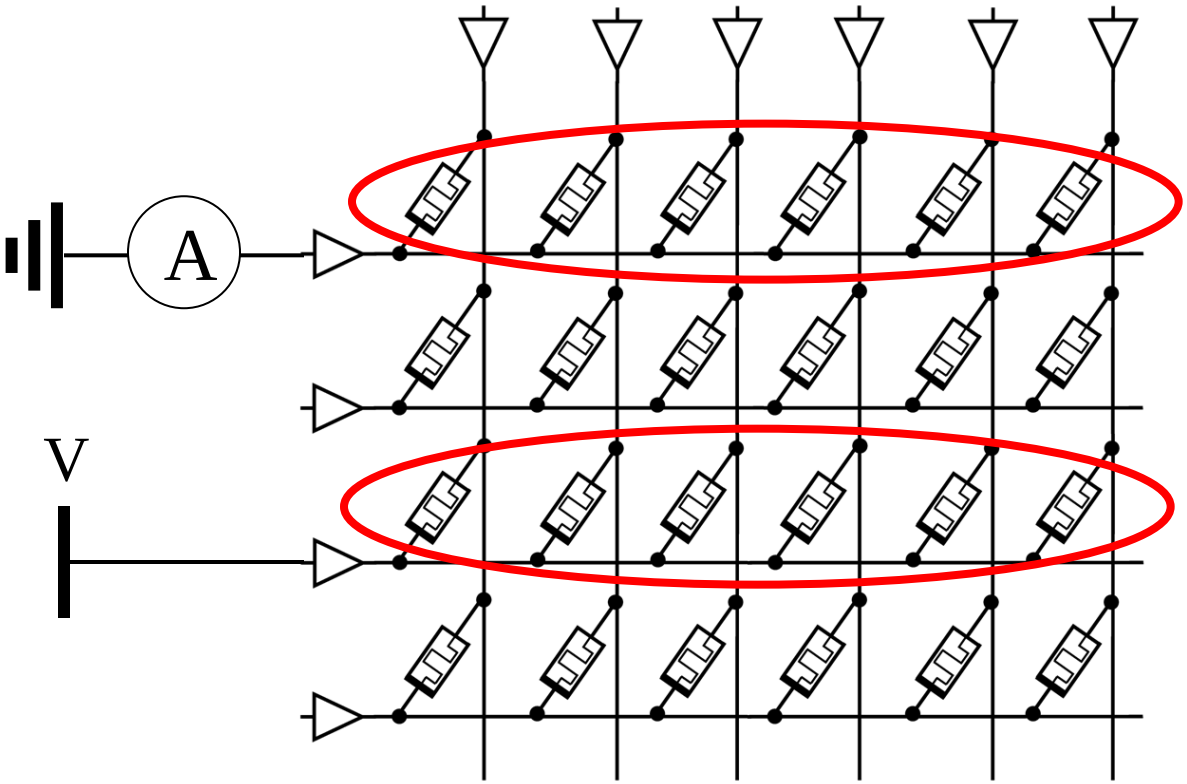
Question:

Can we compute without moving the data?

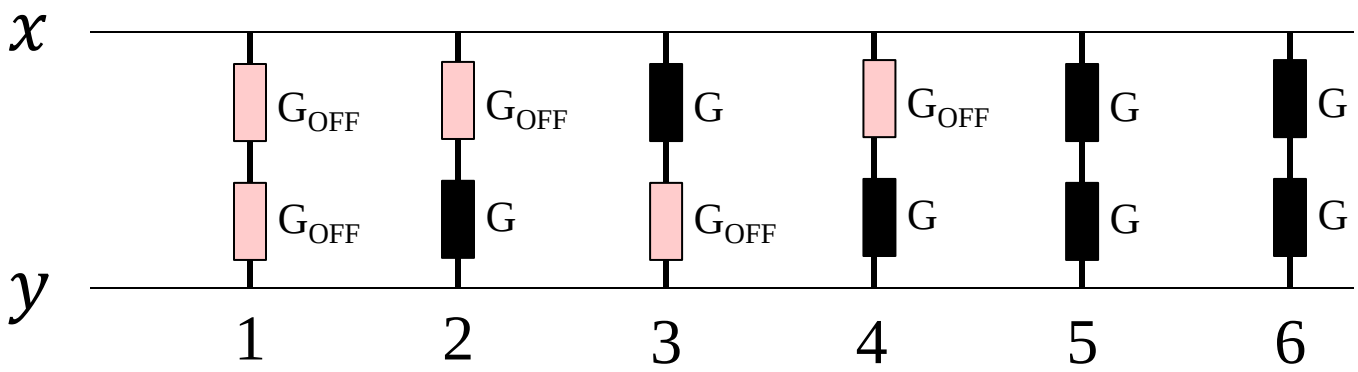
ICRI-CI
Intel Collaborative Research Institute
Computational Intelligence
intel

