Optimization Projects assignments

Group members	Group ID	Project ID	Assigned metaheuristic	Presentation day
• Maab Chaoui • Mohamed Benabdelwahad • Djalal Hacisni • Sohaib Mousselmal • Meriem Mebarek Mansouri	**A3**	**P1**	**Simulated Annealing SA**	**Thursday, May, 16th 16/05/2024**
• Bouchra Manar BENKEHIL • Kaouthar HANI • Kawther GUOUAL BELHAMIDI	**A2**	**P1**	**Ant Colony Optimization ACO**	**Wednesday, May, 15th 15/05/2024**
• Salamani Amine • Zitouni Manel • Chellal Abdelhak • Zenakhri Aicha	**A6**	**P2**	**Variable Neighborhood Search VNS**	**Thursday, May, 16th 16/05/2024**
• Djaid Douaa • Afra Hana • Cheboui Fatma Imene • Bakhouche Rachel • Kara Laouar Manel	**A5**	**P1**	**Ant Colony Optimization ACO**	**Sunday, May, 19th 19/05/2024**
• Zerrouki Fella Manel • Ouari Meriem • Litim Chiraz • Belhadj Sara • Korichi Anfal	**B5**	**P2**	**Tabu Search TS**	**Sunday, May, 19th 19/05/2024**
• Mohammed Chaker Baaziz • Mouhcen Tahar Chouireb • Haroune Rezki • Ishak Abbassi • Bouzid Ahmed	**A1**	**P2**	**Ant Colony Optimization ACO**	**Sunday, May, 19th 19/05/2024**
• HAMMAD Zineb Salima • ZEMALI Mohammed Anis • BOUZIANE Abdenour • BAZOUZI Maha • BENCHOUFI Isra	**C5**	**P1**	**Variable Neighborhood Search VNS**	**Sunday, May, 19th 19/05/2024**
• soumia bouaouina • soumia bouyahiaoui • farouk omar zouak • larbi saidchikh • mani selma	**A4**	**P1**	**Genetic Algorithm GA**	**Wednesday, May, 22nd 22/05/2024**
• Billal Chaouche • Mohmoud Badlis • Oumaima Daif • Abd-eldjalil Taibi • Hamza Benzaoui	**B2**	**P2**	**Tabu Search TS**	**Wednesday, May, 15th 15/05/2024**

• Amrouche Faycal • Rezki Abderahim • Boukhalfa Houssem Eddine • Korichi Ayoub • Rahmani Anis	**B6**	**P2**	**Genetic Algorithm GA**	**Thursday, May, 16th 16/05/2024**
• AMMAR KHODJA Lilia • RAHMOUNI Rahil • MIRA Bouchra • BENDIF Besmala • ACHOURI Anfel	**B4**	**P2**	**Simulated Annealing SA**	**Wednesday, May, 22nd 22/05/2024**
• HOCINE Nihal • AIT ABDELLAH Imene • BOUALLEM Lina • KEDDACHE mouhanned	**B3**	**P2**	**Genetic Algorithm GA**	**Thursday, May, 16th 16/05/2024**
• Khelif Selma • Zetili Zineb • Arab Sarra • Lalagui Baraa Fatima Zohra	**C2**	**P1**	**Tabu Search TS**	**Wednesday, May, 15th 15/05/2024**
• Oumaima Maatar • Rym Selmani • Djamila Amani Hamza • Hiba Ilmain	**B1**	**P1**	**Variable Neighborhood Search VNS**	**Sunday, May, 19th 19/05/2024**
• Boubekeur Farida • Medjri Yasmina • Guendouz Nour-el-Yakine • Mohamed Nadir Hamou • Belarbi Abdenour	**C6**	**P1**	**Ant Colony Optimization ACO**	**Thursday, May, 16th 16/05/2024**
• Amira Boudaoud • Yasmine kaced • Sarah Mahmoudi • Nesrine Abdelhak • Manel Merabet	**D2**	**P2**	**Genetic Algorithm GA**	**Wednesday, May, 22nd 22/05/2024**
• AOUAMRI Farah • TEHARI Thouria • LABAR lina Nadjeh • Guettab thilelli • Lazizi selma	**C1**	**P2**	**Variable Neighborhood Search VNS**	**Sunday, May, 19th 19/05/2024**
• Ikhlef Lyna • Bensedira Kaouther	**D6**	**P1**	**Simulated Annealing SA**	**Sunday, May, 19th 19/05/2024**
• Ibrahim El Khalil Attia • Sohaib Houhou • Keddouri Faid	**D5**	**P2**	**Simulated Annealing SA**	**Thursday, May, 23rd 23/05/2024**
• Nassima OUKALI • Sabrine CHAHED • Wafa BOUCHIBANE	**D1**	**P1**	**Ant Colony Optimization ACO**	**Thursday, May, 23rd 23/05/2024**

