Online optimization of max stretch on clusters

Erik Saule, Doruk Bozdag, Umit Catalyurek

Department of Biomedical Informatics, The Ohio State University
{esaule,bozdagd,umit}@bmi.osu.edu

Scheduling in Aussois 2010

Supported by the U.S. DOE, the U.S. National Science Foundation and the Ohio Supercomputing Center

Outline

The $P_m \mid r_i, p_i, online \mid \max S_i$ problem

Cluster scheduling

A cluster accepts jobs submitted over time. The jobs are independent and uses a single machine. No preemption is allowed.

The $P_m \mid r_i, p_i, online \mid \max S_i$ problem

Cluster scheduling

A cluster accepts jobs submitted over time. The jobs are independent and uses a single machine. No preemption is allowed.

The flow time index

The flow-time $F_i = C_i - r_i$ is the classical choice in cluster scheduling. But it is unfair for small jobs since a 10-hours job waiting for an hour as the same weight as a 10-minutes job waiting for an hour.

The $P_m \mid r_i, p_i, online \mid \max S_i$ problem

Cluster scheduling

A cluster accepts jobs submitted over time. The jobs are independent and uses a single machine. No preemption is allowed.

The flow time index

The flow-time $F_i = C_i - r_i$ is the classical choice in cluster scheduling. But it is unfair for small jobs since a 10-hours job waiting for an hour as the same weight as a 10-minutes job waiting for an hour.

The stretch performance index

The stretch $S_i = \frac{C_i - r_i}{p_i}$ normalizes flow-time and corrects the unfairness of flow-time. But makes the scheduling way more difficult.
Δ denotes the ratio $\frac{\max p_i}{\min p_i}$.

Previous results

Theorem

$1 \mid r_i, p_i \mid \max S_i$ *is NP-Complete in the strong sense [BCM98].*

Theorem

There is no $\Omega(n^{1-\epsilon})$ *approximation algorithm for* $1 \mid r_i, p_i \mid \max S_i$ *for constant* ϵ *unless* $P = NP$ *[BCM98].*

Theorem

$1 \mid r_i, p_i, online, pmpt \mid \max S_i$ *can not be approximated within* $\frac{\Delta^{\sqrt{2}-1}}{2}$ *[LSV08].*

Theorem

First-Come First-Serve is a Δ*-approximation of* $1 \mid r_i, p_i, online, pmpt \mid \max S_i$ *[LSV08].*

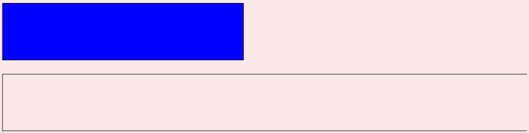
Outline of the Talk

The online one machine case

Theorem

$1 \mid r_i, p_i, online \mid \max S_i$ *can not be approximated within* $\frac{1+\Delta}{2}$.

Proof. (the adversary technique).



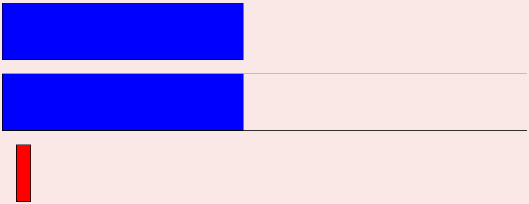
A large task enters in the system.

The online one machine case

Theorem

$1 \mid r_i, p_i, online \mid \max S_i$ *can not be approximated within* $\frac{1+\Delta}{2}$.

Proof. (the adversary technique).



If it is scheduled immediately, a small task is sent.

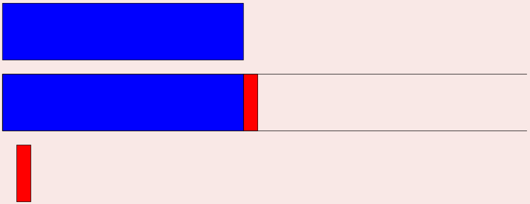
□

The online one machine case

Theorem

$1 \mid r_i, p_i, online \mid \max S_i$ *can not be approximated within* $\frac{1+\Delta}{2}$.

