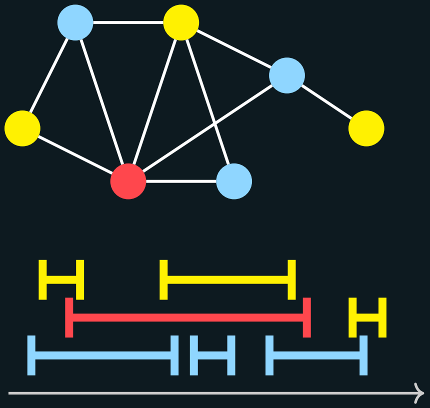
Fair Repetitive Interval Scheduling

Klaus Heeger, Danny Hermelin, **Yuval Itzhaki**, Hendrik Molter, Dvir Shabtay

Motivation

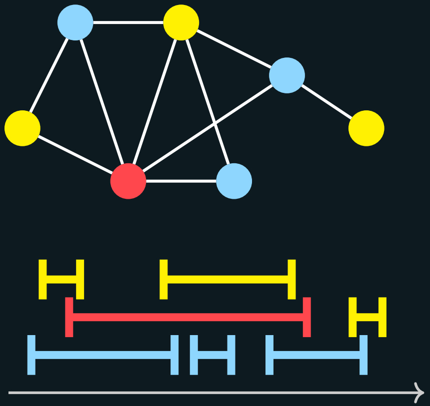
Suppose we are serving the same clients every day.



Motivation

Suppose we are serving the same clients every day.

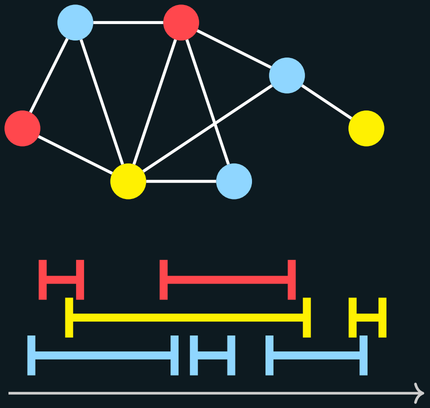
Suppose some machines are *better*.



Motivation

Suppose we are serving the same clients every day.

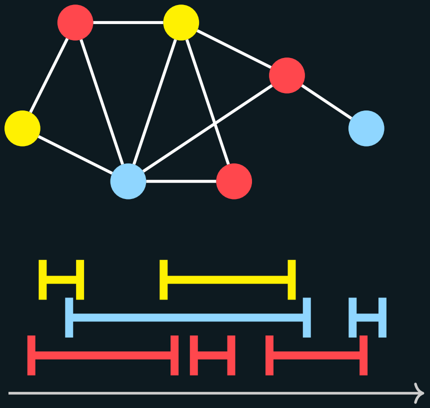
Suppose some machines are *better*.



Motivation

Suppose we are serving the same clients every day.

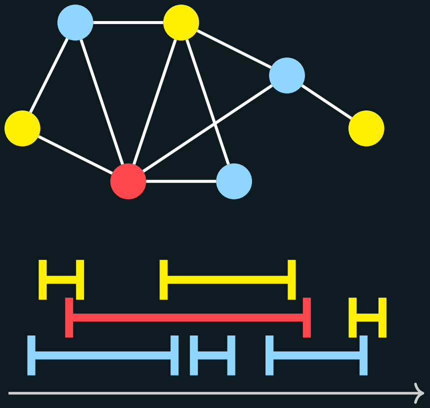
Suppose some machines are *better*.



Motivation

Suppose we are serving the same clients every day.

Suppose some machines are *better*.



