الجمهورية الجزائرية الديمقراطية الشعبية

وزارة التعليم العالي والبحث العلمي

جامعة سعيدةد. مولاي الطاهر

كلية الرياضيات و الإعلام الآلي و الاتصالات السلكية و اللاسلكية

قسم: الإعلام الآلي

Master's thesis in computer science

Speciality : Réseaux Informatique et Systèmes Répartis

Theme

Cycle Time Reduction in Server Production Environments

- **Presented by:**

BOUHAFS DAOUI

FATTAH SOUFIANE

- **Directed by:**

Dr. MANSOUR MEKOUR

HAS university year 2024-2025

Thanks

II thank Almighty God for giving us the health and the will to begin and complete this dissertation. TFirst of all, this work would not be as rich and could not have been possible without the help and supervision of Dr. MANSOUR MEKOUR. I thank him for the quality of his exceptional supervision, for his patience, his rigor and his availability during our preparation of this thesis. II would like to thank the jury president and its members very much for agreeing to examine this work. MMy thanks also go to all my teachers for their generosity and the great patience they have shown despite their academic and professional responsibilities. II deeply thank all the people who have helped and supported me from near or far.

DAOUI SOUFIANE

Dedication

To my dear parents,
You are the roots of my journey, the pillars of my life. Thank you for your sacrifices, your unconditional love, and your faith in me. To my wife,
Your support, your patience, and your presence by my side have been a light in the most challenging moments. This success is also yours. To my children,
Your innocence and your smiles have given me the strength to move forward, a little further each day. May this accomplishment inspire you to believe in your dreams. To my brothers and sisters, Thank you for your affection, your sincere encouragement, and your unwavering solidarity. You are an integral part of this success.
With all my gratitude.
Daoui

Dedication

To the eternal memory of my father,
May eternal rest be granted to him. Your love and memory are with me.
To my dear mother,
For your tireless support and dedication, this work also belongs to you.
To my sisters,
For your affection and presence.
To my wife,
For your love, patience and strength that carried me through.
To my children,
You are my greatest achievement and my constant motivation.
Sofiane

Contents

List of Figures

List of Tables

General Introduction

Operational research, also known as decision support, brings together a set of scientific and technical methods aimed at modeling and solving concrete problems related to the management and optimization of complex systems. It is based on a diverse toolbox: algorithms, data structures, combinatorial optimization, graph theory, algorithmic complexity, linear and nonlinear programming, stochastic processes, probabilities, statistics, multi-criteria methods, etc. By its nature, operational research is an intrinsically multidisciplinary discipline, mobilizing knowledge in mathematics, computer science, economics, industrial management, among others. Today, its applications are widespread in industry and services, through techniques that have become classics such as linear programming or PERT analysis.

One of the major areas of operations research is scheduling. This field of study focuses on the optimal planning of tasks to be executed on limited resources (machines, workers, etc.) while respecting precedence, duration, or availability constraints. One of the most studied models in this area is the Job Shop problem, where multiple jobs must be processed on multiple machines, each job having a specific sequence of operations to be executed in a predefined order. Each machine can only process one operation at a time, and the objective is often to minimize the makespan (denoted C_max), i.e., the total time required to complete all the operations.

Optimizing this makespan is a complex challenge due to the combinatorial nature of the problem. To this end, numerous optimization methods have been proposed, among which genetic algorithms stand out for their robustness and ability to efficiently explore large solution spaces. Inspired by the process of natural evolution, these algorithms use mechanisms such as selection, crossover, and mutation to evolve a population of potential solutions. While they do not necessarily offer exact solutions, they can achieve satisfactory results in a reasonable time, making them particularly attractive for large problems.

This thesis is structured around four main chapters:

- Chapter 1: We discuss customer order management, presenting its foundations, its operational process, the associated technological tools, as well as the challenges encountered and the best practices observed in the industrial field.

- Chapter 2: We present in general terms the scheduling problem, describing its characteristics, its mathematical modeling, its different forms of representation, as well as the main resolution methods.

- Chapter 3: This chapter is devoted to genetic algorithms: we explain their basic principles (variation, adaptation, heredity), their fundamental operators (selection, crossover, mutation), their different models, as well as their overall operating mechanism.

- Chapter 4: We apply the previous knowledge to the resolution of the Job Shop problem using genetic algorithms. We show how this approach can be used to efficiently handle customer order scheduling, taking into account production constraints and aiming to minimize costs and lead times.

The problem of scheduling customer orders is indeed a strategic issue for companies, particularly in make-to-order or make-to-assembly environments, where customer satisfaction depends directly on the responsiveness and efficiency of the production line.

Customer order scheduling is a major challenge for many companies, particularly those operating in make-to-order or make-to-assembly manufacturing. It involves organizing product production and delivery to meet often complex customer demands while optimizing resources and costs.

What is customer order scheduling?

Customer order scheduling involves determining the sequence of tasks to be performed to produce the items ordered by customers, taking into account resource constraints (machinery, personnel, raw materials) and promised delivery times. Unlike "classic" production scheduling, which focuses on individual tasks, customer order scheduling considers each order as a set of interdependent tasks, and the goal is often for all tasks in an order to be completed before the order is considered delivered.

Finally, in the last chapter we present our simulation and explain how we do it.

Chapter 1

Order Management

1.1 Introduction

Customer order management is a central process in manufacturing environments, whether for the manufacture of standardized goods, custom products, or industrial services. This process covers the receipt, validation, planning, execution, and delivery of orders, while ensuring customer satisfaction and resource optimization. In a competitive environment, effective order management improves responsiveness, reduces costs, and strengthens customer loyalty.

1.2 Fundamental concepts

Order management is based on several key principles:

- **Traceability:** Each order must be followed at all stages to ensure transparency.

- **Interdepartmental collaboration:** The commercial production departments and logistics must work in synergy.

- **Automation:** Digital tools reduce human errors and speed up processes.

- **Flexibility:** Systems must adapt to variations in demand and custom orders.

The figure 2 shows a typical dashboard used to visualize the status of real-time orders, including lead times, quantities and priorities.

Order number	Customer	Order date	Total amount	Order Status
CMD001	Company A	2024-03-01	1000 €	In progress
CMD002	Company B	2024-03-05	1500 €	Book
CMD003	Company C	2024-03-10	800 €	On hold

Table 1.1: Order tracking dashboard

1.3 Order Management Process

The order management process can be broken down into several steps, illustrated by the following diagram:

1.3.1 Receipt of the Order

The order is received through various channels: online portal, EDI, email or direct contact. The information collected includes:

- Customer details (name, address, contacts).
- Product specifications (reference, quantity, options).
- Delivery times and commercial conditions.

A CRM (Customer Relationship Management) makes it easier to centralize and organize this data.

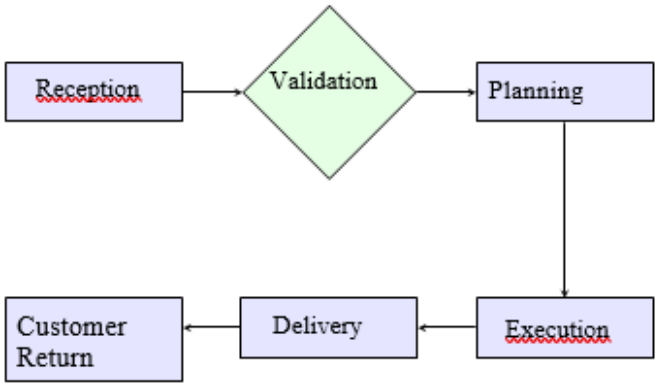

Figure 1.1: **Complete order management process flow**

1.3.2 Validation and Verification

Before going into production, the order is validated to confirm:

- Availability of raw materials.
- Production capacity (machines, personnel).
- The feasibility of the requested deadlines.

In the event of a problem, a negotiation with the customer is initiated.

1.3.3 Production Planning

The order is integrated into the production schedule using an ERP (Enterprise Resource Planning). This step involves:

- Prioritization of orders according to constraints.
- Allocation of resources (machines, materials, labor).
- Optimizing sequences to reduce downtime.

A Gantt chart, like the one in Figure 5, is often used.

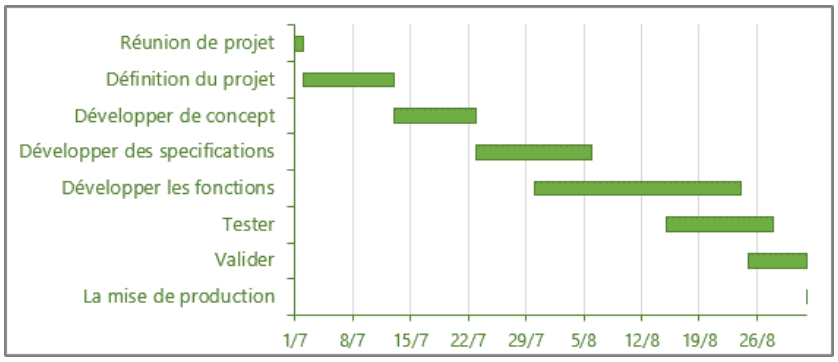

Figure 1.2: **Gantt chart for planning**

1.3.4 Production Execution

Production is launched with real-time monitoring via an MES (Manufacturing Execution System). Activities include:

- Manufactured according to specifications.
- Quality controls at every stage.
- Incident management (breakdowns, faults).

1.3.5 Delivery and Tracking

After manufacturing, the products are prepared for shipping using a WMS (Warehouse Management System). Post-delivery, tracking is carried out to collect customer feedback and identify possible improvements.

1.4 Technological Tools

Digital tools are essential for efficient order management. The table 1 summarizes the main systems used.

Tool	Function	Examples
ERP	Integrated management resources	SAP, Odoo, Oracle NetSuite
CRM	Customer relationship management	Salesforce HubSpot, Zoho
MES	Production monitoring	Siemens, Opcenter, GE Proficy
WMS	Warehouse management	Manhattan, Logiwa, SAP EWM
EDI	Automated data exchange	IBM Sterling, OpenText

Table 1.2: Technological tools for order management

1.5 Challenges and Solutions

Order management faces several challenges:

1. **Growing Personalization:** Customers demand tailor-made products.
   Solution: Use configurators integrated into ERPs.

2. **Supply disruptions:** Raw material shortages are causing delays.
   Solution: Implement buffer stocks and automatic alerts.

3. **Organizational silos:** Poor communication between departments leads to errors.
   Solution: Adopt collaborative platforms like Microsoft Teams or Slack.

4. **Respect for deadlines:** The abandonments increase the pressure on production.
   Solution: Use optimization algorithms for planning.

