

# Location-Allocation of Social Amenities Based on Constraint Programming

Dessislava Petrova-Antonova(✉) and Mihail Stamenov

GATE Institute, Sofia University "St. Kliment Ohridski", Sofia, Bulgaria
{dessislava.petrova,mihail.stamenov}@gate-ai.eu

**Abstract.** The social amenities allocation problem is a challenge in urban planning, requiring an optimal distribution of their locations to accommodate demand while ensuring accessibility. This paper presents a novel approach to solving this problem using Constraint Programming (CP). The constraint modeling language MiniZinc is used to formulate the problem, and the Gecode and Chuffed solvers are utilized to identify feasible allocations. To improve computational efficiency, a bounding strategy is applied, determining upper and lower limits on the number of required kindergartens, leveraging a 0/1 knapsack model and combinatorial search techniques. In addition, a clustering-based preprocessing step is introduced that enhances the selection of candidate kindergartens.

The proposed approach is evaluated on real-world data from Sofia, Bulgaria, demonstrating its ability to produce optimal allocations for kindergartens while maintaining computational feasibility. It is implemented within the urban planning use case of the Urban Digital Twin pilot project of the city. The results highlight the advantages of CP for solving complex spatial allocation problems, offering a scalable framework for future applications in urban infrastructure planning.

**Keywords:** Constraint Programming · Location-Allocation of Social Amenities · Spatial Optimization · Urban Planning

## 1 Introduction

Urban planning plays a crucial role in shaping the livability and functionality of cities, with social amenities serving as key determinants of residents' well-being. Access to essential services such as schools, healthcare facilities, and recreational spaces influences quality of life, economic opportunities, and social equity. Among these, kindergartens are particularly significant as they provide early childhood education and facilitate the participation of the parents' workforce. However, ensuring equitable access to kindergartens poses a major challenge for urban planners, requiring careful consideration of spatial distribution, demographic trends, and infrastructure constraints. Effective planning of these facilities is essential to foster inclusive and sustainable urban environments.

Optimizing kindergarten allocation is a complex problem that requires advanced computational techniques to balance accessibility, capacity, and efficiency. Traditional approaches often struggle with the combinatorial nature of

© The Author(s), under exclusive license to Springer Nature Switzerland AG 2026
O. Gervasi et al. (Eds.): ICCSA 2025 Workshops, LNCS 15897, pp. 438–455, 2026.
https://doi.org/10.1007/978-3-031-97660-5_29

the problem, where multiple factors must be considered simultaneously, such as population density, travel distance, and available infrastructure. Modern urban planning increasingly relies on data-driven methodologies to improve decision-making and resource allocation. In this context, CP provides a powerful framework for solving large-scale spatial optimization problems. Using CP, optimal kindergarten placements can be identified, minimizing inequality in access while ensuring efficient use of resources.

The kindergarten allocation problem is a critical optimization challenge in Sofia, Bulgaria. The objective is to determine the optimal locations to build new kindergartens from a predefined set of possible sites while efficiently distributing children to these kindergartens. The allocation must consider constraints such as capacity limits, proximity, and fairness in distribution. This problem is particularly relevant in densely populated areas where the demand for kindergarten placements exceeds the available slots, requiring an optimal location-allocation strategy.

In this paper, a novel approach is proposed to solve the kindergarten location-allocation problem using CP. The problem is formalized based on MiniZinc, a high-level constraint modelling language, and leverages the power of the Gecode and Chuffed solvers to find optimal solutions. To enhance computational efficiency, Python's multiprocessing module is employed, enabling parallelized execution where each process independently explores solutions for different numbers of required kindergartens. This parallelization significantly accelerates the solution process, making it feasible to handle larger problem instances. The proposed approach provides a scalable and efficient solution to the allocation problem, offering a practical framework that can be adapted to various real-world scenarios. Through experimental evaluation, its effectiveness is evaluated in terms of solution quality and computational performance.

The remainder of the paper is structured as follows. Section 2 introduces the related work. Section 3 is dedicated to the formulation of the problem and the description of the methods used. Section 4 presents the implementation of the proposed approach, while Sect. 5 describes the results obtained. Finally, Sect. 6 concludes the paper.

## 2 Related Work

This section presents the related work, including the traditional methods like integer programming, deferred- acceptance Algorithms and capacitated P-median models, followed by an alternative to the proposed approach models and finishing with CP applications.

### 2.1 Traditional Methods

*Integer Programming (IP)* models have been extensively used for kindergarten allocation, optimizing assignments while considering constraints such as proximity and capacity. For example, [1] applied IP models in Norway to automate the

allocation process, demonstrating improved results over manual methods. However, IP models can become computationally expensive when dealing with large-scale problems, necessitating heuristic methods to improve scalability. *Deferred-Acceptance (DA) Algorithms* are commonly used for stable matching problems, where allocation is based on predefined priorities such as sibling attendance and distance. These algorithms have been successfully implemented in Estonian municipalities, as shown by [2], improving fairness and efficiency. However, balancing trade-offs between fairness and overall efficiency remains a challenge, and the effectiveness of the method depends on the priority rules used.

