SCHEDULING ALGORITHM FOR FLEXIBLE MANUFACTURING CELLS

BY

Ms. NEHA K DESAI

A Thesis

Submitted to the Faculty of Graduate Studies

in Partial Fulfillment of the Requirements

for the Degree of

MASTER OF SCIENCE

DEPARTMENT OF MECHANICAL AND INDUSTRIAL ENGG.

UNIVERSITY OF MANITOBA

CANADA

0-612-23274-3

SCHEDULING ALGORITHM FOR FLEXIBLE MANUFACTURING CELLS

BY

NEHA K. DESAI

A Thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

January, 1997

ACKNOWLEDGMENTS

The work reported in this thesis was completed under the guidance of my dissertation supervisor Dr. S. Balakrishnan to whom I would like to express my gratitude. Without his assistance, advice and patience, this work would not have been feasible. Gratitude is extended to Dr. Neil Popplewell for giving me tremendous insights, support and inspiration. I am also indebted to Dr. Divakar Rajamani for his unconditional guidance and assistance. I want to take this opportunity to thank my colleagues Mike Thomlinson and Roman Kurjewicz for their time and help in coding the software. I would also like to thank the authors of all the references presented in this work.

I must thank all my friends who have relentlessly supported and encouraged me during the course of my studies. My parents and my sisters deserve a special thanks although words can never express my gratitude for their silent support and encouragement.

ABSTRACT

The problem of scheduling the work in a Flexible Manufacturing cell is considered with the criterion of minimizing the idle time of the robot (i.e. the material handling server). This situation arises when the robot services several machines whose processing times are short. Static and Dynamic scheduling is presented for a work cell containing two lines having two machines in each line and no intermediate buffers for work-in-progress. The possible sequence of robot tasks and their cycle times are formulated. Then the development of a heuristic dynamic scheduler for a work cell is discussed. The heuristic algorithm is coded by using the C programming language. Four heuristic scheduling strategies are considered but only the one giving the minimum robot idle time is used. This user friendly software can be used for any cell having a finite number of n machines and n processors (assemblers) that do not have in-process buffers. It can be used even when a machine breaks down as long as the condition of each component of the cell is known at that instant. The overall strategy is evaluated by interfacing the PC-based software, and a robot controller in order to schedule a product mix in the Computer Integrated Manufacturing cell, located at the Automation Laboratory, University of Manitoba. The computations are discussed and summarized.

TABLE OF CONTENTS

	Page No.
ACKNOWLEDGMENTS	ii
ABSTRACT	iii
LIST OF FIGURES	vi
CHAPTER 1. INTRODUCTION	1
CHAPTER 2. LITERATURE REVIEW	4
CHAPTER 3. THE ROBOTIC CELL	12
3.1 Problem Background	12
3.2 Calculation of Cycle Time	14
3.2.1 Cycles with all machines empty at the start of a cycle	16
3.2.2 Cycles with all machines loaded at the start of a cycle	17
3.2.3 Cycles with some machines loaded at the start of a cycle	18
3.3 Computational Experience	19
3.4 Discussion	20
3.5 Summary	21
CHAPTER 4 THE ON-LINE SCHEDULING ALGORITHM	23
4.1 Problem Background	23
4.2 Methodology	24
4.2.1 Rules for the flow of parts	24
4.3 Computational Experience	29
4.4 Discussion	29
4.5 Summary	30

CHAPTER 5. CONCLUSIONS 32

5.1 Contributions 32

5.2 Scope for future work 34

REFERENCES 35

APPENDIX 1 : Cycles 39

APPENDIX 2 : Computation Tables for static scheduling 61

APPENDIX 3 : Computation Tables for dynamic scheduling 97

APPENDIX 4 : On Line Simulation Software Flowcharts 106

LIST OF FIGURES

		Page No.
Figure 3.1	Static Scheduling of Robotic Cell having two machines in two lines	13
Figure 4.1	On-line sequencing controller for dynamic scheduling	28

CHAPTER 1. INTRODUCTION

Recent innovations in flexible manufacturing systems (FMS) have the aim of realizing fully automated manufacturing. In order to efficiently use these automated facilities, however, many planning problems have to be solved. Among them is the scheduling of jobs, which determines when and on what machine the jobs are to be processed and how they are to be transported in the system. Industrial robots play an important role in advanced manufacturing systems. A major application of such industrial robots is the loading, unloading and transportation of jobs in the production system. The robot can be programmed to perform a sequence of mechanical tasks and it can continually repeat that task until reprogrammed to perform another sequence.

Hundreds of robots and millions of dollars worth of computer controlled equipment are inefficient if they are underutilized or spend their time working on the wrong part because of poor planning or scheduling. The aim of scheduling is to optimize some "cost" associated with the manufacturing process. For example, we might wish to maximize throughput (the number of parts produced in a given time), minimize the makespan (the time between the first job of a given production order entering the system and the last job of that order leaving the system), minimize the work in process (number of unfinished jobs in the system), or minimize the average interval (the average interval of a job, also called the sojourn time, is the time that the job spends in the system, from entering the first workstation to leaving the last one), or some combination of these objectives. (Kochhar & Morris, 1990).

The handling of material is expensive in any manufacturing system, and it represents a significant portion of the cost of doing business (Kamoun *et al.* 1995). Depending on the type of manufacturing facility, estimates ranging from 10% to 80% of the total cost have been attributed to material handling (Tompkins and White, 1984). In order to achieve greater flexibility, the setup times of machine are reduced until they are close to zero. As setup times become negligible, the material handling time and its cost become a bottleneck and efficient material handling becomes crucial. Thus, ideas such as "point of use storage" have been adopted to reduce the number of material moves. In "point of use storage", parts are moved directly from machine to machine instead of returning to a storage area between operations. Not only is the number of moves reduced, but so is the in-process inventory (Askin and Standridge, 1993). This is the motivation to consider a cell with no in-process storage buffers and to study efficient ways to organize material handling within it. Robotic cells with no storage buffers between machines are used widely in practice (Asfahl, 1985; Miller and Walker, 1990).

A problem of robotic cell scheduling arising from an automated manufacturing system is considered in this research. The cell consists of two lines and each line has two machines. The robotic cell, which is used to produce a set of parts of the same or different types, is a flow line manufacturing system. Each part has to be processed on machine M_i and then on processor P_i, $i = 1,2...n$, where n is the number of lines of machines stationed in two stages. Jobs are transported by a robot between either an input/output station and a machine or between the machines. There are no storage buffers so that any part in the cell is always either on one of the machines or it is being handled by the robot. Neither the machines nor the robot can be in possession of more than one part at any instant. The robot is not allowed to exchange a job to be transferred to a machine for a job awaiting transportation from that machine because no machine has a buffer storage for work-in-progress (WIP). In other words, a machine cannot release a job even if the job has been completed already unless a robot is available to take the part to the next stage of operation. A robot, in such cells, performs repeated sequences of pickup, move, load, unload and drop operations. Consequently, the performance of the cell will depend on the sequence of the robot's activities.

There have been several studies on the scheduling of robotic cells. The next chapter is devoted to the review of this literature. Chapter 3 is a study of a 2 machines, 2 lines cell with one robot and outlines some of the cycles possible for static scheduling, in which the sequence of robot tasks is predetermined and cannot be changed, and the computational results for a range of data sets are discussed and analyzed. The rules that evolve, which can be used to implement an on-line scheduling algorithm, are also discussed. In Chapter 4, an on-line scheduling algorithm for the same cell and the heuristics on which it is based are explained. The set up, interface and the software on which the simulation can be run are also included. Finally, the results are discussed and summarized in Chapter 5. The scope for future work is also suggested.

A FMS scheduling problem is considered to be a detailed minute-by-minute scheduling of machines, materials handling system, and other support equipment. Given the actual shop conditions and a set of parts with known processing requirements, it is concerned with accomplishing the following tasks :

- schedule actual job release times,
- sequence the jobs and determine the start and completion times of each operation for a wide variety of resources, and
- monitor the execution of the schedule and provide effective contingency handling.

Although scheduling refers to the time-phased allocation of a system's resources, such as machines, tools, materials handling system, etc., it is applied most often to the scheduling of jobs on the machines. However, for a dynamic and highly integrated system such as a FMS, the real time scheduling of the materials handling system and consideration of the limited input/output buffer capacities are also equally important.

The most common approach to a scheduling problem is to look at the operational shop floor and make use of one or more of the multitude of highly dynamic considerations to guide the assignment of jobs to resources. These considerations may be the result of high level strategic decisions relating to the production, inventory management or the response to market demand. Examples include scheduling objectives and the production workload. Other considerations may be generated within the shop floor as the production is in progress. These include resource-based constraints such as the workload distribution among resources and the dynamic status of the work in progress.

With a wide variety of product designs and highly volatile customer demand for better designed products, scheduling methods must be both predictive as well as reactive to dynamic production demands. Sim *et al.* (1994) explore the use of neural networks to learn and store the relative factors that influence the various considerations for dynamic job shop scheduling. Scheduling rules for manufacturing systems have been reviewed briefly by Montazeri and Van Wassenhove (1990). They also analyzed the performance of several dispatching rules by using a modular simulator to mimic the operation of a real-life FMS.

King *et al.* (1993) developed a branch and bound approach which is coupled with quick, effective bounds to optimize the movements of a robot that serves the material handling requirements within a manufacturing cell. They addressed a specific scenario encountered in the development of a furniture manufacturing cell. Their cell model contained two processing machines, one material handling robot, and an input and an output queue for the cell. Each machine has an input queue of its own and the machines are loaded automatically on a first come, first served basis. Queues were assumed to have infinite capacities. The entire system was formulated as a mixed integer Linear Programming (LP) problem. The algorithm developed determined the sequence of jobs in the input queue and the sequence of robot moves to minimize the make span of the job set.

Kise *et al.* (1991) considered flowshop scheduling problems related to automated manufacturing systems in which n jobs are processed sequentially on two machines, M_a and M_b. The job is transported between an input/output station and a machine or between two machines by a single automated guided vehicle (AGV) or a fixed robot with a swiveling arm. This servicing is crucial because no machine has a buffer storage for work-in-process. Hence, a machine cannot release a finished job until the empty AGV becomes available at that machine. Moreover, the AGV cannot transfer an unfinished job to a machine until that machine is empty. They formulated the dynamics of the system and gave an $O(n^3)$ time algorithm based on the well known Gilmore-Gomory (1964) algorithm for finding an optimal sequence that minimizes the maximum completion time (*i.e.* the makespan) of n jobs. They also showed the solution for a case in which the input and output stations are located separately on both sides of a pair of machines and an AGV moves linearly between them. The solution was applied to a small scale manufacturing cell having simple material handling devices.

Yih, Liang and Moskowitz (1993) proposed a hybrid method that combines human intelligence, an optimization technique (the semi-Markov Decision model) and an artificial neural network (ANN) to solve real-time scheduling problems for maximizing the throughput of "good" parts in the system. Their proposed method has three phases: data collection, optimization and generalization. The test bed was a robot scheduling problem in a circuit board production line where one overhead robot is used to transport jobs through a line of five sequential chemical process tanks with no in-process storage buffers. Because chemical processes are involved in this production system, any mistiming or misplacement will result in a defective job. Semi-Markov decision models were used to optimize the throughput based on training cases collected from the simulation. The ANN was then applied to construct a scheduling model that covers the entire state space for real time scheduling.

Yih and Thesen (1991) also presented a class of real-time scheduling problems that can be formulated as semi-Markov decision problems. They presented a non-intrusive "knowledge acquisition" method which identifies the states and transition probabilities that an expert would use to solve these problems. This information was used in the semi-Markov optimization problem. A circuit board production line was employed to demonstrate the feasibility of this model, the objective of which is to develop a sequence of moves that maximizes throughput. They considered a production process that requires a sequential process through two different workstations and an infinitely fast, fork lift truck to move parts between stations. There is no buffer space between two workstations. Parts are always available for loading at the first station and it is always possible to unload parts from the second station.

Gupta and Tunc (1991) developed approximate algorithms to find the minimum makespan in a two-stage, hybrid flowshop in which the second stage consists of multiple identical machines. This paper considers n-jobs to be processed in M stages, with only one machine at stage 1 and m identical machines at stage 2. In view of the NP-completeness of the problem ($1<m<n$), two polynomially bounded, heuristic algorithms are proposed to find an acceptable (*i.e.* optimal or approximately optimal) solution to the problem of minimizing the time in which all jobs complete their processing through both stages. An improved branch and bound algorithm is also described in which the heuristic algorithms are augmented with an existing branch and bound algorithm. The effectiveness of the algorithms in finding the minimum makespan schedules is evaluated empirically and found to increase with more jobs.

Sawik (1995) proposed a heuristic algorithm for scheduling a flexible flowline having no intermediate buffers. The algorithm is a part-by-part heuristic in which a complete processing schedule is determined during every iteration for one part type selected for loading into the line. The selection of the part type and its complete schedule is based on the cumulative partial schedule obtained for all parts selected previously. The decisions in every iteration are made by using a local optimization procedure aimed at minimizing the total blocking and waiting time of the machines along the route of the selected part type. The algorithm, called RITM_NS (Route Idle Time Minimization-No Store) is a special variant, RITM heuristic designed for scheduling flexible flow lines having a limited number of intermediate buffers (Sawik, 1993). The flexible flow line studied had more than two processing stages in series, where each series had more than one identical parallel machine with no intermediate buffers. The system produced N different part types. The single pass, RITM_NS heuristic for scheduling the flexible flow line with no in-process buffer achieved good solutions in a very short CPU run time. An IBM PC/AT was used and the computation time was not greater than one second for the medium sized problems that can be encountered in an industrial practice.

According to Sethi *et al.* (1992) only a few studies have been reported on the scheduling of parts and robot moves in a robotic cell. Baumann *et al.* (1981) derived models to determine robot and machine utilizations for an application in which the machines were serviced by a robot. Bedini *et al.* (1979) considered an industrial robot equipped with two independent arms and developed heuristic procedures for optimizing the work cycle. Kondoleon (1979) analyzed the effects of various robot assembly and system configurations on the cycle time. He used computer modeling to simulate robot motions involving different cycles and the times they take. Maimon and Nof (1985) dealt with control problems in an assembly application in robotic cells having multiple robots. Nof and Hanna (1989) studied the problem of cooperation among robots in a multi-robot system and developed measures of cooperation levels. Drezner and Nof (1985) formulated and developed several approaches for sequencing bin picking and insertion operations in an assembly cell. Seidmann *et al.* (1985) presented a predictive model to describe the production capacity of multi-product robotic cells with stochastic activity times and random feedback flows. Seidmann and Nof (1989) presented operational analysis models of robotic assembly cells in which assembled items may have to be reworked one or more times in the cell. Devedzic (1990) proposed a knowledge-based approach for the strategic control of robots in flexible manufacturing cells. Wilheim and Sarin (1985) considered problems of scheduling parts in a robotic cell for the following machine configurations: parallel identical machines, parallel non-identical machines, and flow line manufacturing. They provided a mathematical programming formulation for the flow line case. However, their studies did not develop any scheduling policy. Sarin (1987) studied the scheduling problems in a robotic cell for a particular application. Rajendran (1994) developed a heuristic algorithm for scheduling in a flowshop and a flow line-based, manufacturing cell with the two criteria of minimizing the makespan and total flow time.

The approach most similar to the work presented in this thesis is that of Sethi *et al.* (1992) and Hall *et al.* (1995). Sethi *et al.* (1992) employed a state space approach to address the problem of sequencing parts and robot moves in a robotic cell where the robot is used to feed machines in the cell. The robotic cell is a flow-line manufacturing system which produced a set of parts that may be either identical or different. The objective was to maximize the long-run, average throughput of the system subject to the constraint that the parts to be produced are in proportion to their demand. Cycle time formulae were developed and analyzed for cells with two and three machines producing a single part type. Both necessary and sufficient conditions were obtained for various cycles to be optimal. They also considered the case of many part types, and formulated the problem of scheduling these parts for a specific sequence of robot moves in a two machine cell as a solvable case of the traveling salesman problem. Hall *et al.* (1995) considered the scheduling of operations in a manufacturing cell that repetitively produce a family of similar parts on two or three machines served by a robot. They provided a classification scheme for scheduling problems in robotic cells. They considered the robot move cycle and the part sequence that jointly minimize the production cycle time or, equivalently, maximize the throughput rate. They provided an efficient algorithm for a multiple part type problem in a two machine cell. This algorithm simultaneously optimizes the robot move and part sequencing. It was tested computationally. For a three machine cell with general data and identical parts, they addressed an important conjecture about the optimality of repeating one unit cycles. They showed that such a procedure dominates more complicated cycles producing two units. For a three machine cell producing multiple part-types, they proved that four out of the six potentially optimal robot move cycles for producing one unit allowed efficient identification of the optimal part sequence. Several efficiently solvable and practical cases were identified, because the general problem of minimizing the cycle time is intractable. Finally they discussed the ways in which a robotic cell converges to a steady state.

Hall *et al.* (1995) did not consider the case in which there could be more than one parallel line of machines. In this study we consider a cell structure consisting of 2 lines of machines with two machines each, arranged in a flow line. The parts produced in such a flow line could be different or the same for each line. Moreover, the processing times on each machine could be different because the machines could be of a different make. There are input and output buffers, each buffer having an infinite capacity, but there are no in-process buffers. The objective is to study a cell consisting of two lines having two machines each. In any cycle for such a cell, at least six tasks have to be undertaken which would result in 120 or, in the general case of n tasks, (n-1)! combinations. Each combination gives a unique cycle. Another objective of this work is to determine the sequence of robot tasks that outperforms the other sequences in terms of cycle time as well as to develop an intuitive idea to see if there are any specific patterns exhibited by the cycles which can aid us in selecting the heuristic scheduling strategy for on-line scheduling involving short machining and robot travel times. A more general case is considered, in the on-line simulation algorithm, in which the number of lines can be greater than two. However, if this number is very large, the robot's travel time does not remain the most important parameter in the calculation of the cycle time. The objective here is to set rules and develop a simulation code for the selected scheduling strategies which are used on-line with the cell located in the Computer Integrated Manufacturing and Automation Laboratory at the University of Manitoba. Another purpose is to ensure that the on-line algorithm works in real-time and that the conditions of each machine are monitored. The overall goal is to identify a few robust cycles which can be used most of the time for the data selected in this research.

The analysis of robot loading becomes complicated when a single robot has the task of servicing several machines in an organized sequence. The level of complexity is shown in the next section. If the automation engineer has timed and planned the operation carefully, the robot can be programmed to anticipate the cycle completions at an appropriate station and move to this station in advance in order to reduce the idle time. In this chapter we study a cell consisting of 2 lines, each having 2 machines, and one robot for material handling. Cycle time formulae are developed and analyzed for producing one part of each type. The cycle times for 50 random data sets are computed and, from the cycle that gives the lowest cycle time, rules are evolved on which an on-line algorithm can be based.

3.1 PROBLEM BACKGROUND

The manufacturing work cell has two lines, each having one machine and one processor (labeled M1, M2 at stage one and P1, P2 at stage two). There is one central material handling robot (labeled R), as shown in Figure 3.1. The workstations at stage 1 are called, for convenience, machines and those at stage 2 are called processors. One input buffer (I) supplies the raw material (or in-process workpieces) to both lines and one output buffer (O) receives the processed workpieces from the lines. Both these buffers have an infinite capacity.

The system can be described as follows.

There is one machine, Mi, and one processor, Pi, in each line i, and each machine and processor has

- no in-process storage buffer,
- the ability to handle one job, and

- operates independently.

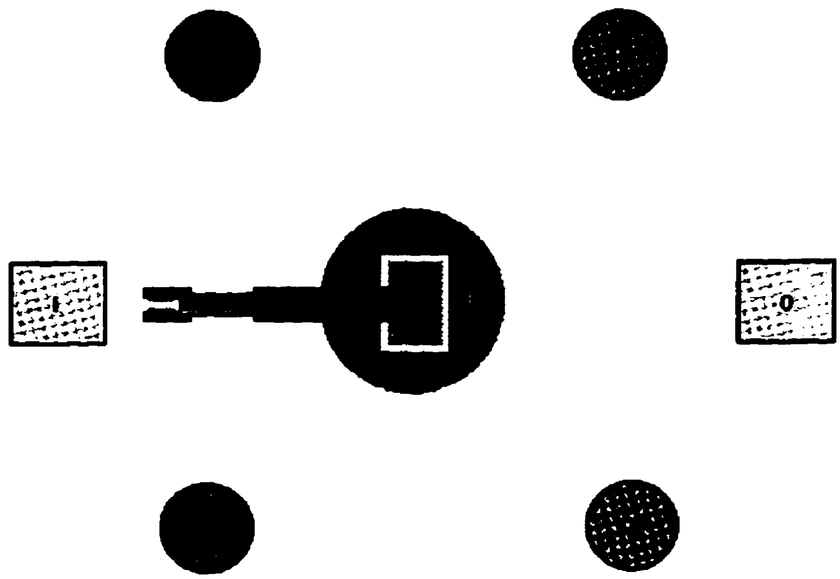


Figure 3.1. Robotic cell with two machines

There is a central material handling robot that can handle, at any instant, only one part. The travel time between the input buffer, I, and any machine, Mi; between machines and processors, and between any processor Pi and the output buffer O is a constant δ. The travel time between any machine or processor and the robot's home position as well as between different points within the system is assumed to be the same and fixed throughout the schedule, for simplicity. However, it can be changed straightforwardly in the algorithm, if necessary.

The robot has a total of three tasks on each line, i. They are, to :

- load a part on machine Mi from the input buffer,
- unload a part from machine Mi and load it on processor Pi, and
- unload a part from processor Pi to the output buffer.

Each line produces one part type in each cycle. The flow of a part is sequential, that is, parts from Mi can go only to Pi. So the path of the part in line *i* is I - Mi - Pi - O, where i = 1, 2, 3....n, and n is the number of lines..

The objective is to minimize the cycle time.

A summary of the assumptions is given below.

1. There are no buffers available for the work-in progress.
2. The machines and processors can each process only one job at a time.
3. No more than two parts of the same type can be in the system at a given instant.
4. The robot cannot simultaneously serve two workstations.
5. A part can go from a machine solely to the corresponding processor, *i.e.* a part from machine M1 can only go to processor P1, a part from machine M2 can only go to processor P2, etc.
6. The operation time at each machine and processor is deterministic and fixed.
7. There cannot be more than 2n parts in the system at any instant as there are no buffers and there are two machines in each line having a capacity of one part each.

