

Enhancing the smart parking assignment system through constraints optimization

Nihal Elkhaldi, Faouzia Benabbou, Nawal Sael

Laboratory of Modeling and Information Technology, Faculty of Sciences Ben M'Sick, University Hassan II Casablanca, Casablanca, Morocco

Article Info

Article history:

Received Jun 2, 2023

Revised Oct 16, 2023

Accepted Nov 15, 2023

Keywords:

Assignment problem

Constrained optimization

Management system

Multi-agent system

Smart parking

ABSTRACT

Traffic in big cities has become a black spot for drivers. One of the major concerns is the parking problem that hinders urban mobility, particularly in big cities and other congested areas. This leads to an increase in accidents, a big consumption of fuel, and a spectacular augmentation of pollution. In this paper, we introduce a parking assignment system grounded in constraint programming to address the growing demand for efficient parking management in smart cities. Our system is designed to meet the requirements of groups of drivers seeking to reserve parking spaces simultaneously within the same period and geographical area. This entails imposing constraints on the desired parking type, including considerations such as walking and driving distances, parking costs, and availability. Within the scope of this study, we propose two formulations: constraint satisfaction programming (CSP) with an objective function and mixed-integer linear programming (MILP). Evaluation shows Choco, a CSP solver, is effective for smaller requests but slower for larger ones, while MILP excels for larger scenarios. Both solvers produce high-quality solutions meeting real-time response requirements. Our research offers innovative solutions for smart city management, considering parking type preferences, costs, and availability. We contribute significantly to parking space assignment methodologies, aiming to alleviate the time-consuming search for parking, reduce accidents, fuel consumption, and pollution.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Nihal Elkhaldi

Laboratory of Modeling and Information Technology, Faculty of Sciences Ben M'Sick

University Hassan II of Casablanca

Driss Harti Sidi othman, Casablanca, BP 7955, Morocco

Email: elkhaldi.nihal@gmail.com

1. INTRODUCTION

Today, numerous cities are aspiring to transform into smart cities, emphasizing human-centered development, effective governance, and a heightened commitment to environmental conservation. However, to achieve this vision, cities must confront significant challenges, foremost among them being the development of digital technologies and interconnected devices that enhance public services and enable informed decision-making based on available data. Smart cities draw upon an array of emerging technologies, including artificial intelligence and the internet of things (IoT), to address various critical issues. These encompass transportation and mobility management, security enhancements, healthcare services, optimization of public lighting energy consumption, waste collection efficiency, and environmental sustainability. Notably, the management of distributed parking within metropolitan areas stands as a pivotal challenge within the smart city context, given its profound implications for drivers, the environment, and the

business landscape [1]. Furthermore, the automotive industry in Morocco has witnessed significant growth in recent years, contributing to a rise in private car ownership. This surge results in a higher volume of vehicles circulating in major cities such as Casablanca. The absence of an efficient parking management system has led to a growing demand for parking slots, particularly in the city center. Owing to extended pick-up times, the scarcity of parking spaces and the dearth of real-time traffic information, drivers often resort to inappropriate parking locations, obstructing traffic flow and impeding pedestrian mobility, thereby exacerbating traffic congestion issues. The concept of smart parking seeks to offer intelligent solutions for monitoring parking facilities, providing drivers with up-to-date traffic status information, and guiding them to locate optimal parking spaces. These parking lots are equipped with interconnected sensors that relay pertinent data to drivers via their smartphones, encompassing the maximum parking capacity, the number of vacant spaces, and parking lot geo-location. These integrated systems are typically referred to as smart parking management systems (SPMS). The strategic objective of SPMS is to optimize parking in urban areas where numerous drivers are searching for parking in various locations. The parking assignment process needs to be highly efficient in order to accommodate all driver constraints. In a prior study, we introduced SPMS based on a multi-agent system (MAS) [2], known as the multi-agent smart parking management system (MASPMS). In the MASPMS model [3], each agent assumes responsibility for one or multiple tasks, working in coordination to offer a real-time intelligent system for parking slot reservations. This system comprises eight key agents that collaborate to expedite the search for available parking spaces: i) parking lot, ii) zone, iii) reservation, iv) synchronization, v) geo-location, vi) classification, vii) orientation, and viii) security agents.

The reservation agent (RA) relies on the work of what we refer to as a classification agent (CA), which forms the core of the parking assignment system (PAS). The primary challenge in developing an intelligent PAS is twofold: ensuring compliance with individual drivers' specific requirements and delivering real-time responses to a multitude of drivers. Firstly, each driver possesses a unique set of criteria that govern their parking preferences, encompassing factors such as the type of parking facility (e.g., covered and open-air), driving distance, walking distance, and parking fees. Secondly, the system must be capable of offering real-time responses to diverse drivers simultaneously searching for available parking spaces across different areas within the city. This challenge necessitates the effective allocation of parking facilities that cater to the needs of all drivers, concurrently. A SPMS disseminates real-time information to drivers about the current parking scenario, guiding them to the best-suited parking facilities based on their specific requirements. Three critical factors must be taken into account in the PAS: capacity, distinct assignments tailored to individual drivers, and adherence to drivers' specific requirements. First and foremost, the real-time parking capacity and availability determine whether a reservation request is accepted or declined. Secondly, it is crucial to ensure that different drivers are assigned to separate parking spaces to avoid overlaps. Lastly, the proposed parking allocation should align with user requirements, including considerations such as the minimum walking distance to reach the driver's intended destination. The challenge in parking space assignment stems from the multitude of constraints imposed by users. Each reservation request for a parking space comprises a diverse array of constraints, reflecting the unique needs of many drivers. Expressing these rules explicitly can be particularly challenging due to the inherent diversity in drivers' requirements. Moreover, the task becomes even more complex when a substantial number of drivers simultaneously request parking spaces on the road, coupled with the continuous fluctuations in parking slot availability. Thus, the assignment of real-time and optimal solutions within a PAS is a formidable challenge.

