Flow Shop Scheduling with Synchronous Material Movement

Kwei-Long Huang

Dept of Information Management
National Taiwan University

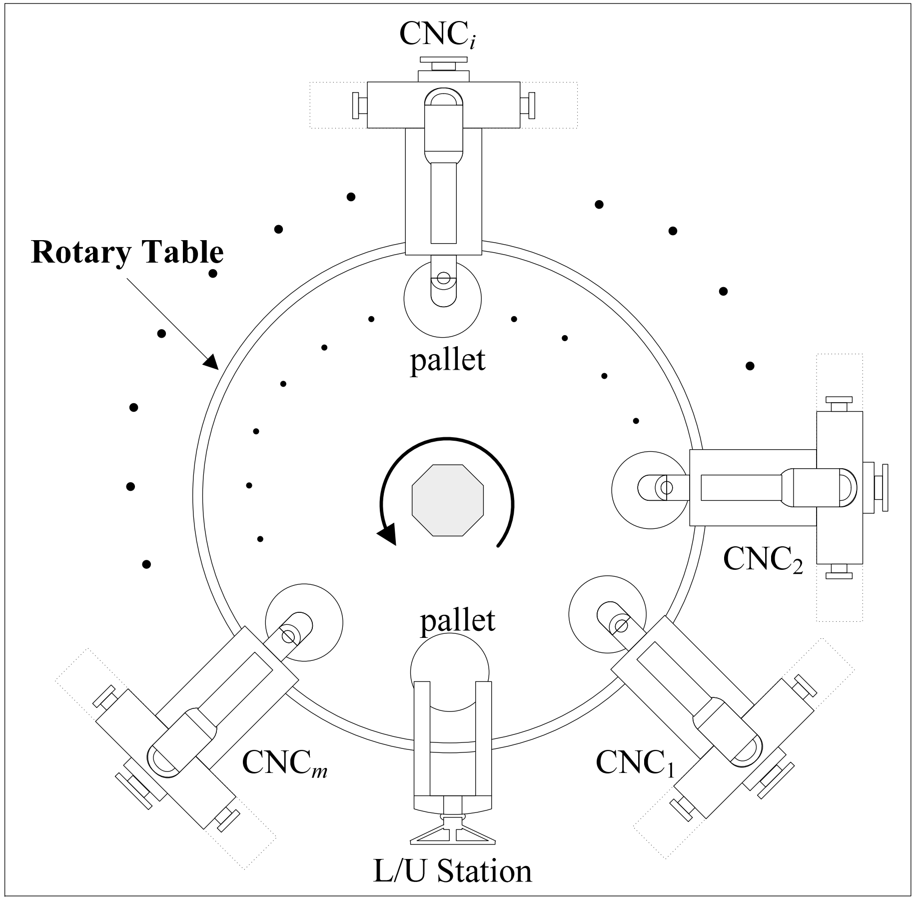
Information Management Seminar, National Taiwan University, Dec 12, 2008

Agenda

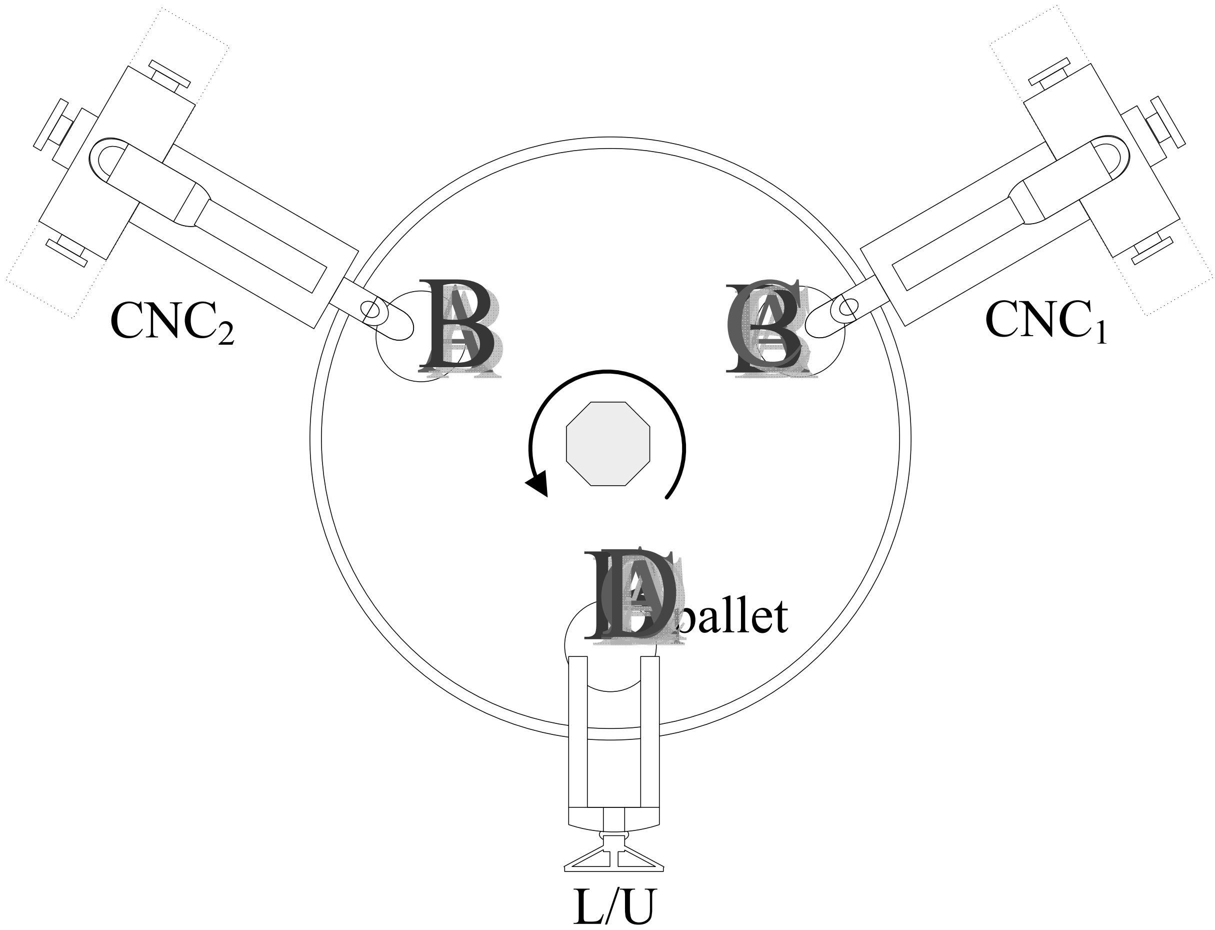
- Introduction
- T-line machining center with one machine
  - Complexity
  - Common unloading times
  - Dynamic programming algorithm
  - Lower bound
  - Heuristic algorithms and computational results
- T-line machining center with *m* machines
  - Dynamic programming algorithm
  - Lower bound
- Conclusions and future research

Synchronous Material Movement

- An automated manufacturing cell which consists of a loading/unloading (L/U) station, *m* machines, and a rotary table.
- The L/U station and these machines surround the rotary table.



Flow Shop Scheduling with Synchronous Material Movement



Flow Shop Scheduling with Synchronous Material Movement (con't)

- An application to this type of manufacturing cells is called T-line machining center produced by Milacron Cincinnati

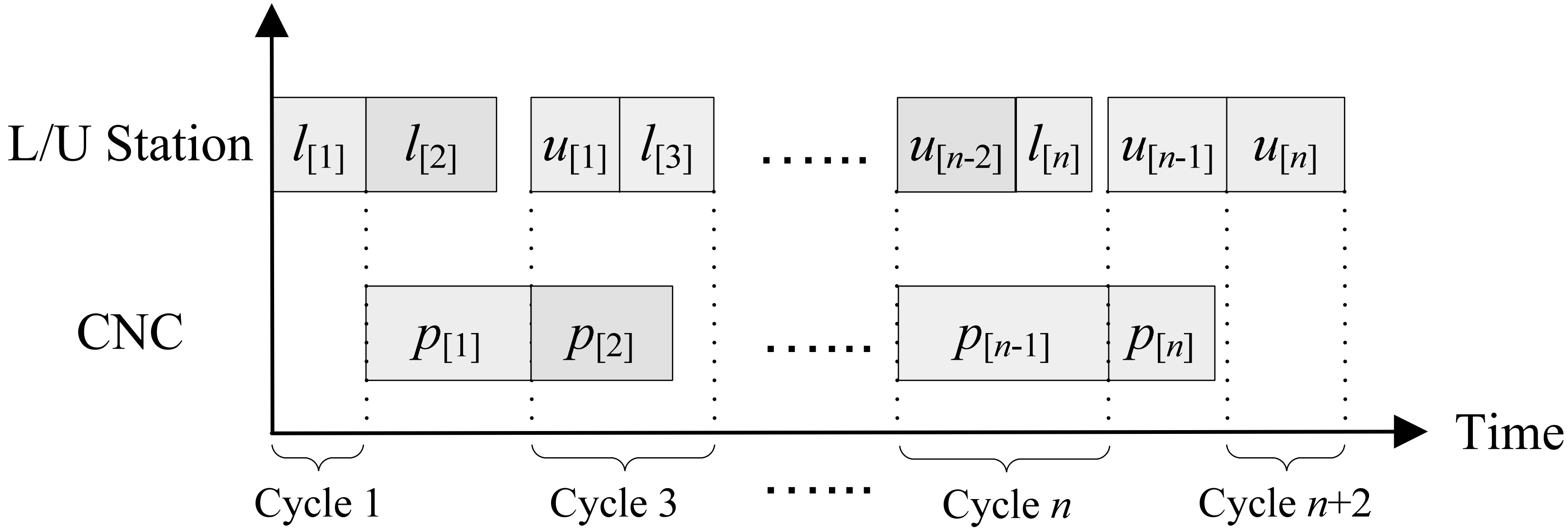
Agenda

- Introduction
- **T-line machining center with one machine**
  - Complexity
  - Common unloading times
  - Dynamic programming algorithm
  - Lower bound
  - Heuristic algorithms and computational results
- T-line machining center with *m* machines
  - Dynamic programming algorithm
  - Lower bound
- Conclusions and future research

T-line Machining Center with One Machine

- Problem assumptions:
  - Given n jobs and all jobs are available at time zero
  - No preemption
  - The machine can only process one job at a time
  - The rotary table with two pallets
  - Loading time, processing time, and unloading time for job j denoted as l_j, p_j and u_j, respectively
  - Objective is to minimize the makespan (C_{max})
    - Transportation times are neglected

Gantt Chart


L/U Station
$l_{[1]}$
$l_{[2]}$
$u_{[1]}$
$l_{[3]}$
……
$u_{[n-2]}$
$l_{[n]}$
$u_{[n-1]}$
$u_{[n]}$
CNC
$p_{[1]}$
$p_{[2]}$
……
$p_{[n-1]}$
$p_{[n]}$
Time
Cycle 1
Cycle 3
……
Cycle n
Cycle n+2

Complexity: NP-hard

- The scheduling problem with the makespan objective in a T-line machining center is NP-hard
- Show this problem is equivalent to a problem known as NP-hard
- Numerical matching problem with target sums (NMTS)
  - Strongly NP-Complete (Garey and Johnson 1979)
  - Two sets $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$
  - Target set $C = \{c_1, c_2, \ldots, c_n\}$

Can $A \bigcup B$ be partitioned into n disjoint sets D_k,

each containing exactly one element from set A

and one element from set B, such that,

$$c_k = a_{[k]} + b_{[k]},\ a_{[k]}, b_{[k]} \in D_k \text{ for } k = 1, \ldots, n\ ?$$

Decision problem (P)

- Given a number of jobs with loading, processing and unloading times that have to be processed in a one-machine T-line machining center, does there exist a schedule, say σ, with makespan $C_{max}(\sigma)$ which is less than or equal to a given value Z?

