Advanced Engineering Technology and Application
An International Journal

http://dx.doi.org/10.18576/aeta/060301

Flow Shop Programming of Robotic Cells with Job-dependent Transportation and Set – up Effects

A. P. Nagwai, Samson. Agboola[*] *and M. A Olusegun*

Department of Statistics, Ahmadu Bello University Zaria, Nigeria.

Received: 11 Feb. 2017, Revised: 2 Apr. 2017, Accepted: 5 Apr. 2017
Published online: 1 Sep 2017

Abstract: A flexible manufacturing cell consisting of two machining centers, several automated storage retrieval stations, and a mobile transporting robot is considered. The problem is to schedule jobs on machines so as to minimize the makespan, with the effects of transportation and set-ups to be taken into account. The problem is studied with the aid of a graph model, and an exact algorithm of cubic complexity is derived based on the Gilmore- Gomory algorithm for the travelling salesman problem

Keywords: programming; sequencing; modeling; networks and graphs; flexible manufacturing systems

1 Introduction

Small – scale flexible manufacturing cells with a few machines for parts processing and robots for their transportation are quite commonly found in real applications. Most of the multi-machine flowshop scheduling problems arising in this environment are known to be NP- hand, for which no exact algorithm running in reasonable computational time are found, except for very special cases. One of the well-solvable cases is the famous two-machine, n-job flowshop problem with unbounded inter-machine storage, which was studied by ‘Johnson (1954) who solved it in 0(n log n) time. Various modifications of the latter problem, taking into account the transportation and set-up times have been developed by many authors (Lawler *et al* 1993).

Another type of flowshop scheduling model describes the robotic cells having no intermediate storage between the machines for work-in-progress (‘no-WIP-storage’), as is often the case in industry. Two basic modifications of the no-WIP-storage model are known. The first one does not include input / output automated storage-and-retrieval stations (AS / RS). This modification has been investigated, among others, by Levner (1969), Panwalker (1991) and Stern and Vitner (1990)who have discussed its relations with the Johnson flowshop problem and the travelling salesman problem (TSP).

A more complicated modification of the no – WIP – storage scheduling model, involving AS/R stations, has been presented by Kise *et al.* (1991). They considered the two-machine one-robot scheduling problem and solved it in cubic time using the Gilmore – Gomory algorithm for TSP as a subroutine. This model was restricted to the case of a single AS/R station, job-independent transportation times, and set-ups included in processing times.

Kabadi and Fazle (1999) worked on Gilmore–Gomory type traveling salesman problems (TSP) where they implement the GG scheme in NP-hard for classes of TSP. However, they identify some subclasses of GG which the GG scheme can be implemented in time of polynomial. In the problems, they identify, generalize and unify polynomially testable and polynomially solvable subclasses of the TSP.

Carlier *et al.* (2010) investigate Optimization-Based Heuristic for the Robotic Cell Problem where problem arises in automated cells and is a complex flow shop problem with an single transportation robot and a blocking constraint by proposing an approximate decomposition algorithm were it breaks the problem into two scheduling problems that are solved sequentially: a flow shop problem with additional constraint (blocking and transportation times) and a single machine problem with precedence constraints, time lags, and setup times

Antonio *et al.*, (2014) worked on Production processes in Cellular Manufacturing Systems (CMS) often involve groups of parts sharing the same technological requirements in terms of tooling and setup, that is, issue of scheduling such parts through a flow-shop production

[*] Corresponding author e-mail: abuagboola@gmail.com

layout is known as the Flow-Shop Group Scheduling (FSGS) problem where they proposed an hybrid metaheuristic procedure integratingfeatures from Genetic Algorithms (GAs) and Biased Random Sampling (BRS) searchtechniques with the aim of minimizing the total flow time, *i.e.*, the sum of completion times of all jobs.

Sarah *et al.*, (2015), they considered the work on flexible flow shop scheduling with unrelated parallel machines at each , were the number of stages and machines vary at each stage and each machine can process specific operations. There proposed problem, that is , transportation of parts, loading and unloading parts are done by robots and the objective function is finding an optimal sequence of processing parts and robots movements to minimize the makespan and finding the closest number to the optimal number of robots. Their contribution of the study was to present the mixed integer linear programming model for the problem which considers release times for parts in scheduling area, loading and unloading times of parts which transferred by robots.

