Meal delivery, customer experience versus driver's welfare

Author: Justin Vonk
Bachelor Thesis, Yingjie Fan

Abstract

The aim of this thesis is to be a part of the research in online meal delivery. It is focussed on delivery by delivery platforms like Uber Eats and Grub hub. It simulates days of a delivery platform by using real life data. Different days are simulated by changing the speed of the drivers, the interval between optimization times, the length of the shift of the drivers and the number of drivers throughout the entire day. We look at the changes from out the point of view of the drivers, the customers, and the delivery platform. The values that we are interested in are the click to door times, total distance driven by the drivers, payment of the drivers and the amount of unserved customers. We concluded that changing the speed isn't that worth it for the stakeholders, as we expected. Changing the interval didn't do much either. We were still able to shorten the length of the shifts to 80% and were still able to deliver almost all the orders. Changing the number of drivers throughout the day seemed to be better to do than to change the lengths of the shifts.

Table of contents

1. Introduction

Online meal delivery is a big part of our economy and is expected to grow even bigger. In 2022, it was around 154.84 billion euros and is expected to be 399.55 billion euros in 2030. With a compound annual growth rate (CAGR) of about 12.58 during the period 2022-2030 according to (The Brainy Insights, 2022)

In the figure below (figure 1) it shows the growth in North America.
Here the CAGR is expected to be 18.3 in the period from 2022 till 2030 according to (Grand View Research, 2022). As shown, the biggest part of the delivered meals are from a platform and not from the restaurant itself. This thesis is also focussed on the platform to the customer side of the market.

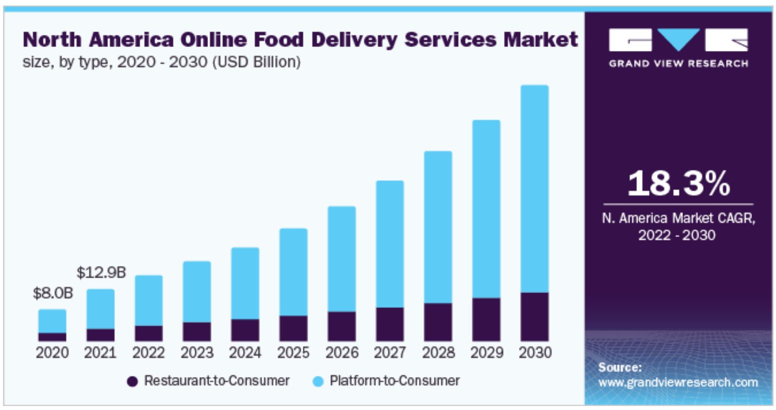

Figure 1: size of Online Food Delivery Services Market in North America

Our research extends existing research, The Meal Delivery Routing Problem (Reyes et al., 2018). Which is a paper that introduces the meal delivery routing problem. Part of the datasets are used and we extended their performance summary program. The performance summary, checks the quality of the solutions and calculates the results. The code that they used to get their results wasn't included. So that is created for this research.

The datasets that are used are a simulation of a day for a delivery platform. With all the orders, drivers and restaurants of that day. To get the results, different parameters were changed to create different days and thereby different solutions. To create the solutions, an optimization cycle (figure 2) is used. This cycle is run a lot of times throughout the entire day at every optimization time. Once the day is finished and all the cycles are completed. The solution is sent to the performance summary.

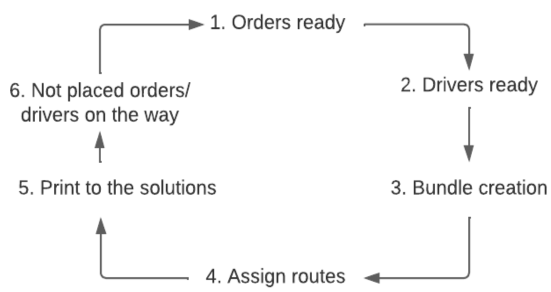

Figure 2: optimization cycle

The big difference between the new solutions and the solutions of the existing research, is that the new solutions use a central point ‘a depot’ while the existing solutions allow the drivers to ‘almost’ roam freely. The new one created a depot where all the couriers start and get their assignments from. So when the couriers start their shift they all start at the depot and after an assignment they all have to return to the depot. This location is the average of all the restaurants that are used in the dataset. So for every run there is a different depot location.

We are researching what the influence is of changing the speed of the drivers, the optimization time (interval), the length of the working shifts and the number of couriers (capacity). We look at these from the view of three stakeholders: the customers, the couriers, and the delivery platform.

So the research question is:

What is the influence of changing the speed, interval, length of the shifts and the capacity of the drivers on the drivers, customers, and delivery platform?

2. Literature review

2.1 Paper summary

The existing paper (Reyes et al., 2018) gives an introduction of the Meal delivery routing Problem, which studies dynamic delivery operations. It introduces a model of meal delivery operations that formalizes a few main structural features: multiple pick-up points (restaurants), dynamic order arrivals, delivery capacity in the form of courier shifts and the possibility to pick up multiple orders. To help with their study, they used a deterministic dynamic framework. The model uses real life data to give a solution of how the orders of different restaurants should be bundled and delivered by the couriers that are available. Some assumptions are that there is no limit of how many orders may be combined into one bundle with different drop-off locations and that the travel time of the couriers is constant.

2.2 Background research

The research field of delivery routing problems research isn't that big yet if you compare it to different research fields, because it is relatively new. But there is still some interesting research done with different views and solutions to the problem. For instance, (Steever et al., 2019) takes a look at the option of split delivery services. Where a customer is able to order at two different restaurants, but only has to place one order. Then two drivers go to different restaurants and bring the meals to the customer. It does increase the freshness of the food, but is a bit more expensive for the restaurants. Non-split delivery is even able to be as fast compared to split delivery. So it is only to make it easier for the customer. (Zhu et al., 2020) also focuses on customer experience by trying to predict the Order Fulfillment Cycle Time (OFCT). They use machine learning to predict the OFCT values and compare them to the real values. (Song et al., 2016) is more focused on perishable foods, but still looks at the vehicle routing problem. It creates a model using hypothetical data. (Xue et al., 2021) uses a two stage model that looks at the different periods and regions of the meal delivery problem. Dividing both the periods and regions in smaller portions. The first stage minimizes the drivers per period, the second stage focuses on the transportation capacity to minimize the time of the delivery rider's schedule.

(Ding et al., 2019) uses an algorithm to draw a scope that determines which restaurants a customer is allowed to order from. It appears that the scopes that are created by the algorithm work better than the manually created ones. (Liu et al., 2020) also uses a radius, but not around the customer. It has a central depot where all the meals are cooked and prepared and are delivered from. It also takes in mind the driver's routing behavior and the uncertain service times to get an even better framework. Using these two uncertainties improves the utilization of the drivers, but requires a lot more effort. (Liu et al., 2018) also looked at the driver's behavior and found that the drivers tend to deviate from the theoretical shortest distance tour. So they constructed a delivery tour length function based on historical data. It showed that their data-driven order assignment works better than the more classical models that are used for the vehicle routing problem. (Ulmer et al., 2020) looked at the uncertainties that the customers are unknown until they place an order and the ready times of the food. To address these challenges, they implemented an anticipatory customer assignment policy. This relies on a time buffer and postponement, so it is more ready for the dynamism and uncertainties of the meal delivery problem. This policy outperforms the intuitive benchmark. It improves the objectives of all the stakeholders that are involved.