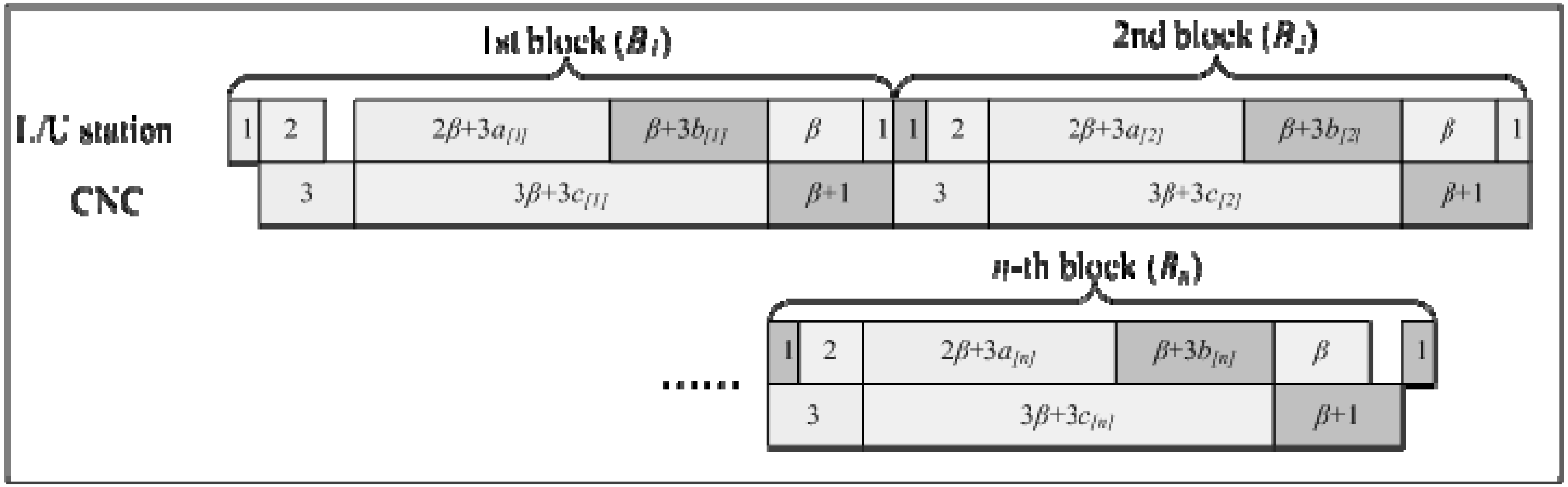
Complexity (con't)

- Given any instance of NMTS, the instance P1 for Problem P can be constructed as follows:
  - Contain three types of jobs
  - Ja-Type: $l(\mathrm{Ja}_i) = 1,\ p(\mathrm{Ja}_i) = 3,\ u(\mathrm{Ja}_i) = 2\beta + 3a_i,\ 1 \le i \le n,$
  - Jb-Type: $l(\mathrm{Jb}_i) = \beta + 3b_i,\ p(\mathrm{Jb}_i) = \beta + 1,\ u(\mathrm{Jb}_i) = 1,\ 1 \le i \le n,$
  - Jc-Type: $l(\mathrm{Jc}_i) = 2,\ p(\mathrm{Jc}_i) = 3\beta + 3c_i,\ u(\mathrm{Jc}_i) = \beta,\ 1 \le i \le n,$
  - Z value: $Z = 2 + 4n + 4n\beta + 3C_0$

    where $\beta = 3\max\{c_i \mid 1 \le i \le n\}$ and $C_0 = \sum_{i=1}^{n} c_i$

Complexity (con't) "the If part"

- Assume there exists a solution to NMTS
- A sequence can be formed as follows:



$$C_{\max} = 2 + 3n + 3n\beta + 3\sum_{i=1}^{n} c_i + n\beta + n = 2 + 4n + 4n\beta + 3C_0 = Z$$

Complexity (con't) "the Only If part"

- Assume that for instance P1 there exists a sequence, say σ, and its makespan $\leq$ Z. Then we have to show that there exists a solution to NMTS
- What does sequence σ look like?
- Lemma 1: on the CNC machine, the first and last cycles have one unit of idle time, respectively
- Lemma 2: on the L/U station, the second and second last cycles have one unit of idle time, respectively

Complexity (con't) "the Only If part"

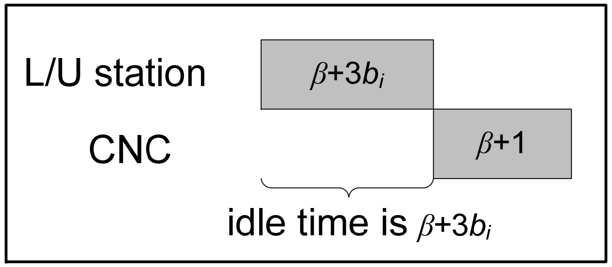
- Lemma 3: Jobs in sequence σ are sequenced in the following form

$$[Ja,\ Jc,\ Jb, Ja,\ Jc,\ Jb,\ldots, Ja,\ Jc,\ Jb]$$

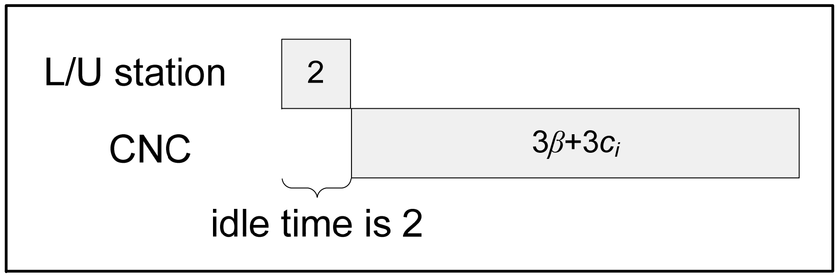
- Ideas of the proof
  - Consider first three jobs
  - If the first job is not Ja-type
  - If the second job is not Jc-type
  - If the third job is not Jb-type

Complexity (con't) "the Only If part"

- If the first job is not Ja-type
  - The first job is Jb-type

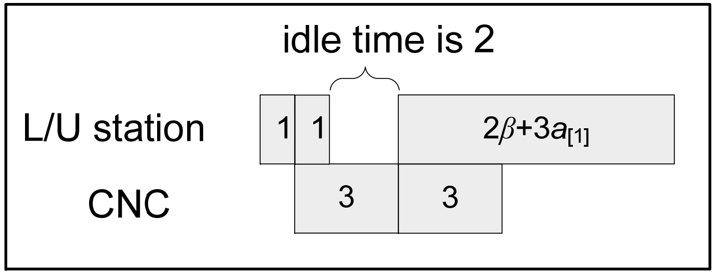


  - The first job is Jc-type

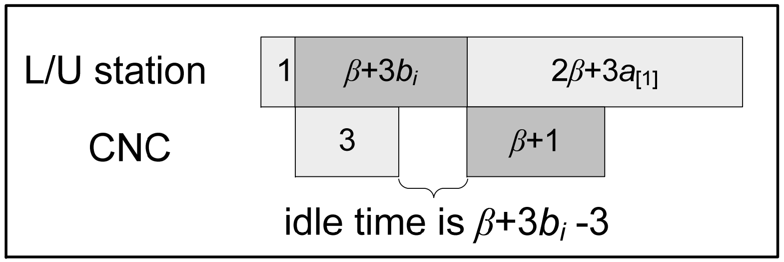


Complexity (con't) "the Only If part"

- If the second job is not Jc-type
  - The second job is Ja-type

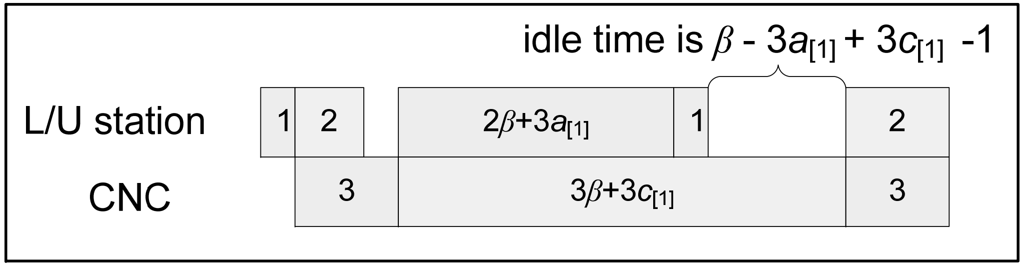


  - The second job is Jb-type

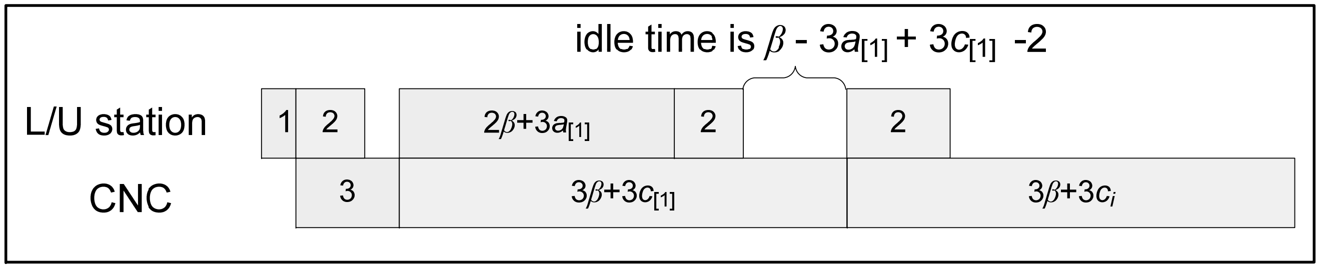


Complexity (con't) "the Only If part"

- If the third job is not Jb-type
  - The third job is Ja-type



  - The third job is Jc-type



Complexity (con't) "the Only If part"

