The Traveling Repairman Problem App for Mobile Phones: a Case on Perishable Product Delivery

M.E. Bruni, M. Forte, A. Scarlato, P. Beraldi

Abstract Delivering perishable food as soon as possible has been always a challenge for producers, amplified in recent years, by a more and more competitive global market. The problem can be tackled as a routing problem with consideration of the arrival time at the customer's location, taking into account the perishability of the products planned to be delivered. Since in real-world applications customers' requests dynamically arrive during the execution of the transportation process, building vehicle routes in an on-going fashion is a challenge to be addressed. This paper describes a mobile solution that heavily relies on the use of mobile phones and integrates a well known heuristic method for the problem at hand. A case concerning the delivery of perishable food to a set of restaurants will serve as a base for illustrating the potential benefits of such a system.

Key words: Traveling Repairman Problem, Mobile app, Perishable product

---

M.E. Bruni
Department of Mechanical, Energy and Management Engineering, University of Calabria, Rende, Cosenza, Italy, e-mail: mariaelena.bruni@unical.it

M. Forte
Department of Mathematics and Computer Science, University of Calabria, Arcavacata di Rende (CS) 87036, marcuzzuhp@gmail.com, University of Calabria, Rende, Cosenza, Italy, e-mail: marcuzzuhp@gmail.com

A. Scarlato
Department of Mathematics and Computer Science, University of Calabria, Arcavacata di Rende (CS) 87036, scarlatoalessandro19@gmail.com, University of Calabria, Rende, Cosenza, Italy, e-mail: scarlatoalessandro19@gmail.com

P. Beraldi
Department of Mechanical, Energy and Management Engineering, University of Calabria, Rende, Cosenza, Italy, e-mail: patrizia.beraldi@unical.it

1 Introduction

Delivering perishable food as soon as possible has been always a concern for producers and carriers. For many products that have short life span the element of time is the biggest challenge faced by these companies. In order to gain a competitive advantage over the competitors, the food delivery companies should deliver food within short lead times, accurately and with acceptable quality. Notwithstanding this has been always a challenge for producers, it has been amplified, over recent years, by a more and more competitive global market. The term perishable is applied for the products that start deteriorating as soon as they are produced. In general there are two kinds of deterioration. The first one makes the product outdated after a specified time (like blood). The second type, makes the products quality unpleasant as time passes, as in the case of flowers, vegetables and foods which are labelled as highly perishable because they deteriorate significantly fast and, as a by-product, the costumers are sensitive about their freshness. For instance, a grocery shop which sells vegetables definitely wants them to be as fresh as if they have been brought from the farm. It is noteworthy that the revenues of food suppliers are dependent on the condition and freshness of the products. As a matter of fact, delivering low-quality products may impose a penalty on the supplier, decreasing the collected revenue. The above-mentioned factors highlight the importance for the suppliers to organize an efficient and effective delivery in order to maximize the freshness of the products delivered to the customers and to maximize the collected profit at the same time. This entails solving a vehicle routing problem with consideration of the arrival time at the customer's location, taking into account the perishability of the products planned to be delivered. This calls for models which aim at maximizing the revenue (profit minus perishability costs). These two objectives are in conflict with each other and an appropriate trade-off is needed.

The Traveling Repairman Problem with profits (TRPP) is an extension of the Traveling Repairman Problem (TRP), where a time-dependent profit is associated with each customer and the objective is to maximize the total collected revenue, which is a decreasing function of the arrival time at the nodes. Hence, it is an appropriate modeling framework for the problem at hand.

In real-world applications, customer requests dynamically arrive during the execution of the transportation process. Building vehicle routes in an ongoing fashion, in such a way that customer requests arriving dynamically are efficiently and effectively served, is a challenge addressed in on-line routing approaches. Although dynamic routing problems and quantitative methods for on-line routing have been discussed in the scientific literature since a seminal paper of Psaraftis [19, 20], the technology required for implementing on-line routing methods is more recent. In particular, we refer to the mobile technology, which emerges as a new business process characterised by mobility, reachability, localisation and personalisation. The user of a mobile device can access networks, products and services while on the move. This is important in context like ours, where the driver must be timely informed about the next stop to approach and the dispatching centre must be informed about the driver's location and the status of the delivery. The availability of a mobile communication system alone is, however, not sufficient. It is important that the communication system is properly and friendly interfaced with the firm system and data bases. If well designed, the mobile system will help the driver in routing decisions, taking also into account real-time traffic information (up-to-date information on weather and road conditions and detours) improving the convenience,the safety and the efficiency of travel.