Crossbar for Similarity

Similarity



Vector-Pair Measurement



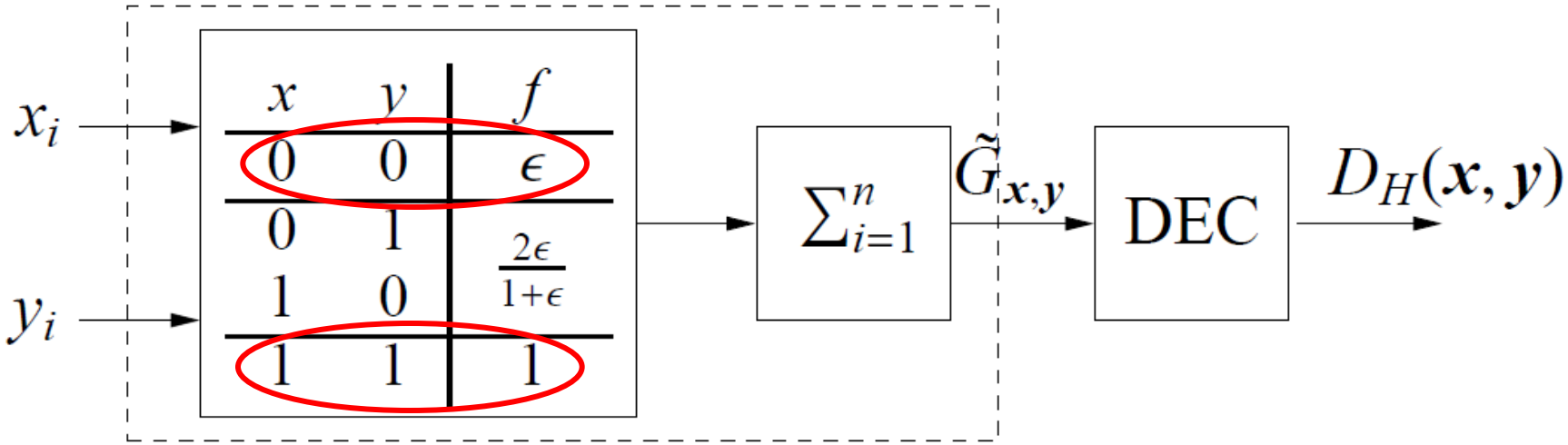
$$G_{OFF} = \epsilon G$$

$$G_{x,y} = \sum_{i=1}^{n} x_i y_i \frac{G}{2} + x_i(1-y_i)\frac{\epsilon G}{1+\epsilon} + (1-x_i)y_i\frac{\epsilon G}{1+\epsilon} + (1-x_i)(1-y_i)\frac{\epsilon G}{2}$$

Measurement Post-Processing

Computing objective:

Compute the Hamming distance from the resistance measurement



Case 1: Known Weights

Explicit decoding function:

Given $W_H(x), W_H(y)$, for any $\epsilon < 1$

$$D_H(\boldsymbol{x}, \boldsymbol{y}) = \frac{1+\epsilon}{(1-\epsilon)^2}\left[(1-\epsilon)(W_H(\boldsymbol{x}) + W_H(\boldsymbol{y})) + 2n\epsilon - 2\tilde{G}_{\boldsymbol{x},\boldsymbol{y}}\right]$$

Case 1: Known Weights

Explicit decoding function:

Given $W_H(x), W_H(y)$, for any $\epsilon < 1$

$$D_H(\boldsymbol{x}, \boldsymbol{y}) = \frac{1+\epsilon}{(1-\epsilon)^2}\left[(1-\epsilon)(W_H(\boldsymbol{x}) + W_H(\boldsymbol{y})) + 2n\epsilon - 2\tilde{G}_{\boldsymbol{x},\boldsymbol{y}}\right]$$

Need 3 measurements!

