New York Subway Challenge: Linear Optimization

Iñigo Martinez[1], Jon Otegui[1], Francisco J. Planes[1], Luis Tobalina[1] and John E. Beasley[2],
[1]CEIT and TECNUN, University of Navarra, Manuel de Lardizabal 15, 20018 San Sebastian, Spain
[2]Mathematical Sciences, Brunel University, Kingston Lane, UB8 3PH, Uxbridge, UK

Abstract:

The New York Subway Challenge is a competition where participants must pass through all the New York Subway stations in the shortest time possible. Each station has been considered as a node of a graph, and the time needed to travel between stations as the arc weights. The main issue of this graph approach has been the systematic generation of subtours, due to the special characteristics of the graph and the inability to find proper constraints. In order to deal with this, in this work we propose the creation of the directed line graph of the original network. With this procedure and by means of the linear optimization methods, we have found the optimal solution for the New York Subway Challenge.

Contact: imartinezl@alumni.tecnun.es fplanes@ceit.es, or John.Beasley@brunel.ac.uk
Supplementary Information: Supplementary data are available online.

1 INTRODUCTION

The New York City Subway [1] is the largest rapid transit system in the world by number of stations. Overall it has 468 stations, divided into 24 different lines, with a track length of 1060 kilometres. Out of curiosity, the average weekday subway ridership is 4.7 million, about 1.449 billion a year.

The Subway Challenge is a competition where participants must pass through all the New York Subway stations in the shortest time possible. The fastest time to travel to all New York City Subway stations is 21 hr 49 min 35 sec and was recently achieved by Matthew Ahn (USA) on 16 January 2015. As a matter of fact, this record time was confirmed by the Guinness Book of Records [2].

Officially, there are three Classes of Competition: Class A (Covering all lines), Class B (Touching all stations) and Class C (Passing all stations). In this work we will focus on the Class B competition. The goal of the competition is to set a record minimum time by planning a route through the New York City Transit System and making a timed run over that route [3].

In order to find the optimal path for this problem, a graph procedure is presented. Basically, each node stands for a station, and each edge represents the connection between two stations in one direction. The cost of activating each arc is related to the Euclidean distance between the stations, that is then converted to a time measure by way of the mean velocity of the train.

What we are looking for is the optimal path that passes through every node of the graph at least once. Initially, we tried to solve this graph by means of Integer Linear Programming (ILP) formulation, as done in problems like Traveling Salesman Problem (TSP). In this sense, the constraints applied were not enough to ensure a good solution. In fact, we had to deal with a systematic subtour generation, because if every station has to be visited at least once, then there are certain stations that must be visited more than once in order to achieve that principle. This is mainly caused by the sparse nature of the graph. For example, in a set of stations with a final branch off station, we should visit those stations at least twice, which in fact is a subtour. Applying the subtour constraints to this graph would lead to an infeasible problem, but not implementing them would favour the generation of subtours.

There have been related attempts to solve the Subway Challenge. Recently, Wolfgang A. Welz, in the 6th chapter of his thesis "Robot Tour Planning with High Determination Costs" [4], faced this problem from a similar perspective. He creates a "subway graph", based on three steps: First of all, for each subway line the stations are duplicated. The second step is to generate directed edges to represent the two possible directions of travelling (forward and backward). And finally, include the changing arcs between the nodes of each station. With this called "subway graph", he implements three formulations. A TSP transformation, a subtour ILP formulation and a flow-based ILP formulation. The subtour ILP formulation, which is the one we are interested in, is solved with a specific iterative algorithm, varying a subset of stations C, which initially is a feasible solution. The subtour constraints are applied only for this subset C, meaning that it is dynamic formulation.

The solution we propose relies on the transformation of the original graph into its directed line graph. Working with the directed line graph allows us to use an easy ILP formulation to solve the Subway Challenge. The creation of the directed line graph is exhaustively explained in the chapter of "Methods", but it can be summed up on two simple steps.

