UDC 004

V. Petrov, Ye. Batrak, N. Tsopa

COMPARATIVE ANALYSIS OF HYBRID TASK ALLOCATION STRATEGIES IN ROBOTIC DELIVERY SYSTEMS

Abstract: This article presents a comparative analysis of task allocation strategies in robot delivery systems. The study shows that with increasing load, the efficiency of complex algorithms significantly outperforms basic methods that focus only on system parameters at a given moment in time. The hybrid methods with forecasting and the Hungarian algorithm approach proved to be the most effective across all metrics. They improve delivery time, reduce failures, and ensure better scalability.

Keywords: robotic systems, task allocation, greedy algorithm, Hungarian method, route optimization, simulation, ETA prediction, intelligent logistics, hybrid strategies, system, architecture, modeling, automatic control.

Introduction

In today's world, there is a steady trend towards the automation of business processes, among which the logistics and delivery industry occupies a special place. One of the most promising areas is the introduction of autonomous robotic systems for delivery. This approach opens up significant opportunities to optimize costs, increase the efficiency of logistics operations, and ensure round-the-clock availability of delivery services.

Main challenges and tasks of the system

The efficient operation of an autonomous robot fleet requires not only the development of reliable hardware, but also the creation of an intelligent control system. This system should ensure the optimal distribution of tasks among the available robots, taking into account a complex set of interrelated factors, including the distance to the destination, the current battery level of the battery pack, the current workload of each robot, and the predicted time of the task. In addition, an extremely important aspect is the development of efficient and safe routes for robots [6] to travel that would take into account dynamic environmental changes, such as traffic congestion or temporary obstacles.

Purpose, approach and practical value of the study

In the context of this study, we investigate the process of designing and implementing a system capable of distributing tasks between delivery robots and generating optimal routes for their movement, taking into account various constraints and performance criteria.The purpose of this work is to develop an integrated approach to autonomous delivery management that ensures the efficient distribution of tasks between delivery robots and the selection of optimal routes for their execution, taking into account a set of relevant constraints. To achieve this goal, an in-depth analysis of the subject area and existing solutions was carried out, a multilevel architecture of the software system was designed, the key components of the developed system were implemented, and their effectiveness was evaluated in simulation. The relevance of this study is due to the need to improve the efficiency of automated logistics systems [5] in the face of a constant increase in demand for fast, convenient and contactless delivery. The results of this study can be practically applied in real logistics platforms, serve as a basis for the development of robotic system simulators, and become the basis for further research in this area.

System architecture and implementation

This section describes the general structure of the developed autonomous delivery system, its components, and functional interconnections. The system is implemented as a modular multi-level architecture that includes a client interface, server logic, router, simulation environment, and database (Fig.1). This structure allows for flexibility, scalability, and ease of testing various task distribution strategies.

