Variations of Conservative backfilling to improve fairness

Presented by
Lizzy Shakman & Oscar Gardella

Introduction

Introduction

Use PC and DC with FCFS priority order to improve fairness

Formalize “fairness” with two ideas:

1. Jobs should not be delayed by later-arriving jobs.
2. Jobs should get a proportional share of system resources.

Definitions of fairness

Fair start time

Jobs must *not* be delayed by other jobs backfilling (benign backfilling)

Strict Fair Start Time (Strict FST): The starting time a job gets if no jobs arrived after it.

Issue is inaccurate estimates can create strict FSTs that are not feasible.

Relaxed Fair Start Time (Relaxed FST): The starting time a job gets if no jobs arrived after it, *but* the job is also not allowed to backfill.

Avoids infeasible fair start time sets

Fair start time

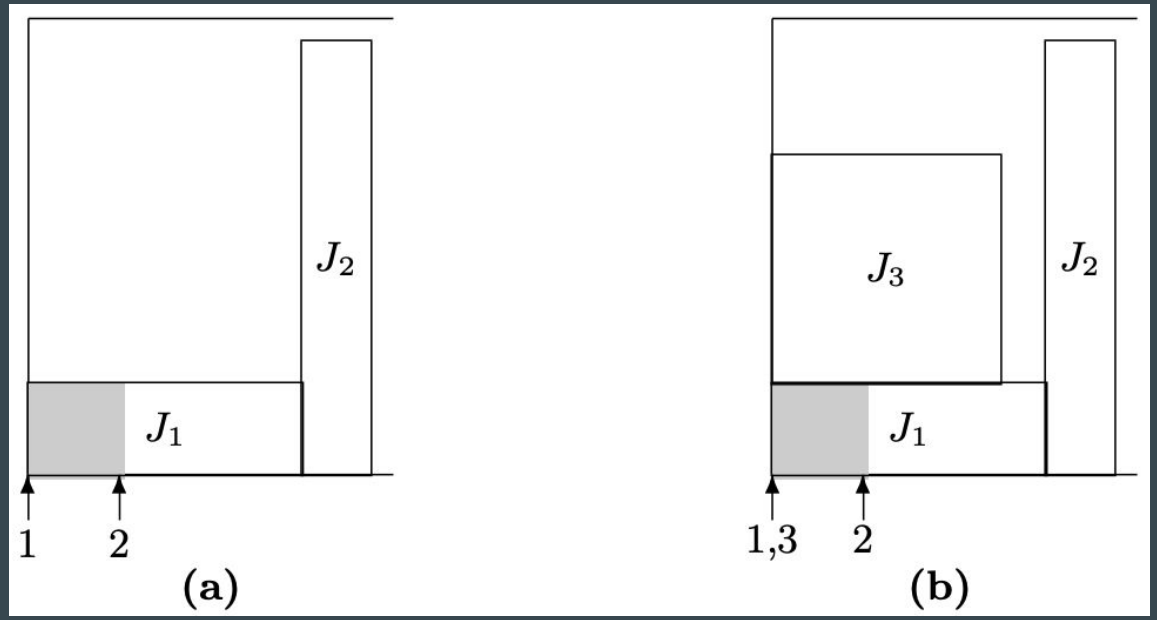


Fig. 2. Instance where strict fair start times are infeasible. Anticipated schedule (a) before and (b) after arrival of job J3. The shaded region and the block of job J1 shows its actual length and estimated time respectively. Labels below the figures indicate the strict fair start time of each job.

Resource equality

Each *active* job deserves an equal share of system resources.

Two subtleties in dividing system resources equally:

1. No job's fair share of the processors may exceed the number it wants to use.
2. Fair shares are based on the number of processors in use rather than the total system size.

Unweighted fair share

$$\int_{a_i}^{c_i} \min\left\{\frac{\text{util}(t)}{\text{active}(t)}, p_i\right\} dt$$

Weighted fair share

$$\int_{a_i}^{c_i} \min\left\{\frac{p_i}{\sum_{J_j \text{ is active}} p_j} \cdot \text{util}(t), p_i\right\} dt$$

*At each job arrival or completion, we increase the fair share values to reflect the contribution since the last arrival or completion event.