First, for each station, we create as many nodes as leaving arcs has that station. Each node of the line graph represents an arc of the former graph. Since each node belongs to a station and also denotes a direction to another station, we must join each node with all the nodes of the station it is pointing to. This joining corresponds to the second step. With this procedure, the changing stations, and also the order, the source and the target of the optimal path are obtained.

There are some differences between our method and the one presented by Welz. First of all, what we want is to find the optimum path for the challenge, while Welz obtains a tour. If we attend to the rules of the challenge, it is obvious that a path is much more efficient than a tour. Secondly, we implement a different integer linear formulation on a directed line graph, which has not been used yet. Our model is them optimized using the Branch-and-Bound algorithm in CPLEX [5].

In the next section we formally introduce the directed line graph creation and mathematical model for the optimization. Then we will explain how this model was applied to the New York Subway Challenge and the results obtained will be presented too. Finally, we will discuss the results and compare them with the current record.

2 METHODS

Given a particular subway system, we aim to find a route visiting all the stations with the minimum cost. A subway system can be modelled as a finite directed graph $G = \{V_G, E_G\}$ (Figure 1A). The set of nodes $V_G = \{1,2 \dots, n\}$ represents the different stations in the system. Two nodes are selected as the start and the end of our route. The set of arcs are defined as $E_G = \{e_G \mid e_G = (i,j), \forall\, i,j \in V_G,\ d_{ij} = 1\}$, where $d_{ij} = 1$ if stations i and j are directly connected together, 0 otherwise. As subway lines are typically bidirectional we shall assume that $d_{ij} = d_{ji}$. Each arc $e_G = (i,j)$ involves a traversal cost if it is used equal to c_{ij}, which is not necessarily symmetric.

Technically, the solution to our problem is a walk and not a simple path, since, as shown in Figure 1B, we may need to visit a station (node) more than once. For example in Figure 1B if the start node is node 1 and the end node is node 5 using arcs $x_{12}, x_{23}, x_{32}, x_{24}$ and x_{45} will enable us to visit all of the nodes, but node 2 will be visited twice. The logic behind our approach is that the solution to the subway problem typically will require cycles of order 2, as in Figure 1B with the cycle of order 2 visiting node 3.

With this in mind, the graph G is transformed using the line graph operation. We denote $L = \{V_L, E_L\}$ the directed line graph of G. L is obtained by associating a node with each edge of the root graph G and connecting two nodes if and only if their corresponding edges in G form a path of length 2. Specifically, $V_L = E_G$, while $E_L = \{e_L \mid e_L = (u,v),\ \forall\, u,v \in V_L, D_{uv} = 1\}$. We define D_{uv}=1 if, assuming that $u = (i,j),\ v = (j,k),\ i,j,k \in V_G$, then $d_{ij} = 1$ and $d_{jk} = 1$ so there is a path from I to k through j. Each arc $e_L = (u,v)$ involves an activation cost equal to $C_{uv} = c_{ij} + a_{uv}$, where a_{uv} denotes the transition time needed (when applicable) for changing subway lines at the intermediate station j when using arcs (i,j) and (j,k). Figure 1C represents the directed line graph for the root graph in Figure 1A.

As, in principle, source and target nodes are not fixed, we add two artificial nodes S and T to graph L. We define for all nodes in L an entering arc from S and a leaving arc to T (with these arcs being incorporated into D in a natural way) The cost of these auxiliary arcs is 0. With this, as detailed below, we give the possibility to the model to select the start and end node without adding a significant number of additional variables and constraints. Figure 1D represents the final graph L used in this work.

