The V-Way Cache:
Demand-Based Associativity via Global Replacement

Moinuddin K. Qureshi David Thompson Yale N. Patt

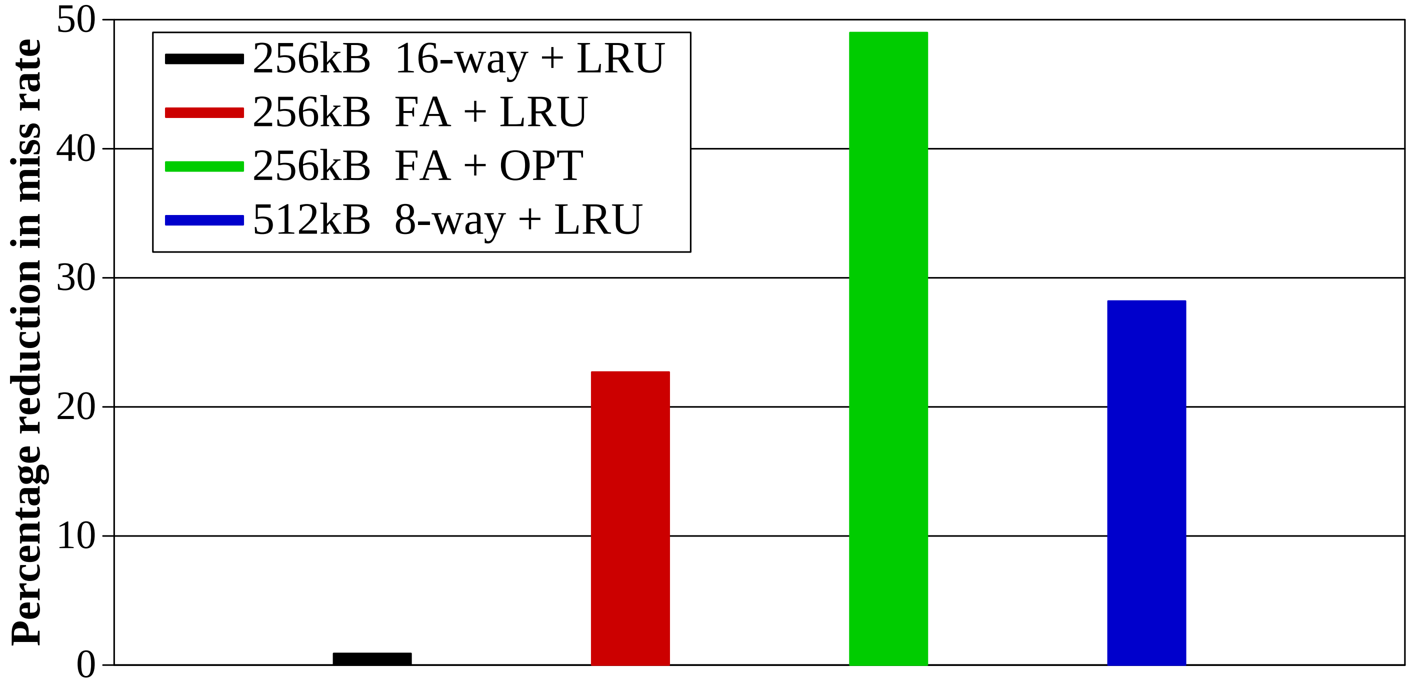
Department of Electrical and Computer Engineering,
The University of Texas at Austin.
{moin, dave, patt}@hps.utexas.edu

Introduction

- Need for efficient management of secondary caches.
- Ideal cache: fully associative with OPT replacement.

Introduction

- Need for efficient management of secondary caches.
- Ideal cache: fully associative with OPT replacement.



Fully Associative Caches: Cost v/s Benefit

- Benefits
  - Conflict miss elimination
  - Global Replacement (finds the best victim)

- Cost
  - Significant increase in the number of tag comparisons
  - Increased access latency
  - Increased power consumption

Fully Associative Caches: Cost v/s Benefit

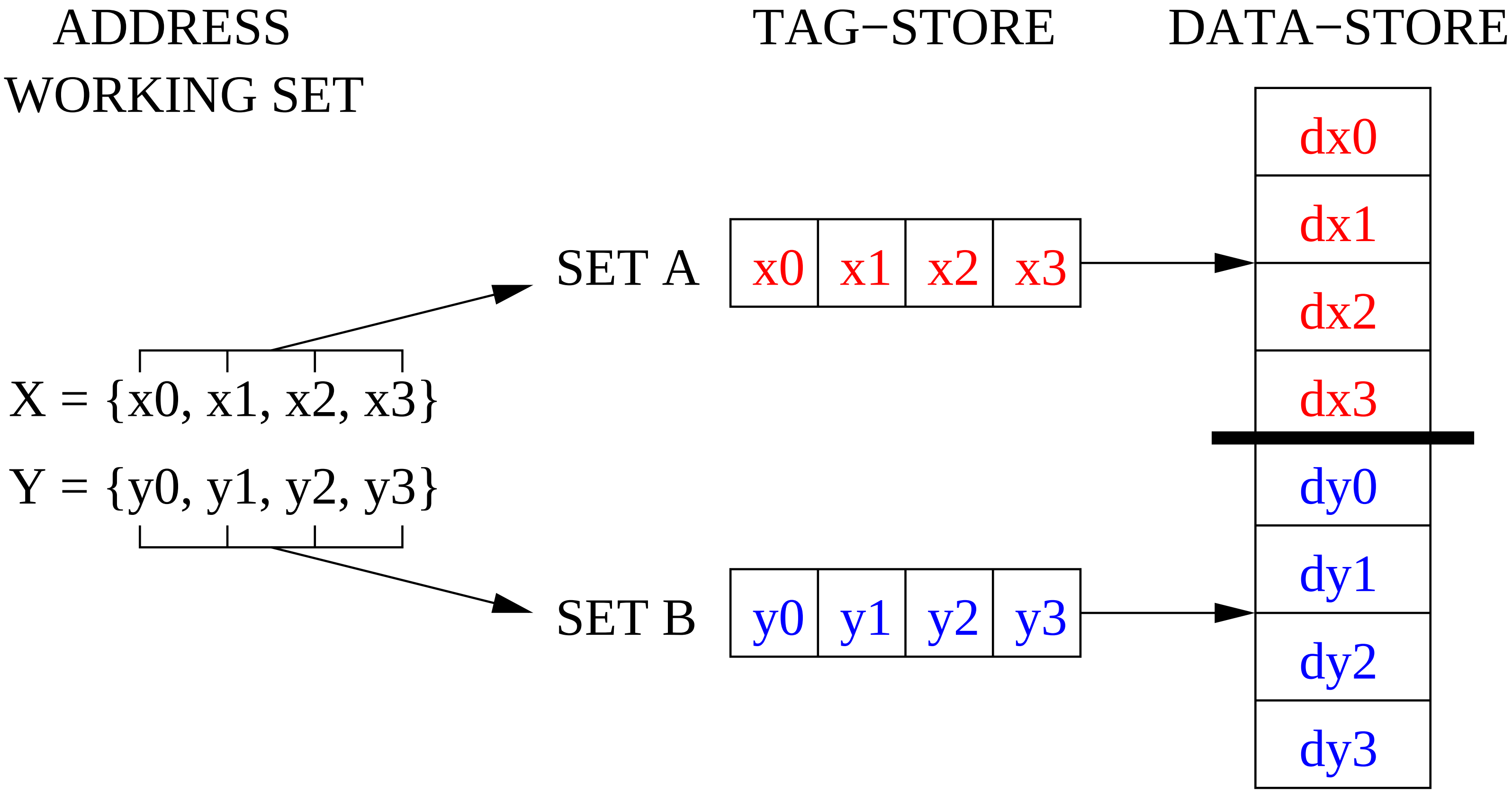
- Benefits
  - Conflict miss elimination
  - Global Replacement (finds the best victim)
- Cost
  - Significant increase in the number of tag comparisons
  - Increased access latency
  - Increased power consumption

Can we get the benefits of a fully associative cache without paying the cost?

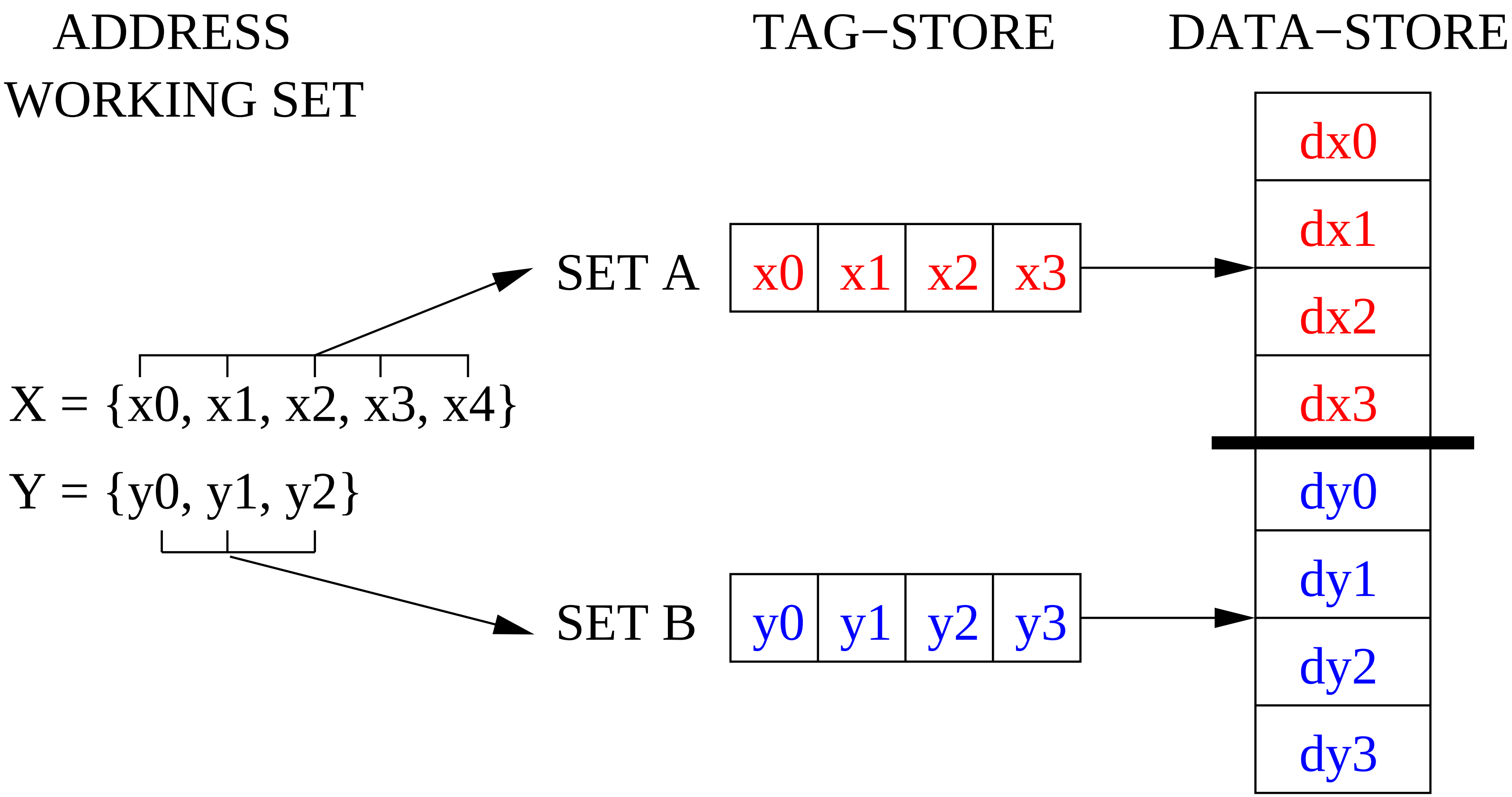
Outline

- Introduction
- Example of Local and Global Replacement
- The V-Way Cache
- Evaluation
- Related Work and Conclusion

