Logistics of Humanitarian Operations Research (World Foood Programm)

User Manual

Milena Akemann, Theresa Baader, Sabrina Primus,
Léo Simpson, Christophe Vetter

October 14, 2020

Contents

1 Notice

This document contains extracts of our technical documentation. It is fully enough for understanding how to use the decision tool but if you would like to get more detailed information about the algorithm itself, feel free to have a look at our technical documentation on GitHub.
At first we're going to give a short description of the problem, after that you can find the manual how to use the tool.
Have fun! :)

2 Problem description

Given instances of a supply chain and a time horizon T, we are designing a decision tool to output a delivery schedule for discrete time steps $t = 1, ..., T$ which manages the food supply of the given set of schools using the given warehouses and given vehicle fleets. The overall goal is to construct routes for the whole planning period with minimal costs, i.e. to minimize the sum of transportation and storage costs while making sure that enough food is available at the schools and some further conditions are satisfied.

In our problem setting we consider a supply chain of three levels. On the first level we have a single central warehouse, where all the food is stored. The central warehouse itself is not part of the optimization since we assume that it has unlimited inventory but no (variable) inventory holding costs. From the central warehouse the food is delivered to many small warehouses which build the second level. Each of those warehouses is then again connected to plenty of schools -the third level- which need the food for the children. For the transportation every warehouse has its own vehicle fleet of heterogenous vehicles and those vehicle fleets can be different between any two warehouses regarding the amount and the types of vehicles available.

Designing smart routes between the facilities and assigning vehicles to these routes with minimal costs to deliver a certain amount of food to the schools S describes the classical Vehicle Routing Problem (VRP). In our case however this only corresponds to the routing component of our problem.

The other component includes the management of the inventory at the schools and the warehouses. That means in contrary to the VRP where one has given order quantities at the facilities which have to be delivered, we are only given upper and lower bounds for the inventory at the facilities which have to be satisfied at every time step t and consumption rates per time step depending on the amount of pupils at the schools. The upper bounds correspond to the limited storage capacities at the warehouses and schools while the lower bounds shall assure that the schools are not running out of food and are hence able to serve meals to all their pupils at every time step. Hence we do not only decide on which truck from which warehouse visits which school, but we also decide on

the frequency to go there (at which timesteps) and on how much to deliver to the schools for every single visit by finding a balance between transportation costs and inventory holding costs. This gives us more flexibility for optimization on the one hand but also increases the complexity of the problem on the other hand.

Regarding the costs, the total costs of a proposed schedule include the transportation cost, which depend on the total amount of kilometres driven in the whole period, and the inventory holding costs in schools and warehouses. For the warehouses we are only given fixed costs for each warehouse but no variable costs depending on the amount of food stored there. However we do not consider the choice of which warehouses to use as part of our optimization problem, hence the fixed costs just correspond to a constant that is added to the costs of every solution. Therefore the storage costs in the warehouses do not affect the optimum and are therefore not part of the objective function. In the schools the situation is slightly different: Storing food in schools is actually free but since these are no proper storage facilities we assume a certain percentage of food stored in schools to get lost. Since every meal that gets lost can't be fed to a child we try to avoid this from happening by not storing too much food in schools. In order to be able to put these opportunity costs or loss costs into a mathematical expression we consider them like variable inventory holding costs. This way the more you store in a school the more expensive it gets especially compared to the non-existing variable costs in the small warehouses. Now it becomes more clear what finding a balance between the different types of costs actually mean. Depending on the real values, is it better to drive to a school more often delivering only small quantities which leads to higher transportation costs, or to visit it only as often as necessary and putting in a higher quantity each time which increases the storage costs?

The combination of the two components routing and inventory management with the overall goal of minimizing the occuring costs makes our problem a so called Inventory Routing Problem (IRP).

The described setting can be modeled on a directed graph as shown in Figure 1 where the nodes are given by all the facilities and the edges correspond to the routes between them. Or more formally, let C denote the central warehouse, W the set of small warehouses and S the set of schools. Then we have a graph $G = (V, E)$, where

$$V = C \cup W \cup S, \tag{1}$$

$$E = \{(v, w), (w, v) \text{ s.th. } v \in C \cup S,\ w \in W \cup S\} = (V \times V) \setminus (W \times W). \tag{2}$$

This definition of the edge set E arises as we could also serve schools directly from the central warehouse if that is cheaper but we do not want trips between small warehouses.