*Capacitated P-Median Models* focus on minimizing the overall commute distances for children while ensuring that kindergarten capacities are not exceeded. [3] demonstrated the application of such models in school allocation systems in Greece. While effective in optimizing proximity-based allocation, these models require extensive preprocessing of data and are highly sensitive to parameter settings, making implementation complex. *Hybrid Evolutionary Methods*, such as combining genetic algorithms (GA) with traditional optimization techniques, have been explored for capacitated location-allocation problems. [4] proposed a hybrid GA method that improves solution quality while maintaining computational efficiency. However, these methods require careful tuning of parameters and may struggle with convergence, especially for large problem instances.

The *Spatial Decision Support Systems (SDSS)* combine Geographic Information Systems (GIS) with optimization techniques to assist in making location-allocation decisions. [3] demonstrated the integration of SDSS for school placement problems, leveraging spatial data to optimize placements. However, implementing SDSS requires expertise in both spatial analysis and optimization, making it a resource-intensive approach.

### 2.2 Alternative Methods and Their Limitations

Several other methods have been explored in the literature for solving similar allocation problems.

The *Genetic Algorithms (GA)* are used to encode potential solutions as chromosomes and evolve better allocations over generations [5,7]. However, GAs require careful parameter tuning and can be computationally expensive, making them less reliable for large-scale problems [6]. Data Envelopment Analysis (DEA) is applied to evaluate the efficiency of kindergarten allocation by analyzing input-output relationships [8]. However, DEA primarily serves as a benchmarking tool rather than a direct solution method. *Particle Swarm Optimization (PSO)* simulates the movement of particles in a search space to optimize allocation decisions [8]. However, PSO struggles with handling discrete constraints like capacity limits and can converge prematurely to suboptimal solutions.

*Mixed-integer programming (MIP)* has been used for modeling the problem as a set of linear equations, optimizing objectives like travel distance and capacity balancing [1]. Despite its accuracy, MIP becomes computationally infeasible for large-scale problems without heuristic adaptations.

### 2.3 Constraint Programming

CP is highly effective in solving various types of optimization and decision-making problems. There are several areas, where CP excels.

In the context of *Scheduling Problems*, CP is widely used in scheduling tasks, such as nurse scheduling, operating room scheduling, and production scheduling. It provides flexibility in handling complex constraints and improves efficiency in generating optimal schedules according to [13–15]. For *Resource Allocation*, CP is effective in resource allocation problems, such as tool allocation in manufacturing systems [16] and resource scheduling in high-performance computing [17]. It handles constraints like resource capacities and deadlines efficiently according to [18]

In *Combinatorial Optimization*, CP is powerful for solving combinatorial problems [19], including the traveling salesman problem, knapsack problem, and crew rostering [20]. It ensures optimal solutions through systematic search and constraint propagation [21]. Regarding the *Planning and Decision-Making*, CP is used in planning tasks, such as maintenance scheduling for heritage buildings and traffic incident management, where it optimizes resource use and meets complex constraints [22,23]. For *Hybrid Approaches* CP is often integrated with other techniques, such as reinforcement learning and integer programming, to enhance its performance in solving large-scale problems [24].

Following the advances of CP, the proposed approach aims to provide a scalable and efficient solution to the kindergarten allocation problem, offering a practical framework that can be adapted to various real-world scenarios.

## 3 Study Area and Data

### 3.1 Study Area

The study area is the most populous city in Bulgaria, with a population of more than 1 221 785 in the year 2020, according to [25]. This city is spread over 245.5 km2 [26]. It builds upon previous work, extending the spatial analysis of kindergarten accessibility in Sofia [27,28]. The data consists of three main components: the pedestrian network, residential buildings, and kindergartens.

### 3.2 Datasets

The *pedestrian network* serves as the foundation for accessibility analysis. It was developed by Sofiaplan for network analysis purposes, ensuring proper connectivity using GIS topology checks. Previous work on network analysis details the preprocessing steps, including calculating travel distances and times while resolving overlapping vertices.

To estimate demand, the number of residents per statistical unit was obtained from the National Statistical Institute. Using cadastral data, the gross floor area (GFA) of *residential buildings* is calculated and distributed demographic data accordingly. The number of children per building was estimated based on

age distribution data, resulting in a floating-point approximation of potential kindergarten demand.

On the supply side, due to the lack of precise capacity data, the potential *kindergarten* capacities are estimated based on lot area and legal regulations. This estimation provides insight into the maximum possible capacity expansion without acquiring additional land. The spatial distribution of kindergartens and their potential capacities form the basis for the subsequent allocation analysis.

### 3.3 Data Preprocessing

To integrate the datasets into a unified framework, residential buildings and kindergartens were mapped onto the pedestrian network as graph vertices. Dijkstra's algorithm was used to compute the shortest walking distances from each building to every kindergarten. Distances exceeding 15 min were set to a high constant value (1,000,000) to effectively represent inaccessibility. The final dataset consists of:

- A distance matrix containing walking distances between buildings and kindergartens.
- An array representing the capacities of potential kindergartens.
- An array indicating the estimated number of children per residential block.

The complete dataset comprises 83,089 buildings and 26,084 potential kindergarten placements. From this dataset, two smaller subsets were extracted: a **neighborhood-level dataset**, consisting of 348 buildings and 59 kindergartens, and a **street-level dataset**, containing 17 buildings and 5 kindergartens. All experiments in this study are conducted exclusively on these smaller datasets.

Figure 1 and Fig. 2 illustrate the spatial distribution of buildings and kindergartens in the neighborhood-level and street-level datasets, respectively.

