

GrocerQuick Route Optimisation – Problem Solving Report

Jayce Atkins – 2401799

University of Wales Trinity St David

Introduction

GrocerQuick is a growing online grocery delivery service that has started to run into problems with how deliveries are handled day to day. Drivers are often sent on routes that don't really make sense, with stops spread far apart and no clear order. Late deliveries are becoming more common, and this is starting to affect both customer satisfaction and staff workload. Fuel costs are also creeping up, mostly because vans are spending too much time on the road.

At the moment, there isn't a proper system in place to plan delivery routes based on distance, time or actual driver availability. Orders are assigned in a fairly loose way, without checking if the locations are close together or if the route fits into a driver's shift. Because of this, drivers can end up travelling back and forth across the same areas, wasting time and fuel. Once a route is sent out, there's also no real way to adjust it if traffic builds up or something unexpected happens.

Bigger companies like Ocado and Amazon Fresh deal with this using advanced logistics systems, live traffic data and automated routing. GrocerQuick doesn't have access to that level of technology. Budget limits, small teams and lack of technical support mean that any solution has to be realistic and easy to manage. There's no point suggesting something that looks impressive but can't actually be used properly.

This report looks at what can realistically be done to improve delivery routes without changing everything at once. The focus is on using simple planning methods and well-known techniques to reduce inefficiencies. By improving how orders are grouped and checked before drivers leave, GrocerQuick could cut fuel use and improve delivery times without needing expensive new software.

Breaking Down the Problem

What's Going Wrong

Delivery work often sounds simpler than it really is. In practice, each route has to deal with traffic conditions, delivery time windows, driver shift lengths and vehicle limits. If one delivery runs late, it can easily affect everything else on that route. A small delay at the start of a shift can turn into a much bigger problem by the end of the day.

One of the main issues at GrocerQuick is that deliveries aren't grouped by location. A driver might make their first drop in the north of the city, then be sent south for the next one, only to come back north again later. This routing wastes fuel and makes it very hard to stick to delivery times. It also puts extra pressure on drivers, who end up rushing to make up for poor planning..

There's also no proper check to see if a route is even doable before it's sent out. Delivery windows are often tight, and no allowance is made for traffic or slow handovers. When things go wrong, the response is usually reactive rather than planned. Customers complain, drivers get stressed, and the same mistakes happen again the next day.

Drivers try to deal with this in different ways. Some change the order of their stops based on experience, while others use exactly what they've been given even if it doesn't make sense. This creates inconsistency. Two routes with the same number of deliveries can have very different outcomes just because of how they were planned.

Prioritising Needs Using MoSCoW

To stop the solution becoming too complicated, the MoSCoW prioritisation method was used [1]. This method helps break requirements into four groups: Must have, Should have, Could have and Won't have. It's useful when time and resources are limited.

MoSCoW Priorities

- **Must:** Deliveries must not be more than 30 minutes late. Fuel usage needs to be reduced.
- **Should:** Orders should be grouped by delivery area. Traffic conditions should be considered where possible.
- **Could:** Simple tools could be added later to predict delays or highlight risky routes.
- **Won't:** AI automation will not be used at this stage because of cost and complexity.

Using MoSCoW helps keep the focus on what actually matters right now. Improving order grouping and reducing unnecessary travel would have an immediate impact without needing major changes. More advanced features might be useful later,, but they aren't essential at this point.

This also avoids trying to fix everything at once. When too many changes are made together, it becomes hard to tell what's actually working. Starting with the basics makes the system easier to manage and easier to improve over time.

Key Constraints

GrocerQuick operates under several real world real time constraints that limit what kind of solution is realistic. These aren't just technical problems, but operational ones too.

Some of the main constraints include:

- Orders are assigned manually without checking proximity
- Delivery times are not checked for feasibility before dispatch
- Driver shift lengths are not matched to route complexity
- Traffic information is limited or ignored during planning
- There is no clear process for reviewing route performance

These issues don't exist on their own. They interact with each other. Poor grouping leads to longer routes, longer routes lead to late deliveries, and late deliveries increase pressure on drivers. Looking at the problem as a system helps explain why fixing just one issue on its own doesn't solve everything [4].

There is a lack of usable data. Without tracking of delivery times or route distances, planning decisions are often based on guesswork rather than evidence. This makes it harder to learn from past mistakes.

The human side matters as well. Some drivers know the area very well and can work around bad routing, others don't. A system that relies too much on individual judgement will always be inconsistent. Adding more structure helps reduce that reliance and makes performance more predictable.

How Routes Could Be Planned Better

To reduce the problems GrocerQuick are facing, a more organised way of planning routes is needed. It doesn't need to be complicated or expensive. What matters is stopping the obvious mistakes, such as sending drivers back and forth across the city or giving them routes that can't actually be finished on time.

The idea is to introduce a planning stage before routes are sent out. Orders would still come in the same way as they do now, but instead of being assigned straight away, they would be grouped and checked first. This alone could prevent a lot of the delays that are happening now.

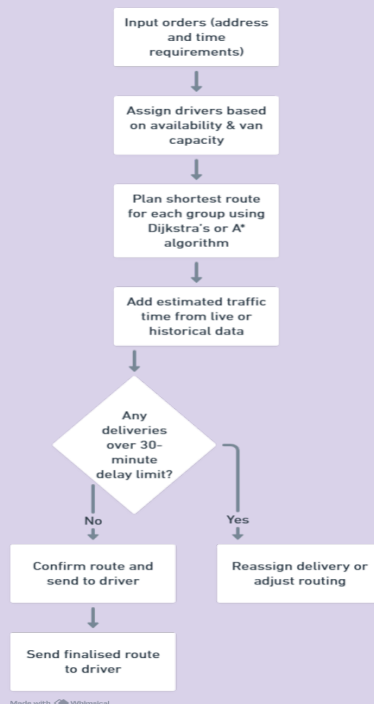
Planning Flow Overview

The route planning flow starts when orders are confirmed. At this stage, basic details like address and delivery time are recorded. Instead of treating each order as separated, the

next step is grouping them into delivery zones. This helps to reduce unnecessary travel and makes routes easier to manage.