In recent years, several studies have focused on enhancing parking management and assignment through diverse approaches, including machine learning, as exemplified in [4], [5], as well as optimization algorithms [6]. However, the traditional first-in-first-out (FIFO) technique becomes inadequate when the system is confronted with a simultaneous influx of reservation requests across various areas, each containing differing numbers of available parking spaces. In such scenarios, there arises a necessity to devise a real-time system capable of efficiently accommodating multiple users' requests, respecting their constraints, and optimizing the assignment process. This study aims to introduce a system designed for the management of multiple reservation requests that occur concurrently in a MASPMS responsible for overseeing various parking facilities. The system takes into careful consideration several constraints pertinent to drivers. The drivers' requirements under consideration encompass factors such as parking capacity, walking distance from a designated parking site to their destination, parking fees, the probability of availability, and the driving distance from the user's present location to the allocated parking space. To achieve this objective, we have employed two distinct problem assignment formulations. The first utilizes constraint satisfaction problems (CSP) [7], while the second is based on mixed-integer linear programming (MILP) [8]. In both cases, the primary goal is to allocate parking spaces to users while adhering to their constraints or optimizing the proposed solutions.

In what follows, in section 2, we propose a review of related work. Section 3 proposes an outline of some backgrounds. Section 4 presents our materials and used methods. Simulation results based on the three solvers Gurobi optimizer, Choco, and Google OR-Tools are illustrated in section 5. Section 6 finally discusses the conclusion and further research.

2. LITERATURE REVIEW

Several researchers in the literature have expressed an interest in the parking assignment problem. In this context, numerous models and techniques have been presented. For example, Zhao *et al.* [9] proposed a method for an intelligent PAS that can provide guidance information to vehicles in real-time. They develop a linear optimization-based algorithm for a specific assignment problem by keeping parking requests in a queue for a set period. Kim *et al.* [10] presented a PAS with two goals: i) minimize parking expenses and ii) balance parking demand across numerous lots using a MILP. To resolve the assignment problem in this study, they used a technique known as the alternating direction multipliers method (ADMM). Numerical results prove that the ADMM outperforms Greedy by 36% in parking utilization. Ratli *et al.* [11] described a dynamic PAS to maximize parking occupancy and provide total customer happiness. They proposed a combination of the estimation of distribution algorithm (EDA) and a local search (LS) procedure based on MILP. To enhance the approach's performance, the problem was decomposed, and therefore, resolution time was reduced. The approach adopted in [12] presented a PAS for a group of parking. The price system and time-of-arrival balance constraints are included in their model. The authors used a standard ant colony optimization (ACO) algorithm that improved further by employing adaptive techniques, to resolve the issue. ACO is a nature-inspired algorithm and it is one of the swarm intelligence algorithms.

Hakeem *et al.* [13] presents a free parking system (FPS). It is an allocation system for assigning free street parking places to city drivers. It offers a dynamic assignment method that manages a collection of driver requests that separately reach the system by allocating available spots to each driver individually. The proposed parking location is the closest to the destination, reducing overall trip time. Alfonsetti *et al.* [14] proposed PAS based on the walking distance as constraints. The authors here proposed to model the global social benefit of users by developing a distributed algorithm based on Lagrange's theory of duality. A parking assignment model is presented in [15]. The authors suggest a variable neighborhood search-based heuristic to generate approximations for larger instances as well as a 0-1 programming to compute exact solutions. Sadhukhan [16] proposed an electronic parking management system (E-parking), based on parking space monitoring according to the time lag between the sending and the reception of signals which means a reservation request. They have developed an application on which users send their reservation requests via graphical user interface (GUI). It enables the user to learn about the availability of different parking facilities across the city.

Kotb *et al.* [17] proposed a PAS for parking management named i-parking. This approach aims to achieve three goals: i) increasing parking revenue, ii) increasing the use of parking resources, and iii) minimizing cost and walking time. Their system is based on mathematical modeling using MILP. They used the IBM ILOG CPLEX (CPLEX) software to solve the problem. Moreover, they came up with scalability techniques for their system like grouping the number of resources, splitting the area, and controlling reservations. An intelligent parking management system is proposed in [18], which exploits detection technologies like ground sensors and LED devices. The mathematical modeling of the proposed assignment problem is made by MILP, which guarantees a feasible solution by satisfying the constraints. Rahayu and Mustapa [19] presented a secure technology global system for mobile communications (GSM)-based parking place reservation system. They carry out two architectures for two modules: parking reservation and monitoring using GSM. The purpose is to ensure the security of the vehicles in the parking lot, enhance parking management efficiency, and reduce the parking search times. The reservation procedure was analyzed in [20] as a resource allocation problem. Their MILP-based solution has the goal of reducing the tracking cost. The model offers real-time bookings with fixed prices. An auction-based reservation system is proposed in [21]. To estimate the changes in parking time following altering parking prices, the authors provided a parking time model based on the study of real parking data. They compared four parking management strategies: i) the typical no-reservation parking scheme; ii) a parking reservation system that includes auctions; iii) the increase in parking prices due to the use of a reservation system; and iv) reduced parking expenses with the use of reservations.

Shiyao *et al.* [22] carried out a parking management system that measures the occupancy status of parking spaces using pressure sensors and the ZigBee sensor network to determine the status of parking spaces' occupancy. Reservations through a website were included to reduce overall parking problems. Nevertheless, the study focused mostly on technologies or the communication medium between drivers and systems and did not address a model or algorithm that efficiently handles booking in such a way that considerable benefits were produced. Abidi *et al.* [23] presented a hybrid genetic algorithm to solve an parking slot assignment problem for groups of drivers (PSAPG) respecting the following constraints: the

walking distance between the parking lot and the destination, expected parking cost, and time restriction. Their mathematical modeling is based on linear optimization problems. Caliskan *et al.* [24] focused on establishing a system to monitor parking occupancy status and providing a predictive model of parking occupancy to improve the prediction of free space in parking lots. Hanif *et al.* [25] presented a new reservation system via short message service (SMS). The authors have only proposed a communication system by SMS by integrating the micro-remote terminal unit (RTU) and the microcontroller and not a process of reservation of place for the allocation of parking lots. Errouso *et al.* [26] created an integrative method based on fuzzy logic to address the problem of parking allocation for road users, whether they are professionals or regular citizens. The method is based on processing parking requests from all areas of a city and redistributing them to balance the occupation load of the different regions. Zhang *et al.* [27] proposed an approach for detecting parking locations based on the deep convolutional neural network (DCNN). A large-scale labeled data set was created to aid in the investigation of vision-based parking place recognition. This approach has confirmed good effectiveness and efficiency through comprehensive testing.

