Order number: . . . . .

Thesis submitted to the

UNIVERSITY OF MOHAMED BOUDIAF – MSILA

FACULTY OF MATHMATICS AND COMPUTER SCIENCE

DEPARTEMENT OF COMPUTER SCIENCE

In partial fulfillment of the requirements for the degree of

Master in ***Computer science***

Option : Informatique Décisionnelle et optimisation (IDO)

By

Boukraa Samah

Title of the thesis

Simulated annealing algorithm to solve Aircraft Landing Problem

Defended on September 26, 2023 behind the jury composed of:

Dr. Bentercia Rahima	MB University of M'sila	President
Dr. Hemmak Allaoua	MB University of M'sila	Supervisor
Dr. Bougherara Seddik	MB University of M'sila	Examiner

Academic year 2022-2023

DEDICATIONS

إلى نفسي

إلى من زينت حياتي بضيائها، إلى من منحتني القوة والعزيمة لمواصلة الدرب، إلى من علمتني الصبر والاجتهاد إلى الغالية على قلبي أمي أدامك الله تاجا فوق رؤوسنا.

إلى من أحمل اسمه، وبهمته كونت نفسي وبفضله تعلمت كتابة هذه الحروف، إلى من كان ولا يزال سندي في الحياة، أبي الغالي.

إلى من ترك بصمة في حياتي، وساندني في كل الأوقات، إلى من وقف ويقف بجانبي في أصعب الظروف، إلى أخي وسندي.

إلى سندي و قوتي ،أماني و مأمني ،فرحتي وسعادتي إلى شريك حياتي زوجي

إلى ريحانة قلبي، إلى شمعة حياتي أختي

إلى صديقتي العزيزة

وكل من سعتهم ذاكرتي ولم تسعهم مذكرتي...

ACKNOWLEDGMENTS

I'am so grateful to **Dr. Allaoua Hemmak**, my supervisor for their exceptional guidance, and constant encouragement throughout the entire duration of this research and the writing process of this thesis.

Additionally, I would like to express my appreciation to the members of the jury **Dr. Bentercia Rahima** and **Dr. Bougherara Seddik**, for their time and effort in carefully reviewing and evaluating my thesis.

I would like to express my gratitude to all the participants who generously shared their time and experiences by participating in our surveys. Their valuable contributions have provided us with valuable data and insights that have greatly enriched our study.

I would like to acknowledge and express my appreciation to everyone mentioned above for their contributions and support, which have been indispensable in the successful completion of this thesis.

Contents :

General Introduction

Aviation is a modern mode of transportation where this technology facilitated the difficulty of travel which in the past was a hardship for many but like other modes of transport is not without the problem of overcrowding.

Overcrowded air traffic is a phenomenon that occurs when the number of flights increases significantly at the same time and in the same place. Overcrowding can be caused by many factors, including increased demand for air travel, lack of airport and air road infrastructure, and the presence of a large number of aircraft in the air at the same time. Overcrowding delays trips, high accident rate and sometimes low safety level. It also causes inconvenience for travelers and makes the travel process more complicated and stressful.

Congestion is not the only problem in aviation. The ALP (Aircraft Landing Problem) is challenging to manage the arrival and departure of aircraft at the airport efficiently, while ensuring safety and reducing delays. This problem becomes particularly critical in busy airports. The objective is to improve the use of available resources such as runways, gates and air traffic control systems to ensure the smooth flow of aircraft operations. This includes determining the sequence in which aircraft should be allowed to land or take off, taking into account factors such as aircraft type, size and performance, as well as weather conditions and air traffic restrictions. Several factors must be taken into account when solving the problem of aircraft landing, including the time of arrival and departure of each aircraft, the preferred runway and gate allocation, and available airspace resources. The goal is to find an optimal solution that reduces overall delays and increases the efficiency of airport operations. The field of study of aircraft landing is complex and can be looked forward to in many respects. We can mention the reduction of landing time, the reduction of the time spent in the sky, the selection of the optimal runway for landing and the choice of the nearest landing runway this is what our research aims to do.

The aim of our research is to identify the problem of aircraft landing, select a part of the problem and try to identify available ways to improve and solve this problem. Our graduation project consists of 4 chapters focusing on one aspect of aviation and its problems and details of these chapters as follows: Chapter 1 provides a comprehensive overview of aviation and identification

of the problem of aircraft landing, identifying combinatorial optimization and related algorithms. Chapter 2 of this chapter identifies the work related to this problem and the research provided in this area from the time of its emergence and the beginning of studies on it. Chapter 3 of this chapter addresses the proposed algorithm to solve the problem posed by the simulated annealing algorithm defined and general information about it. In his chapter, we propose a simple algorithm to help solve ALP. In chapter 4 and the last chapter, the language used for programming the previously proposed algorithm and the results obtained were identified.

By exploring basic concepts, specific techniques and practical implementation to try to solve part of the problem of aircraft landing, our thesis aims to contribute to progress in this field.

Chapter 1: Aircraft Landing Problem

Introduction

As civilian air transport grows, many aircraft need to land within a short time. Due to a lack of aviation resources, flight delays result in reduced efficiency, economic losses and other negative impacts. For this reason, researchers in the field of scheduling improvement have invested a lot of time and effort in devising flexible and robust scheduling methods that can effectively schedule aircraft departure and landing. The problem of setting the best schedule for aircraft seeking to land at the airport is called Aircraft Landing Problem (ALP).

1. Flight

Flight is the process by which an object, usually an aircraft, moves through the air. It involves generating sufficient lifting to counter the gravitational force and push the body forward. Flying can be achieved through different routes, including using wings and wind currents or through motor propulsion. Aircraft and birds are examples of objects capable of flying. Aviation has several stages

