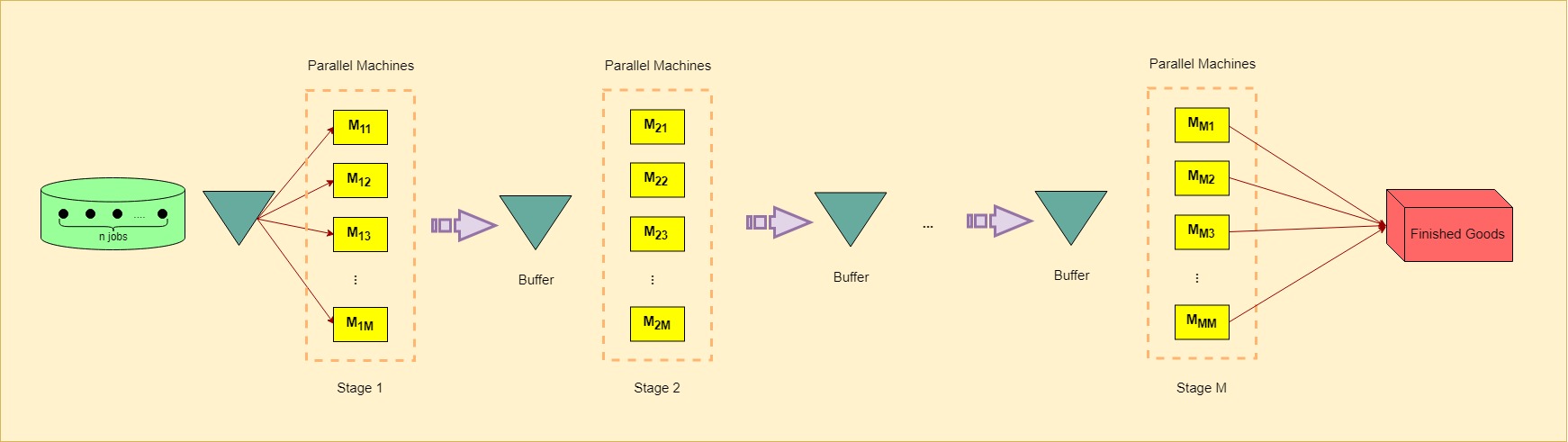
*C-lens Production Flowchart*



*Schematic view of Hybrid Flow shops (HFS)*

**

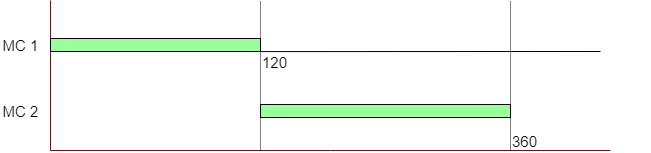
A **hybrid flow shops (HFS)**, alternatively called a flow shop with multiple processors, is an extended system of the ordinary flow shop. That can be defined as:

1. The set of M processing stages m is at least 2.
2. In each stage has set of of identical parallel machines where and
3. All jobs are processed following the same production flow: stage 1, stage 2,. . ., stage m. A job might skip any number of stages provided it is processed in at least one of them.
4. At each stage , each job is to be processed by exactly one of the available parallel machines. As a result, a job is made up of m different tasks, one per stage.
5. The processing time of any job at any stage is a positive integer denoted by
6. Jobs are processed without interruptions.

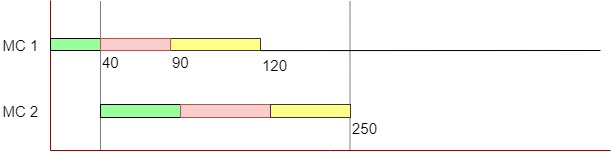
The most commonly studied optimization criterion is the minimization of the makespan or defined as , where is the time at which job is finished at stage *.* is commonly referred to as the completion time of job . The HFS with makespan criterion can be represented as

**Lot streaming** is a technique splitting a lot into *sublots* (also termed *transfer lots*), and processing different *sublots* simultaneously over different machines instead of transferring the entire lot of its item to process in one machine, albeit still maintaining their movement over the machines in accordance with their flow shop configuration.

