Rules Fest 2010

International Conference on Reasoning Technologies

October 11-14 • San Jose, CA USA

Grindwork Corporation
Intelligent Automation

JBoss by Red Hat

IBM

Production Systems Technologies

morris technical solutions

VisionArts communications

The Gap Between the Virtual Machine and the Real Machine

Charles Forgy

Production Systems Tech

How to Improve Performance

Use better algorithms.

Use parallelism.

Make better use of the hardware.

Argument

The Java JVM is too far removed from the actual processor architecture to allow the hardware to be used efficiently.

Outline

- Modern processor architecture.
- Prefetching data.
- Making data structures cache-friendly.
- Using special instruction set features.

Moving a Block of Integers

```
MOV EAX, DWORD PTR [ESI]
MOV DWORD PTR [EDI], EAX
LEA ESI, [ESI+4]
LEA EDI, [EDI+4]
MOV EAX, DWORD PTR [ESI]
MOV DWORD PTR [EDI], EAX
LEA ESI, [ESI+4]
LEA EDI, [EDI+4]
MOV EAX, DWORD PTR [ESI]
. . .
```

Artificial Dependencies

Many instructions could proceed in parallel, except that they reuse the same registers.

Moving With Extra Registers

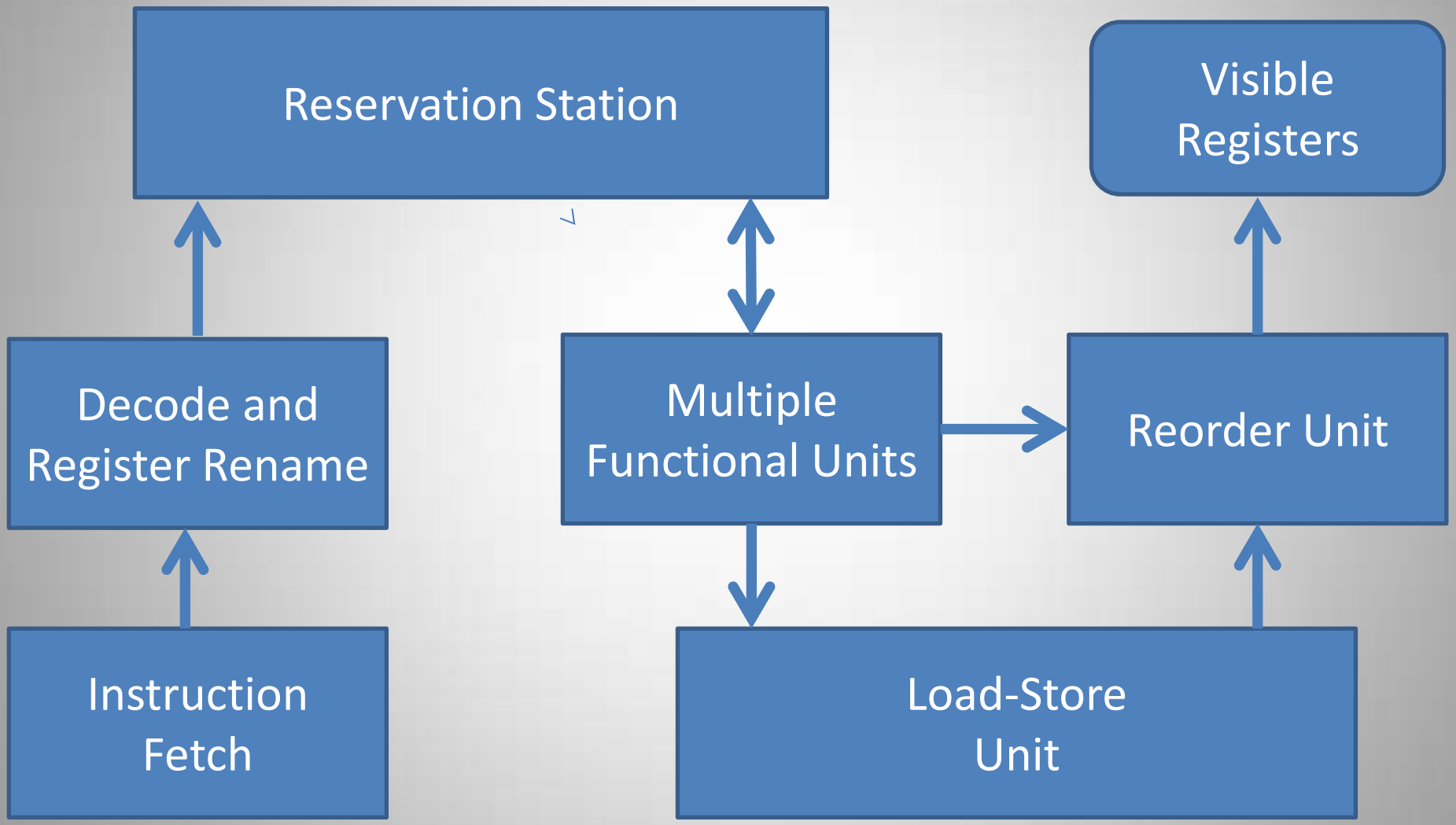
```
MOV TEMP1, DWORD PTR [ESI]
MOV DWORD PTR [EDI], TEMP1
LEA TEMP2, [ESI+4]
LEA TEMP3, [EDI+4]
MOV TEMP4, DWORD PTR [TEMP2]
MOV DWORD PTR [TEMP3], TEMP4
LEA TEMP5, [TEMP2+4]
LEA TEMP6, [TEMP3+4]
MOV TEMP7, DWORD PTR [TEMP5]
. . .
```

Modern Processors

Modern processors perform essentially this kind of renaming internally, allowing multiple instructions to be executed opportunistically (out-of-order execution).