Yazdani and Naderi, (2017) worked on Modeling and Scheduling No-idle Hybrid Flow Shop Problems where they focus on considers the problem of scheduling no-idle hybrid flow shops, by developing a mixed integer linear programming model mathematically formulate the problem. Using commercial software, the model can solve small instances to optimality.

2 Description of the Robotic Cell

Consider a robotic cell consisting of two machining centres (machines), several input AS/RS and a single output AS/ RS

A given set J of n jobs is to be processed on the machines, M_A and M_B, in this order A job is loaded at one of the input AS/R stations and unloaded after its processing at the output AS / R station. Transportations between the input/output stations and a machine and between two machines are performed by a transporting robot. The same robot serves for loading / unloading the jobs

No machine has buffer storage for work-in-progress. The robot can transport only one job at a time. Each machine can process at most one job at a time and is not allowed to interrupt processing once it has started. The problem is to determine in which order (the same for both machines) the jobs are to be processed, and to find the robot's tours, so as to minimize the makespan

Let us consider a fixed order (a sequence) $P = (P_1, P_2, \ldots, P_n)$ of jobs from J to be processed on the machines. The jobs in P are processed according to the following technological rules.

Rule 1

1. The first job, P_1, is loaded on the robot at the AS/ RS station where is stored, and sent to machine M_A. After transporting job P_1, the robot loads it on machine M_A
2. Concurrently with the above transportation and loading, a tool is set on machine $M_A$for performing job P_1
3. Then machine M_A starts processing the first job, the robot unloads it from M_a, transports and loads it on machine M_B
4. Upon completing Job P_1 on machine M_A, the robot unloads it from M_A, transports and loads it on machine M_B
5. Concurrently with the above transportation and loading, a tool is set on machine M_B for performing P_1

Set k: = 1 and then apply Rule 2

Rule 2

1. Machine M_B receives job P_k and starts its processing. Concurrently, the empty robot moves to the input AS/ RS station where job P_{k+1} are stored. Loads it and transport it to machine M_A, then job $P_{k+1}$is loaded by robot on machine M_A. Concurrently with the transportation and loading, a tool on machine M_A is changed for performing job P_{k+1}, if necessary.
2. Machine M_A then starts processing P_{k+1}. Concurrently with processing $P_{k+1}$by machine M_A, the empty robot moves to machine M_B, waits there until job P_k is finished on machined M_B, unloads it, then transfers it to the output AS/RS and unloads it there.
3. The empty robot then moves from the output AS/RS to machine M_A, waits there if job P_{k+1} is not finished on M_A, and then unloads it from M_A, transfers and loads it on machine M_B.Concurrently with the transportation and loading, a tool on machine $M_B$is changed for performingjob P_{k+1}, if necessary

Set k : = k + 1, and , if $k < n$, apply Rule 2. If k = n, apply Rule 3

Rule 3

1. Machine M_B starts processing job P_n
2. The empty robot waits at machine M_B until the last job, P_n, is finished, unloads it and then transfers it to the output AS/ RS, where it loads. Stop

The key points of our model are that the transportation times are job-dependent and the work piece / tool set-ups are separated from processing operations.

Notation

$J = [1, 2, \ldots, n]$: the set of n jobs to be processed.

For $j = 1, \ldots, n$

$L_1(j)$ = the time required for the robot to load job j at the input AS/RS, where it is located;

$T_{1A}{}^1(j)$ = the transportation time needed for the robot to send job j from its input AS / RS to machine M_A.

$L_A(j)$ and $L_B(j)$ = the times spent loading job j on machine M_A and machine M_B respectively

$C_A(j)$ and $C_B(j)$ = the times to set (or change) a tool for processing job j on machines M_A and M_B respectively.