This related work shows that the parking problem is generally treated as an assignment problem. The majority of studies were modeled based on the MILP method. However, only a few studies have considered the execution time of problem-solving despite its usefulness and importance in assignment problem modeling. On the other hand, the majority of the aforementioned technologies merely attempt to address the issues of recognizing available parking lots and guiding drivers to these spaces. Furthermore, certain methods processed reservation requests in a sequential order typically using the FIFO method. When there are many drivers reserved in numerous regions, the view is limited.

3. BACKGROUND

Constraint programming (CP) is a fascinating field that revolves around the principles of computational systems grounded in the art of constraint satisfaction. At its core, CP provides a powerful framework for expressing complex problems in a declarative manner, akin to mathematical modeling. What sets CP apart is its remarkable ability to tackle a wide range of intricate optimization challenges, often characterized by combinatorial intricacies. In essence, CP empowers us to focus on defining the rules and relationships within a problem, allowing the solver to navigate the labyrinth of possibilities and find optimal solutions. This versatility and problem-solving prowess have made CP an indispensable tool in domains ranging from logistics and scheduling to artificial intelligence and beyond, offering insights and solutions that were previously elusive.

3.1. Constraint satisfaction problem

CP deals with the CSP. CSP is a mathematical problem [20]. It is made up of a set of values called domains to be assigned to variables that satisfy a particular set of constraints. Operations research and artificial intelligence both conduct extensive studies on CSP. Formally speaking, we have. The triplet $(X, D, \text{and } C)$ defines a CSP where:

- $X = \{X_1, \dots, X_n\}$ is the set of variables;
- $D = \{D_1, \dots, D_n\}$ is the set of domains, where D_i is the finite set of possible values for the variable X_i ; and
- $C = \{C_1, \dots, C_m\}$ is a set of constraints. Each C_i is defined by the set $X_{i=}\{X_{i1}, \dots, X_{ik}\}$ of k variables involved in C_i .

Concerning the resolution algorithms, we distinguish two main classes. The first is systematic, complete, and extends a set of consistent values for a portion of the problem's variables to find a solution, appending consistent values for other variables continuously until a complete solution is provided, such as the generate and test, backtracking, and forward checking algorithms. The second class of algorithms are stochastic, incomplete, and attempt to find a solution by fixing an inconsistent set of values for all variable. changing an inconsistent value for one variable repeatedly until a comprehensive answer is obtained, as the case of the tabu search [28], genetic algorithm [29], and swarm intelligent algorithms [30]. However, we can find hybrid algorithms where the extension and repair methods are merged.

3.2. Constraint optimization problem

A constraint optimization problem (COP) extends a CSP by introducing an objective function that needs to be optimized. In a CSP, the goal is to find any valid assignment of values to variables that satisfies the given constraints. In contrast, in a COP, we add an objective function that quantifies solution quality and can be either maximized or minimized. To solve a COP, the task is twofold:

- Satisfy constraints: as in CSPs, values must be assigned to variables to satisfy all imposed constraints, ensuring the solution adheres to specified rules.

- Optimize the objective function: simultaneously, the aim is to find the best assignment of values to variables that either maximizes or minimizes the objective function, achieving the desired criteria, such as maximizing profit or minimizing cost.

3.3. Mixed-integer linear programming

Mixed integer programming (MIP) is used to solve several real-world problems such as scheduling, supply chain management, and production planning. When the objective function and the constraints are linear, the problem became MILP. In a MILP, the feasible decision space is described by a set (or some) of affine constraints and the decision variables are integers. We consider that $f^T x$ is the objective function, $Ax \leq b$ are the linear constraints, l and u are simple lower and upper bounds on the problem variables x . When the integer variable bounds are $\{0, 1\}$, they called binary variables. Therefore, a MILP can be stated as:

$$\min f^T x \quad (1)$$

$$Ax \leq b \quad (2)$$

$$l \leq x \leq u \quad (3)$$

$$x \in \mathbb{R}^n \quad (4)$$

$$x_j \in \mathbb{Z} \quad \forall j \in I \quad (5)$$

where: $A \in \mathbb{R}^{m \times n}$, $f \in \mathbb{R}^n$, $b \in \mathbb{R}^m$, $l \in (\mathbb{R} \cup \{-\infty\})^m$, $u \in (\mathbb{R} \cup \{-\infty\})^m$, and $I \subseteq \mathbb{N} = \{1, \dots, n\}$.

Solving MIP and CP problems necessitates a deep and potentially exhaustive search. The "classic" way to solve these problems is called branch and bound. Land and Doig [31] were the first to propose the branch and bound algorithm in 1960, Gupta and Ravindran [32] suggested a branch-and-bound implementation for nonlinear convex problems in 1985 and investigated the implications of problem size on resolution time, as well as various branching and node selection techniques. Branch and bound algorithms consist of exploring the entire search space of possible solutions and providing an optimal solution. Indeed, the approach starts by identifying the best answer to the "relaxation" of the problem without the integer constraints. If the decision variables with integer constraints in this solution take integer values, then no additional work is required. In the case of incompleteness of the solution, i.e. if one or more of the variables do not have an integral solution, the branch and bound method chooses such a variable and "branches", to generate two new subproblems in which the value of this variable is more strictly constrained. These subproblems are solved, and the procedure is continued until a solution that meets all of the integer requirements appears.

4. MATERIALS AND METHODS

4.1. A new parking assignment system using multi-agent system

This section outlines the methodology conducted to propose a parking slot reservation. This service is based on a MASPMS. MAS are systems that have shown great efficiency in modeling distributed, coordinated and complex systems [33], and many management systems are based on them. Indeed, agents can cooperate to provide services to the end user. An agent's actions are determined by a set of rules that represent the desired control policies.