1.6 Best Practices

- Continuing education: Train employees in new digital tools.
- Systems Integration: Ensure interoperability between ERP, CRM, WMS.
- Data analysis: Use analytical tools to anticipate trends.
- Customer feedback: Integrate customer feedback into improvement processes.

1.7 Conclusion

Customer order management is a strategic lever for production companies. By adopting structured processes, advanced technologies and solutions adapted to the

challenges, organizations can improve their operational efficiency and competitiveness. This report detailed every aspect of the process, supported through figures and concrete examples, to offer a complete vision and practical.

Chapter 2

Scheduling

2.1 Introduction

In this chapter, we will provide a general introduction to the scheduling problem. We will begin by defining its foundations and theoretical framework. We will then present the mathematical formulation of the problem, highlighting the constraints and objectives generally associated with it.

The following sections will be devoted to two essential aspects:

- The classification of the different types of scheduling problems, according to the resources, constraints and objectives targeted;

- The representation of these problems, through models and notations allowing their formalization and algorithmic treatment.

Finally, we will conclude this chapter with an overview of the main resolution methods, whether exact, heuristic or metaheuristic, highlighting their advantages, limitations and areas of application.

2.2 General information on scheduling

Scheduling is the process of programming the execution of a project by assigning resources to tasks and setting their completion dates. It is considered a branch of operations research and production management that aims to improve the efficiency of companies in terms of production costs and delivery times.

Scheduling problems appear in all areas of the economy: IT, construction (project monitoring), industry (workshop problems, production management), administration (timetabling). Tasks are the common denominator of scheduling problems; their definition is neither always immediate nor always trivial. For this, it is necessary to program the tasks in such a way as to optimize a certain objective which will be, depending on the case, the minimization of the total duration (this

is the most frequently used criterion) or the respect of order dates or smoothing of labor curves or even the minimization of a cost. Generally speaking, three types of objectives are essential in solving scheduling problems: the efficient use of resources, the shortest possible task execution time and the respect of pre-specified completion dates [10].

The different inputs of a scheduling problem are tasks, potential constraints, resources and the economic function. A scheduling problem involves assigning tasks to resources at given times, such that these tasks are subject to certain restrictions.

Customer order scheduling is much more than just a planning task; it's a critical strategic function that determines a company's ability to satisfy its customers, optimize its resources, and remain competitive in the marketplace. At its core, it involves organizing and sequencing all the operations required to complete and deliver the products or services ordered by customers, while meeting promised deadlines and minimizing costs.

2.3 Formulation of a scheduling problem

2.3.1 Tasks

A task is a job whose completion requires a number of elementary operations. Each elementary operation requires a certain number of time units (its duration) and resource units [12].

There are two types of tasks [9]:

- Preemptible tasks, which can be executed in several steps, thus facilitating the resolution of certain problems.

- Indivisible tasks, which must be executed in a single step and are only interrupted once completed.

2.3.2 Resources

A resource is a technical or human means used to carry out a task. There are several types of resources [9]:

- Renewable resources, which, after being allocated to a task, become available again (machines, personnel, etc.).

- Consumable resources, which, after being allocated to a task, are no longer available (money, raw materials, etc.).

Whether renewable or consumable, the availability of a resource can vary over time. Furthermore, in the case of renewable resources, we mainly distinguish between disjunctive resources which can only perform one task at a time and cumulative resources which can be used by several tasks simultaneously but in limited

number [9].

2.3.3 Constraints

A constraint expresses restrictions on the values that can be taken jointly by variables representing the relationships between tasks and resources. We distinguish between time constraints and resource constraints [12].

Time constraints include:

- The time constraints allocated, generally arising from management imperatives and relating to task deadlines (delivery times, availability of supplies) or the total duration of a project.

- Priority constraints and more generally technological coherence constraints, which describe the relative positioning of certain tasks in relation to others.

- Calendar constraints related to compliance with working hours, etc.

Resource constraints reflect the fact that resources are available in limited quantities. There are two types of resource constraints, linked to the disjunctive or cumulative nature of resources. A disjunctive resource can only be used by one task at a time. On the other hand, in a cumulative resource, the sets of tasks that cannot be carried out simultaneously are of any cardinality [12].

2.3.4 The criteria

A criterion corresponds to qualitative and quantitative requirements to be met in order to assess the quality of the established schedule.

The criteria depending on a given application are very numerous; several criteria can be retained for the same application. The choice of the most satisfactory solution depends on the previously defined criterion(s), which can be classified into two types, regular and irregular.

The different criteria are not independent; some are even equivalent. Two criteria are equivalent if an optimal solution for one is also optimal for the other and vice versa [10]:

- **Regular criteria** are decreasing functions of the completion dates of the operations. Some examples are cited below:

- Minimization of completion dates for actions.

- Minimizing the maximum completion dates of actions.

- Minimizing the average completion dates of actions.

- Minimizing delays in the completion dates of actions.

- Minimizing the maximum delays in the completion dates of actions.

- **Irregular criteria** are criteria that are not regular, that is to say, they are not monotonic functions of the end dates of execution of operations, such as:

- Minimization of outstanding amounts.

- Minimizing the cost of storing raw materials.
- Balancing machine loads.
- Optimization of tool changes.

2.4 Classification of scheduling problems

2.4.1 One-operation models

It is represented in a Single Machine Model and a Parallel Machine Model.

Single-machine model

For a single machine model, a single machine performs all the tasks to be performed. This model is illustrated in Figure 1.

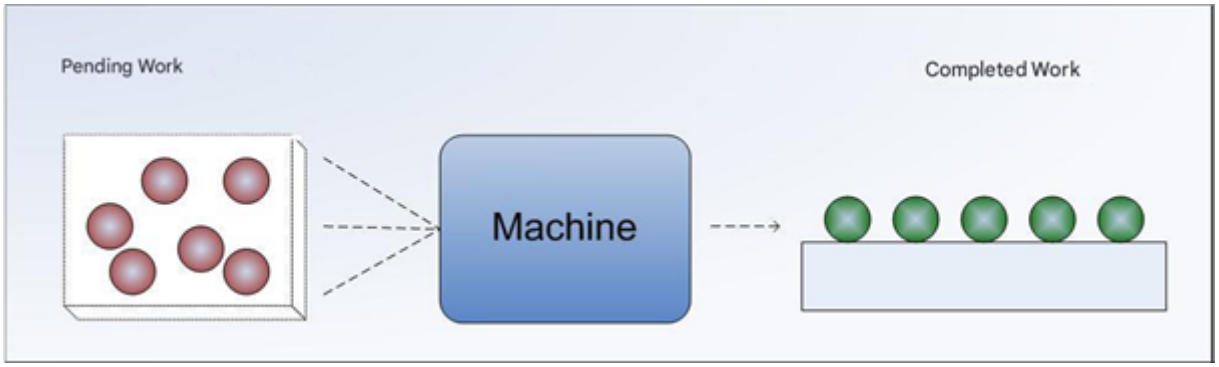

Figure 2.1: Single machine scheduling [a]

Parallel machine model

As for the parallel machine model, when machine i is released, the job is assigned to it, as shown in Figure 1.2. So this model is essential for the industrial sector, especially the textile industry. While parallel machines are classified according to their speed [12]:

- Identical machines (P): the execution speed is the same for all machines M_j and for all jobs J_i [12].

- Uniform machines (Q): each machine M_j has its own constant execution speed. The execution speed is the same for all jobs J_i of the same machine M_j [12].

- Independent machines (R): the execution speed is different for each machine M_j and for each job J_i.

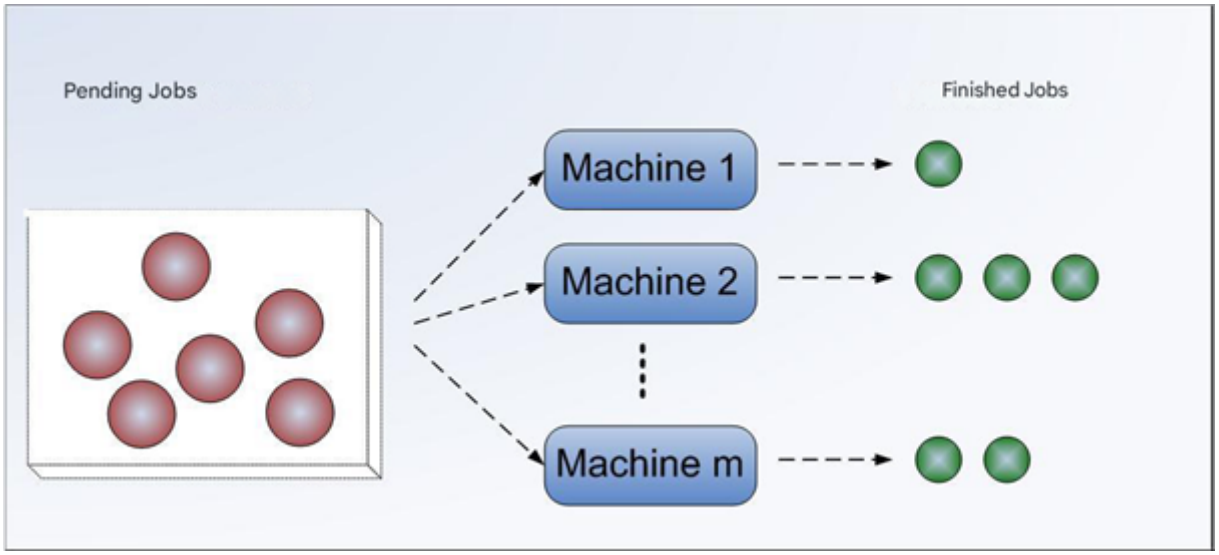

Figure 2.2: Parallel machine model [a]

2.4.2 Multi-operation models

The multi-operation model consists of cases where a job, to be completed, must pass through several machines, each of these machines having its own specificities. There are three models based on the order in which jobs pass through the machines: the flow-shop, job-shop, and open-shop models.

Flow-shop model

The flow-shop model, also called linear model and also called single-path workshops, because all jobs pass through the machines in the same order. As shown in the figure, we have four machines and four jobs. Where the jobs follow the same processing order on the machines.