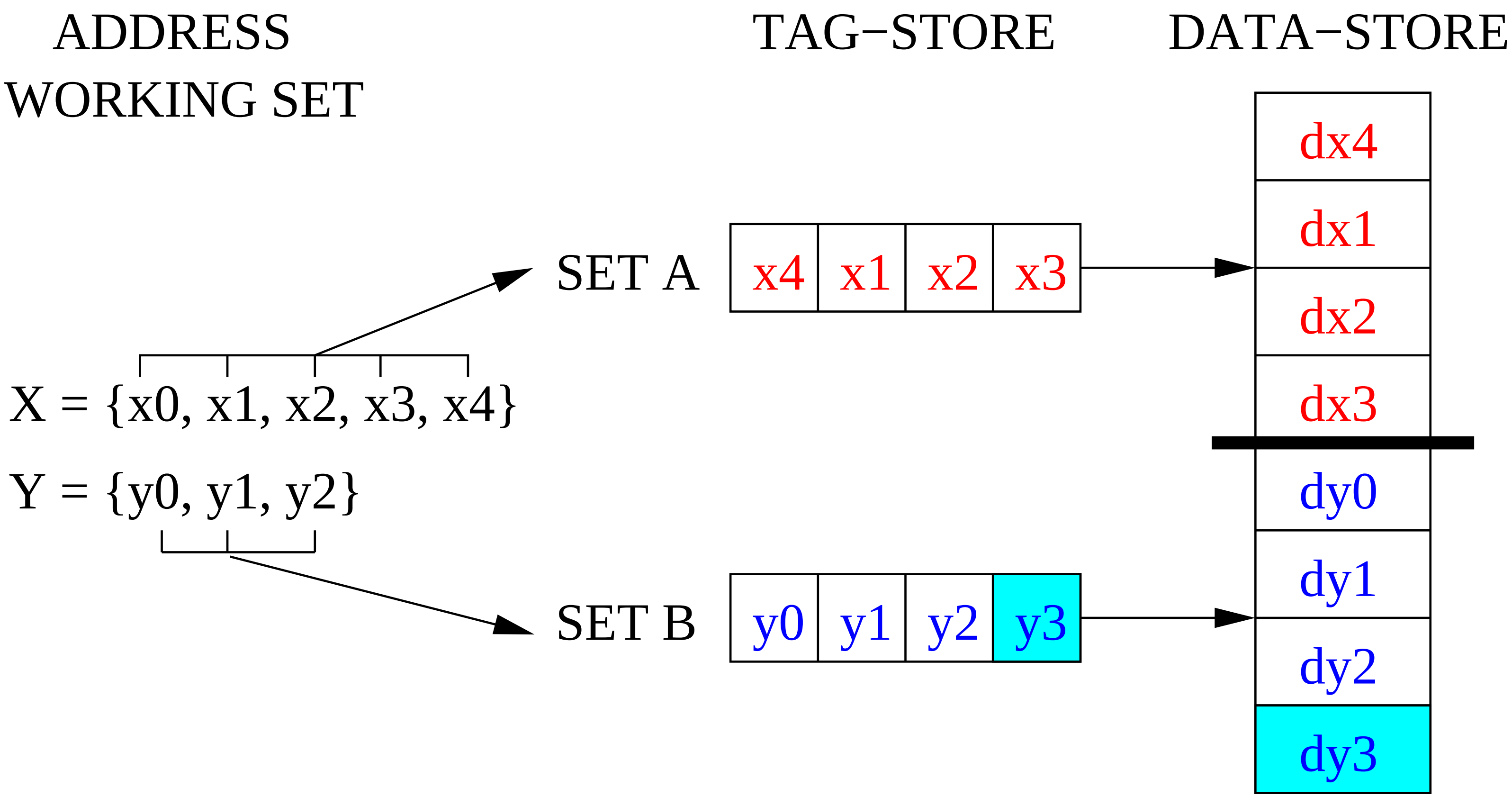
Example of Local Replacement



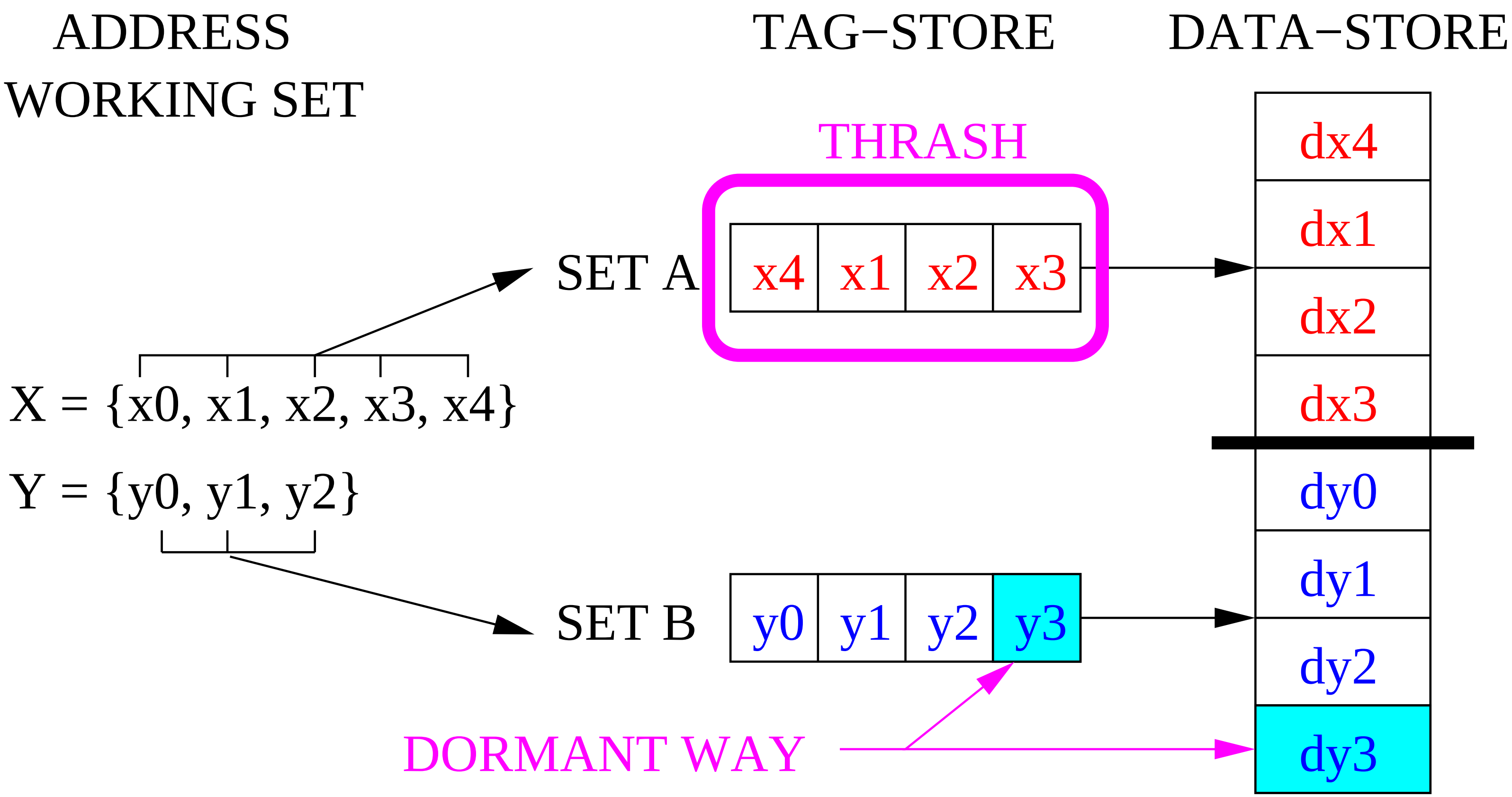
Example of Local Replacement



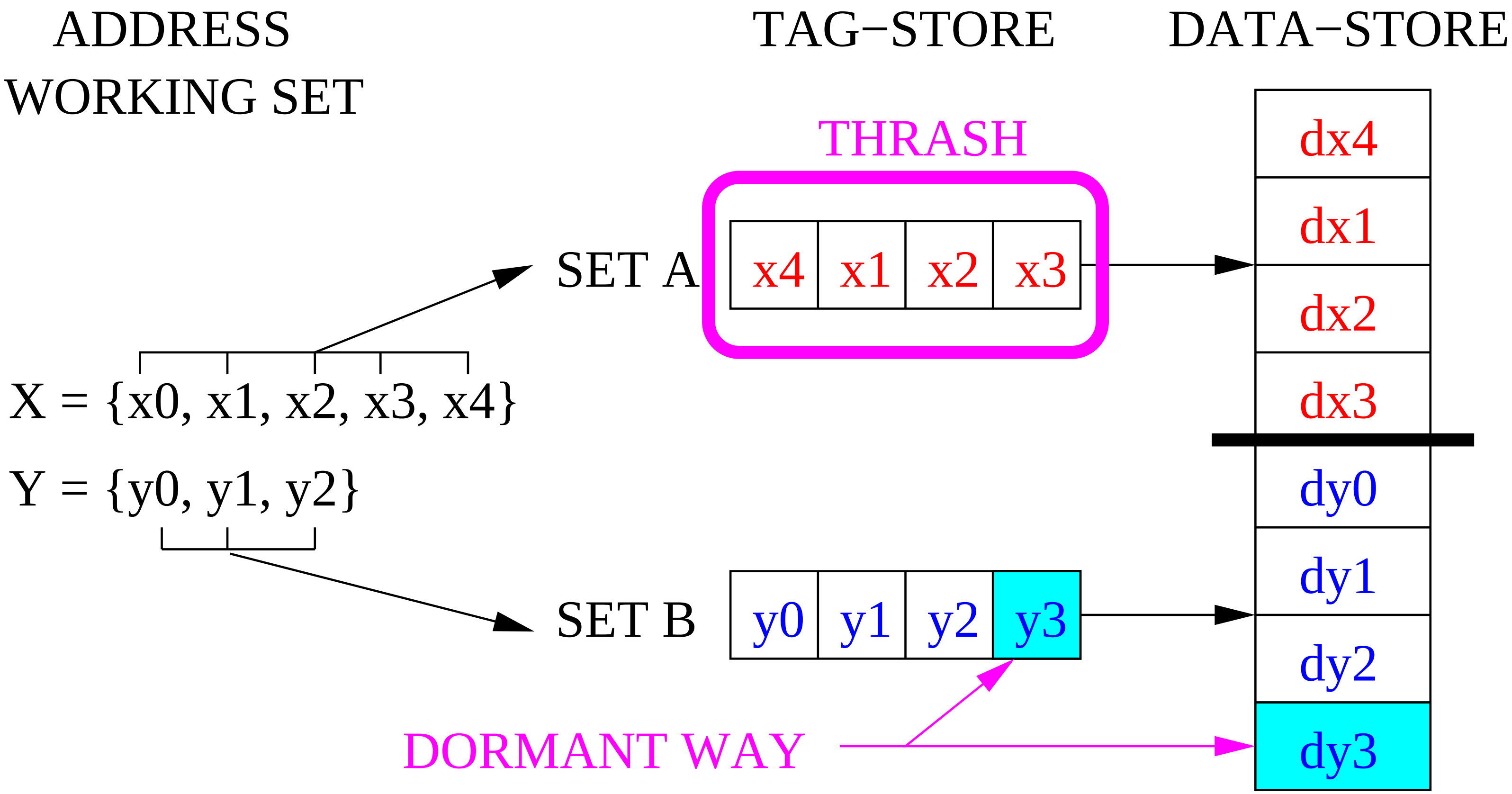
Example of Local Replacement



Example of Local Replacement

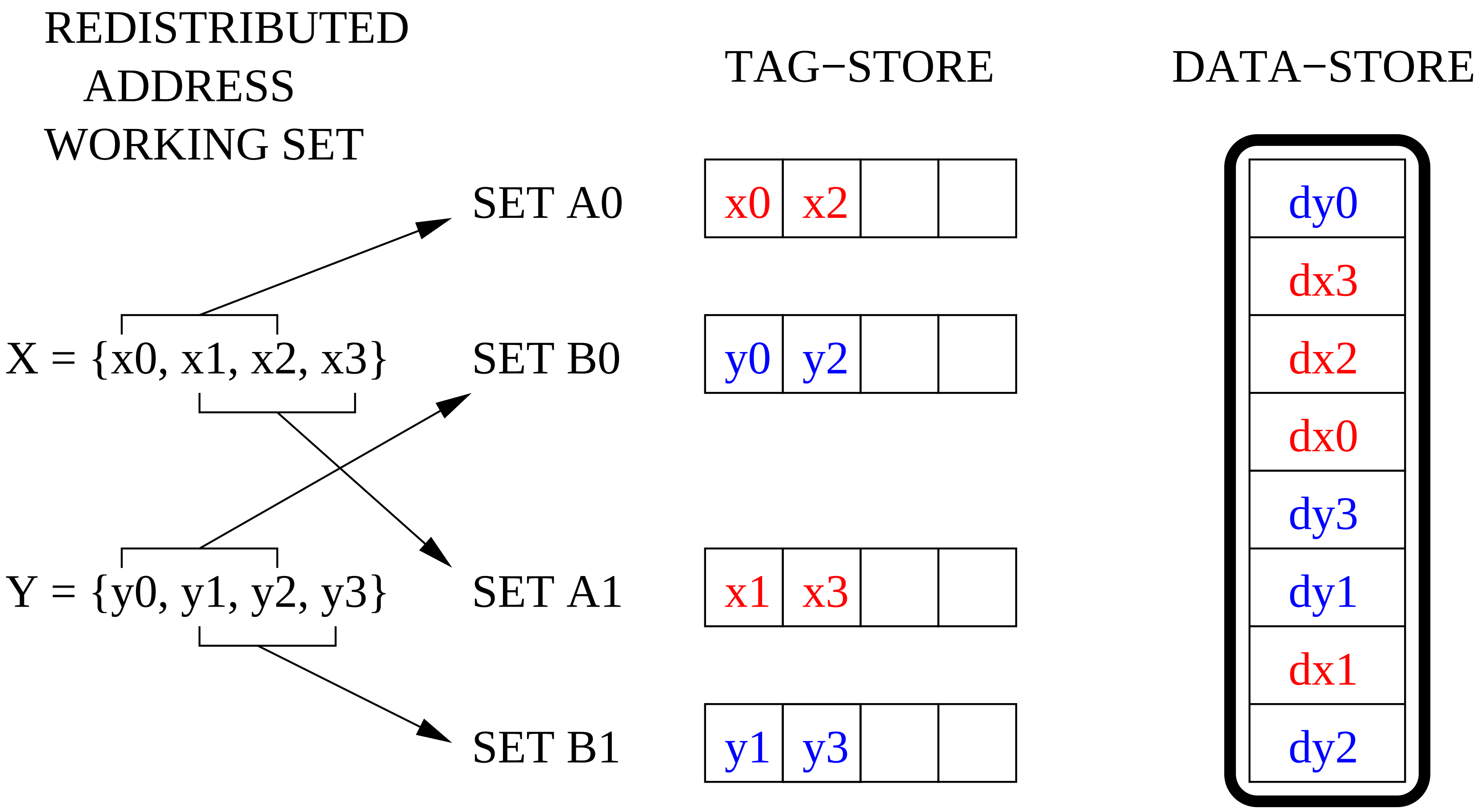


Example of Local Replacement



Static partitioning of resources.

