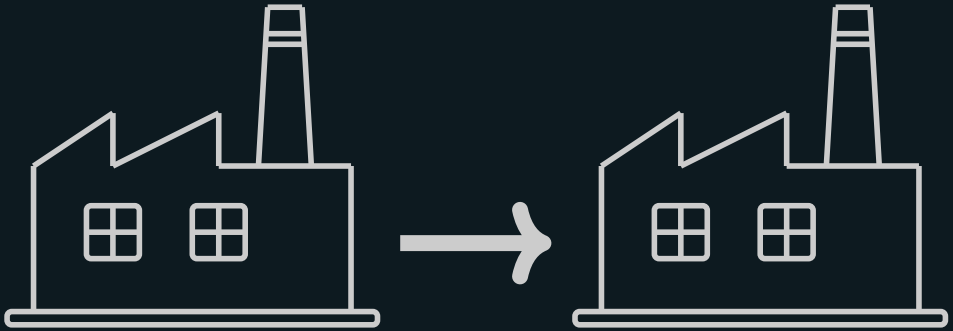
Just-in-Time Scheduling in Two Stage Flexible Flow Shop

Klaus Heeger, Danny Hermelin, **Yuval Itzhaki**, Baruch Schieber, Dvir Shabtay

Flow Shop



Parts-Plant → Assembly-Plant

Problem Definition

Two-Stage Flow Shop Interval Scheduling

$$F2||\sum w_j Z_j$$

Input:

Problem Definition

Two-Stage Flow Shop Interval Scheduling

$$F2||\sum w_j Z_j$$

Input: A set of n *jobs*,

Problem Definition

Two-Stage Flow Shop Interval Scheduling

$$F2||\sum w_j Z_j$$

Input: A set of n *jobs*, each job j has:

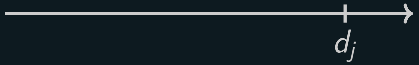
Problem Definition

Two-Stage Flow Shop Interval Scheduling

$$F2||\sum w_j Z_j$$

Input: A set of n *jobs*, each job j has:

- *due date* d_j



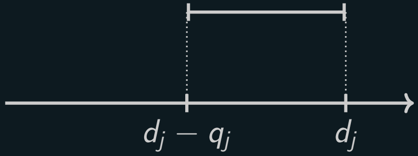
Problem Definition

Two-Stage Flow Shop Interval Scheduling

$$F2||\sum w_j Z_j$$

Input: A set of n *jobs*, each job j has:

- *due date* d_j
- *processing time* q_j



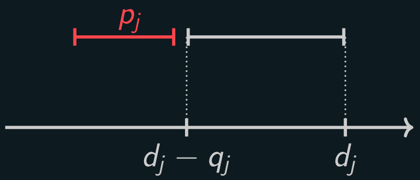
Problem Definition

Two-Stage Flow Shop Interval Scheduling

$$F2||\sum w_j Z_j$$

Input: A set of n *jobs*, each job j has:

- *due date* d_j
- *processing time* q_j
- *preprocessing time* p_j



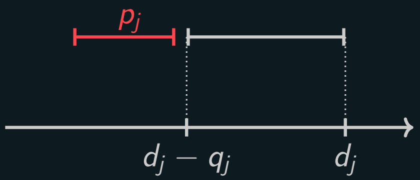
Problem Definition

Two-Stage Flow Shop Interval Scheduling

$$F2||\sum w_j Z_j$$

Input: A set of n *jobs*, each job j has:

- *due date* d_j
- *processing time* q_j
- *preprocessing time* p_j
- *weight* w_j



Problem Definition

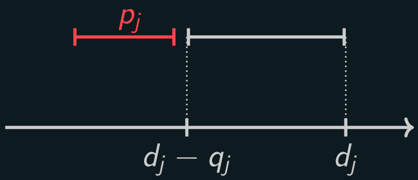
Two-Stage Flow Shop Interval Scheduling

$$F2||\sum w_j Z_j$$

Input: A set of n *jobs*, each job j has:

- *due date* d_j
- *processing time* q_j
- *preprocessing time* p_j
- *weight* w_j

For the preprocessing we have 1 machine, for the processing we have 1 machine.



Problem Definition

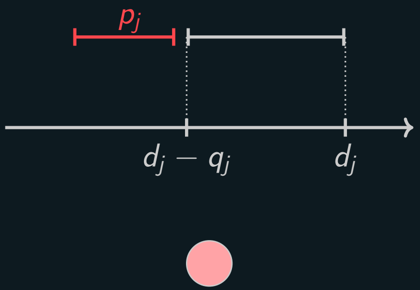
Two-Stage Flow Shop Interval Scheduling

$$F2||\sum w_j Z_j$$

Input: A set of n *jobs*, each job j has:

- *due date* d_j
- *processing time* q_j
- *preprocessing time* p_j
- *weight* w_j

For the preprocessing we have 1 machine, for the processing we have 1 machine.