3.2 CALCULATION OF CYCLE TIME

The objective is to determine the sequence of robot tasks which gives the minimum cycle time and to determine if there is a sequence better than other sequences in terms of cycle time. Hence, the cycle times corresponding to each sequence have to be formulated and determined.

There are six robot tasks that have to be sequenced in one cycle. They are:

(i) load a part on machine M1 from the input buffer,

(ii) load a part on machine M2 from the input buffer,

(iii) unload a part from machine M1 and load it on processor P1,

(iv) unload a part from machine M2 and load it on processor P2,

(v) unload a part from processor P1 to the output buffer, and

(vi) unload a part from processor P2 to the output buffer.

These tasks are the basic activities of the robot. Note that in a one part cycle, every basic activity must be carried out exactly once.

If t is the number of tasks to be scheduled in a cycle, the number of cycles possible for different combinations of task sequences is (t-1)!, which is 120 when $t = 6$. Sequences are classified according to the condition of the machines at the start of the cycle. The possible conditions are:

1. - all the machines are empty,
2. - any one machine is loaded while the remaining three machines are empty,
3. - any two machines are loaded while the other two are empty,
4. - any three machines are loaded and one machine is empty, and
5. - all four machines are loaded and waiting to be serviced.

In the 2nd, 3rd and 4th condition, there are different conditions again depending on which machine(s) are loaded at the start of the cycle.

The cycle time is the duration taken by the robot to load and unload the machines and the time it might have to wait at any machine while that machine is busy. The initial cycles at the start of the schedule may not have the same sequence of robot tasks. The cycle time is calculated only after the cycle has stabilized and reached a state when the sequence of tasks in any cycle is constant. The following parameters influence the cycle time (CT):

δ = robot's travel time (between the input buffer and machines, between machines and processors, between processors and the output buffer). This time also includes the gripper time when the part is picked up or released at various stations.

m1 = machining time at Machine M1 at stage 1 and w1 = wait time at M1,

m2 = machining time at Machine M2 at stage 1 and w2 = wait time at M2,

m3 = machining time at Processor P1 at stage 2 and w3 = wait time at P1,

m4 = machining time at Processor P2 at stage 2 and w4 = wait time at P2, and

m = iteration number when the cycle has stabilized into a constant, regular pattern. At this point, the wait time at any machine or processor is equal to the corresponding wait time at the same machine or processor in the previous cycle and in the subsequent cycle.

The total processing time for a part produced in line 1 is m1 + m3 and the total processing time for a part produced in line 2 is m2 + m4. Moreover the cycle time can be expressed as:

$$CT = R\,\delta + w1 + w2 + w3 + w4, \tag{3.1}$$

where R (equals 12, when n = 2) is the number of robot moves.

The minimum possible cycle time is R δ which occurs if the robot does not need to wait at any machine or processor. This is likely to happen when the travel time of the robot is significantly higher than the machining times at the different workstations. The waiting time at any machine or processor may depend on the waiting time at other machines or processors that the robot visits prior to the wait. Hence, in some cycles, the waiting time calculation is iterative and, when the cycle stabilizes, the wait times corresponding to the same machines in two consecutive cycles are equal.

In the following sections, cycles that the robot tasks can be sequenced in, starting with all machines unloaded or empty, are considered. Several cycles for the condition that a few machines are loaded are also considered. The cycle time formulae are formulated and the cycle time for random data sets are computed to determine which cycle would give the lowest cycle time. The objective is to determine if a few cycles generally tend to outperform for the selected data set.

3.2.1 Cycles with all machines empty at the start of a cycle

Cycles with all machines initially empty is the simplest condition to start a cycle. The waiting time at any machine is independent of the waiting time of any machine in the previous cycle. Thus, the computation of cycles times becomes easy. Also, in the case of a breakdown, all the parts remaining on the machines can be completed on some other machine or they can be scrapped, if the scrap value is low, so that the cell can be brought back to the condition of all machines being empty and ready to be loaded.

When all the machines and processors are empty, the cycle can start with the loading of either machine M1 or machine M2. Regardless, the cycle has to start at the input buffer, I. Considered here are the 10 cycles starting with loading machine M1. The other 10 cycles starting with loading machine M2 are mirror images of these 10 cycles.

The different sequences and corresponding cycle time formulations are given more conveniently in Appendix 1 A.

3.2.2 Cycles with all machines loaded at the start of a cycle

In this section we discuss the possible sequences if all the machines and processors are already loaded at the start of a cycle. The initial cycle can be manipulated to reach this stage. For this condition, machine M1 and machine M2 (after loading these machines) are the only two nodes where the cycle can start because, in order to reach the condition of all machines having parts at the start of a cycle, loading of these machines would be the last task in any cycle. The cycle cannot start at any of the processors at the second stage because, when the robot is at the second stage, it has just completed the task of unloading one of the machines at the first stage. Therefore, the cycle must start at either of the machines at stage one. The next task in the sequence would be unloading processors at stage 2 because, unless the processors are unloaded, the parts on the machines at stage 1 cannot be unloaded. We consider that the last task in the cycle is either the loading of machine M1 or the loading of machine M2.

The different possible sequences and their cycle time formulations are presented in Appendix 1 B.

3.2.3 Cycles with some machines loaded at the start of a cycle

In this section we discuss the possible sequences if some of the machines and processors are already loaded at the start of a cycle. The initial cycle can be manipulated to reach this stage. There can exist three conditions for this situation. They correspond, when n=2, to:

- any one machine/processor is loaded and the other machines/processors are free,
- any two machines/processors are loaded and the other machines/processors are free, and
- any three machines/processor are loaded and the remaining machines/processors are free.

The number of cycles possible for each of the above conditions is large because there are different sub-conditions for each condition.

For the condition that any one machine is loaded, we can have conditions such as:

a) - machine 1 (M1) is loaded and the other machine and processors are free,

b) - machine 2 (M2) is loaded and the other machine and processors are free,

c) - processor 1 (P1) is loaded and the other processor and machines are free, and

d) - processor 2 (P2) is loaded and the other processor and machines are free.

For the condition that any two machines are loaded, we can have

e) - machine 1 (M1) and machine 2 (M2) are loaded,

f) - machine 1 (M1) and processor 1 (P1) are loaded,

g) - machine 1 (M1) and processor 2 (P2) are loaded,

h) - machine 2 (M2) and processor 1 (P1) are loaded,

i) - machine 2 (M2) and processor 2 (P2) are loaded, and

j) - processor 1 (P1) and processor 2 (P2) are loaded.

For the condition that any three machines are loaded, we can have

k) - machine 1 (M1), machine 2 (M2) and processor 1 (P1) are loaded,

l) - machine 1 (M1), machine 2 (M2) and processor 2 (P2) are loaded,

m) - machine 1 (M1), processor 1 (PI) and processor 2 (P2) are loaded, and

n) - machine 2 (M2), processor 1 (PI) and processor 2 (P2) are loaded.

We consider sub-conditions b, c, d, f, g, j, l, and n from the above. Not all cycles are considered here because the objective is to study a selected few for each condition and see if certain cycles tend to outperform the others. The cycles were randomly picked. Hence, each of the main conditions (one, two or three loaded machines) is considered. The different sequences and their cycle time formulations are given in Appendix 1 C.

3.3. COMPUTATIONAL EXPERIENCE

To determine which case provides the minimum cycle time, the cycle time was computed for fifty randomly generated data sets for each cycle. The machining times for the machines were generated from a Scientific Calculator by using the "random number generator". The values considered were between 10 and 100 s to ensure an assembly line in which the operation time is low. These values are representative of typical processing times in packaging, machine tending, assembly and similar manufacturing tasks. The travel time, δ, between stations was kept constant throughout the cycle. However, the analysis was done for three values of δ, namely 5 s, 12 s, and 20 s to study the effect of the robot's travel time and its influence on the cycle time. The values chosen are representative of the actual time the ASEA robot takes to travel between machines, as determined from the existing machine cell in the Computer Integrated Manufacturing and Automation (CIMA) Laboratory, University of Manitoba. The cycle times for each data set were computed for the chosen travel time (δ). The resulting cycle times for these data sets are presented in Appendix 2.

3.4 DISCUSSION

When the robot's travel time is 5 s, 12 s and 20 s, it can be seen from the computed results shown in Appendix 2 that, for the condition of section 3.2.1 (*i.e.* all machines empty at the start of a cycle), sequence 1.A.1 gives the lowest cycle time in 63% of the cases. For the condition of section 3.2.2 (*i.e.* all machines loaded at the start of a cycle), sequence 1.B.4 (and 1.B.10 because both have the same sequence of robot tasks) gives the lowest cycle time for all the data sets considered. The same sequence also gives the lowest cycle time compared to those starting with either all machines empty or a few machines loaded at the start of the cycle. This cycle starts with all the machines loaded. In this particular sequence, the robot first serves one line and then moves to serve the other line. Also, once a machine is unloaded, it is loaded immediately with the next part before the robot moves to the next line. Cells having more than 2 lines were not investigated in this study.

It can be seen from the computed cycle times that, when the robot's travel time is considerably lower than the machining times, the condition that all machines start in a loaded state at the start of a cycle invariably gives the lowest cycle times. Then the robot services another machine and does not wait at a particular machine to unload it, after loading it. **As the robot's travel time increases, the cycles involving fewer robot moves and waiting at particular machines (for the entire time that the machine is operating) give a lower cycle time.** To demonstrate this assertion, high travel times of 30 s, 40 s, and 50 s were used to compute the cycle time. The following tabulated data are examples of the same. The values shown in "**bold**" represent cycle times that are lower than the minimum cycle time obtained by using cycles starting with few or all machines loaded at the start of the cycle. (See sections 3.2.2 and 3.2.3.)

For $\delta = 30$ s, m1 = 29 s, m2 = 24 s, m3= 30 s, m4 = 67 s, the cycle time given by cycles starting with a few or all machines loaded (sections 3.2.2 and 3.2.3) is 360 s, whereas that given by the cycles starting with all machines empty (section 3.2.1) is:

cycle number	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8	1C.9	1C.10
cycle time	360	397	397	**354**	**354**	391	396	**353**	390	390

For $\delta = 40$ s, m1 = 27 s, m2 = 45 s, m3= 77 s, m4 = 55 s, the cycle time given by cycles starting with a few or all machines loaded (sections 3.2.2 and 3.2.3) is 480 s, whereas that given by the cycles starting with all machines empty (section 3.2.1) is:

cycle number	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8	1C.9	1C.10
cycle time	480	495	532	485	522	537	482	**472**	487	524

For $\delta = 50$ s, m1 = 79 s, m2 = 20 s, m3 = 19 s, m4 = 20 s, the cycle time given by cycles in condition 3.2.2 and 3.2.3 is 600 s, whereas that given by the cycles in condition 3.2.1 is:

cycle number	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8	1C.9	1C.10
cycle time	600	**570**	**599**	**570**	**599**	**569**	**599**	**599**	**569**	**598**

3.5 SUMMARY

From the computations, when the robot's travel time is low, say 5 s, the condition that all machines are loaded at the start of a cycle results in a minimum cycle time (makespan). As the robot's travel time increases, say to 12 s, all the cycles starting with a few machines loaded (see section 3.2.3) give an equally low cycle time of 144 s. When the robot travel time is greater than 20 s, the cycles starting with all machines free give the lowest cycle time because, at that point, machining times are significantly lower than the robot's travel time and the cycle times are strongly influenced by the robot travel time.

From the sequence of robot tasks (cycle 1B.4, all machines loaded at the start of the cycle), which invariably gives the lowest cycle times for all the data sets considered (when robot's travel time is less than 20 s), we also get an idea of which rules would apply to cells having more lines. The following two rules would seem reasonable.

- Once a machine is unloaded, it is loaded immediately with the next job before the robot moves to the next line
- The robot first completely serves one line and then serves the other line.

These rules can be used for an on-line simulation of product flow through a cell. However, the scheduling analysis presented in this chapter will not be suitable for situations where machines are expected to breakdown and the schedule must be continuously updated in a dynamic environment (in which the state of machines and the parts change). An on-line scheduling approach is proposed in the next chapter.

In an on-line scheduling approach, the scheduling decision is made when the state of the system changes, such as a job completion, arrival of parts, etc. On-line scheduling is a short-term, decision making process which generates and updates a schedule based on the real-time conditions. This can be referred to as dynamic scheduling because it emphasizes the dynamic nature of the real-time scheduling problem. In this chapter, a dynamic scheduling method is presented. Essentially a knowledge based approach for cell level scheduling, the method is adaptive to changes and can take into account such information as unexpected breakdowns. Initially, however, the decision making for choosing the strategy to be used is static. The cell under consideration can have more than 2 lines of machines at two stages. The workpiece flow and relevant data are identical to those used in Chapter 3. The cycles considered in Chapter 3 produce one part of each type, while in the on-line scheduling approach more than 2 parts per cycle may be produced. In this chapter we develop a simulation code to consider four scheduling strategies and select the best for implementation in the cell for a given data set.

4.1 PROBLEM BACKGROUND

The manufacturing cell is similar to the cell shown in Figure 3.1. The cell consists of :

- two machines M1 and M2 at stage 1,
- two processors P1 and P2 at stage 2,
- and one robot R that services the machines and the processors, as well as the
- input and output buffers.

The objective is to minimize the robot's idle time so as to minimize the cycle time and implement the scheduling algorithm on a model cell.

4.2 METHODOLOGY

In the literature pertaining to sequencing/scheduling, terms such as scheduling rule, dispatching rule, priority rule or heuristic are often used synonymously. Gere (1966), however, attempted to distinguish between priority rules, heuristics, and scheduling rules. He considers priority rules as simply a technique by which a number (or value) is assigned to each waiting job according to some method and the job with the minimum "value" is selected. He considers priority rules as simply a technique by which a number (or value) is assigned to each waiting job according to some method and the job having the minimum "value" is selected. Gere defines a heuristic to be simply a "rule-of-thumb", whereas a scheduling rule can consist of a combination of one or more priority rules and heuristics. Panwalkar *et al.* (1977) present a summary of over 100 scheduling rules. Given only the machining times and the robot's travel time, only the Shortest Processing Time and Largest Processing Time rules are used here for decision-making.

Active schedule generation in a dynamic job shop system involves a quick solution of not only sequencing but also the decision of routing at any particular time. These decisions are based on rules and priorities. The on-line simulation algorithm presented in this thesis is also based on a set of rules that are explained next.

4.2.1 RULES FOR THE FLOW OF PARTS

When the robot completes a task, it has to make a decision as to which job it should perform next. This decision making is based on some rules and priorities. The rules on which the decision-making was based for this research are explained in this section.

There are three different robot tasks that have to be performed on one line of machines.

They are

- loading a part on a machine,

-unloading the machine and loading the part on the corresponding processor, and
-unloading the part from the processor.

In the cell considered, there are two machines. The robot has to decide on which it should load first. Rule 1 will decide which machine should be loaded first.

RULE 1

Initially all machines are loaded. The loading of machines is undertaken according to the priority assigned. Because the only information available is the machining times and the robot's travel time, priorities are assigned according to the machining times. Similarly, the loading of processors is prioritized according to their processing times.

For example, higher priority can be given to:

in machines :

- lower machining time or
- higher machining time

and in processors :

- lower processing time or
- higher processing time.

or any combination of the above is possible.

RULE 2

Unloading a processor is given the lowest priority among the robot jobs. The highest priority is given to unloading a part from a machine and loading it on a processor. Parts from machines cannot be unloaded and reloaded on processors unless the corresponding processors have been unloaded and are free because the system does not have intermediate buffers for work in progress.

For example, a part from M1 cannot be unloaded unless P1 has been unloaded and is free. Unloading a machine and loading the part on a processor are sequential processes, *i.e.* once the robot unloads a machine it can service another machine only after the part is loaded on the corresponding processor. For example, at a certain time the robot unloads a part from M1 but machine M2 also needs unloading, then the robot will first load the part from M1 to processor P1 before servicing M2.

Let us consider a case where the robot has unloaded a processor and the corresponding machine has finished its job. In the meantime, another machine also finishes its job and its corresponding processor has finished a job but is not unloaded. As loading the processors is given higher priority, the robot will first load the processor that has the higher priority. As an example, let us assume that M1 has finished machining. The robot unloads P1 so that the part from M1 can be loaded on P1. While the robot is unloading P1 suppose machine M2 finishes its job. P2 has also finished its job but it has not been unloaded yet. In this case, loading P2 has a higher priority over loading P1. Hence, the robot after unloading P1 will unload P2, unload M2 and reload the part on P2 and then proceed to reload the part from M1 to P1.

RULE 3

The reloading of machines can be done in the following ways:

a) **Reloading a machine as soon as a part from that machine has been loaded on a processor is given highest priority in sequencing the robot jobs.**

   For example, after a part from M2 is loaded on P2 the robot immediately loads M2 before proceeding with other jobs.

b) **Unloading a part from a machine and loading it on the corresponding processor is given higher priority over reloading a machine that has just been unloaded.**

   For example, a part from machine M1 is reloaded on processor P1. Machine M1 is now free. In the meantime M2 has finished machining a part and P2 is free. In this case the robot will first reload the part from M2 to P2 and then load M1 and M2 according to the priority assigned for loading the machines.

RULE 4

There are the following four possibilities for the loading of machines and processors, based on their machining/processing times:

Higher priority can be given to

1. a lower machining time and higher processing time, or
2. a lower machining time and lower processing time, or
3. a higher machining time and lower processing time, or
4. a higher machining time and higher processing time.

Therefore, we can have a combination of strategies for loading machines and processors as well as reloading machines.

i.e., we have eight different strategies: 1(a), 1(b), 2(a), 2(b), 3(a), 3(b), 4(a), 4(b).

It was found through simulation that loading by strategy (a) leaves a machine-processor pair that has the longest processing times unserviced after some time. This happens because the machines that have short processing times get priority each time a decision has to be made and the robot continues to service them. So, while developing the software, only strategies 1b, 2b, 3b and 4b were considered.

4.3 IMPLEMENTATION OF ON-LINE SCHEDULING ALGORITHM :

Software incorporating the four strategies was coded in the C programming language to obtain a schedule of robot tasks given a set of inputs. The real-time scheduling system, shown in Figure 4.1, consists of a controller (computer with scheduling software), input/output interface and the machining cell. The inputs to the software were:

- the number of machine-processor pairs,

- machining and processing times,

- the time from which the schedule must be generated,

- the time at which the schedule generation ends, and

- the robot's travel time.

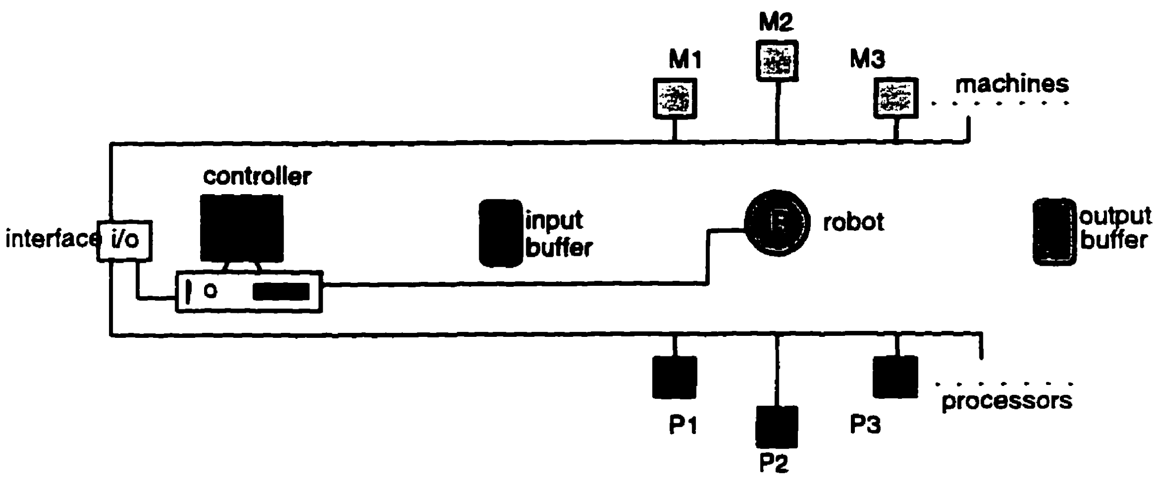


Figure 4.1 The on-line sequencing controller for dynamic scheduling.

Initially, the strategy that gives the minimum cycle time is determined, considering that there will be no machine breakdown. As the cycle time depends upon the robot's travel time and the wait time at the machines (which is the robot's idle time), the software calculates the idle time by using all four strategies and finds the one that gives the minimum idle time of the robot. The time taken by the software to generate the schedule is directly proportional to the total time for which the schedule is generated and the number of lines of machines and processors. For several 3 machine and 3 processor cells, the time taken for generating a schedule for an 8 hour non-stop shift, was found to be 17-20 seconds on a 486 DX PC. It took 21 s for a 5 x 5 machine cell.

Once the best strategy is decided, the real-time scheduling can start based on this strategy and the robot is commanded to do the tasks simultaneously. An experimental system was set up in the Computer Integrated Manufacturing and Automation (CIMA) laboratory at the University of Manitoba. The software was interfaced with the machines and the robot through an input/output interface card. The condition of each machine and the processor of the cell was monitored continuously, by the software using sensors. Scheduling decisions were made based on the conditions of each component of the cell at a given instant and the scheduling set depending upon the strategy used. That is, every time the robot had to make a decision about the task to be done next, it would take the decision considering the feedback sent by each sensor and priorities assigned to the tasks and the machines and processors.

To know whether the machine or processor is ready for service, a corresponding input on the I/O interface card, which monitors the status of that machine, is checked. When there is a part on the machine the corresponding sensor sets the bit to 1, otherwise it is set to 0. For example, when machine M1 is ready to be loaded, the controller checks if the bit corresponding to M1 is set to 1 and then sets the bit on the output card to call the appropriate robot task in order to load machine M1.

The outputs from the software are :

- idle times by using all 4 strategies,
- optimal strategy that gives the minimum idle time,
- a schedule generated by using the optimal strategy and
- the number of parts of each type produced at the end of the schedule.