Example of Global Replacement



REDISTRIBUTED
ADDRESS
WORKING SET

X = {x0, x1, x2, x3}

Y = {y0, y1, y2, y3}

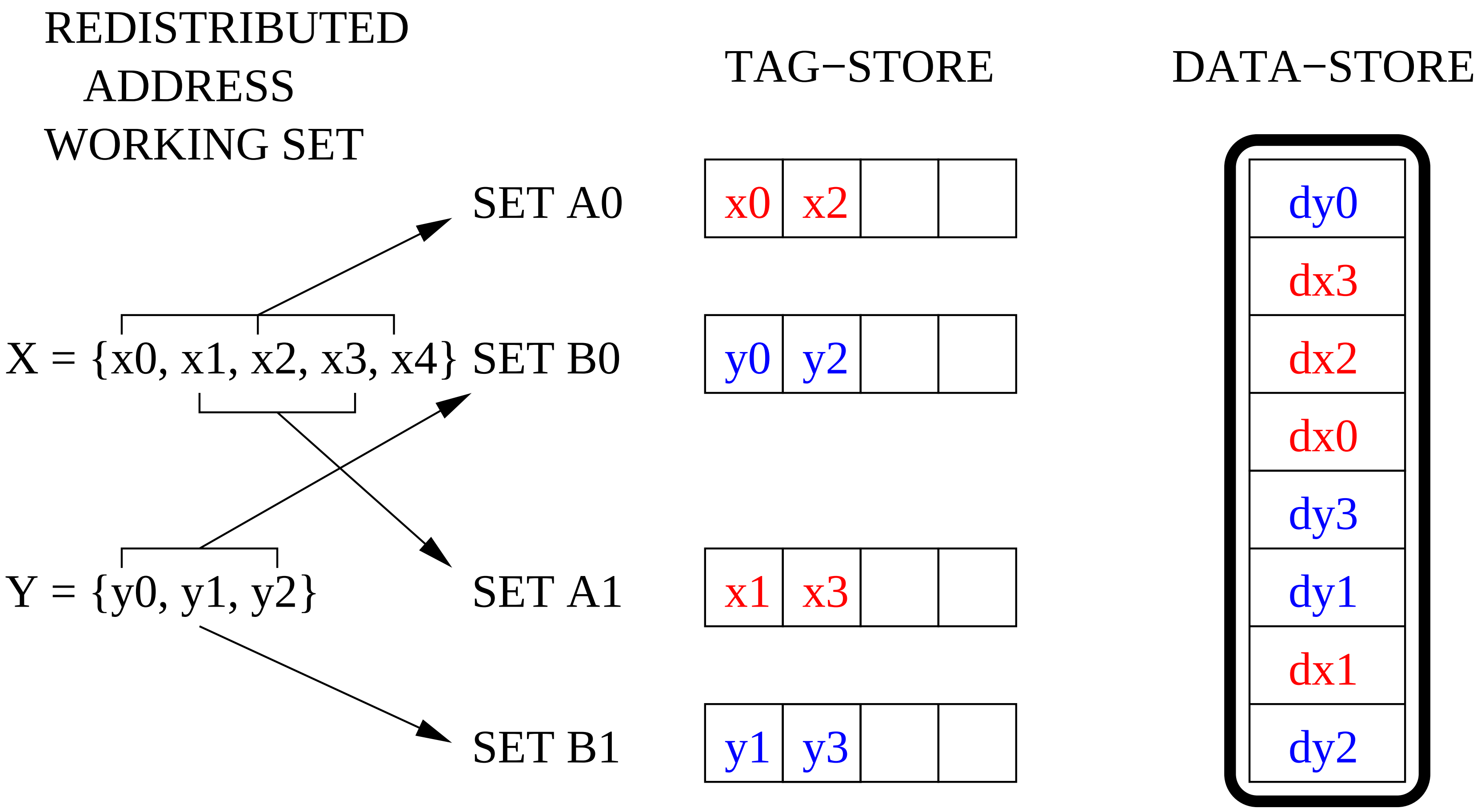
TAG−STORE

SET A0	x0	x2		
SET B0	y0	y2		
SET A1	x1	x3		
SET B1	y1	y3		

DATA−STORE

dy0
dx3
dx2
dx0
dy3
dy1
dx1
dy2

Example of Global Replacement



REDISTRIBUTED
ADDRESS
WORKING SET

X = {x0, x1, x2, x3, x4}

Y = {y0, y1, y2}

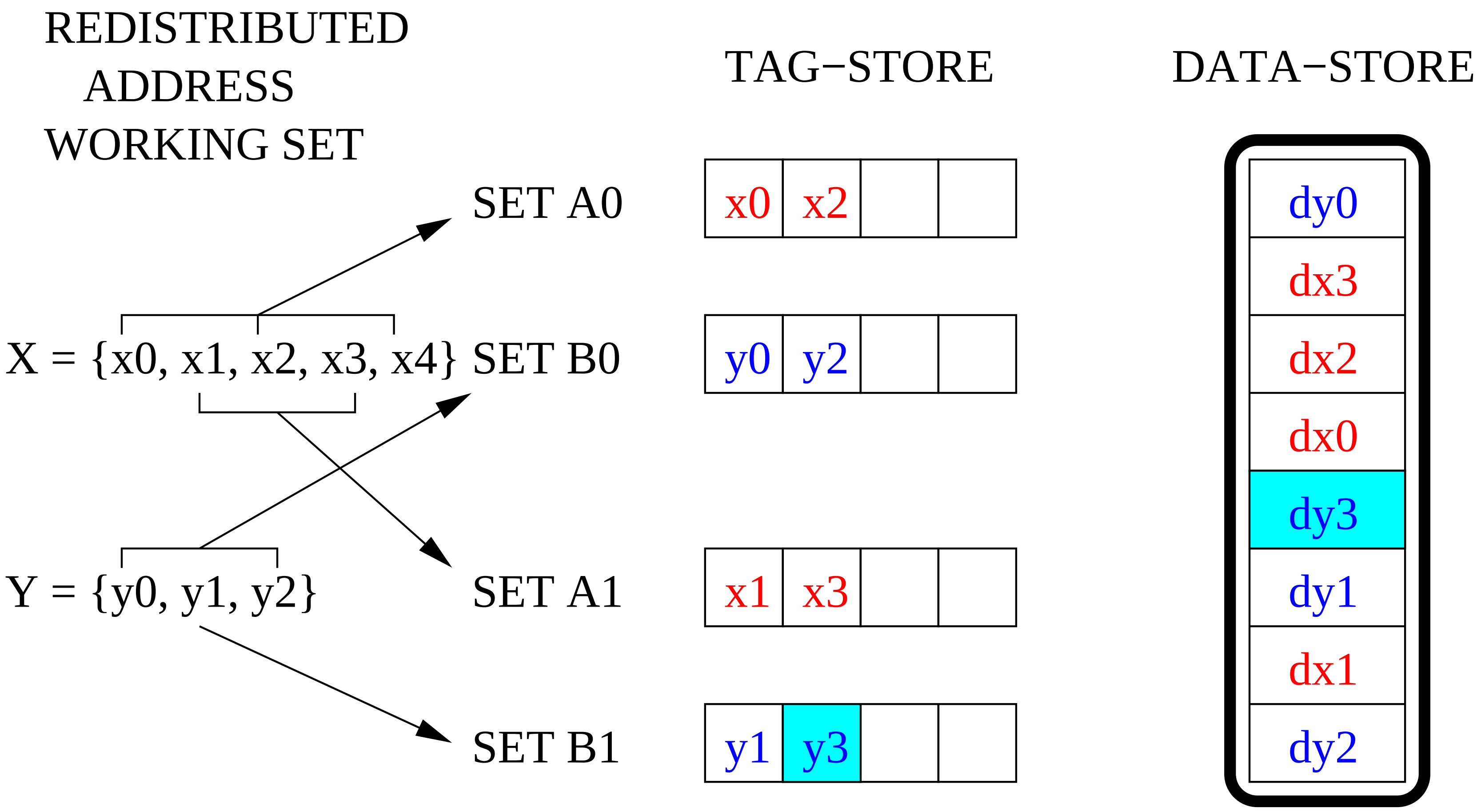
TAG−STORE

DATA−STORE

SET A0	x0	x2		
SET B0	y0	y2		
SET A1	x1	x3		
SET B1	y1	y3		

dy0
dx3
dx2
dx0
dy3
dy1
dx1
dy2

Example of Global Replacement



REDISTRIBUTED
ADDRESS
WORKING SET

X = {x0, x1, x2, x3, x4}

Y = {y0, y1, y2}

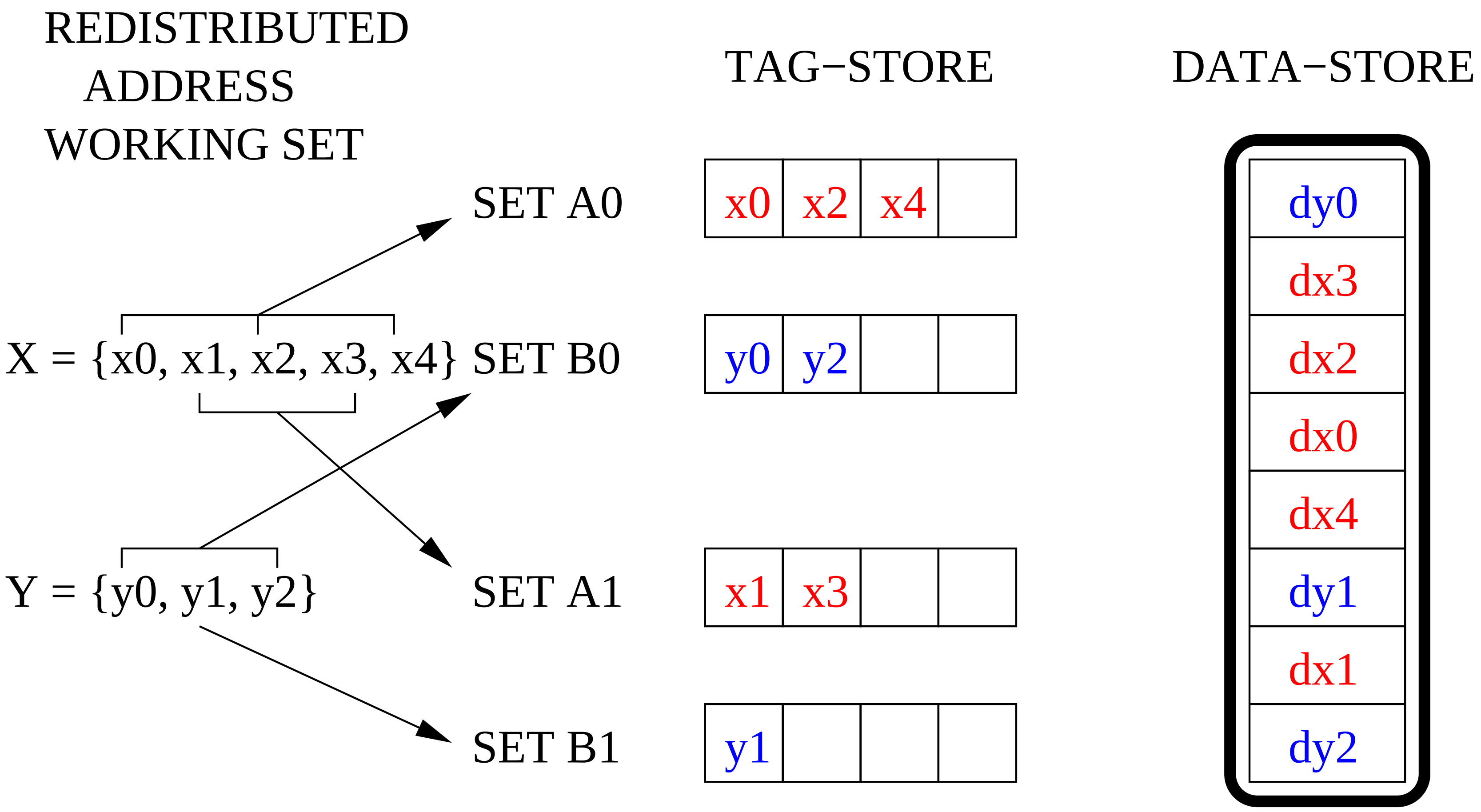
TAG−STORE

SET A0	x0	x2		
SET B0	y0	y2		
SET A1	x1	x3		
SET B1	y1	y3		

DATA−STORE

dy0
dx3
dx2
dx0
dy3
dy1
dx1
dy2

Example of Global Replacement



REDISTRIBUTED
ADDRESS
WORKING SET

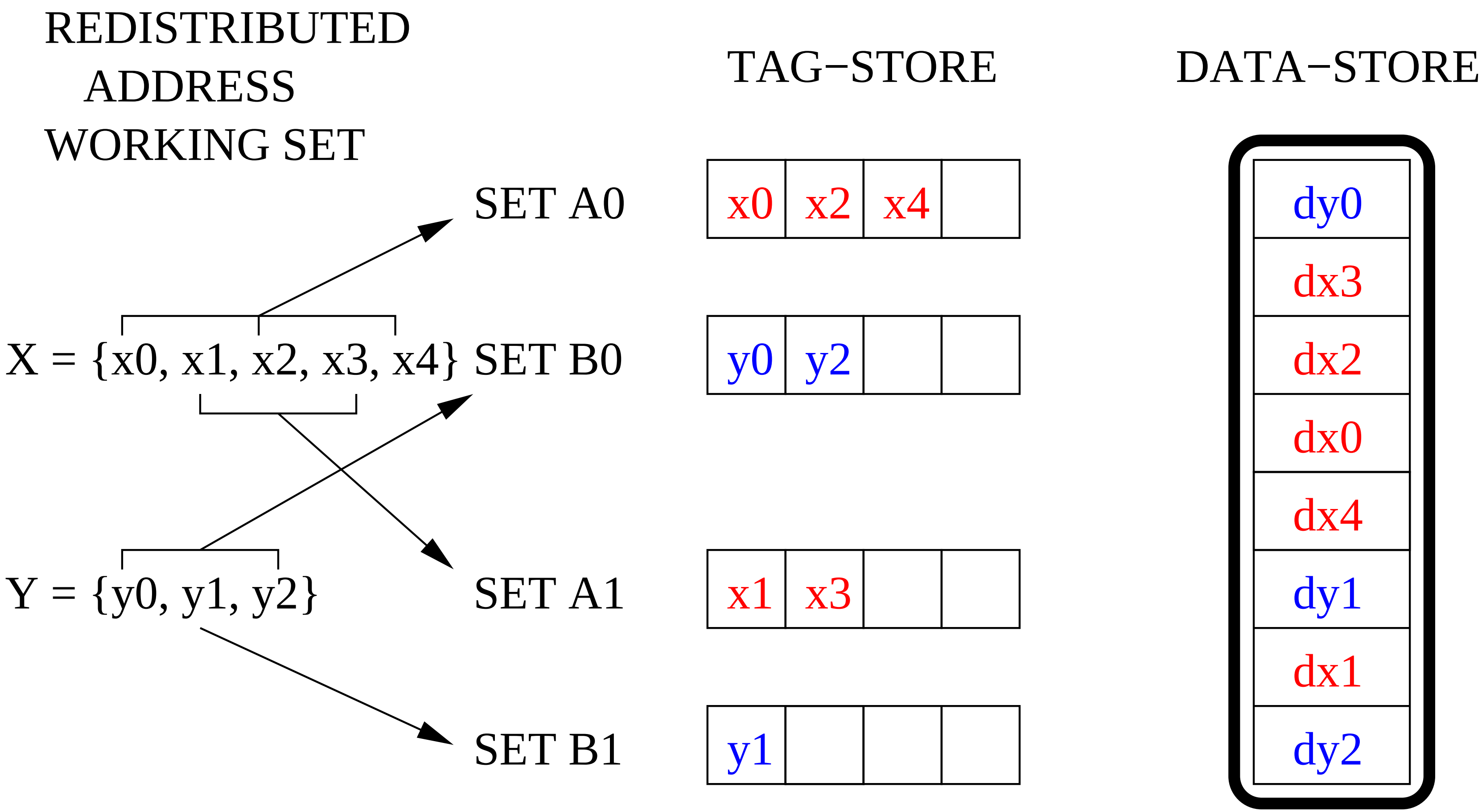
X = {x0, x1, x2, x3, x4}

Y = {y0, y1, y2}

	TAG−STORE			
SET A0	x0	x2	x4	
SET B0	y0	y2		
SET A1	x1	x3		
SET B1	y1			

DATA−STORE
dy0
dx3
dx2
dx0
dx4
dy1
dx1
dy2

Example of Global Replacement



REDISTRIBUTED
ADDRESS
WORKING SET

X = {x0, x1, x2, x3, x4}

Y = {y0, y1, y2}

TAG−STORE

SET A0	x0	x2	x4	
SET B0	y0	y2		
SET A1	x1	x3		
SET B1	y1			

DATA−STORE

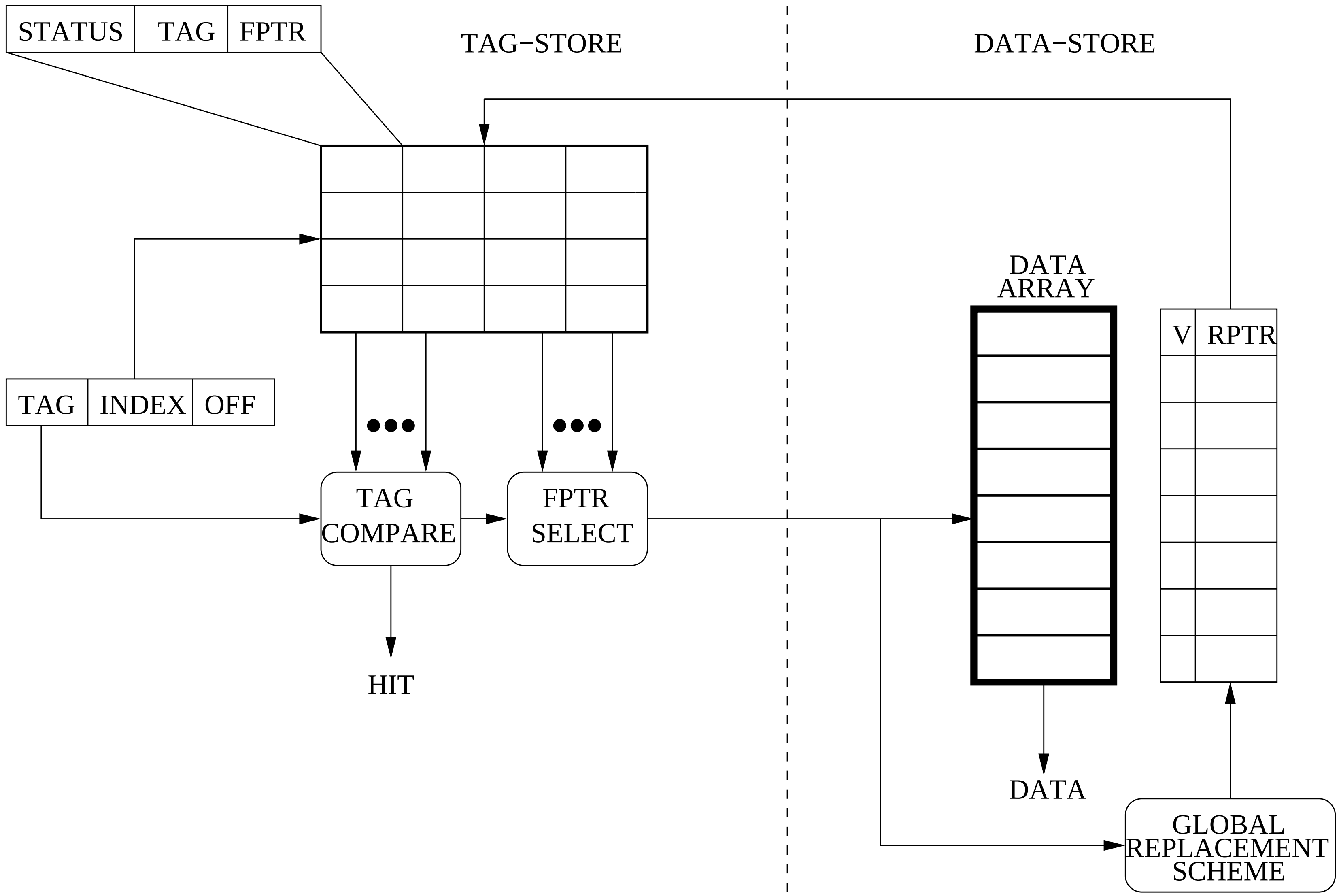
dy0
dx3
dx2
dx0
dx4
dy1
dx1
dy2

Dynamic sharing of resources!!