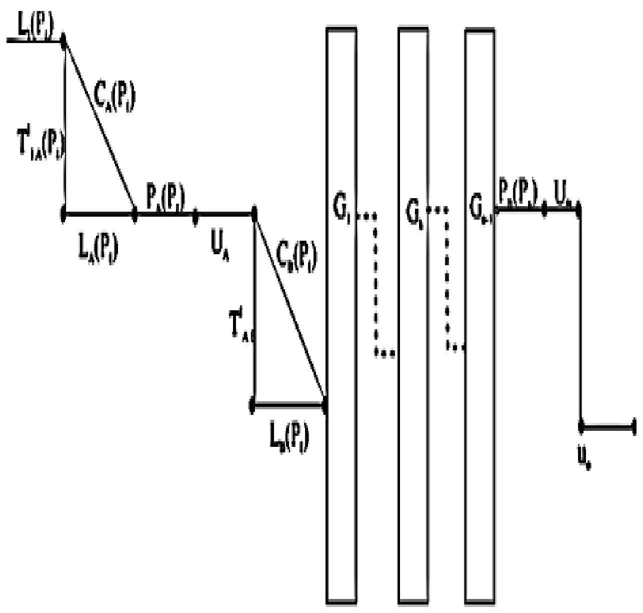

Fig 1. Graph G of the n-job technological process

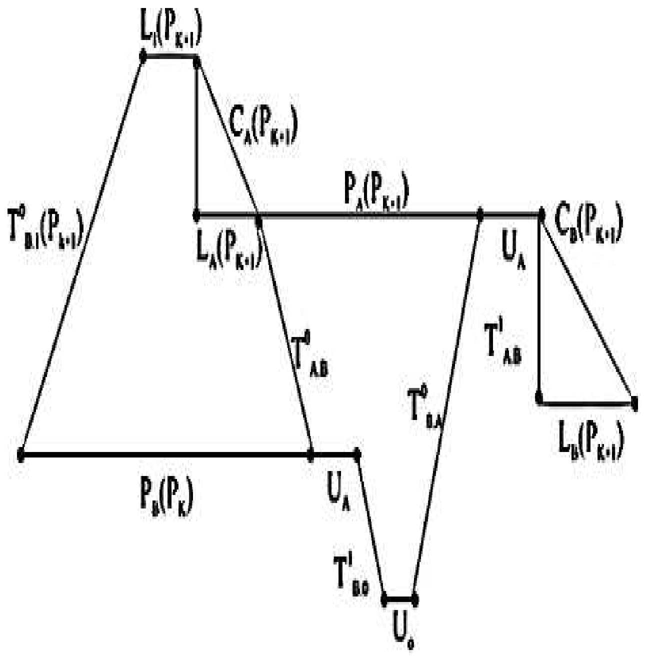

Fig 2. Subgraph Gk the kth Component of graph G

$P_A(j)$ $P_{k+1}$and $P_B(j)$ = the processing times of job j on machines M_A and M_B, respectively;

$U_A$and U_B = the times spent by the robot unloading any job from machines M_A and M_B respectively;

$T_A{}^1.B$ = the transportation time needed for the robot to deliver any job finished on machine M_A for processing on M_B;

$T_{A.B}{}^0$ = the transportation time needed for the empty robot to run from machine M_A to M_B;

$T_{B.1}{}^0(j)$ = the transportation time for the empty robot to run from M_B to that input AS/ RS where job j is stored;

$T_{B.0}{}^1$ = the transportation time for the robot to deliver the finished job from machine M_B to the output AS/ RS;

U_0 = the time required to unload any job at the output AS/RS;

$T_{0.A}{}^0$ = the transportation time needed for the empty robot to run from the output AS/RS again to machine M_A to serve a job finished at that machine.

3 A Graph Model of the Technological Process

Consider the graph representation of the technological process subject to Rules 1-3, corresponding to a fixed sequence P = ($P_1, P_2, \dots, P_n$) (see Figure 1)

The following statement can be proved by induction on n. For a fixed p, the makespan, F(P), is equal to the length of the maximal path in the graph G which comprises three parts: initial (denoted as G_0), middle ($G_1, \dots, G_{n--1}$), and terminal, G_n.