Related Work

Related Work

- Recently established framework [HMN$^+$25]


ELSEVIER

European Journal of Operational Research
Volume 323, Issue 3, 16 June 2025, Pages 724-738

Discrete Optimization

Fairness in repetitive scheduling ☆

Danny Hermelin [a 1], Hendrik Molter [b 2], Rolf Niedermeier [c], Michael Pinedo [d], Dvir Shabtay [a 1]

Show more

Add to Mendeley Share Cite

https://doi.org/10.1016/j.ejor.2024.12.052 ↗ Get rights and content ↗

Related Work

- Recently established framework [HMN+25]
- Studied objectives:

ELSEVIER

European Journal of Operational Research
Volume 323, Issue 3, 16 June 2025, Pages 724-738

Discrete Optimization

Fairness in repetitive scheduling ☆

Danny Hermelin [a 1], Hendrik Molter [b 2], Rolf Niedermeier [c], Michael Pinedo [d], Dvir Shabtay [a 1]

Show more

Add to Mendeley   Share   Cite

https://doi.org/10.1016/j.ejor.2024.12.052

Get rights and content

Related Work

- Recently established framework [HMN$^+$25]
- Studied objectives:
  - Completion time
  - Lateness
  - Number of late jobs

ELSEVIER
European Journal of Operational Research
Volume 323, Issue 3, 16 June 2025, Pages 724-738

Discrete Optimization

Fairness in repetitive scheduling ☆

Danny Hermelin [a 1], Hendrik Molter [b 2], Rolf Niedermeier [c], Michael Pinedo [d], Dvir Shabtay [a 1]

Show more

Add to Mendeley Share Cite

https://doi.org/10.1016/j.ejor.2024.12.052 ↗

Get rights and content ↗

Framework

Fair Repetitive Interval Scheduling

Input:

A single machine and n *clients* each has a job per day for a period of m days.

Every job has (ith day and jth client):

- *processing time* $p_{i,j}$
- *deadline*.

A Quality of Service (QoS) performance measure $Z_{i,j}$.

Output: Feasible and *fair* schedule.

*Fair: a schedule that guarantees every client that $\sum_{j \leq m} Z_{i,j} \geq k$.

Problem Definition

FAIR REPETITIVE **INTERVAL** SCHEDULING

Input:

A single machine and *n clients* each has a job per day for a period of *m* days.

Every job has (ith day and jth client):

- *processing time* $p_{i,j}$.
- *deadline* $d_{i,j}$.

A Quality of Service (QoS) measure: **the number of JIT scheduled jobs per client** .

Output: Feasible and *fair* schedule.

*Fair: a schedule that guarantees every client **at least k JIT scheduled jobs** .

FAIR REPETITIVE INTERVAL SCHEDULING

Input:

A single machine and *n clients* each has a job per day for a period of *m* days.

Every job has (ith day and jth client):

- *processing time* $p_{i,j}$.
- *deadline* $d_{i,j}$.

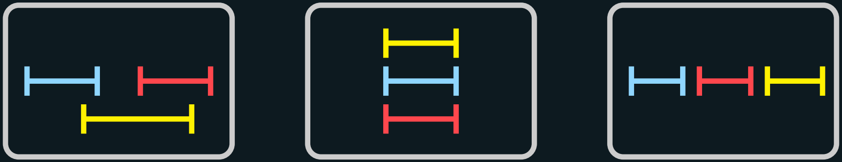
A Quality of Service (QoS) measure: the number of JIT scheduled jobs per client.

Output: Feasible and *fair* schedule.

*Fair: a schedule that guarantees every client at least k JIT scheduled jobs.

Example

Let the fairness parameter be 2 in this example.



Day 1 Day 2 Day 3

Example

Let the fairness parameter be 2 in this example.

We can schedule the jobs as follows such that all clients are served in at least 2 days.

Fairness Parameter

Theorem 1

Fair Repetitive Interval Scheduling is polynomial-time solvable for $k \in \{0, m-1, m\}$ and NP-hard otherwise.