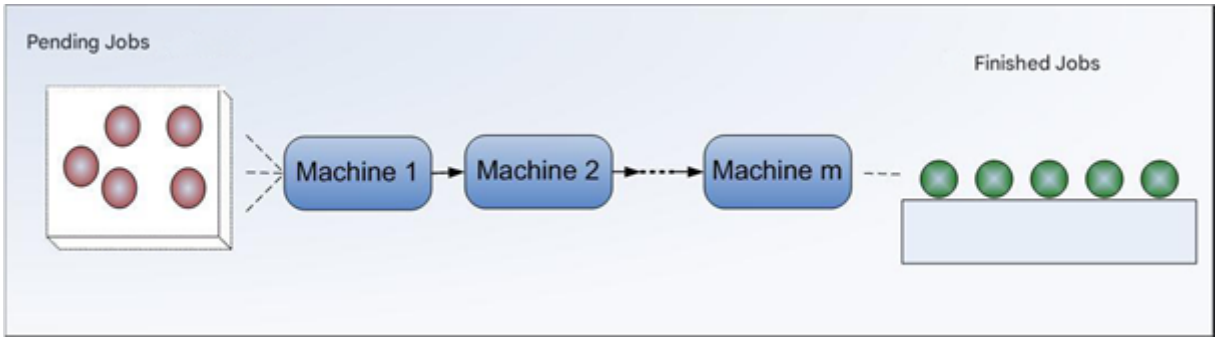

Figure 2.3: Single-path workshops (Flow-shop) [a]

Job-shop model

Regarding the job-shop model, also called multi-path workshops. This model consists of assigning a set J of r jobs $J_1, J_2, ..., J_r$ and M, a set of m machines $M_1, M_2, ..., M_m$. A set O of operations must be scheduled. Each job J_i is composed of a set of k operations, denoted $O_{i1}, O_{i2}, O_{i3}, ..., O_{ik}$ such that these operations are carried out in a well-determined order, and an operation can only belong to a single job. Each operation O is assigned to a machine M_j. The processing time of an operation O_{ij} is denoted P_{ij}. There is an example of a Job-shop model in Table 1 below, composed of two jobs. Each job has three operations. Each operation is to be performed on a given machine. There are three machines in the problem considered. Processing times vary from one operation to another.

Table 2.1: The job-shop model table

	$O_1 = (M_1; 4)$	$O_2 = (M_1; 3)$	$O_3 = (M_1; 2)$
Job 1	$O_1 = (M_1; 7)$	$O_2 = (M_1; 6)$	$O_3 = (M_1; 5)$
Job 2			

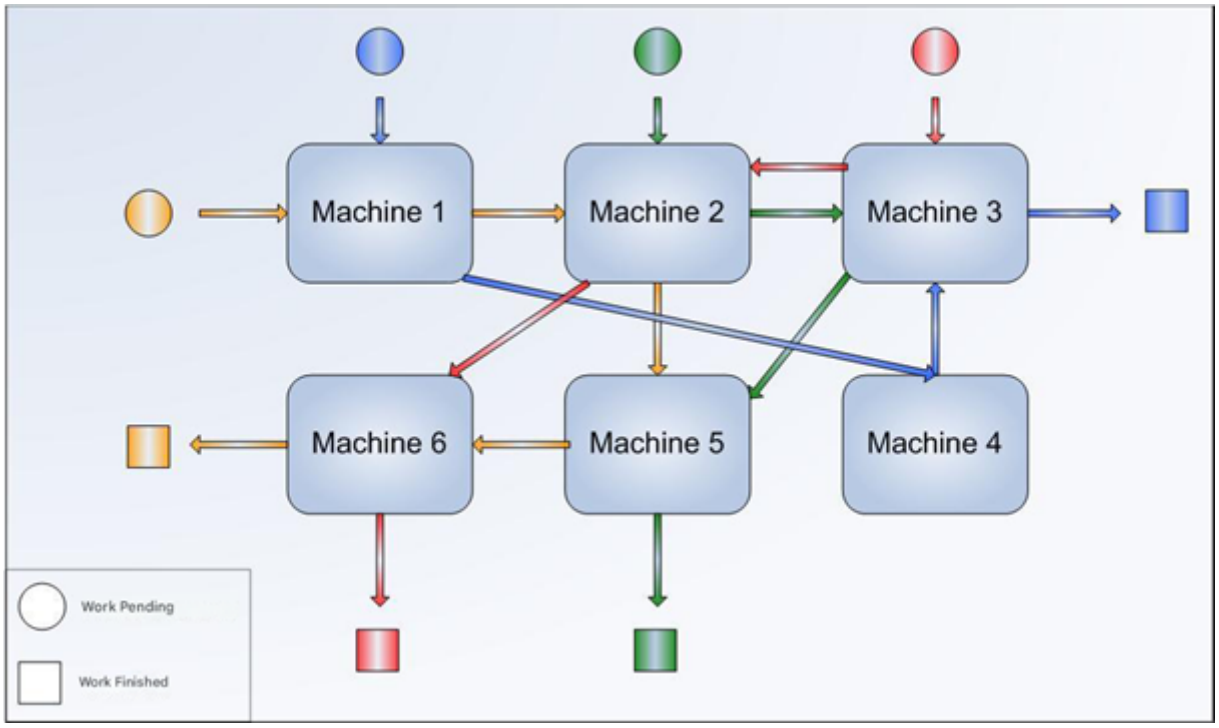

Figure 2.4: Multi-path workshops (Job-Shop) [a]

Linear formalization There are several linear formulations for the job shop and some of them are based on the Manne formulation. In Pham (2008), an evaluation of linear formulations of the job-shop is proposed.

For a Job-Shop problem of n jobs and m machines, we consider the following notations:

- $\tilde{J}$: the set of all jobs; $\tilde{J} = \{1, 2, ..., n\}$;
- I: the set of all operations;
- M: the set of machines $M = \{1, 2, ..., m\}$;
- A_j: the set of all pairs of consecutive operations for job $j \in \tilde{J}$;
- B: the set of all pairs of operations $(i, j) \in I$, $i \neq j$ executed on the same machine;
- I_k: the set of operations executed on the machine $k \in M$;
- P_i: the operating time of operation $i \in I$;
- H: a sufficiently large positive integer;
- x_i: the start date of operation $i \in I$;
- x_τ: the start date of the operation *;

- For each pair of operations $\forall(i,j) \in I$ running on the same machine $k \in M$:

$$y_{ij} = \begin{cases} 1 & \text{if operation } i \text{ is executed before operation } j \\ 0 & \text{otherwise} \end{cases}$$

Mathematical formalization

$$x_j - x_i \geq P_i \quad \forall(i,j) \in A_k, \forall k \in \tilde{J} \quad (1)$$
$$x_j + H(1 - y_{ij}) - x_i \geq P_i \quad \forall(i,j) \in B \quad (2)$$
$$x_i + Hy_{ij} - x_j \geq P_i \quad \forall(i,j) \in B \quad (3)$$
$$x_\tau - x_i - P_i \geq 0 \quad \forall i \in I \quad (4)$$
$$y_{ij} \in \{0,1\} \quad \forall(i,j) \in B \quad (5)$$
$$x_i \geq 0 \quad \forall i \in I \quad (6)$$
$$x_\tau \geq 0 \quad (7)$$

- Constraints (1) ensure: no operation can begin before the end of execution of the operation preceding it.

- Constraints (2) and (3) ensure: machine disjunction constraints and ensure that two operations running on the same machine must be ordered through the use of the binary variable $y_{ij} \in \{0,1\}$ of constraint (5).

- Constraints (4): set the start date of the operation *. This is ensured by the fact that x_τ is greater than all start dates plus processing time $x_i + P_i$ of each operation by $i \in I$.

- Constraints (6) and (7): require that all operation start dates are positive or zero [13].

Open-shop model

The Open-shop model is a less constrained workshop model than the Flow Shop and the Job Shop, such that each job j can have its own order of passage on all machines. This means that the arrangement is not known in advance. Although there are some difficulties in solving the scheduling problem due to the lack of prior arrangement, however, this model allowed us to simultaneously solve two problems: scheduling (determine the path of each job) and schedule the jobs, taking into account the ranges found.

2.5 Representation of scheduling problems

There are three possible kinds of representations of a scheduling problem: the Gantt chart, the Potential-Task graph and the PERT method.

2.5.1 The Gantt chart

The Gantt chart is an excellent method because it is very easy to represent the scheduling solution. It is also called the bar chart, invented by Henry Gantt (1861–1919). This method is used by many project managers. Figure 5 represents a scheduling of five jobs (J_1, J_2, J_3) on 2 identical parallel machines, such that the abscissa axis represents the time and on the ordinate axis appear the machines (M_1, M_2, M_3). On each horizontal line, we put the scheduling of the tasks on this machine. Each task is represented by a bar. The length of this bar is proportional to its duration. At the end, we obtain on the diagram the sequence of operations on each of the machines, with the start and end dates of each task.

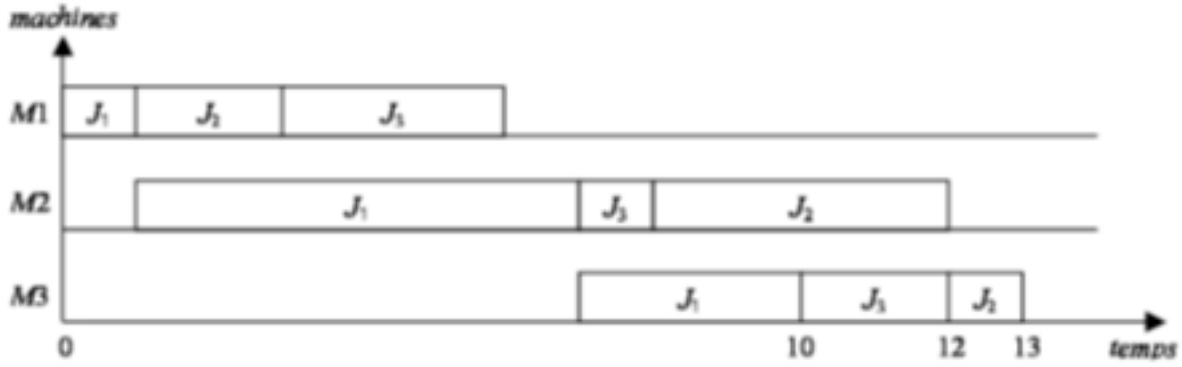

Figure 2.5: Example of Gantt Chart [c]

2.5.2 Potential-Task Graph

This graphical tool was developed using the theory of Petri nets which were mainly used to model dynamic systems with discrete events [11]. In this type of modeling, tasks are represented by nodes and constraints by arcs [3]. Thus, arcs can be of two types:

- The conjunctive arcs illustrating the precedence constraints and indicating the durations of the tasks,
- Disjunctive arcs indicating resource constraints [8], [1].

2.5.3 PERT (Program Evaluation and Research Task) method

The PERT method is a technique for managing scheduling in a project. It is represented in the form of a graph of a network of tasks, several tasks which, thanks to their dependency and chronology, all contribute to obtaining a finished product.

As such, the PERT method first involves:

- A precise breakdown of the project into tasks.
- Estimating the duration of each task.
- The appointment of a project manager responsible for monitoring the project, reporting back if necessary and making decisions in the event of deviation from forecasts.