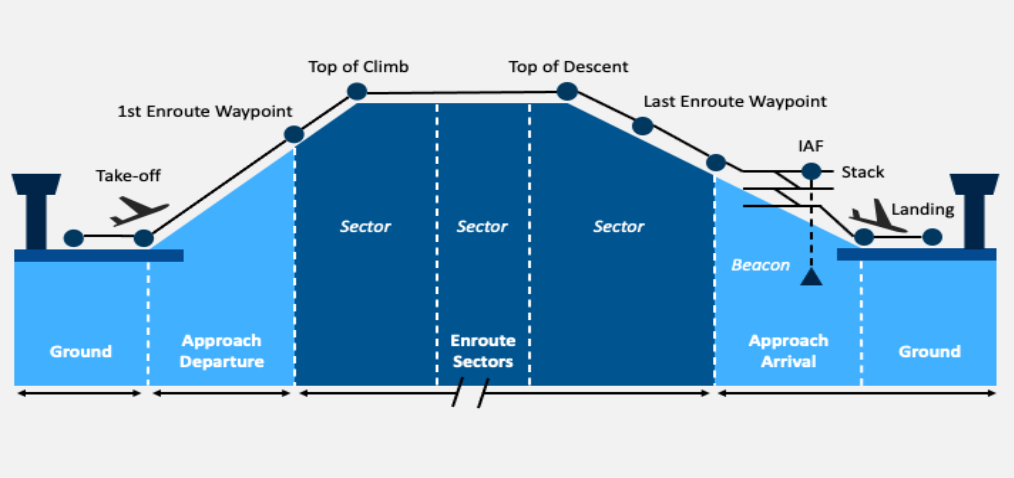

Figure 1 : Flight phases

There are usually four stages for flight**:**

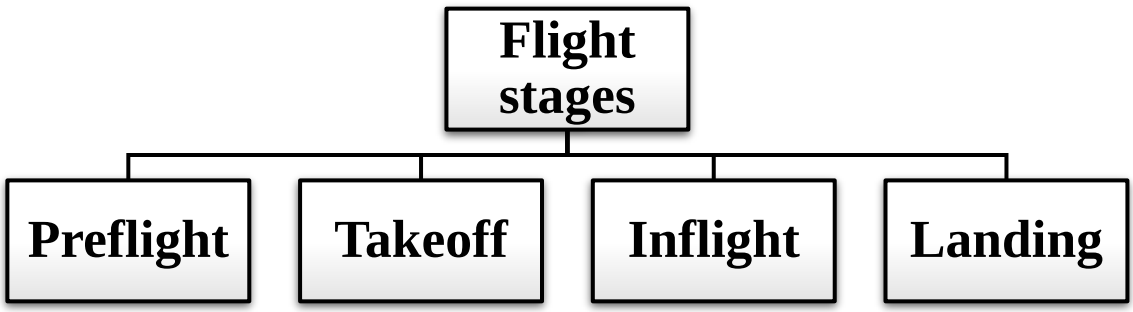

Figure 2: The most famous flight stages

- **Preflight:** This phase includes all of the preparations for the trip, including boarding the plane, receiving boarding permits...
- **Takeoff:** The aircraft lifts off from the runway once all passengers have boarded and the requisite preflight inspections have been performed.
- **In-flight:** During this part of the flight, the aircraft is traveling at the target height.
- **Landing**: During this stage, the aircraft descends, prepares for landing, and touches down on the runway.

2. Aircraft Landing (AL)

Aircraft landing, also known as touch-down, is the process by which an aircraft gradually descends and touches the ground after completing a flight. The landing is typically carried out on a designated runway at an airport.Before the actual landing, the pilot goes through a series of procedures to prepare for the approach. This includes contacting air traffic control, receiving landing clearances, and configuring the aircraft for landing. The pilot determines the approach path, speed, and descent rate based on factors such as weather conditions, aircraft type, and air traffic.As

the aircraft approaches the runway, the pilot maneuvers it to align with the centerline using control inputs such as ailerons, elevators, and rudder. The landing gear is extended, and the aircraft's speed is gradually reduced to a safe touchdown speed. During the final moments of landing, the pilot aims to achieve a gentle touchdown by reducing the descent rate. The aircraft's main landing gear makes initial contact with the runway, followed by the nose or tail gear, depending on the aircraft's configuration. The pilot then applies thrust reversers or uses brakes to slow down the aircraft and bring it to a complete stop.Aircraft landings require precise control and coordination to ensure a safe and smooth landing. Pilots undergo extensive training to master the skills necessary for accurate landings in various conditions, including crosswinds, low visibility, and challenging terrains. The landing phase is considered one of the critical stages of flight and requires utmost attention and skill from the flight crew

3. Factors Affecting Aircraft Landing

There are factors that can impact the landing of an aircraft, including;

- **Weather conditions:** The weather elements, like winds turbulence, fog, rain or snow can make it challenging for an aircraft to land safely. Limited visibility, wind patterns and severe weather conditions may require the pilot to divert to an airport.
- **Runway conditions:** The state of the runway can affect how an aircraft land. If the runway is wet icy or covered in snow it can reduce braking performance. Increase the risk of skidding or hydroplanning.
- **Weight and balance of the aircraft:** The weight and balance of an aircraft are crucial for a landing. If the aircraft is too heavy it may need a runway for landing. Additionally, how weight is distributed within the aircraft affects stability during landing.
- **Aircraft type and performance:** Different types of aircraft have characteristics that influence their landing capabilities. Factors like size, speed, wing configuration and approach characteristics all contribute to the landing process.
- **Proficiency of flight crew:** The expertise and experience of the flight crew play a role, in ensuring a landing. Their capacity to accurately evaluate the conditions upon landing make judgments. Carry out the landing procedure can significantly impact the end result.

- **Air traffic control:** The coordination and communication, with air traffic control can also have an impact on the landing process. Pilots must adhere to instructions provided by air traffic control regarding clearance for landing approach routes and maintaining separation, from aircraft.
- **Mechanical issues:** Any mechanical issues with the aircraft's systems, such as landing gears, brakes, or hydraulics, can affect the landing process. If these systems are not functioning properly, it can hinder or prevent a safe landing.
- **Pilot fatigue and human factors:** Pilot fatigue, stress, distractions, or other human factors can negatively impact landing performance. It is crucial for pilots to be well-rested, alert, and focused during landing to ensure a safe and precise touchdown.

Overall, a combination of these factors needs to be considered to ensure a safe and successful landing.

4. What is ALP

Aircraft landing problem is NP hard problem and it is a complex mathematical problem that involves determining the optimal time and path for an aircraft to land. This problem can be further complicated by various factors such as weather conditions, air traffic, runway availability, and other operational constraints One of the main challenges in aircraft landing optimization is to minimize the time spent by an aircraft in the air, while also ensuring safety and efficiency. Another challenge is to optimize the use of available runways and reduce congestion at airports, especially during peak hours.

5. What is NP-hard problem:

An NP-hard problem is a type of computational problem for which no known algorithm can solve all instances of the problem efficiently in terms of time complexity. Formally, an NP-hard problem is at least as hard as the hardest problems in the complexity class NP (**nondeterministic polynomial time**).

More precisely, a problem is NP-hard if every problem in NP can be reduced to it in polynomial time. This means that if you had an efficient algorithm to solve an NP-hard problem, you could use it to solve any problem in NP efficiently.

Importantly, NP-hardness does not require a problem to be in NP,it simply means that it is "hard" in a certain sense. Many NP-hard problems are of practical interest and have real-world applications, despite their computational difficulty.