## 4 Problem Formulation and Methods

The problem of selecting optimal locations for kindergartens in an urban setting is considered, with the objective of **minimizing the number of kindergartens constructed while simultaneously reducing the total walking distance for children**.

The problem can be formally described as follows:

- Let $B$ denote a set of buildings, each with a given number of children.
- Let $K$ denote a set of potential locations where kindergartens can be located.
- Let $d_{ij}$ denotes the distance from each building $i$ to each potential kindergarten location $j$.
- Let $\bar{C}_j$ denotes maximum capacity of kindergarten $j$ and the number of children in building $i$ is given by $c_i$.

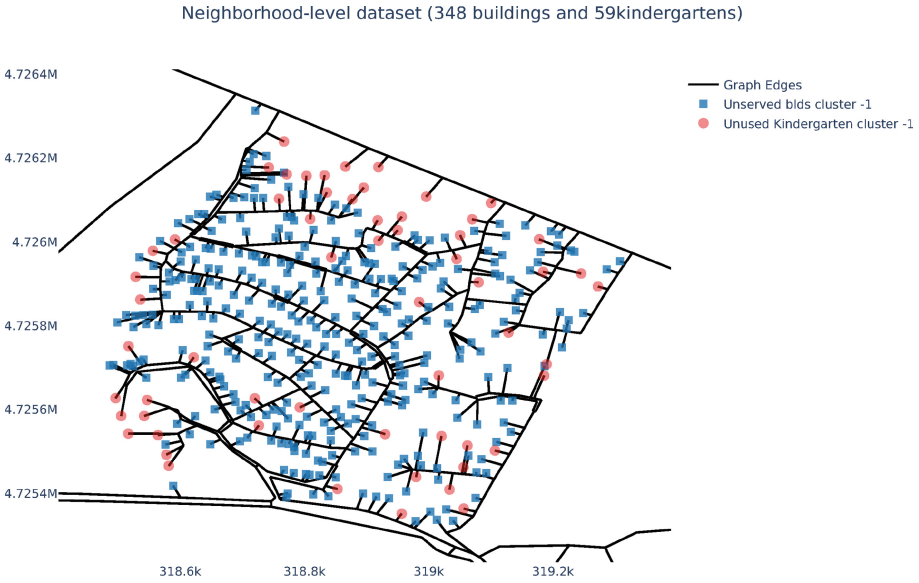


**Fig. 1.** Spatial representation of the neighborhood-level dataset (348 buildings and 59 kindergartens).




**Fig. 2.** Spatial representation of the street-level dataset (17 buildings and 5 kinder-gartens).

The objective is to **select a subset $K'$ of $K$** and **assign each building to exactly one selected kindergarten** while minimizing the total distance the children have to walk. It can be formalized as two consecutive **p-median problems**. A function-based perspective can also be formulated, where the objective is to identify a function defined as follows:

$$\mathrm{f} : B \rightarrow K' \tag{1}$$

where $K' \subseteq K$ is the set of selected kindergartens. This function must respect feasibility constraints while optimizing the objective.

### 4.1 p-Median Formulation

Since the objective is to minimize both the number of newly established kindergartens and the total walking distance for all children, a two-step optimization approach is necessary. First, the minimum number of required kindergartens is determined. Once this value is obtained, it is incorporated as a constraint in the subsequent optimization step, which focuses on minimizing the total distance between children and their assigned kindergartens. Both steps rely on the same set of decision variables:

- Let $x_{ij} \in \{0, 1\}$ is 1 if children from building $i$ are assigned to kindergarten $j$, 0 otherwise.
- Let $y_j \in \{0, 1\}$ is 1 if kindergarten is built at location $j$, 0 otherwise.

**Minimization of New Kindergartens.** The number of new kindergartens is minimized as follows:

$$\min \sum_{j \in K} y_j \tag{2}$$

where $y_j$ is a binary decision variable indicating whether a kindergarten is established at location $j$ ($y_j = 1$) or not ($y_j = 0$). The set $K$ represents the potential locations where kindergartens can be built.

Each building must be assigned to exactly one kindergarten, which is represented by the following constraint:

$$\sum_{j \in K} x_{ij} = 1, \quad \forall i \in B \tag{3}$$

where $x_{ij}$ is a binary decision variable indicating whether building $i$ is assigned to kindergarten $j$ ($x_{ij} = 1$) or not ($x_{ij} = 0$). The set $B$ represents the set of buildings, and $K$ denotes the set of potential kindergarten locations.

The buildings must be assigned only to selected kindergarten plots where kindergartens will be built, ensuring that no assignments are made to unused plots, as defined by the following constraint:

$$x_{ij} \leq y_j, \quad \forall i \in B, \forall j \in K \tag{4}$$

Moreover, the total number of children assigned to a kindergarten $j$ must not exceed its capacity $\bar{C}_j$, and if $y_j = 0$ (the kindergarten is not built), then no children can be assigned to it, which is represented by a third constraint as follows:

$$\sum_{i \in B} d_i x_{ij} \leq \bar{C}_j y_j, \quad \forall j \in K \tag{5}$$

where $d_i$ represents the number of children in building $i$, and $\bar{C}_j$ denotes the maximum capacity of kindergarten $j$. The constraint ensures that the total demand for kindergarten seats at a specific location does not exceed its capacity. Additionally, when $y_j = 0$, the right-hand side of the inequality becomes zero, enforcing that no children are assigned to plots where kindergartens are not constructed. This effectively prevents the allocation of buildings to non-selected plots while ensuring that capacity constraints are respected. Although constraint (5) inherently prevents assignments to unbuilt kindergartens by ensuring that the total number of assigned children does not exceed the available capacity, it does not explicitly forbid individual assignments to non-selected plots. The constraint (4) directly enforces that a building cannot be assigned to a kindergarten unless that kindergarten is constructed. Without this constraint, an assignment could still be technically feasible as long as the total assigned children remain within a capacity of zero, but this does not explicitly restrict individual building assignments. Therefore, constraint (4) is necessary to establish a strict logical dependency between $x_{ij}$ and $y_j$ at the assignment level.