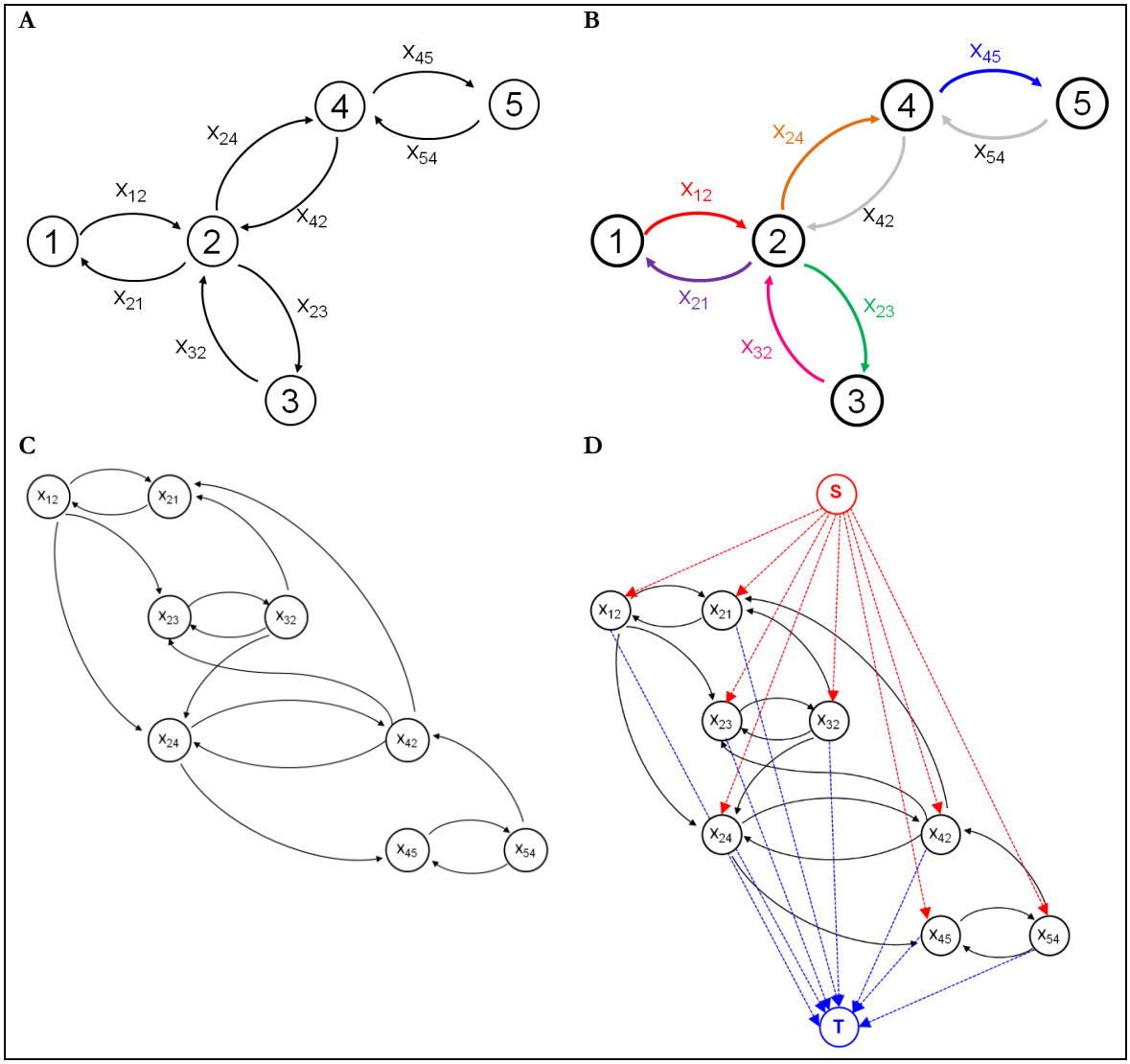

Figure 1: Graph model used for an example subway system

Having introduced our graph model for the subway system, we have two types of variables in our optimization problem. First, we define x_{uv} , which takes the value 1 if the arc $e_L = (u, v) \in E_L$ is active, 0 otherwise. Second, we have auxiliary integer variables w_u to avoid cyclic solutions, which indicate the order in which each node $u \in V_L$ is visited in our solution.

The objective function of the model is:

$$\text{minimize} \sum_{u \in V_L} \sum_{v \in V_L, D_{uv}=1} C_{uv} x_{uv} \qquad (1)$$

Our solution is technically a path in *L* from node S to node T satisfying certain constraints. We introduce below these constraints.