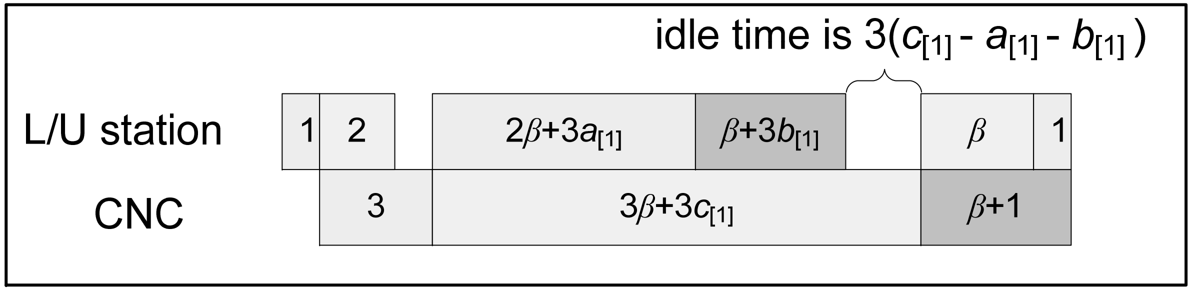
- Lemma 4: In sequence σ, every three jobs are grouped as a block, we will have

$$a_{[k]} + b_{[k]} = c_{[k]},\ k = 1, \ldots, n$$

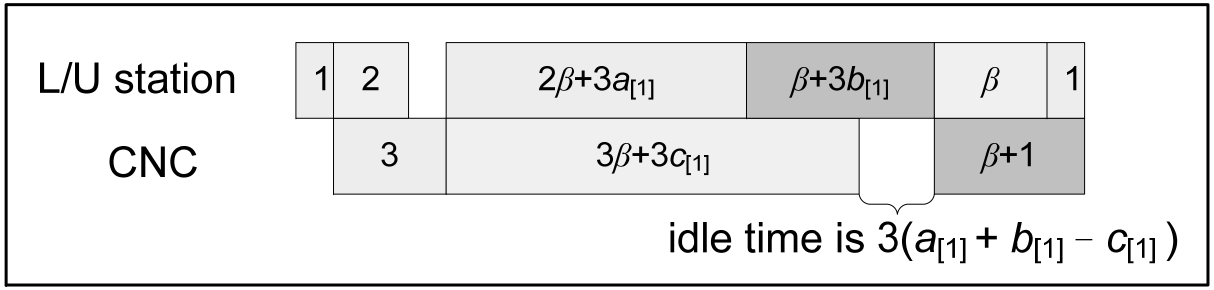
- Ideas of the proof
  - Consider the first three jobs
  - If $a_{[1]} + b_{[1]} < c_{[1]}$
  - If $a_{[1]} + b_{[1]} > c_{[1]}$

Complexity (con't) "the Only If part"

- If $a_{[1]} + b_{[1]} < c_{[1]}$



- If $a_{[1]} + b_{[1]} > c_{[1]}$

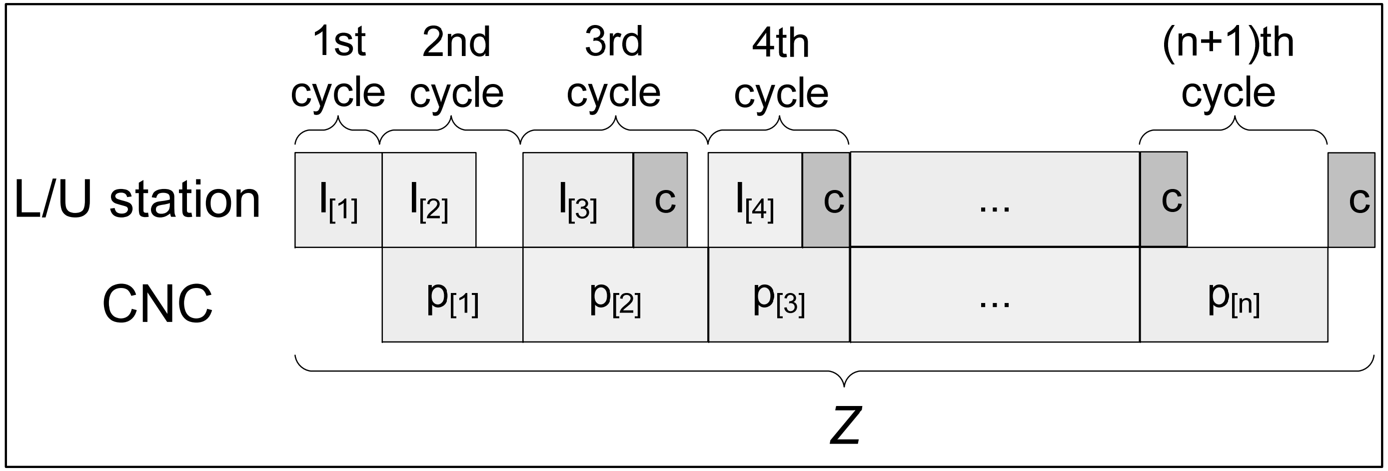


Complexity (con't) "the Only If part"

- Through Lemmas 1~4, we show that there exists a solution to NMTS
- Theorem 1: the scheduling problem with the makespan objective in a T-line machining center is NP-hard

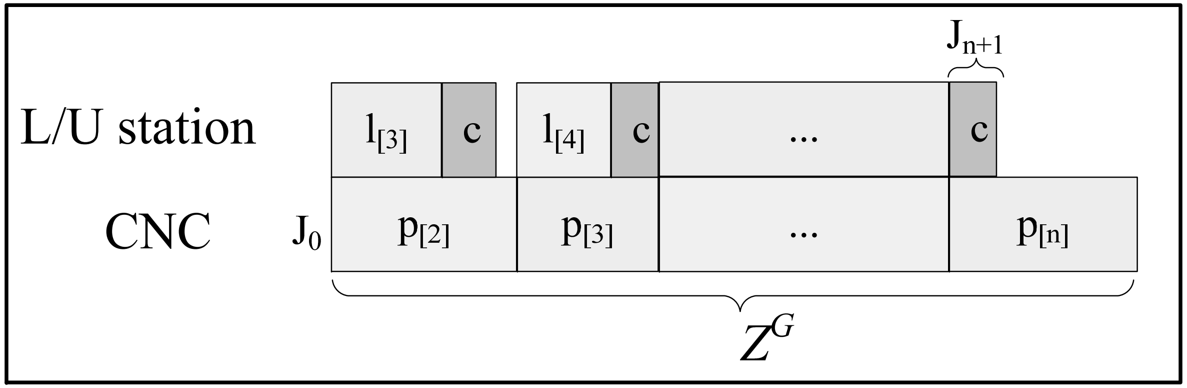
Common Unloading Times

- Assume all unloading times are equal to a constant (denoted as c)
- Given a sequence, the schedule on machines looks like as follows



Common Unloading Times (con't)

- Ignore the first two cycles and the last cycle, it will become a two-machine flow shop problem with blocking



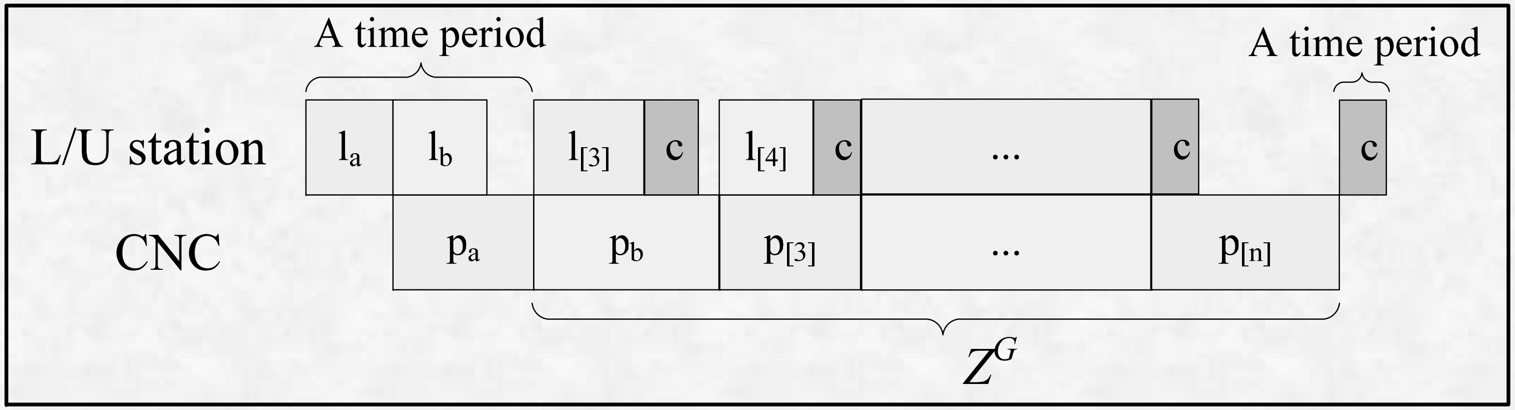
- The Gilmore-Gomory (G-G) algorithm (1964) can solve the problem with blocking optimally

Common Unloading Times (con't)

- Given the first two jobs, say a and b, in an optimal sequence, the rest of job sequence can be generated by the G-G algorithm
- Problem G:
  - Let $l_j = l_j + c$; $j \neq a$ and b
  - Additional two jobs
    - J_0: $l_0 = 0,\ p_0 = p_b$
    - J_{n+1}: $l_{n+1} = c$ and $p_{n+1} = 0$

Common Unloading Times (con't)

- Lemma 5: J_0 is sequenced in the first position
- Lemma 6: J_{n+1} is sequenced in the last position
- The optimal makespan of Problem G is Z^G
- The optimal makespan of the original problem (Z) is $Z^G + l_a + \max(l_b,\ p_a) + c$

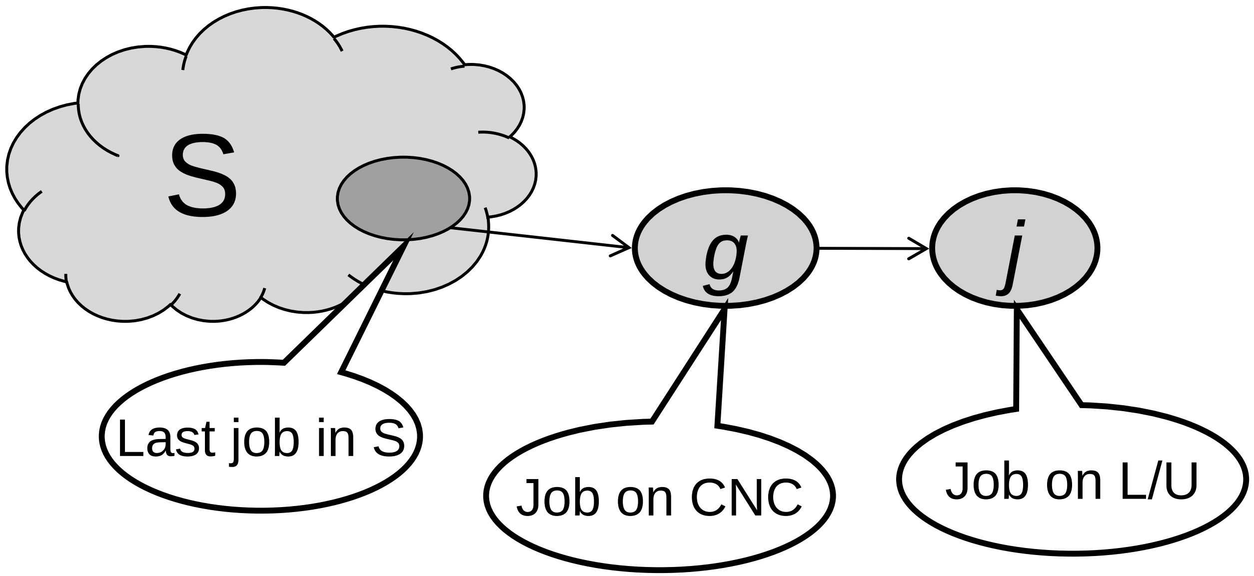


Common Unloading Times (con't)

