Second year project II (EBS2003)

DHL Case

Maastricht University

Econometrics & Operations Research

Name & Student ID:	Eunji Kim	I6073884
	Aline Fattal	I6093879
	Zhenzheng Jia	I6100566

Supervisor: Alain Hecq, Matthias Mnich

Date submitted: July 10, 2017

Contents

Part I

Management Summary

Dear DHL LLP managers, we are students from Maastricht university who have worked on a project of 'Forecasting and transporting shipments of 2015'. We would like to give you a brief explanation of what we have achieved during our work on the case. Note that the case is composed in two parts which are based on the Econometrics and Operations research.

For the Econometrics part, our aim is to forecast the aggregated weights of May and June 2016 based on the data of 2015. To do the work, firstly we had to preselect the data. As we are given with 577,412 observations, we have selected top 20 big customers and decided to work with the shipments made by them. Even though the 20 customers we chose are only 0.19 percent of the whole number of customers, the total number of shipments made by them is 43 percent. Hence, we thought it is quite reasonable and efficient to work with the shipments made by the top 20 big customers as the data is reduced yet still can observe a trend from the regression. This part was easily done by using excel. One remark about data set is that the data was corrupted more than we expected. Hence, if you can implement more efficient method to clean the data, it would be so much easier for the ones who work on data analysis part. After the data cleaning step, we have performed the unit root and results that the aggregated weights for each service are stationary. Then, we have tested for seasonality and dynamics of the model. As a result, we have found that adding weekday dummies gives better fit than adding monthly dummies and the best ARMA model is an AR(2) and MA(3). Lastly, we have added explanatory variables and realized volatility to see if there is any impact on our regression. We can conclude that adding explanatory variables gives much better fit as an adjusted R-squared value increased significantly than without them. Furthermore, we have found that adding realized volatility gives insignificant affect to our regression. Hence, we can conclude that business cycle indicators do not give that much of an impact on our regression. Note that throughout testing the regression with respect to different criteria, we have forecasted aggregated weight of service 72h of May and June 2016. From observing the graphs, there are large amount of transactions during the weekdays and low amount during the weekend. The forecast graphs are plotted in each section of the paper.

Under the Operations research part, our aim is to plan for the shipments between the station to which it is brought from the supplier and the station that it needs to be delivered as a destination. We used the program java to complete our mission. We are given with the information excel files that contain substitutable cities in case if there are no route from city A to city B, road legs, cost information and all the shipments. To implement the data into the program, we have modified the given excel files and created classes. Detail process can be found in the second part of the paper. Then we have separated parts of our process into finding a feasible route of each shipment using heuristic. Next, we have implemented a function to find the feasible route for the all the shipments. We have used the main ideas of the 'Local search algorithm' in order to improve the heuristic and find an optimal route based on the results of the feasible route function. As a result, we have that the feasible routes returned by the feasible route function is the same as the optimal routes for all the shipments. Therefore, we can conclude that the feasible routes are also in fact the optimal routes. The running time of our program is very fast given the large input data, with about 438 ms.

As we worked on the actual big data problem for the first time, we had some issues to figure out many problems at the beginning but our general impression is we did start to enjoy working with the real case. I hope you enjoy reading our report as we enjoyed working with the case!

Part II

EM part

1 Introduction

This paper is a report on the econometrics part for the case introduced by the company DHL during the course of 'Second year project II (EBS2003)'.

The Lead Logistics Partner (DHL LLP) is a neutral partner and it's a global forwarding division within the DHL. It is responsible for managing change across the entire supply chain by using continuous improvement and cost reduction with the help of lean logistics processes and optimized logistics network.

The case will try to give a forecast of the volume of DHL (=aggregated weights) for the big customers for the month of May and June in 2016. In order to make some inference, we need to clean and preselect the data. In the initial data, there are 577,412 observations (shipments). For every shipment, we have the following information in the excel file which are the ID, the hawbOrg (house air waybill Origin, in other words the origin station), the date available for shipment, the origin country, destination station, name of the supplier, airline identifier (with a digit number), mawb (master air waybill which is a transport document), csne_ name (unique identifier for consignee name), ctrycode (country for destination), descrop (the type of shipment 24h,48h or 72 h), descmnprod(not relevant), arrival date (of the shipment), ActlFlightDate (which corresponds to the date of start of transportation) and finally the ActualWeight.

2 Outline of the paper

In this report, we have followed the steps mentioned in the case. We have started with the selection of the 'big customers', which corresponds to the top 20 suppliers. Then, we have aggregated the data on the daily basis and have fixed the corrupted data before checking the proprieties of the series. Next, we have tested for unit roots in aggregated weights of shipments for each service (24h, 48h and 72h). Furthermore, we tested for seasonality by using monthly and weekday dummy variables and we looked at the dynamics of the residuals to determine the best ARMA model. Finally, we have tried to forecast the volume for the months mentioned above using the best model found and including some extra explanatory variables and the business cycle.

3 Pre-selection of the data

3.1 Description

In order to make inferences about the database, we have restricted the analysis of DHL to only the 'Big Customers'. To select the 'Big Customers' we have considered different criteria such as the volume of shipments. We have decided to select 20 significant suppliers based on the calculations of the aggregated weights of the shipments.

The reason for a selection of 20 suppliers versus 50 suppliers, 10% or 20% quantile is that the 20 suppliers permits to reduce the data by about 37% representation of the weights of the whole sample and 43% of the total observations while choosing the 50 top suppliers corresponds to about 50% of the weights of whole sample as well as the total observations. Hence, the reduction of data more than 50% would make our analysis clearer yet not losing so much information.

3.2 Implementation (explain used excel)

After the reduction by finding 'big customers', we got 249,472 observations out of the original data. In table 2, the 'top 20 big customers' with their aggregate weight are presented. The range of aggregate weight contributed by the 'top 20 biggest customers' is from 3449945.9 ton by 'Supplier_ 4867' to 482112.3 ton by 'Supplier_ 8658'. In the table 2, we highlighted the effects of the data after the reduction by only taking the 'top 20 biggest customer'. That is the 0.19% of total customers which takes up 37.46% of the total weight and 43.21% of the total number of shipments.

Rank	Name	Aggregate Weight	Rank	Name	Aggregate Weight
1	Supplier_4867	3449945.9	11	Supplier_736	816428.65
2	Supplier_941	2963258.6	12	Supplier_2552	781995.6
3	Supplier_5688	2530002.9	13	Supplier_8675	771298.5
4	Supplier_10235	1349652.3	14	Supplier_3920	721834.5
5	Supplier_3467	1294469.6	15	Supplier_2260	703632.1
6	Supplier_937	930898.7	16	Supplier_1809	640070.3
7	Supplier_8442	929930.3	17	Supplier_7018	638661.9
8	Supplier_5262	885444.1	18	Supplier_5299	528799.5
9	Supplier_1844	870178	19	Supplier_7226	514337.1
10	Supplier_3847	864645.1	20	Supplier_8658	482112.3

Table 1: Top 20 suppliers forms the biggest aggregated weight of their shipments

****	Total sample	Subsample	Percentage of the whole sample
Number of customers	10325	20	0.19%
Total weight of shipments	60,509,057.97	22,667,596	37.46%
Total number of shipments	577,412	249,472	43.21%

Table 2: The effects after data reduction by 'Big Customers'

4 Times series Aggregation

After the selection of the biggest customers we have aggregated the weights on daily basis. We have decided to focus our analysis on the whole sample year of 2015, since it contains less than 500 observations. In total, the sub-sample contains 365 observations. In this time series, we have found only 3 non-working days such as the 24, 30 and the 31 of December, therefore we just filled those cells with value 0.

Next, we have proceeded to 'fix' the data. In a big part of the data, there were many dates which were missing and incorrect. We have decided to fill up these empty and wrong dates' cells. At some occasions, there was no sending date but only an arrival date(example in row 5 of table 3), we filled in the sending date(ActlFlightDate) by the date cell. In other observations, some arrival dates were missing (examples in row 3 and 4 of table 3), we filled in the dates by the service time that the customer took. For example, if customer chose 24hr service, we set the arrival date is one day after sending date and if customer chose 48hr service, we set the arrival as two days after sending date, and so on. Therefore, all the empty date cells were filled in. For those cells that have dates, but the dates(MM/DD/YYYY or DD/MM/YY) are different as the standard dates form(DD/MM/YYYY) (examples in row 1, 5, 6 and 7 of table 3). We changed the dates to the standard dates form.

hawbOrg	date	org	dst	name	ctrycode	ArrivalDate	ActlFlightDate
LUX	301014	LU	IST	Supplier_10224	TR	11/03/2014	11/01/2014
BCN	71014	ES	PRG	Supplier_485	CZ	10/09/2014	10/09/2014
QHZ	101214	NL	VIE	Supplier_3932	AT		12/11/2014
QHZ	101214	NL	GRQ	Supplier_3932	NL		12/11/2014
SPL	240215	NL	FRA	Supplier_5299	DE	26/02/2015	
BFS	120814	GB	STR	Supplier_8202	DE	13/08/2014	08/12/2014
DUB	31214	IE	BHX	Supplier_2095	GB	12/04/2014	12/03/2014

Table 3: Examples of missing and incorrect dates

Then, new explanatory variables have been introduced in order to increase the fit of the model such as the number shipments (nbrshipment$_s$ with $s = 1, 2, 3$), number of suppliers (nbrshipment$_s$ with $s = 1, 2, 3$), average weight of the shipments (Avgw$_s$ where $s = 1, 2, 3$), standard deviation of the weights of shipments (sdw$_s$ where $s = 1, 2, 3$), and of course the total number of shipments and suppliers. A sample of the data (From 1/01/2015 to 10/01/2015) for the month of January is featured in the table below. Since the separated customers types' table is extremely wide, we set the sample table only with daily total average weight, standard deviation and customer. The setting of the separated data is similar as the daily aggregated data.