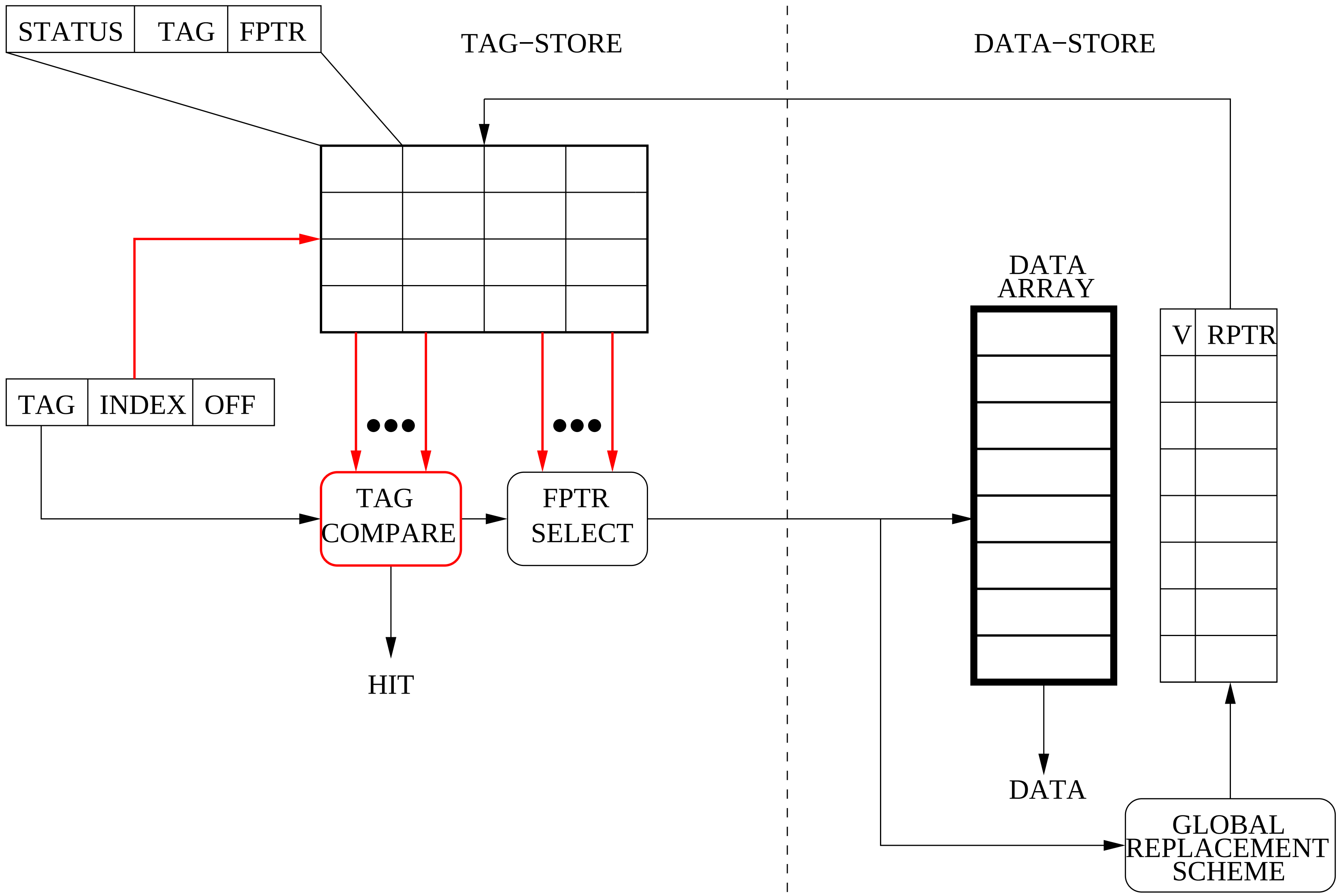
Outline

- Introduction
- Example of Local and Global Replacement
- The V-Way Cache
- Evaluation
- Related Work and Conclusion