- Given any two jobs in the first two positions, apply the G-G algorithm to obtain the minimum makespan corresponding to that pair of jobs
- The one with the smallest makespan among these $n(n-1)$ pairs of jobs forms an optimal sequence for the original problem
- The complexity of the G-G algorithm is $O(n\log n)$
- The complexity of the proposed algorithm is $O(n^3\log n)$

DP Algorithm (One)

Optimal value function (OVF): $f_i(S, g, j)$ = minimum completion time for processing job g on CNC, unloading the last job in S and loading job j at the L/U station, given that the i jobs in S have already been completed. (4.4)



DP Algorithm (One) (con't)

Recurrence relation (RR):

$$f_i(S, g, j) = \min_{k\in S}\{f_{i-1}(S\setminus\{k\}, k, g) + \max\{p_g, u_k + l_j\}\};\ i = 1,\dots,n-2 \quad ; \quad \{g, j\} \subseteq N \quad ;$$

$$S \subseteq N\setminus\{g, j\}\ ,\ |S| = i. \tag{4.6}$$

Boundary condition (BC): $f_0(\varnothing, g, j) = l_g + \max\{p_g, l_j\};\ \{g, j\} \subseteq N$. (4.7)

Answer (ANS):

$$\min_{g\subseteq N}\{f_{n-1}(S, g, \varnothing)\} \tag{4.8}$$

where $f_{n-1}(S, g, \varnothing) = \min_{k\in S}\{f_{n-2}(S\setminus\{k\}, k, g) + \max\{p_g, u_k\}\} + u_g \quad ; \quad g \subseteq N \quad ; \quad S = N\setminus\{g\}$,

$$|S| = n-1. \tag{4.9}$$

DP Algorithm (One) – Computational Analysis

Stage		Number of combinations	Addition	Comparison
Boundary condition ($i=0$)		$n(n-1)$	1	1
Recurrence relation ($1 \le i \le n-2$)		$n(n-1)C_i^{n-2}$	$2i$	$i+(i-1)$
Answer	f_{n-1} ($i=n-1$)	n	n	$n-1+n-2$
	Minimum makespan	1	0	$n-1$

Number of additions:

$$\approx n(n-1)(n-2)2^{n-2}$$

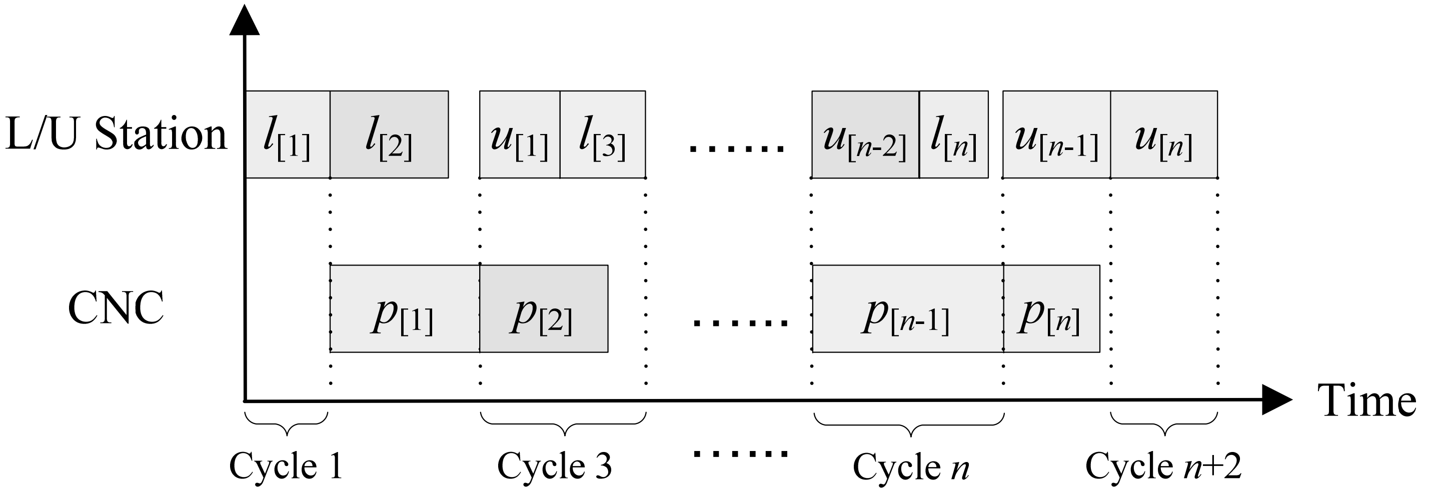
Number of comparisons:

$$\approx n(n-1)(n-2)2^{n-2}$$

$$O(n^3 2^{n-2})$$

Lower Bound (One)

- Lemma 7: a lower bound $\min_{i=1,\ldots,n} l_i + \sum_{j=1}^{n} p_j + \min_{i=1,\ldots,n} u_i$
- Lemma 8: a lower bound $\sum_{j=1}^{n} (l_j + u_j)$



- Theorem 2: $\max\{\min_{i=1,\ldots,n} l_i + \sum_{j=1}^{n} p_j + \min_{i=1,\ldots,n} u_i,\ \sum_{j=1}^{n} (l_j + u_j)\}$ is a lower bound

Two-phase Heuristic Algorithm

- Constructive phase:
  - Form a solution in a recursive manner by trying to insert one or more unscheduled job into one or more positions of a partial schedule until all jobs are sequenced
  - Two constructive heuristics are proposed:
    - Constructive Algorithm Selection (CAS)
    - Constructive Algorithm Insertion (CAI)

- Improvement phase:
  - Improve the quality of the solution iteratively from an initial sequence
  - Modified neighborhood search algorithm

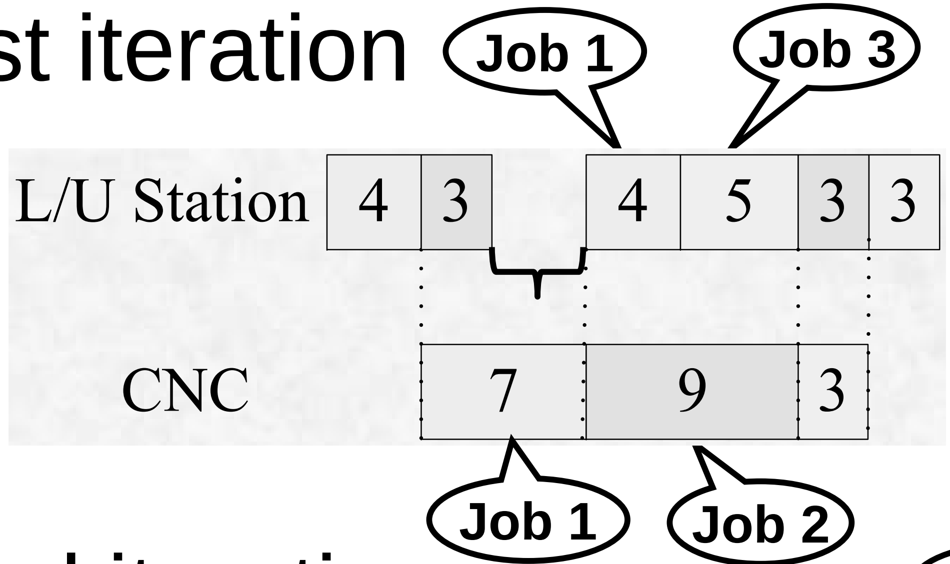
Constructive Algorithm: CAS Heuristic

- Select the summation of a job's loading time and a job's unloading time closest to a given processing time
- An example:

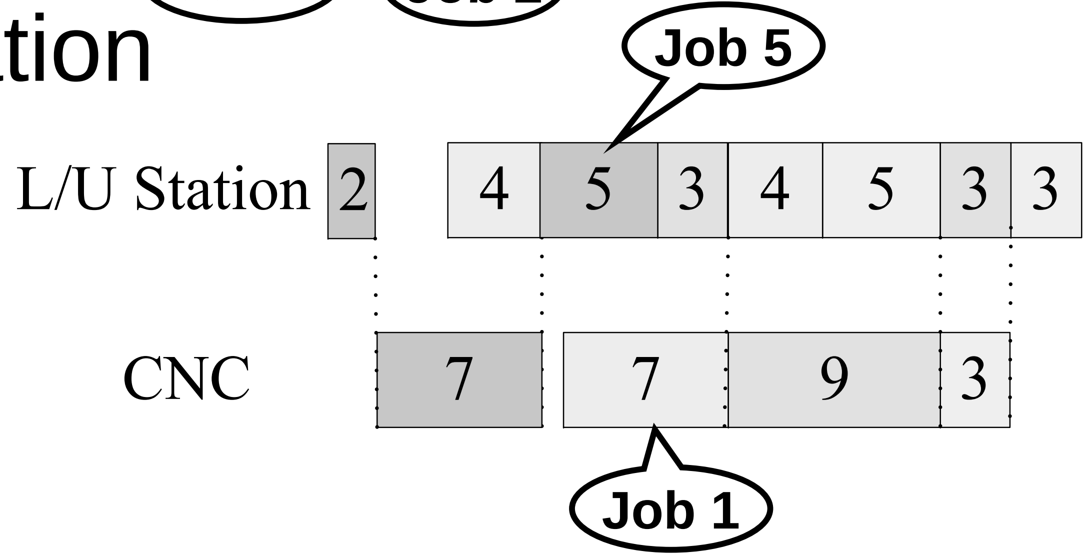
Job	l_j	p_j	u_j
1	4	7	4
2	3	9	3
3	5	3	3
4	2	5	1
5	2	7	5

Constructive Algorithm: CAS Heuristic (con't)