• Siniane Mira Thiziri • Bouderka Maroua • Farez Samah Ikram	**C3**	**P1**	**Genetic Algorithm GA**	**Thursday, May, 16th** **16/05/2024**
• Bendi Mohamed abderraouf • Fadoua chourouk Djekaoua • Maachou Mohammed Imad	**E6**	**P2**	**Tabu Search TS**	**Sunday, May, 19th** **19/05/2024**
• Makdour Salah Eddine • hassici Rayan Zakaria • Yagoub Douaa Manel • Mers Wafaa • Oulad Ali Merouane	**D3**	**P2**	**Variable Neigborhood Search VNS**	**Wednesday, May, 22nd** **22/05/2024**
• Daoud Anaïs • Baghdadi Kamar HibatAllah	**E3**	**P1**	**Ant Colony Optimization ACO**	**Wednesday, May, 22nd** **22/05/2024**
• khezzane dina • lina amdirt • hafida boukedjar	**C4**	**P2**	**Genetic Algorithm GA**	**Wednesday, May, 22nd** **22/05/2024**
• Daoudi Loukmane • Benkhedda Mohamed Serradj Eddine • Slimani Yazid • Ouahioune Raid Abderrezak • Kadri Mohammed Mouncef	**D4**	**P2**	**Ant Colony Optimization ACO**	**Thursday, May, 23rd** **23/05/2024**
• Kermache Adlane • Nadjib Bentayeb • Djelmami Brahim	**E1**	**P1**	**Genetic Algorithm GA**	**Thursday, May, 23rd** **23/05/2024**
• Hamaidi Mohamed Idriss • Yousfi chaker • Saidoun Seif Allah • Abdoun rayan • Atamna Alaeddine	**X1**	**P2**	**SA**	**Thursday, May, 23rd** **23/05/2024**
• Bouazzouni Mohamed Amine • Boukeffa zakaria	**X2**	**P1**	**VNS**	**Thursday, May, 23rd** **23/05/2024**

Projects

Project 1 : The Flow Shop Scheduling Problem (FSSP)

Project ID : P1

Problem definition:

The Flow Shop Scheduling Problem (FSSP), appearing in both permutation and non-permutation forms, is a prominent challenge in operations research. It serves as a vital tool for modeling manufacturing and production planning scenarios, both theoretical and practical. In essence, the problem entails scheduling a set of jobs on a series of machines. The goal is to establish a production timetable where each job undergoes processing on each machine in a predetermined order, typically dictated by the technological process. The primary objective is to minimize the total completion time of all jobs. In the permutation variant, jobs are processed on each machine in a fixed sequence, whereas the non-permutation variant lacks such constraints.

Constraints :

Constraint (1) : The processing of operations cannot be interrupted.

Constraint (2) : The minimum and maximum machine idle times are respected, also the operation ***a*** on machine *i* is only started after previous operation on *i* (i.e., machine predecessor of *a*) has completed.

Constraint (3) : That operation ***a*** of job *j* is started only after previous operation of job ***j*** (i.e., the technological predecessor of *a*) has completed.

Let $J = \{1,2, \ldots , n\}$ and $M = \{1,2, \ldots , m\}$ be sets of *n* jobs and *m* machines, respectively. Each job *j* is composed of *m* operations from the set $Oj = \{lj + 1, lj + 2, \ldots , lj + m\}$, where $lj = (j - 1)$.

Each job *j* has to be processed on the machines in the natural order: $1 \to 2 \to \cdots \to m$.
The processing time of job *j* on machine *i* (i.e., operation $lj + i$) is denoted $pi, j > 0$. Moreover, let $r^{\wedge}i \geq 0$ denote the minimum idle time machine *i* is allowed between the processing of subsequent operations. Similarly, $d^{\wedge}i \geq r^{\wedge}i$ denotes the maximum idle time.

Decision Variable: While the order in which jobs are processed is fixed, **the processing order** for each machine is a decision variable.

The schedule (S, C) consists of two matrices. The first matrix is given by $S = [Si, j]\times n$, where Si, j is the starting time of the *j*-th operation to be processed on machine *i* in. Similarly, we can define the matrix of operation completion times $C = [Ci, j]\times n$, where element Ci, j is the completion time of *j*-th operation to be processed on machine *i*.