Examples of NP-hard problems include the traveling salesman problem, the knapsack problem, the Boolean satisfiability problem (SAT), and many others. These problems are central to theoretical computer science and have important implications for algorithm design and complexity theory.

6. Classification of ALP

ALP may be split into single runway issues and multiple runway problems using the basic model.

6.1 Single runway problem (SRP)

The single runway problem refers to a situation in which an airport or air traffic control system has only one runway available for both takeoffs and landings. This can create various challenges and inefficiencies in managing the flow of aircraft, especially during high traffic periods.

Some of the key issues associated with the single runway problem include:

- **Limited capacity:** With only one runway, the airport can handle a limited number of aircraft movements per hour. This can result in delays, especially during peak hours or when there are adverse weather conditions.
- **Potential conflicts:** Arriving and departing aircraft may have to share the same runway and airspace, leading to potential conflicts and increased separation requirements. This requires careful coordination and sequencing of aircraft operations.

- **Departure sequencing:** When there is a large volume of departures queued up, air traffic controllers must carefully manage the sequence of aircraft taking off to maintain safe distances between them.
- **Landing sequencing:** Similarly, when numerous aircraft are approaching to land, controllers must manage the landing sequence to ensure safe spacing between aircraft.
- Increased holding patterns: When the airport is experiencing high traffic volume, air traffic controllers may need to place arriving aircraft in holding patterns to delay their arrival and manage the flow of traffic.
- **Increased fuel consumption and emissions:** Delays and holding patterns can lead to increased fuel consumption and emissions, as aircraft spend more time in the air.

To mitigate the challenges posed by the single runway problem, some airports may implement advanced scheduling and sequencing systems, improve runway capacity through infrastructure enhancements, or expand the number of runways if feasible. Additionally, air traffic controllers play a crucial role in efficiently managing the traffic flow to minimize delays and maintain safety during peak periods.

6.2. Multi-runway problem (MRP)

A multi-runway problem refers to a scenario in airport management where there are multiple runways available for aircraft takeoffs and landings. The challenge is to efficiently allocate and schedule the use of these runways to maximize the capacity and minimize delays in air traffic. This problem arises when an airport has more than one runway, which is common for large international airports or airports with high air traffic volume. The goal is to ensure safe and efficient operations by optimizing the runway usage. Factors that need to be considered when dealing with a multi-runway problem include:

a. **Runway Configurations:** Each runway may have different lengths, directions, and capacities, which can affect the types and sizes of aircraft that can use them.
b. **Arrival and Departure Patterns:** Airports typically have specific arrival and departure routes for aircraft, and these patterns need to be considered when allocating runways.

c. **Aircraft Mix:** Different types of aircraft have varying takeoff and landing requirements, and this must be taken into account when deciding which runway is most suitable for each aircraft.
d. **Weather Conditions:** Weather conditions, such as wind direction and speed, can affect the choice of runway and may require adjustments to the runway allocation to ensure safe landings and takeoffs.

To solve the multi-runway problem, airport managers and air traffic control systems employ various techniques and technologies. This may include sophisticated algorithms and optimization models that consider all the factors mentioned above. Additionally, real-time monitoring and communication systems are used to coordinate and adjust the runway allocation as needed. The effective management of a multi-runway problem is crucial for maintaining efficient airport operations, reducing delays, and ensuring the safety of aircraft and passengers.

7. Combinatorial optimization:

Combinatorial optimization is a branch of optimization that deals with finding the best possible solution from a finite set of possible solutions. It involves making choices from a set of discrete alternatives to optimize a given objective function. One example of a combinatorial optimization problem is the Traveling Salesman Problem (TSP). In this problem, a salesman is given a list of cities and needs to find the shortest possible route that visits each city exactly once and returns to the starting city. The objective is to minimize the total distance traveled. Another example is the Knapsack Problem. In this problem, there is a set of items, each with a weight and a value, and a knapsack with a maximum weight capacity. The objective is to maximize the total value of the items placed into the knapsack while keeping the total weight within the capacity constraint. Other combinatorial optimization problems include the Job Scheduling Problem, Graph Coloring Problem, Maximum Cut Problem, and many more. These problems have numerous applications in various fields such as logistics, manufacturing, telecommunications, and computer science. Solving combinatorial optimization problems

Often requires the use of efficient algorithms and techniques like dynamic programming, branch and bound, genetic algorithms, simulated annealing, and linear programming.

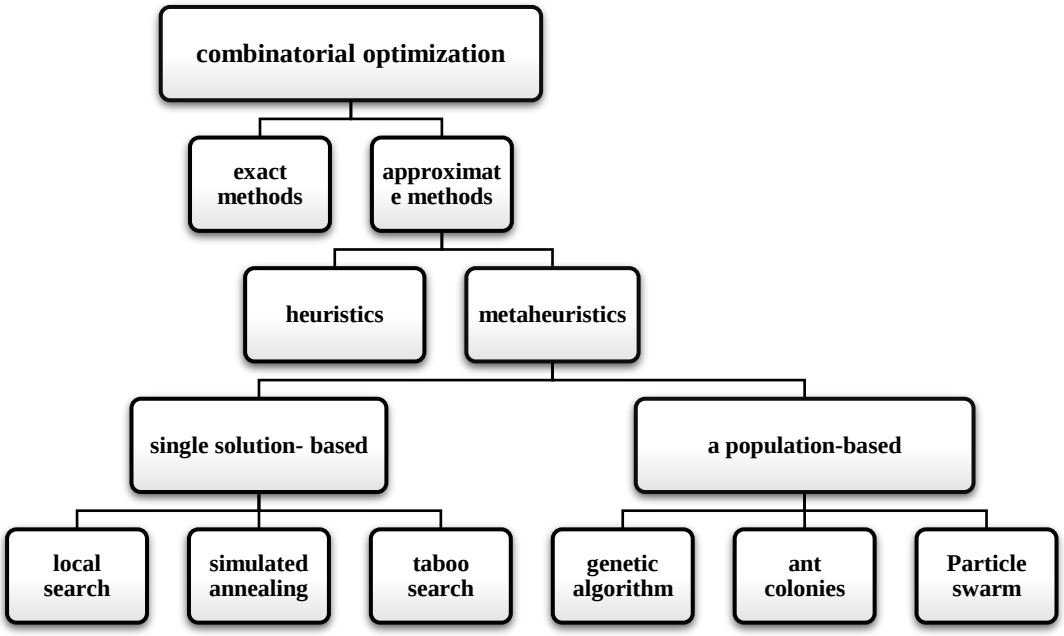

Figure 3 : Method Classification.

