# An optimization tool for process planning and scheduling

Mathieu BETTWY[1], Karine DESCHINKEL[2], Samuel GOMES[1]

[1] IRTES-M3M, Université de Technologie Belfort Montbéliard (UTBM),France
[2] FEMTO-ST Institute, UMR 6174 CNRS-University of Franche-Comté, France

**Abstract.** Process planning and scheduling are one of the most important functions to support flexible planning in a manufacture. The planning and scheduling should be solved simultaneously and not sequentially for productivity improvements in manufacturing. In this paper, we propose an optimization tool based on genetic algorithm (GA) approach to help person in charge of process planning and scheduling to find the most promising sequence of operations considering a choice of machines on which to perform the operations. Minimizing makespan is the evaluation criteria.

**Keywords:** Process planning, Process scheduling, Genetic algorithm

## 1 Introduction

Many industries are trying to best optimize the whole system to deal with a global manufacturing industry more competitive. This will require to reconsider the supply chain. As a result, companies must migrate from separated planning processes toward the integrated planning process to provide competitive products while reducing costs and / or production time.

We define the problem as an integrated problem of process planning and scheduling (IPPS). A lot of work has been done on this subject and various approaches have been used to solve the problem.

Tan (2000) [6] presents a review of the research in the process planning and scheduling area and discusses the extent of applicability of various approaches. They show that the efficient planning considering the alternative machines results in reduced lead-time and in improved overall machine utilization.

Moona and Seo (2005) [4] develop an evolutionary algorithm (EA)-based to solve some flexibility problems on shop floor.

Yuan and Xu (2013) [7] show an heuristic algorithm to figure out large-scale shop floor problem.

Li, Shao, Gao and Qian (2010) [3] develop a hybrid algorithm (HA) based-approach has been developed to facilitate the integration and optimization of process planning and scheduling in same time.

Kim and Choi (2014) [2] propose to solve the backward on-line job change scheduling problem, a production system-based simulation methodology.

However, the main weakness of the models introduced from this two decades is that they consider alternative machines for each operation for a fixed assembly sequence. The originality of this paper is to consider a set of alternative assembly sequences and to provide the best one optimizing multiple criteria. These criteria values are obtained through the resolution of the IPPS based on genetic algorithm method. In this paper, we consider a IPPS problem for a plant composed of a set of alternatives machines, multiple product flows and various assembly sequences. The alternative machines have different capabilities and require unequal processing time for an operation. Section 2 is dedicated to the problem definition. Section 3 gives details of our approach. The paper ends with a conclusion where our contribution is summarized and planned future work is discussed.

## 2 Problem definition

In many manufactures, operations have a different possible way to be done with a set of alternative process plans. Fig. 1 shows an example of different flow to produce in a shop floor.

Shoop floors include several machines, which have different functions, processing time, and capabilities. Therefore, actual optimization should be done by considering the available machines and their ability. Fig. 2 shows the different steps of our proposed methodology for determining process planning and scheduling.

Our study is based on previous work done by ROBERT [5] concerning the tool "Orasse Product". This tool aims to guide the users during the definition of the assembly sequence of the generic product of a family. It goes from the concept with its list of components and the modular product architecture (from the adaptation of the FAST diagram) to the assembly sequence defined by the assembly planner thanks to Orasse Product features. From an assembly and kinematic scheme (first quarter of Fig. 2),