Problem instance: *Consider the* ***FSSP*** *instance from* ***Table 1*** *for* $n = 5$, $m = 3$

Table 1. Problem instance from Example 1 for $n = 5$ and $m = 3$.

a	$p_{a,1}$	$p_{a,2}$	$p_{a,3}$	$p_{a,4}$	$p_{a,5}$	$\widehat{r}_a$	$\widehat{d}_a$
1	2	1	2	1	3	1	5
2	1	2	1	2	1	1	2
3	2	2	3	2	1	0	0

1) Give the general mathematical formulation of the problem (do not use the instance).
2) Study the complexity of the problem.
3) Give a solution using Gantt diagram for the problem instance in **Table 1**.
4) Use the assigned metaheuristic to solve the problem instance in **Table 1**.
5) Study the results of the proposed resolution method.

Project 2 : The Capacitated Vehicle Routing Problem (CVRP)

Project ID : P2

Problem definition:

The Vehicle Routing Problem (VRP) involves identifying the best routes for a fleet of vehicles to cater to a specified set of customers, making it a highly significant and extensively researched combinatorial optimization issue. It is recognized as NP-hard, implying its complexity. An extension of the classic Traveling Salesman Problem (TSP), the Capacitated Vehicle Routing Problem (CVRP) requires finding a Hamiltonian circuit that visits a designated set of points exactly once with the lowest possible cost.

The traditional Vehicle Routing Problem (VRP), often referred to as the Capacitated VRP (CVRP), aims to create efficient delivery routes where each vehicle undertakes only one route, all vehicles share identical attributes, and there exists a single central depot. The objective of the VRP is to determine a collection of low-cost vehicle routes ensuring that every customer is visited precisely once by one vehicle, each vehicle initiates and concludes its journey at the depot, and the vehicles' capacity constraints are not violated.

The VRP is described by a set of homogenous vehicles (denoted by), a set of customers C and a directed graph G. The graph consists of $|\boldsymbol{C}| + \mathbf{2}$ vertices where the customers are denoted $\mathbf{1}$, $\mathbf{2}$, . . . , $\boldsymbol{n}$ and the depot is represented by the vertex **0** and **n+1**. The set of vertices $\mathbf{0}, \mathbf{1}, \ldots, \boldsymbol{n} + \mathbf{1}$ is denoted as N. The set of arcs (denoted by A) represents connections between the depot and the customers and among the customers. No arc terminates at vertex 0 and no arc originates from vertex n+1. With each arc (i,j), where $\boldsymbol{i} \neq \boldsymbol{j}$, we associate a cost (distance) Cij Each vehicle has a capacity **q** and each customer **i** has a demand di. It is assumed that $\boldsymbol{q}, \boldsymbol{d}$, $\boldsymbol{cij}$ are nonnegative integers.

Constraints :

Constraint (1) : Each customer is visited exactly once

Constraint (2) : No vehicle is loaded more than its capacity allows it to.

Constraint (3) : Each vehicle leaves the depot 0, after arriving at a customer the vehicle leaves that customer again and finally arrives at the depot n+1.

Decision variable:

For each arc $(i, j), i \neq j, i \neq n+1, j \neq 0$ and each vehicle k, $x_{ijk}$is defined as :

$$x_{ijk} = \begin{cases} 1 \; if \; vehicul \; k \; is \; using \; arc \; j \\ 0 \qquad\qquad\qquad Otherwise \end{cases}$$

We want to design a set of minimal cost routes, one for each vehicle, such that each customer is serviced exactly once and every route originates at vertex 0 and ends at vertex $n+1$.

Problem instance : Consider **11 nodes** (customers) where node **1** is a depot and **m=6 vehicles**. Capacity of each vehicle is **q=100.** The requirements of the nodes are **d = (0, 5, 20, 10, 20, 85, 65, 30, 20, 70, 30)**. The distance **(cost)** matrix **C** is in the figure below.

0	13	6	55	93	164	166	168	169	241	212
13	0	11	66	261	175	177	179	180	239	208
6	11	0	60	97	168	171	173	174	239	209
55	66	60	0	82	113	115	117	117	295	265
93	261	97	82	0	113	115	117	118	333	302
164	175	168	113	113	0	6	7	2	403	374
166	177	171	115	115	6	0	8	7	406	376
168	179	173	117	117	4	8	0	3	408	378
169	180	174	117	118	3	7	3	0	409	379
241	239	239	295	333	403	406	408	409	0	46
212	208	209	265	302	374	376	378	379	46	0

1) Give the general mathematical formulation of the problem (do not use the instance).
2) Study the complexity of the problem.
3) Give a model of the problem using Graph theory.
4) Use the assigned metaheuristic to solve the problem instance defined previously.
5) Study the results of the proposed resolution method.