The height of the bars in Figure 1 represents the capacity of schools and warehouses, their current stock level is shown in green (almost maximal stock), blue (still sufficient amount of food stored) or red (an early delivery is necessary).

There are some constraints we have to consider when dealing with our problem. One can differentiate between storage constraints and routing constraints.

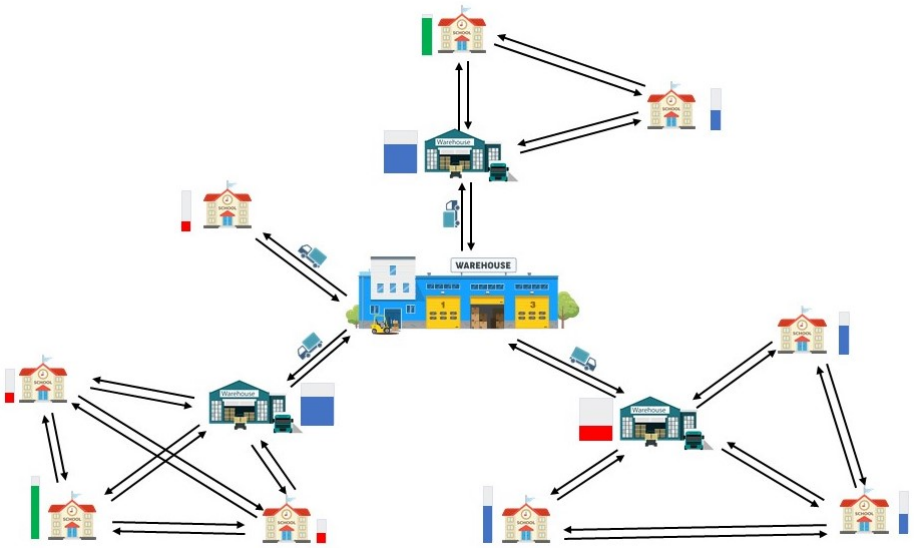


Figure 1: Problem setting

Constraints on Storage:

Lower bound on inventory in schools. The most important constraint is that we do not want children to receive no food at any time since the food served at the schools might be the only proper meal they get on a day. Hence, we need to make sure that at every time t and at every school m there is enough food to serve all pupils. This can be done by setting the lower bound L_m of school m greater than or equal to its consumption rate and expect the inventory I_m^t of the schools to be always greater or equal than the lower bound.

Upper bound on inventory in schools. Every school m has only a limited storage capacity U_m which should not be exceeded.

Lower bound on inventory in warehouses. To ensure the supply of the schools, there has to be enough food stored in the warehouses. For every warehouse n we are given a lower bound $L_n \geq 0$ on the inventory I_n^t.

Upper bound on inventory in warehouses. Clearly every warehouse only has a certain size U_n and we can not store more food at a warehouse than the amount that fits into it. The only special case is the central warehouse for which we assume unlimited capacity.

Constraints on the Routing:

Vehicle capacities. Every vehicle has an individual capacity which shall not be exceeded. Hence each vehicle can only deliver at most as much food as fits into it on one route

Roundtrips. We want every constructed tour to end at the same point where it started.