8. Benefits of Aircraft Landing Optimization

Optimizing aircraft landing can have several benefits, including reducing fuel consumption and emissions, improving safety, and increasing airport capacity. By reducing the time spent in the air and optimizing the use of runways, airlines can save millions of dollars in fuel costs and reduce their carbon footprint. Optimized landing paths can also reduce noise pollution and improve the overall safety of air travel

Conclusion

Improving aircraft landing is a complex problem that requires a multidisciplinary approach, involving experts in mathematics, engineering and computer science. However, the benefits of landing paths and optimal timelines are significant, in terms of cost savings and environmental impact. As air travel continues to grow, the need to improve the efficiency and safety of aircraft's landing will become even more important.

Chapter 2: Stat of the art

Introduction

Finding the optimal answer from a separate set of options known as a viable solution set is the goal of the consensual improvement issue. This group is implicitly characterized by a very small list of constraints that practical solutions must meet. It is limited but has too many items.

It offers an objective function to define the concept of an optimal solution. The best solution (or optimal solution) is one that reduces or increases the goal function. Gives real value to each option. Undoubtedly, there may be more than a perfect solution to the issue of optimal uniformity

1. Review of related work

- Andreussi et al (1981) published a study that dealt with creating a discrete-event simulation model to assess various sequencing options and referred to the issue as the "aircraft sequencing problem." Several simulated situations' computational findings were provided.

- Dear and Sherif (1989,1991) examined the static and dynamic aircraft landing issues and provided a heuristic approach based on limited position shifting for the dynamic aircraft landing problem (single runway). With the restriction that no plane can be relocated more than a predetermined number of places from the position it had in the landing queue based on FCFS, this requires identifying, for a small collection of planes, the best possible positions for them in the landing queue.

- Brinton (1992) offered an exhaustive list of all potential aircraft sequences-based depth-first tree search technique. When the cost of a partially created sequence was higher than the cheapest viable alternative, branches in the tree were cut off.

- In 1995, Beasley et al. gave a mixed-integer zero–one formulation of the issue for the single runway situation before expanding it to the multiple runway scenario [4]. Abela et al. [5] suggested 2 a branch and bound method and a genetic algorithm in 1995 to address the issue of scheduling airplane landings.

- Milan (1997) studied the queueing theory approach to the issue of allocating precedence to airplanes waiting to land.

- Ernst et al. provided a simplex algorithm that assessed landing times based on a little amount of ordering data. This approach was employed in both a branch-and-bound approach and a problem space search heuristic for the single and multiple runway cases, again for up to 50 aircraft [2]. Similar approaches were used by Beasley et al., who then provided expanded results [1].

- In 1998, Ciesielski et al. a genetic algorithm was used to construct a real-time system for airplane landings, and trials were run on landing data for Sydney's airport on its busiest day of the year [6].

- In 2001, Beasley et al. created a population heuristic and applied it to real operational data on airplane landings at the London Heathrow airport [7]. Beasley et al. conducted new research on the dynamic case of the ALP by formulating it as a displacement issue and using heuristics and linear programming [8] to the problem.

- In 2006, Pinol and Beasley reported findings for the available test issues involving up to 500 airplanes and 5 runways using two heuristic techniques: scatter search and the bionomic algorithm [3]

- The dynamic case of the problem for the single-runway case was again studied by Moser et al. in 2007 [9]. They used extremal optimization along with a deterministic algorithm to optimize a landing sequence.

- In 2008 Tang et al. developed a multi-objective evolutionary method to reduce both the overall cost incurred and the total time of arrival that is scheduled [10].

- In 2009, Bencheikh et al. formulated the ALP as a work shop scheduling issue and used hybrid approaches that combine genetic algorithms and ant colony optimization [11]. In order to modify the landing timings of the aircraft in a certain landing sequence and lower

the overall penalty cost, the same authors introduced an ant colony method and a novel heuristic in 2011 [12].

- In 2012, Simulated annealing combined with variable neighbourhood search and variable neighbourhood descent was proposed as a hybrid meta-heuristic method [13].

Conclusion :

The problem of aircraft landing has evolved over time and from the time of its emergence, attempts have begun to solve it and improve the solutions obtained. In this chapter, we have learned about the work related to the problem of aircraft landing and various attempts to solve it .

Chapter 3 : Algorithm design for the ALP

Introduction

Over time, several attempts have been made to solve and improve the problem of aircraft landing. This chapter provides an overview of how to use the simulator annealing algorithm to solve ALP.

1. Simulated annealing algorithm

1.1 Definition

Is a stochastic optimization algorithm inspired by the annealing process in metallurgy. It is used to find the global minimum of a given function, often in the context of combinatorial optimization problems. The algorithm starts with an initial solution and a high temperature. It then repeatedly generates a new solution by perturbing the current solution and calculating its cost. If the new solution is better than the current solution, it becomes the new current solution. If the new solution is worse, it may still be accepted with a certain probability depending on the temperature and the difference in cost between the new and current solutions. The probability of accepting worse solutions decreases as the algorithm progresses. The temperature is gradually reduced over time, just like in the annealing process. This is done to allow the algorithm to explore the search space more extensively in the early stages where the temperature is high, and then focus more on exploitation as the temperature decreases. As the algorithm progresses, it becomes less likely to accept worse solutions, and converges to a global optimum as the temperature approaches zero. The process of gradually reducing the temperature and exploring the search space is similar to cooling molten metal to form a solid with minimal defects, hence the term "simulated annealing". Simulated annealing algorithm has been successfully applied to various optimization problems, such as the traveling salesman problem, graph coloring, and job scheduling. It offers a good balance between exploration and exploitation, making it a powerful tool for solving complex problems where other optimization algorithms may get stuck in local optima.

1.2 General algorithm:

General algorithm of SA

```
SA(S) {
        // read data input…..
        // parameters……
        Tmax = … ;              // high temperature
        Tmin  =… ;              // low temperature
        ro=… ;                  // temperature reduction ration
      ANNEALING= ….     // number of iterations at each value of T
        // define neighbor(x) and f(x) (objective) and constraints
        x0 = random(S) ;   // initial random solution
        x* = x0 ;
        T= Tmax
        While (T>T0)
                { FOR  (i=0; i< annealing; i++)
                        {  x1  =NEIGHBOR(x0);
                           If (f(x1) < f(x0))
                               {   x0   = x1 ;
                                  If (f(x1) < f(x*))   x*= x1 ; }
                         Elseif   (rand (0,1) < exp(-(abs(f(x1)-f(x0))/T)  x0=x1 ;
                }
               T=ro*t ;
         }
      s.o.p(x*,f(x*)) ;
  }
```

1.3 Problems solved by simulated annealing algorithm:

The simulated annealing algorithm is commonly used to solve optimization problems where finding the global optimum is challenging due to the presence of local optima or when an exact solution is difficult to obtain. Some of the problems the simulated annealing algorithm is used to solve include:

- **Traveling Salesman Problem (TSP):** The TSP involves finding the shortest possible route that a salesperson can take to visit a set of cities and return to the starting point. Simulated annealing can be used to search for a near-optimal solution.