2.6 Methods for solving the scheduling problem

2.6.1 Exact methods

An exact method is said to be a useful method when it is used to solve small problems. That is, when the computational time required to reach the optimal solution is not excessive [14]. These methods implicitly examine the entire search space to produce the optimal solution:
- Dynamic programming
- Linear programming
- Branch-and-Bound tree-based methods.

2.6.2 Approximate methods

These methods are considered for scheduling problems in which we do not find an optimal solution in a reasonable time [17]. Among these methods, heuristics and meta-heuristics.

Heuristics

Heuristics depend on empirical methods, as they are built on simplified rules to optimize one or more criteria. The general principle of this category of methods is to integrate decision strategies to construct a solution close to the optimal one while seeking to have a reasonable computation time [7].

Examples of heuristics:

- RANDOM: The operation is randomly selected from among the operations not yet scheduled.
- SPRT (Shortest Remaining Processing Time): This is the operation in which the operating time is shorter than that of the other operations.
- LPT (Longest Processing Time): This is the operation in which the operating time is longer than that of the other operations.

Meta-heuristics

A meta-heuristic is often defined as a procedure that best exploits the structure of the problem under consideration, with the aim of finding a solution of reasonable quality in as little computation time as possible [19]. The main meta-heuristics are those based on single-solution meta-heuristics (local search (LS), simulated annealing (SA), taboo search (TS)), and solution-population meta-heuristics (genetic algorithms (GA) and ant colony optimization (ACO) as well as the differential evolution (DE) algorithm).

In the context of production planning, the customer order scheduling problem can be modeled in a similar way to a Job Shop problem. Each customer order is considered a task or job, composed of several subtasks representing the different types of products or operations required to complete it. These subtasks must be executed on specific resources (machines or servers), according to capacity, order, and availability constraints.

Similar to the classic Job Shop, where each job follows a sequence of operations on different machines, a customer order follows a processing process on parallel, often heterogeneous servers, each with distinct speeds and capacities. The cycle time of an order is then equivalent to the makespan of a job, i.e., the time required to execute all the subtasks that compose it. This analogy makes it possible to apply classic scheduling techniques (such as heuristics or meta-heuristics such as PSO, GA, etc.) from the Job Shop domain, to the dynamic and optimized management of customer orders in an industrial or logistics environment.

2.7 Related work

In this chapter, we present articles from researchers who conducted research on our thesis topic, and we summarize each article as follows:

- (2004)Nourah Al-Angari, AbdullatifALAbdullatifsuccessfully apply parallel genetic algorithms to solve task scheduling problems. Fitness evaluation is the operation that consumes the most CPU time, which affects the AG performance. The proposed synchronous master-slave algorithm outperforms the sequential algorithm in the case of complex and high generation problems.

- (2019) JiaLuoa, ShigeruFujimurab and Didier El propose a Flow Shop scheduling model, using the peak power value considering new functions. As the problem is strongly NP-hard, due to new government legislation, customers' environmental concerns and the ever-increasing energy cost, energy efficiency has become a critical parameter in industrial manufacturing processes

in recent years. Most efforts, considering the energy issues in scheduling problems, have focused on static scheduling. But in fact, scheduling problems are dynamic in the real world with new jobs uncertain after the execution time. Indeed, they develop a parallel hybrid priority-based genetic algorithm with a predictive reactive full rescheduling approach. In order to achieve speedup to address the short answer in the dynamic environment, the proposed method is designed to be highly consistent with the NVIDIA CUDA software model. Finally, numerical experiments are conducted and show that their approach can not only achieve better performance than the traditional static approach, but also obtain competitive results by significantly reducing the time requirements.

- (2018) J.Adan, A.Akcay, J.Stokkermansb and R.VandenDobbelsteen deal with A hybrid genetic algorithm to improve the scheduling process, whose main features are an improved crossover mechanism for local search, two additional fast local search procedures and a user-controlled multi-objective adjustment function. Tests with real production data show that this multi-objective approach can achieve the desired balance between production time, setup time and delays, producing practically achievable high-quality production schedules.

- (2006) R.Nedunchelian, K.Koushik, N.Meiyappan, V.RaghudéveloppentA genetic algorithm to dynamically schedule heterogeneous tasks to heterogeneous processors in a distributed environment. The scheduling problem is known to be NP-complete. Genetic algorithms, a meta-heuristic search technique, have been successfully used in this area. The proposed algorithm uses multiple processors with centralized control for scheduling. Tasks are taken in batches and are scheduled to minimize execution time and balance processor loads. According to their experimental results, the proposed parallel genetic algorithm (PPGA) significantly decreases the scheduling time without adversely affecting the makespan of the resulting programs.

- (2013) Frank Werner gives an overview of some genetic algorithms for Shop Scheduling problems. In a Shop Scheduling problem, a set of Jobs must be processed on a set of machines in such a way that a specific optimization criterion is satisfied. Depending on the restrictions on the technological routes of the jobs, a distinction is made between Flow Shop (each Job is characterized by the same technological route), Job Shop (each Job has a specific route) and Open Shop (no technological route is imposed on the Jobs). He also considers some extensions of Shop Scheduling problems such as Hybrid or Flexible Shop (at each processing step we can have a set of parallel machines) or the inclusion of additional processing constraints such as control-

lable processing times, release times, setup times or the no-wait condition. After giving an introduction to basic genetic algorithms discussing short solution representations, initial population generation, selection principles, application of genetic operators such as crossover and mutation, and termination criteria, it discusses several genetic algorithms for particular problem types with an emphasis on their common characteristics and differences. Here, it focuses mainly on single-criteria problems (minimizing the validity time or a particular sum criterion such as total completion time or total delay), but briefly mentions some work on multi-criteria problems. It discusses some computational results and compares them with those obtained by other heuristics. Furthermore, it also summarizes the generation of benchmark instances for shop scheduling problems and gives a brief introduction to the use of the program package "LiSA-ALibrary of SchedulingAlgorithms" developed at "Otto-von-Guericke-University Magdeburg" to solve shop scheduling problems, which also includes a genetic algorithm.

- (2007)Kheireddine MERHOUM and Messaoud DJEGHABA develop an application that allows minimizing the makespan for a flexible job-shop scheduling problem they used the Genetic Algorithm. However the flexible job-shop scheduling problem in the literature is considered as a difficult problem to solve in the field of combinatorial optimization, its complexity is of type NP-complete in the strong sense. The objective is to show the performance of genetic algorithms (metaheuristics) in solving this kind of problem.

- (1997) Shyh-Chang Lin, Erik D. Goodman, and William F. Punch described a GA for job shop scheduling problems. Using the Giffler and Thompson algorithm, they created two new operators, THX crossover and mutation, which better convey the temporal relationships in the schedule. The approach produced excellent results on job shop scheduling problems. They tested many parallel GA models and scales in the context of job shop scheduling problems. The hybrid model composed of a Coarse-grained Genetic Algorithm with connected nodes in a fine-grain GA-style topology performed best, appearing to successfully integrate the advantages of both coarse-grain and fine-grain GAs.

- (1998) Erick Cantú-Paz organizes and presents in a unified manner some of the most representative publications on parallel genetic algorithms. To organize the literature, the article presents a categorization of the techniques used to parallelize GAs, and shows examples of all of them. However, since the majority of research in this area has focused on parallel GAs with multiple populations, the survey focuses on this type of algorithms. In addition, the article describes some of the most important problems in the modeling and

design of parallel GAs with multiple populations and presents some recent advances.

- (1993) Harpal Mainipresents genetic algorithms—evolutionary algorithms based on an analogy with natural selection and survival of the fittest—applied to a combinatorial NP-Complete optimization problem: minimizing the lifetime of a Flow Shop No Wait (FSNW). This is an important optimization criterion in real-world situations, and the problem itself has practical significance. We restrict our applications to a three-machine Flow Shop no wait problem known to be NP-complete. The stochastic assumption is that the processing times of the Jobs are described by normally distributed random variables. It discusses how this problem can be translated into a TSP problem, using the concept of a starting interval. Genetic, sequential, and parallel algorithms are then applied to search the solution space, and it presents the algorithms and empirical results.

- (2017) ArtanBerisha, EliotBytyc and ArdeshirTershnjaku tried with many techniques to find the most appropriate and fastest way to solve the problem. With the emergence of multi-core systems, parallel implementation was considered to find the solution, their approaches try to combine several techniques in two algorithms: coarse-grained algorithm and multi-threaded tournament algorithm. The results obtained from two algorithms are compared using algorithm evaluation function. Considering the execution time, coarse-grained algorithm performed twice as well as multi-threaded algorithm.

- (2018) JiaLuo and Didier EL BAZtheir work has been devoted to genetic algorithms (GA) to search for optimal solutions to shop scheduling problems. Due to NP-hardness, the time cost is always heavy. With the development of high-performance computing (HPC), interest has focused on parallel GAs for shop scheduling problems. They present the recent works on solving shop scheduling problems with the use of parallel GAs. It presents the most representative publications in this field by categorizing parallel GAs and analyzes their designs based on the frameworks.

- (1993) Hsiao-Lan Fang, Peter Ross, and Dave Corne describe a GA approach that produces reasonably good results very quickly on job-shop scheduling problems, better than previous efforts using genetic algorithms for this task, and comparable to existing conventional search methods. The representation used is a variant of one known to work quite well for the traveling salesman problem. It has the considerable merit that crossover will always produce legal schedules. A new method for improving performance is examined based

on dynamic sampling of convergence rates in different parts of the genome. Their approach also promises to efficiently solve the Open Shop Scheduling problem and the job-shop scheduling problem.

- (2007)Dudy Lim, Yew-Soon Ong, Yaochu Jinb, Bernhard Sendhoff and Bu-Sung Lee present Genetic Hierarchical Parallel Algorithm Framework with the Use of Grid Computing (GE-HPGA). Framework is developed using standard Grid technologies and has two distinctive features, firstly an extended Grid RPC API to hide the high complexity of Grid environment, and secondly for transparent resource discovery and selection. To evaluate the practicality of the Framework, a theoretical analysis of the proposed possible speedup is presented. An empirical study on GE-HPGA using a benchmark problem and a realistic airfoil shape optimization problem for various grid environments having different communication protocols, cluster sizes, processing nodes, at geographically disparate locations also indicates that the proposed GE-HPGA using grid computing offers a credible framework to significantly accelerate scalable design optimization in science and engineering.

- (2004) Murat Yildizoglu and Thomas Vallée present the basic mechanisms of these algorithms and an overview of their applications in economics, accompanied by a representative bibliography.