Experimental results

Traces

Name	Full file name	# jobs
ANL-Intrepid	ANL-Intrepid-2009-1.swf	68,936
CTC-SP2	CTC-SP2-1996-2.1-cln.swf	77,222
DAS2-fs1	DAS2-fs1-2003-1.swf	39,348
DAS2-fs2	DAS2-fs2-2003-1.swf	65,380
DAS2-fs3	DAS2-fs3-2003-1.swf	66,099
DAS2-fs4	DAS2-fs4-2003-1.swf	32,952
KTH-SP2	KTH-SP2-1996-2.swf	28,489
LANL-CM5	LANL-CM5-1994-3.1-cln.swf	122,057
LLNL-Atlas	LLNL-Atlas-2006-1.1-cln.swf	38,143
LLNL-Thunder	LLNL-Thunder-2007-1.1-cln.swf	118,754
LPC-EGEE	LPC-EGEE-2004-1.2-cln.swf	220,679
SDSC-BLUE	SDSC-BLUE-2000-3.1-cln.swf	223,669
SDSC-DS	SDSC-DS-2004-1.swf	85,006
SDSC-SP2	SDSC-SP2-1998-3.1-cln.swf	54,041

Fig. 3. Traces used in simulations

Fair start time: DC

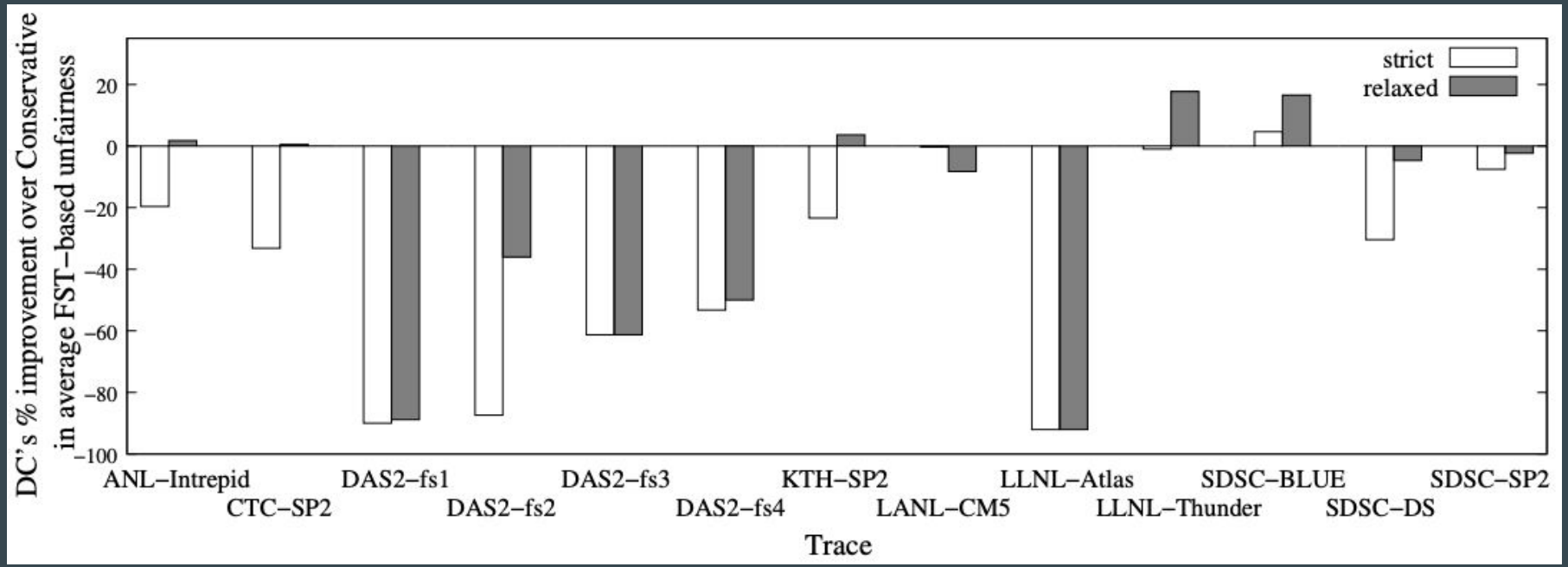


Fig. 4. Improvement in average strict and relaxed unfairness of DC over Conservative. Not shown is LPC-EGEE for which all algorithms except DC produce average unfairness of 0; DC gives unfairness ~ 0.102 for both ($-\infty$ improvement).