There is also research that changes the problem itself. (Chen et al., 2021) transforms the dynamic delivery problem to a static optimization problem. A best-heuristic algorithm is used to quickly generate a partial solution. To further create the whole solution with high quality, multiple tie-breaking operators are designed. It results in a reveal of the success of adaptive mechanisms to utilize hybrid operators and the use of Machine learning techniques to assist decision-making for optimization problems. (Zou et al., 2021) uses Double Deep Q Networks bases Online to Offline to assign orders to all the couriers. The DQN-based dispatcher results in a similar completion time as the Traveling Salesman Problem, but DQN has a higher computational efficiency than the TSP based dispatcher. (Boza et al., 2022) uses three ways to solve the problem. The first one only checks one courier at a time using Q-learning algorithms. The second one solves a one courier model using Double Deep Q Networks and uses this one result to create a model containing more couriers. The third one uses DDQN to solve the multiple courier model, considering all the couriers. The second one, the single courier model, is the best option according to the research. (Joshi et al., 2022) introduces a new algorithm called FoodMatch that maps the vehicle assignment problem to that of a minimum weight perfect matching on a bipartite graph. The evaluations show that FoodMatch has a lower delivery time compared to (Reyes et al., 2018).

3. Methods

3.1 Difference existing paper

This research is based on existing research (Reyes et al., 2018). The biggest difference is that this paper uses a central point, the depot. This is where all the couriers start their shifts and thereby start their assignment from. After they finish their assignment, they have to go back to the depot to wait for their new assignment. Another big difference is the way the experiments are done and the datasets that are used for the experiments. (Reyes et al., 2018) has varying sizes of order and courier sets, varying travel times, varying structure of courier schedules and varying preparation times. While our research is more focused on varying traveling speeds, optimization times, number of couriers and lengths of courier shifts. There are a few smaller differences:

- (Reyes et al., 2018) only lets orders from the same restaurant in a bundle, new one also lets orders from different restaurants in the same bundle.
- (Reyes et al., 2018) has no limit on a bundle, new one has a limit of two orders.
- (Reyes et al., 2018) allows updating a courier that is on the way, the new one can only update a driver when it is back at the depot.

3.2 Framework explanation

To get the results, we used two different programs. The first one is for the solution creation. Which is created for this research. The other program is the performance summary. This one is based on what is given by the existing research (Reyes et al., 2018). It is extended to be able to also calculate all the values that were needed for this research. The framework can be seen in figure 1.

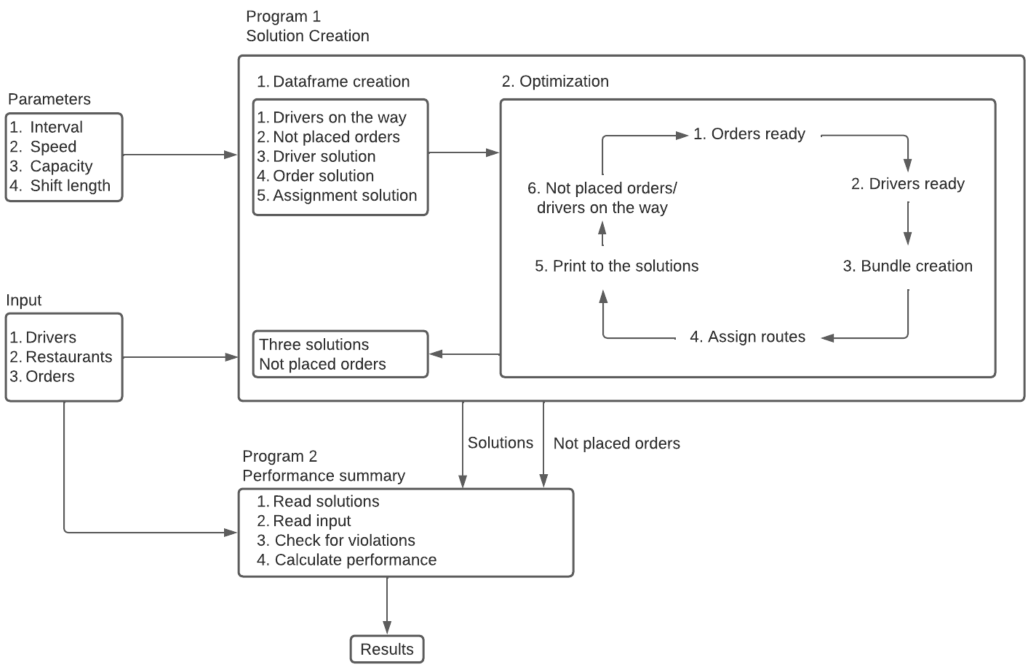

Figure 1: framework

The first program, the solution creation, takes two different inputs. The one at the top left, the parameters, tells us which parameters are used for the simulation, the parameters get explained in section 3.3. And the second one, Input, at the bottom left, contains all the information about that day. We see the drivers, where all the drivers have their beginning and end time. The restaurants, with all the restaurants of that day and their locations. Last, we have the orders: the dropoff location, the restaurant where it needs to be picked up, the placement time and the ready time.
Once all the input is given, the solution creation can start. The solution creation creates five different data frames to track everything that happens. The first one, drivers on the way, keeps track of all the drivers that aren't at the depot. So which are on their way and also how late they get back. The second one, not placed orders, keeps track of the orders that weren't included in the last optimization cycle. These orders will be given to the next optimization time to be put first in line to get put into a bundle. Then we have the three solution data frames: Driver solution, keeps track of every step of every driver. Order solution, contains all the info of every delivered order, for instance: which driver delivered it and how late it was dropped off. At last, the assignment solution contains all the bundles that were created and how late they were delivered and by which driver.
Now that everything is prepared, the optimization cycles can start. Every cycle starts with determining the orders that need to be delivered. With checking the ready times (1) of the orders that fall in the current optimization time, but also the orders that were not delivered from the last optimization cycle. Then the drivers that can be used are determined (2). When both the orders and drivers are selected, the bundle creation can start. The bundle creation (3) first determines the ideal amount of bundles:

Algorithm 1 Calculate number of bundles

```
input: orders, drivers
if zero drivers or orders then                  ▷ No drivers or orders
    number of bundles is zero
end if
if more drivers than orders then                ▷ drivers > orders
    number of bundles is number of orders
else                                            ▷ orders > drivers
    number of bundles is number of drivers
end if
```