- (2016) Rakesh Kumar PHANDEN represents Job Shop Scheduling which is an important and complex problem for a manufacturing system. It is a well-known and popular problem having an NP-hard (non-polynomial) characteristic of finding the optimal or near-optimal solution (schedules) quickly. In Job Shop Scheduling, a set of "N" numbers of Jobs is processed through "M" number of a given set of machines. It has to be processed in the prescribed order using the feasible sequence of operations for a job. Therefore, due to its complex nature, searching for approximate solutions is chosen over searching for the exact solution which involves a higher cost. Various metaheuristic techniques are used in order to find the suboptimal solution for the job shop scheduling problem. Genetic algorithm and Variable Neighborhood Search (VNS) method are the preferred techniques which are better known for global and local solution searching, respectively. VNS works as to augment the GA approach. In the present work, multi-agents are proposed to find the near-optimal solution for the job shop scheduling problem using GA and VNS approach in parallel. The multi-agent system is preferred due to its parallel operation capability and robustness along with intelligence elucidation. In the proposed system, many hosts in the network are hosted by agents. JADE is used to set up the communications. Each agent is designed to perform the specific task namely Initialization Agent (IA), Processing Agent

(PA) and Coordination Agent (CA) for initial generation of the population, to schedule the operations on the machines, to find the distinctive host and to perform the migrations between different populations respectively. The objective is to find an optimal value of makespan for the job shop scheduling problem. The performance of the system is evaluated through a case study and it reveals that the proposed approach is efficient enough to find the optimal solution. Future work involves introducing perturbation agents (DAs) for both internal and external perturbations.

- (2012)Mostafa Akhshabi, JavadHaddadnia and Mohammad Akhshabi use a parallel GA to solve Flow Shop scheduling problems to minimize the makespan. According to their experimental results, the proposed parallel genetic algorithm (PPGA) significantly decreases the CPU time without adversely affecting the makespan.

- (2004) Fabien PICAROUGNE, Gilles VENTURINI and Christiane GUINOT present a parallel genetic algorithm (GA) that crawls the Web in order to find relevant documents in the context of business intelligence. They show how the Internet information retrieval problem can be modeled as an optimization problem: the Internet is a search space structured as a graph, and an evaluation function can be defined from the user's query. The GA manages a population of Web pages and decides which pages to crawl. The parallel architecture of Geni Miner II is distributed over a local network or the Internet, with each client having the possibility to register and become part of the search engine. They also show that the parallel GA achieves better results compared to a metasearch engine, and that it significantly decreases the time needed to obtain documents. Finally, the first tests carried out with real users show the potential of this system and the future directions to be considered.

- (2003) Michelle Moore presents parallel genetic algorithms that are applied to the NP-complete problem of scheduling multiple tasks on a cluster of computers connected by a shared bus. Experiments reveal that the parallel programming algorithm develops very accurate programs when parameter directives are used.

- (2019)Yuri N. Sotskov, Natalja M. Matsveichuk, and Vadzim D. Hatsura study two-machine Shop Scheduling problems provided that lower and upper bounds of the durations of 'n' Jobs are given before scheduling. An exact value of the job duration remains unknown until the end of the Job. The objective is to minimize the makespan (schedule duration). They address the question of how best to execute a schedule if the job duration can take

a real value of the given segment. Scheduling decisions can consist of two phases: an offline phase and an online phase. Using information about the lower and upper bounds for each available Job duration in the offline phase, a scheduler can determine a minimal dominating set of schedules (DS) based on sufficient conditions for schedule domination. The DS optimally covers all possible realizations (scenarios) of uncertain job durations in the sense that, for each possible scenario, there is at least one schedule in the DS that is optimal. The DS allows a scheduler to quickly make an online scheduling decision whenever additional information about job completion is available. A scheduler can choose an optimal schedule for most possible scenarios. We developed algorithms to test a set of conditions for schedule domination. These algorithms are polynomial in the number of jobs. Their time complexity does not exceed. Computational experiments have shown the effectiveness of the developed algorithms. If there were no more than 600 tasks, the 1000 instances of each tested series were solved in one second at most. A case with 10,000 tasks was solved in 0.4 s on average. Most instances of nine tested classes were solved optimally. If the maximum relative error of the working time was not greater than, then more than the tested instances were solved optimally. If the maximum relative error was equal to, then the tested instances of the nine classes were solved optimally.

- HOUNNOU Amèdédjihundé H, FIFATIN François-Xavier, DUBAS Frédéric, CHAMAGNE Didier and VIANOU Antoine develop a new sizing concept with the use of bi-objective optimization with NSGA II genetic algorithms. The two objective functions considered are the investment cost and the efficiency of the penstock. The diameter and length of the penstock are considered as optimization variables. The sizing method is applied to three potential hydroelectric development sites in Benin. For each site, the simulation results present Pareto front curves that represent the set of non-dominated solutions. They noted that the diameter is a determining parameter in the bi-objective optimization of the investment cost and the efficiency of the penstock. The length is not an optimization parameter but rather is a specific parameter to be defined taking into account the environmental constraints of the site. This study also has the advantage that it can be used to size the penstock for any site.

2.8 Conclusion

This chapter introduced scheduling, a pillar of operations research aimed at optimizing resource allocation and task planning. It covered the theoretical foun-

dations, mathematical formulation, constraints and objectives, as well as problem classification (single machine, parallel, flow-shop, job-shop, open-shop). Representation tools (Gantt, PERT, Potential-Task graph) and resolution methods (exact, heuristic, meta-heuristic) were presented. The analogy with the job-shop underlines the importance of scheduling for customer order management, strengthening the efficiency and competitiveness of companies.

Chapter 3

Genetic Algorithms

3.1 Introduction

In this chapter we explain what genetic algorithms are, and their principle in addition to how they work, then do not forget the operators of genetic algorithms then exploit the meta heuristics, finally represent the parallelism and their models.

3.2 Definition

In the 1970s, genetic algorithms were created by John Holland, they are algorithms derived from nature, their paradigm is linked with genetics which is the population, it means a set of solutions, and the individual represents a solution, in addition the chromosome is a component of the solution, finally the gene which is a characteristic, there are three operators of genetic algorithms: selection, crossover and mutation.

3.3 Principles of genetic algorithms

There are 3 principles for genetic algorithms for the evolution of species using Darwin's theory:

3.3.1 Principle of variation

When we have a population, each individual or the population is unique, so these are very important differences, which helps in selection.

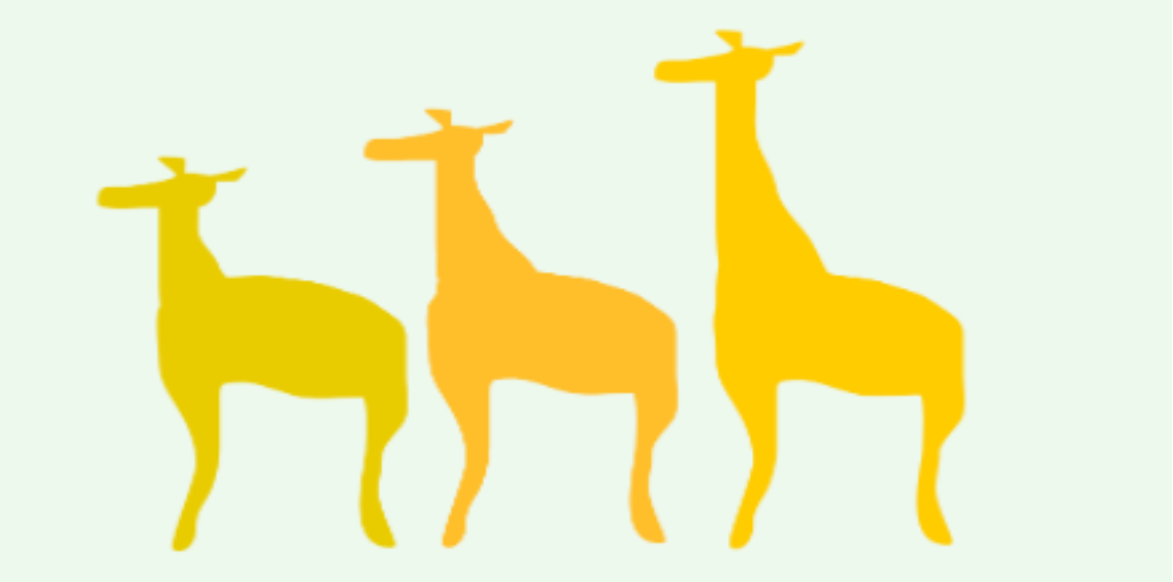

Figure 3.1: Representation of the principle of variation [b]

3.3.2 Adaptation principle

This principle is based on the search for individuals who are able to reach adulthood more easily due to their ability to adapt to their environment and this means that they have the ability to survive and reproduce.

Figure 3.2: Representation of the principle of adaptation [b]

3.3.3 Principle of heredity

Individuals must have hereditary characteristics that are passed on to their descendants, as this ensures the evolution of the species while retaining beneficial properties for individuals.

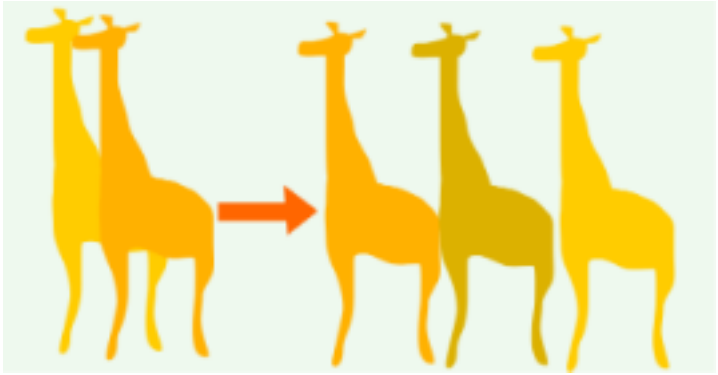

Figure 3.3: Representation of the principle of heredity [b]

3.4 Genetic algorithm operators

3.4.1 The selection operator

Selection is the application of the adaptation principle of Darwin's theory. It means the way to choose individuals based on knowledge of which individuals are best suited in order to have a population of solutions closest to converging towards the global optimum. There are several selection techniques:

- Selection by rank: we will rank the individuals based on their score and we will choose the individuals who have the best adaptation scores.