ICRI-CI
Intel Collaborative Research Institute
Computational Intelligence

Case 2: Unknown Weights

Non-explicit decoding function:

For $0 < \epsilon < 1/(n-1)$

$$D_H(\boldsymbol{x}, \boldsymbol{y}) = \frac{\tilde{G}_{\boldsymbol{x},\boldsymbol{y}} - N - \epsilon(n-N)}{\epsilon \frac{1-\epsilon}{1+\epsilon}}$$

Where N is the unique integer that satisfies $0 \leqslant D_H(\boldsymbol{x}, \boldsymbol{y}) \leqslant n$

Case 2: Unknown Weights

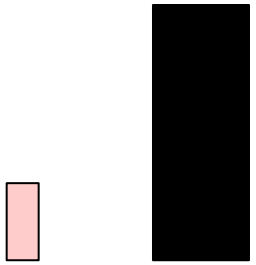
Non-explicit decoding function:

For $0 < \epsilon < 1/(n-1)$

$$D_H(\boldsymbol{x},\boldsymbol{y}) = \frac{\tilde{G}_{\boldsymbol{x},\boldsymbol{y}} - N - \epsilon(n-N)}{\epsilon\frac{1-\epsilon}{1+\epsilon}}$$

Where N is the unique integer that satisfies $0 \leqslant D_H(\boldsymbol{x},\boldsymbol{y}) \leqslant n$

Vector length n small given ϵ



G_{OFF} G_{ON}

ICRI-CI Intel Collaborative Research Institute Computational Intelligence

Allowing Longer Vectors

Can we achieve

for any $0 < \epsilon < 1/const$

?

Easy: Doubling Construction

$c(x) =$ | x | $\sim x$ |

$c(y) =$ | y | $\sim y$ |

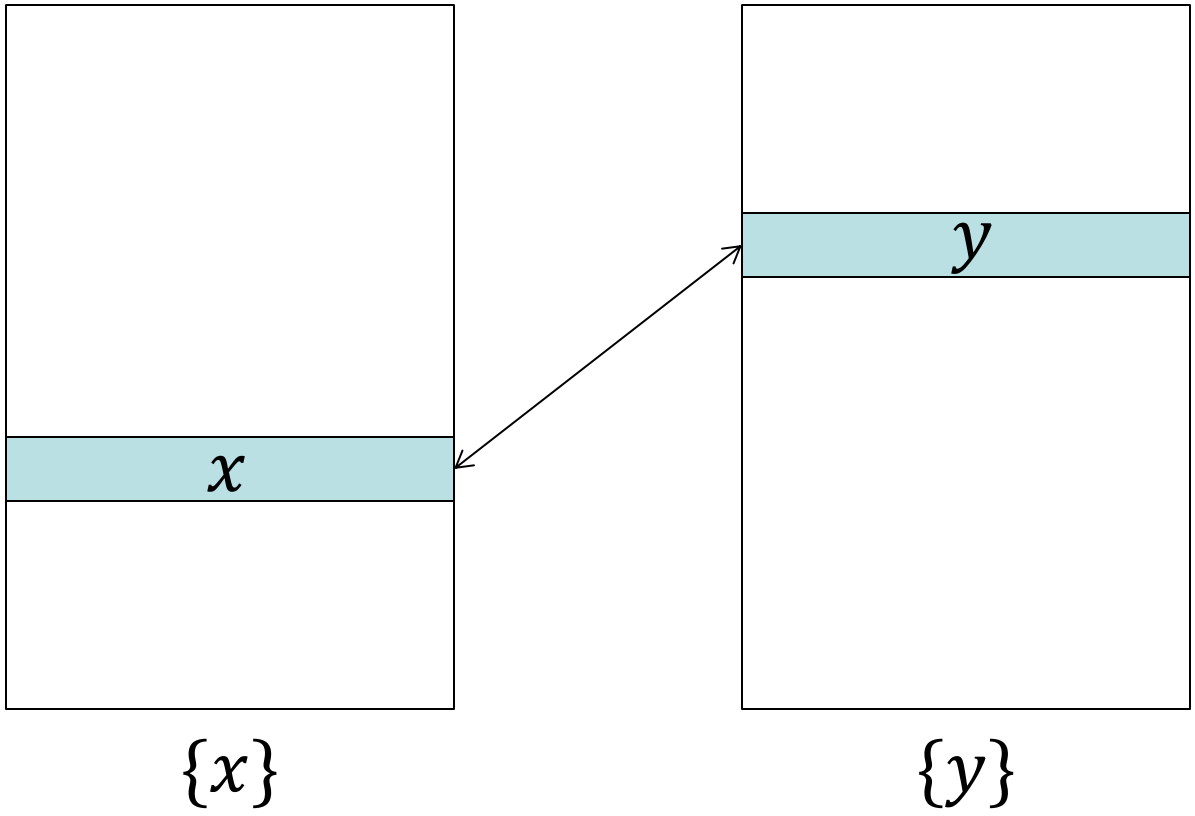
1. $W_H(c(x)) = W_H(c(y)) = n \Rightarrow D_H(c(x), c(y))$
2. $D_H(x, y) = D_H(c(x), c(y))/2$

