NTNU

Innovation and Creativity

Low-Cost Open-Page Prefetch Scheduling in Chip Multiprocessors

Marius Grannæs   Magnus Jahre   Lasse Natvig

Oct 14th 2008

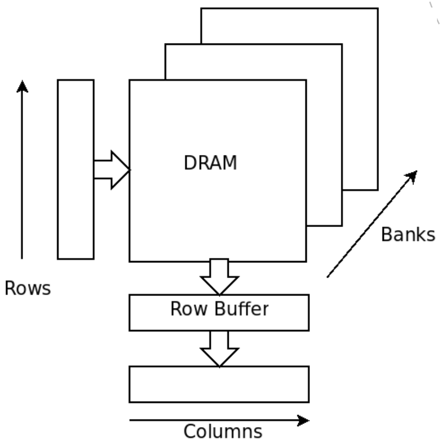
Idea

- Modern DRAM is complex
- DRAM matrix (capacitors) $\Rightarrow$ slow, but high density
- It is much faster to access data on the same row (logic)
- First Ready - First Come, First Served (Rixner et al.)
- Prefetching usually prefetches data in close spatial proximity (sequentially, small strides)

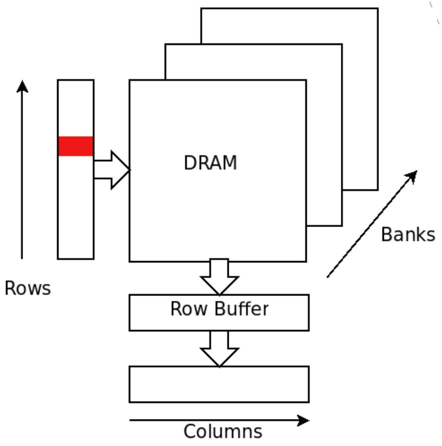
Can we exploit open pages to improve prefetching?

NTNU
Innovation and Creativity

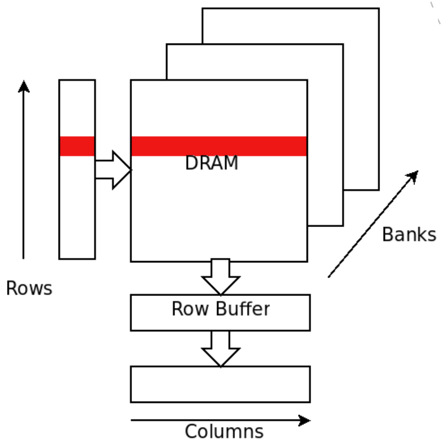
The 3D structure of modern DRAM



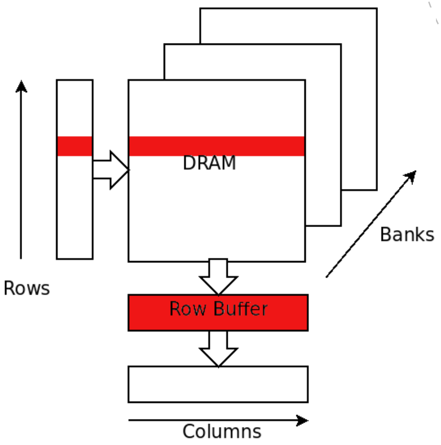
The 3D structure of modern DRAM



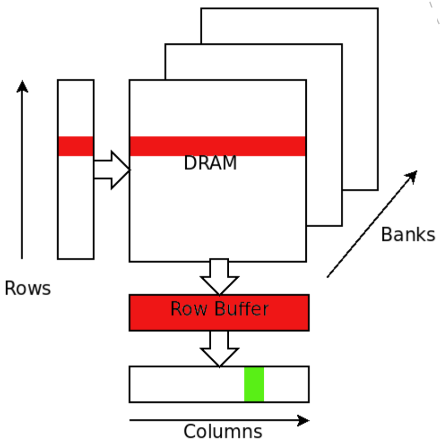
The 3D structure of modern DRAM



The 3D structure of modern DRAM



The 3D structure of modern DRAM



Example

Suppose a processor requires data at locations X_1 and X_2 that are located on the same page at times T_1 and T_2.
There are two separate outcomes:

Case 1:

The requests occur at roughly the same time:

1. Read 1 (T_1) enters the memory controller

NTNU
Innovation and Creativity

Case 1:

The requests occur at roughly the same time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened

Case 1:

The requests occur at roughly the same time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Read 2 (T_2) enters the memory controller

NTNU
Innovation and Creativity

Case 1:

The requests occur at roughly the same time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Read 2 (T_2) enters the memory controller
4. Data X_1 is returned from DRAM

NTNU
Innovation and Creativity

Case 1:

The requests occur at roughly the same time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Read 2 (T_2) enters the memory controller
4. Data X_1 is returned from DRAM
5. Data X_2 is returned from DRAM

NTNU
Innovation and Creativity

Case 1:

The requests occur at roughly the same time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Read 2 (T_2) enters the memory controller
4. Data X_1 is returned from DRAM
5. Data X_2 is returned from DRAM
6. The page is closed

NTNU
Innovation and Creativity

Case 1:

The requests occur at roughly the same time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Read 2 (T_2) enters the memory controller
4. Data X_1 is returned from DRAM
5. Data X_2 is returned from DRAM
6. The page is closed

Although there are two separate reads, the page is only opened once.

Case 2:

The requests are separated in time:

1. Read 1 (T_1) enters the memory controller

Case 2:

The requests are separated in time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened

Case 2:

The requests are separated in time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Data X_1 is returned from DRAM

Case 2:

The requests are separated in time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Data X_1 is returned from DRAM
4. The page is closed

NTNU
Innovation and Creativity

Case 2:

The requests are separated in time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Data X_1 is returned from DRAM
4. The page is closed
5. Read 2 (T_2) enters the memory controller

NTNU
Innovation and Creativity

Case 2:

The requests are separated in time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Data X_1 is returned from DRAM
4. The page is closed
5. Read 2 (T_2) enters the memory controller
6. The page is opened again

NTNU
Innovation and Creativity

Case 2:

The requests are separated in time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Data X_1 is returned from DRAM
4. The page is closed
5. Read 2 (T_2) enters the memory controller
6. The page is opened again
7. Data X_2 is returned from DRAM

NTNU
Innovation and Creativity

Case 2:

The requests are separated in time:

1. Read 1 (T_1) enters the memory controller
2. The page is opened
3. Data X_1 is returned from DRAM
4. The page is closed
5. Read 2 (T_2) enters the memory controller
6. The page is opened again
7. Data X_2 is returned from DRAM
8. The page is closed

The page is opened and closed twice. By prefetching X_2 we can increase performance by **reducing latency** and **increase memory throughput**.

NTNU
Innovation and Creativity

When does prefetching pay off?

If we prefetch when pages are open, we can reduce the number of times we open pages.
$\Rightarrow$ Lower latency and better utilization of the bus.
But prefetching isn't 100% accurate. What is the break-even point?

- — Minimum activate to precharge time
- — Multiple banks
- — Address bus contention vs data bus contention

$$Prefetching\ Accuracy * Cost\ of\ Prefetching < Cost\ of\ Single\ Read \tag{1}$$

NTNU
Innovation and Creativity

When does prefetching pay off?

If we prefetch when pages are open, we can reduce the number of times we open pages.
$\Rightarrow$ Lower latency and better utilization of the bus.
But prefetching isn't 100% accurate. What is the break-even point?

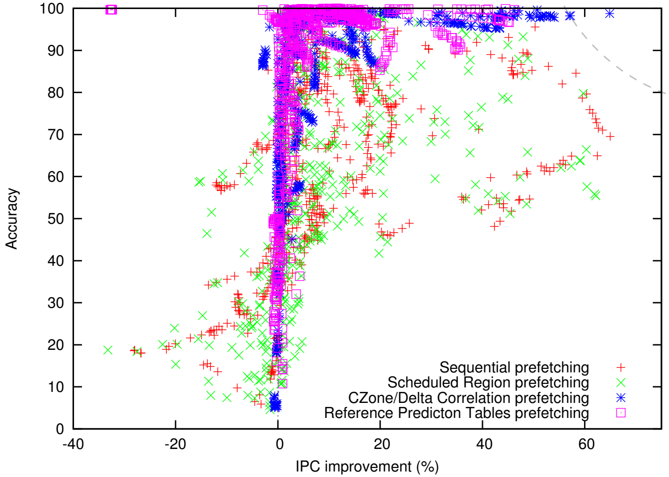
- — Minimum activate to precharge time
- — Multiple banks
- — Address bus contention vs data bus contention

$$\textit{Prefetching Accuracy} * \textit{Cost of Prefetching} < \textit{Cost of Single Read} \tag{1}$$