If a machine breaks down, the machine sends a signal to the microcomputer. The software then eliminates that machine from the cell matrix and reschedules the robot's tasks for the new (n-1)*(n-1) cell according to the optimal strategy for this new cell. The computational time to determine the best strategy for the modified cell will depend on the number of machines in the cell. When the machine re-enters the cell after the problem has been rectified, the software reschedules the tasks for the original n*n matrix with the corresponding priorities. The scheduling strategy at this time might be different because the conditions of the cells have changed from the initial situation. When the tasks are done, the software outputs the robot's idle time and the number of parts of each type produced.

The general structure of each strategy and the software programmed in the C language are given in Appendix 4.

4.4 COMPUTATIONAL EXPERIENCE

To determine which schedule provides the minimum cycle time, the cycle time was computed for fifty data sets that were previously considered for the static scheduling. The travel time, δ, between stations was kept constant throughout the cycle. However, the analysis was done for three values of δ, namely 5 s, 12 s, and 20 s, for the reasons stated in section 3.3. In order to compare the results from the on-line algorithm to those presented in Chapter 3, the same data set was used. The cycle times for each data set were computed for the chosen travel time (δ) and the results are presented in Appendix 3. The cycle times computed in Chapter 3 are for a cycle that produces only two parts (one of each type) in each cycle. For the on-line scheduling implementation, it was observed that, in some cases, the number of parts produced per cycle was more than 2.

4.5 DISCUSSION

When the machining and processing time is low and the robot's travel time is high, we find that the cycle time is low but the idle time of the machines and processors is high. High machining/processing time and low robot travel time results in a higher cycle time and lower machine/processor idle time. Reducing the machining/processing time and robot travel time by a factor of x lowers the robot's idle time by a factor of approximately x.

Other than this observation, no relation could be found between the machining times and the idle time of the robot.

On computing the cycle times for an on line scheduling implementation, it was observed that, in some cases, there are more than one part of each type produced in each cycle (Refer to Appendix 3). This is because the cycle is a combination of two or more of the cycles listed in Chapter 3. The cycles overlap and minimize the time taken per part. The ratio of the parts produced may vary. We see from Table 3.1 that the Shortest Processing Time rule gives a lower average cycle time per part compared to the Largest Processing Time rule. For a larger robot travel time (for example, the 20 s of Table 3.3), we find that all strategies give the same cycle time for the data considered in this thesis.

4.5 SUMMARY

Dynamic scheduling gives a lower cycle time when the ratio of parts produced in a cycle is not critical. This is favorable when the same type of parts is produced in different lines. It could happen when the processing times are different on different machines and the ratio of part types is not important but the total number of parts is. Moreover, the chances that the system goes haywire are reduced because the dynamic conditions of the cell are monitored continuously and the cell is kept operating at high if not optimal efficiency.

5.1 CONTRIBUTIONS

The problem of scheduling a Flexible Manufacturing Cell was considered with the criteria of minimizing the idle time of the robot (material handling server) in order to minimize the cycle time (makespan). This situation arises when the robot attends to a number of machines and processors and the machining and processing times are relatively short. Static scheduling has been developed for a work cell having two machines in two lines and no intermediate buffers for work-in-progress.

A heuristic based, dynamic on-line methodology for a robotic work cell has also been presented. Four scheduling strategies were considered. A knowledge based, scheduling software coded using the C programming language automatically picks the strategy that produces the minimum robot idle time. This user friendly software can be used for any n machine x n processor cell that does not have buffers. It can also be used for generating alternate (efficient) robot sequences when a machine or processor breaks down, provided the status of each component of the cell is known at that instant. The strategy was evaluated by interfacing the PC-based software and the robot controller in order to schedule a product mix in a Computer Integrated Manufacturing cell.

In static or predetermined scheduling, the sequence of robot tasks is decided before the cycle starts and the robot follows that sequence unless the sequence is changed by the operator. If only one part of each type is produced in each cycle, static scheduling seems to give lower or equal cycle times when compared with those from dynamic scheduling. When the demand is unimportant and the number of parts produced per cycle is not restricted to one of each type, dynamic scheduling gives an average production time per part which is either lower or equal to that generated by static scheduling. Table 5.1 shows the minimum cycle times obtained by using dynamic and static scheduling as well as cases where the number of parts produced is more than two.

Table 5.1.

Operating Time				Dynamic Scheduling			Static Scheduling	
M1	**M2**	**P1**	**P2**	**Cycle time**	**Parts per cycle**	**Cycle time (2 parts per cycle)**	**Cycle time**	**Parts per cycle**
50	32	70	58	90	2	90	**85**	2
33	33	16	93	113	3	**76**	108	2
32	95	41	22	122	3	**82**	110	2
27	45	77	55	97	2	97	**92**	2
98	58	48	24	236	5	**95**	113	2

The above table shows that when the number of parts produced per cycle is allowed to be more than two, dynamic scheduling gives a lower cycle time than static scheduling.

The cycle which gives the lowest cycle time for static scheduling has the following conditions at any time. If the robot is working on one line, **the machines on the other line/s are in a loaded condition. The machine that is unloaded is loaded immediately again.** For a lower robot travel time, this cycle gives the lowest of cycle time most of the times.

The same rules apply when the number of lines is greater than two. The way the robot is scheduled to serve the different lines is based on the rule that the loading of a machine follows immediately after the unloading of that machine. Only after both the machines of a line are served, will the robot move to the next line.

Only 2 lines with 2 machines were considered in this research. However, the same rules could be applied to a cell having more than two lines. But, beyond a certain value, which depends on the machining times and robot travel time, the waiting at machines would be significantly reduced because the robot is shuttling between a larger number of machines which would give them sufficient time to work on a part. This will reduce the robot's waiting time in any cycle and also minimize the cycle time.

5.2 SCOPE FOR FUTURE WORK

The static scheduling results for the 2 machines and 2 lines cell should be extended to the general case of a cell having integer m machines and n lines. The present on-line scheduling algorithm is designed with this extension in mind. It works for any finite number of machine-processor pairs. Also, the cut-off point in the robot's travel time when the rules of the previous section no longer give a lower cycle time could be researched. This point might depend on the machining times and their relation to the robot travel time.

A contingency schedule in the case of a breakdown can be considered for static scheduling. It has not been considered in this work. Breakdowns have been considered for dynamic scheduling by removing the line which is not functional. Also, heuristic rules can be developed for a machine returning after repair to be given a higher priority than the other machines. The algorithm can be modified to also consider the due-date of products. This would mean, however, that more user-inputs would be required. At present, the rules are based mainly on the machining times and the robot's travel time.

REFERENCES

Allahverdi. A and Mittenthal. J, 1994, Scheduling on M Parallel Machines Subject to Random Breakdowns to Minimize Expected Mean Flow Time. *Naval Research Logistics,* 41, 677-682.

Arosio. M and Sianesi. A, 1993, A Heuristic Algorithm for Master Production Schedule Generation with Finite Capacity and Sequence Dependent Setups. *International Journal of Production Research,* 31(3), 531-554.

Asfahl. C. R, 1992, *Robots and Manufacturing Automation.* John Wiley & Sons, New York, NY.

Atallah. M. J and Kosaraju. S. R, 1988, Efficient Solutions to Some Transportation Problems with Application to Minimizing Robot Arm Travel. *SIAM Journal on Computing,* 17(5) 849-869.

Baumann. W, Birner. R, Haensler. J, Hartmann. P, and Stevens. A. G, 1981, Operating and Idle Times for Cyclic Multi-Machine Servicing. *The Industrial Robot,* 44-49.

Bedini. R, Lisini. G. G, and Sterpos. P, 1979, Optimal Programming of Working Cycles for Industrial Robots. *Journal of Mechanical Design, Transactions of the ASME,* 101, 250-279.

Chang. Y and Sullivan. R. S, 1990, Schedule Generation in a Dynamic Job Shop. *International Journal of Production Research,* 28 (1), 65-74.

Cheng. T.C.E and Sin. C. C. S, 1990, A state-of-the-art Review of Parallel-Machine Scheduling Research. *European Journal of Operations Research,* 47, 271-291.

Devedzic. V, 1990, A Knowledge-Based System for the Strategic Control Level of Robots in Flexible Manufacturing Cell. *International Journal of Flexible Manufacturing Systems*, 2(4), 263-287.

Drezner. Z and Nof. S. Y, 1984, Optimal Bin Picking and Insertion Plans for Assembly Robots. *IIE Transactions*, 16(3), 262-270.

Egbelu. P. J and Roy. N, 1988, Material Flow Control in AGV/unit Load based Production Lines. *International Journal of Production Research*, 26(1), 81-94.

Gere. G. S. Jr., 1966, Heuristics in Job Shop Scheduling. Management Science, 13, 167-190.

Gilmore. P and Gomory. R, 1964, Sequencing a One-State Variable Machine: A Solvable Case of the Travelling Salesman Problem. *Operations Research*, 12, 655-679.

Gupta. J. N. D, 1988, Two-Stage, Hybrid Flowshop Scheduling Problem. *Journal of Operational Research Society*, 39(4), 359-364.

Gupta. J. N. D, and Tunc. E. A, 1991, Schedules for a Two-Staged Hybrid Flowshop with Parallel Machines at the Second Stage. *International Journal of Operations Research*, 29(7), 1489-1502.

Hall. N. G, Kamoun. H and Sriskandarajah. C, 1994, Working paper to appear in *Operations Research.*

Kayis. B and Cheng Beng. G. D, 1994, A PC-Based Production Scheduling System using a Mixed Integer Programming Approach. *International Journal of Production Research*, 32(6), 1331-1346.

Kise. H, Shioyama. T and Ibaraki. T, 1991 Automated Two - Machine Flowshop Scheduling : A solvable case. *IIE Transactions*, 23(1), 10-16.

Kondoleon. A. S, 1979, Cycle Time Analysis of Robot Assembly Systems. *Proceedings of the Ninth Symposium on Industrial Robots*, 575-587.

Maimon. O. Z, and Nof. S.Y, 1985, Coordination of Robots Sharing Assembly Tasks. *Journal of Dynamic Systems Measurement and Control. Transactions of the ASME*, 107(4), 299-307.

Montazeri. M and Wassenhove. L. N. V, 1990, Analysis of Scheduling Rules for FMS. *International Journal of Production Research*, 28 (4), 785-802.

Nof. S. Y. F and Hanna. D, 1989, Operational Characteristics of Multi-Robot Systems with Cooperation. International Journal of Production Research, 27(3), 477-492.

Panwalkar. S. S and Iskander. W, 1977, A Survey of Scheduling Rules. *Operations Research*, 25(1), 45-61.

Rajendran. C, 1994, A Heuristic for Scheduling in Flowshop and Flowline based Manufacturing with Multicriteria. *International Journal of Production Research*, 32 (11), 2541-2558.

Sabuncuoglu. I and Hommertzheim. D. L, 1992, Dynamic Dispatching Algorithm for Scheduling Machines and Automated Guided Vehicles in a Flexible Manufacturing System. *International Journal of Production Research*, 30(5), 1059-1079.

Sarin. S. C, 1987, Mathematical Analysis of a Roboticized Production Cell. *INFOR*, 25(1), 46-56.

Sawik. T. J, 1995, Scheduling flexible flow lines with no in-process buffers. *Operations Research*, 33(5), 1357-1367.

Seidmann. A, Schweitzer. J. P and Nof. S. Y, 1985, Performance Evaluation of a Flexible Manufacturing Cell with Random Multiproduct Feedback Flow. *International Journal of Production Research*, 23(6), 1171-1184.

Seidmann. A and Nof. S. Y, 1989, Operational Analysis of an Autonomous Assembly Robotic Station. *IEEE Transactions on Robotics Automation*, 5(1), 4-15.

Sethi. S. P, Sriskandarajah. C, Sorger. G, Blazewicz. J and Kubiak. W, 1992, Sequencing of Parts and Robot Moves in a Robotic Cell. *The International Journal of Flexible Manufacturing Systems,* 4, 331-358.

Shaw. M. J, 1988, Dynamic Scheduling in Cellular Manufacturing Systems: A Framework for Networked Decision Making. *Journal of Manufacturing Systems,* 7(2), 83-94.

Shaw. M. J, Park. S and Raman. N, 1992, Intelligent Scheduling with Machine Learning Capabilities: The Induction of Scheduling Knowledge. *IIE Transactions,* May 1992, 156-168.

Sim. S. K, Yeo. K. T and Lee. W. H, 1994, An Expert Neural Network System for Dynamic Job Shop Scheduling. *International Journal of Production Research,* 32(8), 1759-1773.

Watson. E. F and Egbelu. P. J, 1989, Scheduling of Jobs through Parallel Non Identical Machine Cells. *Journal of Manufacturing Systems,* 8(1), 59-67.

Wilhelm W. E and Sarin. S. C , 1985, A Structure for Sequencing Robot Activities in Machine Loading Applications. *International Journal of Production Research,* 23(1), 47-64.

Yih. Y, Liang. T and Moskowitz. H, 1993, Robot Scheduling in a Circuit Board Production Line: An Hybrid OR/ANN Approach. *IIE Transactions,* 25 (2), 26-33.

Yih. Y and Thesen. S, 1991, Semi Markov Decision Models for Real-Time Scheduling. *International Journal of Operations Research,* 29(11), 2331-2346.

APPENDIX 1A

1A. Cycles with all machines empty at the start of a cycle

Presented here are the cycles starting with all machines in the cell being empty. The first case is explained in detail, which will facilitate understanding the other cycles, which are shown in a condensed form.

1.A.1) The loading sequence can be symbolically represented as:

I => M1 => I => M2 => M1 => P1 => M2 => P2 => P1 => O => P2 => O => I

The task sequence is shown in fully tabulated form below. The task numbers are also identified in Figure 1.A.1. The above cycle can be explained as follows:

Task number	Time	Robot Task
1.	δ	load machine M1
2.	2δ	go to input buffer
3.	3δ	load machine M2
4.	4δ	go to machine M1
	$4\delta+w1$	wait at machine M1 (w1)
5.	$5\delta+w1$	unload machine M1 and load machine P1
6.	$6\delta+w1$	go to machine M2
	$6\delta+w1+w2$	wait at machine M2 (w2)
7.	$7\delta+w1+w2$	unload machine M2 and load machine P2
8.	$8\delta+w1+w2$	go to machine P1
	$8\delta+w1+w2+w3$	wait at machine P1 (w3)
9.	$9\delta+w1+w2+w3$	unload machine P1
10.	$10\delta+w1+w2+w3$	go to machine P2
	$10\delta+w1+w2+w3+w4$	wait at machine P2 (w4)
11.	$11\delta+w1+w2+w3+w4$	unload machine P2
12.	$12\delta+w1+w2+w3+w4$	go to input buffer

Task number 12 marks the conclusion of one cycle and then the cycle repeats. The moves underlined are the basic six activities that the robot must perform in each cycle to produce one part of each type. At the end of the cycle we see that the cycle time equals the sum of the robot's total travel time and the waiting time at different machines.

The symbolic representation will be used from now on in Appendix 1A, 1B and 1C to describe the robot's moves.

The cycle time (CT) for this sequence is:

CT : $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where: $w1(m) = \max.\ \{ 0, a - 3\,\delta \}$

$w2(m) = \max.\ \{ 0, b - 3\,\delta - w1(m) \}$

$w3(m) = \max.\ \{ 0, c - 3\,\delta - w2(m) \}$

$w4(m) = \max.\ \{ 0, d - 3\,\delta - w3(m) \}$

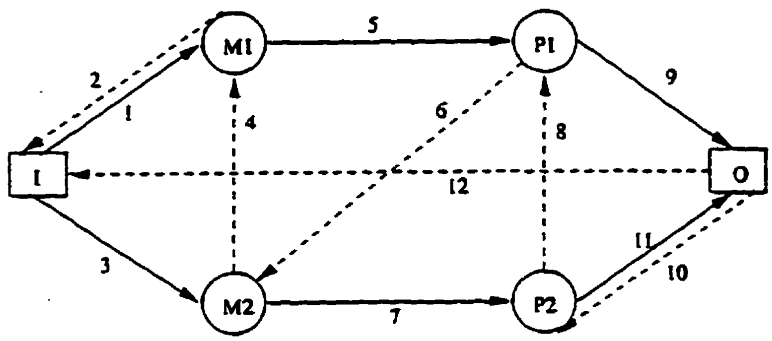


Figure 1A.1

1.A.2) The second option shown in Figure 1.A.2 is represented symbolically as :

I => M1 => I => M2 => M1 => P1 => M2 => P2 => wait => O => P1 => O => I

CT : $11\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where: $w1(m) = \max.\ \{ 0, a - 3\,\delta \}$,

$w2(m) = \max.\ \{ 0, b - 3\,\delta - w1(m) \}$,

$w3(m) = \max.\ \{ 0, c - 4\,\delta - w2(m) - w4(m) \}$, and

$w4(m) = \max.\ \{ 0, d \}$.

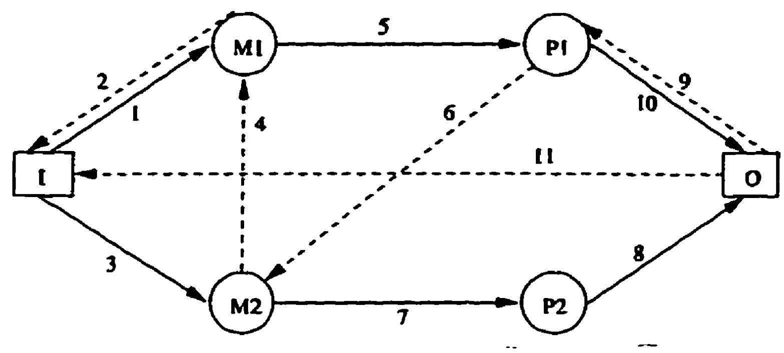


Figure 1A.2

1.A.3) The third option shown in Figure 1.A.3 is represented symbolically as :

I => M1 => I => M2 => M1 => P1 => wait => O => M2 => P2 => wait => O => I

CT : $10\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where: $w1(m) = \max.\ \{ 0, a - 3\,\delta \}$,

$w2(m) = \max.\ \{ 0, b - 4\,\delta - w1(m) - w3(m) \}$,

$w3(m) = \max.\ \{ 0, c \}$, and

$w4(m) = \max.\ \{ 0, d \}$.

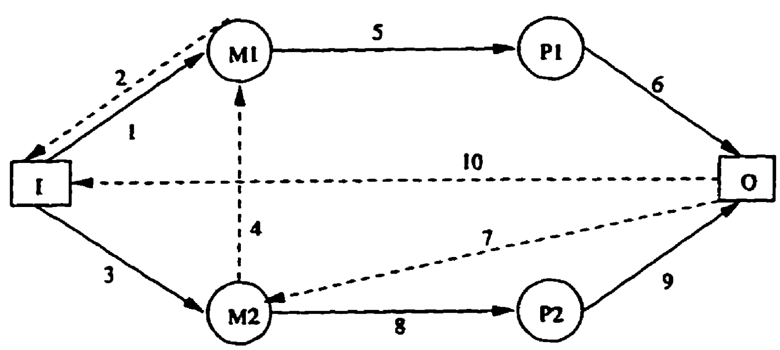


Figure 1A.3

1.A.4) The fourth option shown in Figure 1.A.4 is represented symbolically as :

I => M1 => I => M2 => wait => P2 => M1 => P1 => P2 => O => P1 => O => I

CT : $11\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where: $w1(m) = \max.\ \{ 0, a - 4\,\delta - w2(m) \}$,

$w2(m) = \max.\ \{ 0, b \}$,

w3(m) = max. { 0, c - 3 δ - w4(m) }, and

w4(m) = max. { 0, d - 3 δ - w1(m) }.

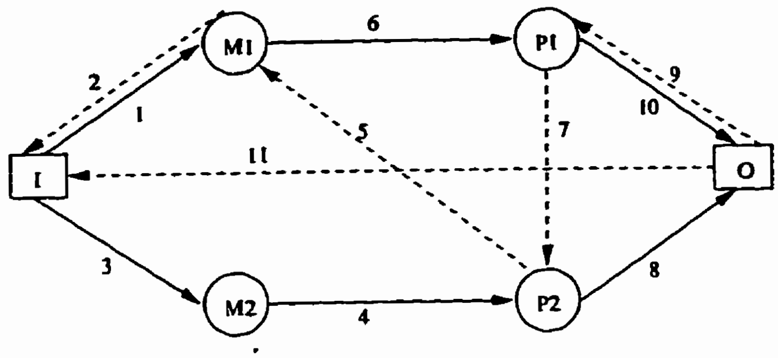


Figure 1A.4

1.A.5) The fifth option shown in Figure 1.A.5 is represented symbolically as :

I => M1 => I => M2 => wait => P2 => M1 => P1 => wait => O => P2 => O => I

CT : 10 δ + w1(m) + w2(m) + w3(m) + w4(m)

where: w1(m) = max. { 0, a - 4 δ - w2(m) },

w2(m) = max. { 0, b },

w3(m) = max. { 0, c }, and

w4(m) = max. { 0, d - 4 δ - w1(m) - w3(m) }.

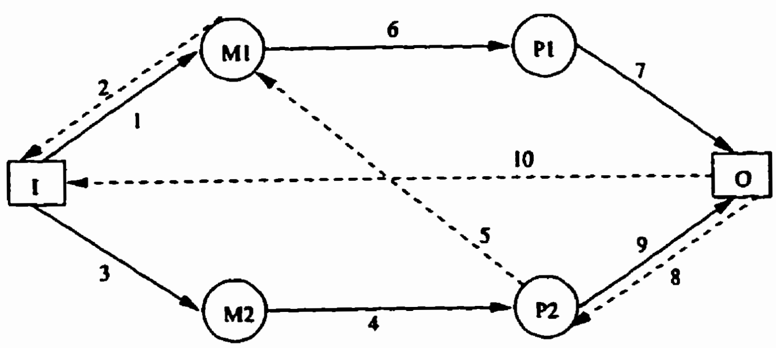


Figure 1A.5

1.A.6) The sixth option shown in Figure 1.A.6 is represented symbolically as :

I => M1 => I => M2 => wait => P2 => wait => O => M1 => P1 => wait => O => I

CT : 9 δ + w1(m) + w2(m) + w3(m) + w4(m)

where: w1(m) = max. { 0, a - 3 δ - w2(m) - w4(m)}, w2(m) = max. { 0, b },

w3(m) = max. { 0, c }, and

w4(m) = max. { 0, d }.