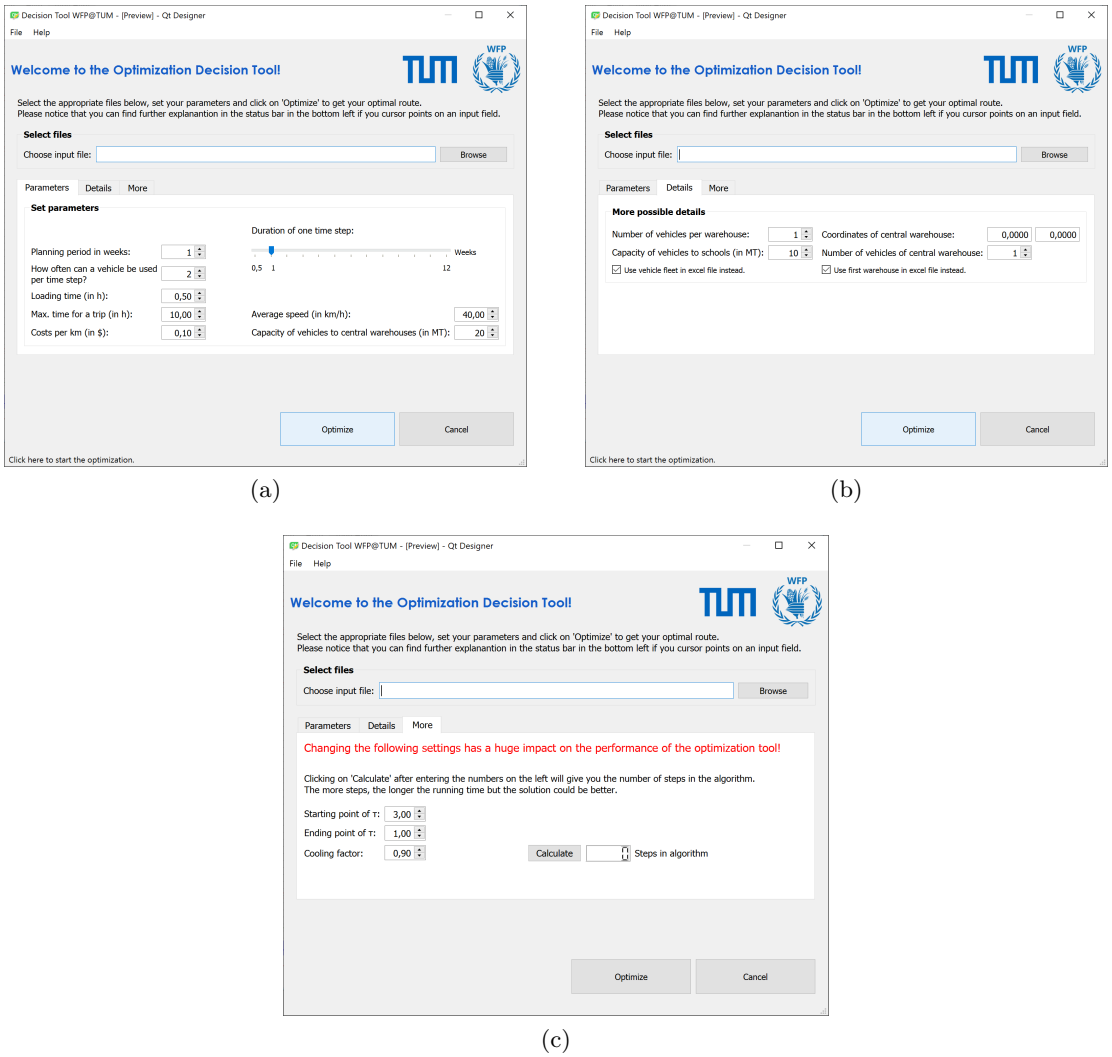
Length of a route. Every tour shall not take longer than 10 hours (more general $\mathcal{T}_{max}$) including the loading and unloading time l at the warehouses and the schools respectively. For these events an average time of half an hour (0.5 hours) is estimated.

Since there exist a wide range of possible settings and underlying assumptions for IRPs, many authors tend to specify the name of their tackled problems a little bit more. Maintaining that course, we will call our problem a 2 ECHELON MULTI-DEPOT MULTI-MIXED VEHICLE INVENTORY ROUTING PROBLEM (2E-MD-MMV-IRP). The 2 echelon comes from the observation that the replenishment of the schools happens in 2 steps. First the food is carried from the central warehouse to the small warehouses. After a while it is passed on from there to the schools. Multi-depot just refers to the fact that we consider scenarios with more than one small warehouse. Multi-mixed vehicle states that we are using multiple vehicles and they are not all of the same type so we consider mixed fleets.