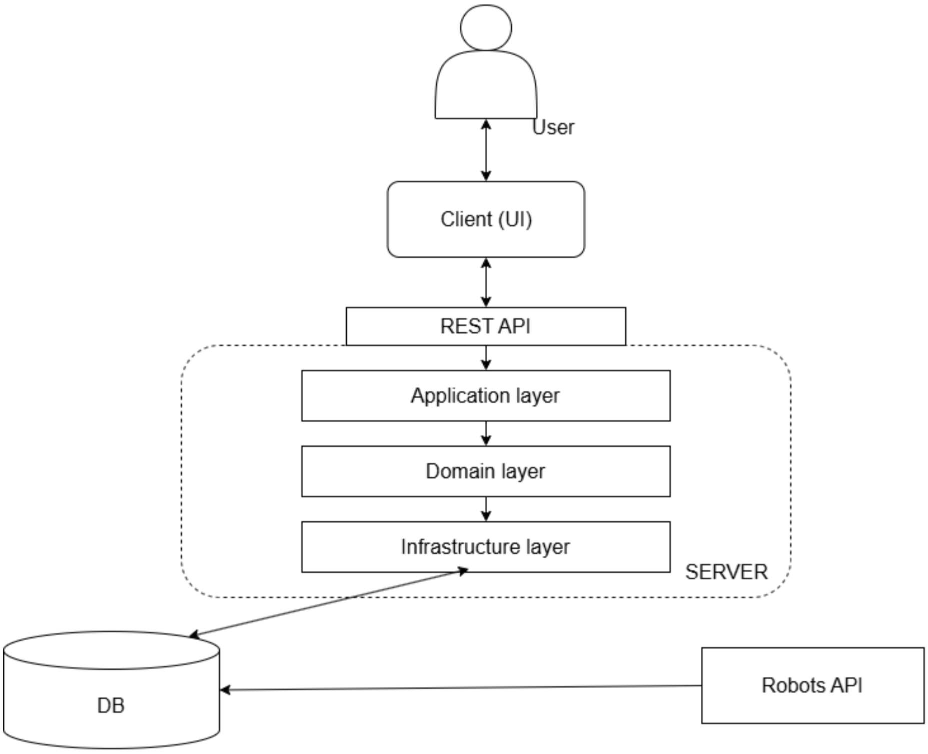

Figure 1. The general structure of the autonomous delivery system

The client interface provides users with tools to create delivery requests, track order status, and interact with the system in real time. The server logic handles core business processes such as task allocation, robot management, order lifecycle coordination, and data validation. The routing module calculates optimal delivery paths using geographic data and efficient pathfinding algorithms. The simulation environment enables controlled testing of system behavior under various demand conditions by emulating robot movement, delivery timings, and virtual time progression. The database stores all operational data, including orders, robot statuses, locations, and historical performance metrics.

Implemented distribution algorithms

In this paper, several approaches to task distribution among autonomous robots have been implemented and studied. Each of them has its own characteristics, advantages, and disadvantages, which manifest themselves depending on the current system load, the number of active robots, the topology of the environment, and the complexity of orders. Let's take a closer look at each of the algorithms. The distribution of tasks between robots is a key aspect of the system, as its efficiency affects the overall delivery performance. Several strategies were implemented in the study, ranging from the simplest to the hybrid. This subsection describes the basic principles of each implemented strategy and their logic, while the following sections will provide a comparative assessment of their performance criteria.

Random assignment algorithm.

The simplest approach implemented in the system is a random selection of the contractor. Its main purpose is to provide basic functionality to test the system's performance and compare it with more advanced methods. The algorithm works according to the following principle: from the set of robots that currently have the “free” status, the one that meets the minimum requirements of the order (load capacity, charge level, absence of an active task) is randomly selected. If there are several such robots, the selection is made randomly.

Advantages:

– speed of realization and execution;
– simplicity of logic - does not require complex calculations.

Disadvantages:

– complete lack of optimization by time, route, or ETA;
– can lead to a significant imbalance in the workload between robots.

This approach is used only as a baseline for comparing the performance of more complex strategies.

Greedy algorithm

This algorithm is based on the idea of assigning the closest robot to the order pickup point that meets the technical requirements. For each new order, the algorithm calculates the distance to all available robots and assigns the one with the shortest distance to the delivery start point. The available charge, volume and weight capacity are also taken into account.

Advantages:

– simplicity of implementation and logic;
– relatively low consumption of computing resources;
– provides a noticeable improvement over random distribution.

 ISSN 1560-8956

Disadvantages:

– ignores busy robots that may be released before the free robot arrives at the point;
– it does not take into account ETA prediction or the state of the robot queue.

The greedy algorithm is effective in environments with a moderate number of robots and orders, but can deteriorate as the workload increases.

Greedy algorithm with forecasting

An improvement of the previous method involves taking into account busy robots. The algorithm calculates the expected completion time of the current task for each robot and models the ETA to the new task. Thus, a robot that is currently busy, but will soon complete the delivery and quickly reach a new point can be selected instead of a remote but free one. Additional parameters that are taken into account: expected time to be free, current location and speed, battery status at the time of the new task.

Advantages:

– better load balancing;
– less robot downtime;
– more optimal delivery ETA.

Disadvantages:

– greater complexity of the algorithm;
– requires more computing resources.

This approach has shown better results in environments with a dynamic order flow and a large number of robots.

Hybrid strategy with the Hungarian algorithm

This approach combines a greedy strategy with periodic global reallocation of tasks using the Hungarian algorithm. The basic idea is as follows: the greedy algorithm is used by default to make quick decisions, but in case of a queue of unfulfilled orders or inefficient assignment, the system switches to global optimization mode. The Hungarian Method allows you to find the optimal assignment of tasks to performers so as to minimize the total ETA or other metric (for example, energy consumption) [1]. The assignment matrix is formed from current orders in the queue and available/busy robots.

Advantages:

– high efficiency under heavy load;
– possibility of global balancing and optimization.

Disadvantages:

– high computational complexity - O(n3);
– it is necessary to define clear criteria for switching to this mode.

The system implements a mechanism for automatically launching the Hungarian algorithm only when the number of orders in the queue exceeds a certain threshold or when an excessive load on a limited number of robots is detected. This avoids excessive CPU load while maintaining the benefits of global optimization.