So, if there are no orders placed or drivers available in the current optimization cycle, the number of bundles is zero. If there are more drivers than there are orders, the number of bundles is the same as the number of drivers. For instance, there are five drivers and three orders. Then the orders will be separated into three bundles with one order. These bundles will be given to the first three drivers. If there are more orders than drivers, the number of bundles is the same as the amount of drivers. All the bundles will then be filled up with all the orders, until the orders are all handed out or all the bundles have two orders in them. So full.
When the number of bundles is determined, we can now fill them up with the orders. The following algorithm is used:

Algorithm 2 Bundle Creation

```
create not placed orders
for order in orders do                                    ▷ Loop over all orders
    while bundle in bundles do                            ▷ Loop over bundles
        if bundle is empty then
            add order to route
            go to next order
        else
            go to next bundle
        end if
    end while
    if order is not placed then
        set shortest time to infinite
        set best bundle to empty
        while bundle in bundles do:                       ▷ Loop over bundles
            if bundle is not full then:
                calculate travel time
                if travel time is shorter than shortest time then
                    set shortest time to travel time
                    set best bundle to current bundle
                else
                    go to next bundle
                end if
            end if
        end while
        if placed then
            add order to best bundle
        else
            add order to not placed orders
        end if
    end if
end for
```

To summarize, the second algorithm. It loops over all the orders. Then it fills all the bundles with one order. If all bundles contain one order, it goes further to determine the best combinations of orders for the bundles. It does this by calculating the travel distance from the dropoff location of the first order, to the restaurant of the second order. Then the combination with the shortest travel time is determined. The travel times get calculated by firstly calculating the euclidean distance between two points. Then the distance gets divided by the selected speed to get the travel time. The order gets placed into that bundle and it goes further with the next order. When all the bundles are full and there are still orders left, they are added to the not placed orders list to get sent to the next optimization time.

When the bundles are created, the drivers get all the routes assigned (4). Simply by giving the first bundle to the first driver on the list, and so on. When all the routes are assigned all the information that is needed is stored in the three solution data frames (5). Then as a last step (6) there are two things left to do. The orders that weren't placed in a bundle are stored in the not placed order data frame. For all the drivers that were assigned the return time gets calculated and are stored in the drivers on the way data frame.

Now the first program is finished and the three solution data frames and the not placed orders are given to the second program, the performance summary. The second program also takes the drivers, restaurants and orders information as an input. So it also knows how the simulated day looks. Then the program checks the created solution for violations. For example, that all the steps of a driver are in the driver solution, so that a driver doesn't teleport. If there is a violation or if there are more of them. The program shows us which violation is breached and shows us where the mistake is. If there are no violations the program calculates all the values of interest and gives us the wanted results.

3.3 Parameter explanation

There are four different parameters that get changed for the experiments to get different situations. The four are: speed, interval, length of the shifts and the number of drivers.

Speed:
Tells us how fast the couriers drive when ordering their orders. It is in meters per minute. The standard speed is 320 meters per minute, which is about 19 km/h. We checked 90, 100, 125 and 150% to check what happens if the couriers go even faster than they already go to see what big of a difference it would make. But also what happens if we slow them down a bit.

Interval:
The interval tells the program what the interval is between the optimization processes. So when the program can create new bundles with the orders ready and not served at that time and give the bundle as an assignment to the couriers that are available. The different intervals used are 2, 3, 4 and 5 minutes. This is because (Reyes et al., 2018) used 2 and 5 as standards, so we wanted to check how big the steps are between those two. As standard, we used the 5-minute interval, simply because (Reyes et al., 2018) also used that as their standard.

Length of shifts:
Tells how much of the working time the couriers have, they actually work. Used to shorten the working shifts of the couriers. There are 4 different variations for this parameter as well, which are also 100, 90, 80 and 70%. For this parameter, we check the length of each courier shift and shorten it to the given percentage.

Number of drivers:
Tells us which of the dataset containing the couriers needs to be used for the run of the program. Used to use smaller datasets of couriers. There are 4 different sets, 100, 90, 80 and 70% off the total drivers. These different sets are made by taking 10% of the total of drivers while making the next dataset. So for instance for the first dataset there are 113 couriers. To make the 90% version 11 drivers were removed from the file, not completely random, but based on the shifts they have. So that eventually we still have almost the same variability of courier shifts.

3.4 Experiments

The experiments we did are based on the different parameters that are involved in the first code. Interval, speed of the drivers, the amount of drivers and the length of the working time of the couriers.

We have 4 different comparisons:

1. interval vs speed
2. speed vs capacity
3. capacity vs length of shifts
4. length of shifts vs speed

The data sets that are used are the first three given by (Reyes et al., 2018). The first three gave us the best results. We also tried to do the other seven datasets, but the data that came out of that wasn't good enough to make conclusions on. Because most of the time an assumption did get violated, so the solution was infeasible.

The four things we are interested in are:

1. wait time: average click-to-door time
2. total distance: total kilometers driven by the drivers
3. payment: average amount of salary of the drivers for that day
4. unserved-customers: amount of not delivered orders

Payments to drivers is 10 euro per delivered order, or 15 euro per hour, whichever is higher.

For each parameter, we used four different settings:

1. interval: 2, 3, 4 and 5 minutes
2. speed: 288, 320, 400 and 480 meters per minute (90, 100, 125, 150%)
3. n-drivers: 100, 90, 80 and 70%
4. working-time: 100, 90, 80 and 70%

4. Results

So in total we now have 12 tables, four comparisons with three datasets. With 16 lines, 4 times 4 parameter settings, each line gives us 6 blocks of data: the first two (table 1: interval and speed) are the parameters we compare, the other four are the performance metrics we are researching (table 1: wait_time, total_distance, payment and unserved_customers):

	interval	speed	wait-time	total-distance	payment	unserverd_customers
0	2	288	38,59	3442,067	49,18	0
1	2	320	36,83	3460,795	48,95	0
2	2	400	33,77	3489,8	49,23	0
3	2	480	32,1	3495,328	48,74	0
4	3	288	39,27	3447,579	49,37	0
5	3	320	37,45	3454,659	49,24	0
6	3	400	34,5	3479,638	48,83	0
7	3	480	32,83	3482,696	48,86	0

Table 1: Example of the used datasets, with left the parameters and right the results

Sadly there are 6 lines that are not feasible, throughout all the tables, this has something to do with the fact that the second program has some limitations on which the solution must apply to. In these 6 cases, the limitation is that the driver can only pick up orders before his off-time. And in these 6 runs, one (or more) of the orders gets picked up after the off-time of the courier it was assigned to. To get data where we can take conclusions out of, we first have the raw data as seen as above. Then we did some calculations to see the difference between the standard one and the different parameters we chose. For example below here:

Dataset 1	90% (40.12)	125% (35.46)	150% (34.16)
100% (38.65)	+3.8%	-8.3%	-11.6%

Table 2: Example of the tables with the calculated averages, speed in percentage (wait time in minutes)