Fairness Parameter

	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$
$m=1$	(1,1)						
$m=2$	(1,2)	(2,2)					
$m=3$	(1,3)	(2,3)	(3,3)				
$m=4$	(1,4)	(2,4)	(3,4)	(4,4)			
$m=5$	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)		
$m=6$	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	
$m=7$	(1,7)	(2,7)	(3,7)	(4,7)	(5,7)	(6,7)	(7,7)

Fairness Parameter

$m=1$	(1,1)						
$m=2$	(1,2)	(2,2)					
$m=3$	(1,3)	(2,3)	(3,3)				
$m=4$	(1,4)	(2,4)	(3,4)	(4,4)			
$m=5$	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)		
$m=6$	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	
$m=7$	(1,7)	(2,7)	(3,7)	(4,7)	(5,7)	(6,7)	(7,7)
	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$

Theorem 2

FAIR REPETITIVE INTERVAL SCHEDULING *is NP-hard for* $k = 1$ *and* $m = 3$*.*

Fairness Parameter

Theorem 2

FAIR REPETITIVE INTERVAL SCHEDULING *is NP-hard for* $k = 1$ *and* $m = 3$.

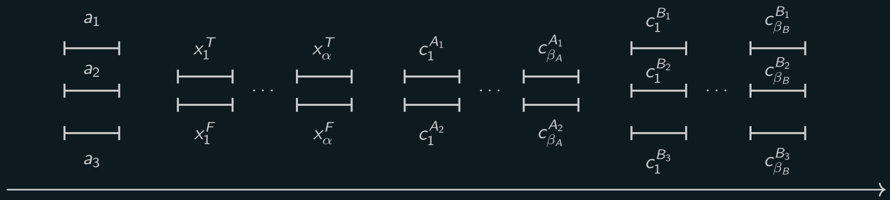
Reduction from $[2-3]$ BOUNDED SAT:

אוניברסיטת בן-גוריון בנגב
جامعة بن غوريون في النقب
Ben-Gurion University of the Negev

Theorem 2

Fair Repetitive Interval Scheduling *is NP-hard for* $k = 1$ *and* $m = 3$.

Reduction from $[2 - 3]$ Bounded SAT:



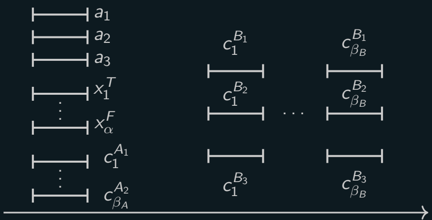
Day 1

Fairness Parameter

Theorem 2

FAIR REPETITIVE INTERVAL SCHEDULING *is NP-hard for* $k = 1$ *and* $m = 3$.

Reduction from $[2-3]$ BOUNDED SAT:



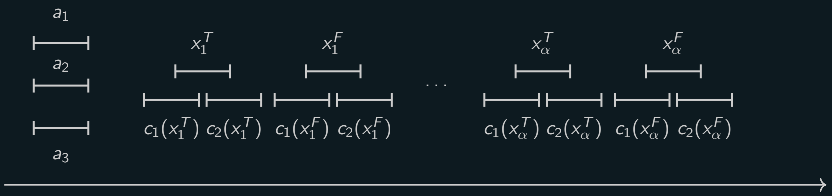
Day 2

אוניברסיטת בן-גוריון בנגב
جامعة بن غوريون في النقب
Ben-Gurion University of the Negev

Theorem 2

Fair Repetitive Interval Scheduling *is NP-hard for* $k = 1$ *and* $m = 3$*.*

Reduction from $[2-3]$ Bounded SAT:



Day 3 - The Validation Day

Fairness Parameter

$m = 1$	(1,1)						
$m = 2$	(1,2)	(2,2)					
$m = 3$	(1,3)	(2,3)	(3,3)				
$m = 4$	(1,4)	(2,4)	(3,4)	(4,4)			
$m = 5$	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)		
$m = 6$	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	
$m = 7$	(1,7)	(2,7)	(3,7)	(4,7)	(5,7)	(6,7)	(7,7)
	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$	$k = 7$

Fairness Parameter

	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$
$m=1$	(1,1)						
$m=2$	(1,2)	(2,2)					
$m=3$	(1,3)	(2,3)	(3,3)				
$m=4$	(1,4)	(2,4)	(3,4)	(4,4)			
$m=5$	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)		
$m=6$	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	
$m=7$	(1,7)	(2,7)	(3,7)	(4,7)	(5,7)	(6,7)	(7,7)