**Minimization of Distances.** After optimizing the first problem, the number of required kindergartens is determined. To incorporate this result into the second optimization step, an additional parameter, $p$, is introduced, representing the exact number of kindergartens to be built.

The total weighted walking distance is minimized through the following objective function:

$$\min \sum_{i \in B} \sum_{j \in K} c_i d_{ij} x_{ij} \tag{6}$$

where $c_i$ denotes the number of children in building $i$, $d_{ij}$ represents the walking distance from building $i$ to kindergarten $j$, and $x_{ij}$ is a binary decision variable indicating whether the children from building $i$ are assigned to kindergarten $j$ ($x_{ij} = 1$) or not ($x_{ij} = 0$).

To enforce that exactly $p$ kindergartens are built, a new constraint is introduced:

$$\sum_{j \in K} y_j = p \tag{7}$$

where $y_j$ is a binary decision variable indicating whether a kindergarten is constructed at location $j$ ($y_j = 1$) or not ($y_j = 0$). By integrating Equation (7) into the optimization model, the solution guarantees that exactly $p$ kindergartens are selected while minimizing the overall walking distance, as defined in o (6).

### 4.2 Constraint Programming for Feasibility of a Given Kindergarten Count

CP can struggle with scalability when dealing with large or highly complex problems, as the computational effort required for constraint propagation and search can grow exponentially with problem size. For this reason, no direct CP formulation is provided for minimizing the number of kindergartens needed.

Instead, different methods were used to determine bounds for $p$, the required number of kindergartens. A binary search is performed over $p$ to efficiently find the minimal feasible number of kindergartens. In this formulation, $p$ is treated as a parameter, and the model verifies whether a given number of kindergartens is sufficient to accommodate all children while satisfying the problem constraints.

**Decision Variables.** The initial approach to formulating the CP model considered an array of size $|B|$, where each element represented the index of the assigned kindergarten. However, this formulation resulted in a search space of $|K|^{|B|}$, leading to significant computational complexity. Furthermore, enforcing a global constraint to limit the number of utilized kindergartens increased propagation overhead, further complicating the model's efficiency.

To mitigate these challenges, a more structured approach was adopted by introducing two separate arrays. The first array tracks the selected kindergartens, while the second assigns buildings to the indices of these selected kindergartens. Given that the number of kindergartens to be used, denoted as $p$, is predefined, the length of the first array is explicitly set to $p$, ensuring that exactly $p$ kindergartens are utilized in the final allocation.

Formally, each building $i \in B$ is assigned to one of the $p$ selected kindergartens, represented by the variable:

$$x_i \in \{1, \ldots, p\}, \quad \forall i \in B \tag{8}$$

where $x_i$ denotes the index of the kindergarten assigned to building $i$. Simultaneously, the indices of the selected kindergartens are stored in an array of length $p$, where each entry corresponds to a kindergarten from the original set of potential locations:

$$y_j \in K, \quad \forall j \in \{1, \ldots, p\} \tag{9}$$

This formulation significantly reduces the search space while maintaining flexibility in the assignment process, ensuring computational feasibility while adhering to the problem constraints.

**Constraints.** Two constraints are defined related to the kindergarten capacities and uniqueness of the selected kindergartens. The kindergarten capacity is constrained as follows:

$$\sum_{\{i \in B | x_i = j\}} c_i \leq C_{y_j}, \quad \forall j \in \{1, \ldots, p\} \tag{10}$$

where the total number of children assigned to kindergarten $j$ must not exceed its capacity $c_j$.

The second constraint ensures that each selected kindergarten is unique (no duplicates), enforcing an ordering among the selected kindergartens:

$$y_j < y_k, \quad \forall j < k, \quad j, k \in \{1, \ldots, p\} \tag{11}$$

where $y_j$ and $y_k$ represent the indices of the selected kindergartens from the set of potential locations. This constraint guarantees that the kindergartens are chosen in a strictly increasing order, preventing the same kindergarten from being selected multiple times.

By enforcing this ordering, the search space is significantly reduced. Without this constraint, the model would differentiate between permutations of the same set of selected kindergartens, treating different orderings of the same selection as distinct solutions. This would result in a search space of:

$$|B|^p \times P(p, |K|) = |B|^p \times \frac{|K|!}{(|K| - p)!} \tag{12}$$

where $|B|^p$ accounts for the possible assignments of buildings to the selected kindergartens, and $P(p, |K|)$ represents the number of ways to arrange $p$ kindergartens from the set $K$.

