Non-Clairvoyant Scheduling with Predictions

Zoya Svitkina

joint work with Manish Purohit and Ravi Kumar

TTIC workshop, July 31 2018
Algorithmic frameworks

- **Online algorithms**
  - Some problem parameters are unknown at the time of decisions
  - Competitive ratio: guarantee for worst-case input

- **Offline algorithms**
  - All parameters are known upfront
  - Exact or approximate -- typically better guarantee than corresp. online
Algorithmic frameworks

- **Online algorithms**
  - Some problem parameters are unknown at the time of decisions
  - Competitive ratio: guarantee for worst-case input

- **Offline algorithms**
  - All parameters are known upfront
  - Exact or approximate -- typically better guarantee than corresp. online

- **Algorithms with predictions**
  - Have predictions for parameters, but they are not necessarily correct
  - Competitive ratio as a function of error
    - high error: guarantee for worst case close to online
    - low error: better guarantee close to offline
Algorithms with predictions

Framework introduced in

- Revenue optimization with approximate bid predictions. 
  - Set reserve prices in an auction based on predicted bids

- Competitive caching with machine learned advice. 
  - Cache eviction strategy based on items' predicted next arrival
Motivation

- ML model trained on past instances that makes predictions based on observable features
  - **Scheduling**: user name, job name -> processing time
  - **Auctions**: bidder features, item features -> bid
  - **Caching**: past access pattern -> next time a page will be accessed
  - **Ski rental**: weather forecast -> number of skiing days
Goals

- No assumptions about error distribution
- Patterns change so prediction can be wrong
  - Want to have worst-case guarantees
  - Also want to derive benefit if the prediction happens to be good

$\eta$: measure of prediction error (problem-specific)
$c(\eta)$: competitive ratio as a function of prediction error

robustness = $\sup_{\eta} c(\eta)$  consistency = $c(0)$
(compare to online)  (compare to offline)
Non-clairvoyant scheduling with predictions

- 1 machine
- Minimize sum of completion times
- Preemption
- No release dates
- Processing times unknown, have predictions
- Assume shortest job $\geq 1$
Existing results without predictions

- Clairvoyant case (known processing times):
  - Shortest Job First is optimal

- Non-clairvoyant case:
  - Round-robin: Time-share between all unfinished jobs
  - 2-competitive, which is best possible
    [Motwani, Phillips, Torng 1994]
Algorithms with prediction

- **Round-robin**
  - Still 2-competitive
  - No benefit from predictions

- **Shortest Predicted Job First (SPJF)**
  - Optimal for perfect predictions
    (even for imperfect ones as long as they give the correct ordering)
  - Factor $n$ off in the worst case with bad predictions

- **Combine the two**
Analysis of Shortest Predicted Job First algorithm

- **Notation**
  - $x_j$ actual processing time of job $j$ (unknown to the algorithm)
  - $y_j$ predicted processing time of job $j$
  - $\eta_j = |x_j - y_j|$ prediction error of job $j$
  - $\eta = \sum_j \eta_j$ total L1 prediction error

- **Example**
  - Actual job sizes 1, 1, 1, 2. Predicted sizes 1, 1, 1, 1.
  - $OPT = 1 + 2 + 3 + 5 = 11$. $SPJF = 2 + 3 + 4 + 5 = 14$.
  - $\eta = 2 - 1 = 1$
  - $SPJF - OPT = 14 - 11 = 3 = \eta \times (n-1)$
## Analysis of Shortest Predicted Job First algorithm

\[
\text{ALG} = \sum_{j=1}^{n} x_j + \sum_{(i,j): i < j} (d(i,j) + d(j,i)) \quad \leftarrow \text{how much jobs delay each other}
\]

\[
= \sum_{j=1}^{n} x_j + \sum_{(i,j): i < j, y_i < y_j} x_i + \sum_{(i,j): i < j, y_i \geq y_j} x_j = \sum_{j=1}^{n} x_j + \sum_{(i,j): i < j, y_i \geq y_j} x_i + \sum_{(i,j): i < j, y_i \geq y_j} (x_j - x_i)
\]

\[
\leq \sum_{j=1}^{n} x_j + \sum_{(i,j): i < j} x_i + \sum_{(i,j): i < j, y_i \geq y_j} \eta_i + \eta_j = \text{OPT} + \sum_{(i,j): i < j, y_i \geq y_j} \eta_i + \eta_j \leq \text{OPT} + (n-1)\eta
\]

- Using assumption that all job sizes \( \geq 1 \)
  - \( \text{OPT} \geq \frac{n^2}{2} \)
  - competitive ratio of SPJF is at most \( 1 + \frac{2\eta}{n} \)
Combining two algorithms

- **Round-robin** with competitive ratio $2$, **SPJF** with $1 + \frac{2\eta}{n}$

- **Time-share** between the two
  - **SPJF** at a rate of $\lambda$ $\rightarrow$ competitive ratio $\frac{1 + \frac{2\eta}{n}}{\lambda}$
  - **Round-robin** at a rate of $1-\lambda$ $\rightarrow$ competitive ratio $\frac{2}{1-\lambda}$
Combining two algorithms

- **Round-robin** with competitive ratio 2, **SPJF** with $1 + 2\eta/n$
- **Time-share between the two**
  - **SPJF** at a rate of $\lambda$ $\rightarrow$ competitive ratio $\left(1 + \frac{2\eta}{n}\right) / \lambda$
  - **Round-robin** at a rate of $1-\lambda$ $\rightarrow$ competitive ratio $\frac{2}{1-\lambda}$
- **Algorithms running concurrently don't hurt each other**
- **Overall competitive ratio is the minimum of the two**
  - robustness $\frac{2}{1-\lambda}$, consistency $\frac{1}{\lambda}$
  - e.g. for $\lambda=3/4$, it is 8-robust and 4/3-consistent
  - beats 2 for good predictions
Open problems

- **Scheduling with predictions**
  - Release dates
  - Multiple machines

- **Extending other online algorithms to use predictions**
  - k-server
  - Metrical task system
  - Online matchings
  - ...