An important constraint in our formulation is that we need to visit all the nodes in the root graph G, but not necessarily in the directed line graph L. For that, we denote $A_i = \{e_G \mid e_G \in E_G, e_G = (i, j),\ \forall j \in V, d_{ij} = 1\}$ the subset of arcs of G that leaves node $i \in V_G$. These subset of arcs in G are nodes in L.

$$\sum_{\forall u \in A_i} \left[\sum_{\substack{v \in V_L, \\ D_{uv}=1}} x_{uv} \right] \geq 1 \ ; \ \forall i \in V_G \qquad (2)$$

We also make sure that, in every node u of the graph L, the activated arcs entering and the activated arcs leaving are equal.

$$\sum_{\substack{v \in V_L, \\ D_{uv}=1}} x_{uv} - \sum_{\substack{v \in V_L, \\ D_{vu}=1}} x_{vu} = 0 \ ; \ \forall u \in V_L | u \neq S, T \qquad (3)$$

In the source and target node, S and T, we force one particular departure and arrival, respectively:

$$\sum_{v \in V_L} x_{Sv} = 1 \ ; \qquad \sum_{u \in V_L} x_{uT} = 1 \qquad (4)$$

We may fix the source and/or the target node if required. For example, if we are interested in selecting the node $i \in V_G$ as source node, we only need to add the constraint below. Fixing the target node is similar.

$$\sum_{\forall u \in A_i} x_{Su} = 1 \qquad (5)$$

Finally, we need to avoid the creation of subtours. We use standard constraints for that:

$$w_u - w_v + (|E_G| - 1) \cdot x_{uv} \leq (|E_G| - 2) \quad \forall u, v \in V_L \qquad (6)$$

where $|E_G|$ represents the cardinality of E_G, i.e. the number of edges in graph *G*.

3 RESULTS

Applying this model to the New York Subway System requires the original graph to be assembled. The initial data was obtained from the MTA (Metropolitan Transportation Authority) webpage, specifically from the "Developer Resources" section [6].

There a GTFS package [7] was downloaded, which is intended for phone apps development. For the graph creation we focused in the text file "stops.txt", where among other data, the index of terminals and their latitude and longitude is specified.

Making use of this data, the Mercator projection was used to plot the latitude and longitude of each terminal into a map (Figure 2).

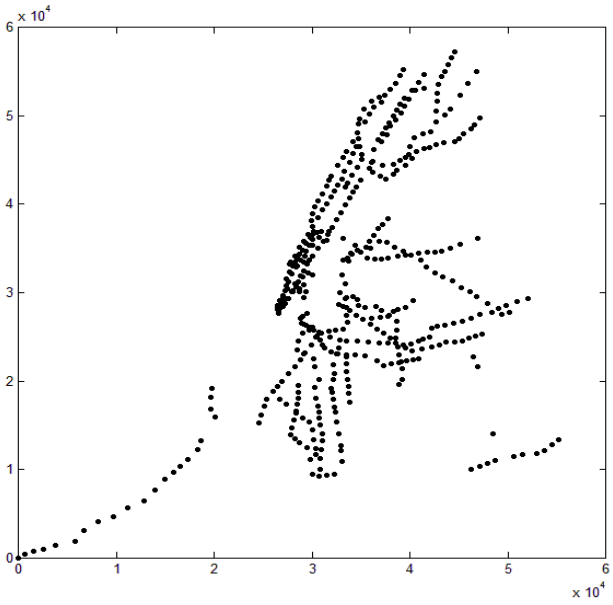

Figure 2: Position of the NY subway terminals

Having the nodes of the graph, we could not find any information to generate the edges of the graph. To do so, we had to introduce manually the lines that belonged to each terminal. So as to get the edges, we worked with the set of terminals of each line, creating the complete graph of those nodes. Then, we applied the minimum spanning tree (MST) to the complete graphs of each line, and we obtained the real edges of the graph, due to the straight configuration of the lines. Finally, we had to duplicate all the edges, in order to get the return path for the trains. The figure 3 illustrates the final graph G, with each line represented by a different colour.