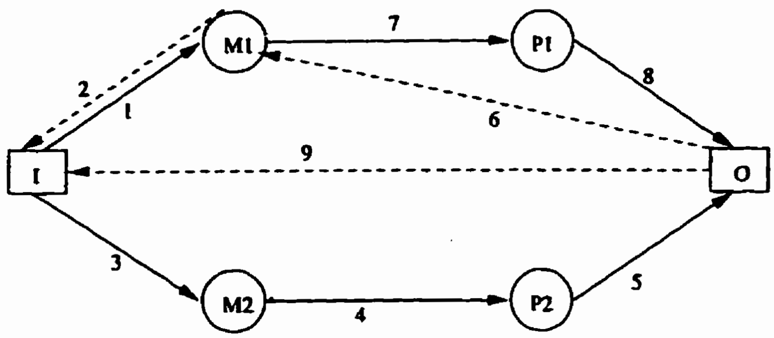


Figure 1A.6

1.A.7) The seventh option shown in Figure 1.A.7 is represented symbolically as :

<u>I => M1 => wait => P1</u> => <u>I => M2</u> => P<u>1 => O</u> => <u>M2 => P2 => wait => O</u> => I

CT : 10 δ + w1(m) + w2(m) + w3(m) + w4(m)

where: w1(m) = max. { 0, a }.

w2(m) = max. { 0, b - 3 δ - w3(m) },

w3(m) = max. { 0, c - 3 δ }, and

w4(m) = max. { 0, d }.

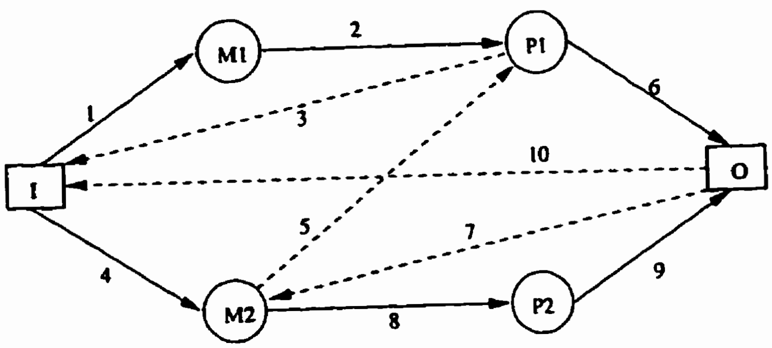


Figure 1A.7

1.A.8) The eight option shown in Figure 1.A.8 is represented symbolically as :

<u>I => M1 => wait => P1</u> => <u>I => M2 => wait => P2</u> => P<u>1 => O</u> => P<u>2 => O</u> => I

CT : 10 δ + w1(m) + w2(m) + w3(m) + w4(m) where: $w1(m) = \max.\ \{\ 0, a\ \}$,

$w2(m) = \max.\ \{\ 0, b\ \}$,

$w3(m) = \max.\ \{\ 0, c - 4\ \delta - w2(m)\ \}$, and

$w4(m) = \max.\ \{\ 0, d - 3\ \delta - w3(m)\ \}$.

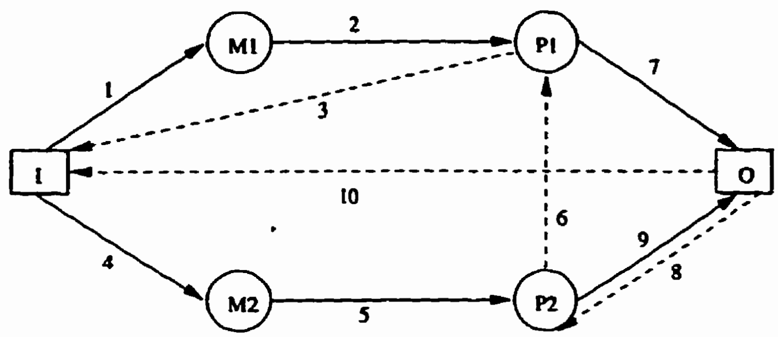


Figure 1A.8

1.A.9) The ninth option shown in Figure 1.A.9 is represented symbolically as :

I => M1 => wait => P1 => I => M2 => wait => P2 => wait => O => P1 => O => I

CT : $9\ \delta + w1(m) + w2(m) + w3(m) + w4(m)$

where: $w1(m) = \max.\ \{\ 0, a\ \}$,

$w2(m) = \max.\ \{\ 0, b\ \}$,

$w3(m) = \max.\ \{\ 0, c - 5\ \delta - w2(m) - w4(m)\ \}$, and

$w4(m) = \max.\ \{\ 0, d\ \}$.

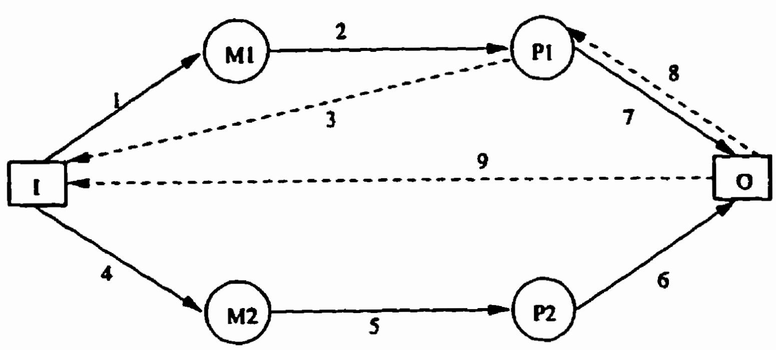


Figure 1A.9

1.A.10) The tenth option shown in Figure 1.A.10 is represented symbolically as :

I => M1 => wait => P1 => wait => O => I => M2 => wait => P2 => wait => O => I

CT : $8\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where: $w1(m) = \max.\ \{ 0, a \}$,

$w2(m) = \max.\ \{ 0, b \}$,

$w3(m) = \max.\ \{ 0, c \}$, and

$w4(m) = \max.\ \{ 0, d \}$.

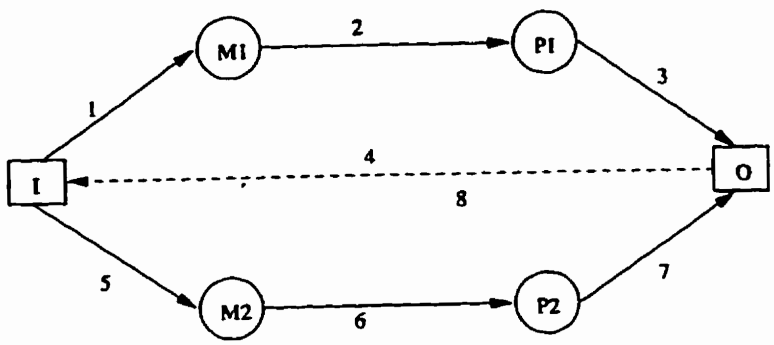


Figure 1A.10

APPENDIX 1B

1B Cycles when all the machines are loaded at the start of a cycle.

The robot has just finished loading a part at machine M1 and the cycle starts after this task. All the machines are loaded with parts before the cycle starts.

1.B.1) The first option shown in Figure 1.B.1 is represented symbolically as :

M1 => P1 => O => M1 => P1 => P2 => O => M2 => P2 => I => M2 => I => M1

CT: $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{\, 0, a - 3\,\delta - w3(m-1) \,\}$,

$w2(m) = \max.\ \{\, 0, b - 9\,\delta - w3(m-1) - w1(m) - w4(m-1) \,\}$,

$w3(m) = \max.\ \{\, 0, c - 9\,\delta - w4(m-1) - w2(m) \,\}$, and

$w4(m) = \max.\ \{\, 0, d - 9\,\delta - w3(m) - w1(m) \,\}$.

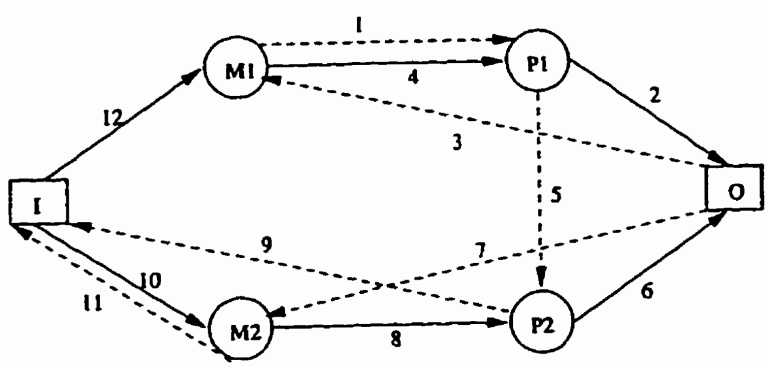


Figure. 1B.1

1.B.2) The second option shown in Figure 1.B.2 is represented symbolically as :

M1 => P1 => O => P2 => O => M2 => P2 => I => M2 => M1 => P1 => I => M1

CT: $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{\, 0, a - 9\,\delta - w3(m-1) - w4(m-1) - w2(m-1) \,\}$,

$w2(m) = \max.\ \{\, 0, b - 9\,\delta - w1(m) - w3(m-1) - w4(m-1) \,\}$,

$w3(m) = \max.\ \{\, 0, c - 3\,\delta \,\}$, and

$w4(m) = \max.\ \{\, 0, d - 9\,\delta - w1(m) - w3(m) \}$.

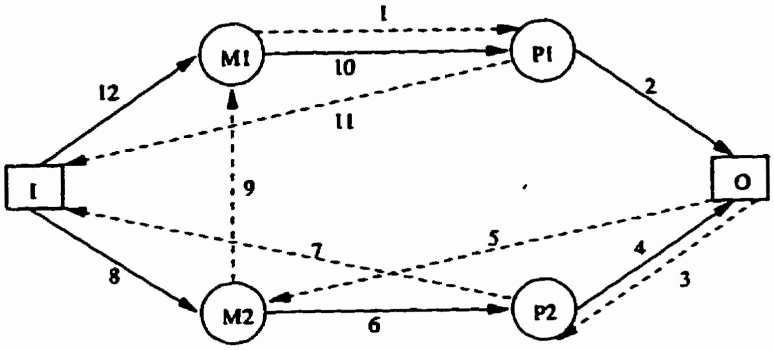


Figure. 1B.2

1.B.3) The third option shown in Figure 1.B.3 is represented symbolically as :

M1 => P1 => O => P2 => O => M1 => P1 => M2 => P2 => I => M2 => I => M1

CT: 12 δ + w1(m) + w2(m) + w3(m) + w4(m)

where w1(m) = max. { 0, a - 5 δ - w3(m-1) - w4(m-1) },

w2(m) = max. { 0, b - 9 δ - w3(m-1) - w4(m-1) - w1(m) },

w3(m) = max. { 0, c - 7 δ - w2(m) }, and

w4(m) = max. { 0, d - 7 δ - w3(m) }.

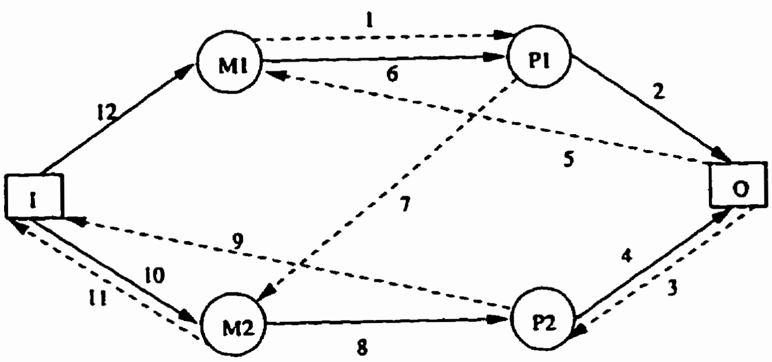


Figure. 1B.3

1.B.4) The fourth option shown in Figure 1.B.4 is represented symbolically as :

M1 => P2 => O => M2 => P2 => I => M2 => P1 => O => M1 => P1 => I => M1

CT: 12 δ + w1(m) + w2(m) + w3(m) + w4(m)

where w1(m) = max. { 0, a - 9 δ - w4(m-1) - w2(m-1) - w3(m-1) },

w2(m) = max. { 0, b - 9 δ - w3(m-1) - w1(m) - w4(m-1) },

w3(m) = max. { 0, c - 9 δ - w4(m-1) - w2(m) }, and

w4(m) = max. { 0, d - 9 δ - w3(m) - w1(m) }.

(This cycle is similiar to cycle 1.B.10)

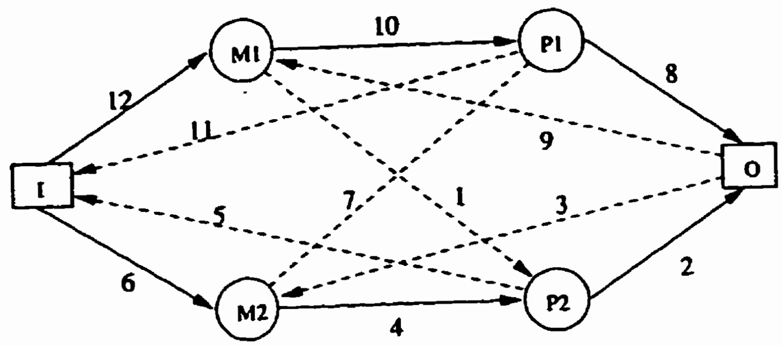


Figure. 1B.4

1.B.5) The fifth option shown in Figure 1.B.5 is represented symbolically as :

M1 => P2 => O => M2 => P2 => P1 => O => M1 => P1 => I => M2 => I => M1

CT: 12 δ + w1(m) + w2(m) + w3(m) + w4(m)

where w1(m) = max. { 0, a - 7 δ - w4(m-1) - w2(m-1) - w3(m-1) },

w2(m) = max. { 0, b - 5 δ - w4(m-1) },

w3(m) = max. { 0, c - 9 δ - w4(m-1) - w2(m) }, and

w4(m) = max. { 0, d - 9 δ - w3(m) - w1(m) }.

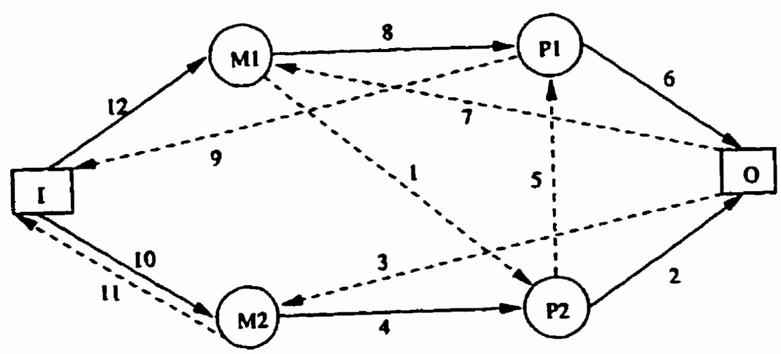


Figure. 1B.5

1.B.6) The sixth option shown in Figure 1.B.6 is represented symbolically as :

M1 => P2 => O => M2 => P2 => P1 => O => I => M2 => M1 => P1 => I => M1

CT: 12 δ + w1(m) + w2(m) + w3(m) + w4(m)

where w1(m) = max. { 0, a - 9 δ - w4(m-1) - w2(m-1) - w3(m-1) },

w2(m) = max. { 0, b - 7 δ - w1(m) - w4(m-1) },

$w3(m) = \max. \{ 0, c - 7\,\delta - w4(m\text{-}1) - w2(m) \}$, and

$w4(m) = \max. \{ 0, d - 9\,\delta - w3(m) - w1(m) \}$.

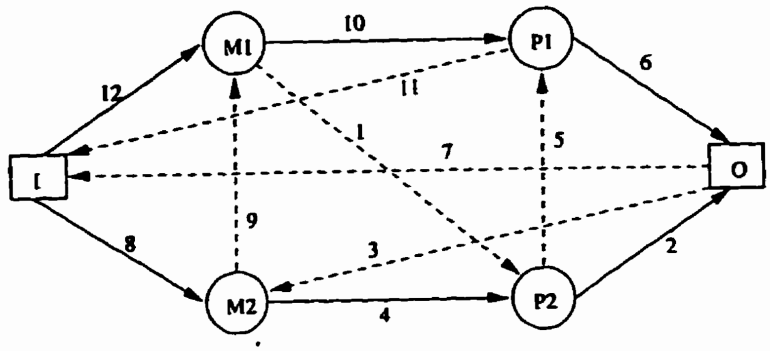


Figure. 1B.6

1.B.7) The seventh option shown in Figure 1.B.7 is represented symbolically as :

M1 => P2 => O => P1 => O => M1 => P1 => M2 => P2 => I => M2 => I => M1

CT: $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max. \{ 0, a - 5\,\delta - w4(m\text{-}1) - w3(m\text{-}1) \}$,

$w2(m) = \max. \{ 0, b - 9\,\delta - w4(m\text{-}1) - w3(m\text{-}1) - w1(m) \}$,

$w3(m) = \max. \{ 0, c - 9\,\delta - w2(m) - w4(m\text{-}1) \}$, and

$w4(m) = \max. \{ 0, d - 5\,\delta \}$.

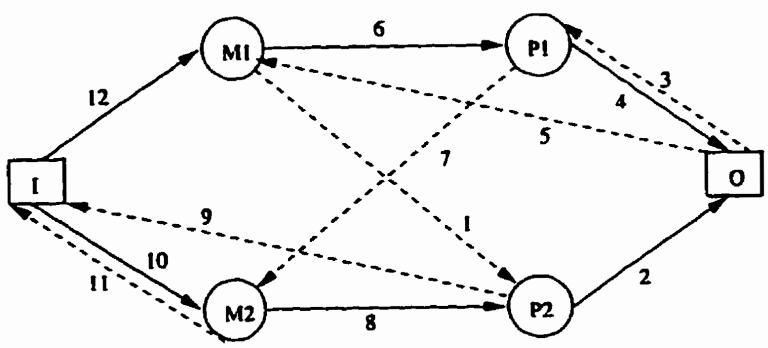


Figure. 1B.7

1.B.8) The eighth option shown in Figure 1.B.8 is represented symbolically as :

M1 => P2 => O => P1 => O => M2 => P2 => I => M2 => M1 => P1 => I => M1

CT: $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max. \{ 0, a - 9\,\delta - w4(m\text{-}1) - w3(m\text{-}1) - w2(m\text{-}1) \}$, w2(m) = max. { 0, b - 9 δ - w1(m) - w4(m-1) - w3(m-1) },

w3(m) = max. { 0, c - 5 δ - w4(m-1) },and

w4(m) = max. { 0, d - 7 δ - w1(m) }.

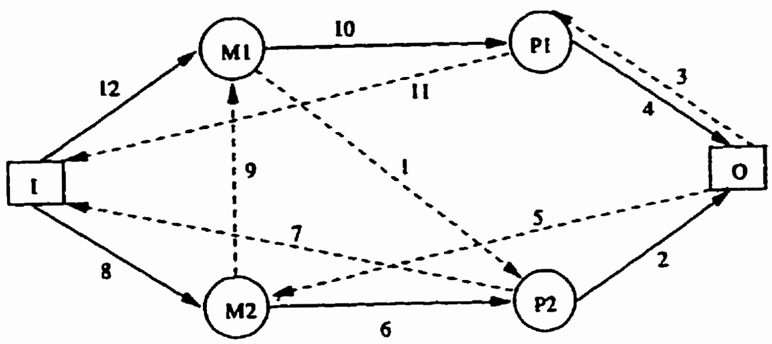


Figure. 1B.8

1.B.9) The ninth option shown in Figure 1.B.9 is represented symbolically as :

M1 => P2 => O => P1 => O => M2 => P2 => M1 => P1 => I => M2 => I => M1

CT: 12 δ + w1(m) + w2(m) + w3(m) + w4(m)

where w1(m) = max. { 0, a - 7 δ - w4(m-1) - w3(m-1) - w2(m-1) },

w2(m) = max. { 0, b - 7 δ - w4(m-1) - w3(m-1) },

w3(m) = max. { 0, c - 7 δ - w4(m-1) }, and

w4(m) = max. { 0, d - 7 δ - w1(m) }.

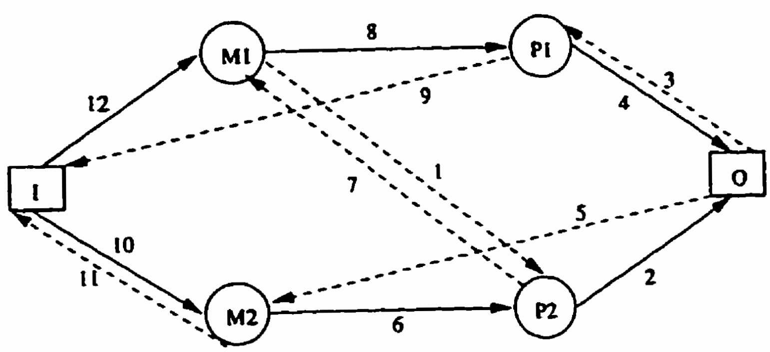


Figure. 1B.9

For the following cycles, the robot start from machine M2 as the robot has just finished loading a part on M2 and the cycle starts.

1.B.10) The tenth option shown in Figure 1.B.10 is represented symbolically as :

M2 => P1 => O => M1 => P1 => I => M1 => P2 => O => M2 => P2 => I => M2

CT: 12 δ + w1(m) + w2(m) + w3(m) + w4(m)

where w1(m) = max. { 0, a - 9 δ - w4(m-1) - w2(m-1) - w3(m-1) },

w2(m) = max. { 0, b - 9 δ - w3(m-1) - w1(m) - w4(m-1) },

w3(m) = max. { 0, c - 9 δ - w4(m-1) - w2(m) }, and

w4(m) = max. { 0, d - 9 δ - w3(m) - w1(m) }.

