548

Lecture 8 - Caching

What is a cache?

- Fast memory that you insert between a slow memory and the processor in order to speed up access times
- Memoization: predecoded instructions
- Bandwidth amplification

What is locality?

- Information you likely want in the future is "near" information you have now
  - temporal locality
  - spatial locality
- Fundamental limit? A1, A2, A3, A1, A3, A4,...
  - Distance metric, timing metric
  - Idealized cache modal (OPT)
  - Probability modals

Why are there multiple cache levels?

- Larger cache is slower than a smaller cache
- Degrees of locality
  - Level 0 cache exploits most of the locality in the access steam => high hit ratio
  - Level N cache has less locality to use => lower hit ratio

Some Multiprocessor concerns

- Level 1 cache is private
  - There is a private state to the processor
  - Speed
- L2 or 3 is shared
  - Some amount of shared state
  - Utilization

Why are there separate I/D caches?

- Different usage
  - Different locality
- Close to where they are used
- Avoid conflicts

Your questions

- When is the trace cache fill not on the critical path?
- Does a trace cache require multiple branch predictions in a cycle?
- How many predictions are in flight in a modern processor?
- Associativity of modern caches?
- Multiprocessor issues for the zcache?
- What about storing DAGs instead of traces?
- Strategies besides BFS in zcache?
- Why does the trace cache store two next addresses?
- Do we really need a trace cache if our processors are only 4 wide?
- High accuracy multiple branch prediction?
- Compare zcache to deadblock prediction?
- Compiler assisted trace cache fill?
- How many branches should a trace contain?
-

Your questions

- Trace caches in modern processors?
- Why not go all the way to fully associative?
- Is die size = power drain in caches?