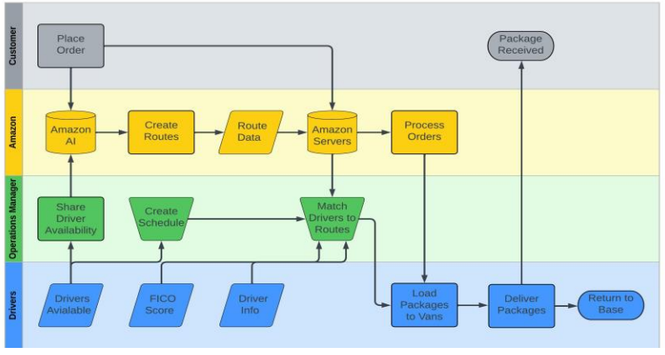
Distributing workload of daily driver assignment using linear programming with historical data

Andrew Freeman (Team Leader), Sarah Wilson, Katy Emerson, Nicolas Alcoreza, Enmanuel Jean Paul De La Gala Gomez
Founder & President – Latanyua T. Robinson

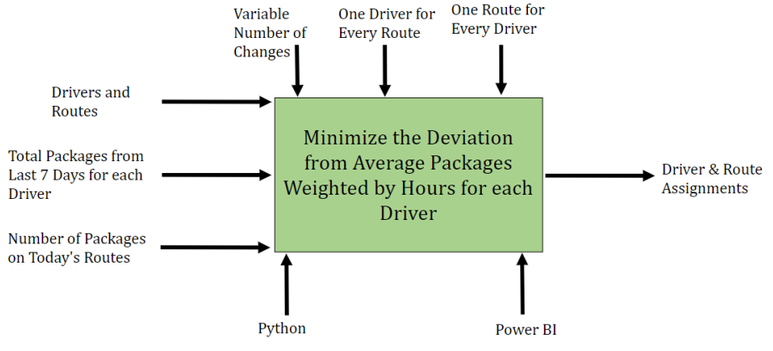
Latrobe – Last-Mile Delivery

Latrobe LLC, is a multi-dimensional firm based in Memphis, TN with divisions in manufacturing consulting, light assembly, and packaging and shipping operations. The division that focuses on shipping acts as a third-party Amazon Delivery Service Partner (DSP) that distributes last-mile Amazon packages to their customers. Latrobe is determined to improve their current work scheduling and the route to driver assignment with the use of a decision support tool. The tool provides the Operations Manager the optimal changes to improve the daily driver schedule once they receive it from Amazon to reduce fatigue.



Balancing Driver Workload

The decision support tool is based on an optimization model whose objective function is to minimize deviation from average packages weighted by hours for each driver. This optimization model is run by Python and integrated using Power BI. This integrated system would be run every morning by Latrobe's Operations Manager to improve driver assignments. The assumption is that it would reduce physical and mental fatigue as well as risk for injury in the process.



Optimization Model For Work Scheduling

The parameters define the number of packages associated with each driver during the given number of days. The span of days to be considered will include the current day as well as the last seven days. A driver will have a parameter that defines the cumulative packages and hours in the past seven days and is updated daily. Similarly, there are parameters that define the hours and packages of a route. Each route must have one driver. Each driver must have one route. The objective function is to minimize the sum of the drivers' deviation from the average number of packages weighted by hours across all drivers. This approach assumes that creating a more balanced workload reduces driver fatigue. The model also assumes that the number of packages assigned to each driver is one of the main causes of fatigue due to a driver needing to carrying more packages, make more stops, and make more trips from the van.

Parameters:

d = the index for the driver

$|D|$ the set of drivers

j = the index for the route

$|J|$ = the set of routes

P_d = the number of historical packages for driver d

p_j = the number of packages on route j

H_d = the number of historical hours for driver d

h_j = the number of hours on route j

$Y_{dj} = \begin{cases} 0 \text{ if } d = j \text{ because this is a default assignment} \\ 1 \text{ otherwise} \end{cases}$

Y_{dj} is binary $\forall d \; \forall j$

N = the penalty value

Decision Variable:

$X_{dj} = \begin{cases} 1 \text{ if driver } d \text{ is assigned to route } j \\ 0 \text{ otherwise} \end{cases}$

X_{dj} is binary $\forall d \; \forall j$

Constraints:

$\sum_d^D X_{dj} = 1 \; \forall j$ only one driver may be assigned to each route

$\sum_j^J X_{dj} = 1 \; \forall d$ only one route may be assigned to each driver

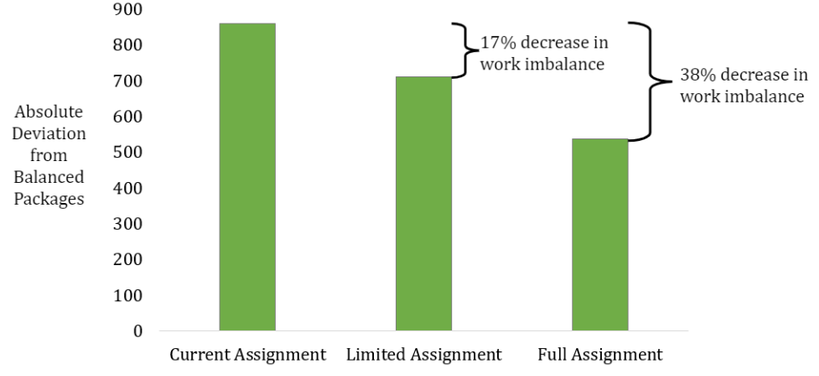
Objective Function:

$$W_{avg} = \left(\sum_{d \in D}^{D} P_d + \sum_{j \in J}^{J} p_j \right) / \left(\sum_{d \in D}^{D} H_d + \sum_{j \in J}^{J} h_j \right)$$

$$C_{dj} = \left| \frac{P_d + p_j}{H_d + h_j} - W_{avg} \right| * (H_d + h_j)$$

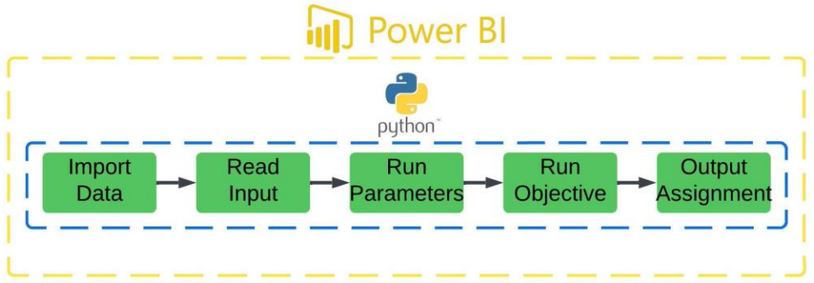
$$Minimize \sum_{j}^{J} \sum_{d}^{D} C_{dj} * X_{dj} + Y_{dj} * N$$

Applying our full set of changes to the initial assignment reduces work imbalance by an average of 38% compared to baseline. The same number of packages are delivered by the same number of drivers. What changes in our assignment is that work is more evenly divided between drivers while accounting for differences in full-time and part-time drivers. This should reduce fatigue, make work schedules fairer, has no direct costs, and may even improve productivity.



Python and Power BI Implementation

This flow chart demonstrates how the Python Programming Structures interact to generate driver assignments. First, the data is pre-processed and is arranged where the relevant information is input into the parameters. This information includes historic driver data on packages for the last seven days and route data on packages for the current day. Next, we read this data into Python as a CSV file. Before running the objective function in Python, we must run the parameters using Pandas. Once we successfully ran the parameters, the mathematical optimization computations are made using the functions contained in the SciPy Library to establish assignments for the drivers. Finally, the Python script is run and displayed in Power BI.



Power BI Output Visualizations

Power BI is used to run the Python script and display the output from the optimization model. The main visual that is displayed in Power BI is a table including the default route to driver assignment, the optimal route to driver assignment, along with additional information on the routes themselves. These visuals help the Operations Manager understand what changes were made and why those changes were made. These Power BI reports are interactive, allowing the user to select a driver or a route and view information specific to that assignment.

Daily Driver Assignment

Route	Route Code	Packages on Route	Commercial Packages	Current Driver Assignment	Current Driver Value	Route Code	Limited Driver Assignment	Limited Driver Value
1	CX0013500	160	20	Driver 1	29.95	CX0013500	Driver 1	29.95
2	CX0013900	285	6	Driver 2	34.39	CX0013900	Driver 2	34.39
3	CX0014000	283	15	Driver 3	36.28	CX0014000	Driver 3	36.28
4	CX0014100	299	31	Driver 4	32.97	CX0014100	Driver 4	32.97
5	CX0014200	262	3	Driver 5	32.87	CX0014200	Driver 24	30.98
6	CX0014300	285	17	Driver 6	34.13	CX0014300	Driver 16	32.11
7	CX0014400	277	50	Driver 7	33.97	CX0014400	Driver 7	33.97
8	CX0014500	305	65	Driver 8	32.13	CX0014500	Driver 8	32.13
9	CX0014600	301	139	Driver 9	33.04	CX0014600	Driver 9	33.04
10	CX0014700	266	10	Driver 10	33.03	CX0014700	Driver 10	33.03
11	CX0014800	275	3	Driver 11	33.09	CX0014800	Driver 11	33.09
12	CX0014900	180	1	Driver 12	30.87	CX0014900	Driver 12	30.87
13	CX0015000	300	47	Driver 13	34.74	CX0015000	Driver 13	34.74
14	CX0015100	268	26	Driver 14	32.72	CX0015100	Driver 14	32.72
15	CX0015200	260	29	Driver 15	30.60	CX0015200	Driver 15	30.60
16	CX0015300	240	43	Driver 16	30.02	CX0015300	Driver 6	31.87
17	CX0015400	254	4	Driver 17	31.17	CX0015400	Driver 17	31.17
18	CX0015500	267	11	Driver 18	33.19	CX0015500	Driver 18	33.19
19	CX0015600	269	39	Driver 19	32.54	CX0015600	Driver 19	32.54
20	CX0015700	261	68	Driver 20	31.82	CX0015700	Driver 20	31.82
21	CX0015800	266	1	Driver 21	33.89	CX0015800	Driver 21	33.89
22	CX0015900	283	46	Driver 22	32.75	CX0015900	Driver 22	32.75
23	CX0016000	284	55	Driver 23	31.84	CX0016000	Driver 23	31.84
24	CX0016100	282	16	Driver 24	30.84	CX0016100	Driver 5	32.77