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

ADDRESS WORKING SET

\[ X = \{x_0, x_1, x_2, x_3\} \]

\[ Y = \{y_0, y_1, y_2, y_3\} \]

TAG–STORE

\[ x_0 \quad x_1 \quad x_2 \quad x_3 \]

\[ y_0 \quad y_1 \quad y_2 \quad y_3 \]

DATA–STORE

\[ \text{dx}_0 \]
\[ \text{dx}_1 \]
\[ \text{dx}_2 \]
\[ \text{dx}_3 \]

\[ \text{dy}_0 \]
\[ \text{dy}_1 \]
\[ \text{dy}_2 \]
\[ \text{dy}_3 \]
Example of Local Replacement

ADDRESS WORKING SET

X = \{x_0, x_1, x_2, x_3, x_4\}

Y = \{y_0, y_1, y_2\}

TAG–STORE

\begin{align*}
\text{SET A} & : \quad x_0 \quad x_1 \quad x_2 \quad x_3 \\
\text{SET B} & : \quad y_0 \quad y_1 \quad y_2 \quad y_3
\end{align*}

DATA–STORE

\begin{align*}
\text{dx}_0 & \\
\text{dx}_1 & \\
\text{dx}_2 & \\
\text{dx}_3 & \\
\text{dy}_0 & \\
\text{dy}_1 & \\
\text{dy}_2 & \\
\text{dy}_3 & 
\end{align*}
Example of Local Replacement

ADDRESS
WORKING SET

X = \{x_0, x_1, x_2, x_3, x_4\}
Y = \{y_0, y_1, y_2\}

TAG–STORE

SET A
x_4 \ x_1 \ x_2 \ x_3

DATA–STORE

SET B
y_0 \ y_1 \ y_2 \ y_3

dx_4

dx_1

dx_2

dx_3

dy_0

dy_1

dy_2

dy_3
Example of Local Replacement

ADDRESS WORKING SET

X = \{x_0, x_1, x_2, x_3, x_4\}
Y = \{y_0, y_1, y_2\}

TAG–STORE

THRASH

DATA–STORE

SET A

\begin{array}{cccc}
x_4 & x_1 & x_2 & x_3 \\
\end{array}

\begin{array}{cccc}
dx_4 & dx_1 & dx_2 & dx_3 \\
\end{array}

SET B

\begin{array}{cccc}
y_0 & y_1 & y_2 & y_3 \\
\end{array}

\begin{array}{cccc}
dy_0 & dy_1 & dy_2 & dy_3 \\
\end{array}

DORMANT WAY
Static partitioning of resources.
Example of Global Replacement

REDISTRIBUTED
ADDRESS
WORKING SET

X = \{x_0, x_1, x_2, x_3\}

Y = \{y_0, y_1, y_2, y_3\}

SET A0

SET B0

SET A1

SET B1

TAG–STORE

DATA–STORE

\begin{align*}
\text{SET A0} & & \text{SET B0} & & \text{SET A1} & & \text{SET B1} \\
x_0 & & x_2 \\
y_0 & & y_2 \\
x_1 & & x_3 \\
y_1 & & y_3 \\
\end{align*}

\begin{align*}
dy_0 \\
dx_3 \\
dx_2 \\
dx_0 \\
dy_3 \\
dy_1 \\
dx_1 \\
dy_2 \\
\end{align*}
Example of Global Replacement

REDISTRIBUTED ADDRESS
WORKING SET

\[ X = \{x_0, x_1, x_2, x_3, x_4\} \]
\[ Y = \{y_0, y_1, y_2\} \]

SET A0
\[ x_0 \ x_2 \]

SET B0
\[ y_0 \ y_2 \]

SET A1
\[ x_1 \ x_3 \]

SET B1
\[ y_1 \ y_3 \]

TAG–STORE

DATA–STORE

\[ dy_0 \]
\[ dx_3 \]
\[ dx_2 \]
\[ dx_0 \]
\[ dy_3 \]
\[ dy_1 \]
\[ dx_1 \]
\[ dy_2 \]
Example of Global Replacement

REDISTRIBUTED ADDRESS
WORKING SET

X = \{x_0, x_1, x_2, x_3, x_4\} SET B0

Y = \{y_0, y_1, y_2\} SET A1

SET A0
\[
\begin{array}{c}
  x_0 \quad x_2
\end{array}
\]

