Non-Preemptive Flow with Rejections

x
Carnegie Mellon University

x

ICALP 2018

Job Scheduling Problem

- Jobs arrive online
  - Release time r_j, processing time p_j, weight w_j
- Single machine (this talk)
  - Finish time t_j: time job j finished by machine ($t_j \geq r_j + p_j$)
- Goal: minimize total flow time:

$$\sum_{\text{job } j} w_j \cdot (t_j - r_j) = \sum_{\text{job } j} w_j \cdot (\text{time } j \text{ stays in system})$$

- Online: do not know future jobs in advance
- Goal: be competitive with best offline
- Different policies: preemption, rejection

Online Is Hard

- Typical setting:
  - Preemption: (can switch jobs before finishing current job)
  - No rejection: must finish all jobs
- [BBCD 03] $n^{\Omega(1)}$ competitive lower bound (randomized)
- Simple lower bound (no preemption, deterministic)
  - $t = 0$: job A, $w_A = 1$, $p_A = 1$
  - As soon as algo processes A, release B with $w_B = M$, $p_B = 1/M$ (M is large)
  - ALG cannot preempt $A \implies ALG \geq M$
  - $OPT = O(1)$
- To be competitive (with offline), online needs different model

Online Models

- [KP 00] Speed augmentation model:
  - *ALG*'s machine is faster by $(1+\epsilon)$
    $\iff$ for *ALG*, $p_j \leftarrow p_j/(1+\epsilon)$
  - Only need to be competitive with offline without speed augmentation
  - [CGKM 09] poly$(1/\epsilon)$-competitive
    - Greedy algo
    - Proof: potential function
    - Also works for multiple, unrelated machines
  - [AGK 12] Proof via dual fitting
- [CDGK 18] Rejection model
  - No preemption: cannot switch jobs until current job done
  - Rejection: can reject ϵ fraction total weight of jobs
  - Competitive with offline preemption, no rejection
  - Intuition: rejecting "random" ϵ-fraction is like $(1+\epsilon)$-speed
- Main result: poly$(1/\epsilon)$-competitive in this model
- Follow dual fitting framework of [AGK 12]

Dual Fitting Framework [AGK 12]

- Dual fitting: systematic way to study online algos
- For job j:
  - $X := $ flow time of HDF policy for jobs 1 to $(j-1)$
  - $Y := $ flow time of HDF policy for jobs 1 to j
  - $\alpha_j := Y - X$
  - α_j is marginal increase in flow time from job j (for HDF)
  - Motivation: α_j represents a dual variable in LP relaxation
- $\sum_j \alpha_j$ is total flow time of HDF algo
- Thm (dual fitting): $OPT \geq \sum_j \alpha_j - ALG$
- Goal: $ALG \leq (1-\epsilon^3) \cdot \sum_j \alpha_j$

  $\implies 1/\epsilon^3$-competitive
- Intuition:
  1. Make ALG "almost-HDF": $ALG \leq (1+\epsilon^3) \cdot \sum_{\text{accepted } j} \alpha_j$
  2. Reject jobs with large α_j (can compute α_j online!)

     $$\implies \sum_{\text{accepted } j} \alpha_j \leq (1-2\epsilon^3) \cdot \sum_{\text{all } j} \alpha_j$$

(1) Make ALG "almost-HDF"

- Problem: online model is no preemption
- Fix: whenever preempt, reject instead
- Goal: maintain almost-HDF, but preempt only ϵ-fraction
- Algorithm outline:
  - For current job j, time t: consider lighter jobs that arrive later than j
    - $r_j \le r_{j'} \le t$ and $w_j' < w_j$
    - HDF should have preempted j for some j'
  - If total weight of such jobs exceeds w_j/ϵ, then preempt j
  - Charge preemption of j to the arrivals of these jobs
    $\implies$ preempt $\le \epsilon$-fraction of total weight
- Thm: if always process a current job j such that

$$\sum_{\substack{\text{unfinished}\\ \text{lighter jobs } j'}} w_{j'} \le w_j/\epsilon,$$

i.e., "almost-HDF", then $ALG \le \sum_{\text{acc } j} \alpha_j + \sum_j w_j p_j/\epsilon$

(2) Reject Jobs with Large α_j

- Goal: reject jobs such that $\sum_{\text{rej } j} \alpha_j \geq \text{poly}(\epsilon) \cdot \sum_j \alpha_j$ and $\sum_{\text{rej } j} w_j \leq \epsilon \cdot \sum_j w_j$
- Recall: can compute α_j values online
- Randomized achieves it in expectation
- Deterministic is header: most technically involved part
    - Bucket into geometric increasing classes based on α_j, w_j, p_j
    - Charge to class with largest geometric power

Open Questions

- Total flow time $\sum_{\text{job } j} w_j \cdot (t_j - r_j)$ is "ℓ_1-norm"
- Can we prove results for ℓ_p norm (constant p)?
- [AGK 12] Speed augmentation setting: same dual fitting technique solves ℓ_p norm
- What about ℓ_∞ norm? name for it?