Date	AW	Sn	AvgW	SDoW	DailyCust
01/01/2015	3307.9	8	413.4875	748.8880184	1
02/01/2015	15185.1	68	223.3102941	281.5395331	10
03/01/2015	3559.4	20	177.97	394.9309708	4
04/01/2015	14670.9	18	815.05	2583.035344	1
05/01/2015	20510.7	78	262.9576923	578.1722402	14
06/01/2015	11654.4	41	284.2536585	784.0931606	11
07/01/2015	17165.5	84	204.3511905	368.1124663	17
08/01/2015	10372.1	48	216.0854167	343.5494743	14
09/01/2015	37367.5	100	373.675	872.7637703	15
10/01/2015	3196.9	23	138.9956522	200.4993164	4

Table 4: Daily aggregated data

5 Times series proprieties

In the next step of this case, we have created the data based on daily basis. The daily aggregated data has been divided based on the service time of the shipments such as Aw_s $(s = 1, 2, 3)$ where AW1 denotes the Aggregated Daily Weights for 24 hours, AW2 stands for 48 hours and AW3 stands for 72 hours.

To test for the unit root problem, the Augmented Dickey-Fuller test has been performed. Firstly, we have performed the test based on the dependent variables of the regression which are Aw_s. As a result, we have found that there is no unit problem for all the series Aw_s $(s = 1, 2, 3)$ which implies that each of different separate services are stationary variables. Hence, the variables are stationary and therefore, we do not need to take the first difference of the aggregated weights.

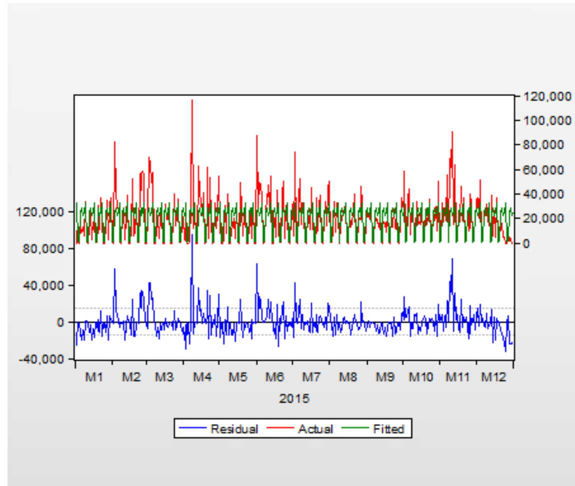
In the table 5, the test results are showed for the series AW3 where there was a probability value of 0.000 which implies that the H0 of a unit problem has been rejected at a 5% significance level. Furthermore, we have performed the same test as mentioned above for the new set of explanatory variables introduced. Then, we found that there is a unit problem only for the variables such as $\text{totalnbrcustomer}_s$, $\text{totalnbrshipment}_s$, nbrshipment_s and nbrshipment_s. To make the series stationary, the first difference of the series in theses variables were taken. Note that the other explanatory variables are stationary. Lastly, from the graph (Graph 1), we observe there is no problem regarding heteroskedasticity and autocorrelation.

Null Hypothesis: AW3 has a unit root
Exogenous: Constant
Lag Length: 6 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-4.179922	0.0008
Test critical values:	1% level	-3.448414	
	5% level	-2.869396	
	10% level	-2.571023	

*MacKinnon (1996) one-sided p-values.

Table 5: Unit root test for the series of AW3



Graph 1: Thegraph of the residuals of AW3

6 Deterministic seasonality

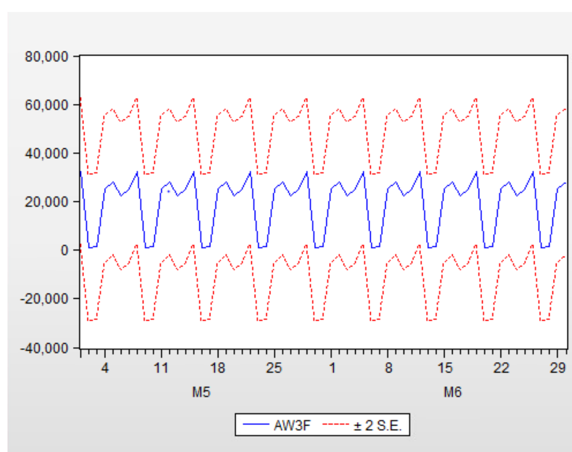
Next step, to test for the seasonality, dummy variables have been introduced in the model. Since we can observe from the graph that the series may have monthly or daily spikes, we took dummies for the weekdays and monthly. Note that to avoid dummy variable trap, we do not include dummy for Sunday on the weekdays dummies and exclude dummy for December on the monthly dummies as the intercept term has been included. Next, the regression of the series Aw_s has been performed twice including only the weekdays dummies and only with the monthly dummies. We observed that including the weekday dummies gives a much better fit than including the monthly dummies. Furthermore, the regression of Aw_3 gives the best model. As a result, we have decided to work with weekly dummies and Aw_3 as the best model for the rest part of our analysis.

Dependent Variable: AW3
Method: Least Squares
Date: 06/22/17 Time: 16:45
Sample: 1/01/2015 12/31/2015
Included observations: 365

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	1718.977	2073.436	0.829048	0.4076
@WEEKDAY=1	23478.81	2932.281	8.007010	0.0000
@WEEKDAY=2	26567.17	2932.281	9.060240	0.0000
@WEEKDAY=3	20685.01	2932.281	7.054239	0.0000
@WEEKDAY=4	23284.09	2918.417	7.978328	0.0000
@WEEKDAY=5	30752.30	2932.281	10.48750	0.0000
@WEEKDAY=6	-911.2250	2932.281	-0.310756	0.7562

R-squared	0.389694	Mean dependent var	19428.03
Adjusted R-squared	0.379465	S.D. dependent var	18980.57
S.E. of regression	14951.76	Akaike info criterion	22.08204
Sum squared resid	8.00E+10	Schwarz criterion	22.15683
Log likelihood	-4022.972	Hannan-Quinn criter.	22.11176
F-statistic	38.09847	Durbin-Watson stat	1.737870
Prob(F-statistic)	0.000000		

Table 6: regression on weekday dummies of AW3.



Graph 2: Forcast of regression on weekday dummies of AW3 for May and June 2016

We can plot the forcast for May and June 2016 of our best model which is based on 72 hour service. There is a downwards trend which means there is a large amount of transactions during the weekdays and low amount during the weekend. Lastly, to check for the misspecification such as the presence of heteroscedasticity in the model and autocorrelation in the residual term, we have performed a White test with cross terms and an LM test. Firstly, from the white test, the null hypothesis of homoscedasticity has been rejected at a 5% significance level. Secondly from the LM test, we strongly reject the null hypothesis of no autocorrelation in the residuals. In order to correct for these misspecifications, the HCSE estimator is used to robustify the standard OLS estimator which corrects for the heteroscedasticity in the model and the HAC estimator is used for the autocorrelation in the model.

7 Dynamics (determine the ARMA model)

In this step, we have tried to determine the best ARMA model for each service. From the residual diagnostics correlogram using the ACF and the PACF, there are no appropriate ARMA models for 24h and 48h service. However, for the 72h service, as you can observe in the appendix 36, the best ARMA model found is an AR(2) and a MA(3).

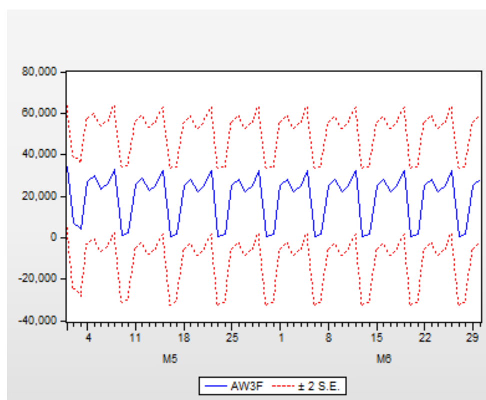
Then, take the regression of the best ARMA model to forcast AW3 for May and June 2016.