The proof is omitted as being analogous to that presented by Levner (1954) and Kise *et al*(1991). In our notation, the length of the maximal path in the initial part, G_0, of G is equal to time $b(p_1)$, needed to fulfill all the operations described in Rule 1 steps (a) – (e), and is a follows:

$b(p_1) = L_1(p_1) + \max\{T^1{}_{1.A}(p_1) + L_A(p_1).\ C_A(p_1)\} + P_A(p_1) + U_A + \max\{T_{A.B}{}^1 + L_B(p_1)\ .\ C_B(p_1)\}$ (**??**)

The middle part of G includes n – 1 identical sub graphs $G_1, G_2, \dots, G_{n--1}$, one of which is depicted in Figure 2

FIG. 2. Sub graph G_k, k = 1, . . . , n – 1 is equal to time $t(p_k)$ needed to fulfill the operations described in Rule 2, Steps (a) – (c), at iteration k, and is as follows:

$t(P_k) = \max\{W^1{}_{k+1}, W^2{}_{k+1}, W^3{}_k\} + U_A + \max\{T^1{}_{A.B} + L_B(p_{k+1}), C_B(p_{k+1})\}$. (2)

Where

$W^1{}_{K+1} = T^0{}_{B.1}(p_{k+1}) + L_1(p_{k+1}) + \max\{T^1{}_{A.B}(p_{k+1}) + L_A(p_{k+1}).\ C_A(p_{k+1})\} + P_A(p_{k+1})$;

$W^2{}_{K+1} = T^0{}_{B.1}(p_{k+1}) + L_1(p_{k+1}) + \max\{T^1{}_{1.A}(p_{k+1}) + L_A(p_{k+1}).\ C_A(p_{k+1})\} + T^0{}_{A.B} + U_B + T^1{}_{B.0} + U_0 + T^0{}_{0.A;}$

$W^2{}_k = P_B(p_k) + U_B + T^1{}_{B.0} + U_0 + T^0{}_{0.A}$

The length of the maximal path in the middle part of G is equal to the time, $t(p_1, .., p_{n--1})$, needed to fulfill all the operations described in Rule 2, Steps 2 (a) – (c), for all k; it is as follows:

$t(p_1, \dots, p_{n--1}) = \sum(p_k)$ (3)

The time $f(p_n)$, needed to fulfill the final operations described in Rule 3, steps (a) – (b), is:

$f(p_n) = P_B(p_n) + U_B + T'{}_{B.0} + U_0$ (4)

Therefore, the time to complete all n jobs in sequence p, is:
F(p) = b(p_1) + t(p_1, .., p_{n--1}) + f(p_n) (5)

4 Analysis of the Objective Function

We start two evident observations.

1. The optimal solution of the problem (i.e. the sequence minimizing (5) does not change if we subtract from F(P) the following constants, c, entering f(p_n) : c = U_B + $T'_{B.0}$ + U_0
2. The optimal solution of the problem does not change if we subtract from F(P) the following quantity, q, not depending on P:

q = ∑[U_A + max {$T^1_{A.B}$ + $L_B(p_k)$, $C_B(p_k)$}],
in which the first term, U_A + max {$T^1_{A.B}$ + $L_B(p_1)$, $C_B(p_1)$}, enters b(p_1) (see (1)) while all other components are from t($P_1, \dots, p_{n-1}$) (see (3) and (**??**))
Objective function (5), after the above subtractions, will be:
F'(P) = b'(p_1) + t'($p_1, \dots, p_{n--1}$) + f'(p_n)
where:
b'(P_1) = $L_1(P_1)$ + max{$T^1_{1.A}(p_1)$ + $L_A(p_1)$. $C_A(p_1)$} + $P_A(p_1)$ (6)
t'($p_1, \dots, p_{n--1}$) = ∑max{ W^1_{k+1} + W^2_{k+1} + W^3_{k+1}}; (7)
f'(p_n) = $p_B(p_n)$ (8) Let us denote:
A(P_k) = max{$W^1_{k,}$ W^2_k}; B(p_k) = W^3_k, k = 1, .., n (9)
Then we find that minimizing F(P) in (5) is equivalent to finding the permutation minimizing the following function:
F'(P) = b'(p_1) + ∑ max (A(P_{k+1}), B(P_k) + f'(p_n) (10)
The latter problem is reducible to a special case of the travelling salesman problem studied by Gilmore (1985) and Gomory (1991), denoted GGP. The proof is given in the Appendix
Thus, one evident way to minimize (10) is to calculate, for each pair ($p_{1,}P_n$), the value of
b'(p_1) +min ∑ max (A(P_{k+1}), B(P_k) + f'(p_n)
and to choose the minimal among the resulting n (n – 1) values.
Since the GGP may be solved in O(n log n) time. In fact, this scheme may be modified to be performed in essentially cubic time. In what follows, we will consider this modified scheme, not detailing the steps, which are the same as those of the Gilmore – Gomory Algorithm. GGA