The advantage of lot streaming is reduction in the makespan value, however, setup and/or transfer times are encountered during the handling of individual sublots.



Processing of a lot without lot streaming



Processing of a lot with lot streaming

{No. of machines}{machine configuration}/{no. of lots}/{sublot type}/{idling}/{sublot sizes}/{objective function}/{special features}.

Assumption:

**Mathematical formulation**

**\* Problem description and assumptions**

Consider a flowshop consisting of several consecutive production stages to process several jobs. Each stage has a known number of unrelated parallel machines. The batch of each job is to be split into several unequal consistent sublots. The sublots are to be processed in the order of the stages, and sublots of certain products may skip some stages. At a given stage, a sublot of a job can be assigned to one of the parallel machines eligible to process that particular job. For each job, there is a sequence-dependent setup time on each eligible machine, and this setup may be anticipatory or non-anticipatory on different stages. Each machine can process at most one sublot at a time. Sublots of different products can be interleaved. The problem is to determine the size of each sublot of each job and the assignment and processing sequence of these sublots on each machine in each stage. The objective function is to minimize the completion time of the last sublot to be processed in the system.

**\* Notations**

In order to present a mathematical model for the problem described above, we define several notations. These notations are explained in the following:

|  |  |
| --- | --- |
| Parameters | |
|  | Number of stages where stages are indexed by |
|  | Number of machines in stage where machined are indexed by |
|  | Number of jobs (products) where jobs are indexed by |
|  | Maximum number of sublots of job where sublots are indexed by |
|  | A set of pairs of stages for job constrained by precedence relations, i.e., the processing of job in stage is followed by its processing in stage |
|  | Processing time for one unit of job on machine in stage |
|  | Batch size of job |
|  | Maximum number of production runs of machine in stage where production runs are  indexed by |
|  | Setup time on machine in stage for processing job following the processing of job on this machine; if , the setup may be called minor setup |
|  | A binary data equal to 1 if setup of job in stage is attached (non-anticipatory), or 0 if this setup is detached setup (anticipatory) |
|  | A binary data equal to 1 if job n needs processing in stage , otherwise 0 |
|  | A binary data equal to 1 if job can be processed on machine in stage , otherwise 0; |
|  | The release date of machine in stage |
|  | Large positive number |
|  |  |
|  |  |
| Variables | |
|  | Completion time of the sublot of job from stage |
|  | Completion time of the run of machine in stage |
|  | Size of the sublot of job |
|  | Makespan of the schedule. It is the maximum of the completion times of all the sublots or, in other words, it is the completions time of the last operation of the last sublot processed in the system |
|  |  |
|  |  |
| Binary Variables | |
|  | Binary variable which takes the value if the run on machine in stage is for the sublot of job ; otherwise |
|  | Binary variable which takes the value if the run on machine in stage is for job ;  otherwise |
|  | A binary variable which takes the value if the potential run of machine in stage has been assigned a sublot to process; otherwise |
|  | A binary variable that takes if sublot is non-zero ; otherwise |
|  |  |
|  |  |

**\* MILP model.**

Using the notation given above, the objective function and the constraints of the proposed MILP mathematical model for the hybrid flexible flowshop lot streaming problem are presented below.

1. Objective:

2. Constraints:

1. The completion time of the sublot of job in stage is equal to the completion time of the run of machine in stage if this production run is assigned to that particular sublot.

|  |  |
| --- | --- |
|  |  |
|  |  |

1. The starting time cannot be less than the release date of the machine

|  |  |
| --- | --- |
|  |  |

1. the requirement that the setup of any production run r > 1 of a given machine cannot be started before the completion time of run r − 1 of that machine

|  |  |
| --- | --- |
|  |  |

1. Any pair of stages , the setup or the actual processing of the first run on machine in stage may not be started before the completion time of run of machine in stage , depending on whether the setup of product type n in stage is non-anticipatory or anticipatory. This constraint is applied if run of machine in stage and that of the first run of machine in stage are both assigned to sublot of job

|  |  |
| --- | --- |
|  |  |

1. The sequence dependent setup time has to be considered by taking into account the type of the job that was processed in run of machine m in stage I for run

|  |  |
| --- | --- |
|  |  |

1. Logical relations among the binary variables

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |
|  |  |

1. a production run of a given machine can be assigned to a sublot if and only if run of that machine is already assigned to another sublot.