By enforcing an increasing order among selected kindergartens, we eliminate duplicate permutations and instead consider only unique combinations, reducing the search space to:

$$|B|^p \times C(p, |K|) = |B|^p \times \frac{|K|!}{p!(|K| - p)!} \tag{13}$$

where $C(p, |K|)$ represents the number of ways to choose $p$ kindergartens from $K$ without considering order. This reduction significantly improves computational efficiency, as the number of valid kindergarten selections decreases from permutations to combinations, leading to a more manageable search space in the constraint programming model.

**Satisfaction vs. Objective Function.** Unlike the previous formulations, there is no need for an objective function here since the purpose of the model is to satisfy the constraints proving that the given number of kindergartens used is a possible solution.

## 5 Implementation of Methods

The proposed approach is implemented in Python, utilizing several libraries to handle different aspects of the problem. GeoPandas is used for processing GeoPackage files, while NetworkX facilitates graph-based operations. NumPy supports matrix computations, and clustering is performed using the K-Means algorithm from Scikit-learn. The constraint programming model is formulated in MiniZinc and executed via the corresponding Python library.

### 5.1 Data Preprocessing

The first step in data processing involves computing the shortest paths between all buildings and kindergartens using Dijkstra's algorithm. Given that the problem is approached through constraint programming, it is crucial to establish meaningful bounds on the number of kindergartens required.

To determine an upper bound, we apply a greedy 0/1 knapsack approach. The kindergartens are sorted by capacity in descending order, and starting from the largest, they are filled sequentially until all children are assigned. In this formulation, kindergarten capacity corresponds to the knapsack size, while the number of children per residential block represents item weights. Since the number of children is stored as a floating-point value, direct application of the knapsack model was not possible. However, analysis of the dataset showed that child counts were precise up to two decimal places, allowing us to scale all values by a factor of 100 to convert them into integers.

The lower bound is derived by systematically generating all possible selections of kindergartens for a given number and evaluating feasibility. The process begins by computing the minimum number of kindergartens required to accommodate all children. This value serves as the initial bound. All possible selections of kindergartens of this size are then generated, and buildings that have access to only one kindergarten are assigned accordingly. A selection is considered valid if this assignment does not violate capacity constraints and all buildings have access to at least one kindergarten. If no feasible combination is found, the process repeats with an incremented number of kindergartens. The complexity of verifying a single combination is linear, while the number of combinations follows the standard binomial coefficient:

$$\binom{|K|}{p} = \frac{|K|!}{p!(|K| - p)!} \tag{14}$$

where $|K|$ is the total number of kindergartens, and $p$ is the selected subset size.

### 5.2 Kindergarten Selection and Constraint Programming Model

Once the search space is bounded, the final selection of kindergartens is determined using a constraint programming model in MiniZinc. The process iteratively searches for the minimum number of kindergartens required by solving the feasibility problem for a given number. If a solution exists, this number becomes the new upper bound, and the search continues within the updated range.

Two distinct MiniZinc models were developed. The first model performs a pure feasibility check, determining whether a given number of kindergartens can accommodate all children while maintaining accessibility constraints. The second model incorporates a clustering-based approach to improve efficiency. Kindergartens are grouped using a feature space based on their distances to buildings, and the largest kindergarten from each cluster is selected as a representative candidate. Buildings that have access to only one kindergarten are immediately assigned. The remaining buildings are sorted by their proximity to the selected kindergartens, forming preference-based clusters where each building expresses a ranked preference for nearby facilities. If an entire cluster of buildings can be assigned to their top choice without violating capacity constraints, the assignment is finalized. The MiniZinc model then takes the pre-assigned buildings and the selected kindergartens as input to optimize the final allocation.

This structured approach ensures that the search for an optimal solution remains computationally feasible while preserving real-world constraints on accessibility and kindergarten capacity.

## 6 Experimental Setup and Results

The proposed approach is evaluated on two datasets: the **neighborhood-level dataset**, which consists of 348 buildings and 59 kindergarten plots, and the **street-level dataset**, a smaller toy example containing 17 buildings and 5 kindergarten plots. The experiments compare different preprocessing methods, as well as the performance of two constraint programming models in solving the kindergarten allocation problem.

### 6.1 Preprocessing Evaluation

To improve computational efficiency and guide the constraint programming solver, a clustering-based preprocessing step was applied using K-Means. Two preprocessing strategies were tested: one where the data was log-transformed and scaled, and another where standard scaling was applied without logarithmic transformation. Additionally, experiments were conducted using both a weighted and an unweighted distance matrix to assess the impact of weighting on clustering quality.

The results indicate that **scaled clustering** of the kindergartens performed slightly better then the **log-transformed clustering** for the neighborhood-level dataset, leading to more balanced groups and improving the efficiency of the subsequent constraint programming models. The usage of weighted matrix is not improving the performance. Figure 3 illustrates the clustering results for **scaled clustering** preprocessing method and Fig. 4 illustrates the clustering results for **log** preprocessing.