- **Knapsack Problem:** The knapsack problem involves maximizing the value of items that can be packed into a knapsack of limited capacity. Simulated annealing can be used to determine the best combination of items that maximizes the total value.
- **Job Scheduling:** Simulated annealing can be employed for scheduling tasks or jobs in a way that minimizes the overall completion time or maximizes resource utilization.
- **Graph Coloring:** In the graph coloring problem, the goal is to assign colors to the vertices of a graph such that no two adjacent vertices have the same color. Simulated annealing can be used to find valid coloring solutions with minimal number of colors.
- **Feature Selection:** Simulated annealing can be utilized for selecting the most relevant or informative features from a large set of features that best represent a given dataset or problem.
- **Neural Network Training:** Simulated annealing is used as a global optimization technique to train neural networks by finding the optimal weights that minimize the error between the predicted and actual outputs.
- **Protein Folding:** The protein folding problem aims to determine the three-dimensional structure of a protein given its amino acid sequence. Simulated annealing can help explore the vast conformational space and find the most stable structure.

These are just a few examples of the many problems that can be tackled using the simulated annealing algorithm. The algorithm's flexibility and ability to find near-optimal solutions make it suitable for a wide range of optimization problems.

1.4 Advantages and disadvantages

As for all things metaheuristic, the simulated annealing algorithm offers many advantages, and also suffers from several default advantages. In this section we cite some of the advantages and disadvantages of this method

1.4.1 Advantages

The main advantages of the simulated annealing algorithm (SA) can be summarized as follows:

- It can deal with stochastic systems and objective functions, in terms of nonlinear and multimodal models, or noisy or strongly constrained data.
- It converges toward the global optimum when the number of iterations tends toward infinity (large enough). • Provides good solutions to the majority of optimization problems.
- It is relatively easy to implement and less sensitive to the size of the problem.
- It is a generalized optimization technique, as it does not rely on all the limiting properties of the model

1.4.2Disadvantages

There are several disadvantages, among which we can mention the following:

- A large number of parameters (initial temperature, temperature drop rate, duration of temperature phases, program stop criteria).
- Frequently setting experimental parameters.
- Excessive calculation time in some applications (performance problem).
- The impossibility of knowing whether the existing solution is optimal or not.

2. Simulated annealing for Aircraft Landing Problem

The simulated annealing algorithm can be used to solve the aircraft problem by representing the possible routes as states and using a cost function to evaluate each state

The cost function takes into account factors such as:

- Distance
- fuel consumption
- Time

As well as any constraints or obstacles that must be avoided

To apply the simulated annealing algorithm to the aircraft problem

- the initial state is typically generated randomly
- the algorithm iteratively perturbs the current state and evaluates the new state using the cost function

The algorithm accepts the new state if it improves the cost function, but also accepts worse states with a certain probability to avoid getting stuck in local optima .

3. Problem Statement

Inputs:

1. The number of aircraft and their locations
2. Number of runways and their positions

Parameters:

1. Maximum iteration.
2. High temperature.
3. Low temperature.
4. Temperature reduction ration.

Outputs:

1. The distance between each plane and the nearest airport to land.
2. The result of the cost function.
3. Total landing time.

Formulation: The aircraft landing problem can be formulated as an optimization problem with the goal of minimizing the path and time taken by all aircraft to land.

Multiple runway aircraft landing problem based on the formulaiton presented in [15].

M : number of aircrafts.

N : number of runways.

P_{ij} : The distance between aircraft i and runway j

a_i : Aircraft position

a_j : Runway position

x_i = the landing time for plane i

α_i = how soon plane i lands before T_i ($i \in M$)

β_i = how late plane i lands after T_i ($i \in M$)

$$L_{ij} = \begin{cases} 1 \text{if plane i land befor j} (i, j \in M; i \neq j) \\ 0 \text{ otherwise} \end{cases}$$

$$z_{ij} = \begin{cases} 1 \; if \; i \; and \; j \; land \; on \; the \; same \; runway \;\; (i, j \in M; i \neq j) \\ 0 \; otherwise \end{cases}$$

$$kir = \begin{cases} 1 \; if \; plane \; i \; land \; on \; runway \; r (i \in M; r \in N) \\ 0 \; otherwise \end{cases}$$

each plane i has a predetermined landing time windows [Ei, Li], and also, a target time Ti ($Ei \leq Ti \leq Li$) at which time the plane is landed with cost 0. S_{ij} is the required separation time between plane i and j (where i lands before j) for landing these on the same runway. As customary in the multiple runway case, we assume that the separation time between two planes on different runways is 0. u_i and v_i denote the unit costs for plane i landing earlier and later than the target time respectively.

The task is to determine the landing time x and allocation variable y for each plane, which ensures that the minimum cost is met while meeting the following requirements:

- Each plane arrives at a specific point within the specified time frame

$$xi \in [Ei, Li] \; \forall i \in M$$

- The criteria for separating the landing of the aircraft from the landing of all successive aircraft on the same runway shall be observed. In the sense that if $W_{ij} = 1$ and $z_{ij} = 1$, then the following holds :

$$x_j \geq x_i + S_{ij} \; \forall i, j \in M; i \neq j$$

The mathematical formulation

// Reduce the total cost of deviation from target times (Ti).

$$\textbf{Min} \sum_{i=1}^{M} (u_i \alpha_i + v_i \beta_i)$$

// Ensure that each plane lands within its own time frame.

$$\mathbf{E_i \leq x_i \leq L_i \; \forall i \in M}$$

//indicate that either plane i must land before plane j (Wij = 1) or plane j must land before plane i ($W_{ij}=1$) .

$$\mathbf{W_{ij} + W_{ij} = 1 \; \forall i, j \in M; i \neq j}$$

//are symmetry constraints (if i and j land on the same runway so do j and i).

$$\mathbf{z_{ij} = z_{ji} \; \forall i, j \in M; i \neq j}$$

//ensure that, if there is any runway r on which plane i and j are both landed, then we force z_{ij} to be 1 (i and j land on the same runway). If $z_{ij} = 0$, then the constraint becomes $0 \geq y_{ir} + y_{jr} - 1$, ensuring that planes i and j can not land on the same runway.

$$\mathbf{z_{ij} \geq y_{ir} + y_{jr} - 1 \; \forall i, j \in M; i \neq j}$$

//are the separation constraints for plane i and j.

$$x_j \geq x_i + S_{ij} z_{ij} - (L_i + S_{ij} - E_j) W_{ji} \forall_i, j \in M; i \neq j$$

//ensure that α_i is at least as big as zero and the time difference between Ti and x_i, and at most the time difference between Ti and Ei

$$\alpha i \geq Ti - xi \ \forall i \in M$$

$$0 \leq \alpha i \leq Ti - Ei \ \forall i \in M$$

$$0 \leq \beta i \leq Li - Ti \ \forall i \in M$$

//relate the landing time (xi) to the time plane i lands before (αi), or after (βi), target (Ti).

$$xi = Ti - \alpha i + \beta i \ \forall i \in M$$

4. The simulated annealing algorithm for the aircraft landing problem :

Objective: Minimize total landing time

Algorithme : ALP

Inputs :

M // number of aircrafts.