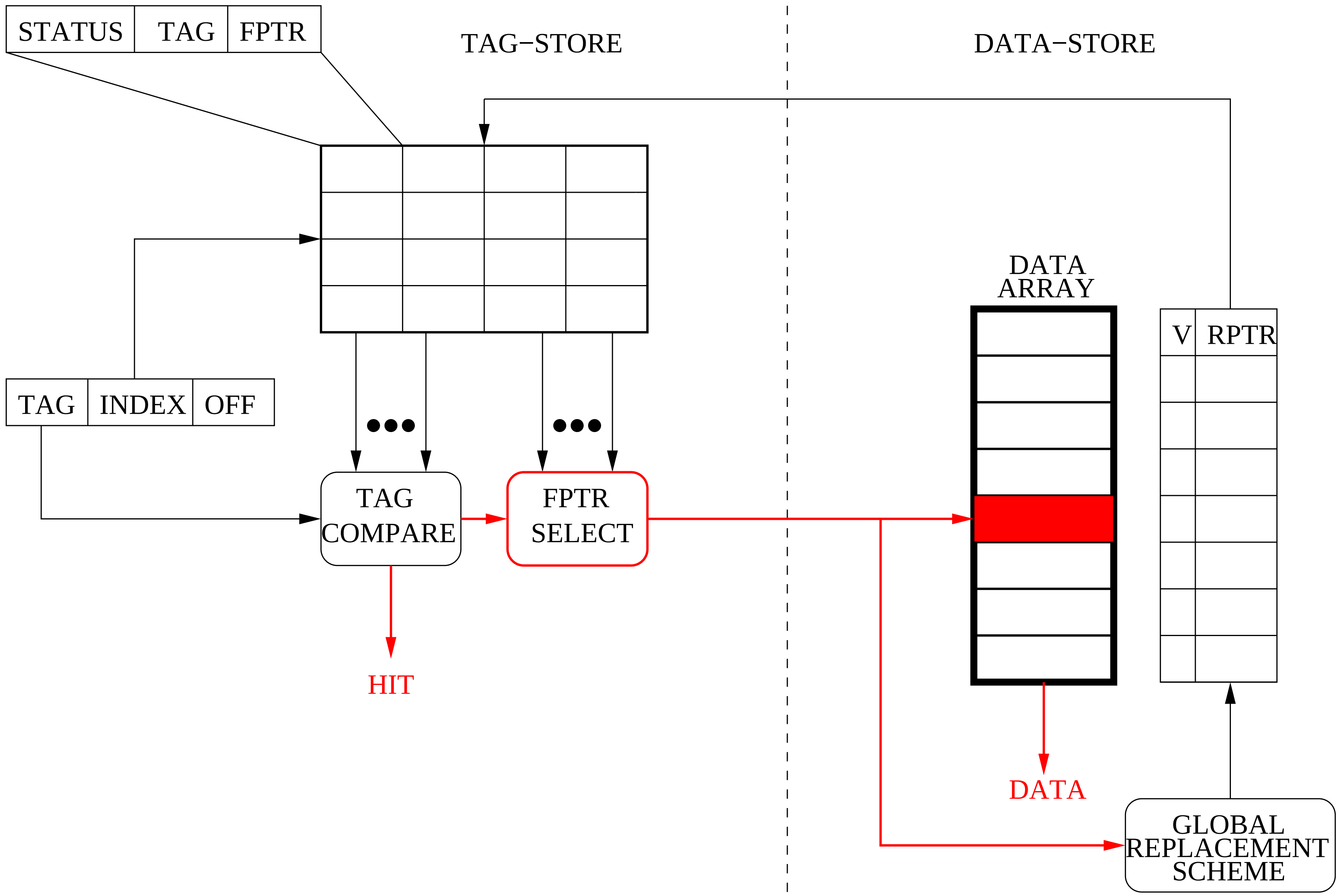
The V-Way Cache



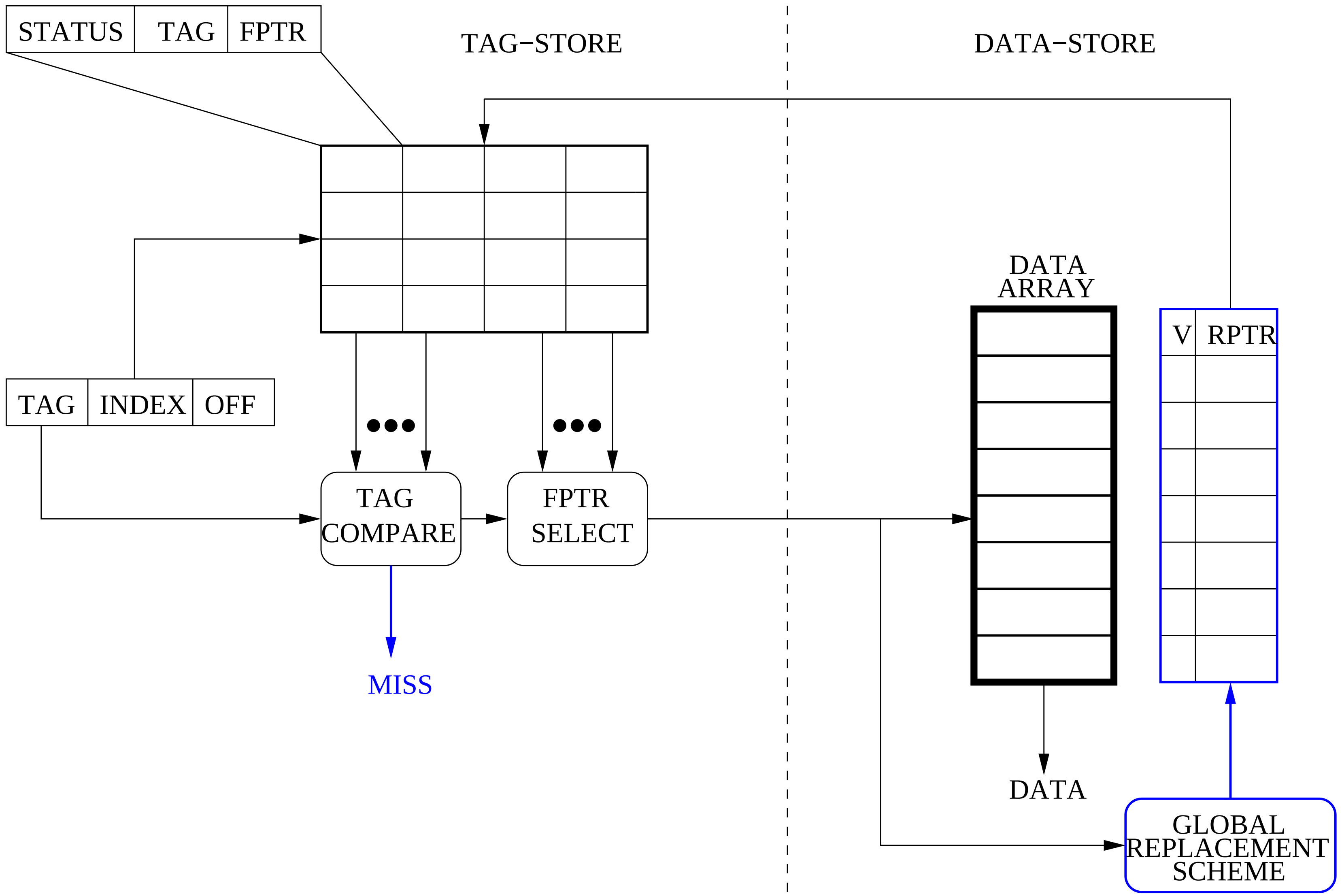
The V-Way Cache



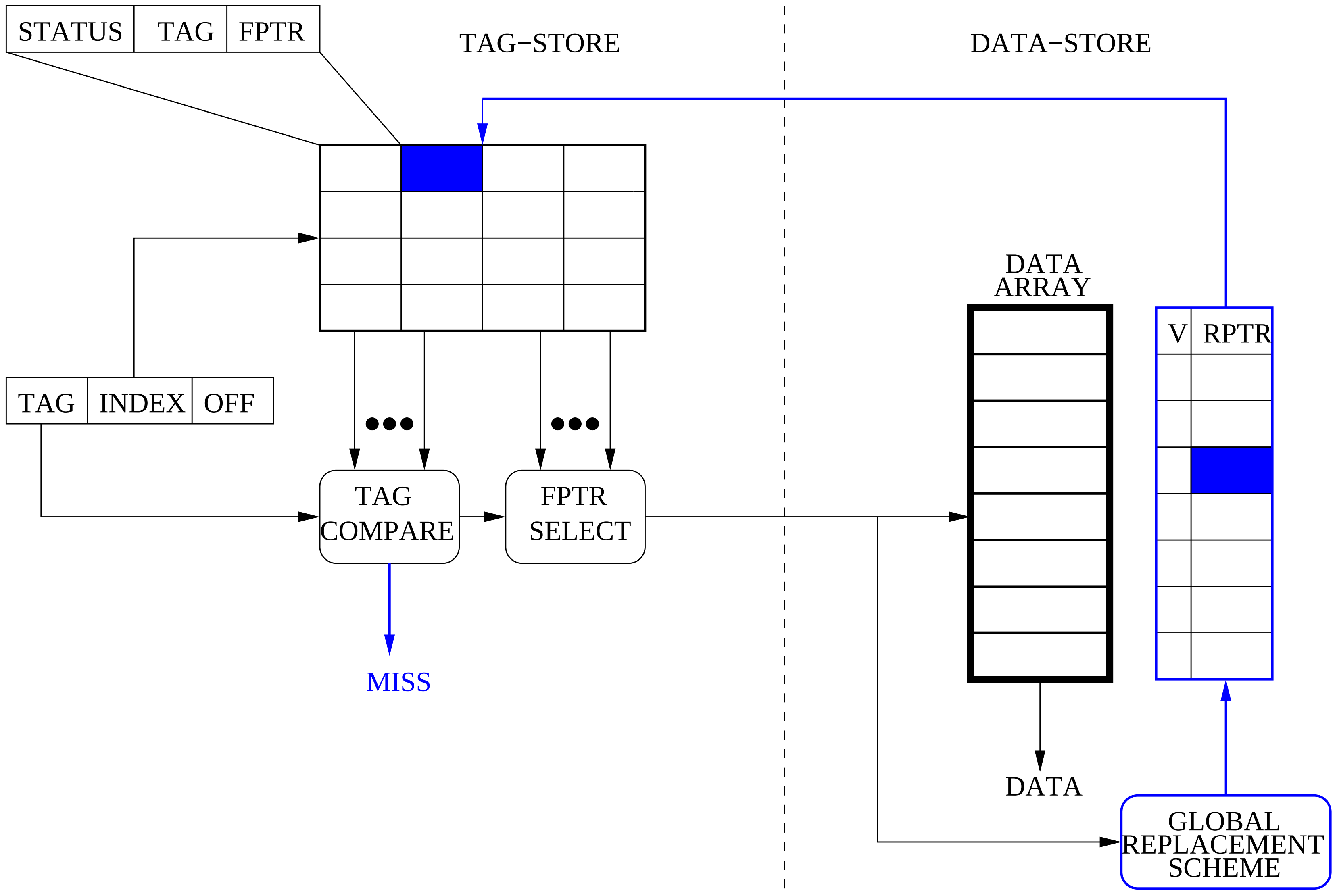
The V-Way Cache



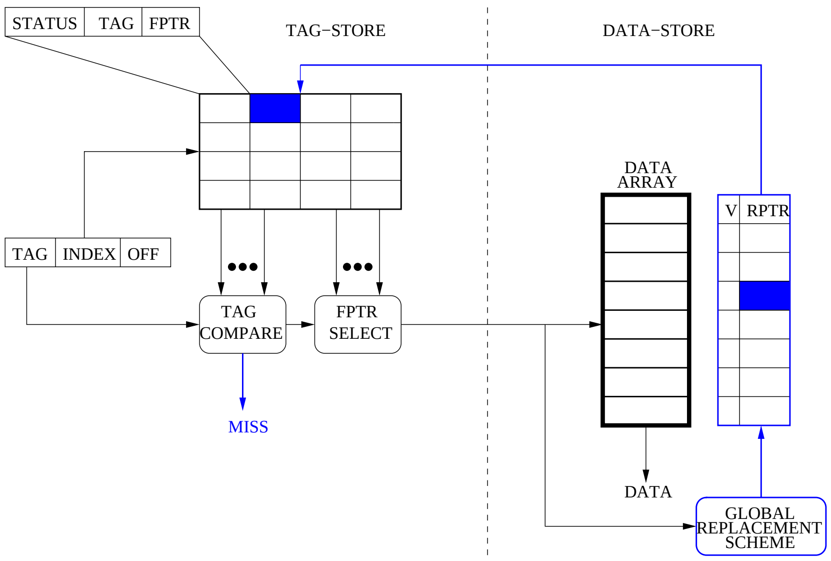
The V-Way Cache



The V-Way Cache

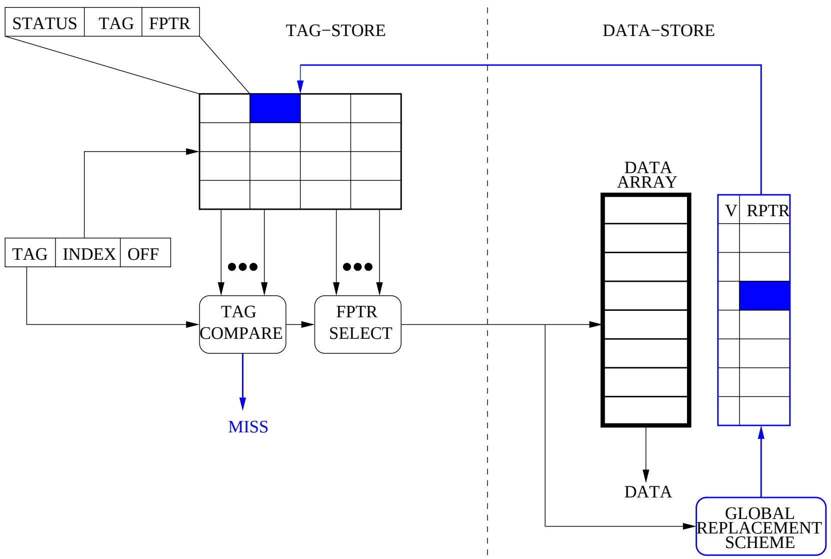


The V-Way Cache



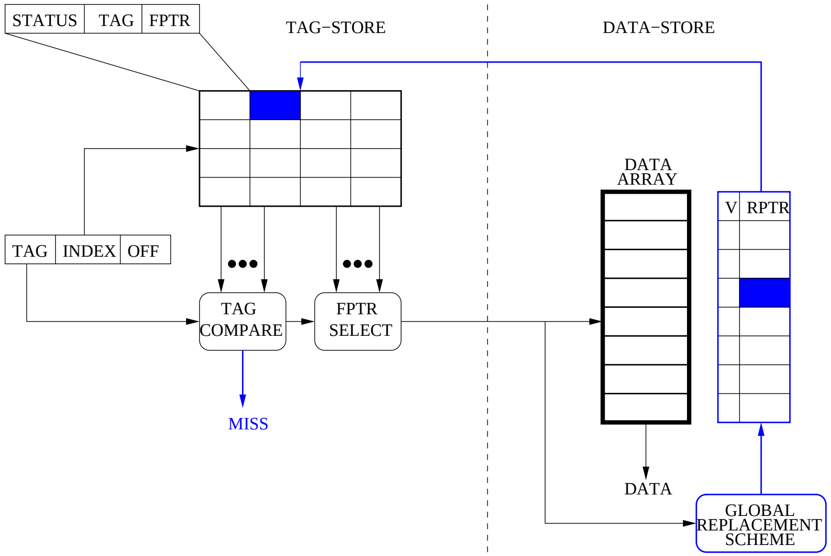
Configuration	Tag Access	Data Replacement