Dependent Variable: AW3
Method: ARMA Maximum Likelihood (OPG - BHHH)
Date: 06/23/17 Time: 14:37
Sample: 1/01/2015 12/31/2015
Included observations: 365
Convergence achieved after 34 iterations
Coefficient covariance computed using outer product of gradients

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	1457.479	6150.355	0.236975	0.8128
@WEEKDAY=1	23484.24	5650.665	4.156013	0.0000
@WEEKDAY=2	26520.30	4956.368	5.350753	0.0000
@WEEKDAY=3	20650.09	5704.546	3.619936	0.0003
@WEEKDAY=4	23409.90	5815.868	4.025177	0.0001
@WEEKDAY=5	30699.17	5865.627	5.233740	0.0000
@WEEKDAY=6	-959.6274	8088.627	-0.118639	0.9056
AR(1)	0.827066	0.858704	0.963156	0.3361
AR(2)	0.008033	0.594279	0.013517	0.9892
MA(1)	-0.745218	0.857098	-0.869467	0.3852
MA(2)	0.150145	0.525220	0.285870	0.7751
MA(3)	-0.098283	0.144144	-0.681839	0.4958
SIGMASQ	2.03E+08	10859742	18.71647	0.0000

R-squared	0.434260	Mean dependent var	19428.03
Adjusted R-squared	0.414974	S.D. dependent var	18980.57
S.E. of regression	14517.67	Akaike info criterion	22.03952
Sum squared resid	7.42E+10	Schwarz criterion	22.17843
Log likelihood	-4009.213	Hannan-Quinn criter.	22.09473
F-statistic	22.51621	Durbin-Watson stat	1.993384
Prob(F-statistic)	0.000000		

Table 7: regression of the best ARMA model



Graph 3: Forcast of regression on weekday dummies and the ARMA model of AW3 for May and June 2016

8 Impact of explanatory variables (perform regression with the extra lags values added to the model)

Dependent Variable: AW3
Method: ARMA Maximum Likelihood (OPG - BHHH)
Date: 06/22/17 Time: 18:00
Sample: 1/02/2015 12/31/2015
Included observations: 364
Convergence achieved after 31 iterations
Coefficient covariance computed using outer product of gradients

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-1560.436	2635.592	-0.592063	0.5542
@WEEKDAY=1	-69.59200	2588.532	-0.026885	0.9786
@WEEKDAY=2	6701.056	2105.324	3.182910	0.0016
@WEEKDAY=3	6300.064	2295.980	2.743954	0.0064
@WEEKDAY=4	6719.204	2167.909	3.099393	0.0021
@WEEKDAY=5	15512.77	2220.975	6.984668	0.0000
@WEEKDAY=6	10955.70	2609.841	4.197843	0.0000
AVGW3	25.98489	3.380873	7.685853	0.0000
D(NBRCUSTOMER3)	-128.8797	219.5341	-0.587060	0.5575
D(NBRSHIPMENT3)	129.0732	8.730075	14.78489	0.0000
SDW3	11.83661	0.820052	14.43398	0.0000
AR(1)	0.757324	2.119961	0.357235	0.7211
AR(2)	0.175540	2.027250	0.086590	0.9310
MA(1)	-0.480840	2.118191	-0.227005	0.8206
MA(2)	-0.266577	1.436447	-0.185581	0.8529
MA(3)	0.004595	0.286407	0.016044	0.9872
SIGMASQ	55123638	3013716.	18.29092	0.0000

R-squared	0.846547	Mean dependent var	19481.40
Adjusted R-squared	0.839472	S.D. dependent var	18979.25
S.E. of regression	7604.224	Akaike info criterion	20.75761
Sum squared resid	2.01E+10	Schwarz criterion	20.93962
Log likelihood	-3760.884	Hannan-Quinn criter.	20.82995
F-statistic	119.6426	Durbin-Watson stat	1.997017
Prob(F-statistic)	0.000000		

Table 8 : regression on the best model with explanatory variables

In this step of the analysis, we have added the explanatory variables to the regression with the ARMA (2,3) model. Note that it is necessary to take the first difference of the variables for nbrcustomers and nbrshipment to make the series stationary.

As observed in the table 8, the regression with the explanatory variables gives a much better fit to the model, with an adjusted R-squared of 0.83 while in the other model without those variables has a value of 0.41. In other words, the goodness of fit of the model has increased with these added explanatory variables.

Note that there is no possibility of having co-integration since we have stationary series. For the Engle and Ganger test which is a test for co-integration, we need to have that the series are non- stationary I(1) series. But in our analysis, we strongly reject the null hypothesis of non-stationary series. Therefore, we know that we do not satisfy the assumption of the Engle and Granger test which implies we cannot test for the co-integration in the model.

9 Impact of business cycle

It is likely that the volume is affected by the business cycle fluctuations and in particular by the uncertainty of the economy. To see if that is the case, we have included a measure of the volatility of financial markets. We have chosen to add realized volatility of the following indices such as the S&P500 for America, FTSE100 for Europe and Hang Seng for Asia expecting adding these variables would affect our regression. However, we can observe that the values for the realized volatility are very are low which implies that they become insignificant in the regression. This could be explained from observing the small difference of the adjusted-R squared value with and without these variables.

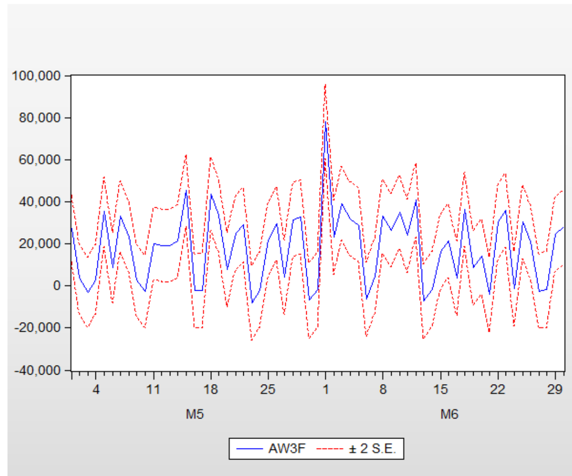
Next, observe that there is no trend in the forcast in the buinsness cycle therefore it is insignificant. This contradicted our expectation that the realized volatility would affect our regression.

Dependent Variable: AW3
Method: ARMA Maximum Likelihood (OPG - BHHH)
Date: 06/22/17 Time: 18:02
Sample: 1/02/2015 12/31/2015
Included observations: 364
Convergence achieved after 32 iterations
Coefficient covariance computed using outer product of gradients

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	-1516.327	2610.293	-0.580903	0.5617
@WEEKDAY=1	189.0191	2624.022	0.072034	0.9426
@WEEKDAY=2	7114.971	2287.924	3.109793	0.0020
@WEEKDAY=3	6711.550	2424.806	2.767871	0.0059
@WEEKDAY=4	7144.171	2378.785	3.003286	0.0029
@WEEKDAY=5	15897.81	2360.035	6.736260	0.0000
@WEEKDAY=6	11091.58	2612.377	4.245779	0.0000
AVGW3	26.04492	3.416341	7.623630	0.0000
D(NBRCUSTOMER3)	-109.5136	221.8456	-0.493648	0.6219
D(NBRSHIPMENT3)	128.9844	8.728367	14.77761	0.0000
SDW3	11.76949	0.836647	14.06746	0.0000
RVAM	1008848.	12805594	0.078782	0.9373
RVASIA	-617803.0	5190814.	-0.119019	0.9053
RVEU	-8240858.	24402808	-0.337701	0.7358
AR(1)	0.744237	1.856587	0.400863	0.6888
AR(2)	0.182958	1.772650	0.103212	0.9179
MA(1)	-0.464863	1.853396	-0.250817	0.8021
MA(2)	-0.271259	1.248020	-0.217351	0.8281
MA(3)	0.008078	0.252614	0.031976	0.9745
SIGMASQ	55046624	3053397.	18.02800	0.0000

R-squared	0.846762	Mean dependent var	19481.40
Adjusted R-squared	0.838298	S.D. dependent var	18979.25
S.E. of regression	7631.973	Akaike info criterion	20.77266
Sum squared resid	2.00E+10	Schwarz criterion	20.98679
Log likelihood	-3760.625	Hannan-Quinn criter.	20.85777
F-statistic	100.0457	Durbin-Watson stat	1.996942
Prob(F-statistic)	0.000000		

Table 9: regression on the best model adding realized volatility variables.



Graph 4: forcast with the buisness cycle

10 Times series vs panel

As the given data in our case is quite broad and could be explained in many ways, we can perform various regressions than just time series. As an example, we can carry out panel data based on our observations. Panel data can be explained by a combination of cross sectional and time series. Hence, it is basically taking multi-dimensional data in a specific time periods. In our case, we can easily observe that our data fits more to time series regression as each observation is based on specific date and we would like to analyse if there is a sequence effect from previous time periods. Furthermore, performing panel data on our case would lead to high variability as the regression is based on one day instead of whole time periods.

11 Conclusion

Working with 577,412 observations was definitely not an easy job as as just cleaning data itself took us few days before starting analysis. Next, we had to make a decision how to define the size of big customers we would like to analyse. After we decided to work with top 20 big customers, we started to analyse by following the steps given to us using the program called 'Eviews'. We were afraid that working with only 20 customers might lead the regressions to have a lower fit. However, in general our analysis came out to be fairly reasonable. Some remarks are as follow. We have found that there exists a relationship between seasonality and shipment observations. Furthermore, adding explanatory variables such as the number of shipments, number of suppliers, average weight of the shipments, standard deviation of the weights and finally the total number of shipments and suppliers improved the fit of our model. One regard that did not meet our assumption is the impact of business cycle. Although we expected adding business cycle variables would affect the model, they were statistically insignificant to the regression. Overall, we are satisfied with the analysis as it turned out to meet our assumptions during our discussion before starting the actual analysis.