Proof. (the adversary technique).



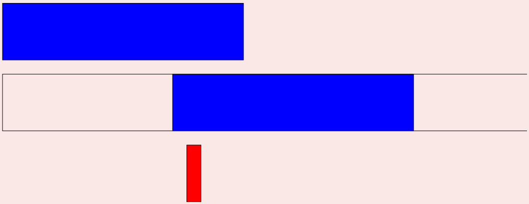
It suffers a large delay (and an unbounded stretch).

The online one machine case

Theorem

$1 \mid r_i, p_i, online \mid \max S_i$ *can not be approximated within* $\frac{1+\Delta}{2}$.

Proof. (the adversary technique).



If the large task is scheduled later, a small task is sent accordingly.

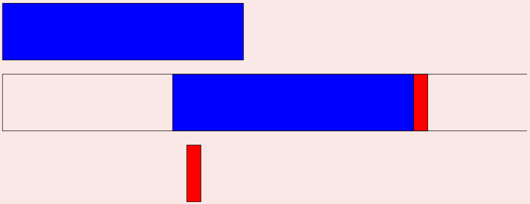
□

The online one machine case

Theorem

$1 \mid r_i, p_i, online \mid \max S_i$ *can not be approximated within* $\frac{1+\Delta}{2}$.

Proof. (the adversary technique).



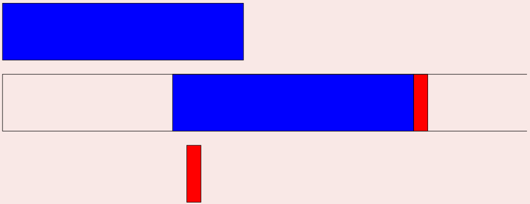
It suffers a large delay (and an unbounded stretch). □

The online one machine case

Theorem

$1 \mid r_i, p_i, online \mid \max S_i$ *can not be approximated within* $\frac{1+\Delta}{2}$.

Proof. (the adversary technique).



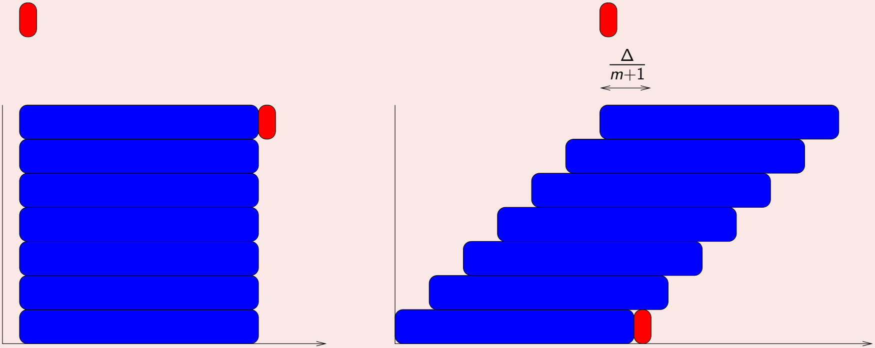
It suffers a large delay (and an unbounded stretch).
The optimal stretch is always less than 2 and the schedule stretch is always $1 + \Delta$. □

The online m machines case

Theorem

$P_m \mid r_i, p_i, online \mid \max S_i$ *can not be approximated within* $\frac{1+\frac{\Delta}{m+1}}{2}$.

Proof.