In table 2 you can see the different waiting times when we change the speed in the first dataset. At the left bottom, the standard speed is shown with the average waiting time in the brackets. Then, when we get all those, we took the average of the three datasets and got these tables below:

Speed
Average for the different driving speeds

	90%	100%	125%	150%
waiting time	40.45 (+4.0%)	38.87 min	36.10 (-7.1%)	34.69 (-10.7%)
total distance	3920.6 (-0.4%)	3936.1 km	3949.4 (+0.4%)	3960.7 (+0.7%)
payment	52.02 (+0.1%)	51.96 euro	52.40 (+1.0%)	52.49 (+1.2%)

Table 3: Different speeds with results

Unserved customers

	90%	100%	125%	150%
Dataset 1	0	0	0	0
Dataset 2	0	0	0	0
Dataset 3	1	1	2	2

Table 4: different speed vs. the number of unserved customers

Changing the speed has the impact on the waiting time as expected. When we lower the speed to 90% the average waiting time increases by 4% and increasing it by 25 and 50% lowers the waiting time by 7.1% and 10.7%. These are big changes, but the change in speed is way bigger. The total distance lowers when the speed decreases and gets higher when we increase the speed. But the changes are under the 1%, so it doesn't do that much. For the payment, we see only a change of 0.1% when we lower the speed. Increasing the speed also only changes the payment by 1 and 1.2%. Which are also not that interesting to look at. Changing the speed by 50% for an increase of 1.2% isn't worth it. Another interesting thing is that the amount of unserved customers gets higher when we increase the speed in the third database. If we look further in on the input files, we can see that the last two placed orders are placed so late that the last order can't be served, and the other can only be served by one driver. And at 125 and 150% speed, that driver isn't back in time to deliver that specific order, so it can't be delivered without violations.

Interval
Average for the different intervals

	5 min	4 min	3 min	2 min
waiting time	38.87 min	38.38 (-1.3%)	37.87 (-2.6%)	37.28 (-4.1%)
total distance	3936.1 km	3930.9 (-0.2%)	3934.2 (-0.0%)	3946.6 (+0.3%)
payment	51.96 euro	51.75 (-0.4%)	52.21 (+0.5%)	52.17 (+0.4%)

Table 5: different intervals with results

Unserved customers

	5 min	4 min	3 min	2 min
Dataset 1	0	0	0	0
Dataset 2	0	0	0	0
Dataset 3	2	1	2	1

Table 6: different intervals vs. the number of unserved customers

Changing the interval only influences the customers, because of the difference in waiting time. The lower the interval, the lower the waiting time gets. The payment only changes by a maximum of 0.5% if we change it from 5 to 3 minutes. Changing it from 5 to 4 minutes even lowers the payment per driver, but only by 0.4%. So the changes are insignificant to look at. The total driven distance has even smaller changes, so isn't interesting either.

Length of shifts
Average for the different working times of the couriers

	100%	90%	80%	70%
waiting time	38.87 min	40.33 (+5.6%)*	43.62 (+12.3%)	46.48 (+19.6%)
total distance	3936.1 km	4102.6 (-0.9%)*	3844.2 (-2.3%)	3734.0 (-5.0%)
payment	51.96 euro	46.715 (-2.6%)*	50.59 (-2.8%)	49.72 (-4.5%)

*infeasible outcome in data

Table 7: different lengths of shifts with results

Unserved customers

	100%	90%	80%	70%
Dataset 1	0	0	0	0
Dataset 2	0	-	0	14
Dataset 3	1	1	1	13

Table 8: different intervals vs. the number of unserved customers

Changing the length of the shifts has a huge influence on all the values. The waiting time changes, as we expected, by a firm amount. The total distance decreases while we lower the total working times of the couriers. But the 70% is actually not that significant to use in a conclusion, because not all the orders are delivered. But there still seems to be a decreasing line if we go from 100 to 70%. The payment per driver also seems to go down. The 70% is still a bit odd to look at, because if not all orders are delivered, the drivers won't get paid for them. When we look at the amount of unserved customers, the first dataset still gets all the orders delivered, even with the 70% working shifts. But if we look at the second and third dataset, we see a big difference when we go from 80 to 70%. Those steps seem to be too big to take. Another interesting thing is that in dataset 3 with 80% shifts there is one unserved customer. That is one less than with 125% speed, and even with a 3-minute interval.

Tables length of shifts with distance per driver and total payment

	Dataset 1		Dataset 2		Dataset 3	
length	total payment	distance per driver	total payment	distance per driver	total payment	distance per driver
100%	5525,75	30,51	5632,5	37,57	9132,5	24,89
90%	5491	30,29	X	X	8705,5	24,65
80%	5507,75	29,83	5576	36,7	8480	24,29
70%	5514,5	29,42	5489,5	35,42	8137,5	23,44

Table 9: different working times with total payment and distance per driver

For the different length of the working shifts, we also checked the total payment and distance per driver. To see if something interesting can be found in there. It shows us that the drivers do have to drive more per hour. The distance per driver does decrease, but the length of their shifts decrease by more. The total payment doesn't tell us much.

Number of couriers

Average for the different number of couriers

	100%	90%	80%	70%
waiting time	38.87 min	39.60 (+1.9%)	40.89 (+5.1%)	41.80 (+7.5%)
total distance	3936.1 km	3920.9 (-0.4%)	3884.5 (-1.3%)	3849.5 (-2.3%)
payment	51.96 euro	55.38 (+6.5%)	61.22 (+17.1%)	68.78 (+32.0%)

Table 10: different number of couriers with results

Unserved customers

	100%	90%	80%	70%
Dataset 1	0	0	0	0
Dataset 2	0	0	0	2
Dataset 3	1	1	1	2

Table 11: different number of couriers vs. the number of unserved customers

The waiting time does increase as expected, but is even at 70% almost under the 40-minute goal of the platform. The total driven distance increases more when we lower the number of drivers. The fascinating part here is the payment, this has the biggest changes. Even with 70% capacity of the drivers, almost all the orders get served. In dataset 3 and 70% capacity, the amount of unserved customers is the same as when we increase the speed by 50%.

Different number of drivers with total payment

	Total payment	Total payment	Total payment
number of drivers	Dataset 1	Dataset 2	Dataset 3
100%	5525,75	5632,5	9132,5
90%	5370,75	5490	8560
80%	5195	5470	8030
70%	5130	5400	7694

Table 12: different number of couriers with the total payment

For the number of drivers, we also checked the total payment, because of the huge difference in payment per driver. The total payment does decrease when the capacity of the drivers gets lower. Yet still almost the same amount of orders get delivered. So lowering the number of drivers has only a bad influence for the customers, because the waiting time gets longer. But both the couriers and the platform get a positive influence from it, the drivers get paid more on average and the platform's total payment also gets lower.

