Data-Centric Execution of Speculative Parallel Programs

Mark C. Jeffrey, Suvinay Subramanian, Maleen Abeydeera, Joel Emer, Daniel Sanchez

{mcj, suvinay, maleen, emer, sanchez}@csail.mit.edu https://bit.ly/swarmhints

1. Motivation

Multicore systems must **exploit locality to scale**
- Locality-aware systems are non-speculative
- **Speculative systems** remain **locality-oblivious**

To scale, the system must map speculative tasks to cores to minimize data movement
Our solution: **spatial hints**
- A SW-given hint denotes a task's likely accesses
- HW maps equal-hint tasks to the same tile
- HW uses hints for locality-aware load balancing

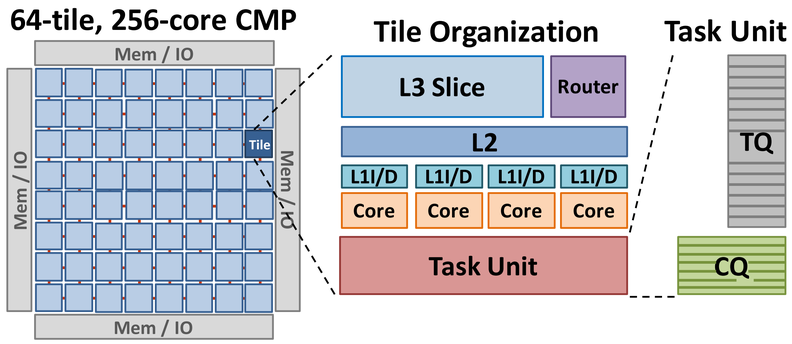
Locality-aware speculation is 3.3x faster at 256 cores
- **localizes accesses** of most data to one tile
- reduces network traffic
- reduces cycles wasted to mispeculation

2. Baseline: Swarm [MICRO'15]

- Programs comprise timestamped tasks
- Tasks can create children with >= timestamp (TS)
- Tasks appear to execute in timestamp order

```
swarm::enqueue(fptr, ts, args...)
```

Supports unordered (TM) and ordered (TLS) parallelism



Task Queues: Hold task descriptors
Commit Queues: Hold speculative state
Microarchitecture scales due to the following features:
- Low-overhead hardware task management
- Large task/commit queues
- Scalable speculation with selective aborts
- High-throughput ordered commits

Efficiently supports tiny speculative tasks

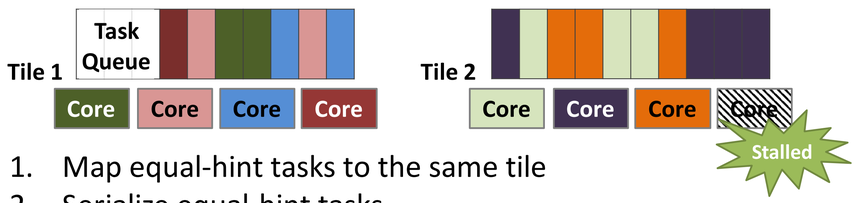
3. Spatial Task Mapping with Hints

A **spatial hint** is an integer value
- given at task creation time
- denotes the data likely to be accessed by the task

We extend the Swarm API with one field, *hint*:

```
swarm::enqueue(fptr, ts, hint, args...)
```

Hardware Mechanisms



1. Map equal-hint tasks to the same tile
2. Serialize equal-hint tasks

Often localizes data accesses within one tile
Avoids running conflicting tasks in parallel

4. Adding Hints to Benchmarks is Easy

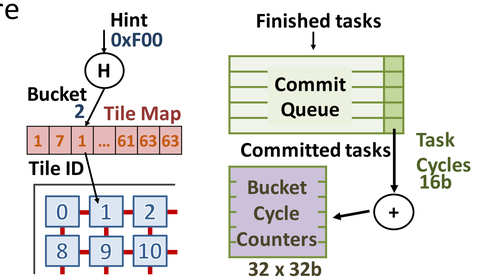
- Requires changing a few lines of code per task
- We observed a few common *patterns* emerge

```
void sssp(Timestamp dist, Vertex* v)
{
 if (v->distance == UNSET) {
  v->distance = dist;
  for (Vertex* n: v->neighbors)
   swarm::enqueue(sssp,
       /*TS=*/ dist + length(v,n),
       /*Hint=*/ cacheLine(n), n);
  }
}
```