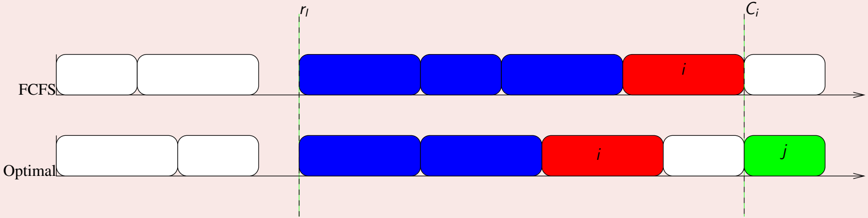
$\square$

First-Come First-Serve on one machine

Theorem ([LSV08])

FCFS is a Δ-approximation algorithm for $1|r_i, p_i, online|\max S_i$.

Proof.



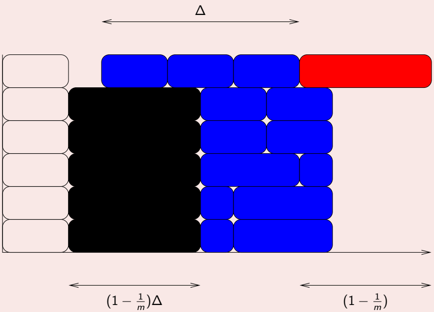
- If optimal has a better stretch for a task (red) then one of the task scheduled between r_l and C_i (the blue tasks) will complete after C_i (green).
- $S_j^* = \frac{C_j^* - r_j}{p_j} \geq \frac{C_i - r_i}{p_j} = \frac{C_i - r_i}{p_i}\frac{p_i}{p_j} = S_i \frac{p_i}{p_j}$

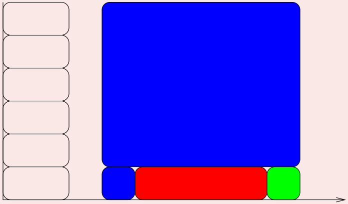
First-Come First-Serve on m machines

Theorem

FCFS is a $\Delta + (1 - \frac{1}{m})(\Delta + 1)$*-approximation algorithm for* $P_m|r_i, p_i, online|maxS_i$.

Proof.





Dual-approximation Algorithm for Stretch using EDF

DASEDF(S)

- It targets a maximum stretch S.
- Task i must complete before the deadline $D_i = r_i + p_i S$.
- Solves the deadline problem.

Dual-approximation Algorithm for Stretch using EDF

$DASEDF(S)$

- It targets a maximum stretch S.
- Task i must complete before the deadline $D_i = r_i + p_i S$.
- Solves the deadline problem.

Earliest Deadline First (EDF)

- Considers the tasks in order of non-decreasing deadline.
- Schedules the tasks as soon as possible.
- If a task starts after its deadline, declares the schedule infeasible.

Dual-approximation Algorithm for Stretch using EDF

DASEDF(S)

- It targets a maximum stretch S.
- Task i must complete before the deadline $D_i = r_i + p_i S$.
- Solves the deadline problem.

Earliest Deadline First (EDF)

- Considers the tasks in order of non-decreasing deadline.
- Schedules the tasks as soon as possible.
- If a task starts after its deadline, declares the schedule infeasible.

DASEDF

- Find the smallest maximum stretch S^* such that the deadline problem is feasible using a binary search.
- Use that schedule until an other tasks is released.

Dual-approximation Algorithm for Stretch using EDF

Lemma

If a schedule that completes each task i before D_i exists, then EDF *creates a schedule where each task i completes before $D_i + (1 - \frac{1}{m})p_i$.*

Dual-approximation Algorithm for Stretch using EDF

Lemma

If a schedule that completes each task i before D_i exists, then `EDF` *creates a schedule where each task i completes before $D_i + (1 - \frac{1}{m})p_i$.*

Theorem

`DASEDF`(S) *returns a solution with maximum stretch less than $S + 1 - \frac{1}{m}$ or ensures that no schedule with maximum stretch of S exists.*

Dual-approximation Algorithm for Stretch using EDF

Lemma

If a schedule that completes each task i before D_i exists, then EDF *creates a schedule where each task i completes before $D_i + (1 - \frac{1}{m})p_i$.*

Theorem

DASEDF(S) *returns a solution with maximum stretch less than $S + 1 - \frac{1}{m}$ or ensures that no schedule with maximum stretch of S exists.*

Theorem

When DASEDF is invoked, let S^* be the optimal stretch for the tasks in queue. DASEDF returns a solution of stretch $S < S^* + 1$.