This paper describes a mobile business solution that heavily relies on the use of mobile phones. An advantage of such a system is its low cost and its ease of use. Furthermore, the integration of algorithms for the solution of routing problems is relatively straightforward. A case concerning the delivery of perishable food to a set of restaurants will serve as a base for illustrating the potential benefits of such a system.

The next Section gives a short overview of the related literature, Section 3 describes the problem and discusses in some detail the above mentioned mobile system and the implementation issues. Section 4 presents the application on perishable products delivery and screen snapshots are presented. Finally, conclusions are given in Section 5.

2 Related work

Customer-centric routing problems, where the customer's satisfaction is taken into account mostly through the arrival time at the customer's location, are broadly referred to as the minimum latency problems or traveling repairman problems (TRPs). The aim is to find a tour, starting from a depot node, which minimizes the sum of the elapsed times (or latencies) to reach a given set of nodes. The problem has been extensively studied by a large number of researchers who proposed several exact and non-exact approaches. Lucena [10] and Bianco et al. [3] proposed early exact enumerative algorithms, in which lower bounds are derived using a Lagrangian relaxation. Fischetti et al. [7] proposed an enumerative algorithm that makes use of lower bounds obtained from a linear integer programming formulation. Different mixed integer programming formulations with various families of valid inequalities have been proposed in the last years [4, 11, 13, 18]. Salehipour et al. [16] first proposed a simple composite algorithm based on a Greedy Randomized Adaptive Search Procedure (GRASP) [8, 9], improved with a variable neighborhood search procedure. In [12], Mladenović et al. presented a general variable neighborhood search metaheuristic enhanced with a move evaluation procedure facilitating the update of the incumbent solution. Silva et al. [17] presented a composite multi-start metaheuristic approach consisting of a GRASP for the construction procedure, and a randomized variable neighborhood descent algorithm for the improvement phase. In [1], Avci et al. presented a new mixed-integer linear model capable of solving small size instances and a simple and effective metaheuristic algorithm which combines a GRASP for initial solution construction and Iterated Local Search (ILS) with an adaptive perturbation mechanism for solution improvement. In particular, the developed GRASP-ILS obtained the best known results for the benchmark instances. For this reason, in this paper, this heuristic has been used as solution method to solve the problem at hand.

Recently, some interesting variants of the TRP have been proposed. Nucamendi-Guillén et al. [15] proposed two new models for the capacitated version and an efficient iterated greedy procedure. For the variant with profits, namely for the TRPP, a stochastic programming model with chance constraints has been proposed in [2]. The aim is to find the travel plan that maximizes the random revenue that can be collected with a given reliability level.

Although many researchers have studied the TRP, the literature on the multi-vehicle TRP (referred to as k-TRP) is surprisingly limited. Recently, Nucamendi-Guillén et al. [14, 15] presented an efficient new formulation, defined on a multi-level network, for the deterministic k-TRP enhanced by an iterative greedy metaheuristic. The k-TRPP under uncertainty has been recently addressed in [5, 6], where a reactive GRASP has been proposed to solve the problem.