Problem Definition

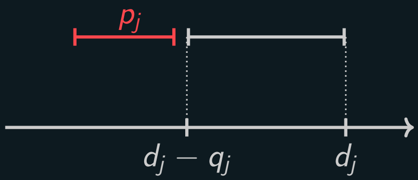
Two-Stage Flow Shop Interval Scheduling

$$F2||\sum w_j Z_j$$

Input: A set of n *jobs*, each job j has:

- *due date* d_j
- *processing time* q_j
- *preprocessing time* p_j
- *weight* w_j

For the preprocessing we have 1 machine, for the processing we have 1 machine.



Problem Definition

Two-Stage Flow Shop Interval Scheduling

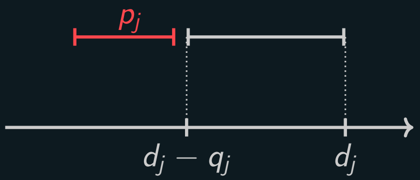
$$F2||\sum w_j Z_j$$

Input: A set of n *jobs*, each job j has:

- *due date* d_j
- *processing time* q_j
- *preprocessing time* p_j
- *weight* w_j

For the preprocessing we have 1 machine, for the processing we have 1 machine.

Output: Maximum weighted JIT schedule.



Problem Definition

Two-Stage Flexible Flow Shop Interval Scheduling

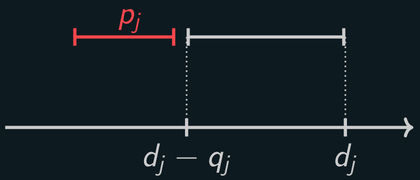
$$F(1, m)||\sum w_j Z_j$$

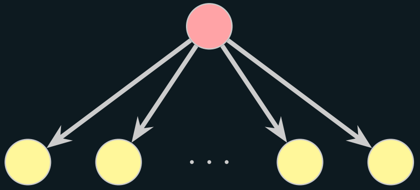
Input: A set of n *jobs*, each job j has:

- *due date* d_j
- *processing time* q_j
- *preprocessing time* p_j
- *weight* w_j

For the preprocessing we have 1 machine, for the processing we have m identical machines.

Output: Maximum weighted JIT schedule.





Related Work

Related Work

JIT Flow Shop

$\boxed{1||\sum w_j Z_j}$

Related Work

JIT Flow Shop

$F2||\sum w_j Z_j$

$1||\sum w_j Z_j$

Related Work

JIT Flow Shop

$F3||\sum w_j Z_j$

$F2||\sum w_j Z_j$

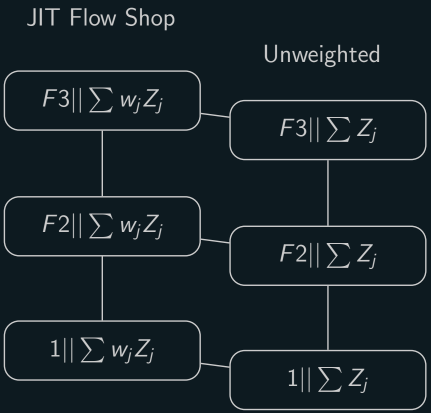
$1||\sum w_j Z_j$

Related Work

Interval Scheduling

Two-Stage ~~Flexible~~ JIT Flow Shop

Three-Stage ~~Flexible~~ JIT Flow Shop



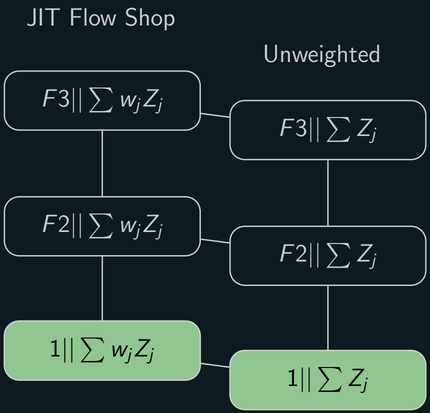
Related Work

Interval Scheduling

- $O(n \log n)$ time solvable single machine [Gol88].

Two-Stage ~~Flexible~~ JIT Flow Shop

Three-Stage ~~Flexible~~ JIT Flow Shop



Related Work

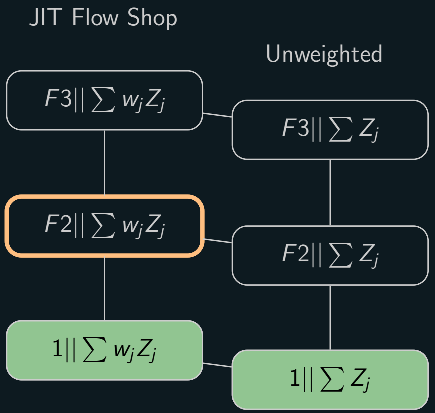
Interval Scheduling

- $O(n \log n)$ time solvable single machine [Gol88].