Furthermore IRPs are categorized by the used replenishment policy. In our case we are considering the Maximum-Level (ML) policy, where the algorithm is free to decide on the amount of food that is distributed as long as all capacity constraints in schools, warehouses and vehicles are respected. There exists another policy which is more restrictive. Applying the Order-up-to-level (OU) policy means that every time a school is visited it needs to be filled up to its maximum storage capacity.

3 Decision Tool in Practice

3.1 User Interface



(a) (b)

(c)

Figure 2: Graphical User Interface with the tabs: Parameters (a), Details (b), More (c)

For designing graphical user interfaces (GUIs) in Python there are several packages one can use. The most common ones are maybe *Tkinter* and *PyQt5* among which we decided for the latter one. The main reason was that *PyQt5* contains a nice little feature called the *Designer* with which one can add buttons, spin boxes and lots more by drag and drop. After converting the created GUI from an html-file to a python-file by the command *pyuic5*, one can then connect the widgets to some events or actions they should

do if one clicks on them. So the *main.py* file opens the interface, reads the data from the excel file, saves all parameters and creates our problem which is then solved by pressing the 'Optimize'-button. Notice furthermore that we added a status bar in the bottom left which shows more detailed explanations for each setting if the cursor is on a button or an input field.

User manual

In the following, we will give a short manual how to use our interface and all possible functions:

Select files

At a first step one has to click on the 'Browse'-button and select the excel file with the input data. This excel file has to be created in a specific form which is described in detail in the next section. Be aware that if the central warehouse has been already listed in the excel file, then it has to be in the first row in the warehouse table.

Parameters

The next step is to choose different parameters. For simplicity they all have a certain default value which should fit for the most cases but can be varied if desired.

Planning period in weeks: Please select here the number of weeks over which the program should optimize. With the slider on the right the duration of one time step can be set, e.g. if a plan for two months should be computed but deliveries should take place only every two weeks, then set the time horizon to eight weeks and the slider to two. The numbers of the slider are the following (from left to right): 0.5, 1, 2, 3, 4, 5, ..., 11, 12

How often can a vehicle be used per time step?: Choose here how often a certain vehicle of a warehouse can be used in one time step, e.g. if your time step lasts two weeks, it is maybe good to have more vehicles available and deliver three times with each one so it can be set to three.

Loading time: This describes the time needed for unloading at each school in hours.

Max. time for a trip: This is the maximal time for one trip with one vehicle in hours.

Costs per km: These are the costs per kilometre in US-dollar for one vehicle.

Average speed: Here an average speed in km/h per vehicle is suggested.

Capacity of vehicles to central warehouse: Since we assume that the central warehouse has an extra vehicle fleet only delivering the small warehouses, one can enter here the capacitiy of those trucks in metric ton.

Details

If one just wants to use the data from the excel file, then nothing has to be changed here. But otherwise one will find some interesting settings here.

Checkbox 'Use vehicle fleet in excel file instead': If this is checked then the algorithm uses the vehicles from the excel sheet. Otherwise, the number of vehicles per warehouse and their capacities can be set here. Please notice that in this case all warehouses have the same number of vehicles all of the same capacity.

Checkbox 'Use first warehouse in excel file instead': If this is checked then the first entry in the table 'Warehouses' is used as the central warehouse. If one would like to add an extra central warehouse, which delivers the smaller ones, then please uncheck the box and input the coordinates of the new central warehouse and its number of vehicles. Please be aware that the capacity of those vehicles is then set equal to the value from the input field 'Capacity of vehicles to central warehouse' before. Furthermore, put the latitude in the first coordinate field and the longitude in the second one.

More

Please notice that the following settings have a huge impact on the performance of the optimization tool and should only be changed by experts of the algorithm.

Starting and ending point of τ: τ is responsible for the amount of steps of the algorithm. You can set a starting and an ending point where the algorithm should then stop searching for other solutions. The higher the difference between these two points, the longer the running time but the solution could be better.

Cooling factor: The cooling factor is a number between 0 and 1 that gets multiplied with τ at the end of each iteration of our algorithm to decrease τ. Since the algorithm stops when τ is less than a certain value, the cooling factor is responsible for the number of iterations. Those you can calculate with the button on the right.

Optimize

And finally press the 'Optimize'-button to run the program and to get a visualization and the output data.