3 Problem description and system implementation

The perishable product delivery is usually performed by third-party carriers. The carriers receive the orders characterized by a position, a profit (price paid by the customer) and a service time. We assume that the capacity of the vehicle is enough to carry all the orders. The objective of the driver is to find a tour that maximizes the total profit, minimizing at the same time the so called latency, that is the waiting time of the customers, which is a proxy of the product freshness. The two conflicting criteria are then considered into the same objective function in the spirit of the TRPP. Moreover, the driver has a time limit on the route duration, that should not exceed a given timelapse, normally four or five hours (a delivery tour starts in the morning and ends at midday). The driver collects the data and feeds them into a software system, which reoptimizes heuristically and in real time the route. The route is calculated and defined upstream by the server. On the basis of the objective function trade-off and for the presence of the route duration constraint, some of the customers may not be visited. The driver will then apologise for the inconvenience and assure subsequent delivery. If the customer is selected to be visited, a corresponding delivery order is triggered to the driver performing the delivery. Drivers who already started their delivery tours may then receive additional orders. After the service, the driver notifies the system about the completed delivery and eventually add the new orders. Then, the route is re-optimized by the system, possibly servicing new customers.

The app requires a series of data to work, some of them are present in the server and therefore supplied when necessary, others are supplied through the driver (also called here operator). Potential customers are represented by nodes, characterized by an Id (used only for technical purposes), a defined geographical position through latitude and longitude, a profit and an expected service time.

Figure 1 shows the system's general architecture. During the design process the system requirements were examined and three main components were identified:

- a practical interface which allows the operator to communicate with the server
- a system able to receive requests and process them in real-time
- an external service that provides information on the routes.

An Android device was chosen as the mobile technology to be used by the operator. This means that http requests can be sent to the central system and it also provides a series of services and integrations with other systems that facilitate the operator during the execution of his tasks. For the Android devices we used the latest version: 28.0.3 of the "build tool", as the "minimum sdk" we used version 15 and as "target sdk" version 26. This allows the application to be run on devices with an Android operating system greater or equal to the "Ice Cream Sandwich" version (4.0.4). There are two actors involved in the process the operator/driver and the the service manager/the product supplier. The service manager is the one who uses the system to plan the work of the operator. His main interaction with the system is to add various points to an existing route. Figure 2 shows the use case diagram.

The core of the system is clearly the central server, which performs various tasks:

- it processes all (http) requests submitted to the system
- it executes the heuristic algorithm
- it queries the external services to find the data necessary for the execution of the algorithm.

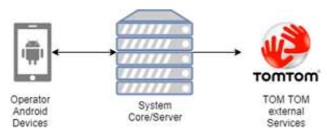

Fig. 1: System Architecture

A fundamental requirement for the performance of the previous tasks is the ability to access data regarding the possible route from a starting point to a destination, the distance, and above all, the time needed for the tour. This information can be obtained from different providers such as "Google Routes" or "TOM TOM Routing API". TOM TOM was chosen because it is the only one that offers a free version of the service. The only version currently usable and available of the TOM TOM API is 1.0. All the communication take place via Web Rest in https with http basic authentication method. Server side Java 1.8 was chosen as it was one of the most mature and advanced technologies. Considering the tasks performed by the server, the Spring Boot framework was the best solution to adopt as it is excellent for the management of back-end systems with API. The framework handles all the technical aspects, allowing developers to focus on the application core instead of the technicalities. The version of Spring Boot used is 2.0.4.

The server is composed of three layers: the communication layer, the logic layer and the model layer. The functional division of the components into different levels makes each independent of each other. There is a physical dependence of the Java classes belonging to the higher levels compared to the lower level classes. The divisions of the components due to functionality and class dependence make the system highly modular and consequently improves maintenance, the addition of new components or their replacements. Some Java classes do not belong to any of the previous levels, as they do not have a functional collocation but rather only deal with technical functions of system configuration.

The communication layer, as the name suggests, deals with the external communication of the system. It has the primary task of receiving incoming requests and, once they are verified, it forwards these requests to the logical level. The classes in this level use Spring Boot annotations, turning a simple class into a restless service controller. Through the injection function, provided by the framework, each controller has a reference to the instance of the logical component to which forward requests.

The logic layer, also called "business layer" or "core layer", is the main component of the system. All the logic of the processes is contained in this section which in turn is divided into different packages: services, algorithm and external. The services component is the entry point for requests sent from the upper level (layer communication), each request is processed in full in this section. During the processing of requests, the services component often uses the other components to deal with specific tasks, such as the generation of a new solution through the algorithm package and/or the acquisition of new data from external services through the external classes package.