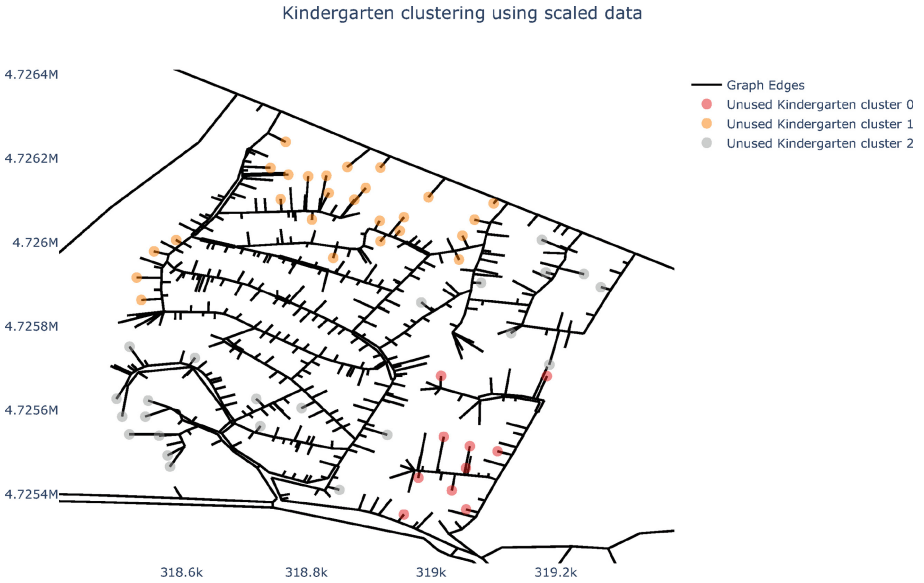


**Fig. 3.** Visualization of kindergarten clustering using scaled data for the neighborhood-level dataset.

### 6.2 Comparison of Constraint Programming Models

Two CP models were compared. The first model directly attempts to solve the allocation problem by determining the optimal selection of kindergartens and assigning buildings to them. The second model incorporates additional preprocessing steps, including clustering, pre-assignment of buildings that have access to only one kindergarten, and prioritization of larger kindergartens within clusters.

The results show that the first CP model struggles with computational feasibility for the neighborhood-level dataset, failing to compute a solution for scenarios requiring three kindergartens. In contrast, the second CP model, which incorporates clustering and additional heuristics, performs significantly better, producing solutions efficiently for the same dataset. Both models, however, successfully solve the allocation problem for the smaller street-level dataset, indicating that the first model is only viable for small-scale instances.

### 6.3 Solver Performance

The CP models were executed using two different solvers: **Chuffed** and **Gecode**. Experimental results indicate that Chuffed outperformed Gecode in terms of computational speed, consistently solving the allocation problem slightly faster across different scenarios. However, both solvers produced the same optimal solutions, demonstrating that the choice of solver impacts runtime performance but not solution quality.

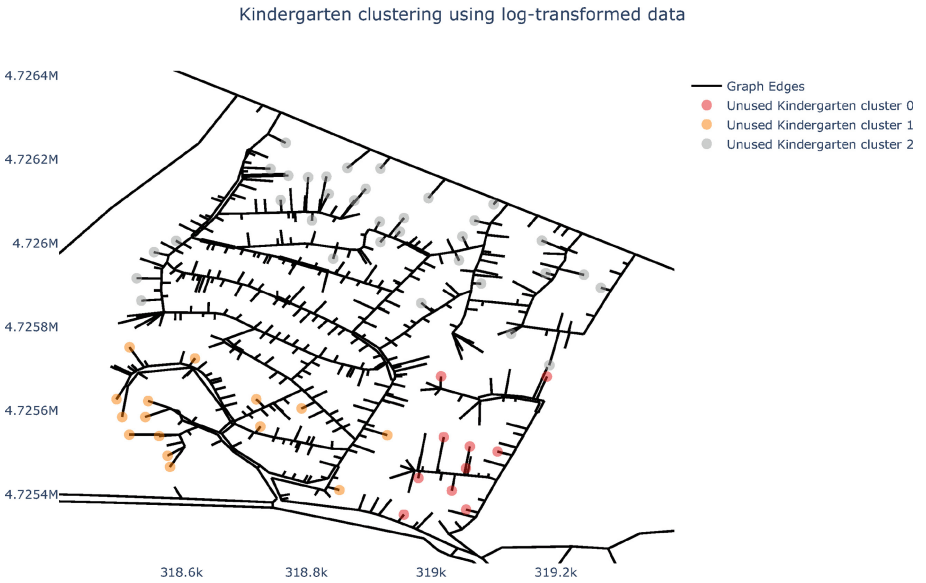


**Fig. 4.** Visualization of kindergarten clustering using log-transformed data for the neighborhood-level dataset.

### 6.4 Evaluation Metrics

To fairly assess the quality of the obtained solutions, the results were evaluated based on how well the kindergartens were distributed across the city. Instead of relying solely on standard optimization metrics such as total walking distance, the clustering balance of the selected kindergartens was also examined to ensure an equitable spatial distribution. The goal was to allocate kindergartens in a way that minimized local congestion while maintaining accessibility for all buildings.

Figures 5 and 6 illustrate the spatial assignment of buildings to kindergartens after clustering and pre-assignment, demonstrating the final distributions used in the second MiniZinc model.