Fair start time: DC

Job	Arrival time	# processors	Processing time	User estimate
J_1	0	90	100	200
J_2	1	45	100	200
J_3	2	40	95	200
J_4	3	90	100	200
J_5	4	45	100	200

Fair start time: DC

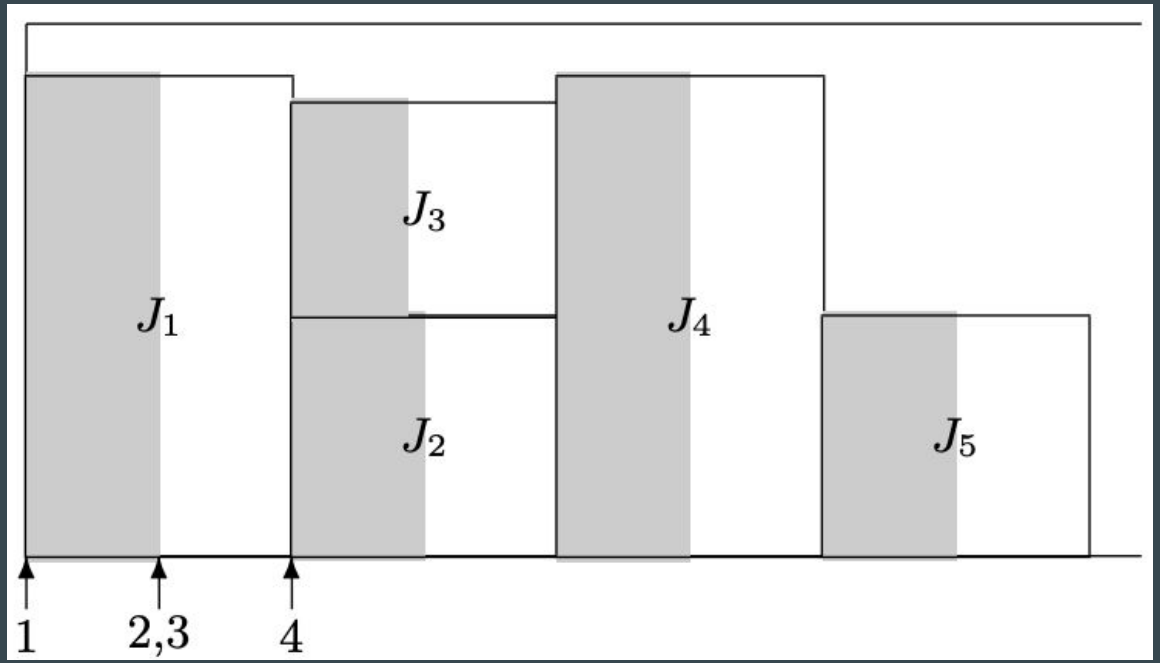


Fig. 5. Profile after all jobs arrive in instance showing DC's potential for unfairness. The shaded region and the block of each job show its actual length and estimated time respectively . Labels below the figures indicate the strict fair start time of each job.