The model layer is a simple container of the "objects" used by the system, this component does not contain any logical role or process.

As far a the heuristic called by the server is concerned, as already mentioned, we have implemented from scratch in Java the GRASP-ILS heuristic proposed in [1]. The algorithm has been tailored to address the route duration

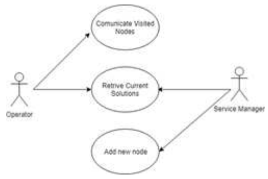

Fig. 2: Use case diagram

constraint by adding into the objective function evaluation a term penalizing the infeasibility of the route duration constraint. We have observed that the algorithm is very fast and the solution time is almost neglectable, in agreement with the real time information that it should provide. Each time the heuristic is called the information is updated: travel times are updated with real-time traffic data of road segments and incoming requests are eventually added to the set of nodes. Using the two kinds of information previously mentioned, the GRASP-ILS algorithm is proposed to either calculate an initial route or to re-optimize the route and the server updates the visual guidance information for the driver in real-time.

The Android application has a single main screen which contains a Fragment which in turn contains a GoogleMap. From this interface the operator can perform different tasks: he can consult the solution (a series of points visited or to visit) and communicate a node already visited. The communication takes place via the http protocol and the APIs exposed by the server are RESTful type. The application contains a specific package that deals with communication, where all the remote requests are asynchronous and are managed by a special class that extends AsyncTask. During the execution of requests, the user is shown a ProgressDialog that indicates the status of the request. The interface shows the points of interest with markers and the route to follow via polylines. The operator can choose whether to display only the nodes belonging to the solution or also the nodes that have been discarded. This option can be activated through a Toggle Button. The interface also includes another button that shows, through a window, the values of the current solution: the profit collected and the total time in seconds ("HH: mm: ss" format).

4 Results

The system was tested considering 78 restaurants in Milan and a deposit (close to the Cathedral of Milan) with profit 0 and service time 0. As a server was used a Windows 10, CPU i7 quad-core, 16 GB RAM ddr4. For each node the service time was generated between 500 and 900 seconds. The profit of

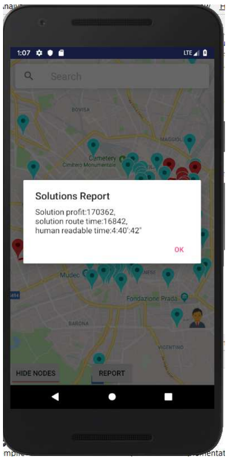

Fig. 3: Initial solution report

a node was calculated according to the restaurant rating (the final values of the profits lie between 10500 and 13500).

Clicking on the "Report" button the window with the details of the solution will be opened. In Figure 3 the report of the initial solution is reported. Analyzing the proposed solution in more detail, the chosen nodes are 15 and the order is as follows: 0, 35, 74, 41, 29, 60, 26, 4, 70, 63, 33, 9, 20, 15, 54, 46. The operator starts the route. After servicing the first seven nodes, a new request arrives with a profit of 16000, slightly higher than the average (12000). Then the driver confirms the visit of the already visited nodes and

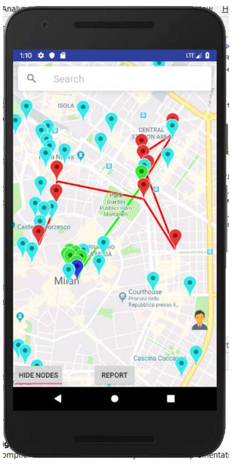

Fig. 4: Updated solution

communicates to the system the presence of this new customer. The system updates the route as shown in Figure 4. Here, the restaurants to be visited are highlighted in red. The blue marker indicates the deposit, while the markers in light blue indicate discarded restaurants. The red line shows the route to follow. The markers and the lines that connect the nodes already visited are represented in green. In the new solution, the profit increases from 170362 to 184485.

5 Conclusions

This paper introduces a mobile communication system that enables route re-optimization. The paper exemplifies that today's communication and information technology allows to quickly build mobile communication systems based on easy-to-handle system components. Such systems can be used to implement mobile business processes for supporting logistic processes by means of optimisation and quantitative methods. The development, investigation and implementation of efficient and fast heuristic methods is then a complementary research area, which can be a promising future research area.