5 Description of the Algorithm

The input of the modified scheme is n + 1 pairs of numbers corresponding ton + 1 cities in the Gilmore – Gomory case of the TSP: n pairs, (A(i), B(i), 1 ≤ i ≤ n, defined by (9) and renumbered so that B(**??**) ≤ B(**??**) ≤ . . . ≤ B(n), and an auxiliary pair of numbers. (A(n + 1), B(n +1) = (K, K), where K ≥ max ($A_1, A_2, \dots, A_n, B_1, B_2, \dots, B_n$)
Step 1 of the modified scheme is the same as the first step in GGA (1991): to arrange the A(i) values in non-decreasing order and or find the permutation ϕ defined by ϕ(j) = q, 1 ≤ i ≤ n, q being such that A(q) is the jth smallest of the A(i)
At step 2, for each pair of indices, I and j, I ≠ j, I, j = 1, . . . , n, we find a partial sorting ϕ_q, which is obtained from the sorting at step 1 by excluding A(ϕ(i)) and A(ϕ(j)).
Step 3, 4 and 5 of our algorithm reproduce three main steps of GGA (1991), namely 'forming an auxiliary graph', finding a maximum spanning tree and finding an optimal permutation respectively. The only difference from GGA is that these steps are repeated for all $\phi_{i,j}$; that is, they are performed n(n – 1) times in turn, for each pair of indices, I and j, i ≠ j, I, j = 1, . . . , n, being excluded
Denote the optimal permutations obtained at step 5 by $P^*_{I,j}$
At step 6, for each optimal permutation, $P^*_{I,j}$, I, j = 1, . . . , n, i ≠ j , compute makespan F($P*_{I,j}$) = b'(i) + f'(j) + t'($P^*_{i,j}$), where b'(i), f'(j) and t'($P^*_{I,j}$) are defined by(6) -(8)
Find permutation P^* minimizing F($P^*_{I,j}$): F'(P^*) = $\min_{i,j}$F($P^*_{I,j}$)
The crucial point is that the sorting at step 1 is performed only once, requiring 0 (n log n) time. For each of n(n – 1) fixed pairs of indexes, Step 2, 3 and 5 run in linear time, and Step 4 in time 'essentially linear' in n (see Gilmore *et al* (1985) , Gabow *et al* (1986).). Step 6 runs in quadratic time. Thus, the total running time of the modified algorithm is essentially cubic.
Notice that if we apply the same structuring of the GGA for solving a special case of the scheduling problem in which the transportation times are job-independent (see Kise *et al.*, 1991) , a version of the modified algorithm running in essentially quadratic time can be readily obtained
An Illustrative Example
In a numerical example below we compare our scheduling model that takes into account job-dependent transportations and set-ups (denoted *Model 1*), with a more rough Model II in which the set- ups are included into the processing times. The makespans computed for an arbitrary schedule, P, in the two models are denoted, respectively, F_I (P) and F_{II}(P)
Table 1: Time parameters for the sample computations (comprising seven jobs)