Part III
OR part

1 Introduction

Under the Operations research part of the case, we need to figure out a route with a minimum cost for each shipment of DHL between the origins and the destinations. The data of all the shipments, road legs, cost information and the file with the information of the substitutable cities are given and we have used the program 'Java' to mainly solve the problem.

To start with our task, a feasible route is necessary to be found. Hence, the preliminary step is to find whether there is a feasible route from the origin city to the destination city. Then, we use a local search algorithm to find an optimal route given a feasible route found from the previous step. Note that we have taken the shipments from 5/01/2015 until 11/02/2015 as an input in the program to reduce the running time as well as we are given to test the process on this specific week.

In this report, there are 3 sections. The first section describes the data structure and the modelling of the problem. Next section explains the algorithm we use to determine the feasible routes and the optimal routes. Then, the last part shows the result that we have achieved.

2 Data structure and the modelling of the problem

2.1 Data structure

To begin tackling this project, we chose to optimize the data as much as possible through excel. We have proceeded numerous modifications to the files given to us. The shipments given to us have all different ID's, origins and destinations, departure dates, times and weights. There is a time constraint that needs to be satisfied such as the shipments need to be delivered within 24hr, 48 and 72 hours.

Firstly, we have decided to include only the shipments from 5/01/2015 until 11/02/2015 and all the shipments outside of this period have been deleted from the excel file since the whole set of data is too large to run in the program and as we are given to analyse that specific week. The shipments have been sorted in increasing order according to the date. Next, we have added a dummy variable where the value is 0 if the shipment is done by airline and else 1. Note that it is only possible to ship the shipments via the road truck or the commercial truck carriers but not by airline. Therefore, the program will report if there are shipments by airline.

Secondly, in the file 'relationsorg' (resp. 'relationsdst'), in the column 'origin' represents the real origin (resp. destination) and the column B represents the city that we can substitute it with. So, we have replaced all the unreachable origins (resp. destinations) in the DATASTEAFT file by their respective city in column B. Unfortunately, some shipments cannot be processed as there are unreachable cities that cannot be replaced by the substitutes. Thirdly, we have replaced all strings in the file by numbers to ease the readability of the data in the program. Thus, we have assigned the values 0 to 6 for the days of Monday to Sunday, replaced the time of any events such as the departure of a truck by the number of minutes elapsed from the beginning of the week and each city assigned by numbers (in Appendix 1).

Lastly, we have extracted the information from the excel file into textfile such that the program can read the data (such as the cities24.txt, shipment27.txt, routes27.txt).

2.2 Modelling of the problem

We have decided to structure the data in the following manner. We have created a class 'city12', 'map12', 'road12' and 'shipment12'. Then, we have created the class 'caseproject20' that contains the main functions of the case and the shipment file which is imported.

Firstly, in the class 'road12', the following variables have been initialized: date at which the truck leaves, the ID of the departure city (= 'origin_nbre'), the 'departure_time', the ID of the destination city (= 'destination_nbre'), the 'arrival_time', the 'maximum_capacity per truck', 'cost', 'unit_cost' where we put the value 1 if the cost depends on the kg amount and else if it is fixed, then take the value 0.

We have assumed that there is a truck leaving every week from the same departure to the same destination. Therefore, an array named 'capacity_week' is created to represent the capacity of the two trucks in two weeks (we assumed that there is another truck with the same departure and destination city leaving next week).

The class 'city12' contains the ID number of the city, the string name of the city and an ArrayList named 'departure_list'. This integer array contains the departure time of the road that departs from that city and the ID of the road. It will be used as well in the function 'put_departure' defined in class 'road12'. This function 'put_departure' will add the road to its corresponding city inside this ArrayList in chronological order. This sorting helps to find the jth road which satisfies the departure time of the shipment. Then, we know that all the other roads after the jth road also satisfies the time constraints and it will be easy to track down all the possible roads which can be used.

Secondly, the class 'map12' contains the roads and the cities as observed from the figure 2 in the Appendix.

Thirdly, the class 'shipment12' contains the following variables: dummy variable if it is by airline or not, the id of the shipment, origin of the shipment, date of shipment destination, the time of shipment, average weight of the shipment and another variable which adds the day and the time of the shipment (in minutes).

Lastly, the class 'caseproject20' contains an instance of the class 'map' and an array of the class type 'shipment12'. Next, it contains the functions to find a feasible route for a given shipment and a function to find an optimal route from the given feasible route. These functions use the class 'map12'. Then, we have a function called 'feasible_route_forallshipment' which calls both functions defined above and finds a feasible (resp. optimal) route for all the shipments.

2.3 Description of the algorithm

2.3.1 Algorithm to find a feasible route for one shipment

We have implemented a constructive heuristic to find a feasible solution for all the shipments. We begin by checking if there is a feasible road from the origin to the destination for a given shipment. If there is no direct road yet, we set the destination from the previous road found to the origin. Then, the function is called by recursion until we reach the final destination. Furthermore, note that the cost is the main criteria of the constructive heuristic to find the feasible route. Lastly, we need to check if the constraints of the capacity are satisfied and the arrival and departure time match the shipment times of each possible route. In the next part of the paper, the key ideas of our thought process will be presented and as well as the steps of the algorithm.

Steps of the algorithm to find a feasible route

1. Some steps before starting to check if there is a direct route or not

1.1 Check if the origin number or the destination number of the shipment is larger than the size of the cities array. If yes, we have decided to put a large value (1000 as origin/destination number) when there was no road to the origin or to the destination (hence, the shipment cannot be send).

1.2. Check if the arrival time of the shipment is within the time due of the shipment. Note that the arrival time is equal to the time at which the shipment is made available plus 18 hours. Then, the time is transformed into minutes for the origin city. The arrival time will be updated by adding 2 hours to the current arrival time since the truck can leave only after 2 hours from the arrival time at the city.

1.3. Check if the origin and the destination of the shipment are same or not.

1.3.1 If it is the same, then the shipment has arrived at the destination and stop the recursion of the function.

1.4. Check if there are no cycles (whether the city is same as one of the visited cities previously).

2. Check for all the roads where the departure time is after the arrival time of the shipment (Details are in the modelling of the problem part).

3. Check if there is a direct route in all the feasible routes.

3.1 If yes, then check if it satisfies the capacity constraint.

3.1.1 If yes, then a feasible direct route from the current city to the destination has been found. The solution and the cost are updated. Finally, the function returns true and the recursion stops.

4. If no direct route is found yet, then we need:

4.1. The following List ”sum_cost_for_route’ is initialized with all the routes leaving the origin city by the cost and ID number of the road respectively.

4.2. Next, bubble sort algorithm is applied to the list defined above such that the first route in the list returns the smallest cost. Then, we pick the first element of this list and check:

4.2.1 If capacity constraint is satisfied. Then, this road is added to the ’roads_array’ and the capacity, arrival, total cost amount and the origin city will be updated and the function will be called by recursion.

4.2.2 If any of them return false (such as the capacity constraint and the feasible route function), then we choose the next road from the list ’sum_cost_for_route’.

4.2.3 The function will return false if there are no more possible roads in the list ’sum_cost_for_route’ to be checked.

2.3.2 Algorithm to find optimal routes for shipments from the feasible solutions

During the work of this case, we have considered many possible algorithms to find optimal routes. For example, we thought of using Dijkstra’s algorithm. However, as we are restricted by the running time, Dijkstra algorithm cannot be applied to find the most optimal route based on the current feasible route solutions that we have found. Next, we have decided to consider local search algorithm. The local search algorithm is an iterative algorithm that moves from one solution S to another S’ according to some neighbourhood structure. We will start from a set of initial feasible solution found with the feasible route function and then the solutions have been improved by applying local changes. This type of changes is called ’iterative improvement heuristic’. Applying this algorithm to our case, if we have a current feasible solution, then search the neighbourhood to find a more optimal solution. More specifically, we have decided to use PLS (stands for Polynomial-time Local Search) which is one of the Local search algorithm. This algorithm finds the most optimal solution among all the feasible routes of each neighbourhood which is searchable in polynomial time.

In the next part, the paper explains the algorithm to find the optimal routes from our feasible route function based on PLS.

Steps of the algorithm to find a feasible route

We use the list of 'roads_array' that we got from the feasible route function in the optimal route function. Hence, pass the list of 'roads_array' as an input argument. Then, we have decided to build the optimal feasible function with the following main structure.

1. The big 'for loop' goes through all the optimal roads from an intermediate city (returned by the feasible route function) to the destination (excluding the destination as a possible city to check from). At the end of this 'for loop', it checks for all feasible roads from all the intermediate cities to the destination.

Since there is no previous road to the first city, we decided to take this case as a special case which explains why the 'for loop' starts at i =1 and not at i =0. The first 'for loop' will stop before the last element since we have added the cost of the route for this shipment at the last position of the list 'roads_array'. Then, the 'for loop' with g returns the gth position in the departure list at the ith city which has a departure time larger than the arrival time.