3.2 Input data

Basically, the user interface takes only one excel file as input data. But that has to be created in a specific way such that the program can get all the data properly. **Therefore, it is important that column or sheet names are not changed.** So how is this structured?

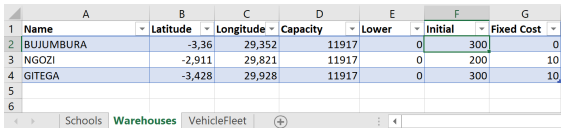
First of all, the excel file should only contain data sets over which the algorithm optimizes in the end. Therefore, there will probably exist one file for each country. Furthermore, the file consists out of three sheets called *Schools*, *Warehouses* and *VehicleFleet*.
The sheet *Schools* contains a table with 11 columns named *Name_ID*, *Total sum of Beneficiaries*, *Total Sum of Commodities*, *Consumption per day in MT*, *Consumption per week in MT*, *Latitude*, *Longitude*, *Capacity*, *Lower*, *Initial* and *Storage Cost* in which *Lower* stands for the minimum lower inventory and *Initial* for the initial inventory level. Each row represents the data for one school. An example of input data of Burundi you can find in Figure 3.

	A	B	C	D	E	F	G	H	I	J	K
1	Name_ID	Total Sum of Beneficiaries	Total Sum of Commodities	Consumption per day in mt	Consumption per week in mt	Latitude	Longitude	Capacity	Lower	Initial	Storage Cost
2	BENGA	409	1,245405	0,0818	0,409	-3,324368	29,45586	17,09	0,41	1,17	0
3	BIGERA I	556	1,849812	0,1112	0,556	-2,861	29,686	32	0,56	1,92	0
4	BIGWA I	664	2,02188	0,1328	0,664	-3,407186	29,39037	20,25	0,67	1,48	0
5	BIGWA II	646	1,96707	0,1292	0,646	-3,407186	29,39037	14,19	0,65	1,64	0
6	BIKINGI	866	2,595	0,1732	0,866	-3,540449	29,852342	6,9	0,87	2,93	0
7	BUBAJI I	1160	1,74	0,232	1,16	-3,460584	29,491987	40	1,16	3,05	0
8	BUBAJI II	548	1,66866	0,1096	0,548	-3,703722	29,918804	3,79	0,55	1,33	0
9	BUGEGA	551	0,827	0,1102	0,551	-3,70739	29,866121	38	0,56	1,73	0
10	BUGOMA	535	1,629075	0,107	0,535	-3,288039	29,37494	5,85	0,54	1,98	0
11	BUHINA	348	1,05966	0,0696	0,348	-3,445552	29,408689	4,57	0,35	0,81	0
12	BUHINYUZA I	941	1,412	0,1882	0,941	-3,765292	29,873677	5,72	0,95	3,5	0
13	BUHINYUZA II	582	0,873	0,1164	0,582	-3,762793	29,885101	9,53	0,59	1,63	0
14	BUHOMBA	1252	3,81234	0,2504	1,252	-3,299757	29,343583	13,4	1,26	3,36	0
15	BUHONGA I	787	2,346015	0,1574	0,787	-3,433189	29,410322	7,13	0,79	2,76	0
16	BUHONGA II	720	2,136	0,144	0,72	-3,433189	29,410322	4,46	0,72	2,38	0
17	BUHONGA III	688	2,09496	0,1376	0,688	-3,433189	29,410322	12,15	0,69	1,77	0
18	BUHONGA IV	597	1,785465	0,1194	0,597	-3,433189	29,410322	2,82	0,6	1,99	0
19	BUKORO	1298	3,86	0,2596	1,298	-3,54099	29,867357	10,35	1,3	4,16	0
20	BUNIHA	1082	3,599814	0,2164	1,082	-3,047	29,873	5,88	1,09	4,14	0
21	BURIZA	970	1,455	0,194	0,97	-3,775186	29,893953	14,43	0,97	3,3	0
22	BUSIGA I	1224	4,018968	0,2448	1,224	-2,879	29,735	45	1,23	4,77	0

Schools | Warehouses | VehicleFleet

Figure 3: Excel data to input schools

The sheet *Warehouses* includes a table with seven columns named *Name*, *Latitude*, *Longitude*, *Capacity*, *Initial* and *Fixed Cost* and each row represents again one warehouse. Please make sure to put the central warehouse in the first row.
Last but not least there is the sheet *VehicleFleet* which holds a third table with five columns named *Warehouse*, *Plate Nr*, *Make*, *Model* and *Capacity in MT* in which each row stands for one vehicle. Here, it is mandatory that the name of the Warehouse in column *Warehouse* is exactly the same as in the column *Warehouse* in the sheet *Warehouses*. An example can be seen in Figure 4.