References

1. Avci M,, Avci MG. A GRASP with iterated local search for the traveling repairman problem with profits. Computers & Industrial Engineering. **113**, 323–332, (2017).
2. Beraldi P, Bruni ME, Laganà D, Musmanno R. The risk-averse traveling repairman problem with profits. Soft Computing. **23(9)** 2979-2993 (2018).
3. Bianco L, Mingozzi A, Ricciardelli S. The traveling salesman problem with cumulative costs. Networks. **23(2)**, 81–91 (1993).
4. Bigras LP, Gamache M, Savard G. The time-dependent traveling salesman problem and single machine scheduling problems with sequence dependent setup times. Discrete Optimization. **5(4)**, 685–699 (2008).
5. Bruni, M.E., Beraldi, P., Khodaparasti, S. A fast heuristic for routing in post-disaster humanitarian relief logistics. Transportation Research Procedia. **30**, 304-313 (2018)
6. Bruni M.E, Beraldi P, Khodaparasti S. A heuristic Approach for the k-Traveling Repairman Problem with Profits under Uncertainty. Electronic Notes in Discrete Mathematics. **69**, 221-228 (2018).
7. Fischetti M, Laporte G, Martello S. The delivery man problem and cumulative matroids. Operations Research. **41(6)**, 1055–1064 (1993).
8. Guerriero F, Bruni, M.E, Greco. A hybrid greedy randomized adaptive search heuristic to solve the dial-a-ride problem. Asia-Pacific Journal of Operational Research. **30(1)**, 1250046-1250067 (2013).
9. Bruni, M.E, Guerriero F, P. Beraldi. Designing robust routes for demand-responsive transport systems. Transportation Research Part E: Logistics and Transportation Review. **70(1)**, 1-16 (2014).

10. Lucena A. Time-dependent traveling salesman problem-the deliveryman case. Networks. **20(6)**, 753–763 (1990)
11. Ezzine IO, Semet F, Chabchoub H. New formulations for the traveling repairman problem. In Proceedings of the 8th International conference of modeling and simulation 2010.
12. Mladenović N, Urošević D, Hanafi S. Variable neighborhood search for the travelling deliveryman problem. 4OR. **11(1)**, 57–73 (2013).
13. Méndez-Díaz I, Zabala P, Lucena A. A new formulation for the traveling deliveryman problem. Discrete applied mathematics. **156(17)**, 3223–3237 (2008).
14. Nucamendi-Guillén S, Martínez-Salazar I, Angel-Bello F, Moreno-Vega JM. A mixed integer formulation and an efficient metaheuristic procedure for the k-Travelling Repairmen Problem. Journal of the Operational Research Society. **67(8)** 1121-34 (2016).
15. Nucamendi-Guillén S, Angel-Bello F, Martínez-Salazar I, Cordero-Franco AE. The cumulative capacitated vehicle routing problem: New formulations and iterated greedy algorithms. Expert Systems with Applications. **113**, 315-327 (2018).
16. Salehipour A, Sörensen K, Goos P, Bräysy O. Efficient GRASP+ VND and GRASP + VNS metaheuristics for the traveling repairman problem. 4OR: A Quarterly Journal of Operations Research. **9(2)**, 189–209 (2011).
17. Silva MM, Subramanian A, Vidal T, Ochi LS. A simple and effective metaheuristic for the minimum latency problem. European Journal of Operational Research. **221(3)**, 513-20 (2012).
18. Van Eijl CA. A polyhedral approach to the delivery man problem. Department of Math. and Computing Science, University of Technology 1995.
19. H.N. Psaraftis, Dynamic vehicle routing problems, B.Golden, A. Assad (Editors), Vehicle routing: methods andstudies, Elsevier Science Publishers BV, 1988, pp. 223–248.
20. H.N. Psaraftis, Dynamic vehicle routing: Status and prospects. Ann Oper Res **61**, 143–164 (1995).