Fairness Parameter

$m = 1$	(1,1)						
$m = 2$	(1,2)	(2,2)					
$m = 3$	(1,3)	(2,3)	(3,3)				
$m = 4$	(1,4)	(2,4)	(3,4)	(4,4)			
$m = 5$	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)		
$m = 6$	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	
$m = 7$	(1,7)	(2,7)	(3,7)	(4,7)	(5,7)	(6,7)	(7,7)
	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$	$k = 7$

Fairness Parameter

	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$
$m=1$	(1,1)						
$m=2$	(1,2)	(2,2)					
$m=3$	(1,3)	(2,3)	(3,3)				
$m=4$	(1,4)	(2,4)	(3,4)	(4,4)			
$m=5$	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)		
$m=6$	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	
$m=7$	(1,7)	(2,7)	(3,7)	(4,7)	(5,7)	(6,7)	(7,7)

Fairness Parameter

$m = 1$	(1,1)						
$m = 2$	(1,2)	(2,2)					
$m = 3$	(1,3)	(2,3)	(3,3)				
$m = 4$	(1,4)	(2,4)	(3,4)	(4,4)			
$m = 5$	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)		
$m = 6$	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	
$m = 7$	(1,7)	(2,7)	(3,7)	(4,7)	(5,7)	(6,7)	(7,7)
	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$	$k = 6$	$k = 7$

Fairness Parameter

Theorem 3

Fair Repetitive Interval Scheduling *is polynomial-time solvable for* $k = m - 1$.

Fairness Parameter

Theorem 3

FAIR REPETITIVE INTERVAL SCHEDULING *is polynomial-time solvable for* $k = m - 1$.

Reduction to 2SAT:

Fairness Parameter

Theorem 3

FAIR REPETITIVE INTERVAL SCHEDULING *is polynomial-time solvable for* $k = m - 1$.

Reduction to 2SAT:

- For every client j and day i we create $x_{i,j}$.

Theorem 3

Fair Repetitive Interval Scheduling *is polynomial-time solvable for* $k = m - 1$.

Reduction to 2SAT:

- For every client j and day i we create $x_{i,j}$.
- We create the conflict clause $(\neg x_{i,j_1}, \vee \neg x_{i,j_2})$ if clients j_1 and j_2 are in conflict on day i.

Fairness Parameter

Theorem 3

FAIR REPETITIVE INTERVAL SCHEDULING *is polynomial-time solvable for* $k = m - 1$.

Reduction to 2SAT:

- For every client j and day i we create $x_{i,j}$.
- We create the conflict clause $(\neg x_{i,j_1}, \vee \neg x_{i,j_2})$ if clients j_1 and j_2 are in conflict on day i.
- We create $\mathcal{O}(m^2)$ validation clause for every client $(x_{i_1,j}, \vee x_{i_2,j})$ for $1 \leq i_1 < i_2 \leq m$.

Fairness Parameter

	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$
$m=1$	(1,1)						
$m=2$	(1,2)	(2,2)					
$m=3$	(1,3)	(2,3)	(3,3)				
$m=4$	(1,4)	(2,4)	(3,4)	(4,4)			
$m=5$	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)		
$m=6$	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	
$m=7$	(1,7)	(2,7)	(3,7)	(4,7)	(5,7)	(6,7)	(7,7)

Fairness Parameter

	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$	$k=6$	$k=7$
$m=1$	(1,1)						
$m=2$	(1,2)	(2,2)					
$m=3$	(1,3)	(2,3)	(3,3)				
$m=4$	(1,4)	(2,4)	(3,4)	(4,4)			
$m=5$	(1,5)	(2,5)	(3,5)	(4,5)	(5,5)		
$m=6$	(1,6)	(2,6)	(3,6)	(4,6)	(5,6)	(6,6)	
$m=7$	(1,7)	(2,7)	(3,7)	(4,7)	(5,7)	(6,7)	(7,7)

Theorem 4

FAIR REPETITIVE INTERVAL SCHEDULING *is NP-hard also when* $d_{i,j} = d_j$.