Simulation and analysis of system performance

Preparing to system simulation

An urban area of approximately 2 km^2 was selected for the study. This area is home to about 7500 people, which determines the potential demand for delivery services. Within this area, a sevis coverage polygon is defined - the area in which robotic delivery will be carried out. From a practical point of view, dividing an urban area into conditional neighborhoods or zones instead of modeling the entire large area as a single entity has a number of advantages.

The division allows for a more efficient distribution of orders and resources between individual zones, which reduces the risk of overloading the system during peak periods in one particular location.

If the system is expanded, you can gradually launch robotic delivery in stages - first in one quarter, then in the next, with minimal costs for rebuilding the logic.

Within a block, it is easier to model shorter routes with fewer intersections and turns, which is critical for autonomous navigation.

Modeling a limited space allows you to track the work of each robot in detail, analyze the system load, and the time it takes to serve an order.

This approach allows us to obtain more accurate results, adapt the system to the needs of a particular site, and ensure the gradual scaling of the project.

To evaluate the system's performance under different conditions, three delivery demand scenarios (Tab. 1) were created: low, medium, and high. Each of them simulates a certain level of user activity within the selected area and allows for a comparative analysis of the effectiveness of task distribution algorithms. Each scenario sets the projected number of orders per virtual day, as well as the number of robots, which corresponds to the behavior of consumers according to statistical data, 1-3% of the district's residents usually make delivery orders per day [4].

Table 1.

Testing scenarious

Demand	Number of orders	Number of robots
Low	75	8
Medium	150	15
High	225	23

This division allows you to analyze the effectiveness of delivery management algorithms under different system loads - from relatively low to intense.

ISSN 1560-8956

In addition to the total number of orders, it is important to take into account their distribution during the day. The model uses a typical hourly schedule of orders. Usually, the lowest activity is observed at night (around 00:00-06:00). In the morning, demand gradually increases, reaching its first peak around noon. After a slight decline in the afternoon, a second peak period occurs in the evening, when the number of orders increases again. After the evening peak, the activity decreases by night. This distribution corresponds to the typical daily behavior of users of delivery services and allows us to simulate a realistic scenario of orders.

For the simulation, we generate robots with different technical characteristics. The carrying capacity varies from 8 to 10 kilograms. Accordingly, the simulation takes into account the permissible weight of the parcel when choosing a robot. All robots have similar dimensions of 40x40x45-50 cm. The maximum speed varies from 4 to 6.5 km/h. The distribution of characteristics allows us to simulate a heterogeneous fleet of robots, similar to those used in real commercial delivery systems. The difference between the models allows us to investigate and evaluate the effectiveness of different task assignment algorithms that take into account the characteristics of each device: from simple random selection to complex assignment using the Hungarian algorithm and load prediction.

For each scenario, scripts were written to generate initial data about the robot. The simulation system is deployed on the server side with support for virtual time. This means that events are modeled in an accelerated mode, not tied to real time - one second of simulation can correspond to a minute of real time, which allows you to quickly simulate the system's operation. All important events and parameters are logged in the system for further analysis. Logging includes event timestamps, robot and order identifiers, distance traveled, delays, and other metrics.

The prepared infrastructure allows for multi-scenario testing for different load levels. That is, the simulation can be run separately for each of the above scenarios in order to compare the results and assess the scalability and reliability of the system. This approach allows us to identify how an increase in the number of orders affects the average delivery time, robot utilization, and other key metrics, as well as to check the level of demand up to which the system can efficiently service orders.

Conducting order processing simulation

After preparing the environment, we ran a series of order processing simulations for each of the three scenarios: low, medium, and high demand. Each simulation run modeled one virtual day of delivery service operation at a given load level. A fixed schedule of orders by hour was used. This was done to ensure a correct comparison. All algorithms received the same order conditions and the same orders as input, the only difference was the intensity of the order flow and the number of robots.

For each scenario, all four algorithms were run sequentially. That is, 12 full-fledged simulations were performed, which allowed us to collect data on the effectiveness of each approach in different conditions.

After all the runs were completed, the results were summarized for comparison. The collected statistics are presented in the final summary tab. 2.

Table 2.