TAG–STORE
\[
\begin{array}{c}
  y_0 \quad y_2
\end{array}
\]

SET B0
\[
\begin{array}{c}
  \quad dy_0
\end{array}
\]

SET A1
\[
\begin{array}{c}
  x_1 \quad x_3
\end{array}
\]

SET B1
\[
\begin{array}{c}
  y_1 \quad y_3
\end{array}
\]

DATA–STORE
\[
\begin{array}{c}
  \quad dx_0
\end{array}
\]

\[
\begin{array}{c}
  \quad dx_3
\end{array}
\]

\[
\begin{array}{c}
  \quad dx_2
\end{array}
\]

\[
\begin{array}{c}
  \quad dy_3
\end{array}
\]

\[
\begin{array}{c}
  \quad dy_1
\end{array}
\]

\[
\begin{array}{c}
  \quad dx_1
\end{array}
\]

\[
\begin{array}{c}
  \quad dy_2
\end{array}
\]
Example of Global Replacement

REDISTRIBUTED
ADDRESS
WORKING SET

X = \{x_0, x_1, x_2, x_3, x_4\} SET B0

Y = \{y_0, y_1, y_2\} SET A1

SET A0

TAG–STORE

x_0 \ x_2 \ x_4

y_0 \ y_2

x_1 \ x_3

y_1

SET B1

DATA–STORE

dy_0

dx_3

dx_2

dx_0

dx_4

dy_1

dx_1

dy_2
Example of Global Replacement

REDISTRIBUTED
ADDRESS
WORKING SET

X = {x0, x1, x2, x3, x4}   SET B0

Y = {y0, y1, y2}   SET A1

SET B1

TAG–STORE

DATA–STORE

x0  x2  x4

y0  y2

x1  x3

y1

dy0

dx3

dx2

dx0

dx4

dx1

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

STATUS  TAG  FPTR

TAG−STORE

INDEX  OFF

TAG COMPARE

FPTR SELECT

DATA−STORE

DATA ARRAY

V RPTR

GLOBAL REPLACEMENT SCHEME

HIT

DATA
The V-Way Cache
The V-Way Cache
The V-Way Cache

STATUS  TAG  FPTR

TAG–STORE

GLOBAL REPLACEMENT SCHEME

DATA–STORE

DATA ARRAY

MISS

TAG COMPARE

FPTR SELECT

TAG INDEX OFF

FPTRTAGSTATUS

TAG–STORE DATA–STORE
The V-Way Cache

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Tag Access</th>
<th>Data Replacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set-Associative</td>
<td>Fast 😊</td>
<td>Local 😞</td>
</tr>
<tr>
<td>Fully-Associative</td>
<td></td>
<td></td>
</tr>
<tr>
<td>V-Way</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
# The V-Way Cache

## Configuration

<table>
<thead>
<tr>
<th>Configuration</th>
<th>Tag Access</th>
<th>Data Replacement</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set-Associative</td>
<td>Fast 😊</td>
<td>Local 😊</td>
</tr>
<tr>
<td>Fully-Associative</td>
<td>Slow 😞</td>
<td>Global 😊</td>
</tr>
<tr>
<td>V-Way</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Diagram of the V-Way Cache](image)

The International Symposium on Computer Architecture - 2005
# The V-Way Cache

## Configuration

<table>
<thead>
<tr>
<th>Set-Associative</th>
<th>Fully-Associative</th>
<th>V-Way</th>
</tr>
</thead>
</table>

## Tag Access

<table>
<thead>
<tr>
<th>Set-Associative</th>
<th>Fully-Associative</th>
<th>V-Way</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fast 😊</td>
<td>Slow 😞</td>
<td>Fast 😊</td>
</tr>
</tbody>
</table>

## Data Replacement

<table>
<thead>
<tr>
<th>Set-Associative</th>
<th>Fully-Associative</th>
<th>V-Way</th>
</tr>
</thead>
<tbody>
<tr>
<td>Local 😞</td>
<td>Global 😊</td>
<td>Global 😊</td>
</tr>
</tbody>
</table>
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