It is polynomial-time solvable when either of the following additionally holds:

- *The number of days* m *is constant.*
- *The processing times are day-independent* $p_{i,j} = p_j$.

Theorem 5

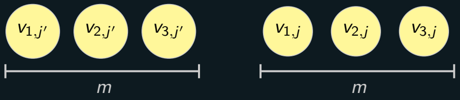
FAIR REPETITIVE INTERVAL SCHEDULING *is NP-hard also when* $p_{i,j} = 2$.

It is polynomial-time solvable when $p_j = 1$

Theorem 5

FAIR REPETITIVE INTERVAL SCHEDULING *is NP-hard also when* $p_{i,j} = 2$.

It is polynomial-time solvable when $p_j = 1$

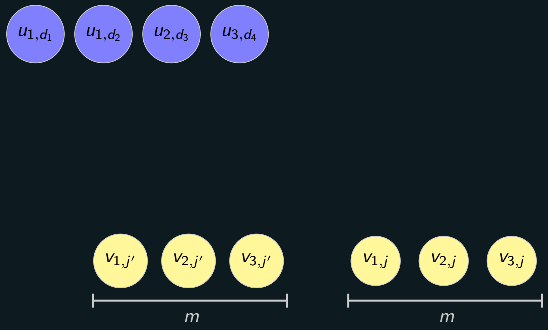


Fair Slot Allocation

Theorem 5

Fair Repetitive Interval Scheduling *is NP-hard also when* $p_{i,j} = 2$.

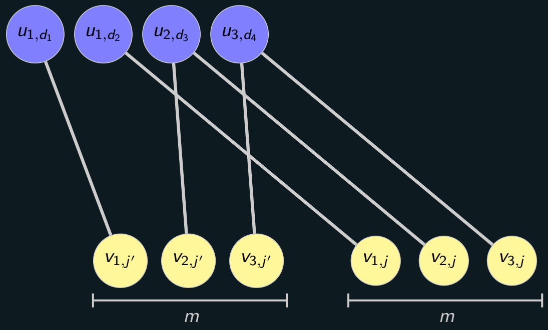
It is polynomial-time solvable when $p_j = 1$



Theorem 5

FAIR REPETITIVE INTERVAL SCHEDULING *is NP-hard also when* $p_{i,j} = 2$.

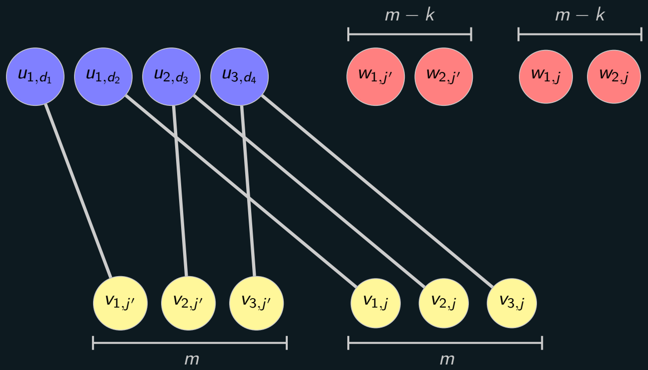
It is polynomial-time solvable when $p_j = 1$



Theorem 5

FAIR REPETITIVE INTERVAL SCHEDULING *is NP-hard also when* $p_{i,j} = 2$.

