Conditional scheduling with varying deadlines

Ben Horowitz
bhorowit@cs.berkeley.edu

Which output: A or B?

- A, B each require 3 milliseconds to compute.
- In 4 milliseconds, one will need to be output.
- Decision about which to output in 2 milliseconds.
- Speculatively start to compute both!
Varying deadlines in Giotto

- I first saw this problem when working on precedence-constrained Giotto scheduling.
- A task is invoked; when the task’s actual deadline is depends on future mode changes.
- Following one set of mode changes, the task may have a 5ms deadline, say. Following another, the task may have a 10ms deadline.
Conditional scheduling problem

Finite state machine:
- Set Vertices of vertices.
- Set Edges of edges.
- For each edge $e$ a number $duration(e)$.
- Initial vertex $v_0$.

Workload:
- Set Tasks of tasks.
- For each $t \in Tasks$, a number $time(t)$.
- For each $v \in Vertices$, $release(v) \subseteq Tasks$.
- For each $v \in Vertices$, $due(v) \subseteq Tasks$.

Game: scheduler vs. environment

- Let $Runs = \text{set of paths of length } \geq 2$.
- Strategy is a function:
  $\sigma: Runs \times Tasks \rightarrow \mathbb{R}$
  $\sum_{t \in Tasks} \sigma((..., v_i, v_{i+1}), t) \leq duration(v_i, v_{i+1})$
When is a strategy winning?

- Consider arbitrary run, position $v_i$.
- Consider arbitrary task $t$ in $\text{release}(v_i)$.
- Find first subsequent $v_j$ at which $t$ is due.
- Let $n = \#$ of times $t$ is released at/after $v_i$, before $v_j$.
- Strategy must allocate $n \times \text{time}(t)$ between $v_i$ and $v_j$.

Related models

  - Tasks have fixed deadlines.
  - EDF is optimal.
  - Question is: how to determine if demand exceeds processing time?
- [Chakraborty, Erlebach, and Thiele, 2001]: Hardness results and approximation algorithm to answer above question.
- Our model generalizes these:
  - Deadlines of tasks vary.
  - Extends to include precedence constraints.
Algorithm for strategy synthesis

1) Construct linear constraints on strategy.
2) Solve using linear programming.
   - A feasible sol’n is a winning strategy.
   - No feasible sol’n: no winning strategy.

- Interval constraints:
  \[ \sigma((1, 2), A) + \sigma((1, 2), B) \leq 2 \]
  \[ \sigma((1, 2, 3), A) + \sigma((1, 2, 3), B) \leq 2 \]
  \[ ((1, 2, 4), A) + \sigma((1, 2, 4), B) \leq 2 \]

- Task constraints:
  \[ \sigma((1, 2), A) + \sigma((1, 2, 3), A) \geq 3 \]
  \[ \sigma((1, 2), B) + \sigma((1, 2, 4), B) \geq 3 \]

Discrete-time conditional scheduling

- What if the scheduler can make decisions only at a restricted set of points? Switching triggered by, e.g., a timer interrupt.
- For simplicity suppose this set is the integers.

- Theorem. Deciding whether a discrete-time problem has a winning strategy is NP-hard.
- Under a reasonable definition of lateness, there is no 2-approximation algorithm unless P=NP.
Tree scheduling vs. DAG scheduling

- Our linear programming algorithm is polynomial-time only if \((\text{Vertices}, \text{Edges})\) is a tree.
- What if the graph is a directed acyclic graph (DAG)?

\textbf{Theorem.} Determining whether a DAG problem has a winning strategy is \text{coNP}-hard.
- I believe this problem is inapproximable also...

Conclusion

- Introduced a novel model, conditional scheduling with varying deadlines.
- Developed polynomial-time schedule synthesis algorithm for tree-shaped problems.
- Discussed computational hardness of discrete-time and DAG problems.
References

- [Baruah 1998a]

- [Baruah1998b]

- [Chakraborty, Erlebach, and Thiele 2001]