Two-Stage ~~Flexible~~ JIT Flow Shop

- NP-hard even when $q_j = 1$ $(m = 1)$ [CY07]

Three-Stage ~~Flexible~~ JIT Flow Shop



Related Work

Interval Scheduling

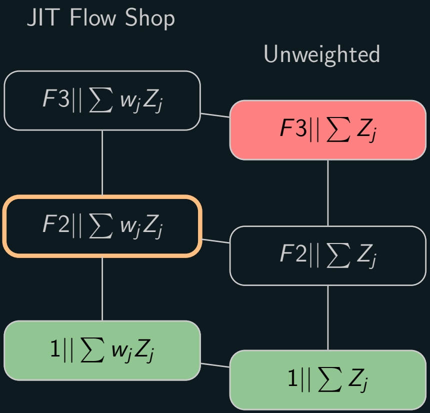
- $O(n \log n)$ time solvable single machine [Gol88].

Two-Stage ~~Flexible~~ JIT Flow Shop

- NP-hard even when $q_j = 1$ $(m = 1)$ [CY07]

Three-Stage ~~Flexible~~ JIT Flow Shop

- Strongly NP-hard even when $w_j = 1$ [CY07].



Related Work

Interval Scheduling

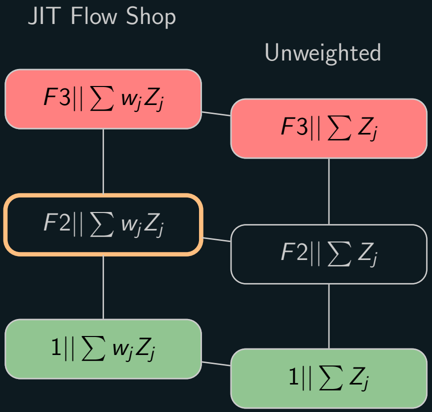
- $O(n \log n)$ time solvable single machine [Gol88].

Two-Stage ~~Flexible~~ JIT Flow Shop

- NP-hard even when $q_j = 1$ $(m = 1)$ [CY07]

Three-Stage ~~Flexible~~ JIT Flow Shop

- Strongly NP-hard even when $w_j = 1$ [CY07].



Related Work

Interval Scheduling

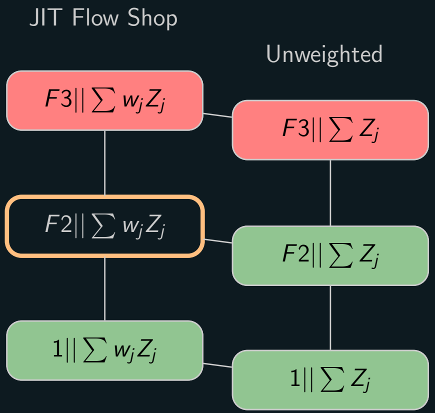
- $O(n \log n)$ time solvable single machine [Gol88].

Two-Stage ~~Flexible~~ JIT Flow Shop

- NP-hard even when $q_j = 1$ $(m = 1)$ [CY07]
- When $w_j = 1$ $O(n^4)$ time solvable [CY07].

Three-Stage ~~Flexible~~ JIT Flow Shop

- Strongly NP-hard even when $w_j = 1$ [CY07].



Related Work

Interval Scheduling

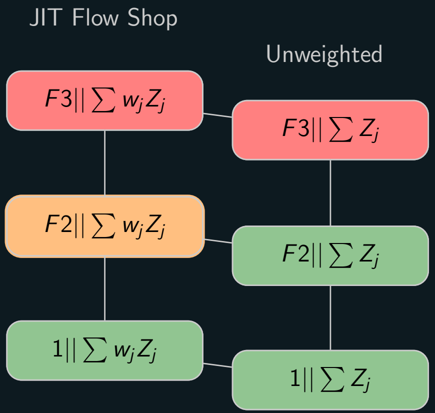
- $O(n \log n)$ time solvable single machine [Gol88].

Two-Stage ~~Flexible~~ JIT Flow Shop

- NP-hard even when $q_j = 1$ $(m = 1)$ [CY07]
- When $w_j = 1$ $O(n^4)$ time solvable [CY07].
- Pseudo-polynomial time solvable [SB12].
- When $w_j = 1$ $O(n^3)$ time solvable [SB12].

Three-Stage ~~Flexible~~ JIT Flow Shop

- Strongly NP-hard even when $w_j = 1$ [CY07].