N // number of runways.

Parameters:

Tmax, Tmin,Annealing ,Alpha

Start :

Read (M,N)

//Calculate total landing time

Land_time=random ()

Tot_land_time+=land_time

T=Tmax

While(T>Tmin)

{

For (i=0;i<Annealing;i++)

{ **//create new randomly solution**

land_time$_1$ =random ()

Tot_land_time$_1$+=land_time_1

If (Tot_land_time_1 <Tot_land_time)

Tot_land_time=Tot_land_time_1

Elseif (exp (abs(Tot_land_time-Tot_land_time_1)) /Tmax>rand(0,1)

Tot_land_time=Tot_land_time_1

}

T=alpha*T

}

Output :

Total landing time

}

end

Objective:

- Set the minimum distance between aircraft and runways
- Aircraft Locations

Algorithme : ALP$_1$

Inputs :

M // number of aircrafts.

N // number of runways.

Parameters:

Tmax,Tmin, Annealing,Alpha

Start :

Read (M,N)

```
air_pos [i]=random ()              //Define the aircraft and runway position
run_pos [j]=random ()
Function cost(X, Y) {              //Define the cost function
                Cost=0;
                Min_dis=1000;
                        For (int i=0;i<M;i++) {
                                For (int j=0;j<N; j++) {
                                        Int dis=abs(X[i]-Y[j])
                                 }
                            if(dis<min_dis)
                              min_dis =dis
                            }
  Cost+=min_dis ;
}
T=Tmax                                                    //Define the simulated annealing algo
While(T>Tmin) {
                For (i=0;k<Annealing;k++){
                                air_pos_1=random ()
                                run_pos_1=random ()
                                                          //Calculate the cost of current solution and the neighbor solution
                cost1=cost (air_pos[i]-air_pos[j])
                 cost2= (air_pos_1[i]-air_pos_1[j])

            }
If (exp(abs(cost1-cost2)) /Tmax>rand(0,1){
                air_pos=air_pos_1
```

run_pos=run_pos_1

}

T=alpha*T

}

Output :

aircraft position

Total minimum distance between aircraft and runways

Note: Actual aircraft landing problem formulation and solution methods may vary depending on specific requirements and available data

Conclusion

The simulated annealing algorithm is a class of optimization algorithms and is one of the oldest algorithms inspired by nature and mimics the process of processing metal materials. They are widely used to solve many improvement problems, including aircraft landing problem. In this chapter we applied the algorithm to the problem studied.

Chapter 4 : Implementation and Results

Introduction

After drafting the problem and identifying some of its details, and after proposing an algorithm to solve it in this chapter, we will implement this algorithm and see the results obtained.

1. hardware elements

In this search I used as hardware element computer hp with the following characteristics

processor Intel Core i5-4210U CPU @ 1.70GHz 2.40 GHz

Installed RAM 4.00 GB

Type of system Operating system 64-bit, x64 processor

2. software elements

Platform used is Windows 10 professional. The development language chosen is C++

3. C++ language

3.1 Definition :

A general-use programming language, C++ is an object-based programming language. Which is considered by many to be the best language for designing large-fronted applications to handle the solid structure of the computer. + + C is a high-level programming language at the same time close to the limited-level aggregation language. It is also a procedural programming language (a program with only procedures and sequences can be written). It is a geo-oriented language (the written program is a class and uses the available characteristics of a card, plurality of shapes, inheritance

and composition...). It is a language originating from the C language invented by Dennis Ritchie in the early 1970 and developed by Bjarne Stroustrup to++ C.

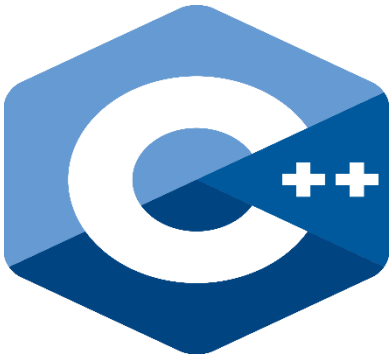

Figure 4 :C++Logo

3.2 Historical overview :

This language dates back to 1979 and was created by Brian Straustrup while working at Bell Laboratories. In fact, he worked on it as an upgraded version of the C language initially named C with grades C with classes, and 99% of C programs worked on it unchanged in the source code.

The name of the language was changed in 1983 to the currently known name C++. In time many features were added until they evolved and reached their current fame**.**

3.3 C++ language uses

- in the construction of operating systems including the construction of different software systems and user software**.**
- C++ language can be used in small applications, but more easy programming language is preferred to specialize in C++ language in more powerful software.
- They are used to create high service software and hardware operators
- Used in the creation and development of different video games due to their high effectiveness in it.

3.4 Advantages and Disadvantages :

3.4.1 Advantages :

- Object-oriented language.
- Multifunctional.
- Comprehensive.
- Powerful, as it is used in many operating systems.
- It is one of the leading languages in the development of large and complex projects.
- Inherited the advantages of C language but added many important software patterns, Encapsulation, Inheritance, Polymorphism, abstraction and classes.
- It provided many functions associated with cash and overload.
- It provides a standard STL library of algorithms, income and exit units.

3.4.2 Disadvantages:

- It is an unsafe language.
- Complex for new learners.
- Difficult to handle and correct when used in web applications.
- Do not support the restoration of Garbage collection resources.
- Do not provide much support to the programmer
- There are no virtual libraries of their own to design graphic interfaces

4. Some apps that used this language :

- Operating systems such as Windows and Apple Os X
- Most Adobe programs such as photoshop, illustrator and Premier
- Google Application like Google Chrome
- Mozila Firefox Internet browser.