Figure 1 presents the chart flow of the proposed system based on eight agents. The parking agent (PA) checks the parking status and transfers the availability data at each change to the RA. RA receives the reservation request from the drivers agent (DA) and sends the collected information to the CA, which is the central server responsible for parking slot assignment, it collects and updates the data received using the geo-location agent (GA). When solving the parking assignment problem, the information gathered is later used to find the answer. The processing of parking requests is triggered by the driver's request via a personal navigation device in a vehicle. A driver enters his destination on the personal navigation system's screen, a desired maximum walking distance from the parking lots to the destination, and the parking request along with the necessary information used in selecting a parking space. Furthermore, the GA completes the driver request by determining the current location of vehicles requesting parking as well as the destination location. Based on receiving drivers' information and GA information, the RA execute an algorithm and proposes the most suitable parking lot meeting the requirement of drivers.

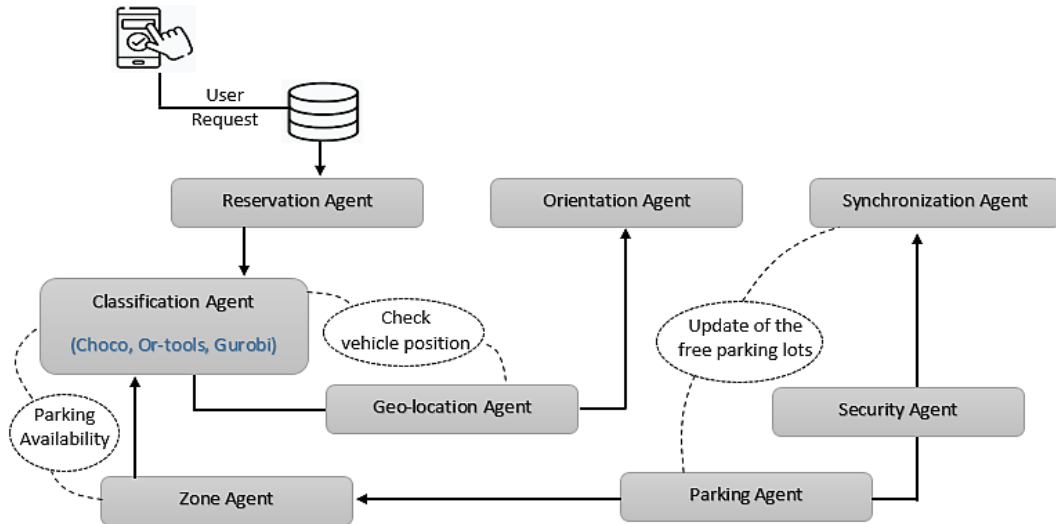


Figure 1. Diagram of agents exploitation upon receipt of the reservation request

4.2. Problem statement

In this section, we propose a mathematical programming model to assign a parking slot to drivers. The parking reservation problem is NP-complete restricted combinatorial optimization problem [26], which can be thought of as a resource assignment problem. One of the most appropriate models for this type of problem is MIP as proved by the literature review. The MIP model gives a concrete description of the problem, and once the model is developed, it can easily be adjusted to reflect changes in the problem, such as removing or adding constraints or objectives. On the other hand, few studies in the literature have used the CSP approach to the best of our knowledge, despite the demonstrated effectiveness in other domains such as the airport gate assignment problem [34]. In this paper, we carry out these two parking assignment models based on CSP and MILP methods. The reservation problem is seen as a problem of multi-variables. Multiple users' requests are emitted to the MASPMS, managing multiple parking lots distributed in a big city, and are processed simultaneously and resolved in real-time while taking the needs of the drivers into consideration.

To present the problem, let us consider the set of reservation requests $R = \{R_1, \dots, R_n\}$ received over a time interval from a group of users $U = \{U_1, \dots, U_n\}$, who book in different destination $Des(U) = \{Des(U_1), \dots, Des(U_n)\}$. The destination $Des(U_i)$ is considered as a parking zone Z_i consisting of m parking lots $P = \{P_1, \dots, P_m\}$. We first list in Table 1, the parameters used in the proposed models. To illustrate our mathematical programming model, we use many symbols. Table 1 shows the symbols and their brief descriptions.

Table 1. Notation and symbols

Symbol	Description
I	$I = \{1, \dots, n\}$ the set of the number of user requests
J	$J = \{1, \dots, m\}$ the set of parking lots in an area
R	The set of reservation requests
Z	The zone in the city
P	The set of parking lots $P = \{p_1, \dots, p_m\}$
p_j	The j^{th} parking lots.
U_i	The i^{th} user.
Cp_j	The number of available parking spots in p_j .
x_{ij}	The assignment matrix. Such as $i \in I$ and $j \in J$.
$Des(U_i)$	The destination of the user.
$Loc - Des(U_i)$	The location of the users' destination.
$Loc - p_j$	The location of parking lots.
w_i	The walking distance between the proposed parking and the user destination.
d_{ij}	The driving distance between the current location of users and the proposed parking.
$Current - loc(U_i)$	The current location of users.
dt_i	The stay time of users U_i in parking lots $\forall i \in I$.
Pr_j	The parking rate of $p_j, \forall j \in J$.

4.3. Constraint optimization problem formulation

In this section, we develop the mathematical model of the parking assignment problem using CP. The main process is to define the variables and their corresponding domains, and then determine the problem constraints. The objective function must be stated if a criterion is to be optimized. We model our problem as a COP as follows:

a) The finite set of variables is defined by two sub-sets X and P

- $X = \{X_1, \dots, X_n\}$ represents the set of users who send their reservation request at the same time.
- $P = \{P_1, \dots, P_m\}$ represents the set of parking lots in the requested zone.

b) The possible values of variables X and P are defined by:

$$D(X_i) = \{1, \dots, m\}; D(P_j) = \{1, 0\}; \forall i \in I \text{ and } j \in J$$

c) The problem's constraints are defined by three constraints

- C1-parking capacity restriction

$$|X_i; i \in I : X_i = j| \leq Cp_j; \forall j \in J \quad (6)$$

The significance of this particular constraint cannot be overstated, as it plays a pivotal role in guaranteeing the integrity of the parking allocation system. This constraint serves as a safeguard to ensure that the number of users assigned to a specific parking spot, denoted as 'j,' does not surpass the available number of free parking spots in that designated area. In essence, it acts as a check and balance mechanism within the system. Without such a constraint, there would be a risk of overloading a parking area beyond its capacity, leading to potential chaos and inefficiency. By imposing this constraint, the system optimally manages the allocation process, preventing any scenario where users are assigned to a parking spot that is already at full capacity. This careful consideration of user-to-parking spot allocation aligns with the fundamental goal of ensuring a fair, organized, and efficient parking management system. It minimizes the likelihood of congestion and frustration while maximizing the utilization of available parking spaces, ultimately enhancing the overall user experience and the functionality of the parking facility.