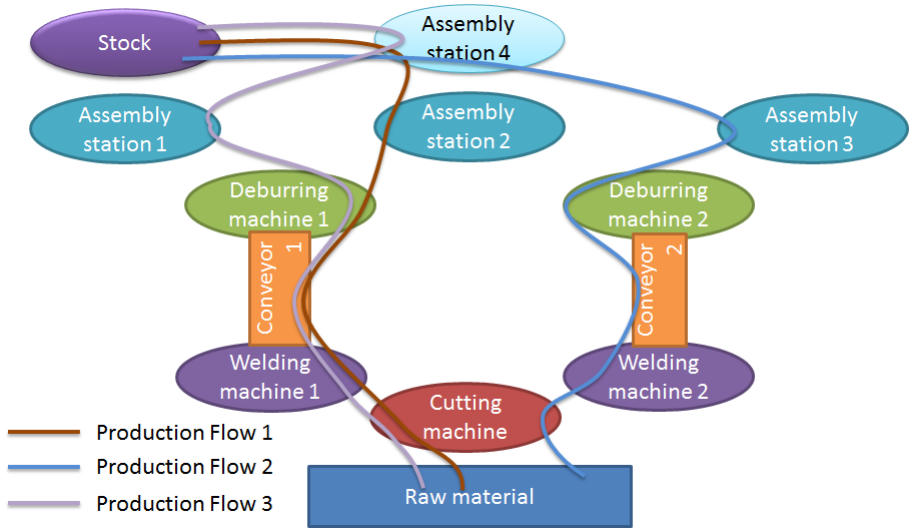


Fig. 1: Production flows in a shop floor

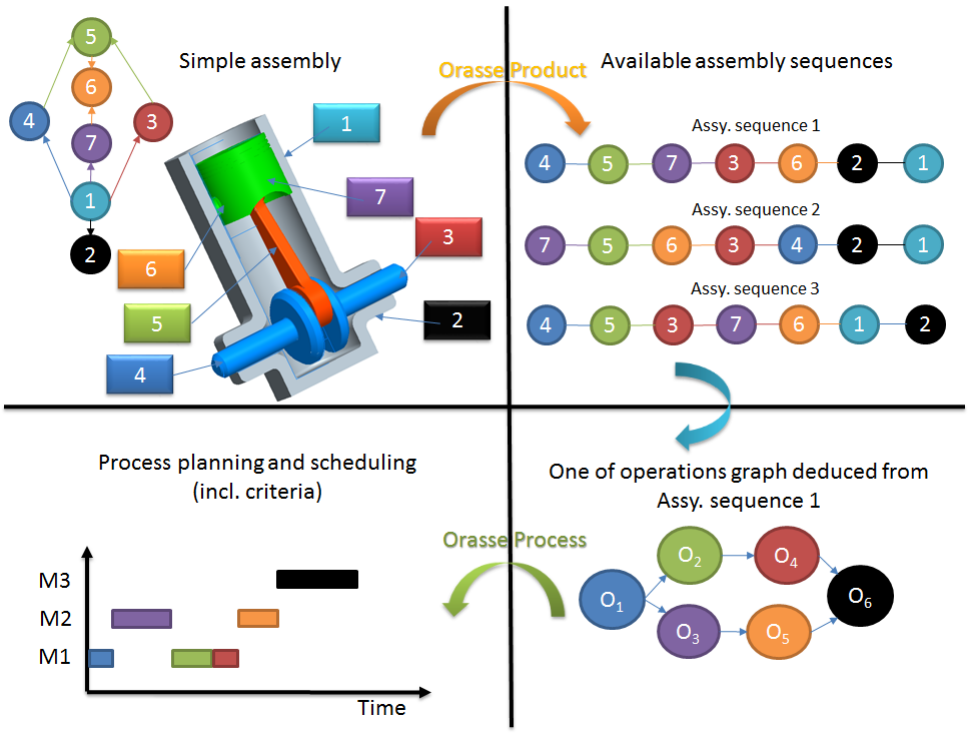


Fig. 2: Proposed Methodology

"Orasse Product" helps assembly planners to build promising assembly sequences. In this tool, physical contacts and precedence constraints between components are modeled by a directed graph and algorithms based on partitioning matrix and graph theory are developed in order to guide the assembly planner during the constitution of the best assembly sequence. Based on the same methodology, we intent to develop "Orasse Process" (Fig. 3). In case of "Orasse Process", we consider an operations graph deduced from an assembly sequence generated by "Orasse Product". Next step consists in determining a process planning and a process scheduling associated to this operations graph regarding to specific criteria. For the moment we have limited our approach to minimizing a single criterion : the makespan.