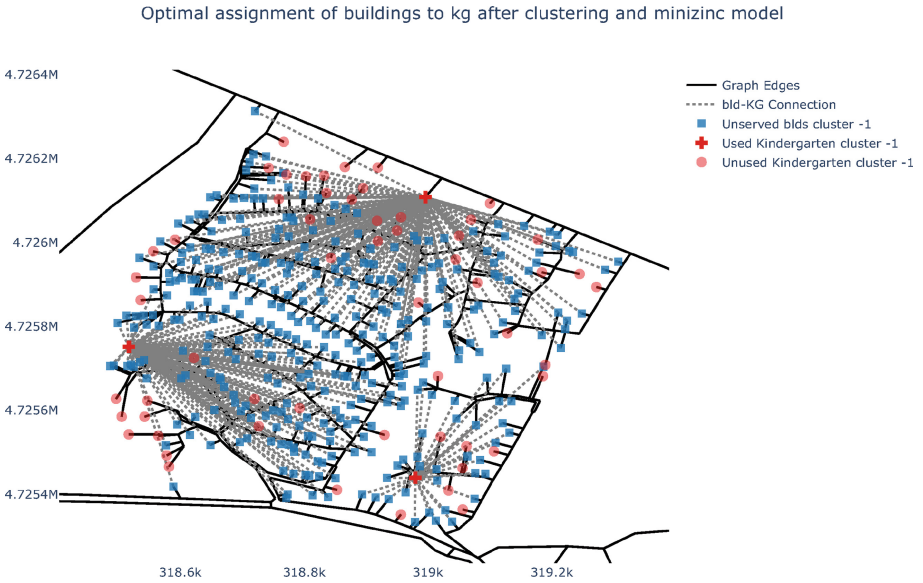


**Fig. 5.** Final assignment of buildings to kindergartens for the neighborhood-level dataset after clustering and pre-assignment.

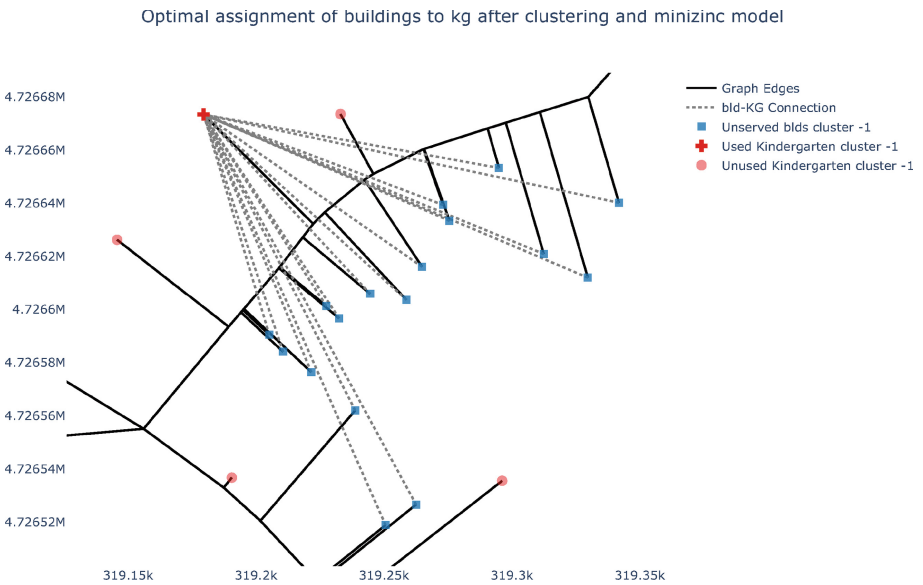


**Fig. 6.** Final assignment of buildings to kindergartens for the street-level dataset after clustering and pre-assignment.

### 6.5 Future Considerations

While the clustering-based preprocessing approach significantly improves computational performance, further refinements could be explored. Future work could integrate demand forecasting models to dynamically adjust the number of required kindergartens based on projected population changes. Additionally, alternative clustering techniques, such as hierarchical clustering or density-based methods, could be tested to determine whether they provide more balanced kindergarten distributions.

The experimental results demonstrate that constraint programming is a viable approach for solving the kindergarten allocation problem, particularly when combined with preprocessing techniques that reduce search complexity. The findings suggest that additional heuristics and domain-specific constraints could further enhance solution quality and computational efficiency.

## 7 Conclusion

This paper presents a CP-based approach to the kindergarten allocation problem, ensuring optimal placement of kindergartens while maintaining accessibility and capacity constraints. By formulating the problem in MiniZinc and leveraging CP solvers, we achieve efficient solutions within a computationally feasible framework.

The proposed method introduces a bounding strategy that effectively narrows the search space by determining realistic upper and lower limits on the number of kindergartens required. Additionally, the integration of a clustering-based preprocessing step improves solution efficiency, reducing computational complexity. The experimental evaluation on real-world data from Sofia confirms the effectiveness of our approach, demonstrating its ability to optimize both the number of kindergartens and the total walking distances for children.

Future work will explore extending the model to incorporate additional constraints, such as land-use regulations and budgetary limitations. Furthermore, adapting the method to dynamic scenarios, where population densities and demand fluctuate over time, could enhance its applicability to long-term urban planning. The combination of CP techniques with machine learning-based predictions for demand forecasting is another promising direction for further research.

**Acknowledgments.** This research is part of the GATE project funded by the Horizon 2020 WIDESPREAD-2018-2020 TEAMING Phase 2 Programme under grant agreement no. 857155 and the programme "Research, Innovation and Digitalization for Smart Transformation" 2021–2027 (PRIDST) under grant agreement no. BG16RFPR002-1.014-0010-C01, and the CU project, funded by Digital Europe Programme DIGITAL-2023-CLOUD-AI-04 under grant agreement no. 101168005.