	A	B	C	D	E	F	G
1	Name	Latitude	Longitude	Capacity	Lower	Initial	Fixed Cost
2	BUJUMBURA	-3,36	29,352	11917	0	300	0
3	NGOZI	-2,911	29,821	11917	0	200	10
4	GITEGA	-3,428	29,928	11917	0	300	10
5							
6							

Schools | Warehouses | VehicleFleet

(a)

	A	B	C	D	E
1	Warehouse	Plate Nr	Make	Model	Capacity in MT
2	GITEGA	CD44A95	RENAULT 6X4	350,34	18,00
3	GITEGA	CD44B02	RENAULT 6X4	350,34	18,00
4	GITEGA	CD44A89	RENAULT4X4	300,19	8,00
5	GITEGA	CD44A91	RENAULT 4X4	300,19	8,00
6	GITEGA	CD44A98	RENAULT 4X4	300,19	8,00
7	GITEGA	CD44A54	TOYOTA DYNA	0	3,50
8	GITEGA	E059AIT	TOYOTA PIC-UP	Land cruiser	1,50
9	GITEGA	CD107-98U	TRAILER	0	15,00
10	BUJUMBURA	CD44A96	RENAULT 6X4	350,34	18,00
11	BUJUMBURA	CD44A52	RENAULT 4X4	300,19	8,00
12	BUJUMBURA	CD44A81	ISUZU	0	4,20
13	BUJUMBURA	CD44A55	ISUZU	0	4,20
14	BUJUMBURA	CD44A86	ISUZU	0	4,20
15	BUJUMBURA	CD44A87	ISUZU	0	4,20
16	BUJUMBURA	CD44A35	TOYOTA DYNA	0	3,50
17	BUJUMBURA	CD44A25	TOYOTA DYNA	0	3,50
18	BUJUMBURA	CD44A31	TOYOTA PIC-UP	Land cruiser	1,50
19	BUJUMBURA	E058AIT	TOYOTA PIC-UP	Land cruiser	1,50
20	NGOZI	CD44A88	RENAULT 6X4	350,34	18,00
21	NGOZI	CD44A94	RENAULT 6X4	350,34	18,00
22	NGOZI	CD44B01	RENAULT 6X4	350,34	18,00

Schools | Warehouses | VehicleFleet

(b)

Figure 4: Excel data to input warehouses (a) and vehicles (b)

3.3 Output data

Since the ability of working with the computed data in the end is indispensable, it is quite important to have a clear and well structured output. Therefore, we produced a visualization as well as an excel file.

3.3.1 Visualization

The visualization is an interactive map, implemented with *plotly*, and opens automatically as an html-file when the program finishes running. In this, one can choose between different time steps in the bottom left corner and see how the routes look like.
The green circles represent the schools, the blue ones the warehouses and the red one the central warehouse. Furthermore, the current inventory levels of schools and warehouses are shown as numbers on the corresponding circles. Names, lower minimum inventory stocks, storage capacities, consumptions per time step and storage costs can be read off if one puts the mouse cursor on the different circles, as in Figure 5. In addition to that, one can zoom in by drawing a rectangle around an area and zoom out by double clicking. In this way routes can probably be recognized more easily and, by setting the cursor on a route line, the distances, quantities for the next schools and the type of the used vehicle are displayed (Figure 6).