- Selection probability proportional to adaptation: this is the roulette or wheel of fortune technique, where the probability of choosing each individual is linked to its adaptation to the problem.

individual	Objective function	percentage
1	30	39%
2	5	7%
3	16	21%
4	10	13%
5	15	15%

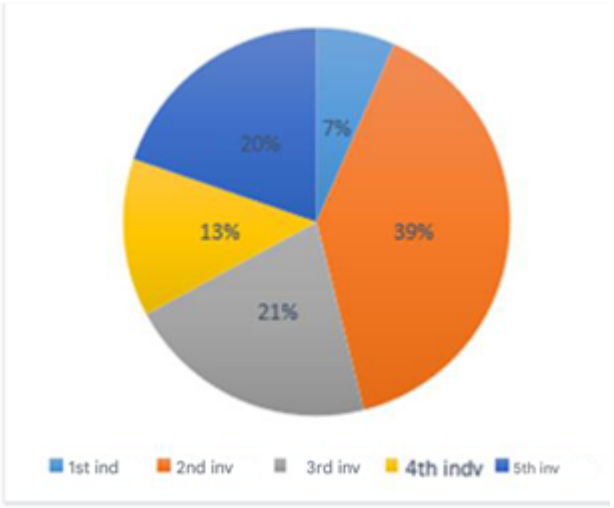

Figure 3.4: Selection probability proportional to adaptation

- Tournament selection: we will select proportionally from pairs of individuals, then we choose from these pairs the individual who has the best adaptation score.

Figure 3.5: Tournament selection

- Uniform selection: we will select randomly, uniformly without taking into account the adaptation value.

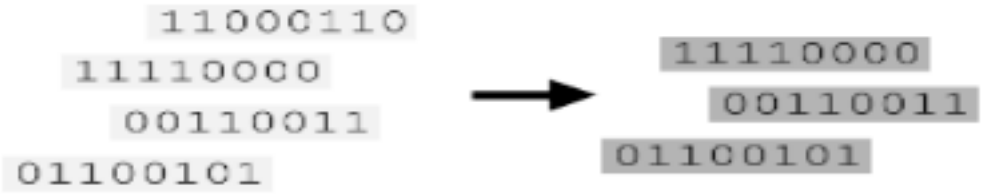

Figure 3.6: Individuals in binary representation after selection [16]

3.4.2 The crossover operator

Crossing over is when two chromosomes share their characteristics. This allows for the genetic mixing of the population and the application of the principle of heredity from Darwin's theory. There are two methods for crossing:

- Simple crossing (one point only):

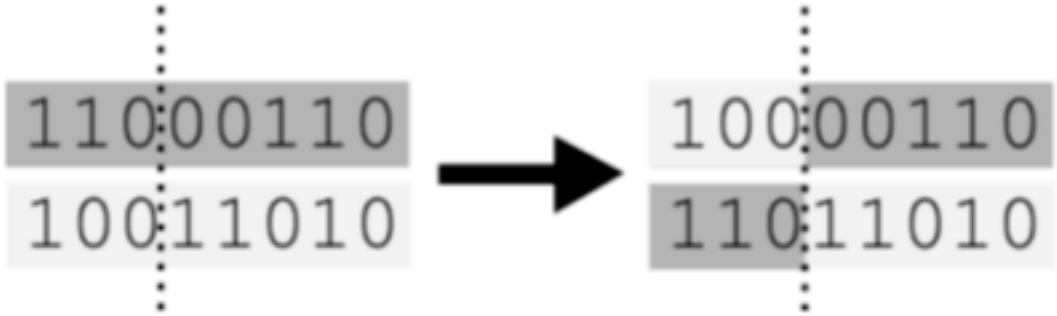

Figure 3.7: Single-point crossing [16]

- Double crossing (two points):

Figure 3.8: Crossing at two points [16]

3.4.3 The mutation operator

Mutation is the application of the principle of variation of Darwin's theory. Mutation is the change of a gene in a chromosome according to a mutation factor (probability). A mutation is carried out on an individual to avoid premature convergence of the algorithm towards a local extremum.

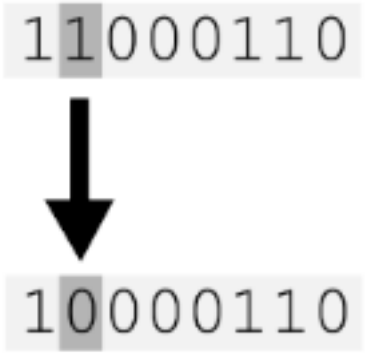

Figure 3.9: Representation of mutation operator [16]

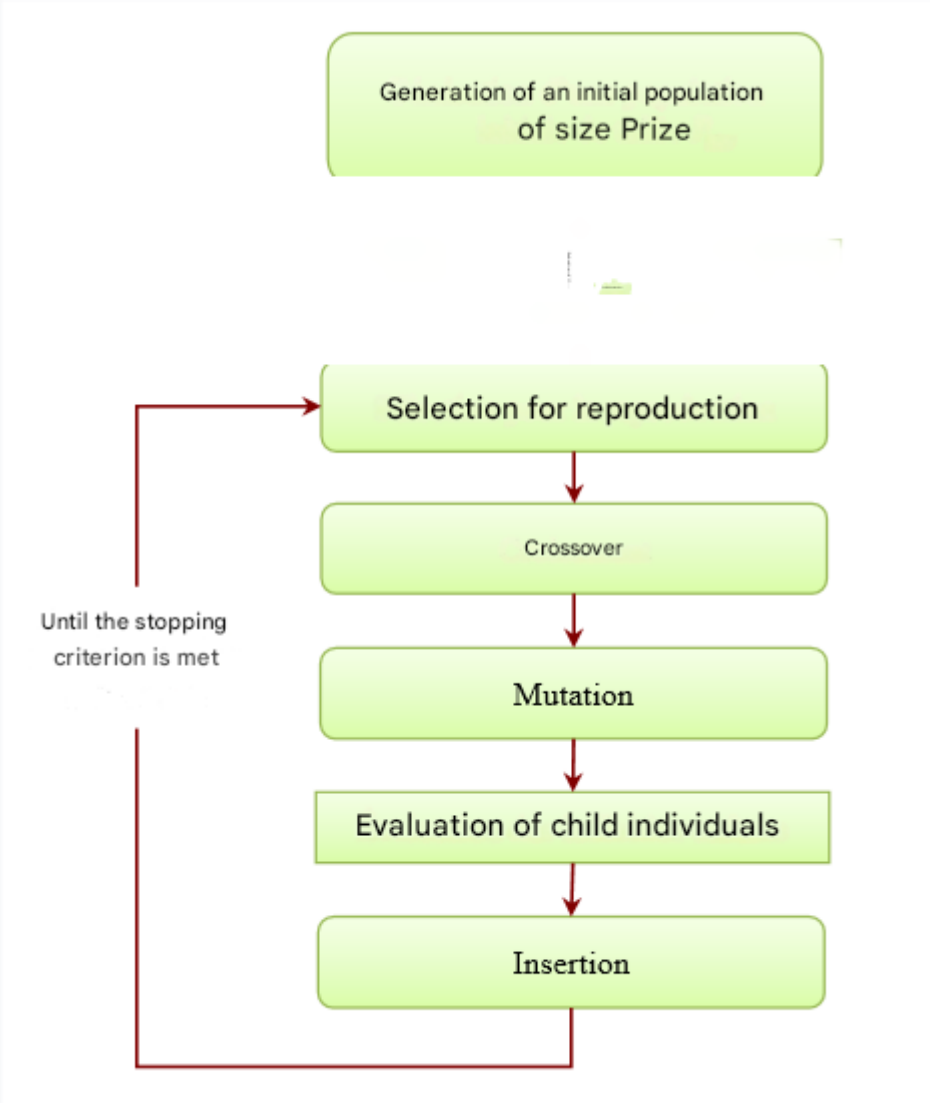

Figure 3.10: The general operation of genetic algorithms

Explanations:

- The start of the algorithm from population creation is random.
- After that we will do the evaluation, i.e., if a solution is available, for this we use ”fitness”, in order to define the adaptation score of the individuals during the selection process.
- All operators apply in loop; at the end, we choose the solution which has the best fitness.

3.5 Meta-heuristics

3.5.1 Simulated Annealing

Simulated Annealing (SA) is a meta-heuristic known to be the oldest. It was proposed by Kirkpatrick in 1983 based on the work of Metropolis, from which it draws its statistical origins. The term annealing is inspired by a process used in metallurgy in which metals are alternated between heating and cooling cycles to minimize the energy of the materials. SA uses a Monte Carlo approach to simulate the behavior of a system that tends to reach thermal equilibrium. It has been proven that by accurately monitoring the cooling rate, the algorithm is able to find the global optimum, although this would take an infinite amount of time. Faster versions like Fast Annealing and Very Fast Simulated Reannealing (VFSR) are used to overcome the delay problem. Simulated annealing has a serious advantage over other methods in that it does not get trapped in local minima [15].

Algorithm 2.1 — The RS algorithm

3.5.2 GRASP

GRASP (Greedy Randomized Adaptive Search Procedure) is a meta-heuristic that consists of a sequence of solutions constructed by a greedy approach and their optimization by exploring their respective neighborhoods [18]. Each iteration of this meta-heuristic consists of a solution construction phase and a neighborhood exploration phase. GRASP tries to take advantage of both the greedy approach and the random approach. GRASP keeps track of the best solution and returns it at the end of the algorithm [15].

Algorithm 2.2 — The GRASP algorithm

3.5.3 Searching with Taboos

Taboo search (TS), proposed in 1986 by Fred Glover [6], explores neighboring solutions and avoids returning to previously visited solutions by maintaining a taboo list.

Algorithm 2.3 — The TS algorithm

3.6 Genetic Algorithms for Customer Order Scheduling

In the complex landscape of manufacturing and logistics, customer order scheduling is a major challenge. Companies must juggle tight delivery times, limited resources

(machines, personnel, raw materials), and product diversity, while aiming to fully satisfy customers and optimize costs. It's not just about manufacturing products, but also about doing so at the right time, in the right order, and delivering them as promised. This problem, often referred to as NP-hard, means there is no quick and easy way to find the perfect solution, especially as the number of orders and constraints increases.

Faced with this complexity, genetic algorithms (GAs) emerge as a robust and powerful solution. Inspired by the process of natural evolution and the survival of the fittest, GAs are heuristic optimization techniques that explore a wide range of possibilities to unearth high-quality solutions. Rather than testing every conceivable combination, which is infeasible for most real-world problems, GAs work with a "population" of potential solutions, gradually improving them over "generations" through mechanisms such as selection, crossover (combining solutions), and mutation (small random changes).

Applied to customer order scheduling, GAs offer a flexible approach to modeling multiple constraints and conflicting objectives (such as minimizing delays while maximizing machine utilization). They enable efficient scheduling plans that meet customer requirements while optimizing the use of company resources. By leveraging the power of computational evolution, genetic algorithms represent a promising avenue for transforming how companies manage and optimize their production processes.