Related Work

Interval Scheduling

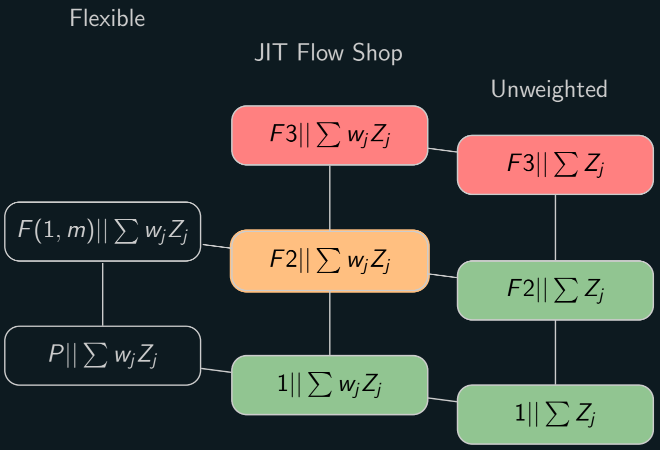
- $O(n \log n)$ time solvable single machine [Gol88].

Two-Stage ~~Flexible~~ JIT Flow Shop

- NP-hard even when $q_j = 1$ $(m = 1)$ [CY07]
- When $w_j = 1$ $O(n^4)$ time solvable [CY07].
- Pseudo-polynomial time solvable [SB12].
- When $w_j = 1$ $O(n^3)$ time solvable [SB12].

Three-Stage ~~Flexible~~ JIT Flow Shop

- Strongly NP-hard even when $w_j = 1$ [CY07].



Related Work

Interval Scheduling

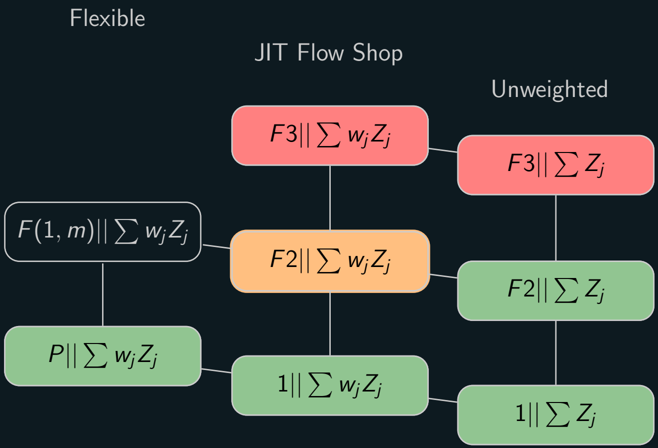
- $O(n \log n)$ time solvable single machine [Gol88].
- $O(n^2 \log n)$ time solvable multiple machines [AS87].

Two-Stage ~~Flexible~~ JIT Flow Shop

- NP-hard even when $q_j = 1$ $(m = 1)$ [CY07]
- When $w_j = 1$ $O(n^4)$ time solvable [CY07].
- Pseudo-polynomial time solvable [SB12].
- When $w_j = 1$ $O(n^3)$ time solvable [SB12].

Three-Stage ~~Flexible~~ JIT Flow Shop

- Strongly NP-hard even when $w_j = 1$ [CY07].



Related Work

Interval Scheduling

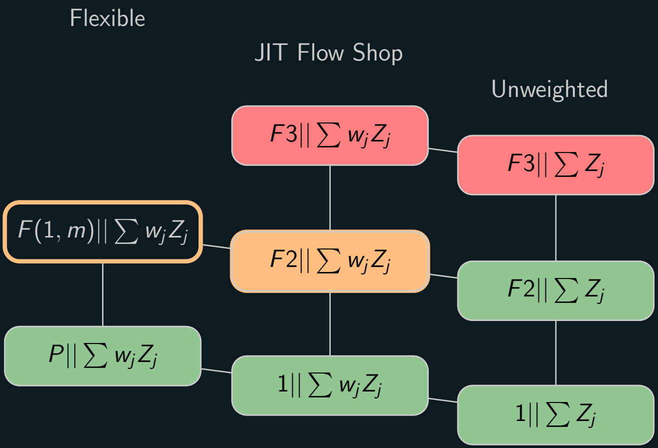
- $O(n \log n)$ time solvable single machine [Gol88].
- $O(n^2 \log n)$ time solvable multiple machines [AS87].

Two-Stage ~~Flexible~~ JIT Flow Shop

- NP-hard even when $q_j = 1$ $(m = 1)$ [CY07]
- When $w_j = 1$ $O(n^4)$ time solvable [CY07].
- Pseudo-polynomial time solvable [SB12].
- When $w_j = 1$ $O(n^3)$ time solvable [SB12].

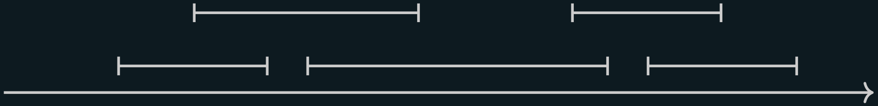
Three-Stage ~~Flexible~~ JIT Flow Shop

- Strongly NP-hard even when $w_j = 1$ [CY07].



