Transactional Contention Management as a Non-Clairvoyant Scheduling Problem

Hagit Attiya Leah Epstein
Hadas Shachnai Tami Tamir

What is Transactional Synchronization?

- A systematic approach for implementing concurrent data structures
- A transaction aggregates a sequence of resource accesses to be executed atomically
  - Like in database systems
- A transaction ends either by committing
  - all of its updates take effect

  or by aborting
  - no update is effective

Optimism

"It is easier to apologize than to ask permission"

- Transactions (jobs) proceed until a conflict occurs
  - J_1 conflicts with an on-going J_2 if J_1 tries to write to a resource previously accessed by J_2

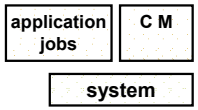
⇨ one job aborts or waits for the other to complete

  - If no conflict occurs, they run in parallel
- Most often, deadlock is impossible
  - Given a consistent rule for handling conflicts
- Progress is guaranteed when there is no contention
  - Obstruction-freedom or solo termination

Livelock: Will it Ever End?

"To apologize lays the foundation for a future offense"

- Contention manager mediates conflicts
  - decides which transaction aborts
- E.g., the greedy contention manager
  - Each job is assigned a unique timestamp reflecting jobs real-time order
  - Schedules a maximal set of non-conflicting jobs
  - If jobs J_1 and J_2 conflict, the job with the smaller timestamp aborts / waits



What Works and Why? In Practice

[Scherer and Scott, CSJP 04]

- Extensive testing
  - Backoff
  - Queuing
  - Aging
  - Randomized
  - Various priority
  - …
- None dominates on all benchmarks

What Works and Why? In Theory

[Guerraoui, Herlihy and Pochon, PODC 05]

Evaluate the throughput, measured by the makespan of a finite set of transactions

- Worst-case total time to complete all transactions

Relative to the makespan guaranteed by an optimal, off-line scheduler

- The competitive ratio

Prove that the greedy CM has $O(s^2)$ competitive ratio

- s is the number of resources

Non-Clairvoyant Scheduling

[Motwani, Phillips and Torng, SODA 1993]

- Scheduler does not know job characteristics a priori
  - Jobs arrive one by one, and their duration is unknown
- Evaluated in comparison with an optimal, clairvoyant scheduler
  - knows the set of jobs, their release times and duration

- Consider centralized algorithms for preemptive scheduling (jobs resume where they were stop)

Bounding the Greedy Contention Manager

THM: The greedy CM has $O(s)$ competitive ratio

- for mostly-write transactions

In fact, holds for every work conserving CM

- always lets a maximal set of non-conflicting transactions run

With the pending commit property

- at any time, some running transaction will execute uninterrupted until it commits

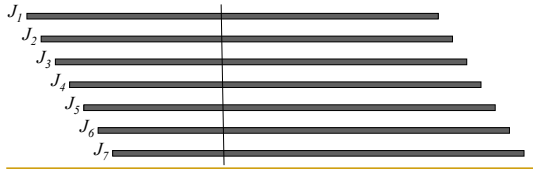
Upper Bound Idea: Optimal Makespan

Assume a job starts with a write access and does not idle

- If $> s$ jobs run concurrently, two access the same resource and at most one will complete

⇨ $\leq s$ jobs run concurrently under the optimal CM

⇨ The makespan of the optimal CM $\geq \frac{1}{s} \sum_{\text{all jobs } J_i} \text{duration of } J_i$



Upper Bound Idea: Makespan of Greedy

The makespan of the optimal CM $\geq \frac{1}{s} \sum_{\text{all jobs } J_i} \text{duration of } J_i$

- In the greedy CM, at least one of the jobs running at some point completes (does not aborts or waits)
  - By the pending commit property

⇨ The makespan of the greedy CM $\leq \sum_{\text{all jobs } J_i} \text{duration of } J_i$

☞ The ratio is at most s

☝ Full proof handles idle times & non-writing accesses

Optimality of the Greedy CM

"...greed... is good. Greed is right. Greed works."
(Gordon Gekko in the movie *Wall Street*)

THM: The competitive ratio of any work conserving CM is $\Omega(s)$

Even if the contention manager is:

- centralized
- does not guarantee the pending commit property

And transactions:

- have the same duration
- available at time 0

Lower Bound: First Requests

Work conserving CM must select an independent set of s/2 jobs

E.g., column 1.