3.7 Application of Genetic Algorithms to Customer Order Scheduling via Job Shop Modeling

Customer order scheduling in a dynamic industrial environment is a complex problem combining capacity, heterogeneous resource, and lead-time constraints. To simulate this problem rigorously and optimize the performance of the production system, we use an approach inspired by the classic Job Shop problem, coupled with a stochastic optimization meta-heuristic: the genetic algorithm (GA).

3.7.1 Modeling by Job Shop

Each customer order is modeled as a job, composed of several operations corresponding to the different products or components to be manufactured. These operations must be executed in a partially defined order and on non-identical parallel resources (machines or servers). Since the environment is constrained by non-preemption, resource allocation must ensure that the operations of an order are completed before delivery is triggered.

So the problem comes down to a Job Shop scheduling with:

- Jobs = customer orders,
- Operations = types of products or treatments by position,
- Heterogeneous machines = servers or production equipment,
- An objective function = minimization of the overall cycle time.

Chromosome Structure:

In our implementation of GA, each individual (or candidate solution) is a chromosome composed of two parts:

1. MS (Machine Selection): a vector indicating which machine each operation is assigned to.

2. OS (Operation Sequence): a vector defining the order of execution of operations on each machine.

This dual coding allows the algorithm to capture both resource allocation decisions and scheduling priorities, thus providing maximum flexibility in finding the optimal solution.

Genetic Operators:

The evolution of the population is ensured by the following operators:

- Machine crossover: partial exchange of machine assignments between two selected parents, while maintaining the sequence of operations.

- Operation crossover: recombines the sequences of operations of two individuals, ensuring that the structure of the jobs is respected.

- Machine mutation: random but guided reassignment of operations to faster machines according to the processing time matrix.

- Operation mutation: local permutation of the order of operations of the same job, followed by a selection of the best configuration via makespan evaluation.

These operators are applied probabilistically, controlled by the GA parameters (Pc for crossover, Pm for mutation), and reinforced by a fitness-based selection strategy measured by total processing time (makespan) or multi-objective criteria including delays and costs.

Decoding and Evaluation

The decoding of a chromosome into a real schedule is done by a specialized module (Decode), which simulates the workshop behavior considering:

- Machine availability dates,
- Technological order of operations,
- Resource conflicts.

This decoder calculates the completion time of each operation, and therefore the delivery time of each order, allowing reliable and realistic evaluation of the solution.

Results and Interest

This approach enables the simulation of complex customer order arrival and processing scenarios, taking into account random variations in workloads, machine capacities, and customer-specific constraints. By integrating Job Shop methods into a scalable structure, GA becomes a powerful tool for dynamic and adaptive scheduling, capable of handling uncertain and multi-product environments.

The results show a significant improvement in average order cycle time, a reduction in bottlenecks, and better resource utilization compared to conventional heuristics. The algorithm is also able to adapt to parameter changes during simulation, making it a robust solution for real industrial systems.

3.8 Main mathematical formulas

This section presents the main mathematical formulas used in the application of genetic algorithms (GA) for solving a customer order scheduling problem, modeled as a Job Shop type problem.

3.8.1 Representation of the Chromosome

A chromosome C is composed of two vectors:
- MS (Machine Selection): machine assignment of each operation,
- OS (Operation Sequence): order of execution of operations.

Formally:

$$C = [\text{MS} \mid\mid \text{OS}] \in \mathbb{N}^{2T}, \quad \text{with } T = \text{total number of operations}$$

3.8.2 Fitness Function

Fitness corresponds to makespan (total processing time):

$$f(C) = \text{makespan}(C)$$

With:

$$C_{\max} = \max_j \{C_j\}, \quad \text{where } C_j \text{ is the end date of job } j$$

Objective:

$$\min f(C)$$

3.8.3 Machine Selection (Intelligent Mutation)

For an operation o_{ij} having several machines:

$$ms_{ij} = \arg\min_{k} \left\{ p_{ij}^{(k)} \mid p_{ij}^{(k)} \neq \infty \right\}$$

3.8.4 Partial Crossing

Partial exchange of machine assignments between two parent chromosomes:

$$\forall i \in R, \quad ms_1^{(i)}, ms_2^{(i)} = ms_2^{(i)}, ms_1^{(i)}$$

After crossover:

$$C_1 = [\text{ms}_1 \mid\mid \text{os}_1], \quad C_2 = [\text{ms}_2 \mid\mid \text{os}_2]$$

3.8.5 Operational Mutation (Local Permutation)

Let $\pi \in S_r$ (set of permutations of r operations):

$$\text{OS}_{\text{mutated}} = \pi(\text{OS}_{\text{original}})$$

Choose the best permutation:

$$\text{OS}_{\text{optimal}} = \arg\min_{\pi \in S_r} f([\text{MS} \mid\mid \pi(\text{OS})])$$

3.8.6 Decoding and Calculating Dates

For each operation o_{ij}:

$$S_{ij} = \max(C_{i(j-1)}, C_{m(o_{ij})}^{\text{prev}})$$

$$C_{ij} = S_{ij} + p_{ij}^{(ms_{ij})}$$

3.8.7 Stopping Criterion

The algorithm stops when:

$$\text{Iterations} \geq \text{MaxIterations} \quad \text{or} \quad f(C_{\text{best}}) \leq \varepsilon$$

3.9 Conclusion

This chapter introduced the foundations of genetic algorithms, exploring their key principles: variation, adaptation, and inheritance. We analyzed the essential operators of these algorithms, namely selection, crossover, and mutation. Finally, we studied genetic algorithms through some of their models, highlighting their mechanisms and applications.

Chapter 4

Evaluation and Experimentation

4.1 Introduction

This work presents our project focused on solving the job shop scheduling problem using genetic algorithms. We begin with a description of the hardware environment used for development, followed by a presentation of the software used. Finally, we detail the tests performed on our program by varying different parameters in order to evaluate its performance and optimize the results.

4.2 Hardware Environment

We used a DELL computer as a hardware environment with the following characteristics:

- Processor: Intel® Core i5-4005U CPU @ 1.70GHz
- RAM: 6.00 GB
- System Type: 64-bit operating system

4.3 Software Environment

Platform used is Windows 10 with the help of the chosen development language is Python.

4.3.1 The Python programming language

Python is a high-level, interpreted, object-oriented, and general-purpose computer programming language. It is known for its clear and easy-to-read syntax, making it popular for learning and developing a variety of projects, including scientific computing, web development, data science, and artificial intelligence.

4.3.2 The PyCharm environment

PyCharm is an integrated development environment used for programming in Python. It allows code analysis and includes a graphical debugger. It also allows unit testing, version control software integration, and supports web development with Django.

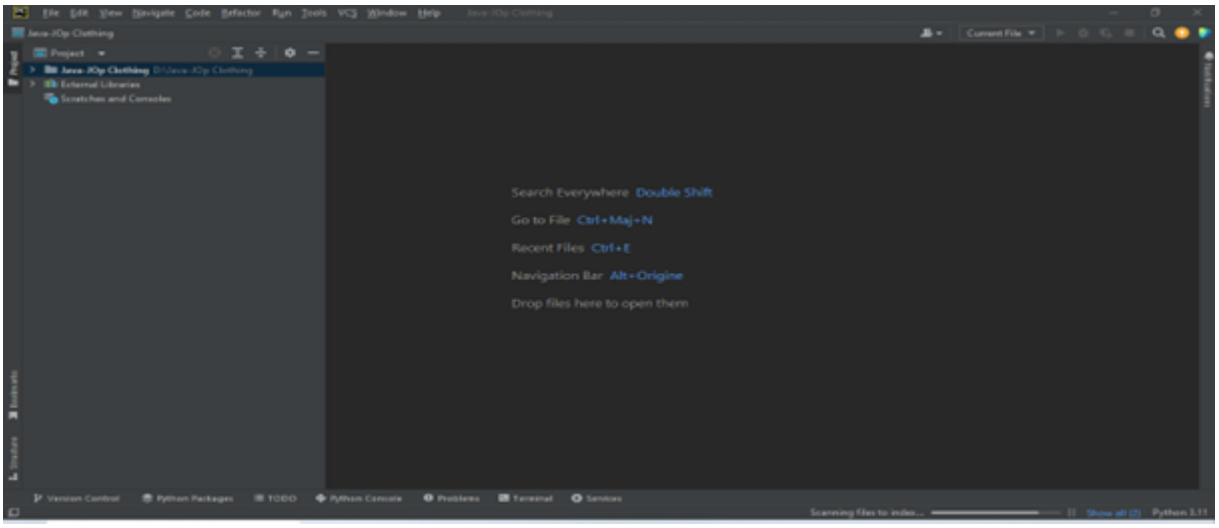

Figure 4.1: PyCharm

4.4 Flexible Workshop Planning Problem with Genetic Algorithm

This project uses a genetic algorithm to solve the job shop scheduling problem.

4.4.1 Create an Instance.py file based on real problems

For example, here is the treatment schedule for each part:

	M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11
Op1	10	0	0	0	0	0	0	0	0	0	0
Op2	0	9	0	0	0	0	0	0	0	0	0
Op3	0	0	14	16	0	0	0	0	0	0	0
Op4	0	0	0	0	15	25	21	0	0	0	0
Op5	0	0	0	0	0	0	0	9	13	25	14

If the number of parts is 5, the `Instance` will be written in the following format:

```
Processing_time=[[10, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999],
                 [9999, 9, 9999, 9999, 9999, 9999, 9999, 9999, 9999, 9999],
                 [9999, 9999, 14, 16, 9999, 9999, 9999, 9999, 9999, 9999],
                 [9999, 9999, 9999, 9999, 15, 25, 21, 9999, 9999, 9999, 9999],
                 [9999, 9999, 9999, 9999, 9999, 9999, 9, 13, 25, 24]]

J={1:5, 2:5, 3:5, 4:5, 5:5}

J_num=5

M_num=11

O_num=25
```

Variable names explanation: - `Processing_time`: The processing schedule for each part, in list format. In the table, the row index represents the sequence number of the operation, the column index that of the machine and the numeric value represents the corresponding processing time. If no machine is selected for the operation, the corresponding value is 9999.

- `J`: The index of each part and the total number of corresponding operations, in dictionary format.
- `J_num`: The number of pieces.
- `M_num`: The number of machines.
- `O_num`: The number of operations for all parts.

4.4.2 Configure the genetic algorithm

Setting up the genetic algorithm in `GA`:

Parameter	Value
Population size (Pop_size)	400
Crossover probability (Pc)	0.8
Mutation probability (Pm)	0.3
Variation selection probability (Pv)	0.5
Mutation method probability (Pw)	0.95
Maximum iterations (Max Iteration)	100

Variable names explanation: - `Pop_size`: The size of the population.
- `Pc`: The probability of performing the crossover operation.
- `Pm`: The probability of performing the mutation operation.
- `Pv`: The probability of choosing the crossover method.

