DHBW
Duale Hochschule
Baden-Württemberg

Java

Memory Model

Die Quantenmechanik der Programmierung

Was ist ein Memory Model?

- Beschreibt die Interaktion von Threads und Speicher, v.a. gemeinsam genutzten Speicher
- Erlaubt dem Compiler wichtige Optimierungen
  - Compiler sind sonst ziemlich hilflos
- Stellt Regeln auf, an die sich Entwickler
  - Halten müssen, um korrekte Programme zu schreiben
  - Halten können, um Programme hardware-unabhängig schnell und korrekt zu schreiben

Moderne Performanz

- Performanz entsteht durch
  - Parallelisierung
  - Latenzverringerung
- Ohne Cores und Caches wären wir ~10 Jahre in der Vergangenheit
- Mit Cores und Caches:
  - Echte Nebenläufigkeit
  - Inkohärenter Gesamtzustand
- Wir brauchen Regeln für das Chaos!

Warum verwendet man Caches?

- Alte CPUs waren langsam
- Modernere CPUs wurden im Vergleich zum RAM so schnell, dass dieser gebremst hat
- Aktuelle CPUs haben so viele Cores, dass der RAM ein Flaschenhals wäre
- Caches bieten also einen sehr schnellen „Privat-RAM“ für jeden Core
- Dieser Privat-RAM ist aber unter Umständen nicht aktuell
  - Cache Coherence ist ein heißes Thema!

Latenzen im Vergleich

Vorgang	Latenz	Relative Latenz
Zugriff L1-Cache	0,5 ns	1 s
Falsche Branch Prediction	5 ns	10 s
Zugriff L2-Cache	7 ns	14 s
Mutex lock/unlock	25 ns	50 s
Zugriff RAM	100 ns	3m 20s
Sende 1 KB über Gbit-LAN	10.000 ns	5h 33m
Lese 1 MB sequentiell aus RAM	250.000 ns	5d 19h
Netzwerk-Roundtrip im LAN	500.000 ns (0,5 ms)	11d 14h
Lese 1 MB sequentiell von SSD	1.000.000 ns (1 ms)	23d 4h
Zugriffszeit Festplatte	10.000.000 ns (10 ms)	231d 12h
Lese 1 MB sequentiell von Spindle-Platte	20.000.000 ns (20 ms)	462d 23h
Netzwerk-Roundtrip US-Europa	150.000.000 ns (150 ms)	9y (= 3472d)

http://softwareyoga.com/latency-numbers-everyone-should-know/
http://blog.codinghorror.com/the-infinite-space-between-words/

Regeln für das Chaos

- Sind abhängig von der Hardware
  - Ganz schlechte Nachricht für Programmierer
- Können aber auch unabhängig davon sein
  - Die Laufzeitumgebung der Programmiersprache sichert gewisse Bedingungen zu
  - Wie diese Bedingungen auf der Hardware umgesetzt werden, ist nicht von Belang
  - Die Laufzeitumgebung darf tricksen, solange die Bedingungen erfüllt sind
- Wir verlassen uns auf ein Modell der Hardware

Ein Modell für den Speicherzugriff

- Memory Model sichert Bedingungen für Speicherzugriffe zu
  - Unabhängig von Cores and Caches!
- Alle normalen Programme basieren auf Speicherzugriffen
- Das Memory Model betrifft uns alle!
- Ohne dessen Beachtung trifft uns das Chaos
  - Works on my machine!
  - Cosmic ray errors

Wikipedia says it best

- In computing, a memory model describes the interactions of threads through memory and their shared use of the data.

Wikipedia says it best

A memory model allows a compiler to perform many important optimizations. Even simple compiler optimizations like loop fusion move statements in the program, which can influence the order of read and write operations of potentially shared variables. Changes in the ordering of reads and writes can cause race conditions. Without a memory model, a compiler is not allowed to apply such optimizations to multi-threaded programs in general, or only in special cases.

Wikipedia says it best

Modern programming languages like Java therefore implement a memory model. The memory model specifies synchronization barriers that are established via special, well-defined synchronization operations such as acquiring a lock by entering a synchronized block or method.

The memory model stipulates that changes to the values of shared variables only need to be made visible to other threads when such a synchronization barrier is reached. Moreover, the entire notion of a race condition is defined over the order of operations with respect to these memory barriers.

Wikipedia says it best

These semantics then give optimizing compilers a higher degree of freedom when applying optimizations: the compiler needs to make sure only that the values of (potentially shared) variables at synchronization barriers are guaranteed to be the same in both the optimized and unoptimized code. In particular, reordering statements in a block of code that contains no synchronization barrier is assumed to be safe by the compiler.

Wikipedia says it best

Most research in the area of memory models revolves around:

- Designing a memory model that allows a maximal degree of freedom for compiler optimizations while still giving sufficient guarantees about race-free and (perhaps more importantly) race-containing programs.
- Proving program optimizations that are correct with respect to such a memory model.

Wikipedia says it best

The Java Memory Model was the first attempt to provide a comprehensive threading memory model for a popular programming language.

After it was established that threads could not be implemented safely as a library without placing certain restrictions on the implementation and, in particular, that the C and C++ standards (C99 and C++03) lacked necessary restrictions, the C++ threading subcommittee set to work on suitable memory model; in 2005, they submitted C working document n1131 to get the C Committee on board with their efforts.

Wikipedia says it best

The final revision of the proposed memory model, C++ n2429, was accepted into the C++ draft standard at the October 2007 meeting in Kona. The memory model was then included in the next C++ and C standards, C++11 and C11.

Wikipedia says it best

The Java programming language and platform provide thread capabilities. Synchronization between threads is notoriously difficult for developers; this difficulty is compounded because Java applications can run on a wide range of processors and operating systems.

To be able to draw conclusions about a program's behavior, Java's designers decided they had to clearly define possible behaviors of all Java programs.

Wikipedia says it best

On modern platforms, code is frequently not executed in the order it was written. It is reordered by the compiler, the processor and the memory subsystem to achieve maximum performance.

On multiprocessor architectures, individual processors may have their own local caches that are out of sync with main memory.

It is generally undesirable to require threads to remain perfectly in sync with one another because this would be too costly from a performance point of view. This means that at any given time, different threads may see different values for the same shared data.