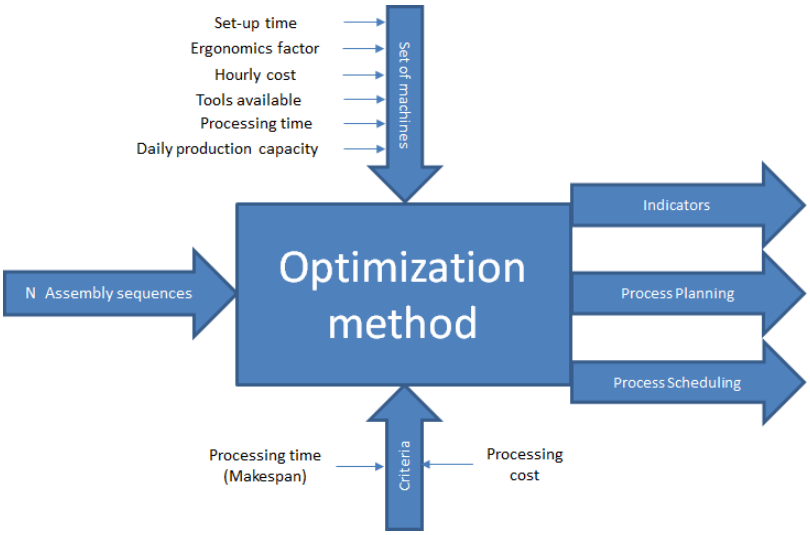


Fig. 3: Aims of Orasse process

## 3 Algorithms to solve IPPS

Genetic algorithm (GA) is well suited to deal with optimization problem with a large searching space as in the IPPS (several assembly sequences, alternatives machines, mutli criteria). Moreover GA produces a set of near optimal solutions compared to other optimization methods which

give only one solution. This can be useful for a person in charge of the shop floor production to select a solution among a reduced number of acceptable solutions taking into account particular criteria. A theoritical foundation of GA and their convergence to an optimal solution can be found in [1].

In this work, each chromosome is represented by a string of priorities: one distinct priority value for each operation. This string of priorities induces a specific order to derive a feasible schedule. The **Makespan algorithm (AF)** explains in details how the schedule is constructed and how the assignment of machines on operations is determined. The value of the resulting makespan is used as the fitness of the chromosome in the **Genetic Algorithm**.

### 3.1 Makespan algorithm

The **Makespan algorithm (AF)** computes the fitness function by considering the priorities $P_i$ assigned to each operation $i$. Input data is a directed graph of operations denoted by $G = (O, E)$ where $O$ is the set of nodes and $E$ is the set of arcs. Each node corresponds to an operation and a precedence constraint between two operations is modeled by an arc. The set of machines and their processing time for each operation are assumed to be known (filled or modified by the user) and data are saved in a matrix $T$ where $T(i, j)$ is the time to operation $i$ on machine $j$ and $T(i, j) = 0$ if operation $i$ is not feasible on machine $j$. The **Makespan algorithm (AF)** processes the operations in the order given by the priorities and assign a machines with the minimum time among available machines for the corresponding operation. An unique sequence with a given makespan $\sigma$ is generated.

### 3.2 Genetic Algorithm

Genetic Algorithms (GAs) are stochastic, population-based search algorithms to deal with multiobjective optimization problems. GAs start by initializing a set (population) containing a selection of chromosomes (individuals). A fitness associated to each individual allows to distinguish between better and worse individuals. A GA iteratively tries to improve the average fitness of a population by construction of new populations. A new population consists of individuals (children) built from the old population (parents) by the use of re-combination operators (crossover and mutation operators). Better individuals have higher probability to be selected for re-combination than other individuals. After some criterion is

met, the algorithm returns the best individuals of the population.
In our case, each chromosome is represented by a string of priorities (an integer between 1 and the number of operations). Figure 4 gives an example of chromosome. The initial population contains individuals randomly generated. The crossover and mutation operators are represented in figure 5. The fitness (here the makespan) of each chromosome of the population is evaluated through the execution of algorithm (AF). After a given number of iterations without improvement of the average fitness of the population, the algorithm stops and provides the best individuals. As we consider a set of alternatives assembly sequences and consequently a set of alternatives operations sequences, we have to repeat the optimization process for different populations. At the end, our optimization tool will provide a set of promising schedules to the person dealing with the shop floor production.

| O1 | O2 | ... | Oi | ... | ON |
|---|---|---|---|---|---|
| 5 | 1 | 2 | 8 | 7 | 3 |

Fig. 4: Chromosome representation

## 4 Conclusion

In this paper, a methodology based on genetic algorithm is proposed to solve IPPS problem by considering various assembly sequences for a same product and a set of machines for a same operation.
We intent to enrich our model by introducing many other criteria as transportation times for all pairs of machines, set-up times between operations, ergonomy and so on. Subsequently we will incorporate specific knowledge of the IPPS in the GA, so which generally improves its efficiency. A software "Orasse Process", proposing users the whole method, is under development.