2. Find a new route and calculate the cost difference.

2.1 We are trying to find a new route which has a direct route from a city to another city within the same neighbourhood. For example, in appendix 4, we have illustrated this step with a set of following cities A, B, C, D and E. Essentially, we are trying to find a direct route between the cities but excluding the roads returned by the feasible route function. All the possible roads found will be saved in the list 'candidate'. The second 'for loop' iterates through all the roads of the departure list of the city determined by the first 'for loop' and it searches for another road connecting to another city in the neighbourhood as explained above.

2.2 Next, we calculate the difference of the cost of this new possible route found and the current feasible route. Then, this possible route found is added to the list 'candidate' with the difference in costs respectively, id number of the road, the ith position at which the current feasible solution is in the 'roads_array' and the cost of this new possible solution (and the number of roads which are deleted).

3. At the end of both 'for loop', a small test is performed to check if the candidate list is empty or not.

3.1. If the candidate list is empty, then the function will return false.

3.2. If the candidate list is not empty, then the function 'bubble sort' is applied to the 'candidate_list'.

3.2.1 If there is a positive cost difference in the sorted list 'candidate_list', then the current solution will be updated by the new solution found. Next, the capacity will be updated. The first highest difference is taken to be the first candidate to ensure that the cost of the current feasible route is reduced by the largest possible amount.

3.2.2. The roads which have been replaced by this new optimal solution found are deleted from the list of 'roads_array' and the new optimal solution is added to the 'roads_array' list. Note that the optimal feasible route function is called. If there are no more possible optimal routes, in other words when the function returns false, all the possibilities in the neighbourhood have been checked. Then, the optimal feasible route is found which updates the current feasible route (this step corresponds to step 7 in appendix 3).

2.3.3 Function to find the feasible route for all the shipments

The function begins by checking if the shipment needs to be transported by airline or not and it will report in the program if it cannot be shipped. The variables that are used in this function is as follow: the 'time_due' which is the time at which the shipment needs to be at the final destination (by 6 am), 'origin_nbre' (the starting city of the shipment), destination_nbre (destination of the shipment), a list 'named_roads' which contains the ID number of all roads used to ship one shipment. The function 'optimizing_current_feasibleroutes' will be called only if there was a feasible route found from the function 'feasible_route_one_shipment'. We can conclude that the main difference between the feasible route function and the optimal route function is the following. In the feasible route function, we just take the first smallest cost road found after the array of all possible roads have been sorted. However, in the optimal feasible route function, we check for all the remaining roads and we choose firstly to update the current feasible solution by the road found where the cost will induce the largest reduction.

3 Output and the solution

3.1 Output of the program

Firstly, the program outputs the results of the program into another text file. It reports if the shipment can be shipped or not. Then, the optimal route function outputs the ID of the shipment, the ID of the roads of the solution in the respective order from the origin to the destination with the cost. Next, the program reports if no feasible route has been found. Lastly, the program outputs the respective running time for all the shipments at a specific day and the last running time value represents the total running time of the program.

3.2 Running time

We observe that the running time of this program is within three minutes for days where the shipments are between 100 and 1000 shipments (In our case, Monday). And for the days where the shipments are more than 1000, the running time is approximately five minutes (In our case, from Tuesday to Sunday). The total running time for this program is 438 ms. Further detailed information can be found in the output text file of the program. (as observed in in Appendix 5)

3.3 Impression about the output and discussion

We have observed that the optimal feasible route function returned essentially the same roads as the feasible route function for all the shipments. Because, the feasible route function has not used more than two intermediate cities to ship the shipment. Therefore, it is very likely that the feasible routes that we have found is indeed the optimal routes.

3.4 Conclusion

Throughout the operations research part of the case, we had to find the feasible routes and the optimal routes for all the shipments and print out the shipments of specific week which is from 5/01/2015 until 11/02/2015. To reach the goal, as it is described in section 2, we have structured the data and inserted them in several classes to implement in the program 'Java'. Next, we have used the heuristic to exam whether there exists a feasible route for each individual shipment and print them out. Then, we have tried to figure out the optimal route among all the shipments that have feasible routes. We have concluded that the feasible routes found are in fact also the optimal solution for all the shipments. Lastly, we have exported the result into a text file. The paper concludes with summing up the result. From the shipments of the week we have used to implement, there are 1816 shipments that have found the feasible route and 2472 shipments that do not have feasible route. Also, we have found that 4210 shipments cannot be transported by the given constraints. Total running time of the program is 438 ms.

Part IV

References

E.H.L. Aarts and J.K. Lenstra (eds.)(1997), *"Local search in combinatorial optimization"*, John Wiley & Sons, Chichester, UK)

Jos´e Fernando Oliveira, Maria Ant´onia Carravilla (2009) *"Heuristics and Local Search [Powerpoint slides]"* Retrieved from http://paginas.fe.up.pt/ mac/ensino/docs/OR/CombinatorialOptimizationHeuristicsLocalSearch.pdf

Part V

Appendix

1 EM part

Null Hypothesis: AW1 has a unit root
Exogenous: Constant
Lag Length: 0 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-18.52637	0.0000
Test critical values:	1% level	-3.448111	
	5% level	-2.869263	
	10% level	-2.570952	

*MacKinnon (1996) one-sided p-values.

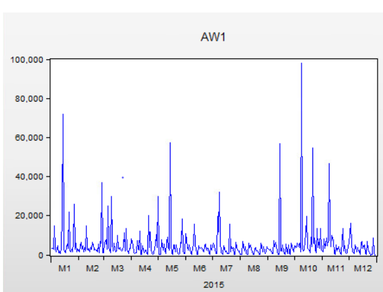
Table 1: Unit root test for the series of AW1

Null Hypothesis: AW2 has a unit root
Exogenous: Constant
Lag Length: 6 (Automatic - based on SIC, maxlag=16)

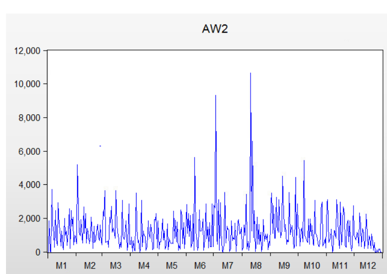
		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-4.849085	0.0001
Test critical values:	1% level	-3.448414	
	5% level	-2.869396	
	10% level	-2.571023	

*MacKinnon (1996) one-sided p-values.

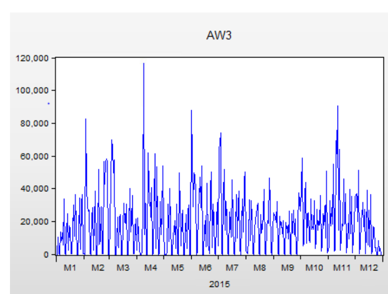
Table 2: Unit root test for the series of AW2



Graph 1: forcast with the buisness cycle



Graph 2: Series plot of AW2



Graph 3: Series plot of AW3

Null Hypothesis: TOTNBRSHIPMENT has a unit root
Exogenous: Constant
Lag Length: 13 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-1.437098	0.5643
Test critical values:	1% level	-3.448782	
	5% level	-2.869558	
	10% level	-2.571110	

*MacKinnon (1996) one-sided p-values.

Table 3: Unit root test for the total number of shipments

Null Hypothesis: TOTNBRCUSTOMER has a unit root
Exogenous: Constant
Lag Length: 16 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-1.491233	0.5371
Test critical values:	1% level	-3.448943	
	5% level	-2.869629	
	10% level	-2.571148	

*MacKinnon (1996) one-sided p-values.

Table 4: Unit root test for the total number of customers

Null Hypothesis: NBRSHIPMENT1 has a unit root
Exogenous: Constant
Lag Length: 6 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-2.376635	0.1491
Test critical values:	1% level	-3.448414	
	5% level	-2.869396	
	10% level	-2.571023	

*MacKinnon (1996) one-sided p-values.

Table 5: Unit root test for the number of shipments of AW1

Null Hypothesis: NBRSHIPMENT2 has a unit root
Exogenous: Constant
Lag Length: 13 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-2.781656	0.0619
Test critical values:	1% level	-3.448782	
	5% level	-2.869558	
	10% level	-2.571110	

*MacKinnon (1996) one-sided p-values.

Table 6: Unit root test for the number of shipments of AW2

Null Hypothesis: NBRSHIPMENT3 has a unit root
Exogenous: Constant
Lag Length: 13 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-1.691230	0.4349
Test critical values:	1% level	-3.448782	
	5% level	-2.869558	
	10% level	-2.571110	

*MacKinnon (1996) one-sided p-values.

Table 7: Unit root test for the number of shipments of AW3

Null Hypothesis: NBRCUSTOMER1 has a unit root
Exogenous: Constant
Lag Length: 13 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-1.672182	0.4446
Test critical values:	1% level	-3.448782	
	5% level	-2.869558	
	10% level	-2.571110	

*MacKinnon (1996) one-sided p-values.