5. Discussion

The amount of unserved customers stays at zero with the first dataset and all the different parameter settings. The second database only sees an increase of 2 and 14 in the step from 80 to 70% in the shift length and number of couriers. The third dataset also keeps the amount at 1 or 2 and also sees one outlier with the 70% working shifts. So we can say that even in extreme conditions, almost all the orders still get delivered.
Increasing the speed doesn't seem interesting to do. An increase of 25% only lowers the average waiting time by 7.1%. Which doesn't feel high enough. Lowering the speed by 10% can be appealing, the same amount of orders get delivered, and the waiting time only increases by 4% on average.
Changing the interval time doesn't change the results as much. Only the waiting times change by more than 0.5% and only with a maximum of 4%. That is only at the 2-minute interval time, which is harder to implement for both the platform and the drivers.
The different lengths of the shifts results in big changes for all three of the results (table 7), up to almost 20% increase of the waiting time and a decrease of almost 5% for both the total distance and the total payment. But 70% isn't significant data, because dataset 2 and 3 have more than 10 unserved customers. Which makes the total distance and payment go down. Same for the total payment and the distance per driver (table 11). The payment per hour does go up for the drivers. For instance, with 80% shift length, they get paid 2.8% less per shift on average, but they have to work 20% less.
When we take a look at the number of couriers available, we see the biggest changes in the total payment per driver. With an increase of 32%. The waiting times also go up, but not as much as with the different shift lengths, namely only 7.5% maximum. The total distance also only decreases by 2.3% maximum. The total payment for the platform also decreases by a lot. Especially for the third dataset (table 10), with a decrease of 15.6%.

6. Conclusion

6.1 Answering the research questions

To answer the research question, we used a table to summarize the influence of changing the four parameters compared to the three stakeholders:

	Drivers	Customers	Platform
Speed	-/+	+	-
Interval	-/+	+	-
Length of shifts	+	-	+
Number of drivers	+	-	+

Table 13: influence of parameters on the stakeholders

Increasing the speed isn’t worth it for the drivers. They have to drive 25% faster for an increase in wage of 1%. Decreasing the speed to 90% seems to be interesting for the drivers. They get paid the same and are still able to deliver the same amount of orders with the standard speed. The customers have to wait shorter, so for them an increase is good. For the platform, it is not worth it to increase the speed. You only put your drivers at risk and still deliver the same amount of orders, and the total payment does go up.
Decreasing the interval time does nothing for the drivers. It is good for the customers, because they have to wait shorter for their meals. Only 4% shorter on the lowest interval time, so it is a small benefit. For the platform, it is not worth it to decrease the interval time. It is harder to implement as a platform, but you don’t gain much from it.
Shortening the length of the shifts is good for the drivers. They get paid more per hour. The waiting times go up, so it has a negative impact for the customers. It has positive benefits for the platform, because the total payment does increase, but they have to stay away from the 70% length of the shifts.
Lowering the capacity of the drivers is good for the drivers. They get paid a lot more compared to the standard capacity. Customers have to wait longer for their meals, so it has a bad influence on them. For the platform, it is worth it to lower the capacity of the drivers throughout the day. The same amount of orders can still be delivered, but you have to pay a lot less to the drivers.

Who can benefit from this research?
This research can be used as an introduction to take in mind the welfare of the drivers. For now, we only looked at what the drivers get paid throughout the entire day. We also tried to look at the safety of the drivers to include the speed here. The central depot is something delivery platforms can take in mind while thinking of an optimal way to deliver meals.

6.2 Outcome comparison to existing papers

The existing paper (Reyes et al., 2018) has way more complex outcomes than the outcomes used in this thesis. A lot more datasets are used as well. From the 10 given datasets only the first three were usable, the distances from the restaurants to the customers and so the depot were too big. Which made it so that there would be way more NOT-FEASIBLE outcomes, as we already had with the used datasets. So the data that is created by using these datasets isn't significant enough to draw conclusions out of. Comparing the results to the (Reyes et al., 2018) is a bit odd, because of the different datasets that were used. Only the main dataset is the same, but both did different things to the datasets. We also used a depot, and they didn't, which will result in big differences.

6.3 Limitations

There are a few limitations for the model. There is a maximum of two orders per bundle. When there are two orders in a bundle, a courier can only go from restaurant one to customer one then to restaurant two and order two. Not from restaurant one to restaurant two and then to the customers based on what the fastest way is. Because there is only one depot, the data that can be used is limited to datasets where the restaurants and houses of the customers aren't that far apart from each other. The model would be better if it was more dynamic, now it is only possible to change the assignment of a courier once it is back at the depot. Even if he is on his way to a restaurant where an order is almost ready that isn't in the courier's assigned bundle of orders.

6.4 Future research

In the future, it might be interesting to have multiple depots instead of one that is in the middle of everything. The couriers could also be implemented less static than they are now. They would be able to roam freely after an assignment and not always have to go back to the depot. Or even say no to an assignment that they get. Or maybe implement more uncertainties for the drivers and customers. Like different service times while delivering orders or different travel times for the couriers. Another interesting thing to do is to look at this problem differently and look for the optimal parameter settings to get the best solution for all three of the stakeholders. With parameter optimization.

6.5 Personal learnings

The model was way harder to make than I thought it would be. So I spend way more time writing the code that should be needed for a thesis. Next time doing research, I have to be more precise in what I want to research and not just have some sort of direction. Because we had a lot of different ideas before we actually started doing research, which got me in time trouble.

References

1: Reyes, D., Erera, A., Savelsbergh, M., Sahasrabudhe, S., & O'Neil, R. (2018). The Meal Delivery Routing Problem. https://optimization-online.org/?p=15139

2: Steever, Z., Karwan, M., & Murray, C. (2019). Dynamic courier routing for a food delivery service. https://www.sciencedirect.com/science/article/pii/S0305054819300681

3: Xue, G., Wang, Z., & Wang, G. (2021, May 25). Optimization of Rider Scheduling for a Food Delivery Service in O2O Business.
https://www.hindawi.com/journals/jat/2021/5515909/

4: Ding, X., Zhang, R., Mao, Z., Xing, K., Du, F., Liu, X., Wei, G., Yin, F., He, R., & Sun, Z. (2019). Delivery Scope: A New Way of Restaurant Retrieval for On-demand Food Delivery Service. https://dl.acm.org/doi/abs/10.1145/3394486.3403353

5: Zhu, L., Yu, W., Zhou, K., Wang, X., Feng, W., Wang, P., Chen, N., & Lee, P. (2020). Order Fulfillment Cycle Time Estimation for On-Demand Food Delivery.
https://dl.acm.org/doi/abs/10.1145/3394486.3403307

6: Chen, J., Wang, L., Wang, S., Wang, X., & Ren, H. (2021). An effective matching algorithm with adaptive tie-breaking strategy for online food delivery problem.
https://link.springer.com/article/10.1007/s40747-021-00340-x

7: Song, B. D., & Ko, Y. D. (2016). A vehicle routing problem of both refrigerated- and general-type vehicles for perishable food products delivery.
https://www.sciencedirect.com/science/article/pii/S0260877415003805