- C2-parking slots restriction

$$P_{X_i} = 1; \forall i \in I \quad (7)$$

The concept of constraints plays an essential role in the smooth operation of systems involving the allocation of parking spaces to users. In this context, constraint effectively establishes a vital link between parking spaces and the people who use them. In the context of this allocation problem, a parking lot is denoted by "P". Each element of this parking lot, represented by "X_i", corresponds to a specific user, with "i" designating his position in the parking lot. The essence of this constraint lies in the fact that these variables must take the value "one". This singular value means that the parking space has been allocated to the "ith" user, which sums up a crucial aspect of the allocation process. By respecting this constraint, the parking space allocation system ensures that each user is precisely associated with the parking space allocated to them, thus improving the efficiency and organization of the parking space management process.

- C3-walking distance restriction

$$dis(Loc - Des(U_i), loc - p_j) \leq w_i; \forall i \in I \text{ and } \forall j \in J \quad (8)$$

d) The objective function

The overarching objective in this context is to achieve a minimization of a specific cost function. This cost function serves as a quantitative representation of the various factors and considerations at play within the problem-solving scenario. The fundamental aim is to reduce this cost function to its lowest possible value, which signifies the optimal outcome or solution:

$$F(X) = \min \sum_{i=1}^n d_{i,X_i} + dt_i Pr_{X_i} \quad (9)$$

Recall that,

- $|A|$ of a set A denote the cardinal (the number of element) of the set A.
- The $dis(x, y)$ is the distance between x and y. This constraint ensures that the distance from the proposed parking does not exceed the maximal walking distance for each user.

- $D = (d_{i,j})_{i,j \in N \times M} = \text{dis}(\text{Current} - \text{loc}(U_i), \text{Loc} - p_j)$ the matrix of the distance of user current location i to the parking j and Pr_{x_i} is parking prices to be paid by the user.

4.4. Mixed-integer linear programming formulation

As previously highlighted, our analysis revolves around two fundamental sets: the set of users or requests, denoted as 'U' and represented as $U = \{U_1, \dots, U_n\}$, and the set of available parking lots, denoted as 'P' and represented as $P = \{p_1, \dots, p_m\}$, all of which are conveniently situated within the same geographical zone. In the pursuit of an effective and well-structured parking allocation system, our model meticulously incorporates a series of constraints. These constraints are the guiding principles that shape the allocation process and ensure that it aligns with our overarching goals and requirements:

- Restriction on parking spaces: only one car may be parked in a lot at a time.

$$\sum_{j=1}^m x_{ij} = 1; \forall i \in I \quad (10)$$

- Parking capacity limitation: the number of vehicles assigned to the parking lot cannot exceed the lot's available capacity.

$$\sum_{i=1}^n x_{ij} \leq Cp_j; \forall j \in J \quad (11)$$

- Permissible walking distance: the proposed car park's distance from the users' destination must not exceed the maximum allowable walking distance.

$$\text{dis}(\text{Loc} - \text{Des}(U_i), p_j) \leq w_i; \forall i \in I \quad (12)$$

The solution is determined by the assignment matrix:

$$X = (x_{ij})_{N \times M}; x_{ij} \in \{0,1\} \quad (13)$$

At the core of our model lies a clear and compelling objective: the minimization of a function denoted as $F(X)$. This function serves as the compass that guides our decision-making process, and it is meticulously composed of two key components that capture the essence of our problem: travel distances and associated fees imposed on the vehicles involved. By seeking to minimize the function $F(X)$, we strike a balance between two critical dimensions: minimizing the physical and logistical costs associated with vehicle travel while also minimizing the financial costs incurred by users. In doing so, our model aims to provide an optimal and sustainable solution that enhances the overall efficiency and affordability of the parking allocation process, ultimately benefiting both users and the broader community.

$$F(X) = \min \sum_{i,j} (x_{ij} \cdot d_{ij} + x_{ij} \cdot dt_i \cdot Pr_j) \quad (14)$$

4.5. Solvers and optimizer

To solve the problem based on MILP, we use the Google OR-Tools solvers and Gurobi optimizer which have repeatedly been awarded in international competitions of MiniZinc solvers [35]. Google OR-Tools is an open-source, rapid, and adaptable software package that is used to solve combinatorial optimization problems. Dedicated to solve problems based on CSP, linear programming (LP), and MILP. Google OR-Tools [36] is created in C++ but also provides wrappers in Python, C#, and Java.

Gurobi optimizer is an advanced solver for mathematical programming, and the corresponding solvers were designed from the ground up to exploit modern architectures and multi-core processors, using the most advanced implementations of the latest algorithms including exhaustive search with the branch and bound algorithm. Gurobi optimizer [37] is a solver with efficient parallel algorithms for large-scale linear programs, quadratic programs, MILP, and mixed-integer quadratic CP. To solve the problem based on the COP approach, we adopt the Choco solver, which is a free and open-source Java library. Users model their problems declaratively by declaring all the constraints that must be satisfied in each solution by taking into account a cost function to be optimized. Then, the problem is solved by filtering algorithms based on constraints alternated with a search mechanism. The first version of Choco-solver in Java was released in 2003. Since then, it has undergone several major revisions, most recently in 2013 [38].