- `Pw`: The probability of choosing the mutation method.
- `Max_Iteration`: The maximum number of evolutionary generations.

After running the code, the following two results are displayed.

Result 1: Scheduling the processing of all parts displayed in the Gantt chart

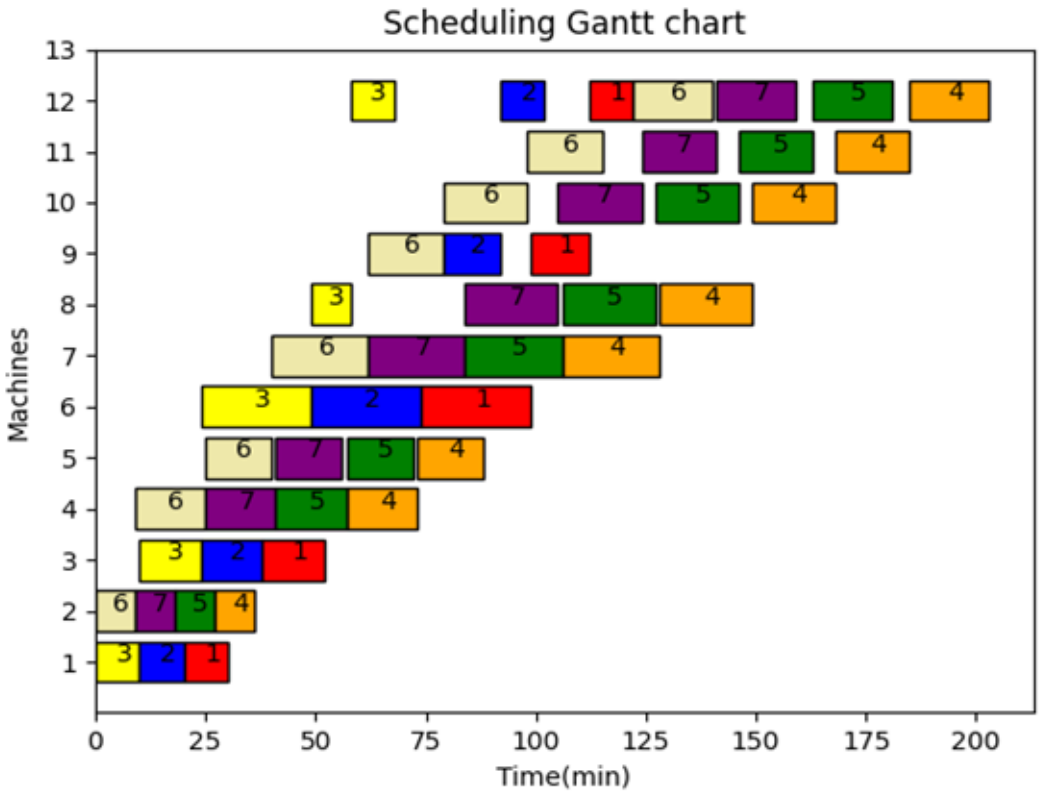

Figure 4.2: The Gantt chart

Result 2: The maximum completion time of each iteration

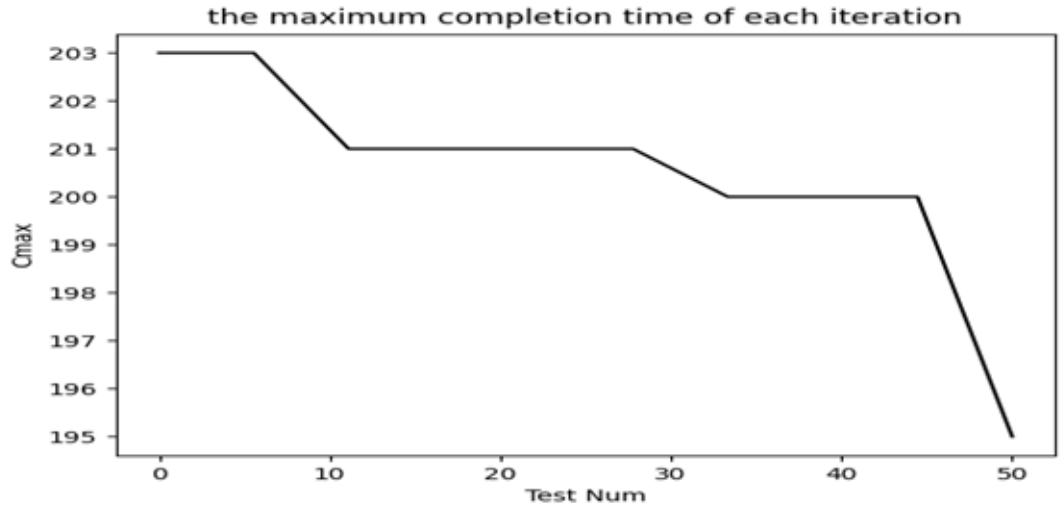

Figure 4.3: Makespan graph

- The Gantt chart gives a detailed view of the optimal production plan found by the algorithm (actual distribution of operations).

- The convergence curve demonstrates the effectiveness of the genetic algorithm, showing how the solution improves with each iteration.

Together, these two graphs allow:

- To visually validate the solution obtained,

- To assess the quality of the generated schedule,

- And to appreciate the performance of the approach based on genetic algorithms to solve the JSP problem.

4.5 Conclusion

This diagram provides a clear visualization of the time distribution of operations on the different machines, highlighting avoided conflicts and good synchronization of tasks. It facilitates understanding of the optimized production flow.

General Conclusion

The main objective of this work was to solve the complex job shop scheduling problem, classified among combinatorial problems, by exploiting genetic algorithms. To do this, we developed a dedicated application based on these algorithms.

The first chapter discussed customer order management, highlighting its strategic importance. The second chapter explored the general concepts of scheduling and its various models. In the third chapter, we analyzed genetic algorithms, their fundamental principles, and operational mechanisms. The fourth chapter detailed our approach to solving the job shop problem using genetic algorithms, presenting our application, its parameters, and explaining its use.

Customer order scheduling plays a key role in improving customer satisfaction, optimizing costs, and ensuring business competitiveness. Effective management relies on appropriate tools, robust methods, and the ability to adapt to dynamic production environments. This study highlights the effectiveness of genetic algorithms in addressing these challenges.

Perspective

The perspectives of this thesis lie in the continuous improvement of the application

developed, notably by exploring hybrid genetic algorithms for a better performance and

integrating multi-objective criteria (costs, delivery times) for a more global optimization of the scheduling. It would also be relevant to study the application of this approach to other complex scheduling problems or to more dynamic production environments, for example by considering the vagaries of production.

Bibliographies

1 AS Jain and S. Meeran, Deterministic job-shop scheduling: past, present and future, European Journal of Operational Research, vol. 113, p. 390-434, 1999.

2 B. Giffler and G. L. Thompson, Algorithms for solving production scheduling problems, Operations Research, (1960) 8(4), 487-503.

3 B. Roy, Modern Algebra and Graph Theory, volume 2, Editions Dunod, Paris, 1970.

4 B. Roy and D. Bouysson, Multi-criteria decision support: Methods and cases. Management Collection Series: Quantitative production and technology applied to management. Economica Edition, Paris, 1993.

5 D. Lamy, Methods and tools for scheduling workshops taking into account additional constraints: energy and environmental, Doctoral Thesis, University of Clermont Auvergne, France, 2017.

6 F. Glover, Future paths for integer programming and artificial intelligence, Computers Operations Research 13 (1986), 533-549.

7 G. Bel and J.B. Cavaillé, Production Scheduling, Éditions Hermès, Paris, 2001.

8 Gotha, Scheduling problems. RAIRO-Operations Research, vol. 27, no. 1, pp. 77-150, 1993.

9 H. Boukef, On the scheduling of flexible job-shop and flow-shop workshops in the pharmaceutical industry: optimization by genetic algorithms and particle swarms, Doctoral Thesis, University of Tunis El Manar, Tunis, 2009.

10 J. Carlier and P. Chrétienne, Scheduling Problems, Edition Masson, Paris, 1988.

11 J. Carlier, P. Chrétienne and C. Girault, Modeling and scheduling problems with Petri nets. Advanced studies in Petri nets. Lecture notes in Computer Science, Springer Verlag, Paris, 1984.

12 J. Kaabi-Harrath, Contribution to the scheduling of maintenance activities in production systems, Doctoral Thesis, University of Franche-Comté, France, 2004.

13 M. Larabi, The job-shop problem with transport: modeling and optimization, Doctoral Thesis, Blaise Pascal University - Clermont-Ferrand II, French, 2010.

14 M. Meziane, Proposal of a scheduling approach for Flexible Job Shop type workshops, Doctoral Thesis, University of Oran, Algeria, 2018.

15 M. Yildizoglu, Presentation of genetic algorithms and their applications in economics, Article, University of Bordeaux, February 2004.

16 N. Liouane, Contribution to the development of a decision-making tool for production scheduling in an uncertain environment, Doctoral thesis, Lillel University of Sciences and Technologies, Lille, 1998.

17 T. Feo and M. Resende, Greedy Randomized Adaptive Search Procedures, Journal of Global Optimization 6, 2 (1995), 109-133.

18 T. Nicholson, Optimization in industry. Editions Longmann Press, London, 1971.

Websites

a https://fr.wikipedia.org/wiki/Ordonnancement _d%27atelier.

b http://igm.univ-mlv.fr/~dr/XPOSE2013/tleroux_genetic_algorithm/fonctionnement.html.

c https://www.researchgate.net/profile/Tianhua _Jiang/publication/figure/gantt-chart-of-problem-solution.

ملخص

في بيئات الإنتاج من نوع Job Shop أمرًا بالغ الأهمية .يتحول كل طلب عميل إلى "مهمة "أو "عدة مهام "مميزة، تتطلب سلسلة من العمليات المتتالية مختلفة .العملاء هذه في أقصر وقت ممكن وبأكثر الطرق فعالية.
إن استخدام خوارزمية جينية في بيئة Job Shop تقليل وقت دورة متطلبات العملاء المعقدة إلى خطط إنتاج قابلة للتحقيق وفعالة.

Abstract

In production environments such as Job Shop manufacturing, like the production of servers, managing customer orders is crucial. Each customer order translates into one or more distinct jobs or tasks, requiring a series of sequential operations on different machines or workstations. The objective is to produce these customer orders in the shortest possible time and in the most efficient way.

The use of a genetic algorithm in a Job Shop environment is an effective method to optimize the management and execution of customer orders by minimizing production cycle time, thereby transforming complex customer requirements into feasible and efficient production plans.