Table 8: Unit root test for the number of customers of AW1

Null Hypothesis: NBRCUSTOMER2 has a unit root
Exogenous: Constant
Lag Length: 13 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-1.654015	0.4538
Test critical values:	1% level	-3.448782	
	5% level	-2.869558	
	10% level	-2.571110	

*MacKinnon (1996) one-sided p-values.

Table 9: Unit root test for the number of customers of AW2

Null Hypothesis: NBRCUSTOMER3 has a unit root
Exogenous: Constant
Lag Length: 16 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-1.264596	0.6469
Test critical values:	1% level	-3.448943	
	5% level	-2.869629	
	10% level	-2.571148	

*MacKinnon (1996) one-sided p-values.

Table 10: Unit root test for the number of customers of AW3

Null Hypothesis: SDW has a unit root
Exogenous: Constant
Lag Length: 0 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-17.03530	0.0000
Test critical values:	1% level	-3.448111	
	5% level	-2.869263	
	10% level	-2.570952	

*MacKinnon (1996) one-sided p-values.

Table 11: Unit root test for the standard deviation of the total weight

Null Hypothesis: SDW1 has a unit root
Exogenous: Constant
Lag Length: 0 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-17.39142	0.0000
Test critical values:	1% level	-3.448111	
	5% level	-2.869263	
	10% level	-2.570952	

*MacKinnon (1996) one-sided p-values.

Table 12: Unit root test for the standard deviation of the AW1

Null Hypothesis: SDW2 has a unit root
Exogenous: Constant
Lag Length: 1 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-11.93860	0.0000
Test critical values:	1% level	-3.448161	
	5% level	-2.869285	
	10% level	-2.570963	

*MacKinnon (1996) one-sided p-values.

Table 13: Unit root test for the standard deviation of the AW2

Null Hypothesis: SDW3 has a unit root
Exogenous: Constant
Lag Length: 8 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-4.455808	0.0003
Test critical values:	1% level	-3.448518	
	5% level	-2.869442	
	10% level	-2.571047	

*MacKinnon (1996) one-sided p-values.

Table 14: Unit root test for the standard deviation of the AW3

Null Hypothesis: AVGW has a unit root
Exogenous: Constant
Lag Length: 0 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-18.30048	0.0000
Test critical values:	1% level	-3.448111	
	5% level	-2.869263	
	10% level	-2.570952	

*MacKinnon (1996) one-sided p-values.

Table 15: Unit root test for the standard deviation of the AW1

Null Hypothesis: AVGW1 has a unit root
Exogenous: Constant
Lag Length: 0 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-18.80379	0.0000
Test critical values:	1% level	-3.448111	
	5% level	-2.869263	
	10% level	-2.570952	

*MacKinnon (1996) one-sided p-values.

Table 16: Unit root test for the average weight of AW1

Null Hypothesis: AVGW2 has a unit root
Exogenous: Constant
Lag Length: 1 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-11.42084	0.0000
Test critical values:	1% level	-3.448161	
	5% level	-2.869285	
	10% level	-2.570963	

*MacKinnon (1996) one-sided p-values.

Table 17: Unit root test for the average weight of AW2

Null Hypothesis: AVGW3 has a unit root
Exogenous: Constant
Lag Length: 8 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-5.057394	0.0000
Test critical values:	1% level	-3.448518	
	5% level	-2.869442	
	10% level	-2.571047	

*MacKinnon (1996) one-sided p-values.

Table 18: Unit root test for the average weight of AW3

Null Hypothesis: AW has a unit root
Exogenous: Constant
Lag Length: 6 (Automatic - based on SIC, maxlag=16)

		t-Statistic	Prob.*
Augmented Dickey-Fuller test statistic		-4.188614	0.0008
Test critical values:	1% level	-3.448414	
	5% level	-2.869396	
	10% level	-2.571023	

*MacKinnon (1996) one-sided p-values.

Table 19: Unit root test for the average weight of the shipments

Dependent Variable: AW1
Method: Least Squares
Date: 06/22/17 Time: 16:29
Sample: 1/01/2015 12/31/2015
Included observations: 365

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	5639.787	979.0077	5.760718	0.0000
@TREND	-2.017380	4.655296	-0.433352	0.6650

Table 20: regression with the trend on AW1

Wald Test:
Equation: EQ05

Test Statistic	Value	df	Probability
F-statistic	140.7836	(6, 358)	0.0000
Chi-square	844.7016	6	0.0000

Null Hypothesis: C(1)=C(2)=C(3)=C(4)=C(5)=C(6)=0
Null Hypothesis Summary:

Normalized Restriction (= 0)	Value	Std. Err.
C(1)	1718.977	2073.436
C(2)	23478.81	2932.281
C(3)	26567.17	2932.281
C(4)	20685.01	2932.281
C(5)	23284.09	2918.417
C(6)	30752.30	2932.281

Restrictions are linear in coefficients.

Table 21: Wald test for the weekday dummies on AW3.

Dependent Variable: AW1
Method: Least Squares
Date: 06/22/17 Time: 16:48
Sample: 1/01/2015 12/31/2015
Included observations: 365

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	3203.145	1669.026	1.919170	0.0558
@MONTH=1	4164.803	2360.359	1.764478	0.0785
@MONTH=2	1837.144	2422.758	0.758286	0.4488
@MONTH=3	2323.268	2360.359	0.984286	0.3256
@MONTH=4	861.7748	2379.948	0.362098	0.7175
@MONTH=5	3038.858	2360.359	1.287456	0.1988
@MONTH=6	464.8748	2379.948	0.195330	0.8452
@MONTH=7	1522.206	2360.359	0.644905	0.5194
@MONTH=8	-623.8258	2360.359	-0.264293	0.7917
@MONTH=9	1366.342	2379.948	0.574106	0.5663
@MONTH=10	6889.861	2360.359	2.918988	0.0037
@MONTH=11	2878.655	2379.948	1.209545	0.2273

R-squared	0.044253	Mean dependent var	5272.624
Adjusted R-squared	0.014471	S.D. dependent var	9360.720
S.E. of regression	9292.744	Akaike info criterion	21.14418
Sum squared resid	3.05E+10	Schwarz criterion	21.27240
Log likelihood	-3846.813	Hannan-Quinn criter.	21.19513
F-statistic	1.485888	Durbin-Watson stat	2.036055
Prob(F-statistic)	0.134581		

Table 22: Regression on monthly dummies on AW1

Dependent Variable: AW2
Method: Least Squares
Date: 06/22/17 Time: 16:50
Sample: 1/01/2015 12/31/2015
Included observations: 365

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	678.9935	219.0301	3.100001	0.0021
@MONTH=1	514.9774	309.7554	1.662529	0.0973
@MONTH=2	629.7707	317.9442	1.980759	0.0484
@MONTH=3	653.8323	309.7554	2.110802	0.0355
@MONTH=4	267.5298	312.3260	0.856572	0.3923
@MONTH=5	273.9323	309.7554	0.884350	0.3771
@MONTH=6	667.5965	312.3260	2.137499	0.0332
@MONTH=7	804.3290	309.7554	2.596659	0.0098
@MONTH=8	701.9613	309.7554	2.266180	0.0240
@MONTH=9	1070.880	312.3260	3.428724	0.0007
@MONTH=10	525.7387	309.7554	1.697271	0.0905
@MONTH=11	600.1881	312.3260	1.921672	0.0555

R-squared	0.047244	Mean dependent var	1236.630
Adjusted R-squared	0.017555	S.D. dependent var	1230.355
S.E. of regression	1219.508	Akaike info criterion	17.08261
Sum squared resid	5.25E+08	Schwarz criterion	17.21082
Log likelihood	-3105.576	Hannan-Quinn criter.	17.13356
F-statistic	1.591283	Durbin-Watson stat	2.156866
Prob(F-statistic)	0.099304		

Table 23: Regression on monthly dummies on AW2

Dependent Variable: AW3
Method: Least Squares
Date: 06/22/17 Time: 16:52
Sample: 1/01/2015 12/31/2015
Included observations: 365

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	14530.10	3358.171	4.326791	0.0000
@MONTH=1	-2054.106	4749.171	-0.432519	0.6656
@MONTH=2	10271.48	4874.721	2.107090	0.0358
@MONTH=3	5487.897	4749.171	1.155548	0.2486
@MONTH=4	9090.733	4788.584	1.898418	0.0585
@MONTH=5	-871.4516	4749.171	-0.183496	0.8545
@MONTH=6	10019.16	4788.584	2.092301	0.0371
@MONTH=7	8648.700	4749.171	1.821097	0.0694
@MONTH=8	1527.229	4749.171	0.321578	0.7480
@MONTH=9	938.3468	4788.584	0.195955	0.8448
@MONTH=10	5827.810	4749.171	1.227122	0.2206
@MONTH=11	10771.54	4788.584	2.249421	0.0251

R-squared	0.058930	Mean dependent var	19428.03
Adjusted R-squared	0.029604	S.D. dependent var	18980.57
S.E. of regression	18697.50	Akaike info criterion	22.54249
Sum squared resid	1.23E+11	Schwarz criterion	22.67071
Log likelihood	-4102.005	Hannan-Quinn criter.	22.59345
F-statistic	2.009521	Durbin-Watson stat	1.806898
Prob(F-statistic)	0.026707		