The International Symposium on Computer Architecture - 2005
Reuse Replacement

The International Symposium on Computer Architecture - 2005
Reuse Replacement

The International Symposium on Computer Architecture - 2005
Reuse Replacement

The diagram illustrates the Reuse Replacement algorithm. It consists of a series of states labeled 00, 01, 10, and 11, connected by transitions labeled INITIATE, TEST, ACCESS, and VICTIMIZE. The REUSE COUNTER TABLE is shown on the right side with the values 10 and 00. The PTR points to the REUSE COUNTER TABLE.
Reuse Replacement

The International Symposium on Computer Architecture - 2005
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

<table>
<thead>
<tr>
<th>Latency (in cycles)</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Probability (victim)</td>
<td>91.3%</td>
<td>96.9%</td>
<td>98.3%</td>
<td>98.9%</td>
<td>99.2%</td>
</tr>
</tbody>
</table>
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

The International Symposium on Computer Architecture - 2005
Comparing Reuse Replacement and LRU

Comparison of miss-rate for LRU and Reuse replacement

<table>
<thead>
<tr>
<th>Bmk</th>
<th>bzip2</th>
<th>crafty</th>
<th>gcc</th>
<th>gzip</th>
<th>mcf</th>
<th>parser</th>
<th>perl.</th>
<th>twolf</th>
<th>vortex</th>
</tr>
</thead>
<tbody>
<tr>
<td>LRU</td>
<td>34.6</td>
<td>1.1</td>
<td>3.8</td>
<td>2.4</td>
<td>29.5</td>
<td>32.7</td>
<td>0.1</td>
<td>36.5</td>
<td>8.5</td>
</tr>
<tr>
<td>Reuse</td>
<td>35.0</td>
<td>1.0</td>
<td>3.8</td>
<td>2.4</td>
<td>29.9</td>
<td>32.9</td>
<td>0.1</td>
<td>35.4</td>
<td>7.1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bmk</th>
<th>vpr</th>
<th>ammp</th>
<th>apsi</th>
<th>facerec</th>
<th>galgel</th>
<th>mesa</th>
<th>swim</th>
<th>amean</th>
</tr>
</thead>
<tbody>
<tr>
<td>LRU</td>
<td>11.0</td>
<td>50.0</td>
<td>34.8</td>
<td>50.7</td>
<td>8.3</td>
<td>3.4</td>
<td>65.3</td>
<td>23.3</td>
</tr>
<tr>
<td>Reuse</td>
<td>10.5</td>
<td>50.0</td>
<td>34.8</td>
<td>50.6</td>
<td>8.5</td>
<td>3.5</td>
<td>65.3</td>
<td>23.2</td>
</tr>
</tbody>
</table>
Storage, Latency, and Energy Cost

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

<table>
<thead>
<tr>
<th>Line-size</th>
<th>Miss-rate reduction</th>
<th>Increase in area</th>
</tr>
</thead>
<tbody>
<tr>
<td>128 B</td>
<td>13.2%</td>
<td>5.8%</td>
</tr>
<tr>
<td>256 B</td>
<td>14.9%</td>
<td>2.9%</td>
</tr>
</tbody>
</table>
Storage, Latency, and Energy Cost

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

<table>
<thead>
<tr>
<th>Line-size</th>
<th>Miss-rate reduction</th>
<th>Increase in area</th>
</tr>
</thead>
<tbody>
<tr>
<td>128 B</td>
<td>13.2%</td>
<td>5.8%</td>
</tr>
<tr>
<td>256 B</td>
<td>14.9%</td>
<td>2.9%</td>
</tr>
</tbody>
</table>

- 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

<table>
<thead>
<tr>
<th>Line-size</th>
<th>Miss-rate reduction</th>
<th>Increase in area</th>
</tr>
</thead>
<tbody>
<tr>
<td>128 B</td>
<td>13.2%</td>
<td>5.8%</td>
</tr>
<tr>
<td>256 B</td>
<td>14.9%</td>
<td>2.9%</td>
</tr>
</tbody>
</table>

Delay due to more tags and FPTR selection: 0.13 ns

Energy in accessing bigger tag-store

<table>
<thead>
<tr>
<th>Parallel lookup</th>
<th>Baseline</th>
<th>V-Way</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.02nJ</td>
<td>0.35nJ</td>
<td>0.40nJ</td>
</tr>
</tbody>
</table>
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

The International Symposium on Computer Architecture - 2005
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