5. EXPERIMENT AND RESULT

5.1. Simulation parameters

We analyze the performance of the proposed models in this section by using solvers of COP and MILP approaches. The study was carried out in order to assess the effectiveness of the COP and MILP solvers using the city of Casablanca as a real use case. We performed a test of statistical significance in order to choose the best solver that was significantly efficient and ensure the scalability of the assignment system. Based on three basic settings, we built three datasets of three instances for each set, with varying degrees of difficulty: 30, 50, and 100 for the number of drivers looking for available parking in an area of 40 car parks. We found that the three solvers offer almost the same total cost of the objective function. This prompted us to do a second experiment to assess the scalability of the approach and see which solution is optimized by scaling to 200, 500, and 1000 users. The area considered in the test is assumed to consist of 100 parking lots.

The experiments presented here focus on how long it takes to solve the problem by using the computation times of our approach parameter and the scalability of the approach. In fact, in constrained programming solvers, optimization is done by computing improved solutions until an optimum is reached. So this can be thought of as solving the model multiple times while adding constraints along the way to prevent the solver from computing a dominant solution. The optimization proceeds as follows: each time a solution is found, store the value of the target variable and issue the cut. The cut is an additional constraint stating that the following solution must be (strictly) better than the previous one. The simulation data was taken from Casablanca city of Morocco, which is considered the largest and crowded city in Morocco and even in the Maghreb Region. The City of Casablanca shares a similar context with many cities in the world, which makes it possible to effectively draw important results.

5.2. Results and analysis

The experiments were conducted according to the number of requests that created three instances (30, 50, and 100). In each instance, a random number is used to determine the users' destination. The drivers' locations are produced within a 300-meter radius of the destination. The driving distance (between the driver's position and the parking lot) and the walking distance (between the parking lot and the destination) can then be determined. The performance of the proposed models is assessed using four parameters:

- Total cost of the objective function: this gives the overall cost values of distance plus the execution time. The lower value of this parameter is preferable since it can eliminate unnecessary time and energy consumption and detrimental effects like air pollution.
- Execution time by seconds.
- Rate cost: the cost of parking according to the stay time of users
- Distance cost: the sum of the driven distance of users from their current positions to the proposed parking lots.

In Table 2, the significance test and performance measure results are provided and succinctly described.

Table 2. Solvers comparison of the first test with 30, 50 and 100 users

Solvers	Instance	Total cost	Time (s)	Rate cost (dhs)	Distance cost (m)
Google OR-Tools	1=30	7,091	0.033	548	6,543
	2=50	13,231	0.054	822	12,461
	3=100	31,231	0.127	1,542	26,689
Gurobi optimizer	1=30	7,091	0.014	548	6,543
	2=50	13,283	0.017	822	12,461
	3=100	31,231	0.032	1,542	29,689
Choco	1=30	7,091	0.3	548	6,543
	2=50	13,283	0.4	822	12,461
	3=100	31,536	0.5	1,561	29,975

Table 3 shows that the results achieved by the solvers are close. Indeed the three solvers give similar results in the two instances (30 and 50 users) regarding the total cost and the sum of each component of the objective function. The third instance of 100 users shows a nuance regarding the total cost. The Choco solver offers a higher total cost than the Gurobi and OR tools solvers. Furthermore, the best running time for the Gurobi solver, which is presented in bold, is readily apparent. The results of the first experiment do not show a significant difference between the solvers, which prompted us to verify if all approaches meet the scalability. This is why we proposed to increase the number of requests set by fixing the number of parking lots in the target zone at 100 parking lots. In the second simulation, we created three new instances based on

three different setups: 200, 500, and 1000 parking-seeking drivers. The results obtained out of 30 independent runs are shown in Table 3. The comparison is based on the following parameters:

- Min-max total cost/run time: the minimum-maximum total cost of the objective function (the driving distance + the rate cost) of the 30 execution of the problem. The minimum execution time obtained from 30 runs.
- Average of total cost/run time: the sum of the total costs obtained during the 30 runs divided by the number of executions (30)/the sum of the run times obtained during the 30 runs divided by the number of executions.
- Type of solution: there are two types of solutions: i) optimal, which offers the best solution while respecting constraints and ii) satisfiability (often abbreviated SAT) that respects constraints without searching for the best solution.

The result of the second experiment showed that the Choco solver reached only satisfiability solutions, contrary to Gurobi and OR-tools solvers that offer optimal solutions in a reasonable execution time. Moreover, in the last instance (1000 users); the Choco solver did not even manage to find a satisfiable solution. In terms of execution time, we can plainly see that Gurobi optimizer is the best in all three cases. In addition, the results of the average of the total cost obtained by Gurobi and OR-tools are close and much better compared to those of Choco. The results exhibit that the Choco solver is poor in solving assignment problems, when it comes to large datasets, it does not even give a response when the number of users exceeds a thousand. This means that Gurobi and OR tools solvers are much better in terms of scalability. Additionally, among these, we can see that Gurobi is faster to run and find the optimal solution which is an important criterion for real-time parking lot assignment.

Table 3. Solvers comparison of the second test with 200, 500, and 1000 users

Solvers	Instance	Minimum		Maximum		Average of total cost	Average of run time (s)	Type of solution
		Total cost	Run time (s)	Total cost	Run time (s)			
Google	1=200	212,423	4.059	222,494	5.218	216,396.61	4.248	Optimal
OR-Tools	2=500	537,141	13.64	458,551	14.798	512,215.66	14.132	
	3=1000	1,074,556	28.8	1,091,819	28.810	1,080,809	26.809	
Gurobi optimizer	1=200	211,773	0.166	225,213	0.156	216,949.83	0.154	Optimal
	2=500	531,710	0.421	547,389	0.374	538,944	0.451	
	3=1000	1,073,680	0.859	1,093,034	0.890	1,082,698	0.874	
Choco	1=200	443,166	67.55	500,903	68.45	465,805.04	67.63	SAT
	2=500	1,146,909	376.5	1,205,056	374.41	1,169,615.15	375.18	
	3=1000	-	-	-	-	-	-	

6. CONCLUSION

In this work, we have proposed a complete description of our reservation system based on the MAS for intelligent parking management for groups of drivers reserving at the same time. A modeling of the smart parking assignment problem based on CP and MILP is proposed in this paper. In the resolution of CP, we used a solver called Choco solver. It was chosen because it is an open-source library. It can manipulate a large number of variables. In this library, more than 70 constraints are defined. Moreover, for the resolution of the MILPs we have chosen two best solvers according to the MiniZinc 2022 competition named Gurobi optimizer and Google OR-Tools solvers. The results of the two-simulation test showed that the Gurobi optimizer performs well in terms of scalability. As a future direction, we intend to add more constraints to the problem such as the preferred parking places. We also plan to propose an algorithm for solving constrained assignment problems based on other optimization algorithms.