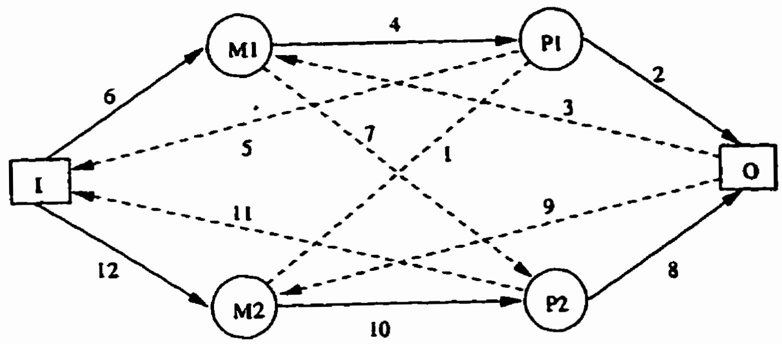


Figure. 1B.10

1.B.11) The eleventh option shown in Figure 1.B.11 is represented symbolically as :

M2 => P1 => O => M1 => P1 => P2 => O => M2 => P2 => I => M1 => I => M2

CT: 12 δ + w1(m) + w2(m) + w3(m) + w4(m)

where w1(m) = max. { 0, a - 5 δ - w3(m-1) },

w2(m) = max. { 0, b - 7 δ -w3(m-1) - w1(m) - w4(m-1) },

w3(m) = max. { 0, c - 9 δ - w4(m-1) - w2(m) }, and

w4(m) = max. { 0, d - 9 δ - w3(m) - w1(m) }.

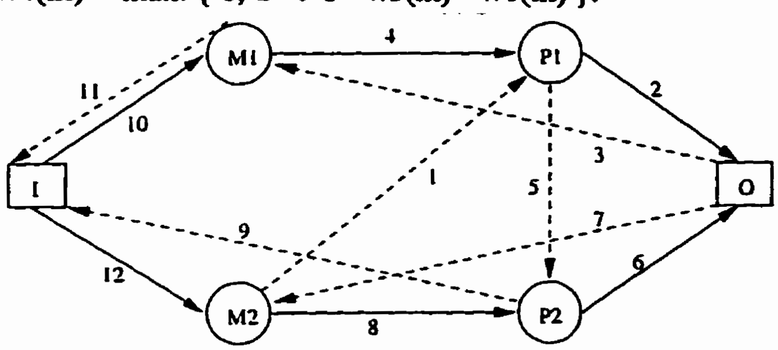


Figure. 1B.11

1.B.12) The twelveth option shown in Figure 1.B.12 is represented symbolically as :

M2 => P1 => O => M1 => P1 => P2 => O => I => M1 => M2 => P2 => I => M2

CT: $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{ 0,\ a - 7\,\delta - w2(m\text{-}1) - w3(m\text{-}1) \}$,

$w2(m) = \max.\ \{ 0,\ b - 9\,\delta - w3(m\text{-}1) - w1(m) - w4(m\text{-}1) \}$,

$w3(m) = \max.\ \{ 0,\ c - 9\,\delta - w4(m\text{-}1) - w2(m) \}$, and

$w4(m) = \max.\ \{ 0,\ d - 7\,\delta - w3(m) - w1(m) \}$.

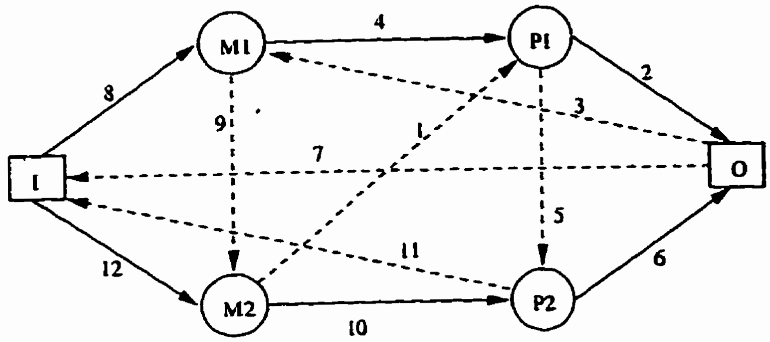


Figure. 1B.12

1.B.13) The thirteenth option shown in Figure 1.B.13 is represented symbolically as :

M2 => P1 => O => P2 => O => M2 => P2 => M1 => P1 => I => M1 => I => M2

CT: $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{ 0,\ a - 9\,\delta - w3(m\text{-}1) - w4(m\text{-}1) - w2(m\text{-}1) \}$,

$w2(m) = \max.\ \{ 0,\ b - 5\,\delta - w3(m\text{-}1) - w4(m\text{-}1) \}$,

$w3(m) = \max.\ \{ 0,\ c - 5\,\delta \}$, and

$w4(m) = \max.\ \{ 0,\ d - 9\,\delta - w3(m) \}$.

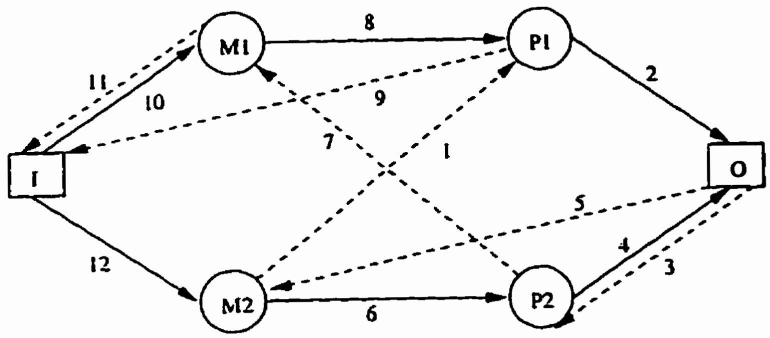


Figure. 1B.13

1.B.14) The fourteenth option shown in Figure 1.B.14 is represented symbolically as :

M2 => P1 => O => P2 => O => M1 => P1 => I => M1 => M2 => P2 => I => M2

CT: 12 δ + w1(m) + w2(m) + w3(m) + w4(m)

where w1(m) = max. { 0, a - 9 δ - w2(m-1) - w3(m-1) - w4(m-1) },

w2(m) = max. { 0, b - 9 δ - w3(m-1) - w4(m-1) - w1(m) },

w3(m) = max. { 0, c - 5 δ - w2(m) }, and

w4(m) = max. { 0, d - 5 δ - w3(m) }.

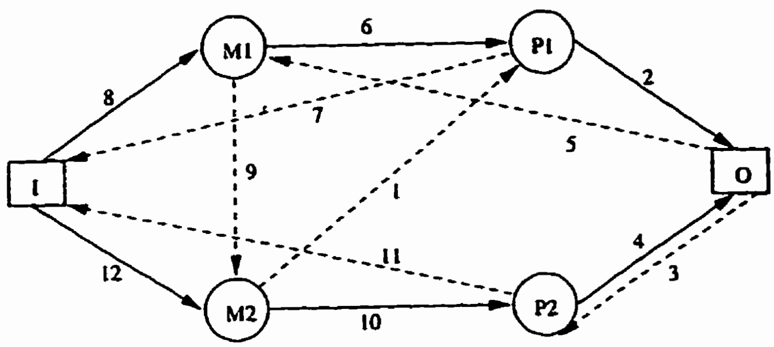


Figure. 1B.14

1.B.15) The fifteenth option shown in Figure 1.B.15 is represented symbolically as :

M2 => P1 => O => P2 => O => M1 => P1 => M2 => P2 => I => M1 => I => M2

CT: 12 δ + w1(m) + w2(m) + w3(m) + w4(m)

where w1(m) = max. { 0, a - 7 δ - w3(m-1) - w4(m-1) },

w2(m) = max. { 0, b - 7 δ - w3(m-1) - w4(m-1) - w1(m) },

w3(m) = max. { 0, c - 7 δ - w2(m) }, and

w4(m) = max. { 0, d - 7 δ - w3(m) }.

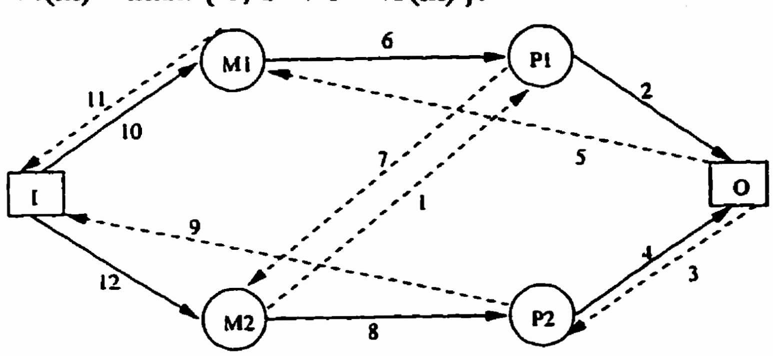


Figure. 1B.15

1.B.16) The sixteenth option shown in Figure 1.B.16 is represented symbolically as :

M2 => P2 => O => P1 => O => M1 => P1 => M2 => P2 => I => M1 => I => M2

CT: $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{ 0, a - 7\,\delta - w4(m-1) - w3(m-1) \}$,

$w2(m) = \max.\ \{ 0, b - 7\,\delta - w4(m-1) - w3(m-1) - w1(m) \}$,

$w3(m) = \max.\ \{ 0, c - 9\,\delta - w2(m) - w4(m-1) \}$, and

$w4(m) = \max.\ \{ 0, d - 5\,\delta \}$.

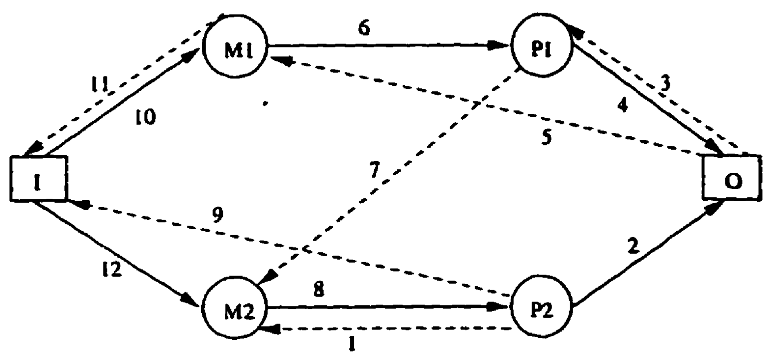


Figure. 1B.16

1.B.17) The seventeenth option shown in Figure 1.B.17 is represented symbolically as :

M2 => P2 => O => P1 => O => M2 => P2 => M1 => P1 => I => M1 => I => M2

CT: $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{ 0, a - 9\,\delta - w4(m-1) - w3(m-1) - w2(m-1) \}$,

$w2(m) = \max.\ \{ 0, b - 5\,\delta - w4(m-1) - w3(m-1) \}$,

$w3(m) = \max.\ \{ 0, c - 7\,\delta - w4(m-1) \}$, and

$w4(m) = \max.\ \{ 0, d - 7\,\delta - w1(m) \}$.

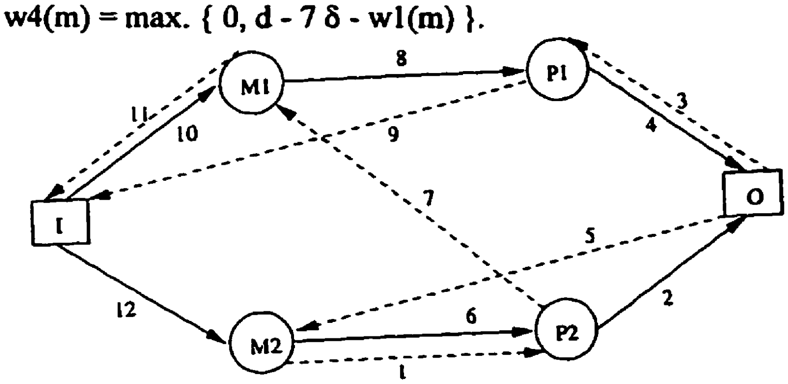


Figure. 1B.17

1.B.18) The eighteenth option shown in Figure 1.B.18 is represented symbolically as :

M2 => P2 => O => M2 => P2 => P1 => O => M1 => P1 => I => M1 => I => M2

CT: $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{\ 0,\ a - 9\,\delta - w4(m\text{-}1) - w2(m\text{-}1) - w3(m\text{-}1)\ \}$,

$w2(m) = \max.\ \{\ 0,\ b - 3\,\delta - w4(m\text{-}1)\ \}$,

$w3(m) = \max.\ \{\ 0,\ c - 9\,\delta - w4(m\text{-}1) - w2(m)\ \}$, and

$w4(m) = \max.\ \{\ 0,\ d - 9\,\delta - w3(m) - w1(m)\ \}$.

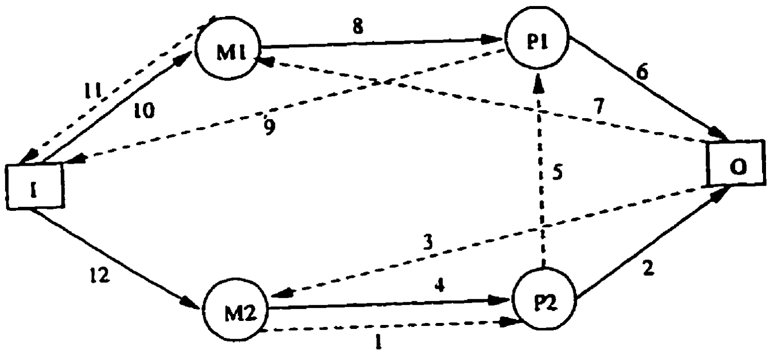


Figure. 1B.18

APPENDIX 1C

1C Cycles with 1, 2 or 3 machines loaded at the start of a cycle

1C.1) Condition (b) listed on page 18 of Section 3.2.3 is shown in figure 1C.1. The symbolic representation of robot move sequence is:

I => M1 => M2 => P2 => I => M2 => M1 => P1 => P2 => O => P1 => O => I

$CT = 12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{\ 0, a - 5\,\delta - w2(m-1)\ \}$,

$w2(m) = \max.\ \{\ 0, b - 9\,\delta - w1(m) - w4(m-1) - w3(m-1)\ \}$,

$w3(m) = \max.\ \{\ 0, c - 3\,\delta - w4(m-1)\ \}$, and

$w4(m) = \max.\ \{\ 0, d - 5\,\delta - w1(m)\}$.

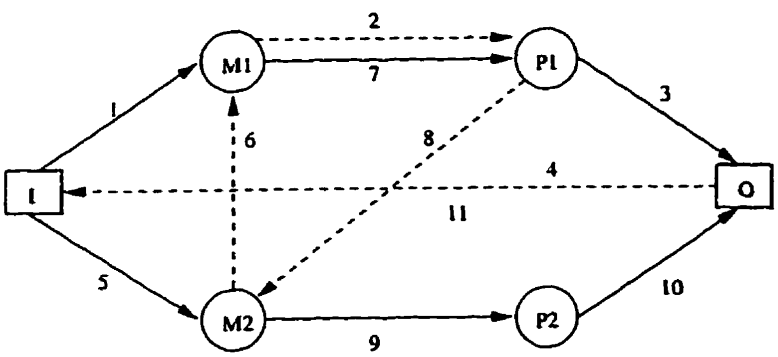


Figure. 1C.1

1C.2) Condition (c) listed on page 18 of Section 3.2.3 is shown in figure 1C.2. The symbolic representation of robot move sequence is:

I => M1 => P1 => O => I => M2 => M1 => P1 => M2 => P2 => wait => O => I

$CT = 11\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{\ 0, a - 5\,\delta - w3(m-1)\ \}$,

$w2(m) = \max.\ \{\ 0, b - 3\,\delta - w1(m)\ \}$,

$w3(m) = \max.\ \{\ 0, c - 6\,\delta - w4(m-1) - w2(m)\ \}$, and

$w4(m) = \max.\ \{\ 0, d\ \}$.

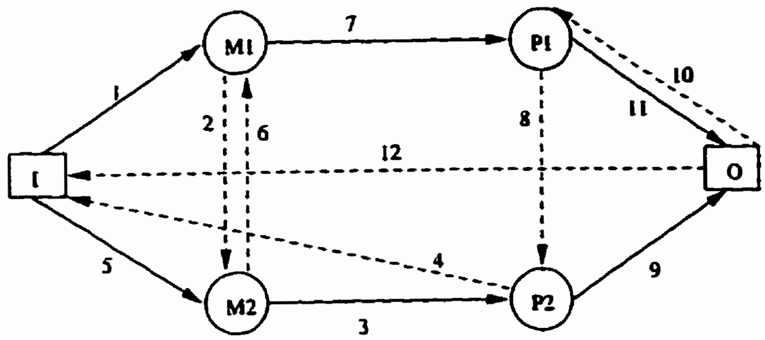


Figure. 1C.2

1C.3) Condition (d) listed on page 18 of Section 3.2.3 is shown in figure 1C.3. The symbolic representation of robot move sequence is:

I => M1 => P2 => O => M1 => P1 => I => M2 => P1 => O => M2 => P2 => I

CT = $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{\ 0, a - 3\,\delta - w4(m-1)\ \}$,

$w2(m) = \max.\ \{\ 0, b - 3\,\delta - w3(m-1)\ \}$,

$w3(m) = \max.\ \{\ 0, c - 3\,\delta\ \}$, and

$w4(m) = \max.\ \{\ 0, d - 3\,\delta\ \}$.

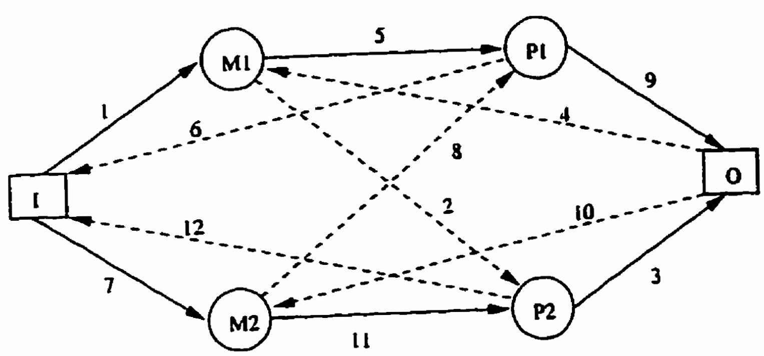


Figure. 1C.3

1C.4) Condition (f) listed on page 18 of Section 3.2.3 is shown in figure 1C.4. The symbolic representation of robot move sequence is:

I => M2 => P1 => O => M2 => P2 => M1 => P1 => I => M1 => P2 => O => I

CT = $12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{\ 0, a - 9\,\delta - w4(m-1) - w3(m-1) - w2(m-1)\ \}$,

$w2(m) = \max.\ \{\ 0, b - 3\,\delta - w3(m-1)\ \}$, w3(m) = max. { 0, c - 7 δ - w4(m-1) }, and

w4(m) = max. { 0, d - 5 δ - w1(m) }.

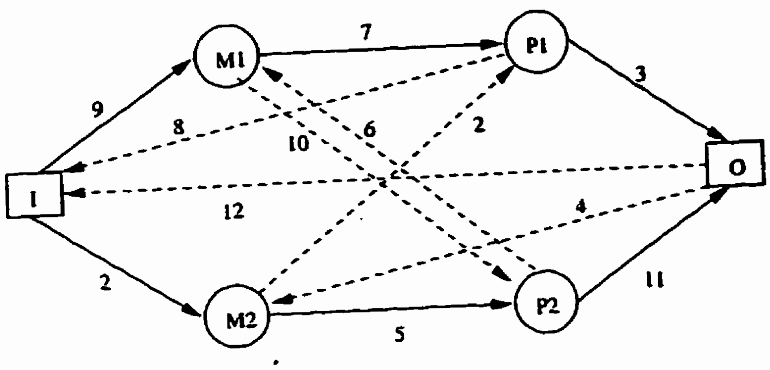


Figure. 1C.4

1C.5) Condition (g) listed on page 18 of Section 3.2.3 is shown in figure 1C.5. The symbolic representation of robot move sequence is:

P2 => O => I => M2 => M1 => P1 => M2 => P2 => P1 => O => I => M1 => P2

CT = 12 δ + w1(m) + w2(m) + w3(m) + w4(m)

where w1(m) = max. { 0, a - 5 δ - w4(m-1) }

w2(m) = max. { 0, b - 3 δ - w1(m) }

w3(m) = max. { 0, c - 3 δ - w2(m) }

w4(m) = max. { 0, d - 5 δ - w3(m) }

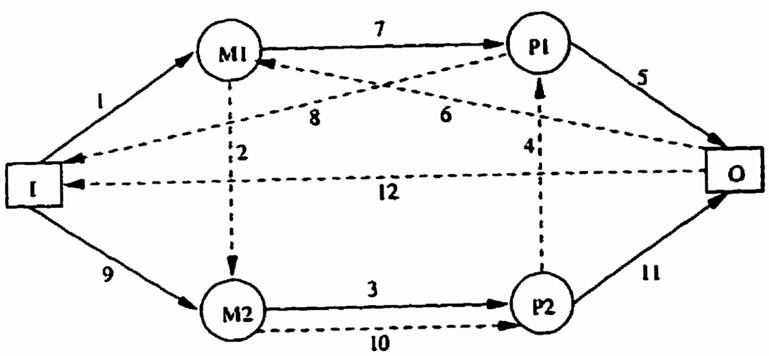


Figure 1C.5

1C.6) Condition (j) listed on page 19 of Section 3.2.3 is shown in figure 1C.6. The symbolic representation of robot move sequence is:

I => M1 => M2 => P2 => P1 => O => M1 => P1 => I => M2 => P2 => O => I

$CT = 12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{\ 0, a - 5\,\delta - w2(m-1) - w3(m-1)\ \}$,

$w2(m) = \max.\ \{\ 0, b - 5\,\delta - w4(m-1)\ \}$,

$w3(m) = \max.\ \{\ 0, c - 9\,\delta - w4(m-1) - w2(m)\ \}$, and

$w4(m) = \max.\ \{\ 0, d - 7\,\delta - w3(m) - w1(m)\ \}$.