Job	L_1	$T^1_{1.A}$	L_A	L_B	C_A	C_B	P_A	P_B	$T^0_{B.1}$
1	0.2	1.9	0.4	2.6	0.1	3.9	2.4	5.7	1.1
2	0.8	1.9	0.5	0.1	2.3	0.3	3.5	9.6	1.1
3	0.4	2.2	0.4	0.4	0.1	0.9	0.3	0.9	0.9
4	0.5	1.9	0.6	0.3	2.5	2.2	4.6	2.5	1.1
5	0.6	2.2	0.4	1.1	0.2	0.3	5.6	1.1	0.9
6	0.2	2.2	0.1	0.4	0.1	1.1	0.5	4.1	0.9
7	0.7	1.9	0.8	0.6	2.5	1.8	3.0	3.8	1.1

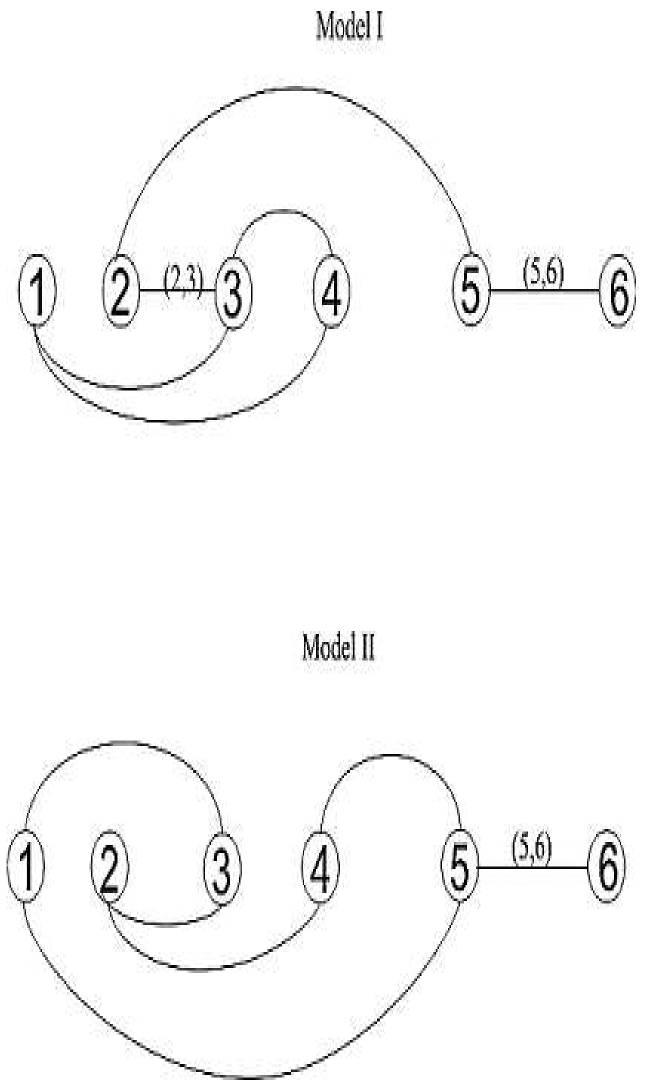

Fig. 3 Graphs Models I and II

In Table 2, those data are transformed, by formulae (6) - (8), into integrated parameters A(j), B(j), b'(j) and f'(j) which will enter F'(P) in (10)

For this numerical example, the results of step of the suggested algorithm, for the fixed pair (5, 6), are presented in Table 3

The auxiliary parameters, A(6) and B(6), are chosen to be k = 15. The results of step 4 are sets of arcs, T_1 and $T_{2,}$ which are to be added to the auxiliary graphs obtained at step 3. They are, corresponding, T_1 = {(5, 6), (2, 3)} for model I, and T_2 = {(5,6)} for Model II. The resulting graphs are depicted in Figure 3

The optimal permutations obtained at Step 6 run out to be different for the two models. They are: P_1 = (2-5-3-6-7-1-4) for Model I, and P_2 = (2-7-5-6-3-1-4) for Model II, with the makespan values, respectively: $F_I(P_1)$ = 76.0, and $F_{II}(P_2)$ = 83.4

We can measure the transportation / set-up effects in flowshop scheduling by estimating relative differences of the optimal makespan values in Models I and II, the so-called modeling errors