## References


1. Geitle, A.H., Johnsen, Ø.K., Ruud, H., Fagerholt, K., Julsvoll, C.A.: Kindergarten allocation in Norway: an integer programming approach. J. Oper. Res. Soc. **72**(7), 1664–1673 (2020)
2. Veski, A., Biró, P., Põder, K., Lauri, T.: Efficiency and fair access in Kindergarten allocation policy design. J. Mech. Inst. Design **2**, 57–104 (2017)
3. Batsaris, M., Kavroudakis, D., Hatjiparaskevas, E., Panagiotis, A.: Spatial decision support system for efficient school location-allocation. Eur. J. Geogr. **12**, 31–44 (2021)
4. Gong, D., Gen, M., Yamazaki, G., Xu, W.: Hybrid evolutionary method for capacitated location-allocation problem. Comput. Industr. Eng. **33**(3–4), 577–580 (1997). ISSN 0360-8352
5. Vasanthi, T., Arulmozhi, G.: Optimal allocation problem using genetic algorithm. Int. J. Oper. Res. (IJOR) **5**(2) (2009)
6. Wang, K.-J., Lin, Y.-S.: Resource allocation by genetic algorithm with fuzzy inference. Expert Syst. Appl. **33**(4), 1025–1035 (2007). ISSN 0957-4174
7. Sharma, J., Singhal, R.S.: Genetic algorithm and hybrid genetic algorithm for space allocation problems- a review. Int. J. Comput. Appl. (0975 – 8887) **95**(4) (2014)
8. Wang, L., Liu, T.: Optimization algorithm of preschool education resource allocation based on data envelopment analysis model. Sci. Program. **2022**, 3858664 (2022)
9. Alp, O., Erkut, E., Drezner, Z.: An efficient genetic algorithm for the p-median problem. Ann. Oper. Res. **122**, 21–42 (2003)
10. Avella, P., Sassano, A., Vasil'ev, I.: Computational study of large-scale p-median problems. Math. Program. **109**, 89–114 (2007)
11. Correa, E.S., Steiner, M., Freitas, A.A., et al.: A genetic algorithm for solving a capacitated p-median problem. Numer. Algorithms **35**, 373–388 (2004)
12. Yaghini, M., Karimi, M., Rahbar, M.: A hybrid metaheuristic approach for the capacitated p-median problem. Appl. Soft Comput. **13**(9), 3922–3930 (2013)
13. Weil, G., Heus, K., Francois, P., Poujade, M.: Constraint programming for nurse scheduling. IEEE Eng. Med. Biol. Mag. **14**(4), 417–422 (1995)
14. Hanset, A., Meskens, N., Duvivier, D.: Using constraint programming to schedule an operating theatre. In: 2010 IEEE Workshop on Health Care Management (WHCM), Venice, Italy, pp. 1–6 (2010)
15. Chan, W.T., Hu, H.: Constraint programming approach to precast production scheduling. J. Constr. Eng. Manage. **6**, 513–521 (2002)
16. Quiroga, O., Zeballos, L., Henning, G., A constraint programming approach to tool allocation and resource scheduling in FMS. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain, pp. 3715–3720 (2005)
17. Heinz, S., Beck, J.C.: Solving resource allocation/scheduling problems with constraint integer programming (2011)
18. Galleguillos, C., Kiziltan, Z., Soto, R.: A constraint programming-based job dispatcher for modern HPC systems and applications (2020)
19. Cappart, Q., Moisan, T., Rousseau, L.-M., Prémont-Schwarz, I., Cire, A.A.: Combining reinforcement learning and constraint programming for combinatorial optimization. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 35, no. 5, pp. 3677–3687 (2021)

20. Caprara, A., et al.: Integrating constraint logic programming and operations research techniques for the crew rostering problem. Softw. Pract. Exp. **28**(1), 49–76 (1998)
21. Michel, L., Hentenryck, P.V. Constraint-based local search. In: Martí, R., Pardalos, P., Resende, M. (eds.) Handbook of Heuristics. Springer, Cham (2018)
22. Ozbay, K., Xiao, W.: Probabilistic programming models for response vehicle dispatching and resource allocation in traffic incident management (2004)
23. Liu, S.-S., Utami, P., Budiwirawan, A., Arifin, M., Perdana, F.S.: Optimization model of maintenance scheduling problem for heritage buildings with constraint programming. Buildings **2023**, 13 (1867)
24. Hooker, J., Ottosson, G., Thorsteinsson, E.S., Kim, H.-J.: On integrating constraint propagation and linear programming for combinatorial optimization. In: AAAI/IAAI (1999)
25. Kalchev, Y.: Cities and their urbanized areas in the Republic of Bulgaria 2010 – 2020, Sofia (2022). https://www.nsi.bg/sites/default/files/files/publications/Urban2010-2020.pdf. Accessed 03 July 2025
26. https://www.sofia.bg/en/web/tourism-in-sofia/more-information-on-sofia. Accessed 03 July 2025
27. Karamitov, K., Petrova-Antonova, D., Hristov, E., Borukova, M.: Supply-demand analysis of urban amenities based on walking accessibility, 3973–3979 (2023)
28. Boyukliyski, S., Petrova-Antonova, D., Somanath, S.: Evaluation of social facilities coverage: a case study of Sofia city. In: IFAC Workshop on Control for Smart Cities CSC 2022. IFAC-PapersOnLine, vol. 55, no. 11, pp. 48–53 (2022)