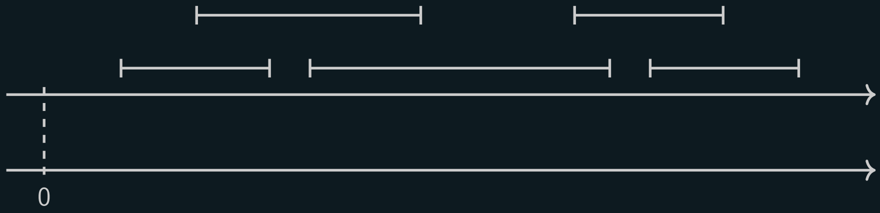
Key Observations

Observation 1



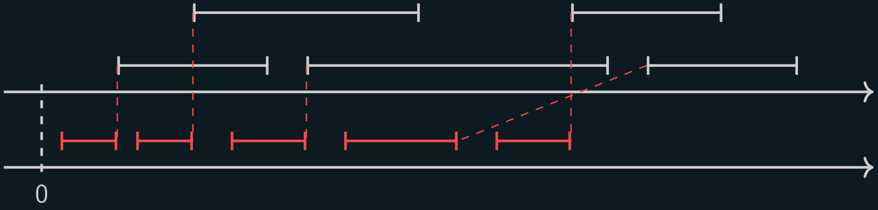
Key Observations

Observation 1



Key Observations

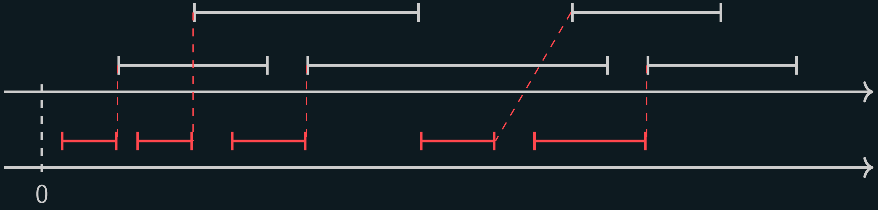
Observation 1



Key Observations

Observation 1

The jobs can be preprocessed in ascending order of start times of the second stage.



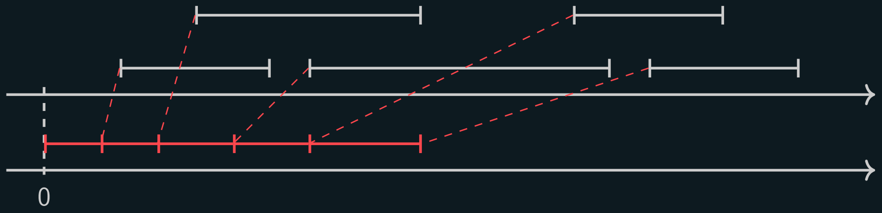
Key Observations

Observation 1

The jobs can be preprocessed in ascending order of start times of the second stage.

Observation 2

Optimally, the first stage machine has no idle time.

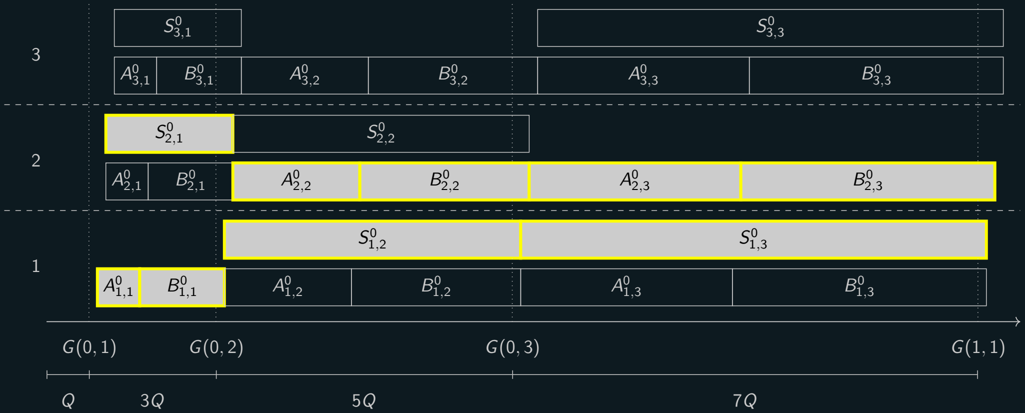


Hardness

HITTING SET:	
Input:	A family $\mathcal{F}$ of m subsets of a universe $U = \{1, \ldots, n\}$, and an integer k.
Question:	Is there a set $H \subseteq U$ with $\|H\| = k$ and $\|H \cap F\| \geq 1$ for every $F \in \mathcal{F}$?

$$U = \{1, 2, 3\} \ , \ \mathcal{F} = \{F_1 = \{2, 3\}, F_2 = \{1, 2\}, F_3 = \{1, 3\}\} \ , \ k = 2$$