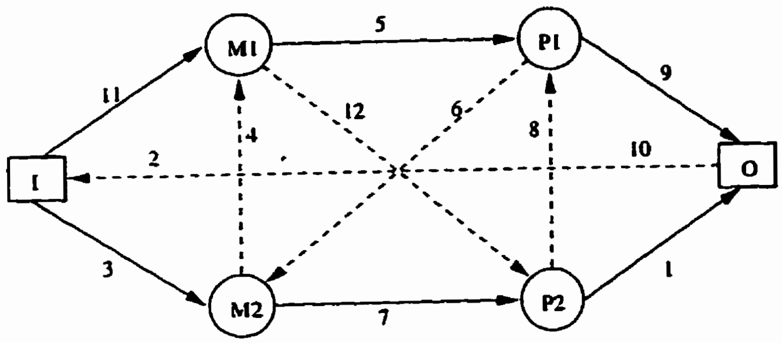


Figure. 1C.6

1C.7) Condition (l) listed on page 19 of Section 3.2.3 is shown in figure 1C.7. The symbolic representation of robot move sequence is:

<u>P2 => O</u> => <u>M2 => P2</u> => <u>I => M2</u> => <u>M1 => P1 => wait => O</u> => <u>I => M1</u> => P2

$CT = 11\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{\ 0, a - 7\,\delta - w4(m-1) - w2(m-1)\ \}$,

$w2(m) = \max.\ \{\ 0, b - 8\,\delta - w1(m) - w3(m-1) - w4(m-1)\ \}$,

$w3(m) = \max.\ \{\ 0, c\ \}$, and

$w4(m) = \max.\ \{\ 0, d - 8\,\delta - w1(m) - w3(m)\ \}$.

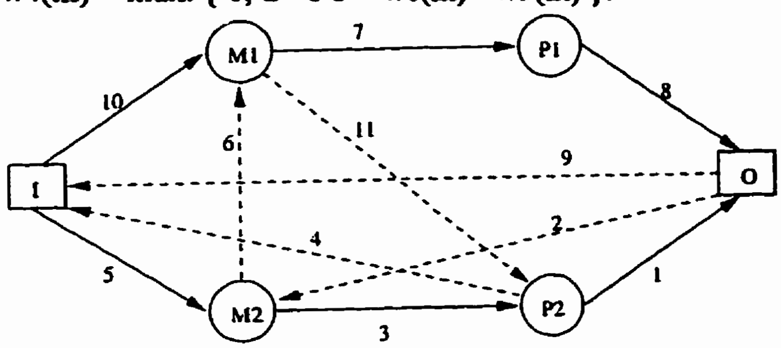


Figure. 1C.7

1C.8) Condition (n) listed on page 19 of Section 3.2.3 is shown in figure 1C.8. The symbolic representation of robot move sequence is:

P2 => O => M2 => P2 => P1 => O => I => M1 => I => M2 => M1 => P1 => P2

$CT = 12\,\delta + w1(m) + w2(m) + w3(m) + w4(m)$

where $w1(m) = \max.\ \{\, 0, a - 3\,\delta \,\}$,

$w2(m) = \max.\ \{\, 0, b - 5\,\delta - w1(m) - w4(m-1)\}$,

$w3(m) = \max.\ \{\, 0, c - 5\,\delta - w4(m-1) - w2(m) \,\}$, and

$w4(m) = \max.\ \{\, 0, d - 9\,\delta - w3(m) - w1(m) \,\}$.

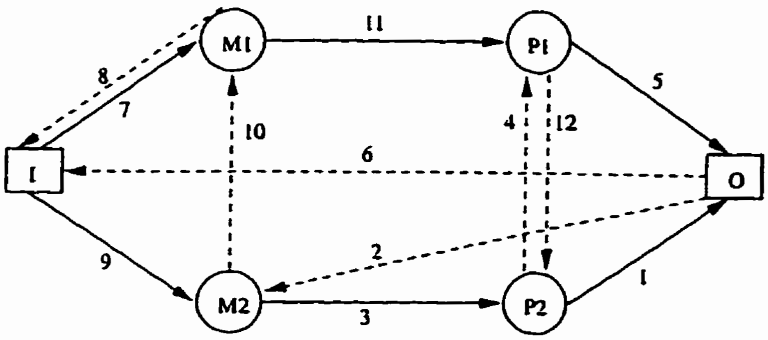


Figure. 1C.8

APPENDIX 2

The "**bold**" values in all the tables given in this appendix are representative of the minimum cycle time. Values shown under columns M1, M2, P1 and P2 represent the machining times. The values in the rest of the columns represent the cycle times for the cycles shown in Appendix 1-A,B and C. All the values shown in this Table are given in seconds.

Table 2.1 Condition: All the machines are empty at the start of a cycle,
Robot Travel Time, $\delta = 5$ s

M1	M2	P1	P2	1A.1	1A.2	1A.3	1A.4	1A.5	1A.6	1A.7	1A.8	1A.9	1A.10
78	90	66	82	202	212	261	212	206	283	285	285	295	356
40	27	30	69	139	149	174	136	126	171	174	171	181	206
64	93	43	17	140	150	159	176	186	198	209	209	219	257
76	97	90	88	215	225	289	227	237	320	296	296	306	391
22	32	78	80	142	152	215	152	160	235	215	169	179	252
83	18	61	58	174	181	237	164	174	182	237	194	204	260
15	83	43	38	151	161	151	166	176	209	171	171	181	219
91	84	41	20	162	152	187	165	175	190	230	230	240	276
33	33	16	93	156	166	177	166	156	187	194	194	204	215
37	94	28	63	187	197	187	197	187	230	229	229	239	262
43	23	23	89	162	172	190	152	142	180	190	190	200	218
27	45	77	55	134	140	194	162	172	222	197	162	172	244
28	97	30	83	210	220	210	220	210	255	243	243	253	278
13	25	92	43	137	127	185	157	167	205	183	135	126	213
50	32	70	58	150	148	213	142	152	205	213	175	185	250

Table 2.1 continued from the previous page

M1 M2 P1 P2	1A.1	1A.2	1A.3	1A.4	1A.5	1A.6	1A.7	1A.8	1A.9	1A.10
37 53 60 92	175	185	224	185	175	250	224	217	227	282
22 93 53 20	143	153	143	186	196	211	170	170	180	228
68 76 66 30	164	154	199	182	192	217	209	209	219	280
30 25 29 38	98	108	132	103	104	137	132	128	138	162
89 56 73 25	192	182	222	182	192	199	222	205	215	283
29 24 30 67	126	136	161	131	121	166	161	155	165	190
81 52 76 51	187	177	243	177	187	224	243	219	229	300
79 20 79 20	188	178	213	178	188	188	213	188	178	238
36 26 36 26	102	102	133	102	112	133	133	123	133	164
63 97 63 97	224	234	258	234	224	302	292	292	302	360
58 73 58 73	176	186	224	186	181	249	239	239	249	302
51 67 86 85	182	192	257	193	203	283	257	238	248	329
55 67 60 86	183	193	236	193	183	258	243	243	253	308
79 36 33 69	178	188	216	145	142	183	219	219	229	257
82 97 43 56	168	178	216	145	155	155	216	180	190	228
98 58 48 24	176	166	205	166	176	176	215	215	225	268
55 73 86 97	200	210	273	210	209	301	273	260	270	351
32 95 41 22	147	157	147	176	186	203	184	184	194	230
84 99 54 80	209	219	253	219	209	278	298	298	308	357
33 89 15 23	142	152	142	152	154	172	180	180	190	200
47 92 54 96	218	228	232	228	218	287	270	270	280	329
88 63 84 20	202	192	227	192	202	212	227	206	216	295
82 56 64 44	176	166	225	166	176	209	225	217	227	286

Table 2.1 continued from the previous page

M1	M2	P1	P2	1A.1	1A.2	1A.3	1A.4	1A.5	1A.6	1A.7	1A.8	1A.9	1A.10
43	67	83	40	156	147	201	190	200	235	201	185	195	273
84	31	41	74	188	198	234	145	155	191	234	224	234	270
50	79	52	96	205	215	233	215	205	272	260	160	270	317
68	92	70	84	206	216	257	216	212	291	279	179	289	354
35	42	67	25	132	122	162	149	159	179	162	137	147	209
65	70	98	29	193	183	227	208	218	242	227	199	209	302
44	61	96	67	170	168	242	197	207	269	242	207	217	308
82	89	40	52	171	181	209	181	179	226	258	158	268	303
70	54	41	23	141	133	169	135	145	163	182	182	192	228
61	96	85	57	183	198	238	221	231	283	249	249	259	339

Table 2.2 Condition : All the machines are loaded at the start of the cycle,
Robot Travel Time,.δ = 5 secs

M1	M2	P1	P2	1B.1	1B.2	1B.3	1B.4	1B.5	1B.6	1B.7	1B.8	1B.9
78	90	66	82	123	111	113	**105**	125	115	117	107	115
40	27	30	69	85	**84**	94	**84**	**84**	**84**	104	94	94
64	93	43	17	109	**108**	**108**	**108**	128	118	**108**	**108**	118
76	97	90	88	121	135	115	**112**	132	122	122	125	122
22	32	78	80	**95**	123	105	**95**	**95**	103	115	113	105
83	18	61	58	128	106	118	**98**	108	**98**	118	**98**	108
15	83	43	38	**98**	**98**	**98**	**98**	118	108	**98**	**98**	108
91	84	41	20	136	**106**	126	**106**	119	109	126	**106**	116
33	33	16	93	**108**	**108**	118	**108**	**108**	**108**	128	118	118
37	94	28	63	**109**	**109**	**109**	**109**	129	119	**109**	**109**	119
43	23	23	89	112	**104**	114	**104**	**104**	**104**	124	114	114
27	45	77	55	**92**	122	102	**92**	**92**	102	**92**	112	102
28	97	30	83	**112**	**112**	**112**	**112**	132	122	118	**112**	122
13	25	92	43	**107**	137	117	**107**	**107**	117	**107**	127	117
50	32	70	58	95	115	95	**85**	**85**	95	93	105	95
37	53	60	92	**107**	**107**	117	**107**	**107**	**107**	127	117	117
22	93	53	20	**108**	**108**	**108**	**108**	128	118	**108**	**108**	118
62	26	90	71	107	135	115	**105**	**105**	115	106	125	115
40	34	85	61	**100**	130	110	**100**	110	110	**100**	120	110
68	76	66	30	113	111	103	**91**	110	101	103	101	101
30	25	29	38	75	74	65	**60**	**60**	**60**	73	64	63
89	56	73	25	134	118	124	**104**	114	**104**	124	108	114

Table 2.2 continued from the previous page:

M1	M2	P1	P2	**1B.1**	**1B.2**	**1B.3**	**1B.4**	**1B.5**	**1B.6**	**1B.7**	**1B.8**	**1B.9**
29	24	30	67	**82**	**82**	92	**82**	**82**	**82**	102	92	92
81	52	76	51	126	121	116	**96**	**106**	101	116	111	**106**
79	20	79	20	124	124	114	**94**	104	104	114	114	104
36	26	36	26	81	81	71	**60**	61	61	71	71	61
63	97	63	97	**112**	**112**	122	**112**	132	122	132	122	122
58	73	58	73	103	103	98	**88**	108	98	108	98	98
51	67	86	85	**101**	131	111	**101**	102	111	120	121	111
55	67	60	86	**101**	105	111	**101**	102	**101**	121	111	111
79	36	33	69	124	94	114	**94**	104	**94**	114	**94**	104
82	97	43	56	127	**112**	117	**112**	132	122	118	**112**	122
98	58	48	24	143	**113**	133	**113**	123	**113**	133	**113**	123
55	73	86	97	**112**	131	122	**112**	**112**	**112**	132	122	122
32	95	41	22	**110**	**110**	**110**	**110**	130	120	**110**	**110**	120
84	99	54	80	129	**114**	119	**114**	134	124	119	**114**	124
33	89	15	23	**104**	**104**	**104**	**104**	124	114	**104**	**104**	114
47	92	54	96	111	111	121	**111**	127	117	131	121	121
88	63	84	20	133	129	123	**103**	113	109	123	119	113
82	56	64	44	127	109	117	**97**	107	**97**	117	99	107
43	67	83	40	**98**	128	108	**98**	102	108	**98**	118	108
84	31	41	74	129	**99**	119	**99**	109	**99**	119	**99**	109
50	79	52	96	**111**	**111**	121	**111**	114	**111**	131	121	121
68	92	70	84	113	115	109	**107**	127	117	119	109	117
35	42	67	25	**82**	112	92	**82**	**82**	92	**82**	102	92

Table 2.2 continued from the previous page:

M1	M2	P1	P2	**1B.1**	**1B.2**	**1B.3**	**1B.4**	**1B.5**	**1B.6**	**1B.7**	**1B.8**	**1B.9**
65	70	98	29	**113**	143	123	**113**	**113**	123	**113**	133	123
44	61	96	67	**111**	141	121	**111**	**111**	121	**111**	131	121
82	89	40	52	127	**104**	117	**104**	124	114	117	**104**	114
70	54	41	23	115	86	105	**85**	95	**85**	105	**85**	95
61	96	85	57	**111**	130	**111**	**111**	131	121	**111**	120	121

Table 2.3 Condition : All the machines are loaded at the start of the cycle,
Robot Travel Time, δ = 5 secs

M1	M2	P1	P2	1B.10	1B.11	1B.12	1B.13	1B.14	1B.15	1B.16	1B.17	1B.18
78	90	66	82	**105**	115	107	125	117	115	117	125	135
40	27	30	69	**84**	**84**	94	**84**	104	94	104	94	**84**
64	93	43	17	**108**	118	**108**	128	**108**	118	118	128	138
76	97	90	88	**112**	122	**112**	132	122	122	122	132	142
22	32	78	80	**95**	**95**	105	113	115	105	115	105	**95**
83	18	61	58	**98**	118	108	**98**	**98**	108	108	**98**	**98**
15	83	43	38	**98**	108	**98**	118	**98**	108	108	118	128
91	84	41	20	**106**	126	116	119	**106**	116	116	119	129
33	33	16	93	**108**	**108**	118	**108**	128	118	128	118	**108**
37	94	28	63	**109**	119	**109**	129	**109**	119	119	129	139
43	23	23	89	**104**	**104**	114	**104**	124	114	124	114	**104**
27	45	77	55	**92**	**92**	**92**	112	102	102	**92**	102	**92**
28	97	30	83	**112**	122	**112**	132	118	122	118	132	142
13	25	92	43	**107**	**107**	**107**	127	117	117	**107**	117	**107**
50	32	70	58	**85**	88	**85**	105	95	95	93	95	**85**
37	53	60	92	**107**	**107**	117	**107**	127	117	127	117	**107**
22	93	53	20	**108**	118	**108**	128	**108**	118	118	128	138
62	26	90	71	**105**	**105**	**105**	125	115	115	106	115	**105**
40	34	85	61	**100**	**100**	**100**	120	110	110	**100**	110	**100**
68	76	66	30	**91**	103	93	111	**91**	101	101	111	121
30	25	29	38	**60**	65	63	64	73	63	73	63	70
89	56	73	25	**104**	124	114	108	**104**	114	114	**104**	**104**

Table 2.3 continued from the previous page:

M1 M2 P1 P2	1B.10	1B.11	1B.12	1B.13	1B.14	1B.15	1B.16	1B.17	1B.18
81 52 76 51	96	116	106	111	101	106	106	101	97
79 20 79 20	94	114	104	114	104	104	104	104	94
36 26 36 26	60	71	61	71	61	61	61	61	71
63 97 63 97	112	122	122	132	132	122	132	132	142
58 73 58 73	88	98	98	108	108	98	108	108	118
51 67 86 85	101	101	110	121	120	111	120	111	112
55 67 60 86	101	101	111	102	121	111	121	111	112
79 36 33 69	94	114	104	94	104	104	104	94	94
82 97 43 56	112	122	112	132	112	122	122	132	142
98 58 48 24	113	133	123	113	113	123	123	113	113
55 73 86 97	112	112	122	121	132	122	132	122	118
32 95 41 22	110	120	110	130	110	120	120	130	140
84 99 54 80	114	124	114	134	115	124	124	134	144
33 89 15 23	104	114	104	124	104	114	114	124	134
47 92 54 96	111	117	121	127	131	121	131	127	137
88 63 84 20	103	123	113	119	109	113	113	109	108
82 56 64 44	97	117	107	99	97	107	107	97	101
43 67 83 40	98	98	98	118	108	108	98	108	112
84 31 41 74	99	119	109	99	109	109	109	99	99
50 79 52 96	111	111	121	114	131	121	131	121	124
68 92 70 84	107	117	109	127	119	117	119	127	137
35 42 67 25	82	82	82	102	92	98	82	92	87
65 70 98 29	113	113	113	133	123	123	113	123	115

Table 2.3 continued from the previous page:

M1 M2 P1 P2	**1B.10**	**1B.11**	**1B.12**	**1B.13**	**1B.14**	**1B.15**	**1B.16**	**1B.17**	**1B.18**
44 61 96 67	**111**	**111**	**111**	131	121	121	**111**	121	**111**
82 89 40 52	**104**	117	107	124	**104**	114	114	124	134
70 54 41 23	**85**	105	95	89	**85**	95	95	89	99
61 96 85 57	**111**	121	**111**	131	**111**	121	121	131	141

Table 2.4 Condition: Some machines are loaded at the start of a cycle,
$\delta = 5$ secs

M1	M2	P1	P2	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8
78	90	66	82	212	164	202	192	138	192	164	164
40	27	30	69	139	104	129	116	94	104	90	90
64	93	43	17	150	127	187	138	128	138	127	128
76	97	90	88	224	186	214	204	148	204	186	186
22	32	78	80	152	123	188	132	105	132	133	120
83	18	61	58	172	164	174	132	118	164	164	164
15	83	43	38	162	98	151	141	118	141	98	118
91	84	41	20	144	152	205	129	126	152	152	152
33	33	16	93	166	128	156	146	118	146	108	108
37	94	28	63	197	109	187	177	129	177	109	129
43	23	23	89	162	124	147	132	114	132	104	104
27	45	77	55	140	124	162	120	92	124	132	124
28	97	30	83	220	118	210	200	132	200	112	132
13	25	92	43	117	137	165	110	107	137	147	127
50	32	70	58	138	140	158	110	88	140	140	140
37	53	60	92	185	127	182	165	117	165	117	107
22	93	53	20	153	108	145	138	128	138	108	128
62	26	90	71	163	172	191	117	105	172	172	172
40	34	85	61	140	154	176	115	100	154	154	154
68	76	66	30	146	154	174	126	111	154	154	154
30	25	29	38	103	79	97	83	65	83	84	79
89	56	73	25	144	182	192	104	124	182	182	182

Table 2.4 continued from the previous page:

M1	M2	P1	P2	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8
29	24	30	67	131	102	127	111	92	111	85	82
81	52	76	51	162	177	187	123	116	177	177	177
79	20	79	20	129	178	188	104	114	178	178	178
36	26	36	26	92	92	102	72	71	92	92	92
63	97	63	97	234	146	224	214	146	214	146	146
58	73	58	73	186	138	176	166	120	166	136	136
51	67	86	85	192	157	201	172	119	172	157	157
55	67	60	86	193	135	183	173	122	173	135	135
79	36	33	69	178	132	145	125	114	132	129	132
82	97	43	56	193	145	209	173	135	173	145	145
98	58	48	24	152	166	186	113	133	166	166	166
55	73	86	97	210	161	213	190	130	190	161	161
32	95	41	22	157	110	157	140	130	140	110	130
84	99	54	80	219	158	213	199	149	199	158	158
33	89	15	23	152	104	152	134	134	134	104	124
47	92	54	96	228	131	218	208	135	208	121	127
88	63	84	20	148	192	202	109	123	192	192	192
82	56	64	44	155	166	176	120	117	166	166	166
43	67	83	40	147	146	156	127	102	146	146	146
84	31	41	74	188	145	155	125	119	145	145	145
50	79	52	96	215	131	205	195	130	195	122	122
68	92	70	84	216	158	206	196	140	196	158	158
35	42	67	25	107	122	132	92	82	122	122	122

Table 2.4 continued from the previous page:

M1	M2	P1	P2	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8
65	70	98	29	151	183	193	123	113	183	183	183
44	61	96	67	168	160	193	148	111	160	160	160
82	89	40	52	181	142	201	161	134	161	142	142
70	54	41	23	123	131	154	99	105	131	126	131
61	96	85	57	193	166	187	173	131	173	166	166

Table 2.5 Condition: All the machines are empty at the start of a cycle

Robot Travel Time, δ = 12 secs

M1 M2 P1 P2	1A.1	1A.2	1A.3	1A.4	1A.5	1A.6	1A.7	1A.8	1A.9	1A.10
78 90 66 82	244	268	310	268	276	346	334	334	358	412
40 27 30 69	181	205	223	192	177	234	229	220	244	262
64 93 43 17	201	206	208	232	256	261	258	277	282	313
76 97 90 88	257	281	338	283	307	383	345	345	369	447
22 32 78 80	188	212	278	208	330	298	264	218	242	308
83 18 61 58	216	237	286	192	216	245	286	243	267	316
15 83 43 38	193	217	201	222	246	272	220	220	244	275
91 84 41 20	204	207	236	221	245	253	279	295	303	332
33 33 16 93	201	225	229	222	198	250	246	243	267	271
37 94 28 63	204	207	236	221	245	253	279	295	303	332
43 23 23 89	204	228	239	208	184	243	252	239	263	274
27 45 77 55	185	196	252	218	242	285	243	211	235	300
28 97 30 83	252	276	252	276	252	318	292	292	316	334
13 25 92 43	200	176	255	213	237	268	232	177	189	269
50 32 70 58	192	204	262	198	2222	268	262	224	248	306
37 53 60 92	217	241	273	241	233	313	273	266	290	338
22 93 53 20	201	209	193	242	266	274	219	235	243	284
62 26 90 71	224	229	307	212	236	295	307	243	267	345
40 34 85 61	197	197	270	215	239	288	270	219	243	316
68 76 66 30	206	202	248	238	262	280	258	264	282	336
30 25 29 38	146	170	187	159	174	200	188	177	201	218
89 56 73 25	234	210	271	225	249	262	271	265	278	339

Table 2.5 continued from the previous page:

M1 M2 P1 P2	1A.1	1A.2	1A.3	1A.4	1A.5	1A.6	1A.7	1A.8	1A.9	1A.10
29 24 30 67	175	199	217	187	174	229	216	204	228	246
81 52 76 51	229	228	292	224	248	287	292	268	292	356
79 20 79 20	230	206	262	206	230	230	262	230	227	394
36 26 36 26	144	158	182	158	182	196	182	182	196	220
63 97 63 97	266	290	307	290	280	365	341	341	365	416
58 73 58 73	218	242	273	242	242	312	288	288	312	358
51 67 86 85	224	248	306	249	273	346	306	287	311	385
55 67 60 86	225	249	285	249	247	321	292	292	316	364
79 36 33 69	220	244	265	201	189	246	268	268	292	313
82 97 43 56	225	249	265	249	260	304	319	319	343	374
98 58 48 24	218	218	254	202	226	238	264	276	288	324
55 73 86 97	242	266	322	266	279	364	322	309	333	407
32 95 41 22	203	213	189	232	256	266	233	247	257	286
84 99 54 80	251	275	302	275	273	341	347	347	371	413
33 89 15 23	197	208	184	221	224	235	229	242	253	256
47 92 54 96	260	284	281	284	266	350	319	319	343	385
88 63 84 20	244	220	276	243	267	275	276	271	279	351
82 56 64 44	218	222	274	216	240	272	274	266	290	342
43 67 83 40	198	203	250	246	270	298	250	234	258	329
84 31 41 74	230	254	283	201	197	254	283	273	297	326
50 79 52 96	244	268	282	268	248	332	306	306	330	370
68 92 70 84	248	272	306	272	282	354	328	328	352	410
35 42 67 25	175	163	212	205	229	242	211	197	210	265

Table 2.5 continued from the previous page:

M1 M2 P1 P2	1A.1	1A.2	1A.3	1A.4	1A.5	1A.6	1A.7	1A.8	1A.9	1A.10
65 70 98 29	235	211	276	264	288	305	276	255	272	358
44 61 96 67	212	224	291	253	277	332	291	256	280	364
82 89 40 52	217	237	258	237	249	289	307	307	331	259
70 54 41 23	183	189	218	191	215	226	231	244	255	284
61 96 85 57	225	249	287	277	301	346	298	298	322	395

Table 2.6 Condition: All the machines are loaded at the start of a cycle,
Robot Travel Time, δ = 12 secs

M1 M2 P1 P2	1B.1	1B.2	1B.3	1B.4	1B.5	1B.6	1B.7	1B.8	1B.9
78 90 66 82	186	174	162	144	174	150	166	150	150
40 27 30 69	148	144	144	144	144	144	153	144	144
64 93 43 17	172	151	148	144	177	153	148	144	153
76 97 90 88	184	198	160	144	181	157	171	174	157
22 32 78 80	144	186	144	144	144	144	164	162	144
83 18 61 58	191	169	167	144	144	144	167	145	144
15 83 43 38	144	151	144	144	167	144	144	144	144
91 84 41 20	199	149	175	144	168	144	175	144	151
33 33 16 93	146	144	153	144	144	144	177	153	153
37 94 28 63	145	144	144	144	178	154	147	144	154
43 23 23 89	151	144	149	144	144	144	173	149	149
27 45 77 55	144	185	144	144	144	144	144	161	144
28 97 30 83	144	144	144	144	181	157	167	144	157
13 25 92 43	144	200	152	144	144	152	144	176	152
50 32 70 58	158	178	144	144	144	144	144	154	144
37 53 60 92	145	168	152	144	144	144	176	152	152
22 93 53 20	144	161	144	144	177	153	144	144	153
62 26 90 71	170	198	150	144	144	150	155	174	150
40 34 85 61	157	193	145	144	144	145	145	169	145
68 76 66 30	176	174	152	144	160	144	152	150	144
30 25 29 38	144	144	144	144	144	144	144	144	144
89 56 73 25	197	181	173	144	149	144	173	157	149

Table 2.6 continued from the previous page:

M1 M2 P1 P2	1B.1	1B.2	1B.3	1B.4	1B.5	1B.6	1B.7	1B.8	1B.9
29 24 30 67	144	144	144	144	144	144	151	144	144
81 52 76 51	189	184	165	144	144	144	165	160	144
79 20 79 20	187	187	163	144	144	144	163	163	144
36 26 36 26	144	144	144	144	144	144	144	144	144
63 97 63 97	171	171	157	144	181	157	181	157	157
58 73 58 73	166	166	144	144	157	144	157	144	144
51 67 86 85	159	194	146	144	151	146	169	170	146
55 67 60 86	163	168	146	144	151	144	170	146	146
79 36 33 69	187	144	163	144	144	144	163	144	144
82 97 43 56	190	151	166	144	181	157	166	144	157
98 58 48 24	206	156	182	144	158	144	182	144	158
55 73 86 97	163	194	157	144	157	146	181	170	157
32 95 41 22	144	149	144	144	179	155	144	144	155
84 99 54 80	192	162	168	144	183	159	168	144	159
33 89 15 23	144	144	144	144	173	149	144	144	149
47 92 54 96	155	162	156	144	176	152	180	156	156
88 63 84 20	196	192	172	144	148	144	172	168	148
82 56 64 44	190	172	166	144	144	144	166	148	144
43 67 83 40	158	191	144	144	151	144	144	167	144
84 31 41 74	192	149	168	144	144	144	168	144	144
50 79 52 96	158	160	156	144	163	144	180	156	156
68 92 70 84	176	178	152	144	176	152	168	154	152
35 42 67 25	144	175	144	144	144	144	144	151	144

Table 2.6 continued from the previous page:

M1 M2 P1 P2	1B.1	1B.2	1B.3	1B.4	1B.5	1B.6	1B.7	1B.8	1B.9
65 70 98 29	173	206	158	144	154	158	149	182	158
44 61 96 67	152	204	156	144	145	156	151	180	156
82 89 40 52	190	148	166	144	173	149	166	144	149
70 54 41 23	178	149	154	144	144	144	154	144	144
61 96 85 57	169	193	145	144	180	156	145	169	156

Table 2.7 Condition: All the machines are loaded at the start of a cycle

Robot Travel Time, δ = 12 secs

M1 M2 P1 P2	1B.10	1B.11	1B.12	1B.13	1B.14	1B.15	1B.16	1B.17	1B.18
78 90 66 82	144	162	144	174	166	150	166	174	198
40 27 30 69	144	144	144	144	153	144	153	144	144
64 93 43 17	144	153	144	177	144	153	153	177	201
76 97 90 88	144	160	147	181	171	157	171	181	205
22 32 78 80	144	144	144	162	164	144	164	144	144
83 18 61 58	144	167	144	145	144	144	144	144	144
15 83 43 38	144	144	144	167	144	144	144	167	191
91 84 41 20	144	175	151	168	144	151	151	168	192
33 33 16 93	144	144	153	144	177	153	177	153	144
37 94 28 63	144	154	144	178	147	154	154	178	202
43 23 23 89	144	144	149	144	173	149	173	149	144
27 45 77 55	144	144	144	161	144	144	144	144	153
28 97 30 83	144	157	144	151	167	157	167	181	205
13 25 92 43	144	144	144	176	152	152	144	152	144
50 32 70 58	144	144	144	154	144	144	144	144	144
37 53 60 92	144	144	152	144	176	152	176	152	161
22 93 53 20	144	153	144	177	144	153	153	177	201
62 26 90 71	144	146	144	174	155	150	155	150	144
40 34 85 61	144	144	144	169	145	145	145	145	144
68 76 66 30	144	152	144	160	144	144	144	160	184
30 25 29 38	144	144	144	144	144	144	144	144	144
89 56 73 25	144	173	149	157	144	149	149	144	164

Table 2.7 continued from the previous page:

M1 M2 P1 P2	1B.10	1B.11	1B.12	1B.13	1B.14	1B.15	1B.16	1B.17	1B.18
29 24 30 67	**144**	**144**	**144**	**144**	151	**144**	151	**144**	**144**
81 52 76 51	**144**	165	**144**	160	**144**	**144**	**144**	**144**	160
79 20 79 20	**144**	163	**144**	163	**144**	**144**	**144**	**144**	**144**
36 26 36 26	**144**	**144**	**144**	**144**	**144**	**144**	**144**	**144**	**144**
63 97 63 97	**144**	157	157	181	181	157	181	181	205
58 73 58 73	**144**	**144**	**144**	157	157	**144**	157	157	181
51 67 86 85	**144**	**144**	145	170	169	146	169	151	175
55 67 60 86	**144**	**144**	146	151	170	146	170	151	175
79 36 33 69	**144**	163	**144**	**144**	153	**144**	153	**144**	**144**
82 97 43 56	**144**	166	**144**	**144**	**144**	157	157	181	205
98 58 48 24	**144**	182	158	158	**144**	158	158	**144**	166
55 73 86 97	**144**	**144**	157	157	181	157	181	157	181
32 95 41 22	**144**	155	**144**	**144**	**144**	155	155	179	203
84 99 54 80	**144**	168	**144**	**144**	164	159	164	183	207
33 89 15 23	**144**	149	**144**	**144**	**144**	149	149	173	197
47 92 54 96	**144**	152	156	176	180	156	180	176	200
88 63 84 20	**144**	172	148	168	**144**	148	148	147	171
82 56 64 44	**144**	166	**144**	148	**144**	**144**	**144**	**144**	164
43 67 83 40	**144**	**144**	**144**	167	**144**	**144**	**144**	151	175
84 31 41 74	**144**	168	**144**	**144**	158	**144**	158	**144**	**144**
50 79 52 96	**144**	**144**	156	163	180	156	180	163	187
68 92 70 84	**144**	152	**144**	176	168	152	168	176	200
35 42 67 25	**144**	**144**	**144**	151	**144**	**144**	**144**	**144**	150

Table 2:7 continued from the previous page:

M1 M2 P1 P2	1B.10	1B.11	1B.12	1B.13	1B.14	1B.15	1B.16	1B.17	1B.18
65 70 98 29	144	149	144	182	158	158	144	158	178
44 61 96 67	144	144	144	180	156	156	151	156	169
82 89 40 52	144	166	144	173	144	149	149	173	197
70 54 41 23	144	154	144	144	144	144	144	144	162
61 96 85 57	144	156	144	180	145	156	156	180	204

Table 2.8 Condition: Some machines are loaded at the start of a cycle,
Robot Travel Time, δ = 12 secs

M1 M2 P1 P2	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8
78 90 66 82	268	192	244	160	174	220	198	192
40 27 30 69	201	153	177	153	144	153	162	148
64 93 43 17	206	155	209	201	177	201	175	177
76 97 90 88	280	214	256	232	181	232	222	214
22 32 78 80	212	186	230	164	144	186	210	162
83 18 61 58	213	192	216	144	167	192	193	192
15 83 43 38	217	151	193	191	167	191	175	167
91 84 41 20	200	180	247	192	175	192	180	199
33 33 16 93	225	177	201	177	153	177	148	146
37 94 28 63	253	147	229	205	178	205	160	178
43 23 23 89	221	173	197	173	149	173	160	151
27 45 77 55	196	185	204	153	144	185	209	161
28 97 30 83	276	169	252	228	181	228	162	181
13 25 92 43	175	200	207	152	144	200	224	176
50 32 70 58	190	178	200	144	144	178	202	168
37 53 60 92	241	176	224	193	152	193	196	145
22 93 53 20	209	161	201	201	177	201	185	177
62 26 90 71	205	200	233	155	146	200	222	200
40 34 85 61	193	193	218	145	144	193	217	182
68 76 66 30	202	182	216	184	160	184	198	182
30 25 29 38	170	144	146	144	144	144	161	144
89 56 73 25	186	210	234	164	173	210	210	210

Table 2.8 continued from the previous page:

M1 M2 P1 P2	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8
29 24 30 67	199	151	175	151	144	151	162	144
81 52 76 51	204	205	229	160	165	205	208	205
79 20 79 20	171	206	230	144	163	206	211	166
36 26 36 26	158	144	144	144	144	144	168	144
63 97 63 97	290	181	266	242	181	242	195	181
58 73 58 73	242	166	218	194	157	194	190	166
51 67 86 85	248	194	243	200	151	200	218	185
55 67 60 86	249	170	225	201	151	200	192	163
79 36 33 69	220	163	187	153	169	163	165	187
82 97 43 56	249	173	251	205	181	205	175	190
98 58 48 24	294	194	228	166	182	194	194	206
55 73 86 97	266	194	255	218	157	218	218	189
32 95 41 22	213	149	203	203	179	203	173	179
84 99 54 80	275	186	255	227	183	227	186	192
33 89 15 23	208	144	197	197	173	197	147	173
47 92 54 96	284	180	260	236	176	236	186	176
88 63 84 20	180	220	244	171	172	220	220	220
82 56 64 44	198	194	218	164	166	194	196	194
43 67 83 40	203	191	198	175	151	191	215	174
84 31 41 74	230	173	197	158	168	173	173	192
50 79 52 96	261	180	247	223	163	223	184	163
68 92 70 84	262	186	248	224	176	224	202	186
35 42 67 25	163	175	175	150	144	175	199	151

Table 2.8 continued from the previous page:

M1 M2 P1 P2	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8
65 70 98 29	195	211	235	178	154	211	230	211
44 61 96 67	224	204	235	176	145	204	228	188
82 89 40 52	237	170	243	197	173	197	172	190
70 54 41 23	173	159	196	162	154	162	173	178
61 96 85 57	249	194	229	204	144	204	217	194

Table 2.9 Condition: All the machines are empty at the start of a cycle,
Robot Travel Time, δ = 20 secs

M1 M2 P1 P2	1A.1	1A.2	1A.3	1A.4	1A.5	1A.6	1A.7	1A.8	1A.9	1A.10
78 90 66 82	292	332	366	332	356	418	390	390	430	476
40 27 30 69	249	289	299	256	257	306	309	276	316	326
64 93 43 17	273	270	264	313	336	333	314	357	354	377
76 97 90 88	305	345	394	347	387	455	401	401	441	511
22 32 78 80	260	300	358	272	310	370	320	274	314	372
83 18 61 58	264	301	342	239	279	317	342	301	339	380
15 83 43 38	263	281	281	303	326	344	276	298	316	339
91 84 41 20	271	271	292	304	325	325	335	375	375	396
33 33 16 93	273	313	309	286	249	322	326	299	339	335
37 94 28 63	277	317	291	317	322	365	334	334	374	382
43 23 23 89	269	309	312	272	246	315	332	295	335	338
27 45 77 55	257	275	332	282	322	357	299	272	307	364
28 97 30 83	300	340	313	340	327	390	348	348	388	398
13 25 92 43	272	263	335	277	317	340	288	238	261	333
50 32 70 58	250	278	328	262	302	340	318	282	320	370
37 53 60 92	273	312	352	305	313	385	329	322	362	402
22 93 53 20	273	273	273	313	346	346	275	315	315	348
62 26 90 71	272	293	363	276	316	367	363	299	339	409
40 34 85 61	265	281	346	279	319	360	326	275	315	380
68 76 66 30	256	266	304	302	342	352	314	344	354	400
30 25 29 38	240	258	267	245	245	272	268	255	273	282
89 56 73 25	282	274	327	289	329	334	327	345	350	403

Table 2.9 continued from the previous page:

M1	M2	P1	P2	1A.1	1A.2	1A.3	1A.4	1A.5	1A.6	1A.7	1A.8	1A.9	1A.10
29	24	30	67	247	287	297	251	254	301	296	260	300	310
81	52	76	51	277	292	348	288	328	359	348	333	364	420
79	20	79	20	278	259	318	259	299	299	318	299	299	358
36	26	36	26	240	246	262	246	262	268	262	262	268	284
63	97	63	97	314	354	363	354	360	437	397	397	437	480
58	73	58	73	266	306	331	306	331	384	344	344	384	422
51	67	86	85	272	312	371	313	353	418	362	343	383	449
55	67	60	86	273	313	346	313	327	393	348	348	388	428
79	36	33	69	268	308	321	265	269	318	348	324	364	377
82	97	43	56	277	313	321	317	340	376	375	379	415	438
98	58	48	24	278	282	310	278	306	310	322	356	360	388
55	73	86	97	290	330	383	330	359	436	378	365	405	471
32	95	41	22	275	277	263	315	336	338	289	327	329	350
84	99	54	80	299	339	358	339	353	413	403	403	443	477
33	89	15	23	269	272	238	309	304	307	285	322	325	320
47	92	54	96	308	348	350	348	346	422	375	375	415	449
88	63	84	20	292	268	332	307	347	347	332	351	351	415
82	56	64	44	266	286	330	380	320	344	330	338	362	406
43	67	83	40	263	267	323	310	350	370	306	310	330	393
84	31	41	74	278	318	339	265	272	326	358	329	369	390
50	79	52	96	295	335	348	335	331	407	365	365	405	437
68	92	70	84	296	336	362	336	362	426	384	384	424	474
35	42	67	25	247	245	292	269	309	314	267	277	282	329

Table 2.9 continued from the previous page:

M1	M2	P1	P2	1A.1	1A.2	1A.3	1A.4	1A.5	1A.6	1A.7	1A.8	1A.9	1A.10
65	70	98	29	283	259	332	328	368	377	332	335	344	422
44	61	96	67	276	288	363	317	357	404	347	312	352	428
82	89	40	52	269	301	314	309	329	361	363	371	403	423
70	54	41	23	250	253	274	274	295	298	293	324	327	348
61	96	85	57	276	313	343	341	381	418	354	357	394	459

Table 2.10 Condition: All the machines are loaded at the start of a cycle,
Robot Travel Time, $\delta = 20$ secs

M1	M2	P1	P2	1B.1	1B.2	1B.3	1B.4	1B.5	1B.6	1B.7	1B.8	1B.9
78	90	66	82	258	246	240	240	240	240	240	240	240
40	27	30	69	240	240	240	240	240	240	240	240	240
64	93	43	17	244	244	240	240	240	240	240	240	240
76	97	90	88	256	256	240	240	240	240	240	240	240
22	32	78	80	240	240	240	240	240	240	240	240	240
83	18	61	58	263	241	240	240	240	240	240	240	240
15	83	43	38	240	240	240	240	240	240	240	240	240
91	84	41	20	271	240	240	240	240	240	240	240	240
33	33	16	93	240	240	240	240	240	240	240	240	240
37	94	28	63	240	240	240	240	240	240	240	240	240
43	23	23	89	240	240	240	240	240	240	240	240	240
27	45	77	55	240	257	240	240	240	240	240	240	240
28	97	30	83	240	240	240	240	240	240	240	240	240
13	25	92	43	240	272	240	240	240	240	240	240	240
50	32	70	58	240	250	240	240	240	240	240	240	240
37	53	60	92	240	240	240	240	240	240	240	240	240
22	93	53	20	242	240	240	240	240	240	240	240	240
62	26	90	71	240	270	240	240	240	240	240	240	240
40	34	85	61	240	265	240	240	240	240	240	240	240
68	76	66	30	248	246	240	240	240	240	240	240	240
30	25	29	38	240	240	240	240	240	240	240	240	240
89	56	73	25	269	253	240	240	240	240	240	240	240

Table 2.10 continued from the previous page:

M1 M2 P1 P2	1B.1	1B.2	1B.3	1B.4	1B.5	1B.6	1B.7	1B.8	1B.9
29 24 30 67	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
81 52 76 51	261	256	**240**	**240**	**240**	**240**	**240**	**240**	**240**
79 20 79 20	259	259	**240**	**240**	**240**	**240**	**240**	**240**	**240**
36 26 36 26	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
63 97 63 97	243	243	**240**	**240**	**240**	**240**	**240**	**240**	**240**
58 73 58 73	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
51 67 86 85	**240**	266	**240**	**240**	**240**	**240**	**240**	**240**	**240**
55 67 60 86	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
79 36 33 69	259	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
82 97 43 56	262	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
98 58 48 24	278	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
55 73 86 97	**240**	266	**240**	**240**	**240**	**240**	**240**	**240**	**240**
32 95 41 22	**240**	266	**240**	**240**	**240**	**240**	**240**	**240**	**240**
84 99 54 80	264	266	**240**	**240**	**240**	**240**	**240**	**240**	**240**
33 89 15 23	**240**	266	**240**	**240**	**240**	**240**	**240**	**240**	**240**
47 92 54 96	**240**	266	**240**	**240**	**240**	**240**	**240**	**240**	**240**
88 63 84 20	268	264	**240**	**240**	**240**	**240**	**240**	**240**	**240**
82 56 64 44	262	244	**240**	**240**	**240**	**240**	**240**	**240**	**240**
43 67 83 40	**240**	263	**240**	**240**	**240**	**240**	**240**	**240**	**240**
84 31 41 74	264	263	**240**	**240**	**240**	**240**	**240**	**240**	**240**
50 79 52 96	**240**	263	**240**	**240**	**240**	**240**	**240**	**240**	**240**
68 92 70 84	248	250	**240**	**240**	**240**	**240**	**240**	**240**	**240**
35 42 67 25	**240**	247	**240**	**240**	**240**	**240**	**240**	**240**	**240**

Table 2.10 continued from the previous page:

M1 M2 P1 P2	1B.1	1B.2	1B.3	1B.4	1B.5	1B.6	1B.7	1B.8	1B.9
65 70 98 29	245	278	**240**	**240**	**240**	**240**	**240**	**240**	**240**
44 61 96 67	**240**	276	**240**	**240**	**240**	**240**	**240**	**240**	**240**
82 89 40 52	262	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
70 54 41 23	250	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
61 96 85 57	241	265	**240**	**240**	**240**	**240**	**240**	**240**	**240**

Table 2.11 Condition : All the machines are loaded at the start of the cycle,
Robot Travel Time, δ = 20 secs

M1 M2 P1 P2	1B.10	1B.11	1B.12	1B.13	1B.14	1B.15	1B.16	1B.17	1B.18
78 90 66 82	240	240	240	240	240	240	240	240	270
40 27 30 69	240	240	240	240	240	240	240	240	240
64 93 43 17	240	240	240	240	240	240	240	240	273
76 97 90 88	240	240	240	240	240	240	240	240	277
22 32 78 80	240	240	240	240	240	240	240	240	240
83 18 61 58	240	240	240	240	240	240	240	240	240
15 83 43 38	240	240	240	240	240	240	240	240	263
91 84 41 20	240	240	240	240	240	240	240	240	264
33 33 16 93	240	240	240	240	240	240	240	240	240
37 94 28 63	240	240	240	240	240	240	240	240	274
43 23 23 89	240	240	240	240	240	240	240	240	240
27 45 77 55	240	240	240	240	240	240	240	240	240
28 97 30 83	240	240	240	240	240	240	240	240	277
13 25 92 43	240	240	240	240	240	240	240	240	240
50 32 70 58	240	240	240	240	240	240	240	240	240
37 53 60 92	240	240	240	240	240	240	240	240	240
22 93 53 20	240	240	240	240	240	240	240	240	273
62 26 90 71	240	240	240	240	240	240	240	240	240
40 34 85 61	240	240	240	240	240	240	240	240	240
68 76 66 30	240	240	240	240	240	240	240	240	256
30 25 29 38	240	240	240	240	240	240	240	240	240
89 56 73 25	240	240	240	240	240	240	240	240	240