8: Zou, G., Tang, J., Yilmaz, L., & Kong, X. (2021). Online food ordering delivery strategies based on deep reinforcement learning.
https://link.springer.com/article/10.1007/s10489-021-02750-3

9: Liu, S., He, L., & Shen, Z. M. (2020). On-Time Last Mile Delivery: Order Assignment with Travel Time Predictors. https://pubsonline.informs.org/doi/abs/10.1287/mnsc.2020.3741

10: Ulmer, M. W., Thomas, B. W., Campbell, A. M., & Woyak, N. (2020). The Restaurant Meal Delivery Problem: Dynamic Pickup and Delivery with Deadlines and Random Ready Times. https://pubsonline.informs.org/doi/abs/10.1287/trsc.2020.1000

11: Liu, S., He, L., & Shen, M. (2018). Data-Driven Order Assignment for Last Mile Delivery.
https://www.researchgate.net/publication/325753030_Data-Driven_Order_Assignment_for_Last_Mile_Delivery

12: Boza, A., Cebik, M., Kavaklioglu, C., Kavuk, E. M., Tosun, A., Sonuc, S. B., Duranel, A., & Basar, A. (2022). Courier routing and assignment for food delivery service using reinforcement learning.
https://www.sciencedirect.com/science/article/pii/S0360835221007750

13: Joshi, M., Singh, A., Ranu, S., Bagchi, A., Karia, P., & Kala, P. (2022). Batching and Matching for Food Delivery in Dynamic Road Networks.
https://dl.acm.org/doi/abs/10.1145/3494530

14: The Brainy Insights (2022) Online Food Delivery Market Size to Hit USD 432.32 Billion by 2030; Technological Interventions in Food Services Platforms by Aggregator to Bolster Growth: The Brainy Insights.
https://finance.yahoo.com/news/online-food-delivery-market-size-115800842.html?guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAADWrpPI7ZSYAs9H9xKEGmsfZHnnI28mXQBE-I_7s0SV2gRP0VjgIk1ngVKYRmH1dO6E-T2I634RgW9ZmZOR0W2FQkWV-K7mguYgmCMqtIkk5eSKLYRP7Zr3h-9ziesRokXN-kR_jwKtIAa-6mAPE8tTmKxUm7SL12b7f23xqNE2q&guccounter=2

15: Grand View Research (2022) Online Food Delivery Services Market Size, Share & Trends Analysis Report By Channel Type (Mobile Application, Websites/Desktop), By Payment Method (COD, Online), By Type, By Region, And Segment Forecasts, 2022 - 2030.
https://www.grandviewresearch.com/industry-analysis/online-food-delivery-services-market

Appendix

results

Interval vs speed
dataset 1

	interval	speed	wait_time	total_distance	payment	unserverd_customers
0	2	288	38,59	3442,067	49,18	0
1	2	320	36,83	3460,795	48,95	0
2	2	400	33,77	3489,8	49,23	0
3	2	480	32,1	3495,328	48,74	0
4	3	288	39,27	3447,579	49,37	0
5	3	320	37,45	3454,659	49,24	0
6	3	400	34,5	3479,638	48,83	0
7	3	480	32,83	3482,696	48,86	0
8	4	288	39,91	3421,167	48,79	0
9	4	320	37,91	3445,861	48,38	0
10	4	400	35,44	3474,43	48,66	0
11	4	480	33,68	3484,226	49,25	0
12	5	288	40,12	3438,717	49,31	0
13	5	320	38,65	3447,973	48,9	0
14	5	400	35,46	3470,362	48,86	0
15	5	480	34,16	3490,094	49,17	0

dataset 2

	interval	speed	wait_time	total_distance	payment	unserverd_customers
0	2	288	40,7	3516,126	60,19	0
1	2	320	38,78	3557,995	59,96	0
2	2	400	35,61	3574,873	59,01	0

3	2	480	33,78	3576,954	59,63	0
4	3	288	41,22	3514,333	59,97	0
5	3	320	39,41	3519,363	59,9	0
6	3	400	35,86	3567,467	59,1	0
7	3	480	34,54	3579,891	59,34	0
8	4	288	42,22	3521,498	60,43	0
9	4	320	40,04	3525,685	59,38	0
10	4	400	37,13	3553,452	59,62	0
11	4	480	35,09	3556,444	59,38	0
12	5	288	42,16	3506,468	59,89	0
13	5	320	40,21	3531,331	59,92	0
14	5	400	37,3	3554,468	59,7	0
15	5	480	35,57	3551,013	59,41	0

dataset 3

	interval	speed	wait_time	total_distance	payment	unserverd_customers
0	2	288	37,21	4840,937	46,61	1
1	2	320	36,23	4821,064	47,59	2
2	2	400	33,94	4848,97	48,43	2
3	2	480	32,77	4853,749	49,38	1
4	3	288	37,92	4814,911	46,79	2
5	3	320	36,76	4828,451	47,49	1
6	3	400	34,52	4850,404	48,4	2
7	3	480	33,29	4847,545	49,46	2
8	4	288	38,4	4818,392	46,66	2
9	4	320	37,16	4820,965	47,49	2
10	4	400	34,97	4858,309	48,61	1

11	4	480	33,79	4862,236	49,57	1
12	5	288	39,06	4816,58	46,86	1
13	5	320	37,75	4828,892	47,07	1
14	5	400	35,53	4823,423	48,65	2
15	5	480	34,35	4841,035	48,9	2

n-drivers vs working time
dataset 1

	n-drivers	w-time	wait_time	total_distance	payment	unserverd_customers
0	113	100	38,65	3447,973	48,9	0
1	113	90	40,57	3423,089	48,59	0
2	113	80	43,04	3370,596	48,74	0
3	113	70	44,36	3324,192	48,8	0
4	102	100	39,4	3450,81	52,65	0
5	102	90	41,22	3403,984	52,83	0
6	102	80	42,92	3351,649	52,91	0
7	102	70	45,71	3305,552	52,65	0
8	89	100	40,95	3412,189	58,37	0
9	89	90	42,32	3369,57	59,13	0
10	89	80	45,67	3342,304	59,12	0
11	89	70	46,37	3280,77	58,58	1
12	79	100	42,03	3364,154	64,94	0
13	79	90	45,85	3326,073	65,47	0
14	79	80	47,26	3308,001	65,16	1
15	79	70	49,3	3283,854	65,53	0

dataset 2

	n-drivers	w-time	wait_time	total_distance	payment	unserverd_customers
0	94	100	40,21	3531,331	59,92	0

