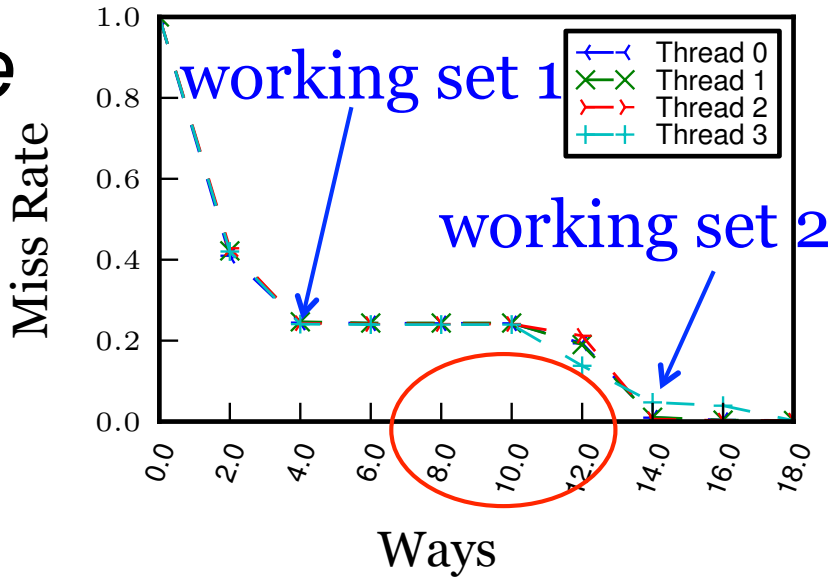
Imbalanced Cache Partitioning for Balanced Data-Parallel Programs

Abhisek Pan & Vijay S. Pai, Purdue University

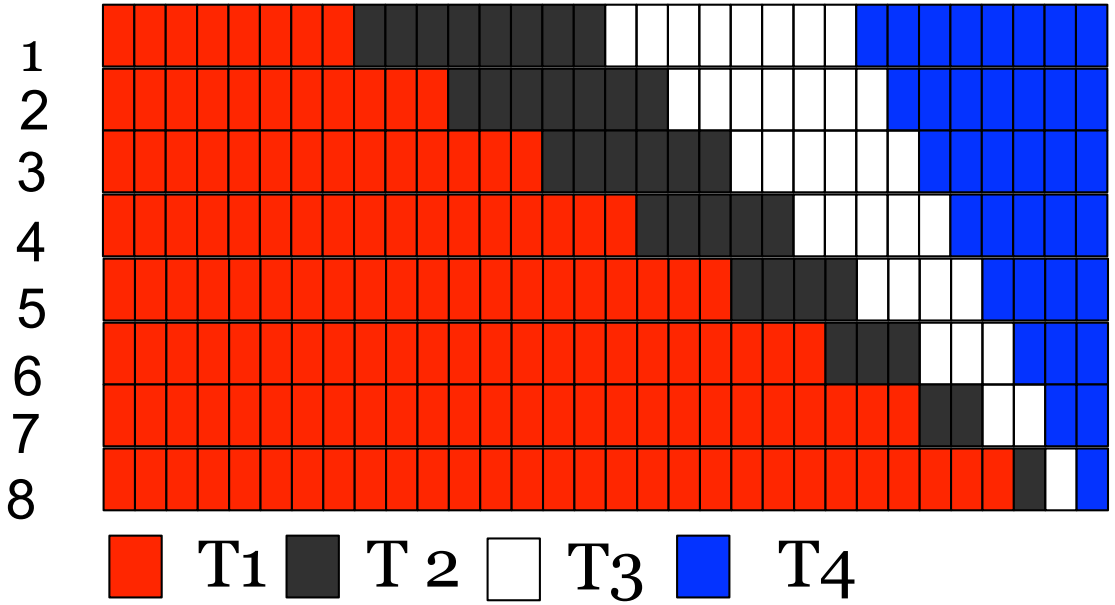
- *Balanced* data parallel programs need *imbalance* in allocation
- High imbalance helps both rewarded and penalized threads
- Prioritizing each thread in turn at a time ensures balanced progress



Inefficient Allocation!

Two-Stage Partitioning Method

- **Evaluation Stage**



- Divide cache sets into segments with different levels of imbalance
- Choose segment with lowest # of misses

- **Stable Stage**
  - Use chosen partition for the entire cache
  - Choose preferred thread in round-robin manner

Evaluation

- Partitioning beneficial only when per-thread working set between the default allocation and the cache capacity
- Improves upon the state-of-the-art runtime partitioning method in most such cases
  - 6% drop in execution time, 17 % drop in misses for 8 MB cache with 4 cores
- Limited overheads in space (way-partitioning, phase detection) and time (evaluation stage)