Table 2.11 continued from the previous page:

M1	M2	P1	P2	1B.10	1B.11	1B.12	1B.13	1B.14	1B.15	1B.16	1B.17	1B.18
29	24	30	67	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
81	52	76	51	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
79	20	79	20	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
36	26	36	26	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
63	97	63	97	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
58	73	58	73	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
51	67	86	85	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	247
55	67	60	86	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	247
79	36	33	69	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
82	97	43	56	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	277
98	58	48	24	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
55	73	86	97	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	253
32	95	41	22	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	275
84	99	54	80	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	279
33	89	15	23	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	269
47	92	54	96	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	272
88	63	84	20	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	243
82	56	64	44	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
43	67	83	40	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	247
84	31	41	74	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**
50	79	52	96	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	259
68	92	70	84	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	272
35	42	67	25	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**	**240**

Table 2.11 continued from the previous page:

M1 M2 P1 P2	1B.10	1B.11	1B.12	1B.13	1B.14	1B.15	1B.16	1B.17	1B.18
65 70 98 29	240	240	240	240	240	240	240	240	250
44 61 96 67	240	240	240	240	240	240	240	240	241
82 89 40 52	240	240	240	240	240	240	240	240	269
70 54 41 23	240	240	240	240	240	240	240	240	240
61 96 85 57	240	240	240	240	240	240	240	240	276

Table 2.12 Condition: Some machines are loaded at the start of a cycle,
Robot Travel Times, $\delta = 20$ secs

M1	M2	P1	P2	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8
78	90	66	82	332	246	296	270	240	270	286	258
40	27	30	69	289	240	249	240	240	240	250	240
64	93	43	17	270	240	277	273	240	273	263	244
76	97	90	88	344	270	304	277	240	277	310	256
22	32	78	80	270	258	278	240	240	258	298	240
83	18	61	58	278	241	264	240	240	241	281	263
15	83	43	38	281	240	263	263	240	263	263	240
91	84	41	20	264	240	295	264	240	264	261	271
33	33	16	93	313	240	273	240	240	240	236	240
37	94	28	63	317	240	277	274	240	274	258	240
43	23	23	89	309	240	269	240	240	240	248	240
27	45	77	55	225	257	257	240	240	257	297	240
28	97	30	83	340	240	300	277	240	277	250	240
13	25	92	43	263	272	272	240	240	272	312	240
50	32	70	58	278	250	250	240	240	250	290	240
37	53	60	92	312	240	272	240	240	240	280	240
22	93	53	20	273	240	273	273	240	273	273	240
62	26	90	71	291	270	281	240	240	270	310	242
40	34	85	61	281	265	266	240	240	265	300	240
68	76	66	30	266	246	264	256	240	256	286	248
30	25	29	38	258	240	240	240	240	240	249	240
89	56	73	25	245	253	282	240	240	253	393	269

Table 2.12 continued from the previous page:

M1	M2	P1	P2	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8
29	24	30	67	287	**240**	247	**240**	**240**	**240**	250	**240**
81	52	76	51	271	256	277	**240**	**240**	256	296	261
79	20	79	20	**240**	259	278	**240**	**240**	259	319	259
36	26	36	26	226	**240**	**240**	**240**	**240**	**240**	276	**240**
63	97	63	97	354	243	314	277	**240**	277	303	243
58	73	58	73	306	**240**	266	253	**240**	253	298	**240**
51	67	86	85	305	266	291	247	**240**	266	306	**240**
55	67	60	86	313	**240**	273	247	**240**	247	280	**240**
79	36	33	69	289	**240**	259	**240**	**240**	**240**	253	259
82	97	43	56	326	**240**	299	277	**240**	277	263	262
98	58	48	24	244	**240**	278	**240**	**240**	**240**	268	278
55	73	86	97	330	266	303	253	**240**	266	306	**240**
32	95	41	22	277	**240**	275	275	**240**	275	261	**240**
84	99	54	80	339	**240**	303	279	**240**	279	274	264
33	89	15	23	272	**240**	269	269	**240**	269	235	**240**
47	92	54	96	348	**240**	308	272	**240**	272	276	**240**
88	63	84	20	243	264	292	243	**240**	264	304	268
82	56	64	44	264	244	266	**240**	**240**	244	284	262
43	67	83	40	271	263	263	247	**240**	263	303	**240**
84	31	41	74	294	**240**	264	**240**	**240**	**240**	261	264
50	79	52	96	335	**240**	295	259	**240**	259	272	**240**
68	92	70	84	336	250	296	272	**240**	272	290	248
35	42	67	25	245	247	247	**240**	**240**	247	287	**240**

Table 2.12 continued from the previous page :

M1 M2 P1 P2	1C.1	1C.2	1C.3	1C.4	1C.5	1C.6	1C.7	1C.8
65 70 98 29	259	278	283	250	**240**	278	318	245
44 61 96 67	288	276	283	241	**240**	276	306	**240**
82 89 40 52	301	**240**	291	269	**240**	269	260	262
70 54 41 23	243	**240**	250	**240**	**240**	**240**	261	250
61 96 85 57	313	265	277.	276	**240**	276	305	241

APPENDIX 3

The tables represent the cycle time generated by the dynamic scheduling software, considering no machine breakdowns. The "bold" values represent the lowest cycle time. Where the number of parts produced in each cycle is more than 2, the Average Cycle time is calculated to represent the time taken to produce 2 parts.

Table 3.1. Robot Travel time : 5 secs

M1	M2	P1	P2	Cycle Time				Average Cycle time				Parts per cycle			
				1b	2b	3b	4b	1b	2b	3b	4b	1b	2b	3b	4b
78	90	66	82	798	798	118	118	**107**	**107**	118	118	15	15	2	2
40	27	30	69	200	89	190	190	80	89	**76**	**76**	5	2	5	5
64	93	43	17	350	350	454	454	**100**	**100**	101	101	7	7	9	9
76	97	90	88	117	117	117	117	**117**	**117**	**117**	**117**	2	2	2	2
22	32	78	80	100	100	100	100	**100**	**100**	**100**	**100**	2	2	2	2
83	18	61	58	234	334	324	325	94	96	**93**	**93**	5	7	7	7
15	83	43	38	342	126	206	206	86	84	**83**	**83**	8	3	5	5
91	84	41	20	111	111	121	121	**111**	**111**	121	121	2	2	2	2
33	33	16	93	113	113	119	119	**76**	**76**	79	79	3	3	3	3
37	94	28	63	124	124	131	131	**83**	**83**	88	88	3	3	3	3
43	23	23	89	126	126	126	126	**84**	**84**	**84**	**84**	3	3	3	4
27	45	77	55	97	97	97	97	**97**	**97**	**97**	**97**	2	2	2	2
28	97	30	83	117	117	123	123	**78**	**78**	82	82	3	3	3	3
13	25	92	43	126	126	126	126	**84**	**84**	**84**	**84**	3	3	3	3
50	32	70	58	90	90	90	90	**90**	**90**	**90**	**90**	2	2	2	2
37	53	60	92	252	252	252	252	**101**	**101**	**101**	**101**	5	5	5	5
22	93	53	20	236	236	226	226	**95**	**95**	91	91	6	6	5	5
62	26	90	71	110	110	110	110	**110**	**110**	**110**	**110**	2	2	2	2

Table 3.1. continued from the previous page

M1	M2	P1	P2	Cycle Time				Average Cycle time				Parts per cycle			
				1b	2b	3b	4b	1b	2b	3b	4b	1b	2b	3b	4b
40	34	85	61	105	105	105	105	105	105	105	105	2	2	2	2
68	76	66	30	96	96	106	106	96	96	106	106	2	2	2	2
30	25	29	38	60	60	68	68	60	60	68	68	2	2	2	2
89	56	73	25	238	238	331	331	96	96	95	95	5	5	7	7
29	24	30	67	110	110	110	110	74	74	74	74	3	3	3	3
81	52	76	51	227	227	329	323	91	91	94	93	5	5	7	7
79	20	79	20	100	100	100	100	67	67	67	67	3	3	3	3
36	26	36	26	60	60	66	66	60	60	66	66	2	2	2	2
63	97	63	97	260	260	260	260	104	104	104	104	5	5	5	5
58	73	58	73	93	93	93	93	93	93	93	93	2	2	2	2
51	67	86	85	106	106	106	106	106	106	106	106	2	2	2	2
55	67	60	86	106	106	106	106	106	106	106	106	2	2	2	2
79	36	33	69	99	99	99	99	99	99	99	99	2	2	2	2
82	97	43	56	530	530	122	122	112	112	122	122	11	11	2	2
98	58	48	24	236	236	246	246	95	95	99	99	5	5	5	5
55	73	86	97	117	117	117	117	117	117	117	117	2	2	2	2
32	95	41	22	122	122	133	133	82	82	89	89	3	3	3	3
84	99	54	80	738	738	124	124	114	114	124	124	13	13	2	2
33	89	15	23	109	109	119	119	73	73	80	80	3	3	3	3
47	92	54	96	250	250	250	250	100	100	100	100	5	5	5	5
88	63	84	20	345	345	345	345	99	99	99	99	7	7	7	7
82	56	64	44	648	348	320	320	92	92	92	92	14	14	7	7
43	67	83	40	103	103	103	103	103	103	103	103	2	2	2	2

Table 3.1 continued from the previous page

M1	M2	P1	P2	Cycle Time 1b	2b	3b	4b	Average Cycle time 1b	2b	3b	4b	Parts per cycle 1b	2b	3b	4b
84	31	41	74	104	104	104	104	104	104	104	104	2	2	2	2
50	79	52	96	144	144	144	144	96	96	96	96	3	3	3	3
68	92	70	84	466	466	114	474	106	106	106	106	9	9	2	9
35	42	67	25	286	87	189	189	82	87	76	76	7	2	5	5
65	70	98	29	435	118	367	367	97	118	105	105	9	2	7	7
44	61	96	67	116	116	116	116	116	116	116	116	2	2	2	2
82	89	40	52	109	109	119	119	109	109	119	119	2	2	2	2
70	54	41	23	381	381	94	94	85	85	94	94	9	9	2	2
61	96	85	57	116	116	116	116	116	116	116	116	2	2	2	2

Table 3.2. Robot Travel Time, $\delta = 12$ secs

M1	M2	P1	P2	Cycle Time				Average Cycle time				Parts per cycle			
				1b	2b	3b	4b	1b	2b	3b	4b	1b	2b	3b	4b
78	90	66	82	144	144	162	162	144	144	162	162	2	2	2	2
40	27	30	69	144	144	144	144	144	144	144	144	2	2	2	2
64	93	43	17	144	144	160	160	144	144	160	160	2	2	2	2
76	97	90	88	145	145	169	169	145	145	169	169	2	2	2	2
22	32	78	80	144	144	144	144	144	144	144	144	2	2	2	2
83	18	61	58	144	144	144	144	144	144	144	144	2	2	2	2
15	83	43	38	144	144	144	144	144	144	144	144	2	2	2	2
91	84	41	20	144	144	163	163	144	144	163	163	2	2	2	2
33	33	16	93	330	330	369	144	132	132	123	144	5	5	6	2
37	94	28	63	144	144	144	144	144	144	144	144	2	2	2	2
43	23	23	89	161	144	144	144	161	144	144	144	2	2	2	2
27	45	77	55	144	144	144	144	144	144	144	144	2	2	2	2
28	97	30	83	216	216	364	360	144	144	146	144	3	3	5	5
13	25	92	43	144	144	144	144	144	144	144	144	2	2	2	2
50	32	70	58	144	144	144	144	144	144	144	144	2	2	2	2
37	53	60	92	144	144	144	156	144	144	144	156	2	2	2	2
22	93	53	20	144	144	144	144	144	144	144	144	2	2	2	2
62	26	90	71	144	144	144	144	144	144	144	144	2	2	2	2
40	34	85	61	144	144	144	144	144	144	144	144	2	2	2	2
68	76	66	30	144	144	148	148	144	144	148	148	2	2	2	2
30	25	29	38	144	144	144	144	144	144	144	144	2	2	2	2
89	56	73	25	144	144	152	152	144	144	152	152	2	2	2	2
29	24	30	67	144	144	144	144	144	144	144	144	2	2	2	2

Table 3.2 Continued from the previous page,

M1 M2 P1 P2				Cycle Time				Average Cycle time				Parts per cycle			
				1b	2b	3b	4b	1b	2b	3b	4b	1b	2b	3b	4b
81	52	76	51	144	144	148	148	144	144	148	148	2	2	2	2
79	20	79	20	144	144	144	144	144	144	144	144	2	2	2	2
36	26	36	26	144	144	144	144	144	144	144	144	2	2	2	2
63	97	63	97	145	145	169	184	145	145	169	92	2	2	2	4
58	73	58	73	144	144	145	145	144	144	145	145	2	2	2	2
51	67	86	85	144	144	158	158	144	144	158	158	2	2	2	2
55	67	60	86	144	144	158	156	144	144	158	156	2	2	2	2
79	36	33	69	144	144	144	144	144	144	144	144	2	2	2	2
82	97	43	56	152	152	169	169	152	152	169	169	2	2	2	2
98	58	48	24	362	362	154	154	145	145	154	154	5	5	2	2
55	73	86	97	145	145	145	145	145	145	145	145	2	2	2	2
32	95	41	22	144	144	144	144	144	144	144	144	2	2	2	2
84	99	54	80	147	147	171	171	147	147	171	171	2	2	2	2
33	89	15	23	144	144	144	144	144	144	144	144	2	2	2	2
47	92	54	96	144	144	144	144	144	144	144	144	2	2	2	2
88	63	84	20	144	144	159	159	144	144	159	159	2	2	2	2
82	56	64	44	144	144	152	152	144	144	152	152	2	2	2	2
43	67	83	40	144	144	155	144	144	144	155	144	2	2	2	2
84	31	41	74	144	144	144	144	144	144	144	144	2	2	2	2
50	79	52	96	144	144	168	316	144	144	168	158	2	2	2	4
68	92	70	84	144	144	164	164	144	144	164	164	2	2	2	2
35	42	67	25	144	144	144	144	144	144	144	144	2	2	2	2
65	70	98	29	146	146	146	146	146	146	146	146	2	2	2	2

Table 3.2 continued from the previous page:

M1	M2	P1	P2	Cycle Time 1b	2b	3b	4b	Average Cycle time 1b	2b	3b	4b	Parts per cycle 1b	2b	3b	4b
44	61	96	67	144	144	144	144	144	144	144	144	2	2	2	2
82	89	40	52	148	148	161	161	148	148	161	161	2	2	2	2
70	54	41	23	144	144	144	144	144	144	144	144	2	2	2	2
61	96	85	57	144	144	157	157	144	144	157	157	2	2	2	2

Table 3.3. Robot Travel Time, $\delta = 20$ secs

M1	M2	P1	P2	Cycle Time 1b	2b	3b	4b	Average Cycle time 1b	2b	3b	4b	Parts per cycle 1b	2b	3b	4b
78	90	66	82	240	240	240	240	240	240	240	240	2	2	2	2
40	27	30	69	240	240	240	240	240	240	240	240	2	2	2	2
64	93	43	17	240	240	240	240	240	240	240	240	2	2	2	2
76	97	90	88	240	240	240	240	240	240	240	240	2	2	2	2
22	32	78	80	240	240	240	240	240	240	240	240	2	2	2	2
83	18	61	58	240	240	240	240	240	240	240	240	2	2	2	2
15	83	43	38	240	240	240	240	240	240	240	240	2	2	2	2
91	84	41	20	240	240	240	240	240	240	240	240	2	2	2	2
33	33	16	93	240	240	240	240	240	240	240	240	2	2	2	2
37	94	28	63	240	240	240	240	240	240	240	240	2	2	2	2
43	23	23	89	240	240	240	240	240	240	240	240	2	2	2	2
27	45	77	55	240	240	240	240	240	240	240	240	2	2	2	2
28	97	30	83	240	240	240	240	240	240	240	240	2	2	2	2
13	25	92	43	240	240	240	240	240	240	240	240	2	2	2	2
50	32	70	58	240	240	240	240	240	240	240	240	2	2	2	2
37	53	60	92	240	240	240	240	240	240	240	240	2	2	2	2
22	93	53	20	240	240	240	240	240	240	240	240	2	2	2	2
62	26	90	71	240	240	240	240	240	240	240	240	2	2	2	2
40	34	85	61	240	240	240	240	240	240	240	240	2	2	2	2
68	76	66	30	240	240	240	240	240	240	240	240	2	2	2	2
30	25	29	38	240	240	240	240	240	240	240	240	2	2	2	2
89	56	73	25	240	240	240	240	240	240	240	240	2	2	2	2
29	24	30	67	240	240	240	240	240	240	240	240	2	2	2	2

Table 3.3 Continued from the previous page,

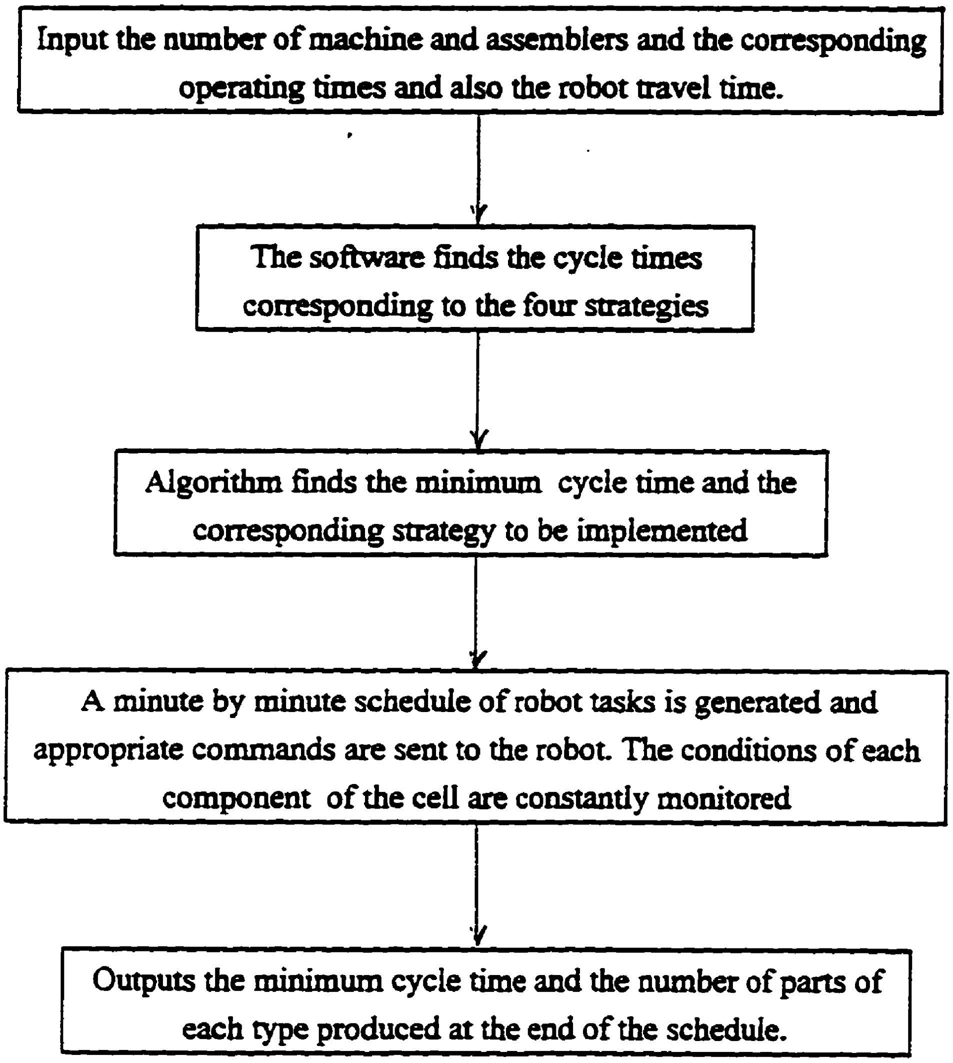
M1	M2	P1	P2	Cycle Time 1b	Cycle Time 2b	Cycle Time 3b	Cycle Time 4b	Average Cycle time 1b	Average Cycle time 2b	Average Cycle time 3b	Average Cycle time 4b	Parts per cycle 1b	Parts per cycle 2b	Parts per cycle 3b	Parts per cycle 4b
81	52	76	51	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
79	20	79	20	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
36	26	36	26	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
63	97	63	97	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
58	73	58	73	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
51	67	86	85	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
55	67	60	86	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
79	36	33	69	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
82	97	43	56	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
98	58	48	24	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
55	73	86	97	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
32	95	41	22	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
84	99	54	80	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
33	89	15	23	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
47	92	54	96	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
88	63	84	20	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
82	56	64	44	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
43	67	83	40	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
84	31	41	74	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
50	79	52	96	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
68	92	70	84	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
35	42	67	25	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2
65	70	98	29	240	240	240	240	**240**	**240**	**240**	**240**	2	2	2	2

Table 3.3 continued from the previous page:

M1	M2	P1	P2	Cycle Time 1b	2b	3b	4b	Average Cycle time 1b	2b	3b	4b	Parts per cycle 1b	2b	3b	4b
44	61	96	67	240	240	240	240	240	240	240	240	2	2	2	2
82	89	40	52	240	240	240	240	240	240	240	240	2	2	2	2
70	54	41	23	240	240	240	240	240	240	240	240	2	2	2	2
61	96	85	57	240	240	240	240	240	240	240	240	2	2	2	2

APPENDIX 4

The general structure of the software is :



Each strategy has been coded as a subroutine according to the rules selected. The general structure of each is :