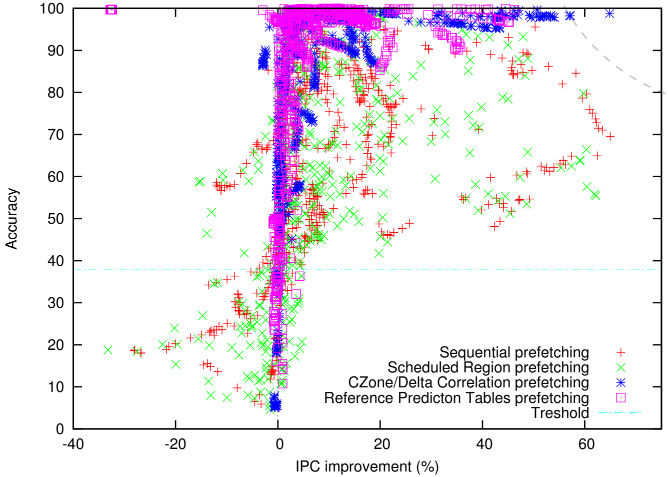
But - Cost is application dependant, and does not lend it self to easy analytical models.

NTNU
Innovation and Creativity

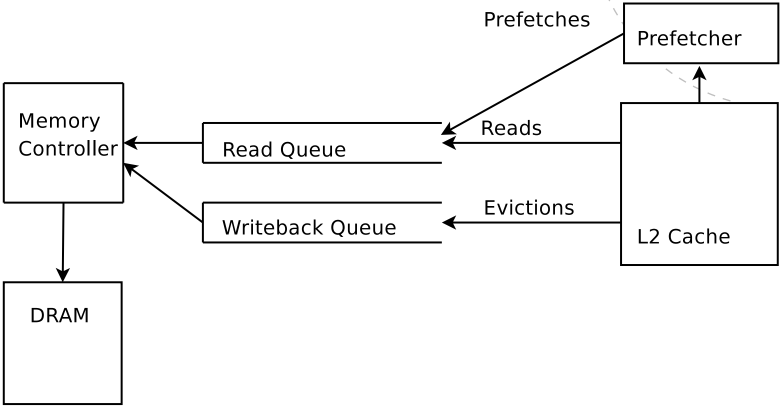
IPC improvement Vs Accuracy



IPC improvement Vs Accuracy



Issuing prefetches



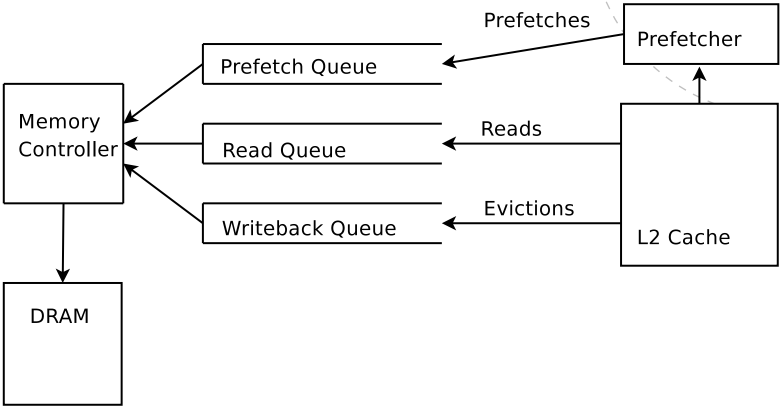
FR-FCFS scheduling

Priority rules:

1. Ready operations (Operations using open pages)
2. CAS (Column selection) over RAS (row selection) commands
3. Oldest request

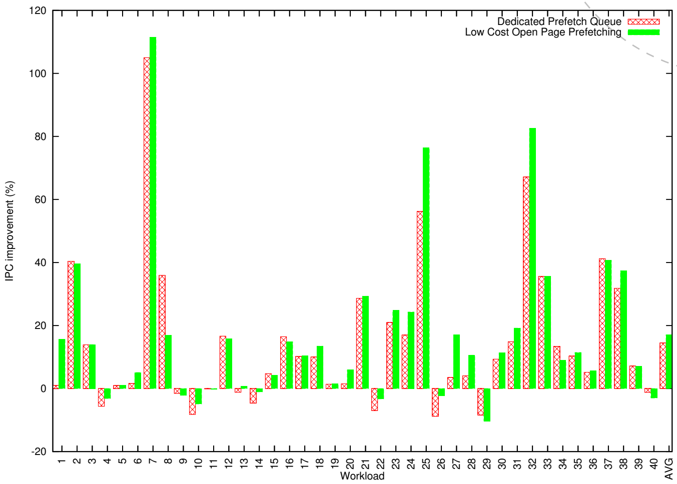
In addition, reads have a higher priority than writes