Instructions are “retired” (i.e., their results are written back to the visible registers) in order.

Abstract Model



Pipelined Memory Access

Memory access is pipelined.

- So memory latency can be overlapped. For instance, the cost of two loads is much less than twice the cost of one load.

When a cache miss is processed, a full cache line is fetched.

- So, access to the rest of the data in that cache line is “free.”

Example: Nehalem

Reservation station buffers:	38
Reorder buffers:	128
Load buffers:	48
Store buffers:	32
Cache line:	64 bytes

(If Hyperthreading is enabled, the buffers are shared between two threads.)

Stalls

If the input data for an instruction are unavailable, or if there are not sufficient free resources, the instruction “stalls.”

Stalls and Out-of-Order Execution

Out-of-order execution allows the processor to perform useful work on other instructions while the stalled instruction waits.

Also, very importantly, it allows stalls to overlap in time.

Cost of Cache Misses

As a rough guide, a level 2 cache miss on a Core 2 processor takes around

- 165 cycles (desktop)
- 300 cycles (server)

Source: “Cycle Accounting Analysis on Intel® Core™2 Processors,” Dr. David Levinthal, Intel Corporation.

Outline

- Modern processor architecture.
- Prefetching data.
- Making data structures cache-friendly.
- Using special instruction set features.

Pointer-Chasing Problem

Many programs (including rule engines) have the characteristics of:

- Working on large numbers of objects, which are linked by pointers.
- Performing only a small amount of processing on each object after it is fetched.

Problem: when an object is fetched, it is unlikely to be in the cache.

Impact of Cache Misses

One simulation study of a hash join operation, showed the processor spending from 73% to 82% of its time stalled on data cache misses.

Source: “Improving Hash Join Performance through Prefetching” by Chen, Ailamaki, Gibbons, and Mowry.

Cache Misses and Rule Engines

Measurements using VTUNE indicate that join nodes in Rete II matchers spend more than 50% of their time stalled on data cache misses.

Abstract Example

```
while (needToProcess(curobj))
{
    process(curobj);
    curobj = curobj.successor;
}
```

Reducing the Impact of Cache Misses

If data is fetched ahead of time, the cache miss latency can overlap with useful work:

```
while (needToProcess(nextobj))
{
    curobj = nextobj;
    nextobj = curobj.successor;
    dummy = nextobj.successor;
    process(curobj);
}
```

Problem with this Solution

Because instructions are retired in-order, this creates an artificial dependency in the code.

Also, it is something of a hack.

A Better Solution

The X86 architecture has a group of instructions that are specifically for this situation: the PREFETCH group.

```
while (needToProcess(curobj))
{
    PREFETCH(curobj.successor);
    process(curobj);
    curobj = curobj.successor;
}
```

But…

There is no way to specify in Java that a PREFETCH should be performed.

Outline

- Modern processor architecture.
- Prefetching data.
- Making data structures cache-friendly.
- Using special instruction set features.

Reducing the Number of Cache Misses

When a processor fetches data into the cache, it fetches more than one word. (Typically 64 bytes.)

So, if related data can be grouped properly, one fetch can bring in the data for several operations.

In addition, if low-level access to memory is used, sometimes explicit pointers can be eliminated, and address arithmetic used instead.

Example: B+ Trees

In-memory databases frequently use some variant of B+ Trees to index the data.

Many researchers have worked on the trees cache-friendly. Among the techniques used are sizing the trees to the cache, aligning the fields on appropriate byte boundaries, using address arithmetic rather that pointers, etc.

JVM Problem

These kinds of techniques cannot be used in Java.

- Java puts objects where it decides to.
- Everything is an object, even arrays.

Example: B+ Node

```
class INode extends Node
{
    int[] keys;
    Node[] pointers;
}
```

Problems:

- The arrays keys and pointers are not in the node; they are separate objects. (More pointer-following means more cache misses.)
- The arrays may or may not be aligned properly for the cache.

Outline

- Modern processor architecture.
- Prefetching data.
- Making data structures cache-friendly.
- Using special instruction set features.

Advanced Instruction Example: SIMD Instructions

The current generation of processors offer SIMD instructions that can process 128 bits at a time. The next generation will support 256-bit SIMD instructions.

A number of researchers have shown that SIMD instructions can significantly speed up join, scan, aggregate, etc. operations.

SIMD and Java

SIMD intrinsics cannot be referred to in a Java program.

Questions?