- 1st iteration



- 2nd iteration



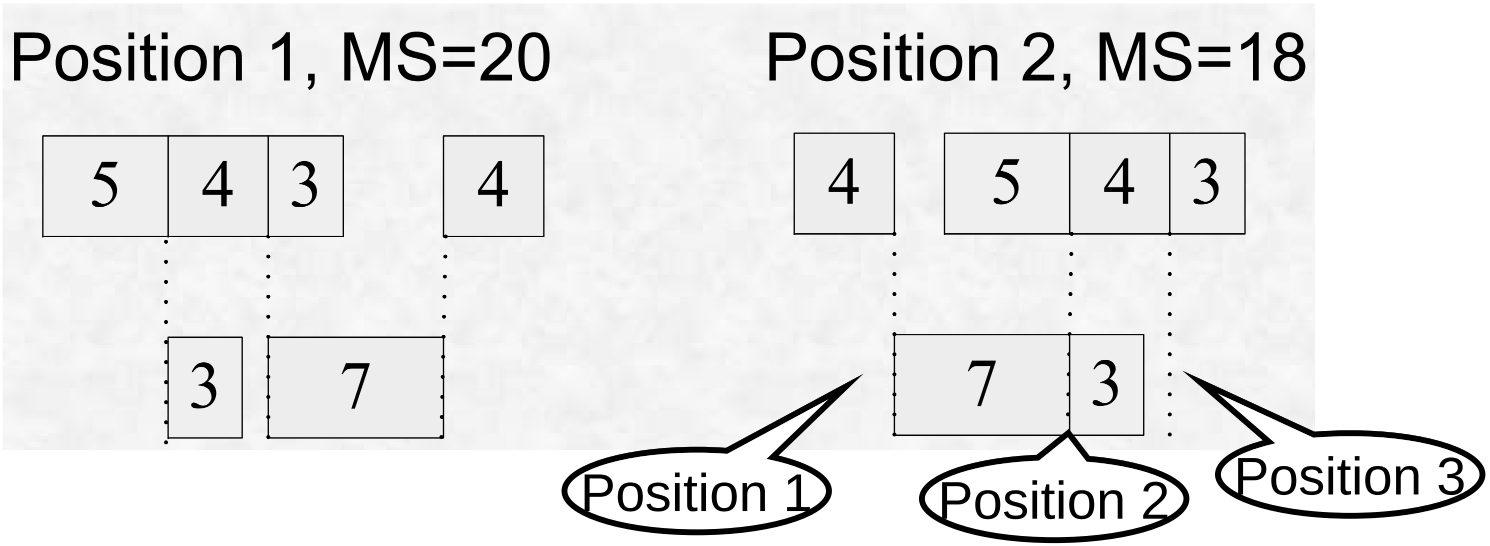
Constructive Algorithm: CAI Heuristic

- Each job has two weights associated with its loading and unloading times
- A larger loading time has a larger weight. The same rule applies to unloading times
- A job with larger sum of these two weights is selected first and it will be inserted in every position of the current sequence
- The selected job is sequenced in the position which yields a minimum makespan

Constructive Algorithm: CAI Heuristic (con't)

- An example:

Job	l_j	w_{lj}	p_j	u_j	w_{uj}	$w_{lj} + w_{uj}$
1	4	4	7	4	4	8
2	3	3	9	3	2	5
3	5	5	3	3	2	7
4	2	1	5	1	1	2
5	2	1	7	5	5	6



Improvement Phase

- Neighborhood search
  - Given an initial solution, generate a set of solutions based on the initial solution
  - Adjacent pairwise interchange
    - n-1 neighborhood solutions
  - Pairwise interchange
    - $n(n$-1)/2 neighborhood solutions
  - The neighbor with the best value is selected as the initial solution for next iteration
  - The search algorithm terminates when no better neighbors can be found

Modified Neighborhood Search Algorithm

- Random selection
  - If no better neighbors can be found, randomly select a neighbor whose makespan is equal to the current best makespan as the initial solution
  - The search algorithm terminates when the number of random selections performed is greater than a threshold
- Pairwise interchange

Computational Analysis

- Two two-phase heuristic algorithms
  - CAS_M
  - CAI_M
- Problem sizes: small, medium, large
- Three settings on operations times
  - $E(p) = E(l)+E(u)$ (E: expected value)
  - $E(p) > E(l)+E(u)$
  - $E(p) < E(l)+E(u)$

Testing Plan

- Nine cases and ten runs for each case

	Small size	Medium size	Large size
Number of jobs (n)	10	19	40
Scenario I (7, 11, 3)	(l_j, p_j, u_j) = (U(1,7), U(1, 11), U(1, 3))		
Scenario II (7, 15, 3)	(l_j, p_j, u_j) = (U(1,7), U(1, 15), U(1, 3))		
Scenario III (10, 11, 4)	(l_j, p_j, u_j) = (U(1,10), U(1, 11), U(1, 4))		

*U denotes the uniform distribution and all operation times are integer.

Computational Results

Scenario	n	DP		CAS_M			CAI_M			LB
		Optimum	Time	S1	S2	Time	S1	S2	Time	
I (7, 11, 3)	10	68.9	0.06	74.4	69.3	0.08	71.1	69.5	0.08	68.2
	19	121.8	344.8	136.0	122.6	0.26	126.0	123.1	0.29	119.6
	40	–	–	280.7	253.2	0.92	261.3	252.7	1.12	247.1
II (7, 15, 3)	10	82.2	0.06	86.8	82.4	0.08	83.4	82.6	0.08	80.7
	19	163.1	356.1	171.4	163.2	0.26	165.5	163.4	0.25	160.3
	40	–	–	333.1	314.7	1.03	321.7	314.8	1.02	310.4
III (10, 11, 4)	10	80.2	0.07	85.2	80.9	0.09	81.8	80.9	0.08	79.5
	19	155.2	354.4	162.0	155.6	0.27	157.6	155.8	0.26	155.2
	40	–	–	335.9	323.0	0.86	323.2	323.0	0.83	322.1

(S1: the constructive stage; S2: the improvement stage)

Computational Results (con't)

- Relative error (RE)

Scenario	n	RE from optimum (%)				RE from LB (%)			
		CAS_M		CAI_M		CAS_M		CAI_M	
		SI	S2	S1	S2	S1	S2	S1	S2
I (7, 11, 3)	10	8.12	0.64	3.30	0.97	9.27	1.72	4.41	2.05
	19	11.73	0.66	3.51	1.09	13.89	2.56	5.48	3.00
	40	–	–	–	–	13.82	2.50	5.80	2.29
II (7, 15, 3)	10	5.73	0.25	1.49	0.48	7.75	2.17	3.44	2.41
	19	5.37	0.09	1.65	0.24	7.63	2.20	3.83	2.37
	40	–	–	–	–	7.45	1.43	3.71	1.46
III (10, 11, 4)	10	6.17	0.93	2.07	0.89	7.19	1.91	3.09	1.86
	19	4.51	0.28	1.62	0.41	4.51	0.28	1.62	0.41
	40	–	–	–	–	4.29	0.29	0.36	0.29

Computational Results (con't)

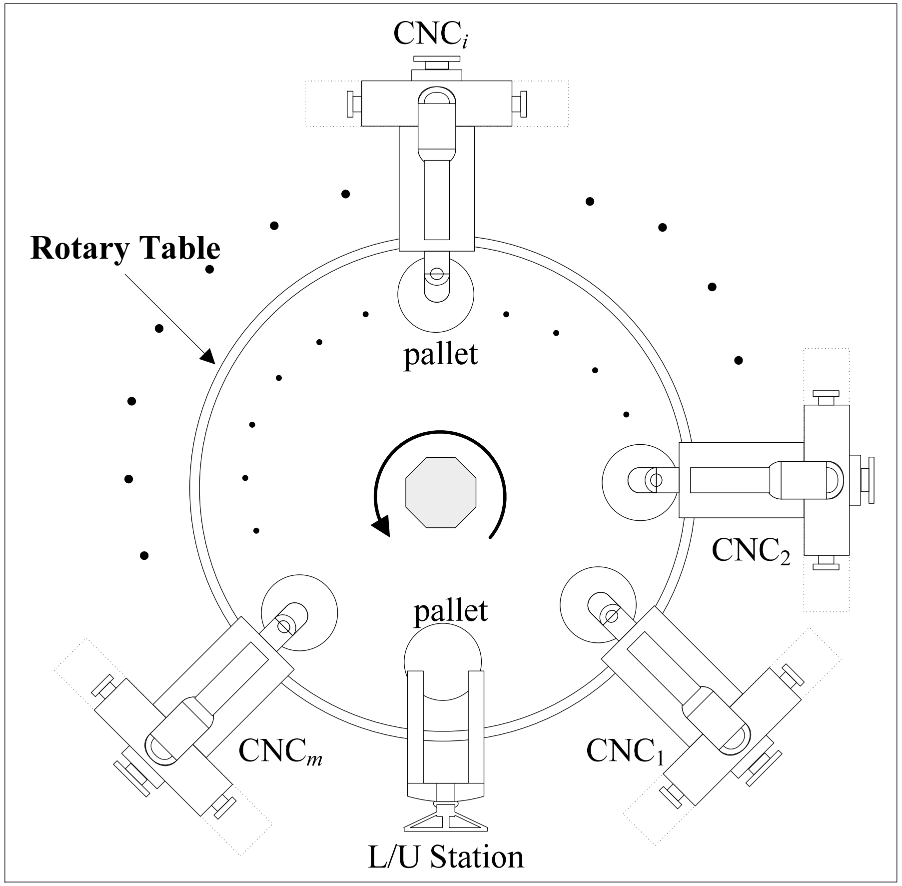
- Both algorithms (CAS_M, CAI_M) can obtain a solution in one second
- The quality of solutions obtained by CAS_M and CAI_M are similar
- Scenario I, on average, the solution from the optimum is 1.1% and from the lower bound is 3%
- CAI outperforms CAS

Agenda

- Introduction
- T-line machining center with one machine
  - Complexity
  - Common unloading times
  - Dynamic programming algorithm
  - Lower bound
  - Heuristic algorithms and computational results
- **T-line machining center with *m* machines**
  - Dynamic programming algorithm
  - Lower bound
- Conclusions and future research