Benchmark	Hint patterns
bfs, sssp, astar, color	Cache-line address
des,nocsim	Object ID
silo	(Table ID, primary key)
genome, kmeans	Multiple

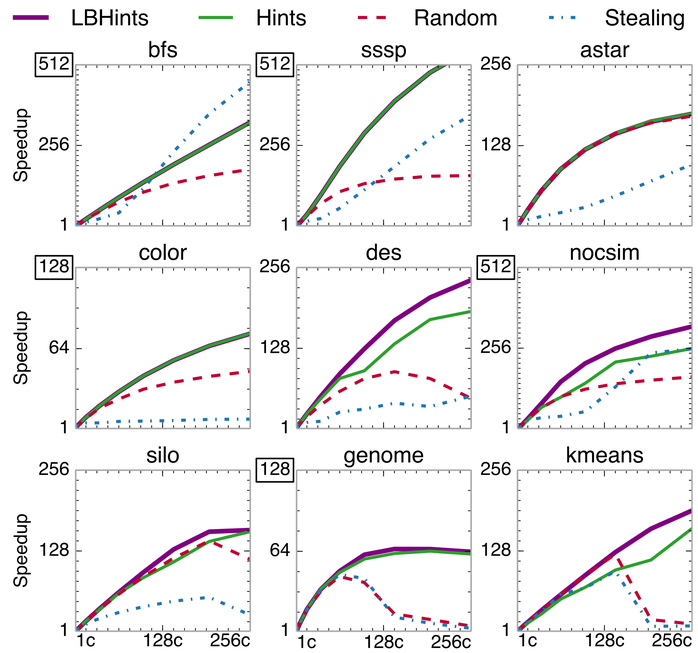
5. Load-Balancing Hints

- Statically mapping hints to tiles can cause hotspots
- Load is hard to measure
- Hint-to-tile mapping using buckets
- Balance committed cycles per unit time across tiles through periodic reconfiguration

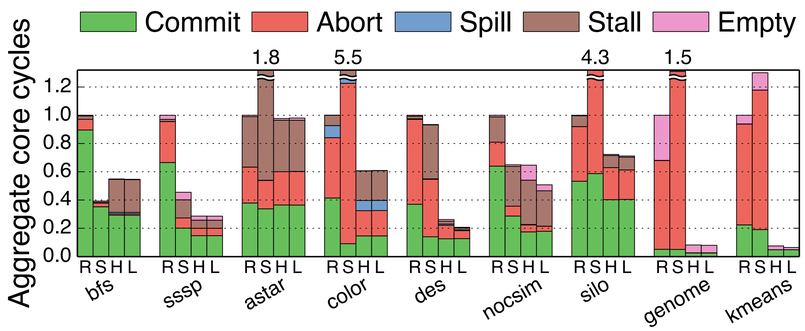


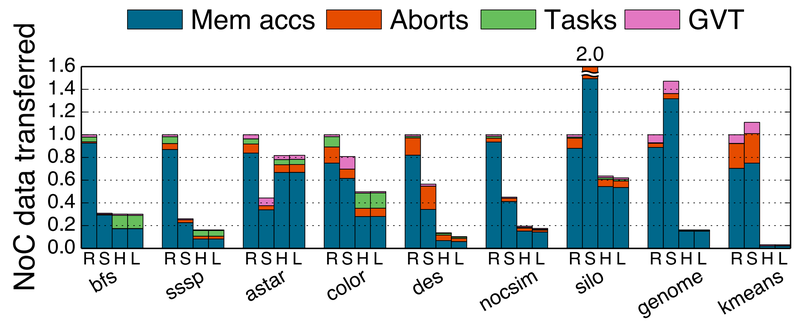
Locality-aware load balance leads to higher scalability

6. Evaluation



At 256 cores:
- Random speedups: 5—180x (gmean 58x)
- LBHints speedups: 63—561x (gmean 193x)





Aborted cycles 6.4x lower; network traffic 3.5x lower