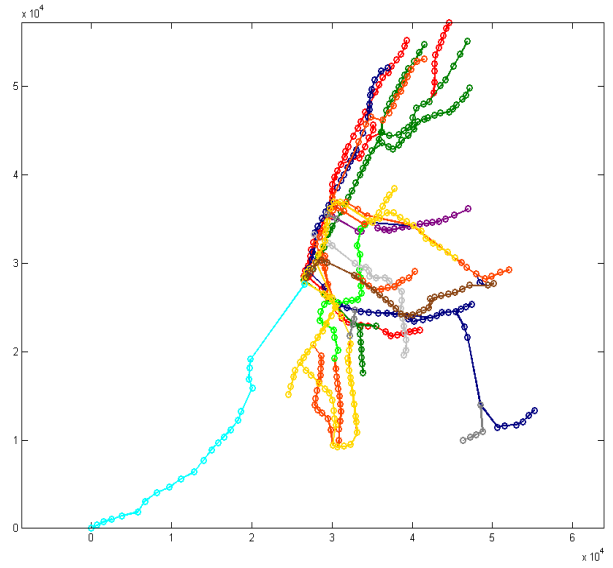

Figure 3: Creation of the subway network, using the MST.

In order to minimize the computational cost of the resolution of the problem, some simplifications were implemented to the subway system model:

- There are some terminals that are so large, that they gather different lines and are located in different positions. In our model we consider two separate terminals.

- There are some express lines that do not stop in all the terminals at rush hours. In our case we avoid these special lines.
- We do not consider the walkway between terminals.
- There is an island in the system, Staten Island, and we replace the actual ferry with a fictitious line.

The weights of the edges are actually the Euclidean distances between terminals. However, what we look for is the fastest path, so length is turned into time using the mean velocity of each line.

The mean velocities (in miles per hour) of the lines is obtained from the figure 4 [8].

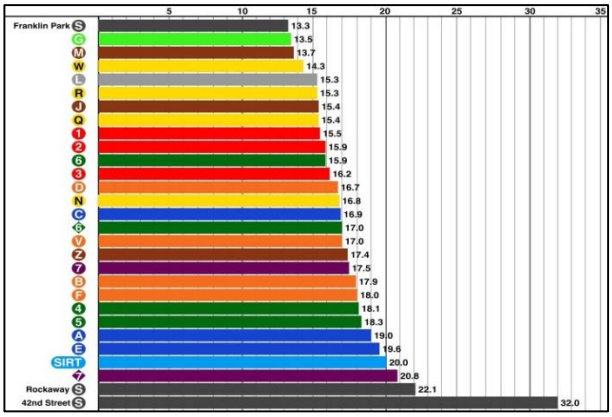

Figure 4: Mean speed of the lines, in miles per hour.

Selecting where to start is a critical decision. As it has been explained in the methods section, the selection of the source and the target node is included in the model itself. From the optimality point of view, it makes sense to start and to finish in a branch off terminal. Otherwise more terminals should be visited to finish in a intermediate station.

In the directed line graph L of New York there are 1078 nodes, 24 of them branch off terminals. Among these 24, the outermost ones should be the source and the target.

To support the source and target election by the model, the shortest path tree (SPT) between these terminals has been calculated. The path that takes more time should contain the optimal source and target. In the following table, a summary of 6 of these paths has been included, considering the outermost four terminals in the map (Figure 5).

Start Terminal	Destination Terminal	Time (min)
A) Tottenville	B) Wakefield 241 St	160
A) Tottenville	C) Jamaica 179 St	150
A) Tottenville	D) Far Rockaway	170
B) Wakefield 241 St	C) Jamaica 179 St	113
B) Wakefield 241 St	D) Far Rockaway	155
C) Jamaica 179 St	D) Far Rockaway	91

The longest SPT is between Tottenville and Far Rockaway, which are in fact the two nodes that our model selects automatically.