Simulation results

Algorithm	Demand (orders./ day)	Number of robots	Completed orders	Avg. ETA (min.)	Way (m.)	Peak queue	Percentage of potential failures
Random	75	8	75	39	171709	2	0
Random	150	15	150	45	358542	5	4
Random	225	23	225	42	500751	13	20
Greedy	75	8	75	29	140964	1	0
Greedy	150	15	150	33	291182	3	6.5
Greedy	225	23	225	38	418550	6	8
Greedy with forecasting	75	8	75	29	138057	1	0
Greedy with forecasting	150	15	150	31	276649	3	6
Greedy with forecasting	225	23	225	35	412785	7	7
Hybrid	75	8	75	29	138057	1	0
Hybrid	150	15	150	30	274952	2	2
Hybrid	225	23	225	27	395683	2	3

The average ETA and total path were calculated using SQL scripts, as the data was stored in the database. The percentage of potential failures was calculated based on the data on orders that were in the queue for more than 1 hour during processing, so the user could refuse because the service was inefficient. The data on this parameter was recorded in logs and checked in the database for status changes.

Peak queues were also determined by logs at times of system load. In all cases, the orders were completed, since there were no specially modeled failures and the system was allowed to process all incoming requests, ignoring the fact that the order was processed for a long time during peak hours.

Analysis of simulation results

The analysis of the results table shows a clear trend that the performance gap between simple and advanced algorithms increases with increasing workload.

Under the lowest load conditions, all algorithms were able to fulfill the maximum number of orders. There would have been no failures in these cases even if the robot had been chosen randomly. The average delivery time for the random selection method was relatively high at 39 minutes, while all other algorithms managed to complete the job in 29 minutes. This indicates that even the greedy algorithm performs satisfactorily under light workloads, and the prediction added in the second iteration did not play a special role in terms of ETA. The total distance traveled was approximately the same for all algorithms (140000 meters) except for random selection. The formation of large queues was also not noticed.

With an increase in the number of orders to 150 per day and the number of robots to 15, the difference in the efficiency of the algorithms became more noticeable. Random selection is significantly inferior in all respects to other algorithms. At this stage, we have already started to record failures with greedy algorithms, which is about 6.5%, and by adding predictions, we were able to reduce it to 6%. The complex algorithm (a combination of the greedy and Hungarian algorithms) significantly improved this indicator, with failures decreasing to 2%, but still remaining. The average ETA increased for all approaches under high load, about 30 minutes in all cases. However, there is a result in terms of the distance traveled, adding predictions was able to reduce it by 5% compared to the usual greedy one. In the case of the complex approach, almost nothing changed, as the Hungarian algorithm worked only once during the peak hour (correcting the failures, but not reducing the distance much). Queues also increased, as expected.

At maximum load, the difference between the algorithms became the most significant. With random selection, the average ETA reached 42 minutes, and with all the improvements, the result was 27 minutes. The total distance with the greedy algorithm was approximately 418 thousand meters and was reduced to 396 thousand meters, and the queues remained at the level of two orders, which is 6 times better than random selection and 3 times better than greedy selection. Using an integrated approach, it was not possible to avoid failures, but their number decreased to 3%, while using a conventional greedy algorithm, the figure was 8%.

At high load levels, the advantages of smart dispatching algorithms are most fully manifested. The integrated approach significantly outperforms the others in all respects.

Evaluation of system efficiency

Based on the simulation, it is possible to assess the overall efficiency of the control system of various algorithms. The evaluation criteria were performance, service speed, resource consumption, level of request accumulation, and service reliability. According to all these criteria, the complex algorithm demonstrated the highest efficiency, especially under heavy loads. Instead, the random algorithm served as a basic example of the lowest efficiency, showing the worst results for each of the indicators. The two variants of the greedy algorithm took an intermediate position.

The average ETA clearly reflects the quality of service for users. The complex method minimized waiting and delivery times. The prediction algorithm performed slightly better than the greedy one, as it better organized the queue and routes thanks to the forecast.

The distance traveled served as a measure of resource consumption. When using the complex algorithm at high load, it was possible to minimize this indicator, even when maintaining a low number of failures and high demand for delivery. Other algorithms did not cope with the task as effectively. The peak queue and bounce rate reflect the algorithm's ability to cope with the influx of requests. The complex algorithm provided the lowest values of these indicators: the queue was almost never formed, and the failures were minimal even with a significant overload (only a few percent of unsatisfied requests). The predictive algorithm also maintained a small maximum queue and a low failure rate, confirming the effectiveness of taking into account future demand. In contrast, the random approach resulted in a substantial queue build-up (several times larger than the other algorithms at peak times) and a high failure rate, meaning that the randomly controlled system could not handle all requests.

The results will highlight the advantages and disadvantages of each algorithm. Random selection is the easiest to implement, requiring no additional computation or information, but its main drawback is its extremely uneven and inefficient performance. It does not scale well under load. With a large number of orders, there are long queues, high delays, and a significant percentage of unfulfilled requests. This method may be acceptable only as a starting point or for a system with a very low load.