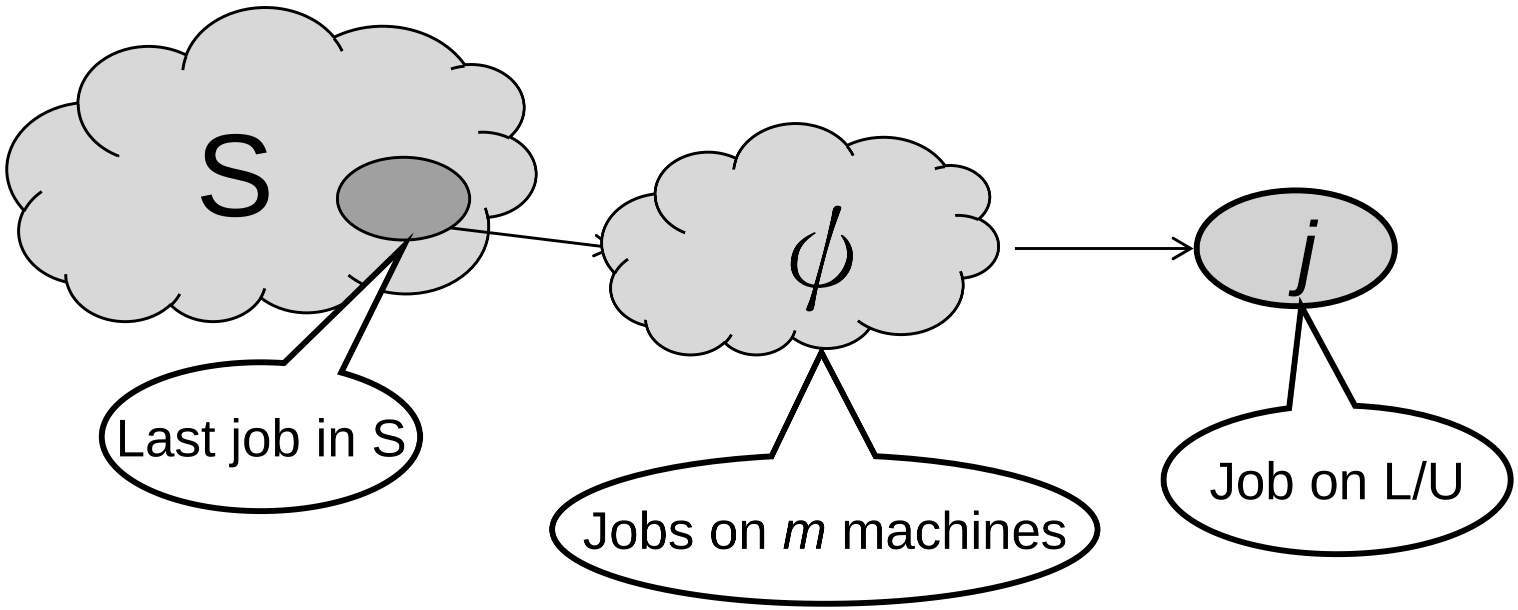
T-line Machining Center with *m* Machines

- The rotary table with *m*+1 pallets



DP Algorithm (*m*)

Optimal value function (OVF): $f_i(S, \Psi, j)$ = minimum completion time for processing jobs in Ψ on machines, unloading the last job in S and loading job j at the L/U station, given that the i jobs in S have already been completed. (5.10)



DP Algorithm (*m*) (con't)

Recurrence relation (RR):

$$f_i(S,\Psi,j)=\min_{k\in S}\{f_{i-1}(S\setminus\{k\},\{k\}\cup\Psi\setminus\Psi_{(m)},\Psi_{(m)})+\max\{p_{\Psi_{(m)}1},p_{\Psi_{(m-1)}2},\ldots,p_{\Psi_{(1)}m},u_k+l_j\}\};$$

$$i=1,\ldots,n-m-1;\ \{\Psi,j\}\subseteq N;\ S\subseteq N\setminus\{\Psi,j\},\ |S|=i. \qquad (5.12)$$

Boundary condition (BC):

$$f_0(\varnothing,\Psi,j)=\sum_{i=1}^{m}\max\{p_{\Psi_{(i-1)}1},p_{\Psi_{(i-2)}2},\ldots,p_{\Psi_{(i-m)}m},l_{\Psi_{(i)}}\}$$

$$+\max\{p_{\Psi_{(m)}1},p_{\Psi_{(m-1)}2},\ldots,p_{\Psi_{(1)}m},l_j\};\ \{\Psi,j\}\subseteq N. \qquad (5.13)$$

DP Algorithm (*m*) (con't)

Answer (ANS): $\min_{\Psi \subseteq N}\{f_{n-m}(S, \Psi, \varnothing)\}$ (5.14)

where $f_{n-m}(S,\Psi,\varnothing) = \min_{k\in S}\{f_{n-m-1}(S \setminus \{k\}, \{k\} \cup \Psi \setminus \Psi_{(m)}, \Psi_{(m)}) + \max\{p_{\Psi_{(m)}1}, \ldots, p_{\Psi_{(1)}m}, u_k\}\}$

$+\sum_{i=1}^{m} \max\{p_{\Psi_{(i+m)}1}, p_{\Psi_{(i+m-1)}2}, \ldots, p_{\Psi_{(i+1)}m}, u_{\Psi_{(i)}}\}$; $\Psi \subseteq N$; $S = N \setminus \Psi$, $|S| = n-m$. (5.15)

DP Algorithm (*m*) – Computational Analysis

Stage		Number of combinations	Addition	Comparison
BC ($i=0$)		$(m+1)!C_{m+1}^{n}$	m	$m(m+1)/2$
RR ($1\le i\le n-m-1$)		$(m+1)!C_{m+1}^{n}C_{i}^{n-(m+1)}$	$2i$	$m\times i+(i-1)$
ANS	$f_{n-m}(i=n-m)$	$m!C_{m}^{n}$	n	$m(n-m)+(n-m-1)+m(m-1)/2$
	Min makespan	1	0	$m!C_{m}^{n}-1$

Number of additions:

$$\approx \frac{n!}{(n-m-2)!}2^{n-m-1}$$

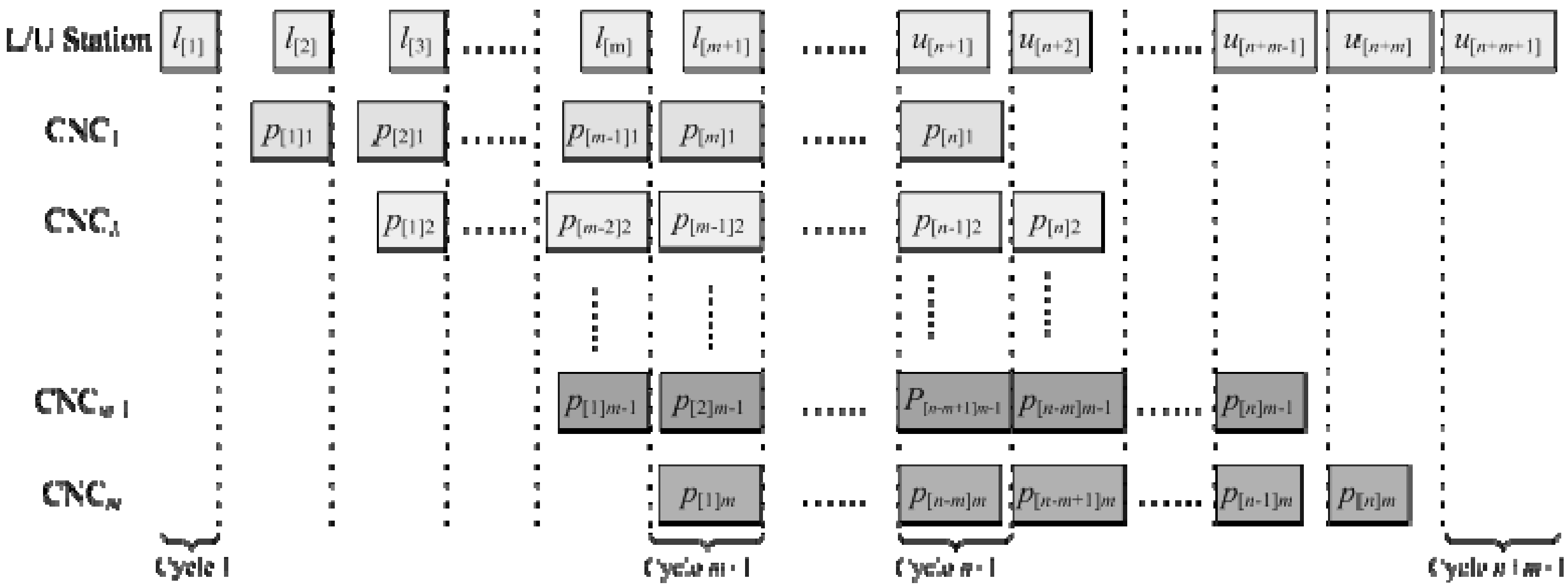
Number of comparisons:

$$\approx \frac{n!(m+1)}{(n-m-2)!}2^{n-m-2}$$

$$O(\frac{n!m}{(n-m-2)!}2^{n-m-2})$$

Lower Bound (*m*)

- Consider a relaxed problem
  - Neglecting loading and unloading times
  - Relax the job order processed by machines
  - Denote the optimal makespan of the relaxed problem as LB_R



Lower Bound (m) (con't) Algorithm for LB_R

- Sort processing times in descending order

Job	p_{j1}	p_{j2}	p_{j3}	p_{j4}
1	5	2	6	3
2	11	8	7	15
3	9	15	6	11
4	7	4	13	2
5	4	13	1	11
6	8	14	9	10
7	11	8	8	7
8	14	15	8	1

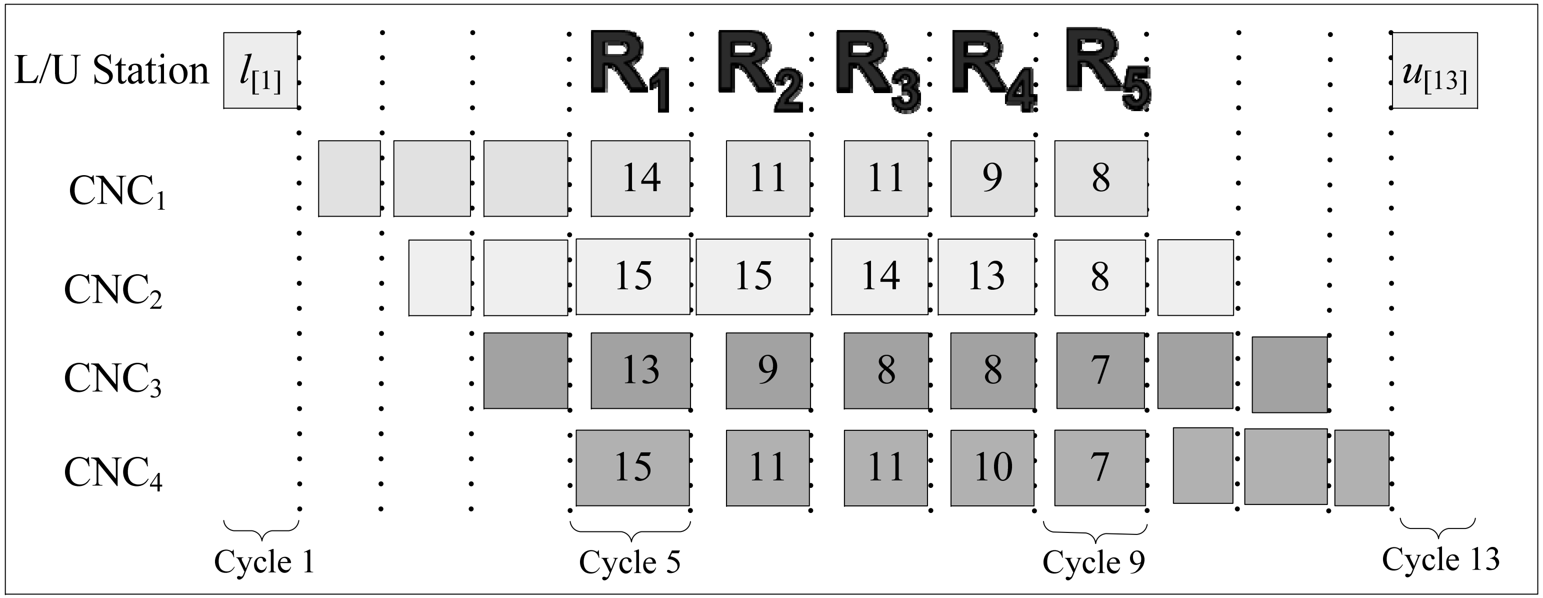
Largest processing times

Set	p_{j1}	p_{j2}	p_{j3}	p_{j4}
R_1	14	15	13	15
R_2	11	15	9	11
R_3	11	14	8	11
R_4	9	13	8	10
R_5	8	8	7	7
R_6	7	8	6	3
R_7	5	4	6	2
R_8	4	2	1	1