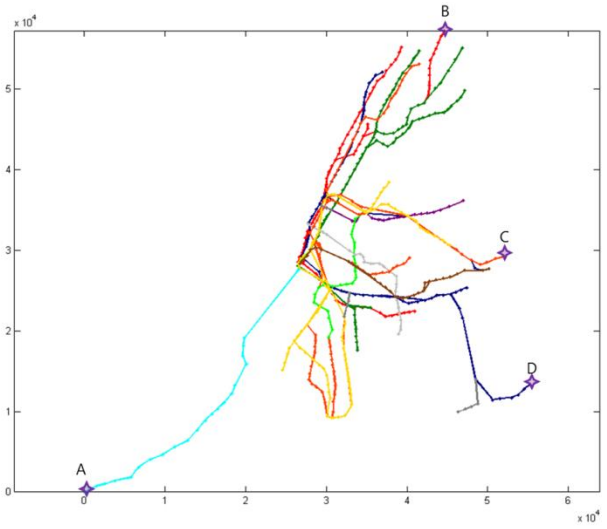

Figure 5: Candidates for the source and target terminals.

The model has been solved using CPLEX 12.5 on a Intel i5-6300U CPU at 2.4 Ghz. The optimal solution for the New York Subway Challenge is shown in the appendix A, where a table with the final path is described. Along with the start and destination terminals, the line used and the time spent in each journey has been included.

The total riding time has been of 1273 minutes (21 hours and 13 minutes), which means that the current Guinness Record is beaten. Obviously, even if is the time travel estimation is good, is nearly impossible to build an exact and precise model such as the real system subway, that for example has a well defined timetable. Therefore the comparison between our solution time and the record is purely anecdotal.

As a kind of curiosity, in this solution there are some terminals that are crossed multiple times. As a matter of fact, the most visited stations are the "Broadway Junction", "Atlantic Av-Barclays Ctr", and "14 St-Union Sq.", that are visited four times. These are the network´s most relavant stations, due to their high connectivity.

4 CONCLUSIONS

The computational cost of the model is high, due to the size of the subway and moreover due to the number of lines, which expands the number of possible paths. In fact, in the first stage of this work, when the graph was created and the model was implemented, only few of the lines were included in the model, in order to assure the validity of the formulation.

With reference to the computational cost, even if the model is hard to solve, it is powerful enough to solve other problems, such as getting the shortest path between two given terminals. Only by removing one constraint, in particular the equation 2, it is possible to transform the subway challenge model in the shortest path problem.

In relation with the mentioned terminals in the "Results" section, it would be interesting to study the minimum cut set of the network, in order to prevent possible failures or maintenance services of certain lines or paths in the subway network.

For future research projects in this matter, it should be noted that one of the most important parameters in the model is the time required to change from one line to another, that is, the transition time a_{uv}. Its value affects notoriously the solution path and of course, the time spent.

In order to take this fact fully into account, a slight modification was implemented in the model. The actual model is not able to detect all the line changes in the solution. Therefore, instead of simply using the directed line graph, more information had to be included in the x_{uv} variable, in order to know which line was using in that path. Hence, the x_{uvp} variable was necessary, with p representing the line used. This little change increased notoriously the size of the graph, making so hard to solve with CPLEX and the present computer resources. As result, only a reduced size of the subway system was solved, involving 6 of the 25 lines.

In this sphere of the transition time, two lines of research are proposed. First of all, the study of a metaheuristic method to facilitate the resolution of the model. Second, due to the special characteristics of the subway system, a method to reduce the number of nodes would be appropriate, making easier the resolution as well. Having included those changes, the influence of the transition time could be calculated.

Additionally, a more complex model of the system could be carried out, by including the timetables of the trains for each terminal. In this way a much more complete and precise model of the network could really help beating the New York Subway Challenge.

To sum it up, this work has been developed in the frame of linear optimization, using only CPLEX. The mathematical formulation exposed in the "Methods" section is usually applied in many graph-based problems, such as the famous "Travelling Salesman" problem. However, with the implementation of the directed line graph, a different perspective of this mathematical formulation has risen up.