The greedy algorithm uses local optimization when assigning each individual order, which gives it an advantage over the random approach. It is able to fulfill orders and reduce waiting time due to the desire to immediately select the "best" current option. However, the disadvantage of the greedy strategy is its short-sightedness. Without taking into account future requests and the global picture, it does not work optimally under high load. This leads to the accumulation of deferred requests in the long run, situations of suboptimal use of robots, which increases the average waiting time and the percentage of failures compared to more "intelligent" algorithms.

The greedy algorithm with forecasting eliminates many of the disadvantages of the basic greedy approach. Using the forecast, the system predicts the location of robots and distributes them according to the information received. The simulation results showed that not in all cases, but in most cases, a simple prediction can significantly improve efficiency.

An integrated approach is the best option. Its advantages were manifested in the best performance on various criteria. This algorithm makes the best use of available resources and responds quickly to changes in the system. The main disadvantage of this algorithm is its high computational complexity.

This study does not take into account the current situation on the road or various exceptional circumstances. The environment is somewhat ideal and does not allow for a full test of the system. However, it gives an idea of the general picture, the trend of improvements made and the evolution of approaches.

Summary

The simulation modeled the operation of an autonomous robotic delivery system in a realistic urban environment with different levels of load. The created infrastructure made it possible to evaluate the system's behavior in low, medium, and high demand conditions, as well as to test the effectiveness of different approaches to task distribution among robots. The simulation results showed that the system is able to operate stably in the face of dynamic order intake, ensuring that tasks are completed within a reasonable time frame, even with an increase in the number of requests. The most important performance criteria were the average delivery time, the number of completed orders, resource utilization, queue accumulation, and the percentage of failures. The analysis by these metrics allowed us to evaluate not only the performance but also the reliability of the system. All simulations were conducted under the same conditions using virtual time, which ensured the objectivity of the results.

The use of the simulation approach made it possible to identify bottlenecks in the decision-making logic and to formulate reasonable conclusions about the feasibility of using more complex optimization methods that take into account the dynamic user behavior and limited resources in a real-time environment. According to the results of the study, the highest efficiency was demonstrated by the complex algorithm, which provided the best performance in all key metrics, including minimum ETA, lowest failure rate, and efficient use of resources, making it the best choice for implementation in real urban environments.

In the future, it is advisable to consider the integration of additional factors, such as weather forecasts, traffic on city streets, and various exceptional situations that may occur in a real-world environment. For example, it is promising to introduce an adaptive mechanism for recalculating routes in the event of road accidents, sudden traffic restrictions, or natural disasters. This can be implemented through public APIs of city services and integration with emergency notification platforms such as Starship Technologies [2] and Amazon Scout [3]. With this flexibility, the system will be able to quickly divert robots to detour routes or temporarily suspend order fulfillment in dangerous areas, while maintaining a high level of safety and customer satisfaction.

REFERENCES

1. *Kuhn H.W.* The Hungarian method for the assignment problem [Electronic resource] / H.W. Kuhn // Naval Research Logistics Quarterly 2. – 1995. – P. 83-97. Available from: https://web.eecs.umich.edu/~pettie/matching/Kuhn-hungarian-assignment.pdf (application date: 05.20.2025)
2. Starship Technologies [Electronic resource]. – Available from: https://www.starship.xyz (application date: 05.20.2025)
3. Amazon Scout Delivery System [Electronic resource]. – Available from: https://www.aboutamazon.com/news/transportation/meet-scout (application date: 05.20.2025)
4. Consumer habits of Amazon Europe shoppers and their influence on market selection [Electronic resource]. – Available from: https://disqover.agency/spozhivchi-zvichki-pokupciv-na-amazon-europe-ta-yak-voni-vplivayut-na-vibir-rinku. (application date: 05.20.2025)
5. *Olkhova M. V.* Smart transport and Logistics for Cities: Lecture Notes [Electronic resource] / M. V. Olkhova // Kharkiv National University of Municipal Economy named after O.M. Beketov – 2021 – 62 p. – Available from: https://eprints.kname.edu.ua/59000/1/5_2020_90%D0%9B_Lecture%20notes_Smart%20transport%20and%20Logistics%20for%20Cities_2020-21_UA.pdf. (application date: 05.20.2025)
6. *Hart P.E.* A Formal Basis for the Heuristic Determination of Minimum Cost Paths / P.E. Hart, N.J. Nilsson, B. Raphael // IEEE Transactions on Systems Science and Cybernetics. – 1968. – Vol. 4, No. 2. – P. 100–107. https://doi.org/10.1109/TSSC.1968.300136