Issuing prefetches

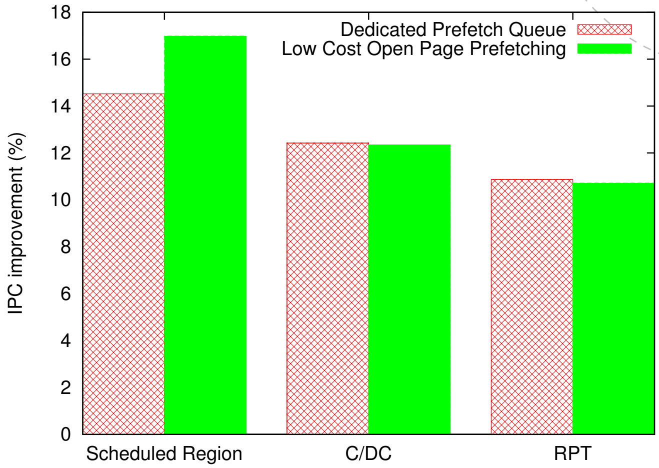


Only issue prefetches, if it hits an **open page** and **accuracy** is estimated to be **acceptable** or the bus is **idle.**

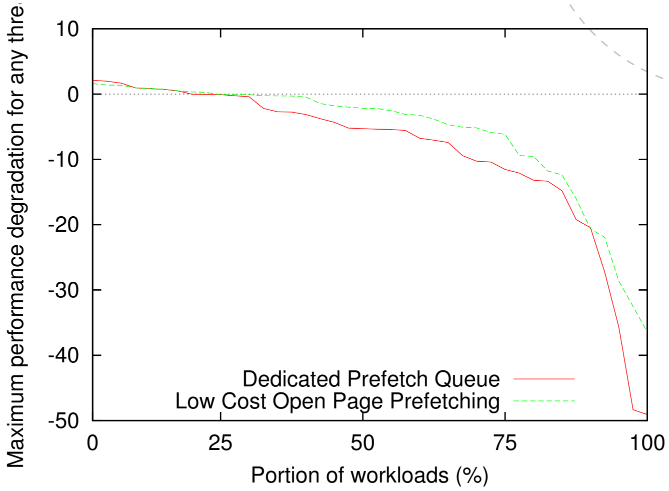
Speedup in IPC relative to no prefetching using a FR-FCFS memory controller



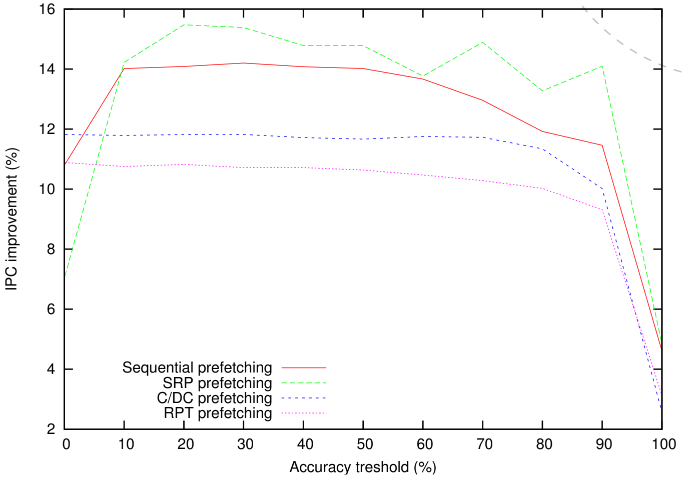
Average speedup in IPC relative to no prefetching.



Maximum IPC degradation for any thread as a function of workloads.



IPC improvement as a function of treshold



Conclusion

- By exploiting open pages it is possible to issue low-cost prefetches.
- Break-even at low accuracy
- Off chip bandwidth is a scarce resource - maximize its usage!

$\Rightarrow$ Coverage becomes more important than accuracy!

NTNU
Innovation and Creativity

Questions?

Thank you for your attention!

NTNU
Innovation and Creativity