	1	2	3	…	k
1	1	1	1	…	1
2	3	3	3	…	3
3	5	5	5	…	5
:	…	…	…	…	:
s/2	s-1	s-1	s-1	…	s-1
1	2	2	2	…	2
2	4	4	4	…	4
3	6	6	6	…	6
:	…	…	…	…	:
s/2	s	s	s	…	s

Lower Bound: Second Requests

Odd jobs all ask for 1

Even jobs all ask for 2

Only two jobs can complete

	1	2	3	…	k
1	1,2	1	1	…	1
2	3,2	3	3	…	3
3	5,2	5	5	…	5
:	…	…	…	…	:
s/2	s-1,2	s-1	s-1	…	s-1
1	2,1	2	2	…	2
2	4,1	4	4	…	4
3	6,1	6	6	…	6
:	…	…	…	…	:
s/2	s,1	s	s	…	s

Lower Bound: Next Set

Similarly…

Only two jobs can complete from the second independent set of s/2 jobs

	1	2	3	…	k
1	1,2	1,4	1	…	1
2	3,2	3,4	3	…	3
3	5,2	5,4	5	…	5
:	…	…	…	…	:
s/2	s-1,2	s-1,4	s-1	…	s-1
1	2,1	2,3	2	…	2
2	4,1	4,3	4	…	4
3	6,1	6,3	6	…	6
:	…	…	…	…	:
s/2	s,1	s,3	2k	…	s

Lower Bound: Repeat

In general, at most two jobs from each independent set can complete

	1	2	3	…	k
1	1,2	1,4	1,6	…	1,s
2	3,2	3,4	3,6	…	3,s
3	5,2	5,4	5,6	…	5,s
:	…	…	…	…	:
s/2	s-1,2	s-1,4	s-1,6	…	s-1,s
1	2,1	2,3	2,5	…	2,s-1
2	4,1	4,3	4,5	…	4,s-1
3	6,1	6,3	6,5	…	6,s-1
:	…	…	…	…	:
s/2	s,1	s,3	s,5	…	s,s-1

Makespan of Non-Clairvoyant Scheduler

After aborting, all jobs request the same resource

Makespan ≈ $\frac{s}{2} \cdot s \approx s^2$

	1	2	3	…	k
1	1,2	1,4	1,6	…	1,s
2	3,2	3,4	3,6	…	3,s
3	5,2	5,4	5,6	…	5,s
:	…	…	…	…	:
s/2	s-1,2	s-1,4	s-1,6	…	s-1,s
1	2,1	2,3	2,5	…	2,s-1
2	4,1	4,3	4,5	…	4,s-1
3	6,1	6,3	6,5	…	6,s-1
:	…	…	…	…	:
s/2	s,1	s,3	s,5	…	s,s-1

Makespan of Clairvoyant Scheduler

Schedule an extended diagonal together

s independent jobs complete

Makespan ≈ s

Competitive ratio ≈ s

	1	2	3	…	k
1	1,2	1,4	1,6	…	1,s
2	3,2	3,4	3,6	…	3,s
3	5,2	5,4	5,6	…	5,s
:	…	…	…	…	:
s/2	s-1,2	s-1,4	s-1,6	…	s-1,s
1	2,1	2,3	2,5	…	2,s-1
2	4,1	4,3	4,5	…	4,s-1
3	6,1	6,3	6,5	…	6,s-1
:	…	…	…	…	:
s/2	s,1	s,3	s,5	…	s,s-1

Wrap Up

- Other results:
  - Jobs that fail, randomized CM
  - Read the paper… ☺
- Lots of possibilities for future research:
  - Makespan measures system progress, what about each individual job?
    - E.g., penalty ratio
  - Mostly-reading / read-only transactions
  - Better use of randomization
    - It helps to have an estimate of how many transactions are competing for each resource