## References

1. David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Longman Publishing Co., Inc., New York, NY, USA, 1989.

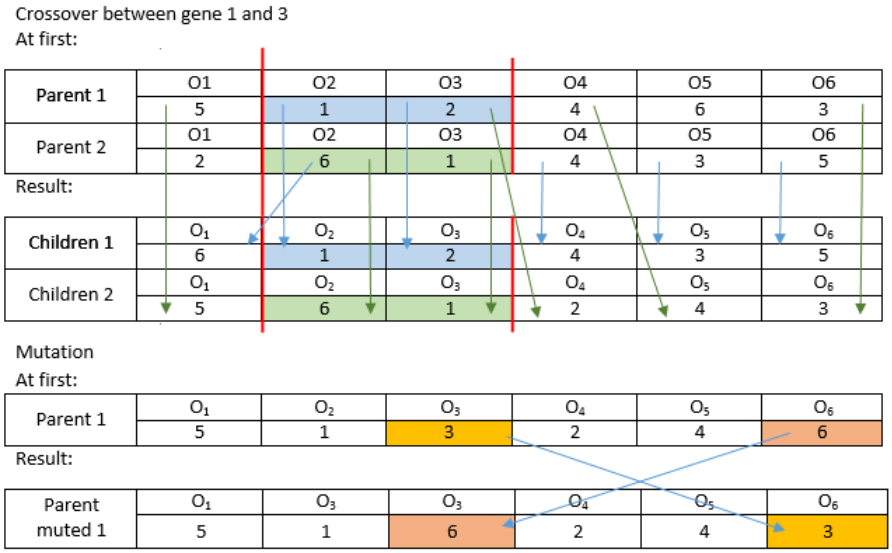


Fig. 5: crossover and mutation

2. Taedong Kim and Byoung K. Choi. Production system-based simulation for backward on-line job change scheduling. *Simulation Modelling Practice and Theory*, 40(0):12 – 27, 2014.
3. Xinyu Li, Xinyu Shao, Liang Gao, and Weirong Qian. An effective hybrid algorithm for integrated process planning and scheduling. *International Journal of Production Economics*, 126(2):289 – 298, 2010.
4. Chiung Moon and Yoonho Seo. Evolutionary algorithm for advanced process planning and scheduling in a multi-plant. *Computers & Industrial Engineering*, 48(2):311 – 325, 2005.
5. Aurelié Robert. *Vers une méthodologie de structuration de la dynamique des interactions au sein dumodèle de conception Multi-Domaines et Multi-Vues - Application à la conceptionde familles de produits modulaires*. PhD thesis, UTBM, 2012.
6. Wei Tan and Behrokh Khoshnevis. Integration of process planning and scheduling—a review. *Journal of Intelligent Manufacturing*, 11(1):51–63, 2000.
7. Yuan Yuan and Hua Xu. An integrated search heuristic for large-scale flexible job shop scheduling problems. *Computers & Operations Research*, 40(12):2864 – 2877, 2013.

**Algorithm 1** AF Algorithm

**Input:**
G: Operations Graph (with O : set of nodes, and E set of arcs)
M: Set of Machines
T: Operating time

| | Machine 1 | Machine 2 | ... | Machine j | ... | Machine M |
|---|---|---|---|---|---|---|
| Operation 1 | T(1,1) | T(1,2) | ... | T(1,j) | ... | T(1,m) |
| Operation 2 | T(2,1) | T(2,2) | ... | T(2,j) | ... | T(2,m) |
| ... | ... | ... | ... | ... | ... | ... |
| Operation i | T(i,1) | T(i,2) | ... | T(i,j) | ... | T(i,m) |
| ... | ... | ... | ... | ... | ... | ... |
| Operation N | T(n,1) | T(n,2) | ... | T(n,j) | ... | T(n,m) |

P: set of priorities; $P_i$ = priority associated with operation $i$

**Output:**
Process scheduling
$S_i$: Starting time of operation $i$
$F_i$: Ending time of operation $i$
$D_j$: Date of availability of machine $j$
Assignment $A(i,j) = 1$ if operation $i$ is assigned to machine $j$ else $A(i,j) = 0$
$\sigma$ : Makespan

**Initialization :**
$G'(O', E') \leftarrow G(O, E)$
**for all** *machine* $j \in M$ **do**
  $D_j = 0$
  **for all** *operation* $i \in O'$ **do**
    $A(i,j) = 0$
  **end for**
**end for**
**Process scheduling :**
**while** $O' \neq \varnothing$ **do**
  **if** all nodes $\in O'$ have predecessor **then**
    Impossible assignment
  **else**
    Select $i \in O'$ with no predecessor and with the highest priority (minimum value of $P_i$)
    Select machine $j$ with smallest $T(i,j)$
    $A(i,j) = 1$
    $S_i = D_j$
    $F_i = S_i + T(i,j)$
    $D_j = F_i$
    $O' \leftarrow O' \setminus i$
    $E' \leftarrow E' \setminus (i,v), v \in O'$
  **end if**
**end while**
$\sigma = \max_{j \in M} D_j$