5 REFERENCES

[1] New York City Subway - Wikipedia

https://en.wikipedia.org/wiki/New_York_City_Subway

[2] New York Subway Challenge Record

http://www.guinnessworldrecords.com/world-records/travelling-new-york-city-subway-in-shortest-time-(underground)

[3] Subway Challenge rules

http://www.rapidtransitchallenge.com/rules.htm

[4] "Robot Tour Planning with High Determination Costs"– Wolfgang A. Welz

https://opus4.kobv.de/opus4-tuberlin/frontdoor/index/index/docId/6044

[5] IBM ILOG CPLEX

http://www.ibm.com/software/commerce/optimization/cplex-optimizer/

[6] MTA developers webpage

http://web.mta.info/developers/

[7] Graph data downloaded package

(http://web.mta.info/developers/data/nyct/subway/google_transit.zip).

[8] Lines mean velocity data

(http://greatergreaterwashington.org/post/5183/average-schedule-speed-how-does-metro-compare/).

APPENDIX A

- Optimum Path Solution Table

Start_Terminal	Line	Destination_Terminal	Time(min)
Tottenville'	SIR	South Ferry Loop'	62
South Ferry Loop'	1	Chambers St'	67
Chambers St'	2	Wall St'	69
Wall St'	2	Chambers St'	72
Chambers St'	1	Times Sq - 42 St'	85
Times Sq - 42 St'	7	Grand Central - 42 St'	87
Grand Central - 42 St'	4	86 St'	95
86 St'	6	125 St'	103
125 St'	4	138 St - Grand Concourse'	105
138 St - Grand Concourse'	5	149 St - Grand Concourse'	107
149 St - Grand Concourse'	2	Central Park North (110 St)'	115
Central Park North (110 St)'	2	135 St'	120
135 St'	3	Harlem - 148 St'	123
Harlem - 148 St'	3	135 St'	125
135 St'	2	E 180 St'	142
E 180 St'	5	Eastchester - Dyre Av'	156
Eastchester - Dyre Av'	5	E 180 St'	170
E 180 St'	2	Wakefield - 241 St'	188
Wakefield - 241 St'	2	149 St - Grand Concourse'	220
149 St - Grand Concourse'	5	138 St - Grand Concourse'	221
138 St - Grand Concourse'	4	Woodlawn'	242
Woodlawn'	4	125 St'	265
125 St'	6	Pelham Bay Park'	295
Pelham Bay Park'	6	Grand Central - 42 St'	340
Grand Central - 42 St'	7	Court Sq'	348
Court Sq'	G	21 St'	349
21 St'	G	Court Sq'	351
Court Sq'	7	Queensboro Plaza'	352
Queensboro Plaza'	N	Astoria - Ditmars Blvd'	361
Astoria - Ditmars Blvd'	N	Queensboro Plaza'	370
Queensboro Plaza'	7	Flushing - Main St'	393
Flushing - Main St'	7	Queensboro Plaza'	415
Queensboro Plaza'	N	Times Sq - 42 St'	427
Times Sq - 42 St'	7	Grand Central - 42 St'	429
Grand Central - 42 St'	6	14 St - Union Sq'	435
14 St - Union Sq'	L	Broadway Jct'	464
Broadway Jct'	C	Hoyt - Schermerhorn Sts'	480
Hoyt - Schermerhorn Sts'	G	Greenpoint Av'	502
Greenpoint Av'	G	Church Av'	539
Church Av'	F	W 8 St - NY Aquarium'	556
W 8 St - NY Aquarium'	Q	Prospect Park'	582
Prospect Park'	W	Park Pl'	585
Park Pl'	W	Prospect Park'	588
Prospect Park'	Q	Coney Island - Stillwell Av'	615