TABLE 2: Integrated Parameters for the two models

Job	B	A	f'	b'
Model I				
1	7.9	6.8	5.7	4.9
2	11.8	7.8	9.6	6.7
3	3.1	7.1	0.9	8.3
4	4.7	8.7	2.5	7.6
5	3.3	9.7	1.1	8.8
6	6.3	6.6	4.1	3.0
7	6.0	7.7	3.8	6.4
Model II				
1	14.4	6.4	13.2	5.0
2	12.2	10.1	10.0	9.0
3	4.4	6.7	2.2	3.4
4	7.2	11.2	5.0	10.1
5	4.7	9.9	2.5	9.0
6	7.8	6.5	5.6	3.1
7	8.4	10.0	6.2	8.9

TABLE 3: Results of Sorting

Job	B	A	q
Model I			
6	15.0	15.0	6
5	7.9	9.7	2
4	6.3	7.7	3
3	6.0	7.1	1
2	3.3	6.8	5
1	3.1	6.6	4
Model II			
6	15.0	15.0	6
5	14.4	10.0	4
4	8.4	9.9	2
3	7.8	6.7	1
2	4.7	6.5	3
1	4.7	6.4	5

1.The error $e_1$relative, time losses incurred by using Model II: e_1 = [$F_1(P_1)$ – $F_n(P_2)$] / $F_1(P_1)$; in our example, $F_I(P_1)$ = 76 versus $F_{II}(P_2)$ = 83.4 produces error e_1 = 10%

2. The error e_2 measures the roughness of criterion F_n of Model II in comparison with the corresponding criterion. F_1 of Model I, which is defined as $e_2 = [F_1(P_1) - F_n(P_1) / F_1(P_1)$; in our example, $F_1(P_1) = 76$, $F_n(P_1) = 84.4$, and error $e_2 = 11\%$

The example shows that, even in the case of small n, including the transportation set – up times into processing times may lead to essential errors. This effect was steadily observed in our computational experiments for various randomly-generated small and medium –sized instances ($n \leq 60$). The same effect is also observed when the transportations / set – up times are averaged or ignored[10]

6 Concluding

Since transportation and set-up operations are expected to consume a large portion of the production time in robotic cells, ignoring or averaging the transportation / set – up times may lead to considerable time losses and noticeable errors in determining the optimal makespan. On the contrary, OR models that explicitly introduce job-dependent transportation / set-up operations, and investigate how the latter are to be scheduled in order to decrease the makespan, can enhance the productivity of the cell. In this paper we have presented such an OR model for a small scale flexible manufacturing cell with job-dependent processing and material handling operations. In this paper, we study the two-machine, no-WIP-storage robotic cell with AS / RS. We introduce several AS/RS (of special importance when processing a large number of jobs), and show that the resultant scheduling problem with job –dependent transportation times may be exactly solved in cubic time using a delicate structuring of the Gilmore – Gomory algorithm.
Using a graph model of the technological process, we solved exactly, in polynomial time, the two-machine, one-robot, no- WIP-storage scheduling problem. We believe that the graph- based approach used in this paper can be extended to schedule efficiency more general robotic cells with several machines and robots

Appendix

Proposition
Minimizing $t'(P_1, \ldots, P_n)$ in (**??**) is equivalent to funding the minimal tour in the Gilmore – Gomory case of the travelling salesman problem, GGP
Proof
Consider the following problem, GGP
GGP: Given n cities $\{1, 2, \ldots, n\}$. Two non-negative numbers, A_i and B_i, are assigned to each city, $i = 1, \ldots, n$
The distance between city I and city j ($I, j = 1, \ldots, n$; $i \neq j$) is defined as follows:
$D(i, j) = \max(A_i, B_j)$
The length of a tour, $T = (p_1, p_2, \ldots, p_{n,} p_1)$, is:
$L(T) = \sum\max(B_{P(0)}, A_{p(I+1)}) + \max(B_{p(n)}, A_{p(\textbf{??})}$ A(**??**)
The problem is to find a tour of minimal length
Consider the GGP with the traditional, (n + 1)th, city with parameters $(A_{n\ +1}, B_{n+1})$ such that $A_{n\ +1} = B_{n+1} = K$, where $K \geq \max(A_1, \ldots, A_n, B_1, \ldots, B_n)$.
The length of an arbitrary tour, $T = (P_{n+1}, P_{1,}P_{2,} \ldots, P_n, P_{n+1})$, according to (A1) is
$L(T) = \sum\max(B_{P(i)}, A_{p(I+1)}) + 2K$
Since 2K is a constant and does not depend on T, the problem of minimizing L(T) is equivalent to that of minimizing $t'(p_1, \ldots, p_{n-1})$ appearing in (**??**), as claimed