Fair start time: PC

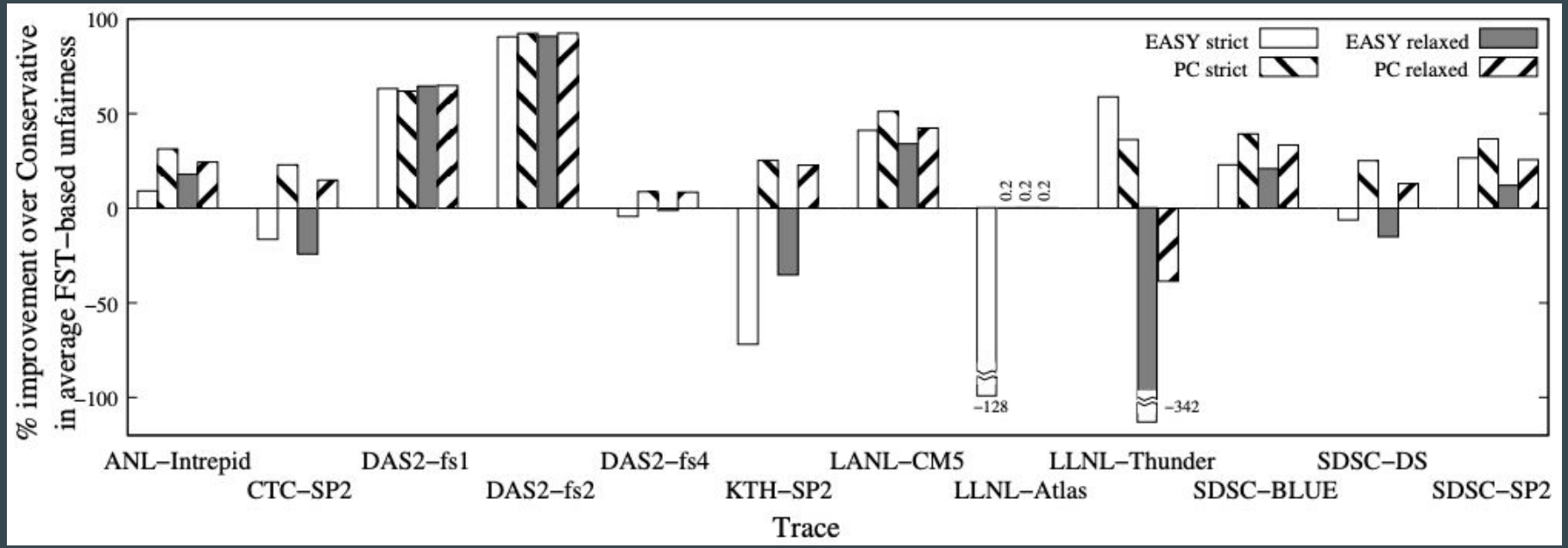
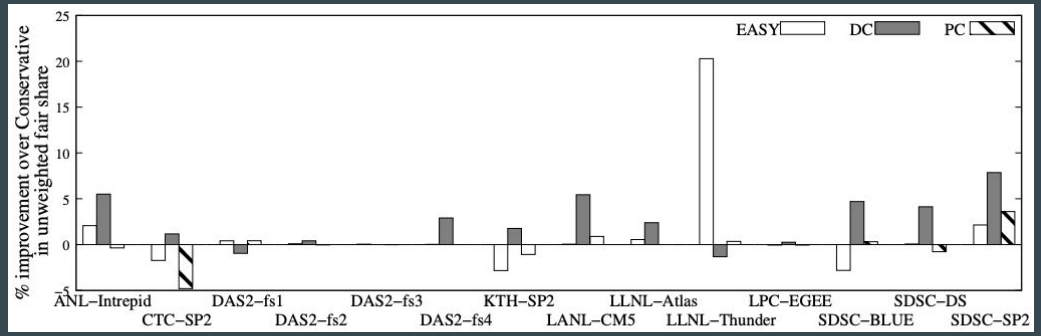


Fig. 6. Improvement in average strict and relaxed unfairness of EASY and PC over Conservative. Not shown are LPC-EGEE (all algorithms except DC produce unfairness of 0) and DAS2-fs3 for which PC gives no improvement and EASY produces “improvements” of −3, 010% and −2, 784% for strict and relaxed unfairness respectively.