Dual-approximation Algorithm for Stretch using EDF

Lemma

If a schedule that completes each task i before D_i exists, then EDF *creates a schedule where each task i completes before $D_i + (1 - \frac{1}{m})p_i$.*

Theorem

DASEDF(S) *returns a solution with maximum stretch less than $S + 1 - \frac{1}{m}$ or ensures that no schedule with maximum stretch of S exists.*

Theorem

When DASEDF is invoked, let S^* be the optimal stretch for the tasks in queue. DASEDF returns a solution of stretch $S < S^* + 1$.

Observation

DASEDF is not an online approximation algorithm. Its performance ratio is at least Δ.

Summing up

Positive Results

- FCFS is a Δ-approximation algorithm on 1 machine [LSV08].
- FCFS is a $(2\Delta + 1)$-approximation algorithm on m machines.
- DASEDF is a "local" 1-additive-approximation on m machines.

Summing up

Positive Results

- FCFS is a Δ-approximation algorithm on 1 machine [LSV08].
- FCFS is a $(2\Delta + 1)$-approximation algorithm on m machines.
- DASEDF is a "local" 1-additive-approximation on m machines.

Negative results

- No approximation better than $\frac{1+\Delta}{2}$ on 1 machine.
- No approximation better than $\frac{1+\frac{\Delta}{m+1}}{2}$ on m machines.

Summing up

Positive Results

- FCFS is a Δ-approximation algorithm on 1 machine [LSV08].
- FCFS is a $(2\Delta + 1)$-approximation algorithm on m machines.
- DASEDF is a "local" 1-additive-approximation on m machines.

Negative results

- No approximation better than $\frac{1+\Delta}{2}$ on 1 machine.
- No approximation better than $\frac{1+\frac{\Delta}{m+1}}{2}$ on m machines.

How to beat Δ ?

Outline of the Talk

Resource Augmentation : Faster machines

Theorem

$1 \mid r_i, p_i, online \mid \max S_i$ *can not be approximated within* $\frac{1+\Delta}{2\rho}$ *using a* ρ *faster machine.*

Resource Augmentation : More machines

Theorem

$1 \mid r_i, p_i, online \mid \max S_i$ can not be approximated within $\frac{\sqrt[\rho]{\Delta}}{\rho+2}$ using ρ machines.

Proof.

Send successively tasks of size $\Delta, \sqrt[\rho]{\Delta}^{\rho-1}, \dots, \sqrt[\rho]{\Delta}, 1$ □

Resource Augmentation : More machines

Theorem

$1 \mid r_i, p_i, online \mid \max S_i$ *can not be approximated within* $\frac{\sqrt[\rho]{\Delta}}{\rho+2}$ *using* ρ *machines.*

Proof.

Send successively tasks of size $\Delta, \sqrt[\rho]{\Delta}^{\rho-1}, \ldots, \sqrt[\rho]{\Delta}, 1$ □

The Split algorithm

The algorithm schedules the tasks of size between $\sqrt[\rho]{\Delta}^{j}$ and $\sqrt[\rho]{\Delta}^{j+1}$ on the jth copy of the system. The local schedule is done by a classical max stretch algorithm called $ALGO$.

Theorem

If $ALGO$ *is an* $f(\Delta)$*-approximation on* m *machines, Split is a* $f(\sqrt[\rho]{\Delta})$*-approximation algorithm using* ρm *machines.*

Reservation Scheme

In the real life, we don't have ρ times more machines.