REFERENCES




- [1] S. Charmonman and P. Mongkhonvanit, "Internet of things in e-business," in *The 10th International Conference on e-Business (iNCEB2015)*, 2015, pp. 1–9.
- [2] S. Kumar and P. R. Cohen, "Towards a fault-tolerant multi-agent system architecture," in *Proceedings of the International Conference on Autonomous Agents*, ACM, 2000, pp. 459–466, doi: 10.1145/336595.337570.
- [3] N. El Khalidi, F. Benabbou, and N. Sael, "Distributed parking management architecture based on multi-agent systems," *IAES International Journal of Artificial Intelligence*, vol. 10, no. 4, pp. 801–809, 2021, doi: 10.11591/ijai.v10.i4.pp801-809.
- [4] E. Simhon, C. Liao, and D. Starobinski, "Smart parking pricing: A machine learning approach," in *2017 IEEE Conference on Computer Communications Workshops, INFOCOM WKSHPs 2017*, IEEE, 2017, pp. 641–646, doi: 10.1109/INFOCOMW.2017.8116452.
- [5] X. Zhang, C. Zhao, F. Liao, X. Li, and Y. Du, "Online parking assignment in an environment of partially connected vehicles: a multi-agent deep reinforcement learning approach," *Transportation Research Part C: Emerging Technologies*, vol. 138, pp. 1–22, 2022, doi: 10.1016/j.trc.2022.103624.