Coney Island - Stillwell Av'	N	59 St'	635
59 St'	R	Bay Ridge - 95 St'	643
Bay Ridge - 95 St'	R	36 St'	656
36 St'	D	Bay 50 St'	676
Bay 50 St'	D	36 St'	696
36 St'	R	14 St - Union Sq'	725
14 St - Union Sq'	6	Brooklyn Bridge - City Hall'	732
Brooklyn Bridge - City Hall'	4	Bowling Green'	735
Bowling Green'	4	14 St - Union Sq'	743
14 St - Union Sq'	L	8 Av'	747
8 Av'	L	14 St - Union Sq'	750
14 St - Union Sq'	N	Times Sq - 42 St'	755
Times Sq - 42 St'	1	Van Cortlandt Park - 242 St'	799
Van Cortlandt Park - 242 St'	1	168 St - Washington Hts'	816
168 St - Washington Hts'	A	Inwood - 207 St'	824
Inwood - 207 St'	A	168 St - Washington Hts'	832
168 St - Washington Hts'	C	145 St'	837
145 St'	B	Bedford Park Blvd'	855
Bedford Park Blvd'	D	Norwood - 205 St'	856
Norwood - 205 St'	D	145 St'	876
145 St'	A	125 St'	879
125 St'	B	7 Av'	893
7 Av'	E	W 4 St'	901
W 4 St'	B	Grand St'	905
Grand St'	B	Broadway-Lafayette St'	907
Broadway-Lafayette St'	F	Essex St'	909
Essex St'	J	Broad St'	916
Broad St'	J	Essex St'	922
Essex St'	F	Jay St - MetroTech'	929
Jay St - MetroTech'	R	DeKalb Av'	930
DeKalb Av'	B	7 Av'	934
7 Av'	B	Atlantic Av - Barclays Ctr'	936
Atlantic Av - Barclays Ctr'	2	Flatbush Av - Brooklyn College'	951
Flatbush Av - Brooklyn College'	2	Franklin Av'	961
Franklin Av'	3	New Lots Av'	977
New Lots Av'	3	Clark St'	1004
Clark St'	3	Atlantic Av - Barclays Ctr'	1009
Atlantic Av - Barclays Ctr'	B	DeKalb Av'	1010
DeKalb Av'	R	Jay St - MetroTech'	1012
Jay St - MetroTech'	A	Canal St'	1020
Canal St'	E	World Trade Center'	1022
World Trade Center'	E	W 4 St'	1027
W 4 St'	F	21 St - Queensbridge'	1043
21 St - Queensbridge'	F	47-50 Sts - Rockefeller Ctr'	1052
47-50 Sts - Rockefeller Ctr'	M	Forest Hills - 71 Av'	1086
Forest Hills - 71 Av'	E	Briarwood - Van Wyck Blvd'	1091
Briarwood - Van Wyck Blvd'	F	Jamaica - 179 St'	1098
Jamaica - 179 St'	F	Briarwood - Van Wyck Blvd'	1106
Briarwood - Van Wyck Blvd'	E	Jamaica Center - Parsons/Archer'	1110
Jamaica Center - Parsons/Archer'	E	Sutphin Blvd - Archer Av - JFK Airport	1111
Sutphin Blvd - Archer Av - JFK Airport'	J	Broadway Jct'	1134
Broadway Jct'	L	Canarsie - Rockaway Pkwy'	1144
Canarsie - Rockaway Pkwy'	L	Broadway Jct'	1153
Broadway Jct'	J	Marcy Av'	1167
Marcy Av'	J	Myrtle Av'	1172
Myrtle Av'	M	Middle Village - Metropolitan Av'	1185
Middle Village - Metropolitan Av'	M	Myrtle Av'	1198
Myrtle Av'	J	Broadway Jct'	1206
Broadway Jct'	C	Euclid Av'	1213
Euclid Av'	A	Ozone Park - Lefferts Blvd'	1222
Ozone Park - Lefferts Blvd'	A	Broad Channel'	1244
Broad Channel'	S	Rockaway Park - Beach 116 St'	1251
Rockaway Park - Beach 116 St'	S	Broad Channel'	1259
Broad Channel'	A	Far Rockaway - Mott Av'	1273