References

[1] S.M Johnson (1954) Optimal two-and three-stage production scheduling with setup times included. Naval Res. Logiest. Q.I, 61-68
[2] E. L lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shmoys (1993) Sequencing and scheduling: algorithms and complexity. In Logistics of Production and inventory. Handbooks in Operations Research and Management Science, Vol.4. (S. Graves, A. Rinnooy Kan and P. Zipkin, Eds). 231- 250. North Holland. Amsterdam
[3] E. V. Levner (1969) optimal planning of parts machining on a number of machines. Automation and Remote Control II, 1972 – 1979
[4] S.S. Panwalker (1991) Scheduling of a two – machine flowshop with travel time between machines J. Opl Res. soc. 42, 609 – 613
[5] H. I. Stern and G. Vitner (1990) scheduling parts in a combined production-transportation work cell. J. Opl Res. Soc 41, 625 -632
[6] H. Kise, T. Shihoyama and T. Ibaraki (1991) Automated two-machine flowshop scheduling: a solvable case. HE Trans.23(1), 10-16
[7] P.C. Gilmore, and R.E. Gomory (1991) sequencing a one state-variable machine: a solvable case of the travelling salesman problem. Opns Res. 12, 655-679
[8] P. C. Gilmore, E. L. Lawler and D. B. Shmoys (1985) well-solved special cases. In the Travelling Salesman Problem: A Guided Tour of Combinatorial Optimization. (E.L. Lawler, A. H. G. Rinnooy Kan and D. B. Shmoys. Eds) pp.87-143. Wiley, Chichester
[9] H. W. Gabow, Z. Galil, T. Spencer and R. E. Tarjan (1986) Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. *Combinatorika* 6, 109-122
[10] E. Levner (1991) A faster algorithm for the two-machine scheduling problem with a robot. Technical Report, department of Statistics, Hebrew University of Jerusalem
[11] Seyedeh Sarah Zabihzadeh, Javad Rezaeian (2015) "Two meta-heuristic algorithms for flexible flow shop scheduling problem with robotic transportation and release time" *Applied Soft Computing* 40 319–330
[12] Antonio Costa, Fulvio Antonio Cappadonna and Sergio Fichera (2014) "A Hybrid Metaheuristic Approach for Minimizing the Total Flow Time in A Flow Shop Sequence Dependent Group Scheduling Problem" *Algorithms 7*, 376-396

[13] Jacques Carlier, Mohamed H aouari, Mohamed K harb eche, Aziz Moukrim.(2010) "An OptimizationBased Heuristic for the Rob otic Cell Problem". *European Journal of Operational Research*, *Elsevier*, 202 (3), 636-645.

[14] Mehdi Yazdani and Bahman Naderi (2017) "Modeling and Scheduling No-idle Hybrid Flow Shop Problems" *Journal of Optimization in Industrial Engineering 21 59-66*

[15] S.N. Kabadi and Md. Fazle Baki. (1999) "Gilmore–Gomory type traveling salesman problems" Computers and Operations Research 26(4), 329-351

Ambi Polycarp Nagwai Professor at Department of Statistics, Ahmadu Bello University Zaria, Nigeria

Samson Agboola B.Sc. Statistics - University of Jos, M.Sc. Statistics - Department of Statistics, Ahmadu Bello University Zaria and Current running Ph.D. Statistics - Department of Statistics, Ahmadu Bello University Zaria, Nigeria

Akinwande Michael Olusegun B.Sc. Statistics and M.Sc. Statistics- Department of Statistics, Ahmadu Bello University Zaria and Master of Information Management (MIM)- Ahmadu Bello University Zaria, Nigeria