- 3D modeling software.
- Amazon

Implementation and Results :

1. Import of the necessary Libraries :

```
#include <iostream>
#include <cmath>
#include <random>
#include <vector>
#include <chrono>
Using namespace std;
```

#include<iostream> : It contains input and output functions and is short for Input Output Stream and includes compulsory in the program if we want to do input and output**.**

#include<cmath> : used to allow the user to perform some math related functions like sqrt () , pow(), cos (), abs () and many other functions

#include<random> : used to get a random number

#include<vector> : It is used to create an object representing a container in which the elements we add are stored serially behind each other giving each of them an Index number.

#include<chrono> : provides a precision-neutral way of handling date and time that is independent of the underlying frameworks used by different systems

using namespace std : It means "**use the name space called std**" and is called the authorized field technique where we use **Namespaces** in our software using the word reserved using to tell the interpreter that the functions and variables are authorized in a specific place ,and if you delete this phrase you will have to put **std: :** Before every order in the program this is very tiring, and std is short for Standard.

2.Define the number of runways and aircraft :

```
Const int num_runways = 5;
Const int num_aircraft = 10;
```

Const : anything that is defined in a way that cannot be re-valued. We can use it with the following things: Variables, Pointers, Objects…

3. Define the parameters:

```
Const double T_max = 1000.0;
Const double T_min = 0.001;
Const double Annealing = 1000;
Const double Alpha = 0.99;
```

4. Define the function to calculate the total landing time :

```
double calculate_total_landing_time (int num_runways, int num_aircraft) {
    double total_landing_time = 0.0;
    for (int i = 0; i < num_aircraft; i++) {
        double landing_time = 0.0;
        for (int j = 0; j < num_runways; j++) {
            landing_time += (double) rand() / RAND_MAX;
        }
        total_landing_time += landing_time;
    }
    return total_landing_time;
}
```

5. Define the simulated annealing algorithm1 :

```
double simulated_annealing(int num_runways, int num_aircraft) {
    double temperature = T_max;
    double Alpha = 0.99;
    int iteration = 0;
```

```
 double existing _solution = calculate_total_landing_time(num_runways, num_aircraft);
 double best_solution = existing _solution;

                                                                  // Start the SA
    while (temperature > T_min && iteration < Annealing) {

                                                          // Generate a new solution
        double new_solution = calculate_total_landing_time(num_runways,
num_aircraft);

                                          // Calculate the acceptance probability
 double acceptance_probability =exp((existing _solution - new_solution)/temperature);

        if (acceptance_probability > rand() ) {
           existing _solution = new_solution;
        }
        if (existing _solution < best_solution) {
            best_solution = current_solution;
        }
        temperature *= Alpha;
        iteration++;

    }

    return best_solution;
}
```

6. Define the aircraft and runway positions

```
vector<int> aircraft_positions(num_aircraft);
vector<int> runway_positions(num_runways);
```

7. Define the cost function

```
double cost(const vector<int>& aircraft_positions, const vector<int>&
runway_positions) {
    double cost = 0.0;
    for (int i = 0; i < num_aircraft; i++) {
        int min_distance = 100000;
        for (int j = 0; j < num_runways; j++) {
            int distance = abs(aircraft_positions[i] - runway_positions[j]);
            if (distance < min_distance) {
                min_distance = distance;
            }
        }
        cost += min_distance;
    }
    return cost;

}
```

8. Define the simulated annealing 1

```
void simulated_annealing_1() {
                    // Initialize the aircraft and runway positions randomly
    for (int i = 0; i < num_aircraft; i++) {
        aircraft_positions[i] = rand() % 100;
    }
    for (int i = 0; i < num_runways; i++) {
        runway_positions[i] = rand() % 100;
    }
    double temperature = T_max;
    int iteration = 0;

                                                        // Run the SA
    while (temperature > T_min && iteration < Annealing) {
                                        // Generate a random neighbor solution
        vector<int> neighbor_aircraft_positions(aircraft_positions);
        vector<int> neighbor_runway_positions(runway_positions);
        int a = rand() % num_aircraft;
        int b = rand() % num_aircraft;
        swap(neighbor_aircraft_positions[a], neighbor_aircraft_positions[b]);
        int c = rand() % num_runways;
        int d = rand() % num_runways;
        swap(neighbor_runway_positions[c], neighbor_runway_positions[d]);
```

```
double current_cost = cost(aircraft_positions, runway_positions);
double neighbor_cost=cost(neighbor_aircraft_positions,neighbor_runway_positions);

                                        // Calculate the acceptance probability
         double accept_proba = exp((current_cost - neighbor_cost) / temperature);

               if (accept_proba > rand()) {
            aircraft_positions = neighbor_aircraft_positions;
            runway_positions = neighbor_runway_positions;
        }
        temperature *= ALPHA;
        iteration++;
    }
    cout << "Final solution: ";
    for (int i = 0; i < num_aircraft; i++) {
        cout << aircraft_positions[i] << " ";
    }
    cout << endl;
    cout << "Final cost: " << cost(aircraft_positions, runway_positions) << endl;

}
```

9. Main function :

It is the main function of the program that starts from when the program is implemented and is present in any program written in C++ language whatever its size or role

```
int main() {
    srand(time(NULL));
    double best_solution = simulated_annealing(num_runways, num_aircraft);
    cout << "The best solution is: " << best_solution << endl;

    simulated_annealing_1();

    return 0;

}
```

Finally, after implementing the above-mentioned programme and experimenting with some values, we have the following results:

When the number of aircraft is **25** and the number of runways is **4** :

The best solution is: 40.9828

Final solution: 13 35 32 30 42 63 95 36 88 10 98 34 51 82 56 97 54 62 9 55 70 13 6 38 21

Final cost: 331

When the number of aircraft is **100** and the number of runways is **3** :

The best solution is: 133.945

Final solution: 36 16 0 95 36 80 99 29 98 48 80 18 24 66 69 62 92 91 77 56 60 84 82 75 19 80 82 38 60 92 77 45 23 1 94 89 78 64 15 85 94 40 70 66 34 79 62 12 18 74 49 30 64 86 79 95 11 91 63 6 45 16 96 51 47 51 18 44 70 43 94 55 24 43 96 40 27 53 0 13 93 7 35 88 5 92 34 17 34 38 6 93 29 66 23 66 47 45 71 75

Final cost: 1940

When the number of aircraft is **2** and the number of runways is **6** :

The best solution is: 2.38447

Final solution: 30 40

Final cost: 14

Conclusion

Implementing the algorithm helps us better understand the problem and detect errors.

In this chapter, we first introduced the used hardware, software and data environments, secondly we explained parts of the program and the role of the functions and libraries used in it, and finally we implemented the proposed annealing simulator algorithm to solve ALP and showed some of the results obtained

The problem of aircraft landing remains a wide area of research and cannot be given the exact solution .