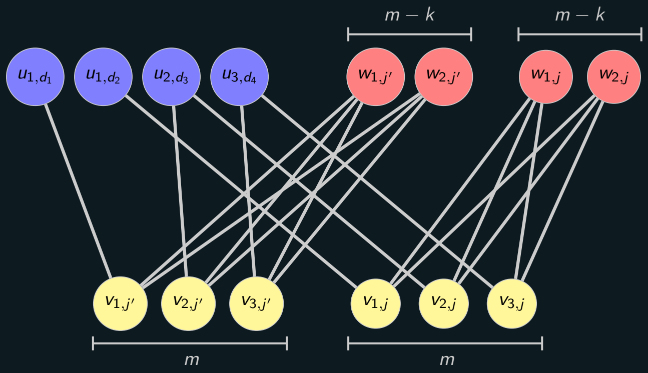
It is polynomial-time solvable when $p_j = 1$



Theorem 5

FAIR REPETITIVE INTERVAL SCHEDULING *is NP-hard also when* $p_{i,j} = 2$.

It is polynomial-time solvable when $p_j = 1$

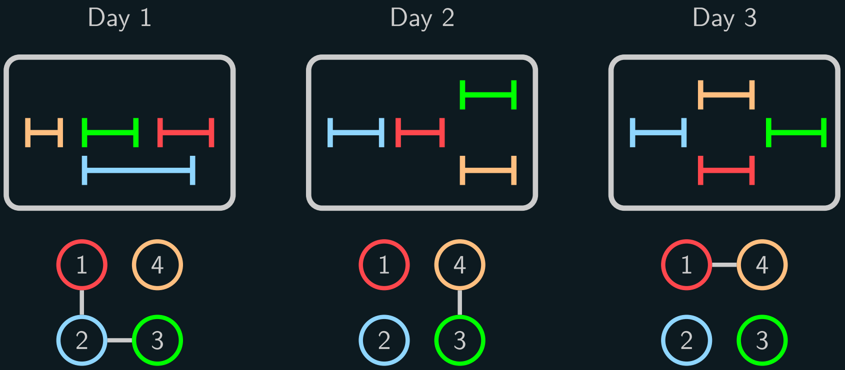


Theorem 6

FAIR REPETITIVE INTERVAL SCHEDULING *is:*

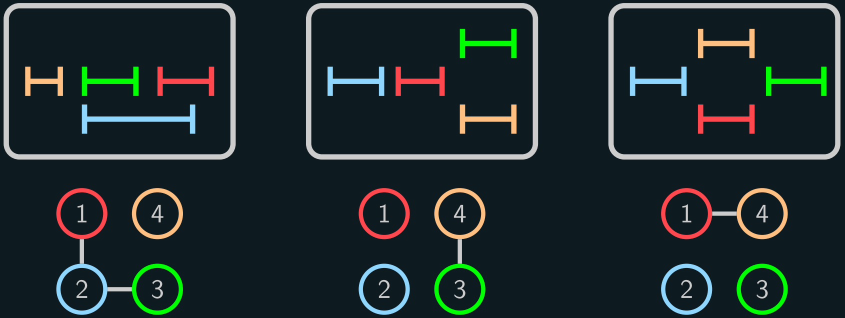
- *NP-hard for a constant number of days m.*
- *NP-hard for a constant treewidth τ.*
- *FPT with respect to $m + \tau$.*

The Conflict Graph

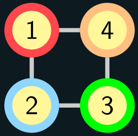


The Conflict Graph

Day 1 | Day 2 | Day 3



The Overall Conflict Graph



Discussion

אוניברסיטת בן-גוריון בנגב
جامعة بن غوريون في النقب
Ben-Gurion University of the Negev

Discussion

Fairness is hard.

Discussion

Fairness is hard.

Interesting generalizations:

Discussion

Fairness is hard.

Interesting generalizations:

- Clients have different fairness-parameter.

Discussion

Fairness is hard.

Interesting generalizations:

- Clients have different fairness-parameter.
- Multiple jobs per client.

Discussion

Fairness is hard.

Interesting generalizations:

- Clients have different fairness-parameter.
- Multiple jobs per client.
- Multiple machines per day.

References

[HMN+25] Danny Hermelin, Hendrik Molter, Rolf Niedermeier, Michael Pinedo, and Dvir Shabtay. Fairness in repetitive scheduling. *European Journal of Operational Research*, 323(3):724–738, 2025.