Hardness



$$U = \{1, 2, 3\}\ ,\ \mathcal{F} = \{F_1 = \{2, 3\}, F_2 = \{1, 2\}, F_3 = \{1, 3\}\}\ ,\ k = 2$$

Dynamic Program

Observation 3

$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

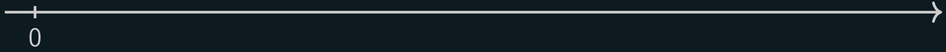
Partial-schedules (over jobs $\{j, \ldots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \ldots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

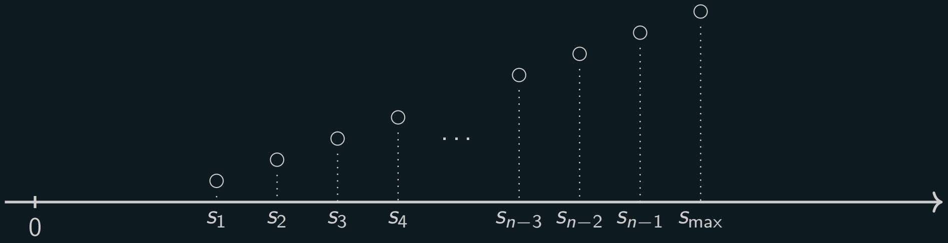


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \ldots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

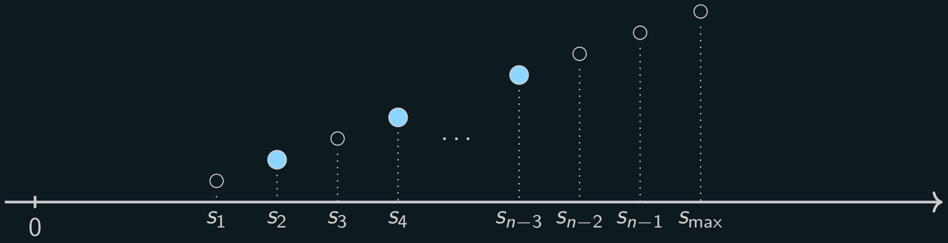


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \dots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

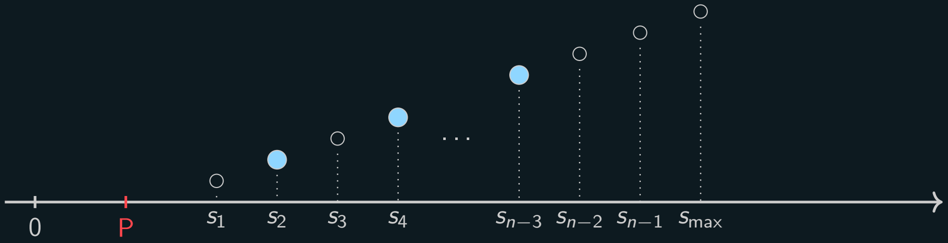


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \dots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

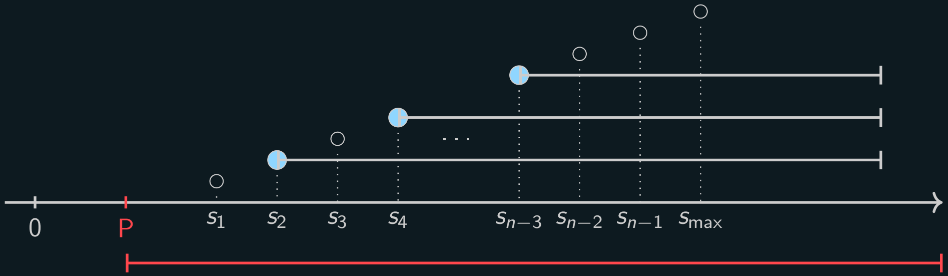


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \ldots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

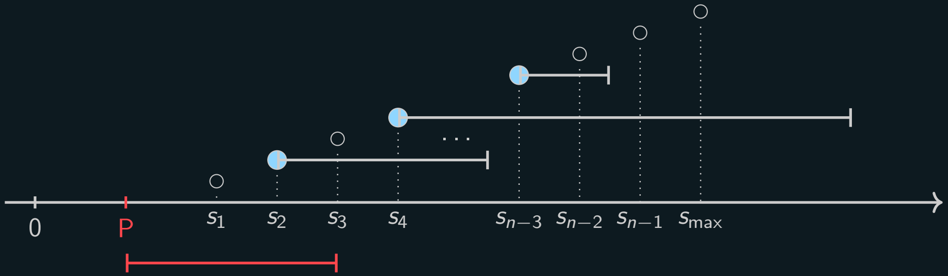


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \dots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