1	94	90	X	X	X	X
2	94	80	43,95	3449,439	59,32	0
3	94	70	47,34	3329,649	58,4	14
4	85	100	41,05	3503,063	64,59	0
5	85	90	X	X	X	X
6	85	80	44,17	3430,723	64,84	0
7	85	70	47,56	3312,643	63,62	12
8	75	100	43,04	3472,531	72,93	0
9	75	90	X	X	X	X
10	75	80	48,08	3423,051	72,97	0
11	75	70	47,15	3325,05	72,79	2
12	66	100	43,59	3426,049	81,82	2
13	66	90	X	X	X	X
14	66	80	47,18	3357,135	81,89	4
15	66	70	50,42	3360,067	82,41	0

dataset 3

	n-drivers	w-time	wait_time	total_distance	payment	unserverd_customers
0	194	100	37,75	4828,892	47,07	1
1	194	90	40,09	4782,006	44,87	1
2	194	80	43,88	4712,518	43,71	1
3	194	70	47,73	4548,162	41,95	13
4	175	100	38,36	4808,756	48,91	1
5	175	90	40,56	4770,834	47,37	1
6	175	80	44,16	4667,539	46,66	1
7	175	70	48,57	4499,466	44,24	16
8	155	100	38,69	4768,784	51,81	1

9	155	90	42,37	4703,85	50,39	1
10	155	80	45,2	4652,892	50,49	1
11	155	70	50,18	4467,237	48,31	16
12	136	100	39,79	4758,168	56,57	2
13	136	90	44,09	4650,985	55,97	1
14	136	80	46,12	4582,834	56,1	2
15	136	70	53,66	4479,326	55,07	8

speed vs number of drivers
dataset 1

	speed	n-drivers	wait_time	total_distance	payment	unserverd_customers
0	288	113	40,12	3438,717	49,31	0
1	288	102	41,21	3402,753	53,04	0
2	288	89	43,24	3387,686	58,88	0
3	288	79	43,37	3359,826	65,44	1
4	320	113	38,65	3447,973	48,9	0
5	320	102	39,4	3450,81	52,65	0
6	320	89	40,95	3412,189	58,37	0
7	320	79	42,03	3364,154	64,94	0
8	400	113	35,46	3470,362	48,86	0
9	400	102	36,26	3454,639	52,8	0
10	400	89	37,69	3431,847	58,76	0
11	400	79	38,16	3417,55	64,75	0
12	480	113	34,16	3490,094	49,17	0
13	480	102	34,4	3466,65	52,9	0
14	480	89	35,55	3445,763	59,21	0
15	480	79	36,57	3403,704	65,06	0

dataset 2

	speed	n-drivers	wait_time	total_distance	payment	unserverd_customers
0	288	94	42,16	3506,468	59,89	0
1	288	85	42,81	3487,37	65,26	0
2	288	75	44,44	3421,633	72,93	2
3	288	66	45,52	3451,193	82,88	0
4	320	94	40,21	3531,331	59,92	0
5	320	85	41,05	3503,063	64,59	0
6	320	75	43,04	3472,531	72,93	0
7	320	66	43,59	3426,049	81,82	2
8	400	94	37,3	3554,468	59,7	0
9	400	85	37,8	3523,469	64,74	0
10	400	75	38,88	3492,306	72,79	0
11	400	66	39,91	3468,058	82,35	0
12	480	94	35,57	3551,013	59,41	0
13	480	85	35,96	3531,118	64,69	0
14	480	75	37,11	3494,963	72,05	1
15	480	66	38,14	3464,017	81,59	1

dataset 3

	speed	n-drivers	wait_time	total_distance	payment	unserverd_customers
0	288	194	39,06	4816,58	46,86	1
1	288	175	39,59	4812,812	48,96	1
2	288	155	40,1	4750,361	51,68	1
3	288	136	41,53	4723,295	57,04	1
4	320	194	37,75	4828,892	47,07	1
5	320	175	38,36	4808,756	48,91	1
6	320	155	38,69	4768,784	51,81	1

7	320	136	39,79	4758,168	56,57	2
8	400	194	35,53	4823,423	48,65	2
9	400	175	35,88	4836,867	50,01	1
10	400	155	36,06	4811,492	52,65	2
11	400	136	36,95	4776,565	57,13	1
12	480	194	34,35	4841,035	48,9	2
13	480	175	34,55	4835,851	51,11	2
14	480	155	34,73	4824,469	53,06	2
15	480	136	35,47	4802,335	57,7	1

speed vs working time
dataset 1

	speed	w-time	wait_time	total_distance	payment	unserverd_customers
0	288	100	40,12	3438,717	49,31	0
1	288	90	43,11	3377,406	49,23	0
2	288	80	46	3354,256	49,29	0
3	288	70	48,3	3317,741	48,4	0
4	320	100	38,65	3447,973	48,9	0
5	320	90	40,57	3423,089	48,59	0
6	320	80	43,04	3370,596	48,74	0
7	320	70	44,36	3324,192	48,8	0
8	400	100	35,46	3470,362	48,86	0
9	400	90	37,95	3435,858	48,54	0
10	400	80	40,3	3412,96	48,29	0
11	400	70	41,61	3342,132	48,46	0
12	480	100	34,16	3490,094	49,17	0
13	480	90	35,9	3437,836	49,08	0
14	480	80	37,02	3401,639	48,69	0

15	480	70	39,73	3337,788	48,43	0

dataset 2

	speed	w-time	wait_time	total_distance	payment	unserverd_customers
0	288	100	42,16	3506,468	59,89	0
1	288	90	X	X	X	X
2	288	80	45,79	3417,262	59,57	3
3	288	70	49,29	3323,87	59,08	12
4	320	100	40,21	3531,331	59,92	0
5	320	90	X	X	X	X
6	320	80	43,95	3449,439	59,32	0
7	320	70	47,34	3329,649	58,4	14
8	400	100	37,3	3554,468	59,7	0
9	400	90	39,24	3486,481	59,23	2
10	400	80	41,4	3414,411	58,69	8
11	400	70	42,52	3358,118	58,14	12
12	480	100	35,57	3551,013	59,41	0
13	480	90	36,78	3545,343	59,26	0
14	480	80	38,38	3468,586	59,02	2
15	480	70	41,18	3358,412	58,81	12

dataset 3

	speed	w-time	wait_time	total_distance	payment	unserverd_customers
0	288	100	39,06	4816,58	46,86	1
1	288	90	41,2	4793,325	44,54	1
2	288	80	45,32	4688,194	43,62	1
3	288	70	50,14	4553,044	41,92	7
4	320	100	37,75	4828,892	47,07	1

5	320	90	40,09	4782,006	44,87	1
6	320	80	43,88	4712,518	43,71	1
7	320	70	47,73	4548,162	41,95	13
8	400	100	35,53	4823,423	48,65	2
9	400	90	37,37	4794,603	46,14	2
10	400	80	41,23	4711,484	44,3	1
11	400	70	45,53	4562,346	42,66	10
12	480	100	34,35	4841,035	48,9	2
13	480	90	36,7	4787,706	46,67	1
14	480	80	40,22	4739,761	45,14	1
15	480	70	43,96	4597,232	43,6	2

Speed

Speed vs waiting time:

Dataset 1	90% (40.12)	125% (35.46)	150% (34.16)
100% (38.65)	+3.8%	-8.3%	-11.6%
Dataset 2	90% (42.16)	125% (37.30)	150% (35.57)
100% (40.21)	+4.8%	-7.2%	-11.5%
Dataset 3	90% (39.06)	125% (35.53)	150% (34.35)
100% (37.75)	+3.4%	-5.8%	-9%
average	90%	125%	150%
100%	+4.0%	-7.1%	-10.7%

Speed vs Total distance

Dataset 1	90% (3438.7 km)	125% (3470.4 km)	150% (3490.1 km)
100% (3448.0 km)	-0.3%	+0.6%	+1.2%
Dataset 2	90% (3506.5 km)	125% (3554.5 km)	150% (3551.0 km)
100% (3531.3 km)	-0.7%	+0.7%	+0.6%
Dataset 3	90% (4816.6 km)	125% (4823.4 km)	150% (4841.0 km)

100% (4828.9 km)	-0.3%	-0.1%	+0.3%
average	90%	125%	150%
100%	-0.4%	+0.4%	+0.7%

Speed vs Payment

Dataset 1	90% (49.31)	125% (48.86)	150% (49.17)
100% (48.90)	+0.8%	-0.1%	+0.6%
Dataset 2	90% (59.89)	125% (59.70)	150% (59.41)
100% (59.92)	-0.1%	-0.4%	-0.9%
Dataset 3	90% (46.86)	125% (48.65)	150% (48.90)
100% (47.07)	-0.4%	+3.4%	+3.9%
average	90%	125%	150%
100%	+0.1%	+1.0%	+1.2%

Interval

Interval vs waiting time:

Dataset 1	4 (37.91)	3 (37.45)	2 (36.83)
5 (38.65)	-1.9%, 0.74 min	-3.1%, 1.2 min	-4.7%, 1.82 min
Dataset 2	4 (40.04)	3 (39.41)	2 (38.78)
5 (40.21)	-0.4%, 0.17 min	-2.0%, 0.8 min	-3.5%, 1.43 min
Dataset 3	4 (37.16)	3 (36.76)	2 (36.23)
5 (37.75)	-1.5%, 0.59 min	-2.6%, 0.99 min	-4.0%, 1.52 min
Average	4	3	2
5	-1.3%	-2.6%	-4.1%

Interval vs total distance

Dataset 1	4 (3445.9 km)	3 (3454.7 km)	2 (3460.8 km)
5 (3448.0 km)	-0.1%	+0.2%	+0.4%
Dataset 2	4 (3525.7 km)	3 (3519.4 km)	2 (3558.0 km)
5 (3531.3 km)	-0.2%	-0.3%	+0.8%
Dataset 3	4 (4821.0 km)	3 (4828.5 km)	2 (4821.1 km)
5 (4828.9 km)	-0.2%	-0.0%	-0.2%

average	4	3	2
5	-0.2%	-0.0%	+0.3%

Interval vs payment

Dataset 1	4 (48.38)	3 (49.24)	2 (48.95)
5 (48.90)	-1.1%	+0.7%	+0.1%
Dataset 2	4 (59.38)	3 (59.90)	2 (59.96)
5 (59.92)	-0.9%	+0.0%	+0.1%
Dataset 3	4 (47.49)	3 (47.49)	2 (47.59)
5 (47.07)	+0.9%	+0.9%	+1.1%
average	4	3	2
5	-0.4%	+0.5%	+0.4%

Working time
working time vs waiting time:

Dataset 1	90% (40.57)	80% (43.04)	70% (44.36)
100% (38.65)	+5.0%, 1.92 min	+11.4%, 4.39 min	+14.8%, 5.71 min
Dataset 2	90% (-)	80% (43.95)	70% (47.34)
100% (40.21)	-	+9.3%, 3.74 min	+17.7%, 7.13 min
Dataset 3	90% (40.09)	80% (43.88)	70% (47.73)
100% (37.75)	+6.2%, 2.34 min	+16.2%, 6.13 min	+26.4%, 9.98 min
Average	90%	80%	70%
100%	+5.6%	+12.3%	+19.6%

working time vs total distance:

Dataset 1	90% (3423.1 km)	80% (3370.6 km)	70% (3324.2 km)
100% (3448.0 km)	-0.7%	-2.2%	-3.6%
Dataset 2	90% (-)	80% (3449.4 km)	70% (3329.6 km)
100% (3531.3 km)	(-)	-2.3%	-5.7%
Dataset 3	90% (4782.0 km)	80% (4712.5 km)	70% (4548.2 km)

100% (4828.9 km)	-1.0%	-2.4%	-5.8%
average	90%	80%	70%
100%	-0.9%	-2.3%	-5.0%

Working time vs payment

Dataset 1	90% (48.59)	80% (48.74)	70% (48.80)
100% (48.90)	-0.6%	-0.3%	-0.2%
Dataset 2	90% (-)	80% (59.32)	70% (58.40)
100% (59.92)	-	-1.0%	-2.5%
Dataset 3	90% (44.87)	80% (43.71)	70% (41.95)
100% (47.07)	-4.6%	-7.1%	-10.9%
average	90%	80%	70%
100%	-2.6%	-2.8%	-4.5%

Number of drivers

Number of drivers vs waiting time:

Dataset 1	90% (39.40)	80% (40.95)	70% (42.03)
100% (38.65)	+1.9%, 0.75 min	+5.9%, 2.3 min	+8.7%, 3.38 min
Dataset 2	90% (41.05)	80% (43.04)	70% (43.59)
100% (40.21)	+2.1%, 0.84 min	+7.0%, 2.83 min	+8.4%, 3.38 min
Dataset 3	90% (38.36)	80% (38.69)	70% (39.79)
100% (37.75)	+1.6%, 0.61 min	+2.5%, 0.94 min	+5.4%, 2.04 min
Average	90%	80%	70%
100%	+1.9%	+5.1%	+7.5%

Number of drivers vs total distance

Dataset 1	90% (3450.8 km)	80% (3412.2 km)	70% (3364.2 km)
100% (3448.0 km)	+0.1%	-1.0%	-2.4%
Dataset 2	90% (3503.1 km)	80% (3472.5 km)	70% (3426.0 km)
100% (3531.3 km)	-0.8%	-1.7%	-3.0%
Dataset 3	90% (4808.8 km)	80% (4768.8 km)	70% (4758.2 km)

100% (4828.9 km)	-0.4%	-1.2%	-1.5%
average	90%	80%	70%
100%	-0.4%	-1.3%	-2.3%

Number of drivers vs payment

Dataset 1	90% (52.65)	80% (58.37)	70% (64.94)
100% (48.90)	+7.7%	+19.4%	+32.8%
Dataset 2	90% (64.59)	80% (72.93)	70% (81.82)
100% (59.92)	+7.8%	+21.7%	+36.5%
Dataset 3	90% (48.91)	80% (51.81)	70% (59.57)
100% (47.07)	+3.9%	+10.1%	+26.6%
average	90%	80%	70%
100%	+6.5%	+17.1%	+32.0%