Real Split(X, T)

- splits the cluster in two parts :
  - the main part of $m - X$ machines.
  - the auxilary part of X machines.
- when a task is released, the scheduling algorithm is run on the main part with the new task.
- if the maximum stretch of the main part is more than T
  - the scheduling algorithm is run on the auxilary part with the new task.
  - the task is allocated to the part that gives the best overall maximum stretch.

Outline of the Talk

Setting

Instance

300 machines; 20000 tasks

- processing time : uniformly distributed in $[1 : \Delta]$
- release time : exponential inter arrival time of parameter λ

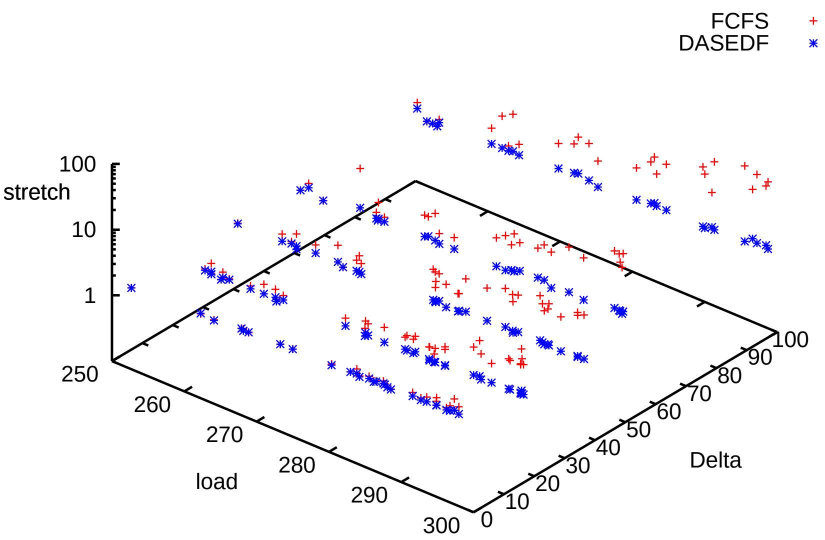
Parameters

- Δ between 5 and 100
- λ to have a load between 220 and 310. (load : average number of task in the system)
- (20 runs per parameter set)

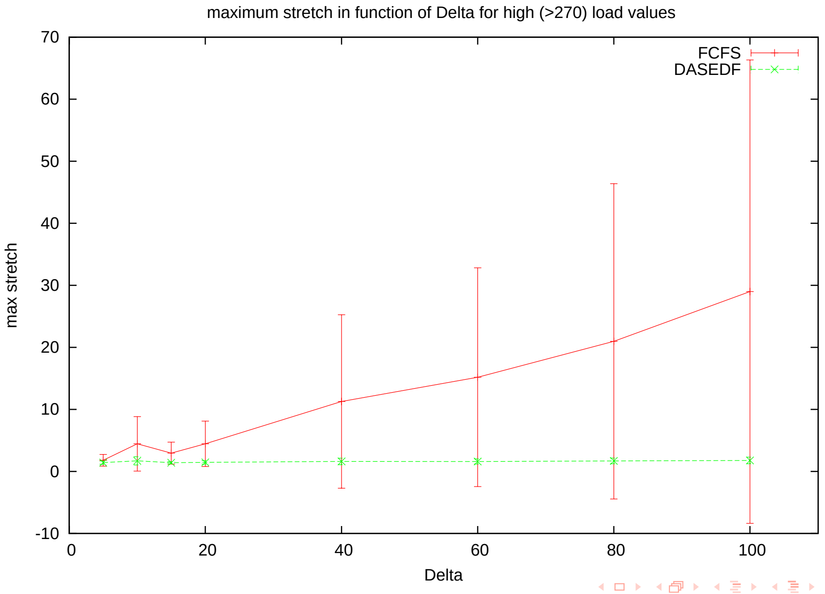
Reservation scheme

- X : between 1 and 30 machines in the auxilary part
- T : stretch threshold between 1.2 and 3 for DASEDF and from 1.2 to 10 for FCFS