A simple visual flowchart can show this process clearly, from order input through to route checks and final dispatch. Having a clear flow also helps staff understand what's happening and where changes can be made if something goes wrong.

Figure 1 - route planning flow chart



How the Process Works

Once orders are logged, they are grouped by location using K-Means clustering [2]. This method groups deliveries that are close together, which reduces long drives between stops. It's not perfect, but it's better than assigning orders at random.

After clustering, drivers are assigned to each group based on how long their shift is and how many deliveries are involved. This avoids situations where a short shift is overloaded or where a driver is sent far outside their usual area.

The next step is planning the actual route. Dijkstra's algorithm is used to calculate the shortest path between all delivery points in a group [3]. This gives a sensible order for stops without relying on guesswork. If traffic

information is available, it can be added at this point, though it doesn't have to be live to still be useful.

Before the route is finalised, a timing check is done. If any delivery is predicted to arrive more than 30 minutes late, the route is flagged. At that point, changes can be made before the driver even leaves. This might mean moving one delivery to another route or adjusting the order slightly.

Once everything checks out, the route is sent to the driver. The goal is that drivers start their shift with a plan that actually makes sense, rather than having to fix problems on the road.

Why This Would Help

This approach fixes several issues at one time. Grouping deliveries reduces backtracking, which cuts fuel usage and saves time [2]. Planning the shortest path helps keep routes efficient and predictable [3]. Together, these steps make it easier for drivers to finish their routes on time.

Another benefit is consistency. When routes are planned using the same logic each day, performance becomes more predictable. It's easier to spot problems and easier to improve the process over time.

It also keeps human judgement in the loop. Dispatchers can still adjust routes if they know about a local issue, like roadworks or difficult access to a building. The plan gives a solid starting point rather than locking anyone into a rigid system.

What the Plan Doesn't Cover Yet

There are limits to what this setup can do intentionally. It doesn't respond to sudden changes like traffic accidents or last-minute cancellations. Drivers may also ignore suggested routes if they think they know a faster way. That can be helpful sometimes, but it can also cause large inconsistencies.

The system also doesn't account for breaks, vehicle capacity differences or special delivery requirements. These issues become more important as delivery volumes increase. In future, more advanced methods like genetic algorithms could help handle these factors [6].

those ideas have been left out on purpose. The aim now is to fix the biggest problems first, then build something not too complex to manage.

Reflection and Justification

The problem solving approach used in this paper focused on breaking the delivery process into smaller parts. Instead of treating late deliveries as a single issue, the analysis looked at how routing, timing and driver assignment interact with each other. This systems thinking view helped explain why small problems keep turning into bigger ones [4].

The MoSCow method was useful in deciding what to focus on first [1]. Late deliveries and fuel waste stood out as the biggest issues. Improving route structure directly addresses both of these. K-Means clustering and Dijkstra's algorithm were chosen because they are well known and relatively easy to implement using free tools [2][3].

More advanced options were considered, including AI based systems. These were ruled out because they require more data, training and technical support than GrocerQuick currently has. A simpler solution that actually gets used is more valuable than a complex one that doesn't.

This plan isn't perfect. It won't stop every late delivery or handle every unexpected problem. But it does create a clearer structure and removes many of the avoidable issues that happen now. That alone would make a noticeable difference.

Over time, the system could be improved. Driver feedback could be collected, traffic patterns analysed and routes refined. But those improvements only make sense once the basics are in place.

Final Thoughts

The biggest improvements don't always come from the most advanced tools. In this case, grouping deliveries properly and checking routes before dispatch could solve a large part of the problem.

This approach fits GrocerQuick's current situation. It's realistic, manageable and flexible enough to grow later. It may not be perfect, but it's a good step towards a more reliable delivery service.

References

- [1] S. McKenna, "MoSCoW prioritization: A guide to managing requirements," *PM101*, Medium, 2020. [Online]. Available: <https://medium.com/pm101/moscowprioritizationaguidetomanagingrequirements413c5adef1cb>
- [2] Y. Liu, J. Li, and X. Zhang, "Solving multiple distribution center locational location problem using KMeans algorithm and center of gravity method," *IEEE Access*, 2020. [Online]. Available: <https://www.researchgate.net/publication/346376771>
- [3] A. Sharma and R. Gupta, "Application of Dijkstra algorithm in finding the shortest path," *International Journal of Computer Applications*, 2023. [Online]. Available: <https://www.researchgate.net/publication/367796376>
- [4] J. Houghton, "Systems thinking and complexity," *The Open University*, 2019. [Online]. Available: <https://oro.open.ac.uk/100414/1/100414VOR.pdf>
- [5] Routific, "What is the vehicle routing problem (VRP)?," *Routific Blog*, 2023. [Online]. Available: <https://www.routific.com/blog/whatisthevehicleroutingproblem>
- [6] Y. Wang, Z. Li, and H. Chen, "An improved genetic algorithm for vehicle routing problem with pickup and delivery and time windows," *IEEE Access*, 2021. [Online]. Available: <https://www.researchgate.net/publication/350762070>