Set-Associative	Fast 🙂	Local ☹
Fully-Associative		
V-Way		

The V-Way Cache



Configuration	Tag Access	Data Replacement
Set-Associative	Fast 🙂	Local ☹
Fully-Associative	Slow ☹	Global 🙂
V-Way		

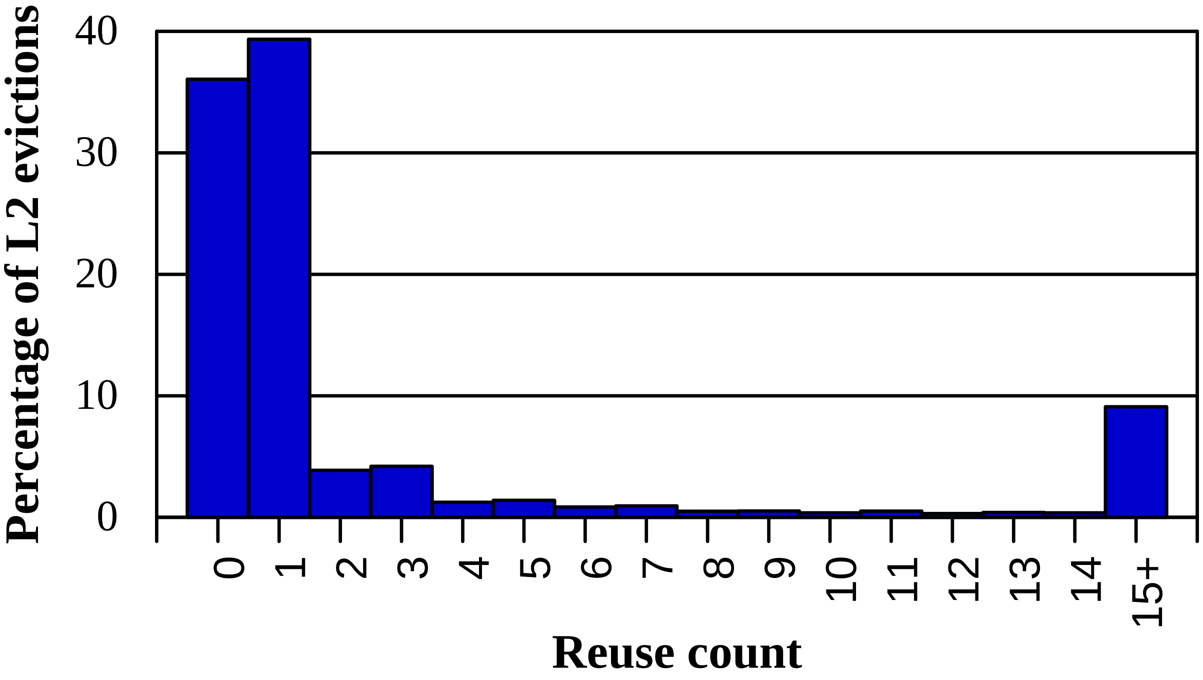
The V-Way Cache



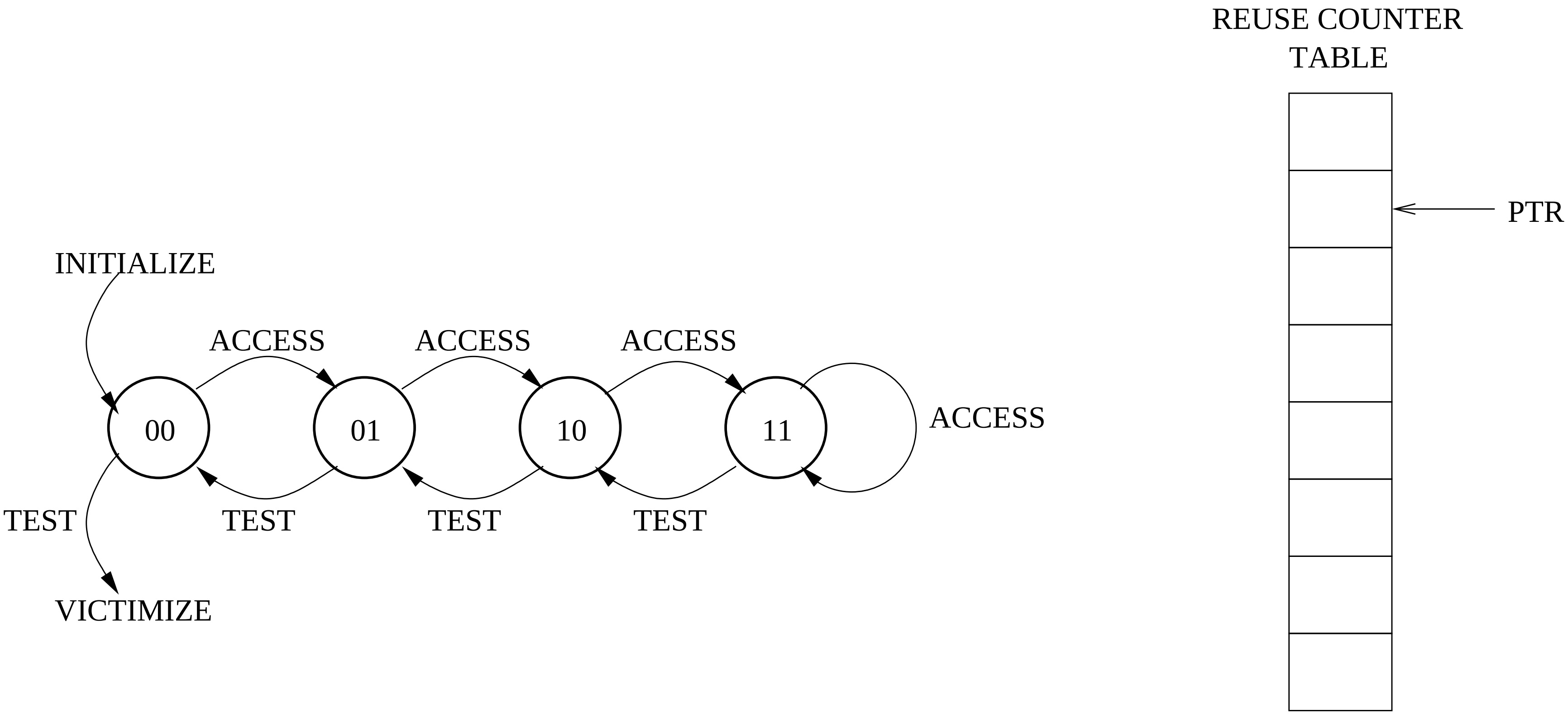
Configuration	Tag Access	Data Replacement
Set-Associative	Fast 🙂	Local 🙁
Fully-Associative	Slow 🙁	Global 🙂
V-Way	Fast 🙂	Global 🙂