- [6] S. Abidi, S. Krichen, E. Alba, and J. M. Molina, "A new heuristic for solving the parking assignment problem," *Procedia Computer Science*, vol. 60, no. 1, pp. 312–321, 2015, doi: 10.1016/j.procs.2015.08.132.
- [7] K. Marriott and P. J. Stuckey, *Programming with constraints: an introduction*. Cambridge, Massachusetts: The MIT Press, 1998, doi: 10.7551/mitpress/5625.001.0001.
- [8] T. Kleinert, M. Labbé, I. Ljubić, and M. Schmidt, "A survey on mixed-integer programming techniques in bilevel optimization," *EURO Journal on Computational Optimization*, vol. 9, pp. 1–21, 2021, doi: 10.1016/j.ejco.2021.100007.
- [9] X. Zhao, K. Zhao, and F. Hai, "An algorithm of parking planning for smart parking system," in *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, IEEE, 2015, pp. 4965–4969, doi: 10.1109/WCICA.2014.7053556.
- [10] O. T. T. Kim, N. H. Tran, C. Pham, T. Leanh, M. T. Thai, and C. S. Hong, "Parking assignment: minimizing parking expenses and balancing parking demand among multiple parking lots," *IEEE Transactions on Automation Science and Engineering*, vol. 17, no. 3, pp. 1320–1331, 2020, doi: 10.1109/TASE.2019.2948200.
- [11] M. Ratli, A. A. El Cadi, B. Jarboui, and A. Artiba, "Dynamic assignment problem of parking slots," in *2019 International Conference on Industrial Engineering and Systems Management (IESM)*, Shanghai, China, 2019, pp. 1–6, doi: 10.1109/IESM45758.2019.8948174.
- [12] Y. Zhang, Z. Zhang, and S. Teng, "Adaptive ant colony optimization for solving the parking lot assignment problem," in *ICNC-FSKD 2018 - 14th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, IEEE, 2018, pp. 91–96, doi: 10.1109/FSKD.2018.8687175.
- [13] A. Hakeem, N. Gehani, X. Ding, R. Curtmola, and C. Borcea, "On-the-fly curbside parking assignment," in *MobiCASE 2016 - 8th EAI International Conference on Mobile Computing, Applications and Services*, ACM, 2016, pp. 1–10, doi: 10.4108/eai.30-11-2016.2267101.
- [14] E. Alfonsetti, P. C. Weeraddana, and C. Fischione, "A semi distributed approach for min-max fair car-parking slot assignment problem," *arXiv-Mathematics*, pp. 1–15, 2014.
- [15] M. Mladenović, T. Delot, G. Laporte, and C. Wilbaut, "The parking allocation problem for connected vehicles," *Journal of Heuristics*, vol. 26, no. 3, pp. 377–399, 2020, doi: 10.1007/s10732-017-9364-7.
- [16] P. Sadhukhan, "An IoT-based e-parking system for smart cities," in *2017 International Conference on Advances in Computing, Communications and Informatics, ICACCI 2017*, IEEE, Sep. 2017, pp. 1062–1066, doi: 10.1109/ICACCI.2017.8125982.
- [17] A. O. Kotb, Y. C. Shen, X. Zhu, and Y. Huang, "iParker-a new smart car-parking system based on dynamic resource allocation and pricing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 17, no. 9, pp. 2637–2647, Sep. 2016, doi: 10.1109/TITS.2016.2531636.
- [18] Y. Geng and C. G. Cassandras, "A new smart parking system infrastructure and implementation," *Procedia - Social and Behavioral Sciences*, vol. 54, pp. 1278–1287, 2012, doi: 10.1016/j.sbspro.2012.09.842.
- [19] Y. Rahayu and F. N. Mustapa, "A secure parking reservation system using GSM technology," *International Journal of Computer and Communication Engineering*, pp. 518–520, 2013, doi: 10.7763/ijcce.2013.v2.239.
- [20] K. C. Mouskos, J. Tavantzis, D. Bernstein, and A. Sansil, "Mathematical formulation of a deterministic parking reservation system (PRS) with fixed costs," in *2000 10th Mediterranean Electrotechnical Conference. Information Technology and Electrotechnology for the Mediterranean Countries*, Lemesos, Cyprus, 2000, pp. 648–651, doi: 10.1109/MELCON.2000.880017.
- [21] S. Hashimoto, R. Kanamori, and T. Ito, "Auction-based parking reservation system with electricity trading," in *Proceedings - 2013 IEEE International Conference on Business Informatics, IEEE CBI 2013*, IEEE, 2013, pp. 33–40, doi: 10.1109/CBI.2013.14.
- [22] C. Shiyao, W. Ming, L. Chen, and R. Na, "The research and implement of the intelligent parking reservation management system based on ZigBee technology," in *Proceedings - 2014 6th International Conference on Measuring Technology and Mechatronics Automation, ICMTMA 2014*, IEEE, 2014, pp. 741–744, doi: 10.1109/ICMTMA.2014.182.
- [23] S. Abidi, S. Krichen, E. Alba, and J. M. M. Bravo, "A hybrid heuristic for solving a parking slot assignment problem for groups of drivers," *International Journal of Intelligent Transportation Systems Research*, vol. 15, no. 2, pp. 85–97, 2017, doi: 10.1007/s13177-016-0123-1.
- [24] M. Caliskan, A. Barthels, B. Scheuermann, and M. Mauve, "Predicting parking lot occupancy in vehicular ad hoc networks," in *IEEE Vehicular Technology Conference*, IEEE, Apr. 2007, pp. 277–281, doi: 10.1109/VETECS.2007.69.
- [25] N. H. H. M. Hanif, M. H. Badiozaman, and H. Daud, "Smart parking reservation system using short message services (SMS)," in *2010 International Conference on Intelligent and Advanced Systems*, IEEE, Jun. 2010, pp. 1–5, doi: 10.1109/ICIAS.2010.5716179.
- [26] H. Errousso, J. El Ouadi, E. A. A. Alaoui, S. Benhadou, and H. Medromi, "A hybrid modeling approach for parking assignment in urban areas," *Journal of King Saud University - Computer and Information Sciences*, vol. 34, no. 6, pp. 2405–2418, 2022, doi: 10.1016/j.jksuci.2020.11.006.
- [27] L. Zhang, X. Li, J. Huang, Y. Shen, and D. Wang, "Vision-based parking-slot detection: A benchmark and a learning-based approach," *Symmetry*, vol. 10, no. 3, pp. 1–18, 2018, doi: 10.3390/SYM10030064.
- [28] G. Barbarosoglu and D. Ozgur, "A tabu search algorithm for the vehicle routing problem," *Computers and Operations Research*, vol. 26, no. 3, pp. 255–270, 1999, doi: 10.1016/S0305-0548(98)00047-1.
- [29] M. Srinivas and L. M. Patnaik, "Genetic algorithms: a survey," *Computer*, vol. 27, no. 6, pp. 17–26, 1994, doi: 10.1109/2.294849.
- [30] A. Chakraborty and A. K. Kar, "Swarm intelligence: a review of algorithms," in *Modeling and Optimization in Science and Technologies*, Springer International Publishing, 2017, pp. 475–494, doi: 10.1007/978-3-319-50920-4_19.
- [31] A. H. Land and A. G. Doig, "An automatic method of solving discrete programming problems," *Econometrica*, vol. 28, no. 3, pp. 497–520, 1960, doi: 10.2307/1910129.
- [32] O. K. Gupta and A. Ravindran, "Branch and bound experiments in convex nonlinear integer programming," *Management Science*, vol. 31, no. 12, pp. 1533–1546, 1985, doi: 10.1287/mnsc.31.12.1533.
- [33] M. Wooldridge and N. R. Jennings, "Intelligent agents: Theory and practice," *The Knowledge Engineering Review*, vol. 10, no. 2, pp. 115–152, 1995, doi: 10.1017/S0269888900008122.
- [34] B. Liang, Y. Li, J. Bi, C. Ding, and X. Zhao, "An improved adaptive parallel genetic algorithm for the airport gate assignment problem," *Journal of Advanced Transportation*, vol. 2020, pp. 1–17, 2020, doi: 10.1155/2020/8880390.
- [35] MiniZinc Team, "MiniZinc-challenge," *MiniZinc Team*, 2014. [Online]. Available: <https://www.minizinc.org/challenge2022/results2022.html> (accessed May 20, 2023).
- [36] Mizux, "OR-tools-Google optimization tools," *Github*. [Online]. Available: <https://github.com/google/or-tools> (accessed May 26, 2023).




- [37] Gurobi, "Gurobi/optimizer - Docker image," *Hub Docker*. [Online]. Available: <https://hub.docker.com/r/gurobi/optimizer> (accessed May 26, 2023).
- [38] C. P. Homme and J. -G. Fages, "Choco-solver: A Java library for constraint programming," *Journal of Open Source Software*, vol. 7, no. 78, pp. 1–6, 2022, doi: 10.21105/joss.04708.

BIOGRAPHIES OF AUTHORS






Nihal Elkhaldi    received the Master in Mathematical Sciences from the Ben M'Sick Faculty of Sciences, Hassan II University of Casablanca, Morocco, in 2016, where she is currently pursuing the Ph.D. in Computer Science. Her research interests include smart cities, intelligent transportation systems, multi-agent systems, constraint programming, and combinatorial optimization. She can be contacted at email: elkhaldi.nihal@gmail.com.



Faouzia Benabbou    is a professor of Computer Science and member of Compute Science and Information Processing Laboratory. She is head of the team "Cloud Computing, Network and Systems Engineering (CCNSE)". She received his Ph.D. in Computer Science from the Faculty of Sciences, University Mohamed V, Morocco, 1997. His research areas include cloud computing, data mining, machine learning, and natural language processing. She can be contacted at email: faouzia.benabbou@univh2c.ma.



Nawal Sael    is a professor of Computer Science and member of Computer Science and Information Processing Laboratory at Faculty of Science Ben M'sik (Casablanca, Morocco). She received her Ph.D. in Computer Science from the Faculty of Sciences, University Hassan II Casablanca, Morocco, 2013 and her engineer degree in software engineering from ENSIAS, Morocco, in 2002. Here research interests include data mining, educational data mining, machine learning, deep learning, and internet of things. She can be contacted at email: saelnawal@hotmail.com.