General conclusion

In conclusion, our study showed the effectiveness of the simulator annealing algorithm in solving the problem of aircraft landing

Through this research, we have been able to achieve results that contribute to this area in terms of optimal achievement of multiple objectives such as reducing aircraft landing time and reducing the distance between each aircraft and the designated landing runway, taking into account the constraints.

In this thesis we encountered several difficulties while studying the problem of landing aircraft and using the SA to solve it. Some challenges include:

- Collect information needed for the start of the study
- Difficulty and complexity of the problem
- Calculation Time
- Evaluation of results
- modelling of constraints

Despite these challenges, we have been able to overcome these difficulties by following the working methodology and constant adjustments. On the other hand, these problems and difficulties have contributed to our understanding of the problem in a clearer way and opened up new research prospects.

In the end, we can say that the simulated annealing algorithm was acceptably successful in solving ALP and paved the way for other research. Our future advances can focus on improving the simulator annealing algorithm techniques and trying to remedy the deficiencies discovered in them by combining them with other optimization algorithms such as ant colonies .

Bibliography

[1] J. Beasley, M. Krishnamoorthy, Y. Sharaiha, and D. Abramson, "Scheduling aircraft landings - the static case," Transportation Science, vol. 34, no. 2, pp. 180–197, 2000.

[2] A. Ernst, M. Krishnamoorthy, and R. Storer, "Heuristic and exact algorithms for scheduling aircraft landings," Networks, vol. 34, no. 3, pp. 229–241, 1999.

[3] H. Pinol and J. Beasley, "Scatter search and bionomic algorithms for the aircraft landing problem," European Journal of Operational Research, vol. 171, no. 2, pp. 439–462, 2006.

[4] J. Beasley, M. Krishnamoorthy, Y. Sharaiha, and D. Abramson, "Scheduling aircraft landings - the static case," The Management School, Imperial College, London SW7 2AZ, England, 1995.

[5] J. Abela, D. Abramson, M. Krishnamoorthy, A. De Silva, and G. Mills, "Computing optimal schedules for landing aircraft," in Proceedings of the 12th National Conference of the Australian Society for Operations Research, Adelaide, pp. 71–90, 1993.

[6] V. Ciesielski and P. Scerri, "An anytime algorithm for scheduling of aircraft landing times using genetic algorithms," Australian Journal of Intelligent Information Processing Systems, vol. 4, pp. 206

[7] J. Beasley, J. Sonander, and P. Havelock, "Scheduling aircraft landings at london heathrow using a population heuristic," Journal of the Operational Research Society, vol. 52, no. 5, pp. 483–493, 2001.

[8] J. Beasley, M. Krishnamoorthy, Y. Sharaiha, and D. Abramson, "Displacement problem and dynamically scheduling aircraft landings," Journal of the operational research society, vol. 55, no. 1, pp. 54–64, 2004.

[9] I. Moser and T. Hendtlass, "Solving dynamic single-runway aircraft landing problems with extremal optimisation," in IEEE Symposium on Computational Intelligence in Scheduling, SCIS., pp. 206–211, 2007.

[10] K. Tang, Z. Wang, X. Cao, and J. Zhang, "A multi-objective evolutionary approach to aircraft landing scheduling problems," in IEEE Conference on Evolutionary Computation, CEC., pp. 3650–3656, 2008.

[11] G. Bencheikh, J. Boukachour, A. Alaoui, and F. Khoukhi, "Hybrid method for aircraft landing scheduling based on a job shop formulation," International Journal of Computer Science and Network Security, vol. 9, no. 8, pp. 78–88, 2009.

[12] G. Bencheikh, J. Boukachour, and A. Alaoui, "Improved ant colony algorithm to solve the aircraft landing problem," International Journal of Computer Theory and Engineering, vol. 3, no. 2, pp. 224–233, 2011.

[13] A. Salehipour, M. Modarres, and L. Naeni, "An efficient hybrid meta-heuristic for aircraft landing problem," Computers & Operations Research, vol. 40, no. 1, pp. 207– 213, 2012.

[14] Sabar, N.R., Kendall, G.: An iterated local search with multiple perturbation operators and time varying perturbation strength for the aircraft landing problem. Omega 56, 88–98 (2015)

[15] J. E. Beasley, M. Krishnamoorthy, Y. M. Sharaiha, and D. Abranmson, Scheduling Aircraft Landings-the static case, Transportation Science, 34(2), 180-197, 2000.

الملخص

في هذا العمل، قمنا بحل مشكلة هبوط الطائرات، باستخدام خوارزمية التلدين المحاكي .حيث تُعرف هذه المشكلة بأنها NP-hard problemمما يعني أن العثور على حل مثالي في فترة زمنية معقولة يمثل تحديًا. سمحت لنا هذه الخوارزمية التي طبقناها باستكشاف مساحة الحل بكفاءة والحصول على نتائج واعدة. بالرغم من أن الحلول التي توصلنا إليها قد لا تكون مثالية إلا أنها كانت قريبة من المستوى المرضي ضمن إطار زمني معقول. حققت خوارزمية التلدين المحاكي تقدمًا كبيرًا في حل مشكلة هبوط الطائرات وفتحت أمامنا آفاقًا جديدة للبحث والتحسين .

الكلمات المفتاحية :مشكلة هبوط الطائرات ، التحسين التجميعي، خوارزمية التلدين المحاكي ،Metaheuristics .

Abstract

In this work, we solved the problem of airplane landing, using the simulated annealing algorithm. This problem is known as NP-hard problem, which means that finding an ideal solution in a reasonable period of time is a challenge. This algorithm we applied allowed us to efficiently explore the solution space and get good results. Our solutions may not be ideal but they were close to the satisfactory level within a reasonable time frame. The simulated annealing algorithm has made great progress in solving the problem of aircraft landing and has opened up new avenues for research and improvement.

Keywords: ALP, Combinatorial Optimization, Simulated annealing, Metaheuristics.

Résumé

Dans ce travail, nous avons résolu le problème de l'atterrissage d'avion, en utilisant l'algorithme de recuit simulé. Ce problème est connu sous le nom de problème NP-hard, ce qui signifie que trouver une solution idéale dans un délai raisonnable est un défi. Cet algorithme que nous avons appliqué nous a permis d'explorer efficacement l'espace de solution et d'obtenir de bons résultats. Nos solutions ne sont peut-être pas idéales, mais elles étaient presque satisfaisantes dans un délai raisonnable. L'algorithme de recuit simulé a fait de grands progrès dans la résolution du problème de l'atterrissage des avions et a ouvert de nouvelles voies pour la recherche et l'amélioration.

Mots clés : ALP, Optimisation Combinatoire, Recuit Simulé, Métaheuristiques.