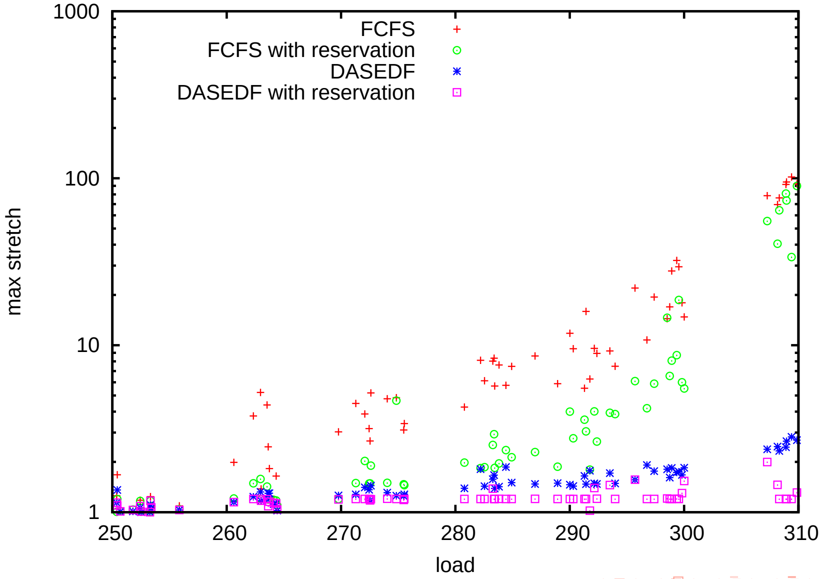
Result - overview



Result - in function of Δ



Results - in function of the load ($\Delta = 100$)



Outline of the Talk

Conclusion

Positive results

- FCFS is a $2\Delta + 1$-approximation on m machines.
- DASEDF is a "local" 1-additive-approximation on m machines.
- DASEDF beats FCFS in simulations.
- Split is a ρ-augmented. $f(\sqrt[\rho]{\Delta})$-approximation algorithm.

Conclusion

Positive results

- FCFS is a $2\Delta + 1$-approximation on m machines.
- DASEDF is a "local" 1-additive-approximation on m machines.
- DASEDF beats FCFS in simulations.
- Split is a ρ-augmented. $f(\sqrt[\rho]{\Delta})$-approximation algorithm.

Machine availability is the key

- Approximation lower bounds drops from $\frac{1+\Delta}{2}$ to $\frac{1+\frac{\Delta}{m+1}}{2}$ when the number of machine increases.
- Machine resource augmentation leads to $\frac{\sqrt[\rho]{\Delta}}{\rho+2}$ approximation lower bound which is better than having faster machine which leads to $\frac{1+\Delta}{2\rho}$.
- The reservation scheme helps in simulation.

What to do next ?

Improvement

- The performance ratio of FCFS on m machines is probably loose.
- Performance ratio of DASEDF ?
- Closing the gap between and $\frac{1+\frac{\Delta}{m+1}}{2}$ (LB) and $2\Delta + 1$ (FCFS). Using a staircase construction to maximize availability perhaps ?
- Select the right parameters for the reservation scheme.

On other models

- Rigid tasks.
- Moldable tasks ("scheduling in Knoxville" 2009 and JSSPP 2010).
- Average stretch.

Thank you

More information

contact : esaule@bmi.osu.edu

visit: http://bmi.osu.edu/hpc/

Research at HPC lab is funded by

M. A. Bender, S. Chakrabarti, and S. Muthukrishnan.
Flow and stretch metrics for scheduling continuous job streams.
In *Proceedings of the 9th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 270–279, 1998.

Arnaud Legrand, Alan Su, and Frédéric Vivien.
Minimizing the stretch when scheduling flows of divisible requests.
Journal of Scheduling, 2008.