for any $0 < \epsilon < 1$

Redundancy n bits

Can We Do Better?

Yes, for bi-partite similarity.



$$\underline{w} \leq W_H(x), W_H(y) \leq \underline{w} + \Delta w = \overline{w}$$

Bi-Orthogonal Construction

Extension vectors:

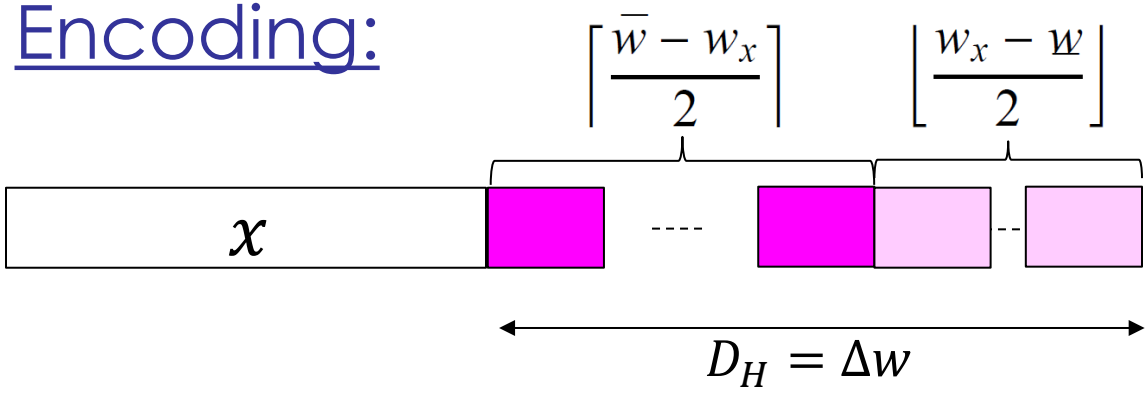
$X_1 =$ **0 0 0 1**

$X_3 =$ **1 1 1 0**

$Y_1 =$ **1 0 0 0**

$Y_3 =$ **0 1 1 1**

Encoding:



$$W_H = \overline{w} + \frac{\Delta w}{2} + 1[w_x \text{ odd}]$$

$$W_H = \overline{w} + \frac{\Delta w}{2} + 1[w_y \text{ odd}]$$

Efficient Coded Similarity

Theorem:

If $\underline{w} \leq W_H(x), W_H(y) \leq \underline{w} + \Delta w$, then $D_H(x,y)$ can be calculated exactly for any $0 < \epsilon < 1/3$

with redundancy $2\Delta w$ bits

Conclusion

- Memory intensive architectures will have high impact on future computing
- At the low level
  - Need to make them happen
  - Low level insights → influence on future architecture
- At the high level
  - Need to develop general abstractions serving many big-data applications