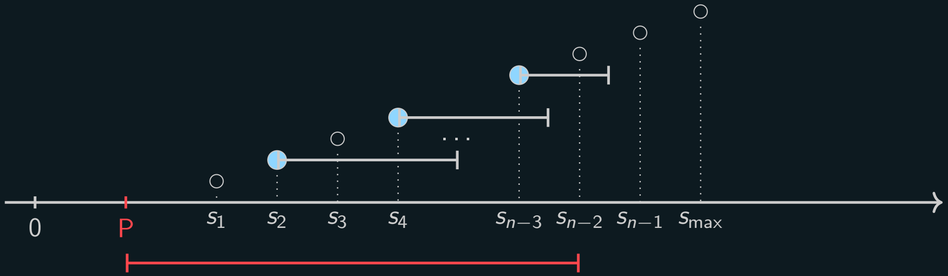


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \ldots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

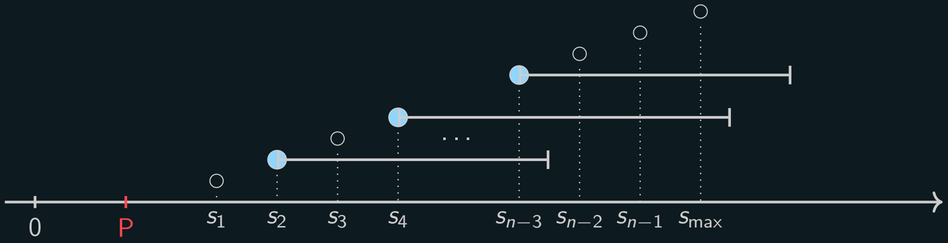


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \ldots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

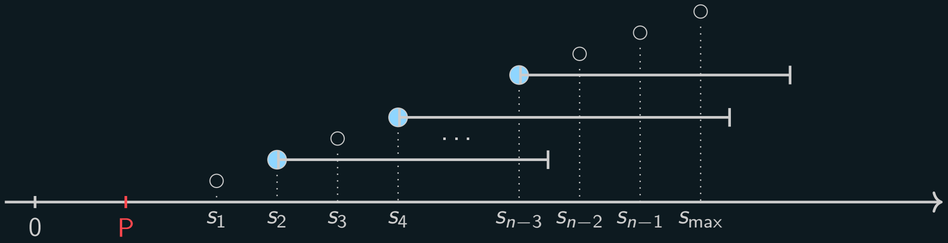


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \dots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

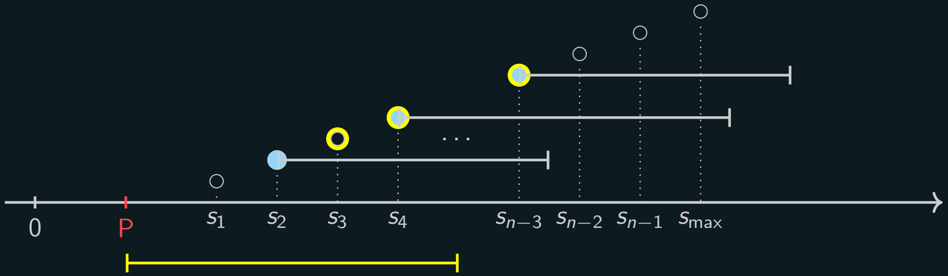


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \ldots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

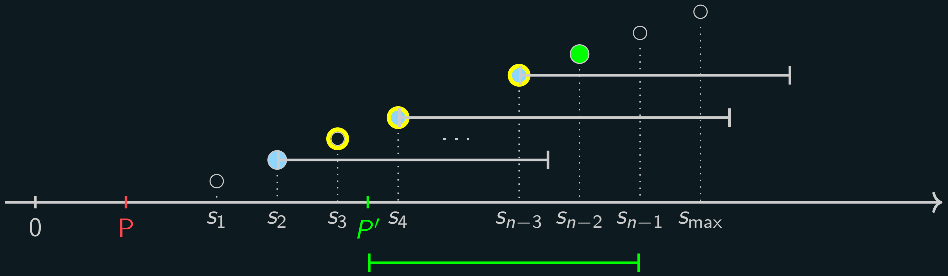


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \ldots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

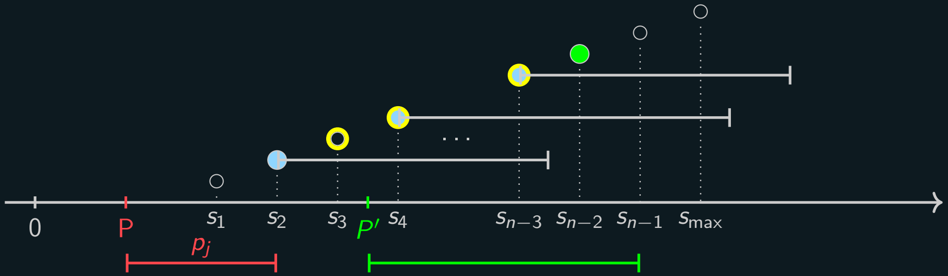


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \dots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".