Table 24: Regression on monthly dummies on AW3

Dependent Variable: AW1
Method: Least Squares
Date: 06/22/17 Time: 16:44
Sample: 1/01/2015 12/31/2015
Included observations: 365

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	2294.188	1259.545	1.821442	0.0694
@WEEKDAY=1	6004.075	1781.265	3.370680	0.0008
@WEEKDAY=2	3099.710	1781.265	1.740173	0.0827
@WEEKDAY=3	5168.423	1781.265	2.901546	0.0039
@WEEKDAY=4	5526.385	1772.843	3.117244	0.0020
@WEEKDAY=5	1857.192	1781.265	1.042625	0.2978
@WEEKDAY=6	-855.7365	1781.265	-0.480409	0.6312

R-squared	0.074037	Mean dependent var	5272.624
Adjusted R-squared	0.058518	S.D. dependent var	9360.720
S.E. of regression	9082.707	Akaike info criterion	21.08512
Sum squared resid	2.95E+10	Schwarz criterion	21.15992
Log likelihood	-3841.035	Hannan-Quinn criter.	21.11485
F-statistic	4.770734	Durbin-Watson stat	1.976113
Prob(F-statistic)	0.000107		

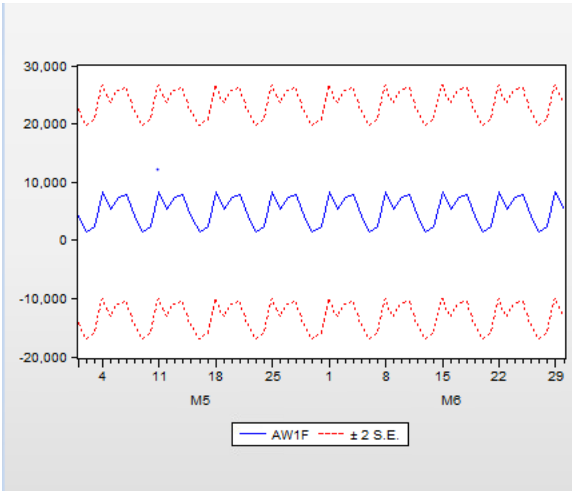
Table 25: Regression on weekday dummies on AW1

Dependent Variable: AW2
Method: Least Squares
Date: 06/22/17 Time: 16:45
Sample: 1/01/2015 12/31/2015
Included observations: 365

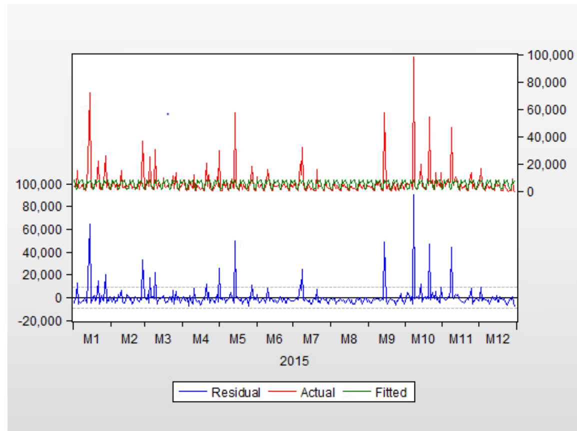
Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	511.0567	144.9628	3.525434	0.0005
@WEEKDAY=1	1966.693	205.0083	9.593235	0.0000
@WEEKDAY=2	687.7298	205.0083	3.354643	0.0009
@WEEKDAY=3	1180.859	205.0083	5.760052	0.0000
@WEEKDAY=4	437.4640	204.0390	2.144021	0.0327
@WEEKDAY=5	910.2837	205.0083	4.440227	0.0000
@WEEKDAY=6	-98.47596	205.0083	-0.480351	0.6313

R-squared	0.290035	Mean dependent var	1236.630
Adjusted R-squared	0.278136	S.D. dependent var	1230.355
S.E. of regression	1045.342	Akaike info criterion	16.76107
Sum squared resid	3.91E+08	Schwarz criterion	16.83586
Log likelihood	-3051.895	Hannan-Quinn criter.	16.79079
F-statistic	24.37502	Durbin-Watson stat	1.914572
Prob(F-statistic)	0.000000		

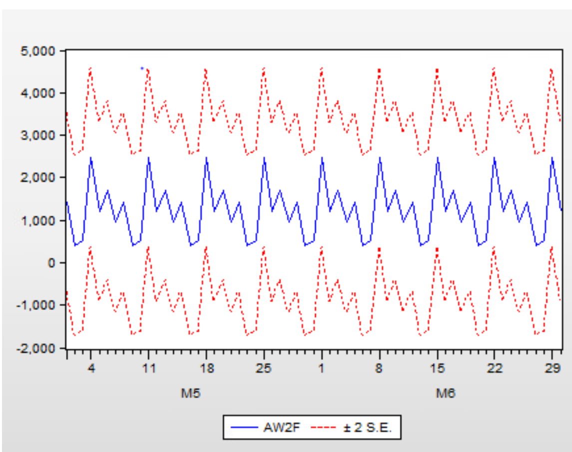
Table 26: Regression on weekday dummies on AW2



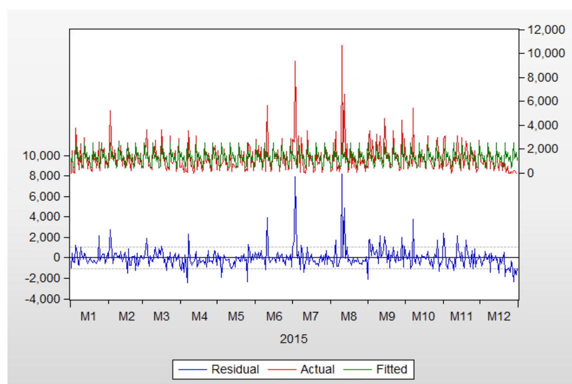
Graph 4: Forcast of regression on weekday dummies of AW1 for May and June 2016



Graph 5: The graph of the residuals of AW1



Graph 6: Forcast of regression on weekday dummies of AW2 for May and June 2016



Graph 7: The graph of the residuals of AW2

Heteroskedasticity Test: White

F-statistic	4.030435	Prob. F(6,358)	0.0006
Obs*R-squared	23.09538	Prob. Chi-Square(6)	0.0008
Scaled explained SS	95.30835	Prob. Chi-Square(6)	0.0000

Table 27 : White test on the regression of the AW3

Breusch-Godfrey Serial Correlation LM Test:

F-statistic	12.30814	Prob. F(2,356)	0.0000
Obs*R-squared	23.60630	Prob. Chi-Square(2)	0.0000

Table 28 : LM test

Date: 06/22/17 Time: 17:16
Sample: 1/01/2015 12/31/2015
Included observations: 365

Autocorrelation	Partial Correlation		AC	PAC	Q-Stat	Prob
		1	0.123	0.123	5.5906	0.018
		2	0.233	0.222	25.706	0.000
		3	0.104	0.058	29.687	0.000
		4	0.069	0.003	31.457	0.000
		5	0.085	0.047	34.174	0.000
		6	0.040	0.008	34.770	0.000
		7	0.131	0.102	41.237	0.000
		8	-0.003	-0.044	41.241	0.000
		9	0.100	0.054	45.011	0.000
		10	0.013	-0.010	45.076	0.000
		11	-0.011	-0.050	45.119	0.000
		12	-0.014	-0.032	45.198	0.000
		13	-0.072	-0.065	47.170	0.000
		14	-0.002	0.006	47.172	0.000
		15	-0.057	-0.023	48.416	0.000
		16	-0.052	-0.054	49.438	0.000
		17	-0.063	-0.031	50.980	0.000
		18	-0.081	-0.045	53.512	0.000
		19	0.032	0.083	53.921	0.000
		20	-0.151	-0.119	62.830	0.000
		21	0.060	0.085	64.217	0.000
		22	0.025	0.094	64.468	0.000
		23	0.047	0.041	65.333	0.000
		24	0.002	-0.029	65.334	0.000
		25	0.025	0.026	65.584	0.000
		26	-0.069	-0.102	67.489	0.000
		27	-0.097	-0.073	71.248	0.000
		28	0.002	0.000	71.250	0.000
		29	-0.030	0.015	71.614	0.000
		30	-0.005	-0.020	71.626	0.000

Table 29: Residual diagnostics correlogram for the 72h service

Dependent Variable: AW3
Method: ARMA Maximum Likelihood (OPG - BHHH)
Date: 06/22/17 Time: 17:48
Sample: 1/01/2015 12/31/2015
Included observations: 365
Convergence achieved after 34 iterations
Coefficient covariance computed using outer product of gradients

Variable	Coefficient	Std. Error	t-Statistic	Prob.
C	1457.479	6150.355	0.236975	0.8128
@WEEKDAY=1	23484.24	5650.665	4.156013	0.0000
@WEEKDAY=2	26520.30	4956.368	5.350753	0.0000
@WEEKDAY=3	20650.09	5704.546	3.619936	0.0003
@WEEKDAY=4	23409.90	5815.868	4.025177	0.0001
@WEEKDAY=5	30699.17	5865.627	5.233740	0.0000
@WEEKDAY=6	-959.6274	8088.627	-0.118639	0.9056
AR(1)	0.827066	0.858704	0.963156	0.3361
AR(2)	0.008033	0.594279	0.013517	0.9892
MA(1)	-0.745218	0.857098	-0.869467	0.3852
MA(2)	0.150145	0.525220	0.285870	0.7751
MA(3)	-0.098283	0.144144	-0.681839	0.4958
SIGMASQ	2.03E+08	10859742	18.71647	0.0000