Smallest processing times

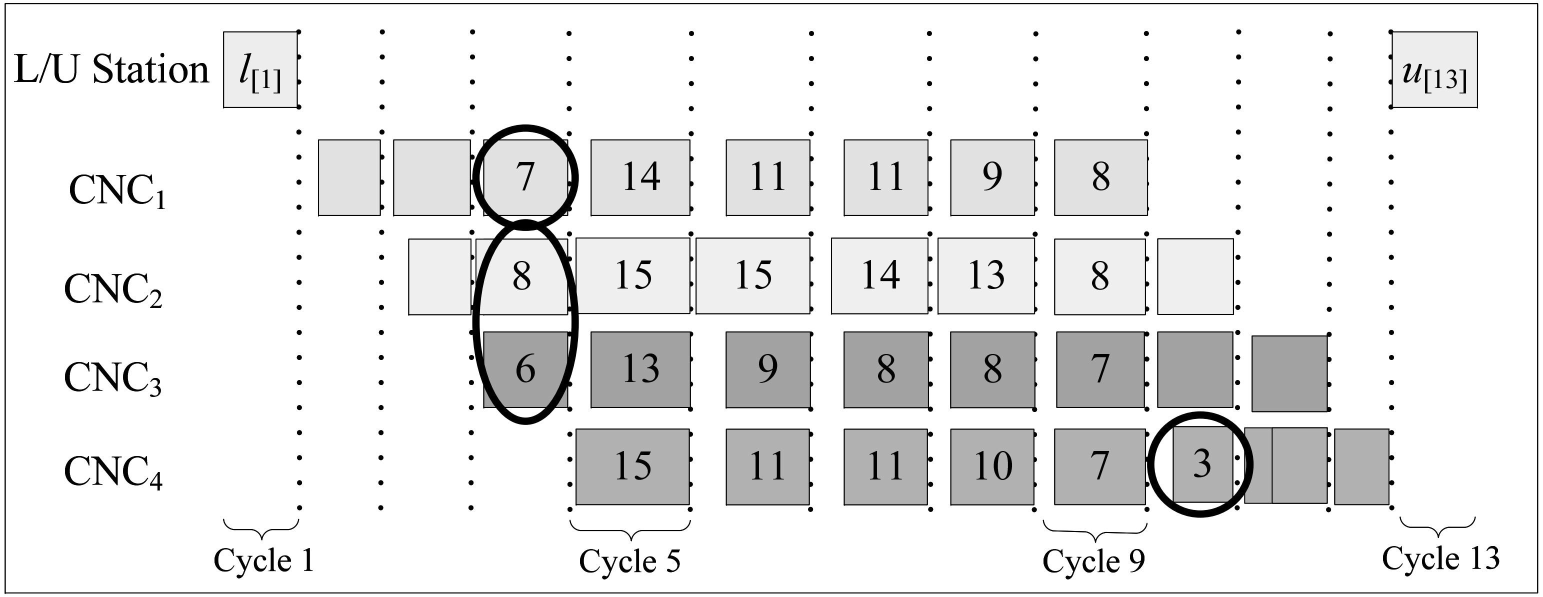
Lower Bound (*m*) (con't) Algorithm for LB_R

- Place the largest processing time on each machine to cycle 5, the second largest to cycle 6, and so on



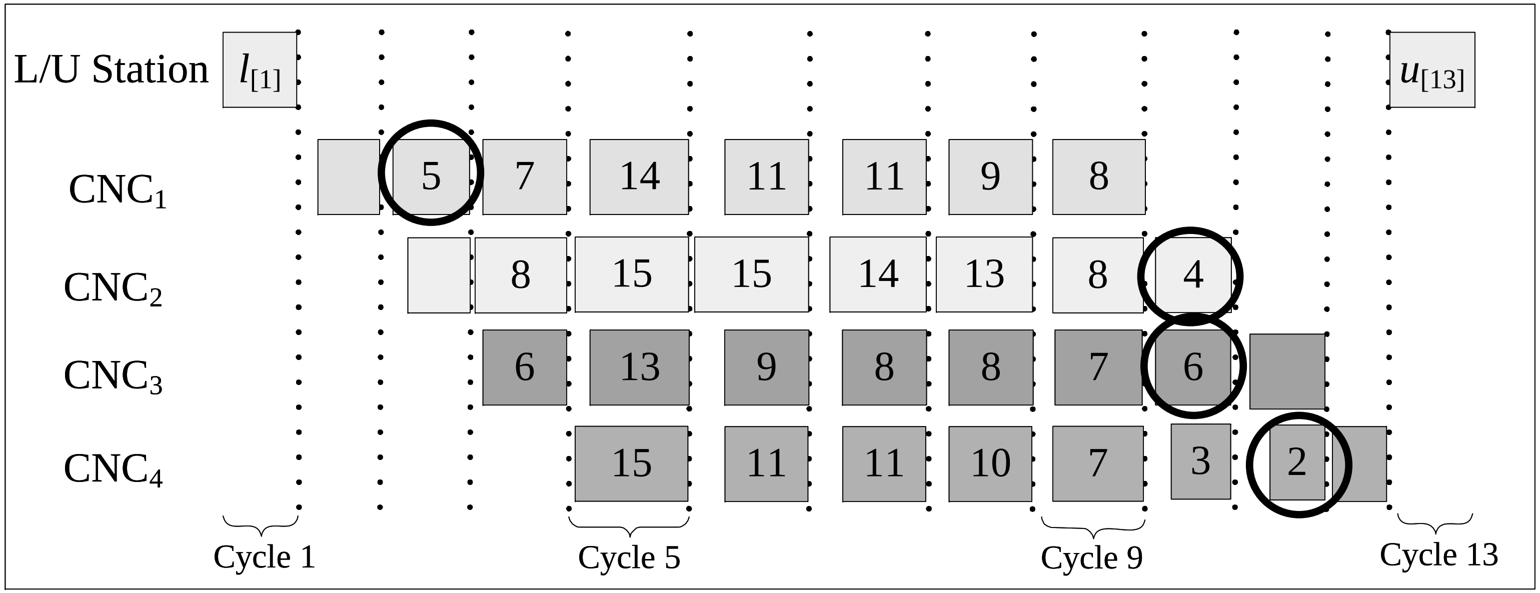
Lower Bound (*m*) (con't) Algorithm for LB_R

- Assign the processing time on machine 1 in R_6 to cycle 4
- Assign the processing time on machine 4 in R_6 to cycle 10



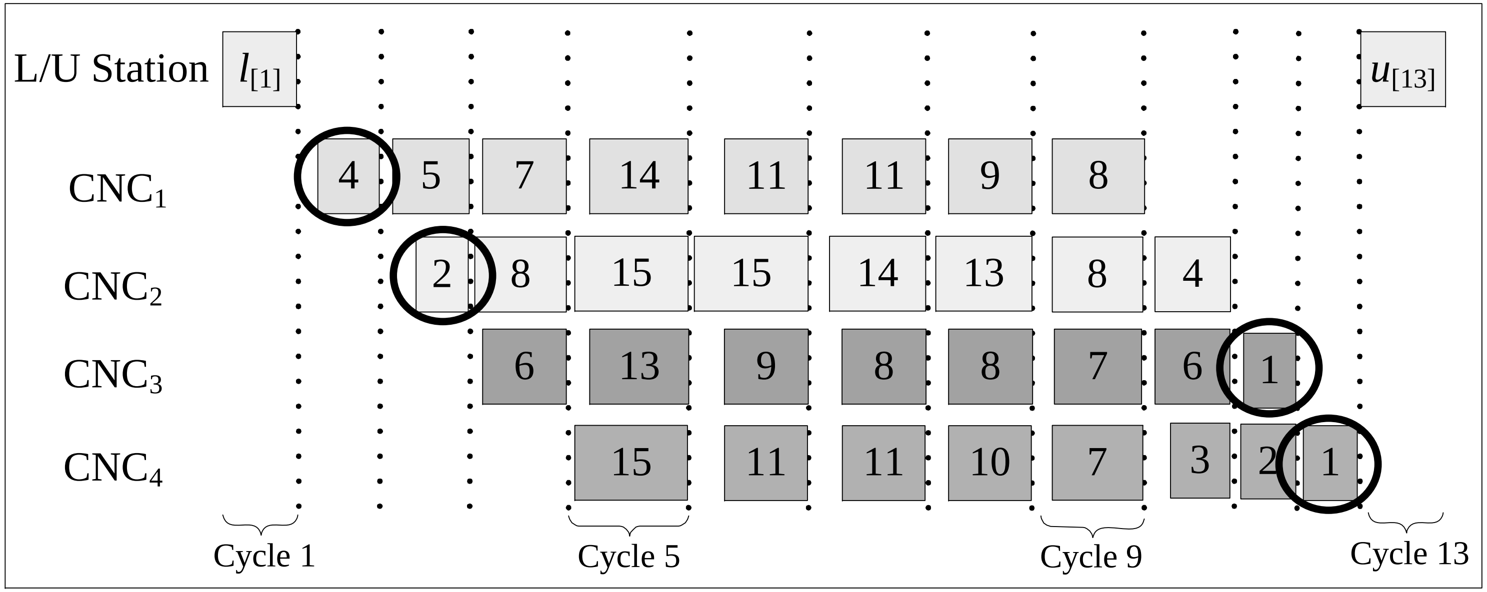
Lower Bound (*m*) (con't) Algorithm for LB_R

- Assign the processing time on machine 1 in R_7 to cycle 3
- Assign the processing times on machines 3 and 4 in R_7 to cycles 10 and 11, respectively



Lower Bound (*m*) (con’t) Algorithm for LB_R

- Assign the processing times in R_8 to each available cycle and the makespan is 91



Lower Bound (*m*) (con't)

- Lemma 9: LB_R is the optimal makespan for the relaxed problem
- Ideas of the proof: interchange any two processing times on a machine will not reduce LB_R
- Theorem 3: $\max\{\sum_{j=1}^{n}(l_j + u_j),\ \min_{i=1,\dots,n} l_i + LB_R + \min_{i=1,\dots,n} u_i\}$ is a lower bound

Agenda

- Introduction
- T-line machining center with one machine
  - Complexity
  - Common unloading times
  - Dynamic programming algorithm
  - Lower bound
  - Heuristic algorithms and computational results
- T-line machining center with *m* machines
  - Dynamic programming algorithm
  - Lower bound
- **Conclusions and future research**

Conclusions

- Synchronous transportation times:
  - The problem has been showed to be NP-hard
  - A DP algorithm has been proposed for a general case with *m* machines
  - Two-phase heuristic algorithms have been developed to obtain near optimal solutions in a very short time for the problems with one machine
  - Lower bounds are derived for the cases with one and *m* machines
  - An algorithm integrated the G-G algorithm has been provided to solve the one-machine problem with a common loading or unloading time

Future Research

- Mateheuristics
- Common processing times for the one-machine case
- For two-machine case
  - Unloading times are zero
  - Processing times on one machine dominate those on another machine
- Different objectives

Thank you.
Questions?

