SAARLAND UNIVERSITY
COMPUTER SCIENCE

Prof. Dr. Jan Reineke
Dr. Andreas Abel
Lucas Biehl

Systemarchitektur SS 2021

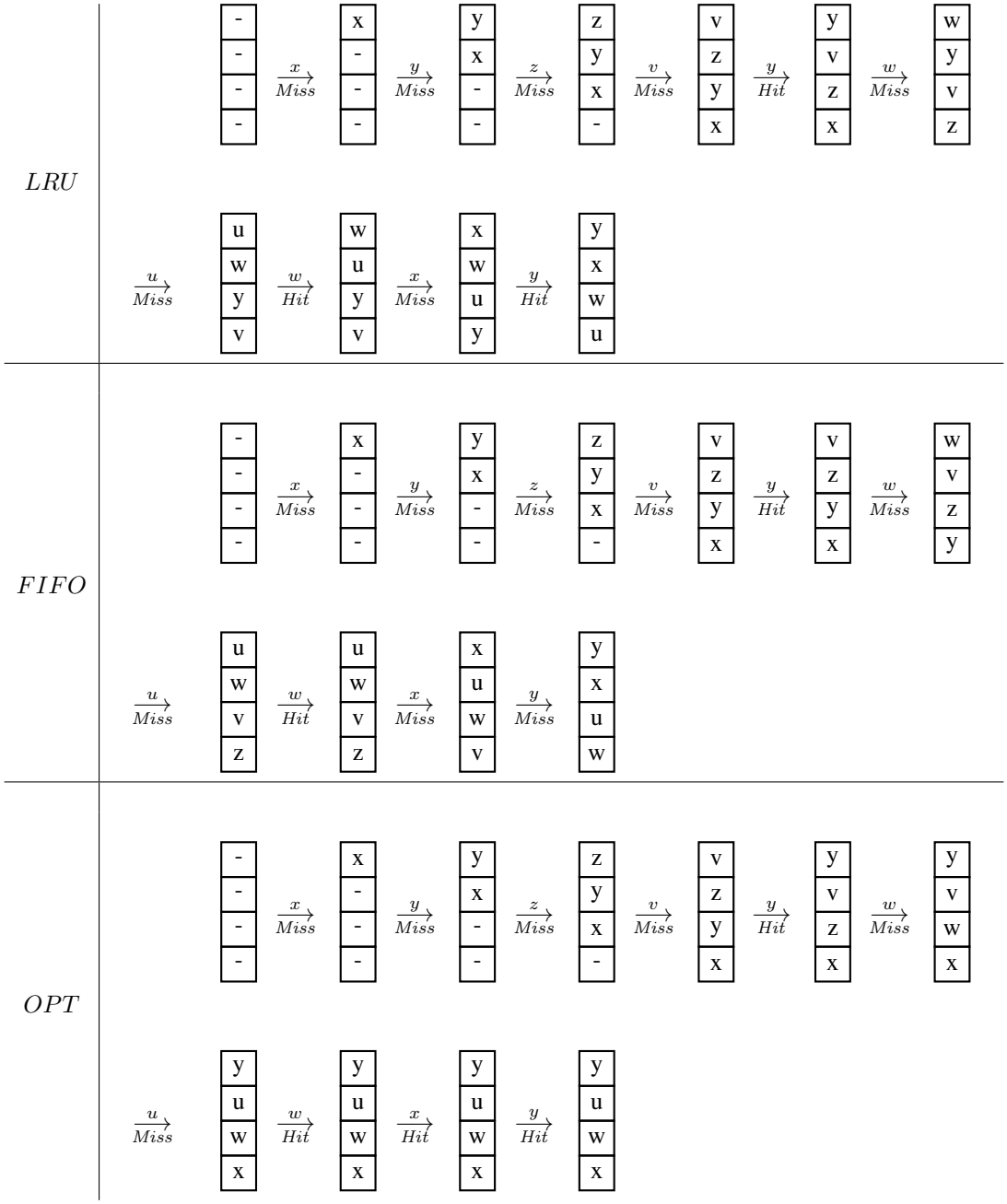
Lösungsskizze 10

Aufgabe 10.1: Speicherzugriffe

1. Das Programm allokiert in der äußeren Schleife Speicher für Arrays unterschiedlicher Größe, wobei die Größe in jedem Schritt verdoppelt wird. Zunächst wird das Array mit 0 initialisiert und die Variable lengthmod eingeführt. Dies ist ein String von Einsen und der höchste zugelassene Zugriff auf das Array. Durch das "& lengthmod" wird sichergestellt, dass alle Zugriffe auch im Bereich des Arrays bleiben. Da steps allerdings fest ist mit 64*1024*1024, ändert sich die Anzahl der Zugriffe nicht bei unterschiedlichen Iterationen, sondern bleibt konstant bei 64*1024*1024.

2. Es fällt auf, dass bei einer steigenden Arraysize auch die gemessene Zeit deutlich ansteigt. Dies lässt sich vor allem mit Caches erklären. Bei einem kleinen Array wird nur auf wenige unterschiedliche Speicherbereiche zugegriffen; diese passen alle in den Cache. Bei einem größeren Array wird auf eine größere Zahl von unterschiedlichen Speicherbereichen zugegriffen; dadurch passen irgendwann nicht mehr alle zugegriffenen Speicherbereiche in den Cache. Die "Sprünge" in den gemessenen Zeiten lassen sich durch die Cachehierarchie erklären: Am Anfang passen alle zugegriffenen Speicherbereiche in den L1-Cache; danach führen Speicherzugriffe zwar zu Misses im L1-Cache, aber immer noch zu Hits im L2-Cache, usw.

Aufgabe 10.2: Ersetzungsstrategien für Caches



SAARLAND UNIVERSITY
COMPUTER SCIENCE

Prof. Dr. Jan Reineke
Dr. Andreas Abel
Lucas Biehl

System Architecture SS 2021

Solution Sketch 10

Problem 10.1: Memory Accesses

1. The program allocates memory for arrays of different sizes in the outer loop, doubling the size in each iteration. First, the array is initialized to 0 and the variable lengthmod is introduced. This is a string of ones and the highest access allowed to the array. The "& lengthmod" ensures that all accesses are valid for the array. However, since steps is fixed at 64*1024*1024, the number of accesses does not change in different iterations, but remains constant at 64*1024*1024.

2. We notice that with an increasing array size, the measured time also increases significantly. This can be explained mainly by caches. With a small array, only a few different memory addresses are accessed; these all fit into the cache. With a larger array, a larger number of different memory addresses are accessed; as a result, at some point not all accessed addresses fit into the cache any more. The "jumps" in the measured times can be explained by the cache hierarchy: At the beginning, all accessed addresses fit into the L1 cache; after that, memory accesses lead to misses in the L1 cache, but still to hits in the L2 cache, etc.

Problem 10.2: Cache Replacement Policies