R-squared	0.434260	Mean dependent var	19428.03
Adjusted R-squared	0.414974	S.D. dependent var	18980.57
S.E. of regression	14517.67	Akaike info criterion	22.03952
Sum squared resid	7.42E+10	Schwarz criterion	22.17843
Log likelihood	-4009.213	Hannan-Quinn criter.	22.09473
F-statistic	22.51621	Durbin-Watson stat	1.993384
Prob(F-statistic)	0.000000		

Inverted AR Roots	.84	-.01	
Inverted MA Roots	.73	.01-.37i	.01+.37i

Table 30: Regression with the ARMA (2,3) model

2 OR part

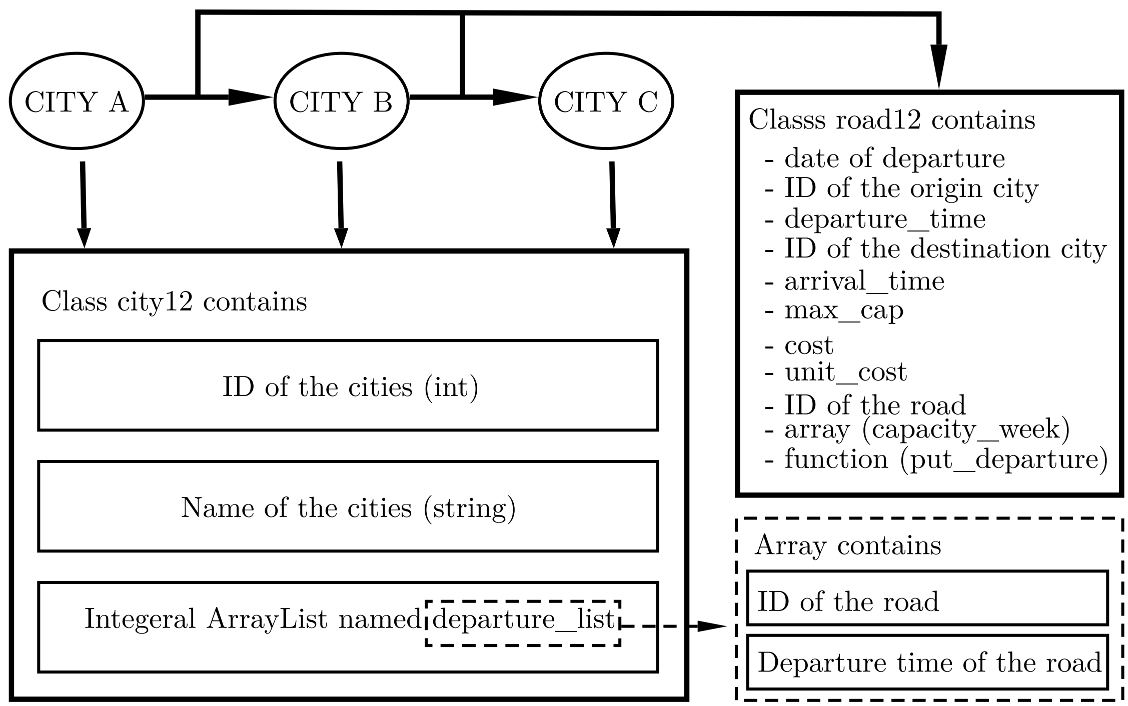
2.1 Appendix 1

Table 5: Table with the cities

City name	number	City name	number	City name	number
AMS (Amsterdam)	0	GOT (Gothenburg)	16	PAR (Paris)	32
ATH(Athens)	1	GRZ (Graz)	17	PRG (Prague)	33
BEG (Belgrad)	2	HEL (Helsinki)	18	RIX (Riga)	34
BHX (Birmingham)	3	INN (Innsbruck)	19	SJJ (Sarajevo)	35
BLL (Billund)	4	LIL(LILLE)	20	SKP (Skopje)	36
BRQ(BRNO)	5	LIS (Lisbon)	21	SOF (Sofia)	37
BRU (Brussels)	6	LJU (Ljubljana)	22	STO (Stockholm)	38
BSL (Basel)	7	LNZ (Linz)	23	SZG (Salzburg)	39
BTS (Bratislava)	8	LUX (Luxembourg)	24	TLL (Talinn)	40
BUD (Budapest)	9	LYS (Lyon)	25	TLL (Tallinn)	41
BUH (Bucharest)	10	MAD (Madrid)	26	VFQ (Vienna)	42
CPH (Copenhagen)	11	MIL (Milano)	27	VNO (Vilnius)	43
CVT (Coventry)	12	MMA (Malmoe)	28	WAW (Warszawa)	44
DUS (Dusseldorf)	13	MSQ (Minsk)	29	ZAG (Zagreb)	45
FAG (Fagersta)	14	ORB (Orebro)	30		
FRA (Frankfurt)	15	OSL (Oslo)	31		

2.2 Appendix 2

Figure 1: Graphic illustration of the class 'map12' (with example of 3 cities named A,B and C)



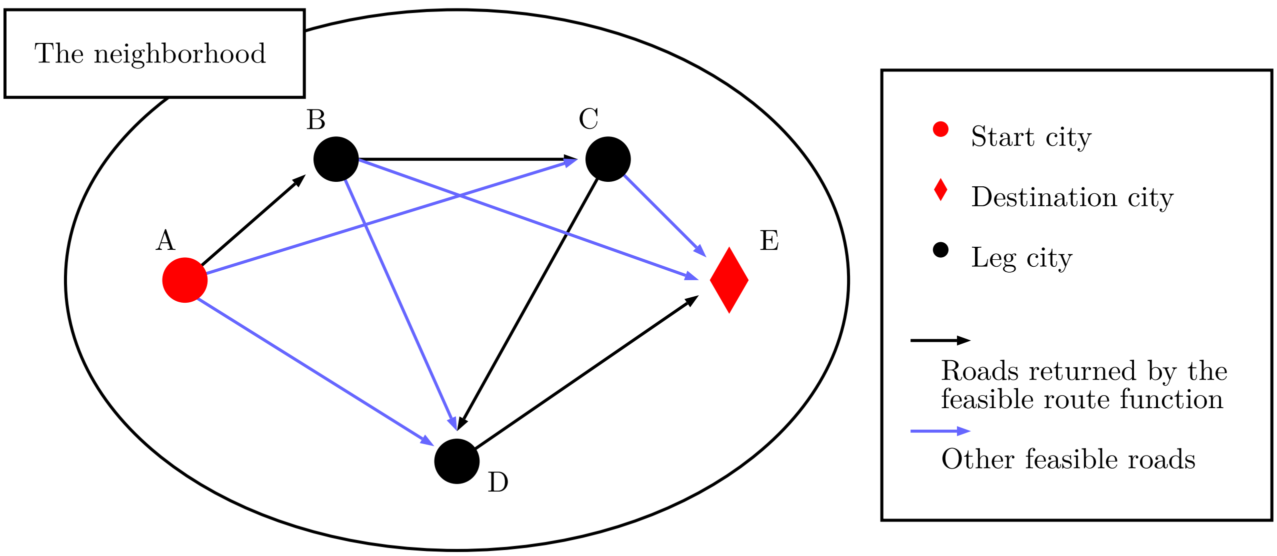
2.3 Appendix 3

Pseudo code of a general local search algorithm (J. F. Oliveira & M. A. Carravilla ,2009)

1. Generate an initial solution $\rightarrow s_0$.
2. Current solution $s_i = s_0$. // current feasible solution
3. Pick $s_j \in V(s_i)$. // the set $V(s_i)$ represents the set of solutions of the neighbourhood of s_i
4. If $f(s_j) < f(si)$, then $s_i = s_j$. // comparing cost of the new solution chosen to the current cost solution
5. Else, $V(s_i) = V(s_i)s_j$. // if the current solution was better the new possible solution will be deleted form the set of solutions in the neighbourhood.
6. If $V(s_i) = \varnothing$, then go to 3. // while the set of solutions in the neighbourhood is not empty
7. Else, END. Local optimal solution $= s_i$. // current solution is updated by the new solution found

2.4 Appendix 4

Figure 2: Local search algorithm illustration



2.5 Appendix 5

Running of the program

the day is 0and the number of shipments is 695 and the time is 1.499694899989E12ms
the day is 1and the number of shipments is 1776 and the time is 122.0ms
the day is 2and the number of shipments is 2601 and the time is 57.0ms
the day is 3and the number of shipments is 3464 and the time is 33.0ms
the day is 4and the number of shipments is 4358 and the time is 22.0ms
the day is 5and the number of shipments is 4551 and the time is 8.0ms
the day is 6and the number of shipments is 4568 and the time is 1.0ms
The total running time is:438.0 ms