Lemma 1

The total processing time on the CNC machine is $Z-2$

$$\sum_{i=1}^{n}(p(Ja_i)+p(Jb_i)+p(Jc_i)) = 3n+n(\beta+1)+\sum_{i=1}^{n}(3\beta+3c_i)$$

$$= 4n+4n\beta+\sum_{i=1}^{n}3c_i = 4n+4n\beta+3C_0 = Z-2$$

Since the smallest loading and unloading times are equal to 1 respectively, the idle time can only occur at the first and last cycles on the CNC machine.

Lemma 2:

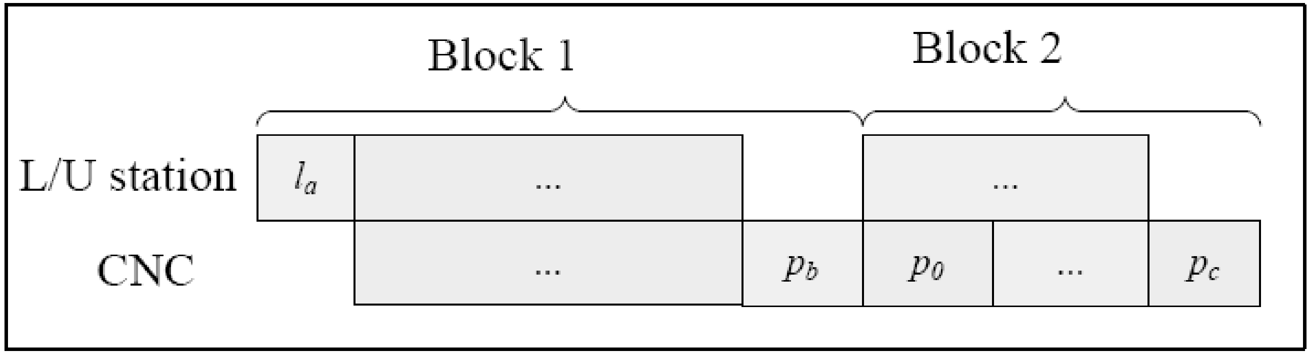
The total operation time on the L/U station is $Z-2$

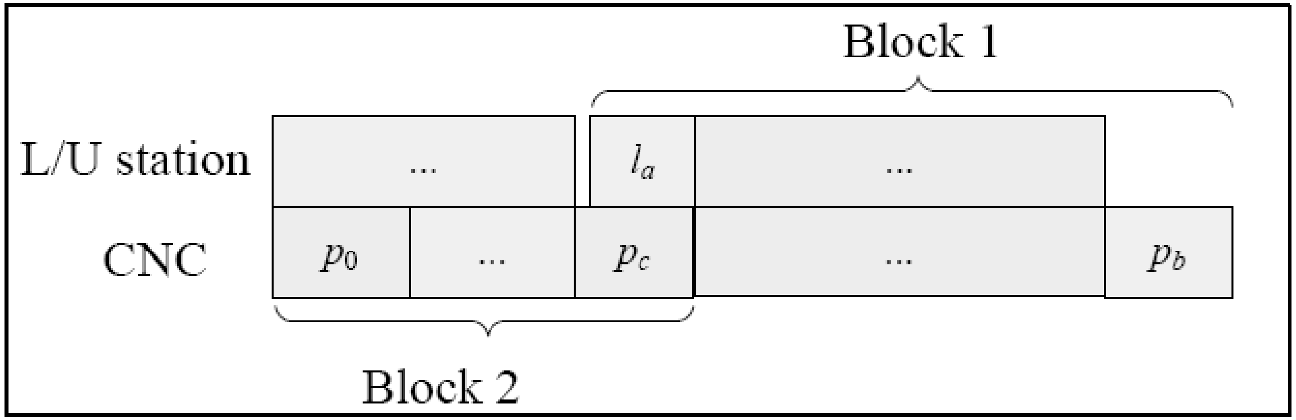
$$\sum_{i=1}^{n}[l(Ja_i)+l(Jb_i)+l(Jc_i)+u(Ja_i)+u(Jb_i)+u(Jc_i)]$$

$$=4n+4n\beta+3\sum_{i=1}^{n}(a_i+b_i)=4n+4n\beta+3C_0=Z-2$$

One unit of idle time can only occur in the second and second last cycles at the L/U station.

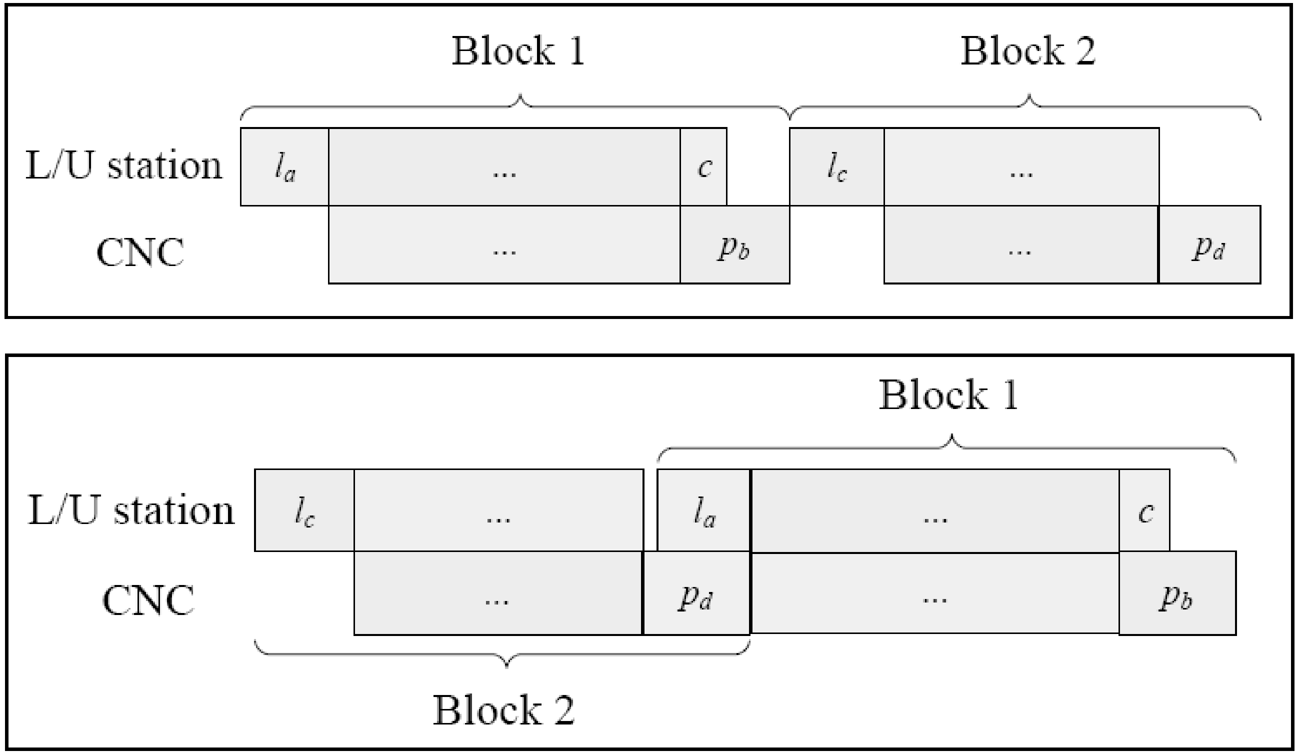
Lemma 5: J_0 is sequenced in the first position





$$Z^1 - Z^2 = l_a + p_b + p_c - \max(l_a,\, p_c) - p_b = l_a + p_c - \max(l_a,\, p_c) > 0$$

Lemma 6: J_{n+1} is sequenced in the last position



$$Z^1 - Z^2 = l_a + \max(c,\ p_b) + l_c + p_d - l_c - \max(l_a,\ p_d) - \max(c,\ p_b)$$

$$= l_a + p_d - \max(l_a,\ p_d) > 0\,.$$

Lemma 9

- Let these two processing times on machine *k* in cycles *r* and *s* be p_k^r and p_k^s, respectively.
- Assume that $C_r \le C_s$, then $p_k^r \le p_k^s$.
- Interchange these two processing times

Lemma 9 (con't)

- Assume that $C_r = p_k^r,\ C_s = p_k^s$
- After interchange,

$$C_r' = p_k^s = C_s \text{ and } C_r = p_k^r \le C_s'$$

$$\boxed{C_r + C_s \le C_r' + C_s'}$$

Lemma 9 (con't)

- Assume that $C_r = p_k^r,\ C_s > p_k^s$
- After interchange,

$$C_r \le C_r' = p_k^s \text{ and } C_s = C_s'$$

$$\boxed{C_r + C_s \le C_r' + C_s'}$$

Lemma 9 (con't)

- Assume that $C_r > p_k^r,\ C_s = p_k^s$
- After interchange,

$$C_r' = C_s = p_k^s$$

$$C_s' = \max\{p_k^r,\ \max_{f \neq k}\{p_f^s\}\}$$

$$\text{(without considering } p_k^s \text{ and } p_k^r,\ \max_{f \neq k}\{p_f^s\} \geq \max_{h \neq k}\{p_h^r\} > p_k^r)$$

$$C_s' = \max_{f \neq k}\{p_f^s\} \geq C_r$$

$$\boxed{C_r + C_s \leq C_r' + C_s'}$$

Lemma 9 (con't)

- Assume that $C_r > p_k^r,\ C_s > p_k^s$
- After interchange,

$$C_r^{'} = \max\{C_r,\ p_k^s\} \geq C_r \ \text{ and } \ C_s = C_s^{'}$$

$$\boxed{C_r + C_s \leq C_r^{'} + C_s^{'}}$$