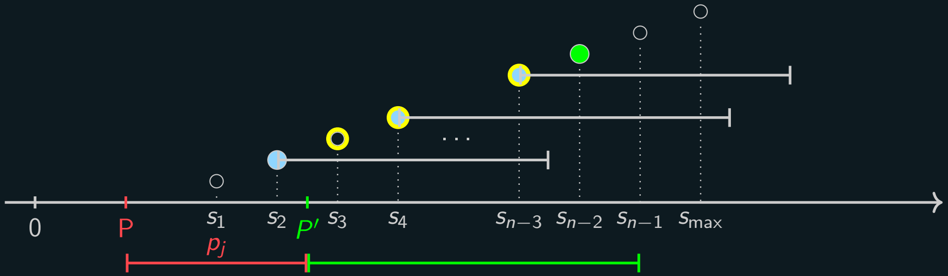


$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Dynamic Program

Observation 3

Partial-schedules (over jobs $\{j, \ldots, n\}$) with equal W, P and $\vec{s}$ are "equivalent".



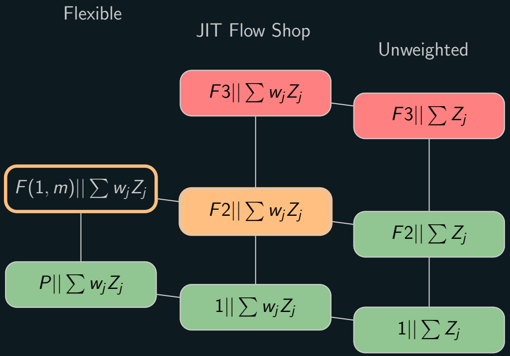
$$T[\vec{s}, W'] = \max \begin{cases} T[X_1, W'], \\ \min \begin{cases} T[X_2, W' - w_j] - p_j, \\ s_j - p_j. \end{cases} \end{cases}$$

Open Questions

- Is $F(1, m)|q_j = q|w_j Z_j$ NP-hard?

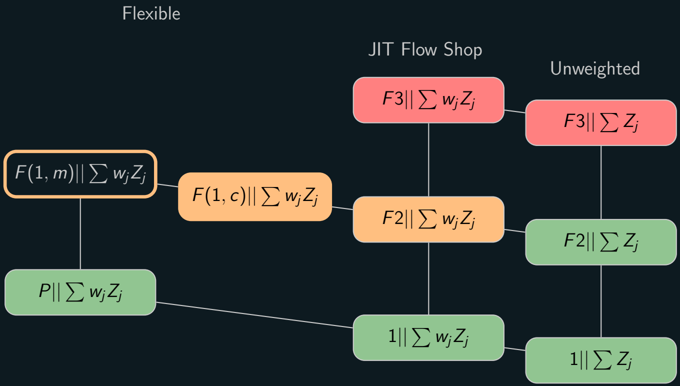
Open Questions

- Is $F(1, m)|q_j = q|w_j Z_j$ NP-hard?



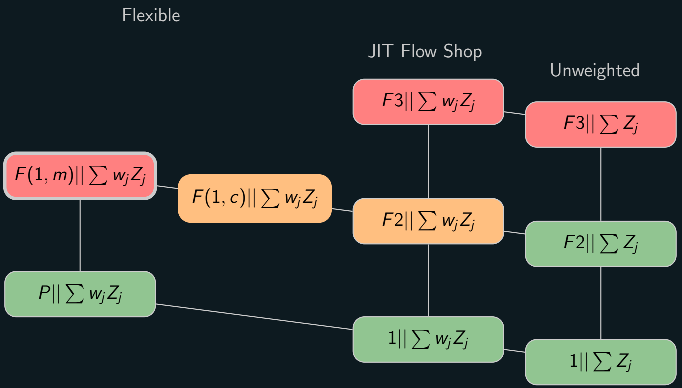
Open Questions

- Is $F(1,m)|q_j = q|w_jZ_j$ NP-hard?



Open Questions

- Is $F(1,m)|q_j = q|w_jZ_j$ NP-hard?



Thanks!

References

[AS87] Esther M. Arkin and Ellen B. Silverberg. Scheduling jobs with fixed start and end times. *Discrete Applied Mathematics*, 18(1):1–8, 1987.

[CY07] Byung-Cheon Choi and Suk-Hun Yoon. Maximizing the weighted number of just-in-time jobs in flow shop scheduling. *Journal of Scheduling*, 10:237–243, 2007.

[Gol88] Martin Charles Golumbic. Algorithmic aspects of intersection graphs and representation hypergraphs. *Graphs and Combinatorics*, 4(1):307–321, 1988.

[SB12] Dvir Shabtay and Yaron Bensoussan. Maximizing the weighted number of just-in-time jobs in several two-machine scheduling systems. *Journal of Scheduling*, 15(1):39–47, 2012.