A Practical Global Replacement Algorithm

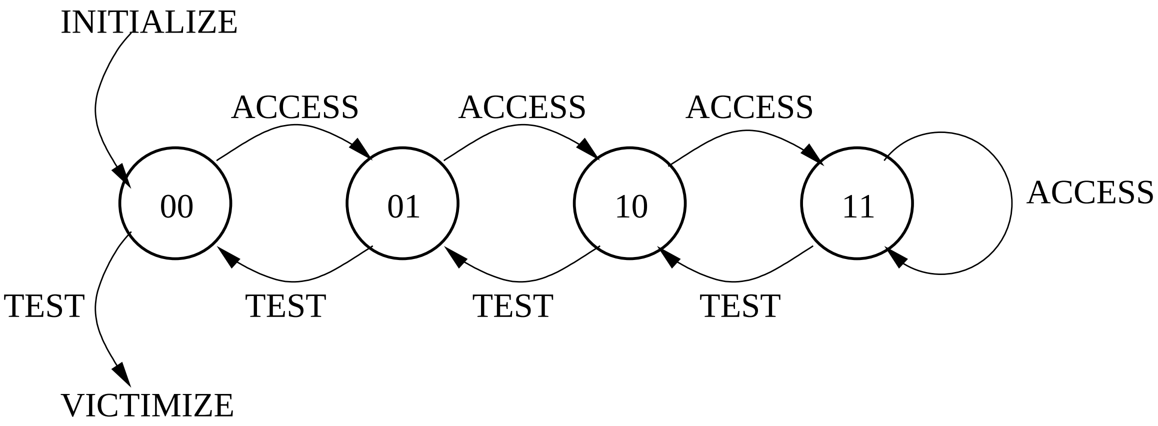
- LRU is impractical because there are thousands of lines
- Second level cache access stream is a filtered version of the program access stream
- Reuse frequency is skewed towards the low end

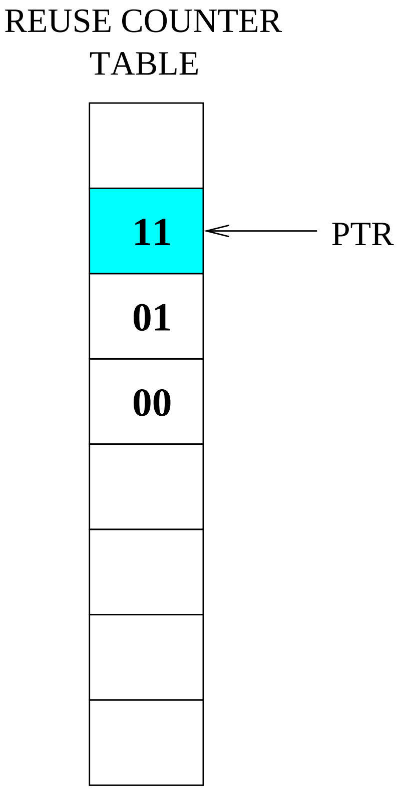


Reuse Replacement

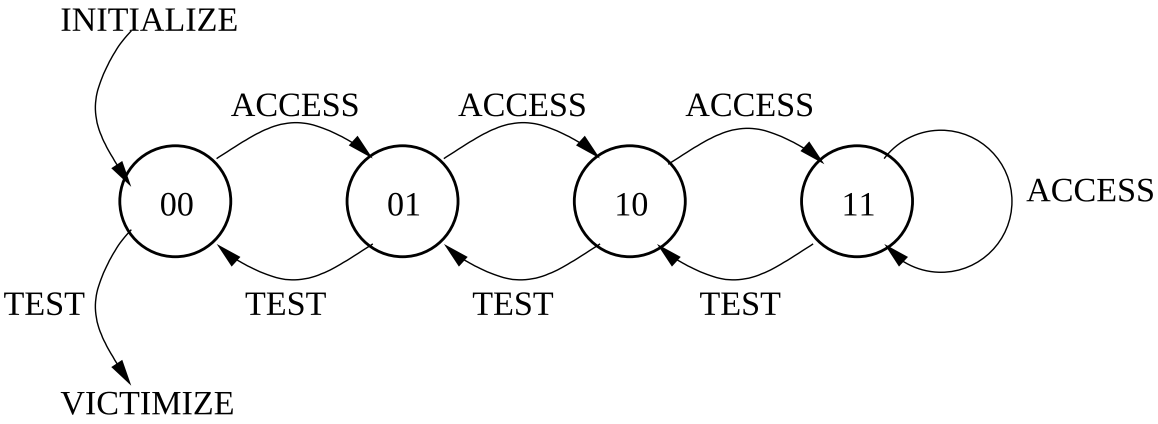


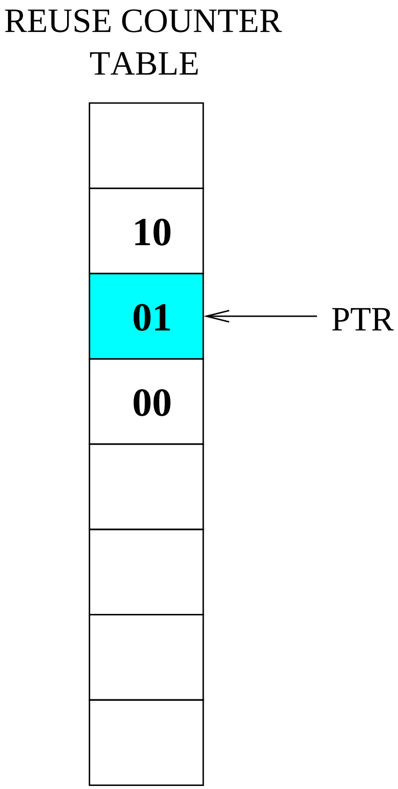
Reuse Replacement





Reuse Replacement

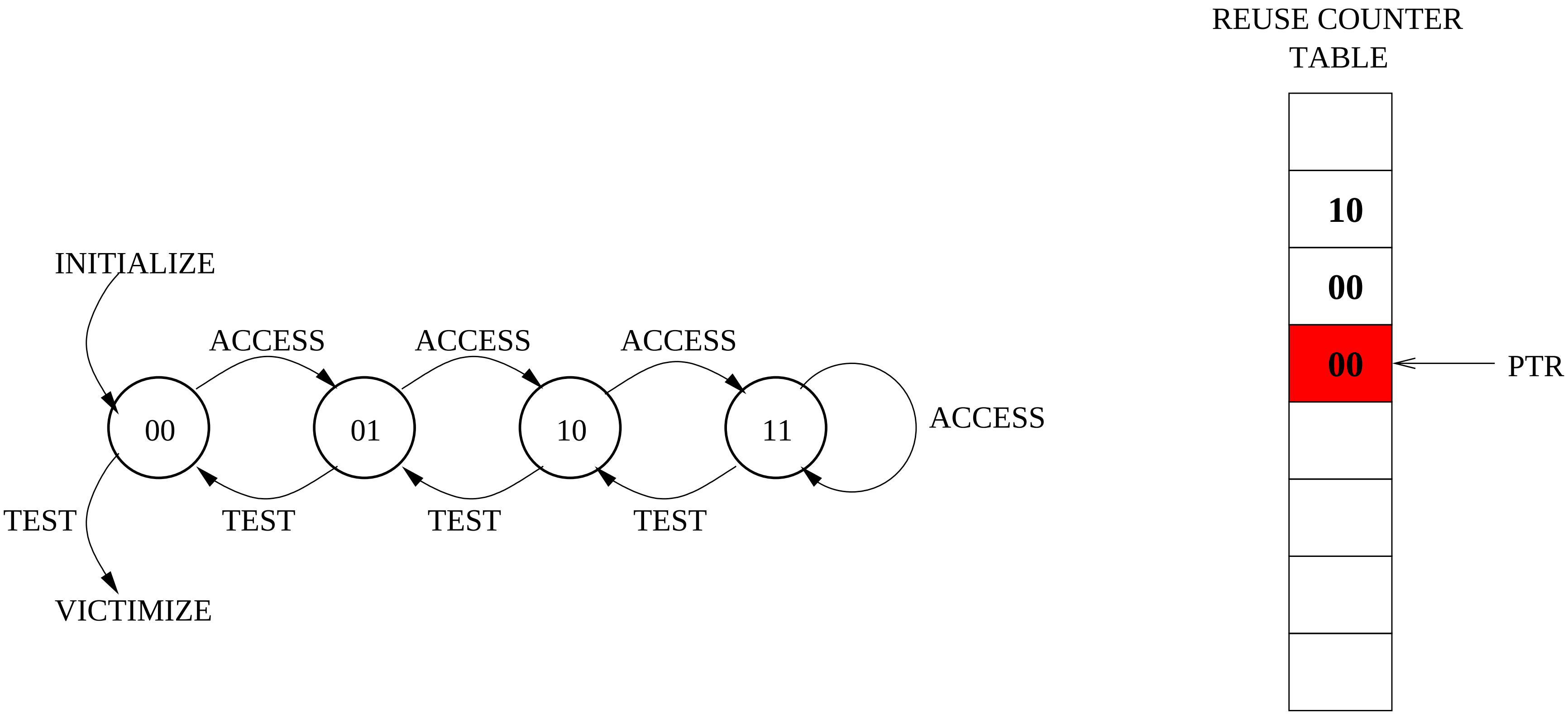




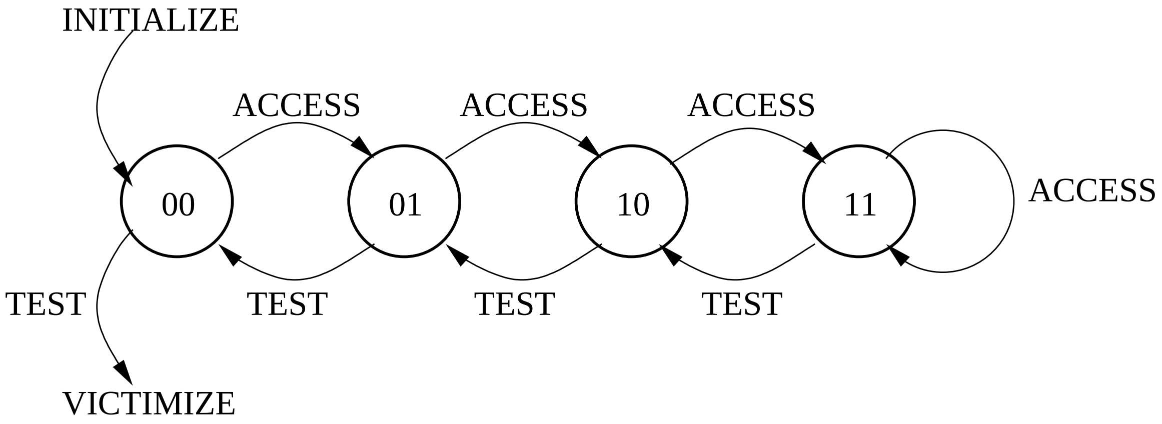
REUSE COUNTER TABLE

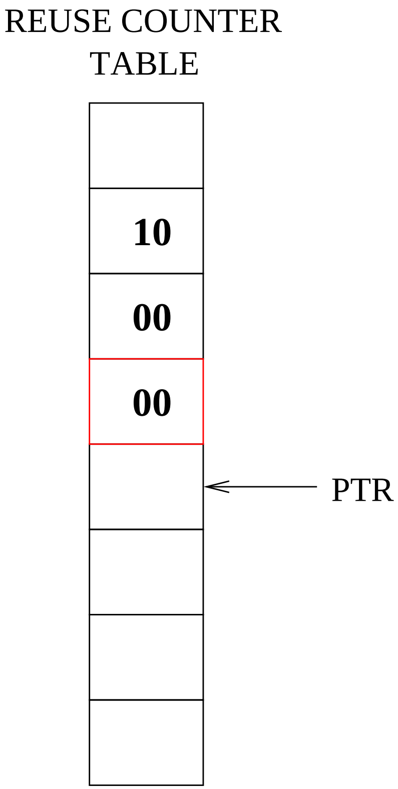
10
01
00

Reuse Replacement



Reuse Replacement





Victim Distance for Reuse Replacement

- Problem of variable replacement latency
  - Average victim distance: 3.9
  - Worst case victim distance: 1888