Unweighted fair share



Weighted fair share

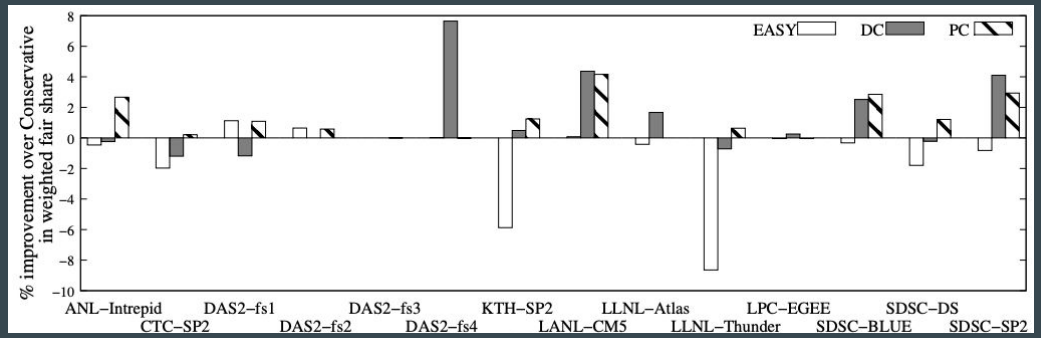


Fig. 7. Improvement in unweighted (top) and weighted (bottom) unfairness (fair share approach) over Conservative.

Response time

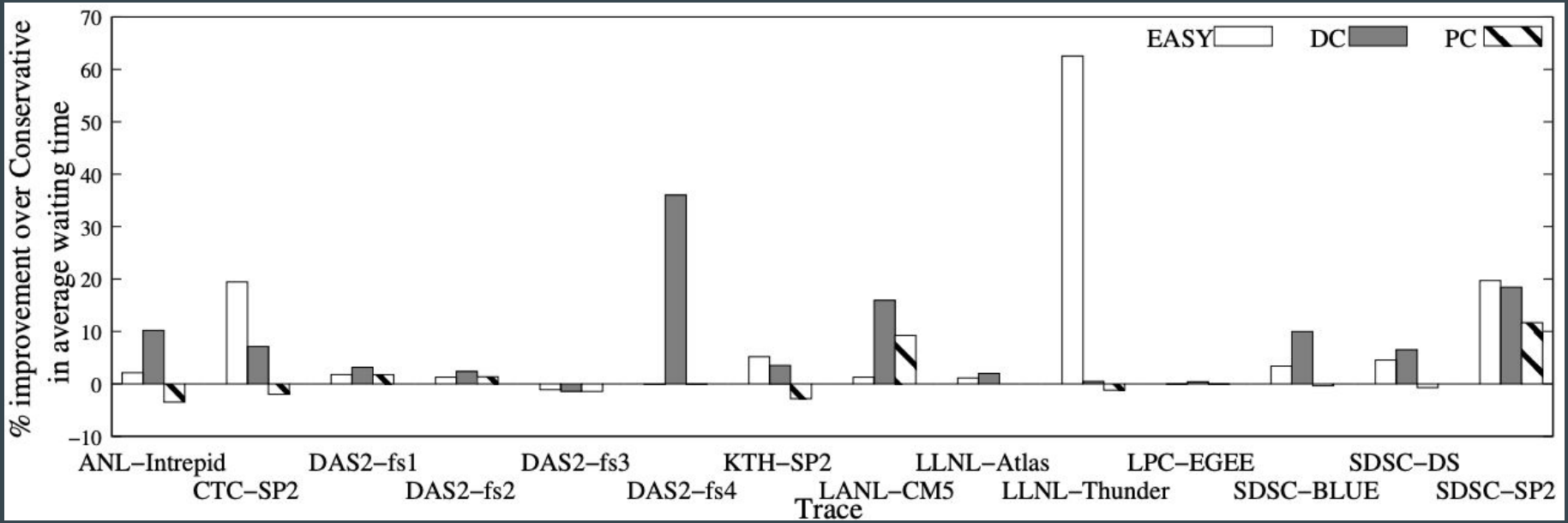


Fig. 8. Improvement in average waiting time over Conservative.

Final thoughts

Discussion

- Does fairness even really matter?
- Is it worth the small cost to runtime?
- Studying better time estimation tools should help improve backfilling
  - Both for speed and fairness
- This paper is important because it examines fairness, not just speed