|  |  |
| --- | --- |
|  |  |

1. Sum of the sizes of the sublots of a given job should be equal to the lot size of that particular job.

|  |  |
| --- | --- |
|  |  |

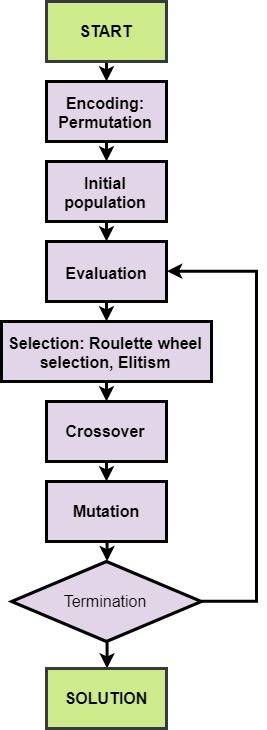
1. If sublot of job n is positive (i.e., ) and it requires processing in stage , then it should be assigned to one of the eligible machines in stage .

|  |  |
| --- | --- |
|  |  |
|  |  |

1. The makespan of the schedule, , is greater or equal to the completion time of any sublot on any stage. At optimality, , takes the value of the completion time of the last sublot to be processed in the system.

|  |  |
| --- | --- |
|  |  |

**GA**



The discussed model is apparently NP-hard. To this end, we develop an efficient procedure based on GA to search for optimal or sub-optimal solutions of the problem. In the following, we explain the solution encoding, evaluation, and genetic operators of the GA tailored to solve the hybrid flexible flowshop lot streaming problem.

**The main characteristics of the problem are:**

|  |  |
| --- | --- |
|  | Number of stages where stages are indexed by |
|  | Number of machines in stage where machined are indexed by |
|  | Number of jobs where jobs are indexed by |
|  | Set of stages visited by job , |
|  | Processing time for one unit of job on machine in stage |
|  | set of eligible machines for the job at stage |
|  | due date for job |
|  | weight (importance) of the job |
|  |  |
|  |  |

**The objectives are:**

1. Makespan minimization

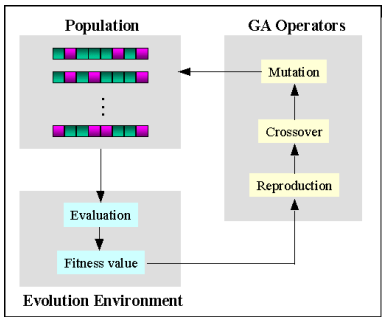
2. Minimization of the weighted sum of Tardiness

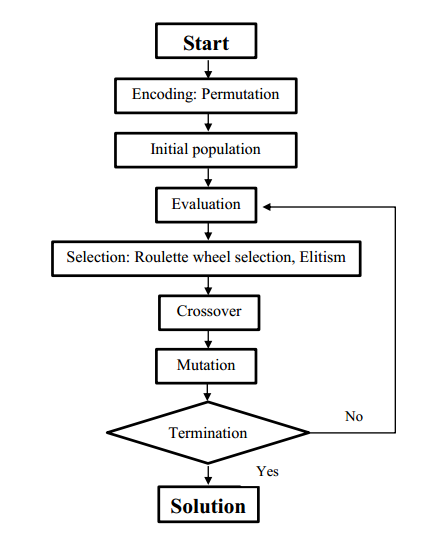
The completion time of a job is the instant in which the last task of this job is  
completed. Thus, the **makespan**  is the completion time of the last system task, i.e.,  
 The **tardiness** of a job is a defined as . Each job is associated with a weight according to its importance.

**Conceptual Models**

The following table shows the basic flow chart relevant to our project.







This program deals with pure flow shop scheduling that only supports static job allocation. The processing time of each job on each machine is constant. Here we input a set of jobs that must be executed in a particular static sequence on a set of machines. The output to the problem is an optimum sequence.