Victim Distance for Reuse Replacement

- Problem of variable replacement latency
  - Average victim distance: 3.9
  - Worst case victim distance: 1888

- Solution
  - Test eight counters each cycle
  - Limit search to five cycles

Latency (in cycles)	1	2	3	4	5
Probability (victim)	91.3%	96.9%	98.3%	98.9%	99.2%

Outline

- Introduction
- Example of Local and Global Replacement
- The V-Way Cache
- Evaluation
- Related Work and Conclusion

Evaluation Outline

- Experimental Methodology
- Reduction in Misses with the V-Way Cache
- Comparing Reuse Replacement and LRU
- Storage, Latency, and Energy Cost
- Impact on IPC

Experimental Methodology

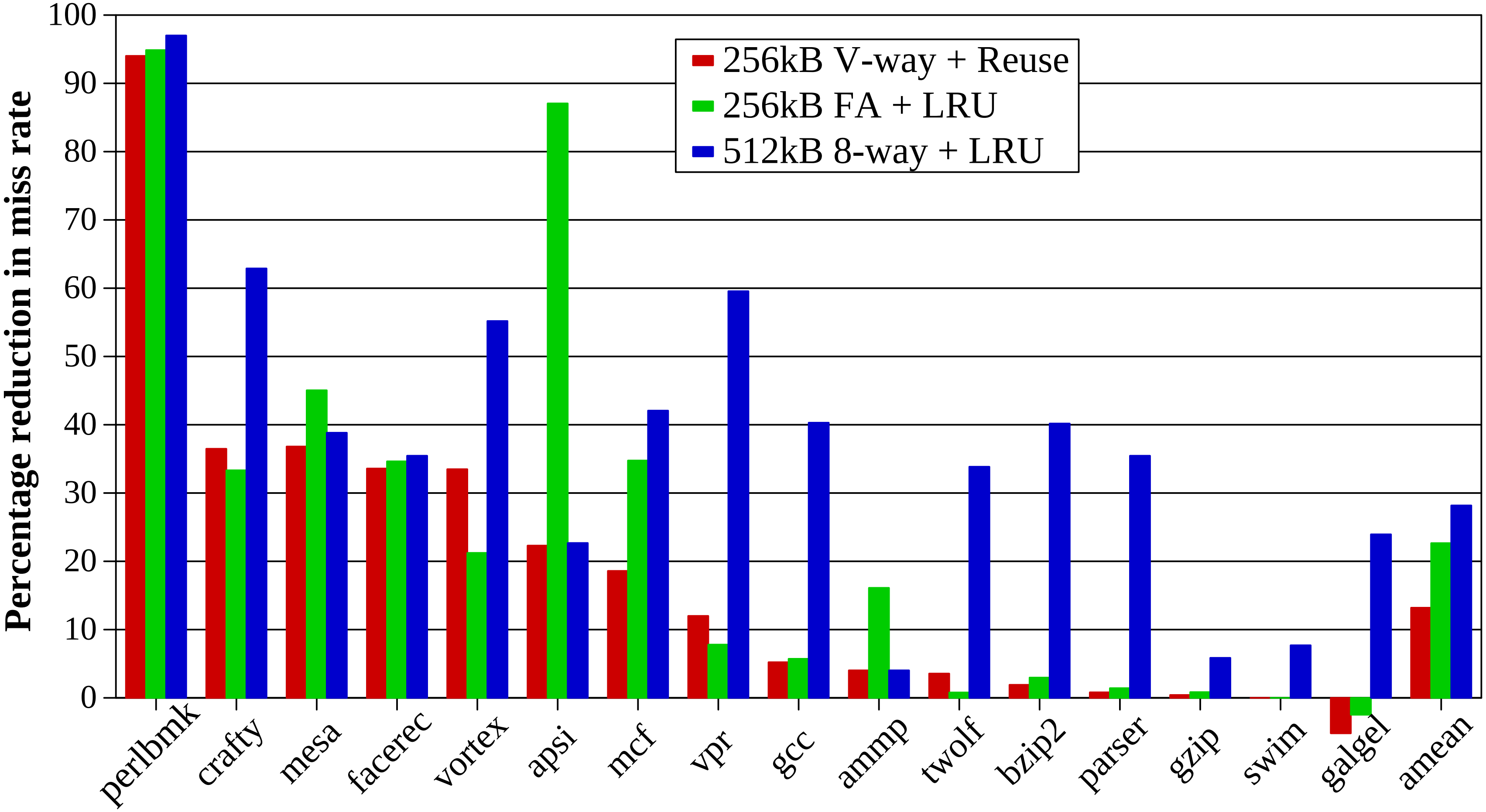
- First level I-cache, D-cache: 16kB, 2-way, 64B linesize, LRU
- Baseline L2: Unified, 256kB, 8-way, 128B linesize, LRU
- Benchmarks: SPEC CPU2000

Reduction in Misses with the V-Way Cache

- Primary upper bound: Fully associative cache
- Secondary upper bound: Double sized cache

Reduction in Misses with the V-Way Cache

- Primary upper bound: Fully associative cache
- Secondary upper bound: Double sized cache



Comparing Reuse Replacement and LRU

Comparison of miss-rate for LRU and Reuse replacement

Bmk	bzip2	crafty	gcc	gzip	mcf	parser	perl.	twolf	vortex
LRU	34.6	1.1	3.8	2.4	29.5	32.7	0.1	36.5	8.5
Reuse	35.0	1.0	3.8	2.4	29.9	32.9	0.1	35.4	7.1

Bmk	vpr	ammp	apsi	facerec	galgel	mesa	swim	amean
LRU	11.0	50.0	34.8	50.7	8.3	3.4	65.3	23.3
Reuse	10.5	50.0	34.8	50.6	8.5	3.5	65.3	23.2

Storage, Latency, and Energy Cost

- Storage needed for extra tags, FPTR, RPTR, and Reuse bits

Line-size	Miss-rate reduction	Increase in area
128 B	13.2%	5.8%
256 B	14.9%	2.9%

Storage, Latency, and Energy Cost

- Storage needed for extra tags, FPTR, RPTR, and Reuse bits

Line-size	Miss-rate reduction	Increase in area
128 B	13.2%	5.8%
256 B	14.9%	2.9%

- Delay due to more tags and FPTR selection: 0.13 ns

Storage, Latency, and Energy Cost

- Storage needed for extra tags, FPTR, RPTR, and Reuse bits

Line-size	Miss-rate reduction	Increase in area
128 B	13.2%	5.8%
256 B	14.9%	2.9%

- Delay due to more tags and FPTR selection: 0.13 ns

- Energy in accessing bigger tag-store

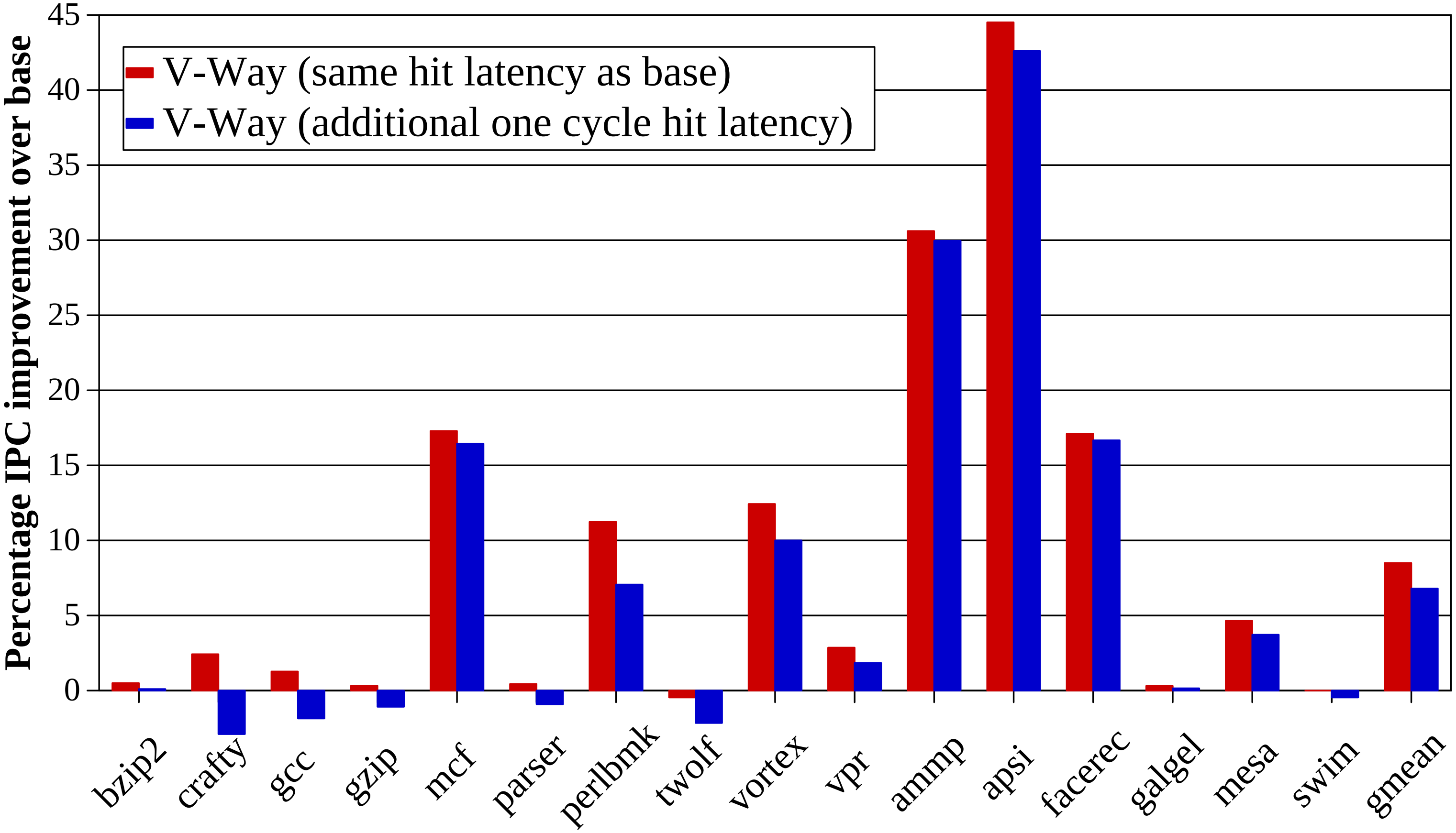
Parallel lookup	Baseline	V-Way
1.02nJ	0.35nJ	0.40nJ

Impact on IPC

- Pipeline: 12 stage, 8 wide with 128 entry reservation station
- L1 hit latency of 2 cycles and L2 hit latency of 10 cycles
- L3/Main memory: access-latency of 80 cycles

Impact on IPC

- Pipeline: 12 stage, 8 wide with 128 entry reservation station
- L1 hit latency of 2 cycles and L2 hit latency of 10 cycles
- L3/Main memory: access-latency of 80 cycles



Outline

- Introduction
- Example of Local and Global Replacement
- The V-Way Cache
- Evaluation
- Related Work and Conclusion

Related Work

- Extra storage for conflict misses: Victim cache [Jouppi ISCA'90]
- Multi-probe techniques
  - Predictive sequential associative cache [Calder+ HPCA'96]
  - Adaptive group associative cache [Peir+ ASPLOS'98]
- Cache indexing function
  - Skewed associativity [Seznec ISCA'93]
  - Prime-modulo indexing [Kharbutli+ HPCA'04]
- Software managed fully associative cache: IIC [Hallnor+ ISCA'00]

Other Possible Applications of the V-Way Cache

- Platform for global replacement with inbuilt shadow directory
- Tag inclusion data exclusion [Piranha ISCA’00]
- Cache compression [Hallnor+ HPCA’05]
- Interaction with NuRAPID [Chishti+ MICRO’03]

Conclusion

- Traditional cache assumes uniform accesses across sets
- Global replacement allows the V-Way cache to vary the number of valid ways depending on the set demand
- Reuse replacement is fast and performs comparable to LRU
- V-Way cache can lower miss-rate and improve performance
- V-Way cache can serve as an infrastructure for other optimizations

Questions