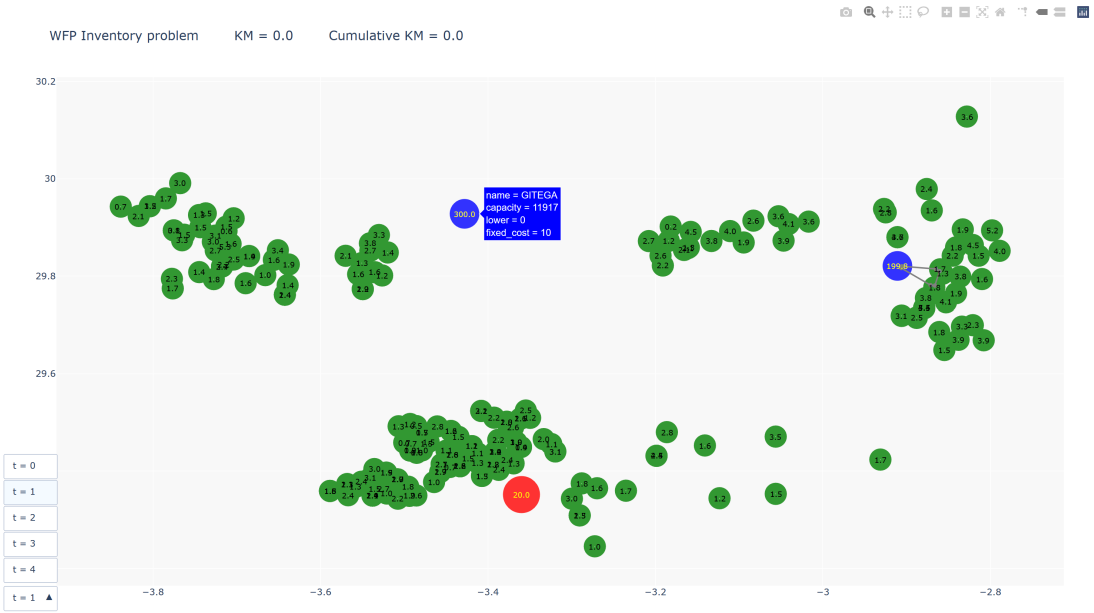


Figure 5: Output visualization

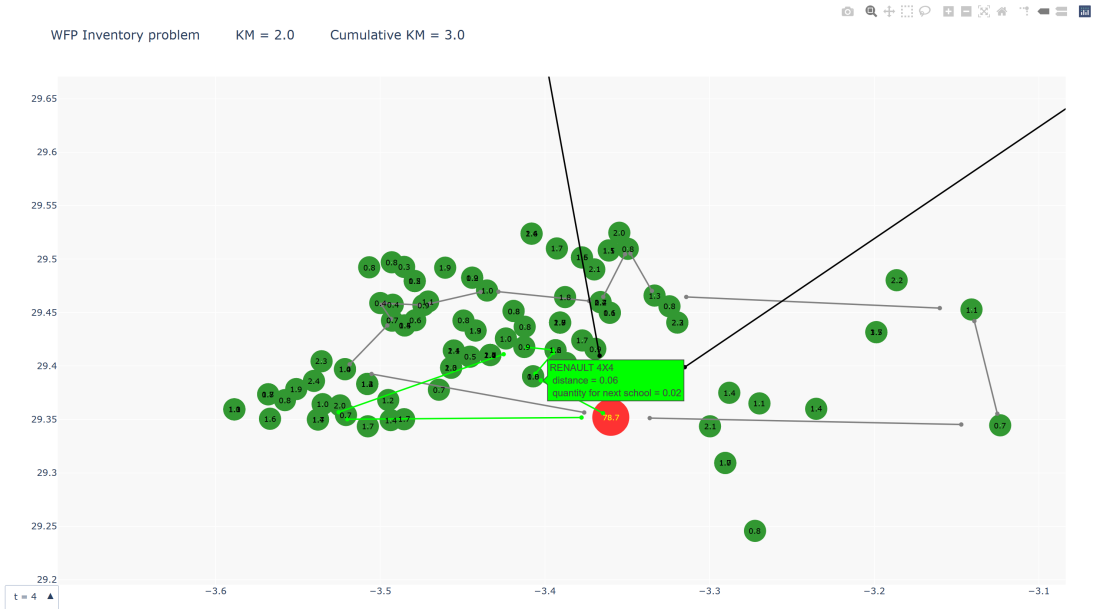
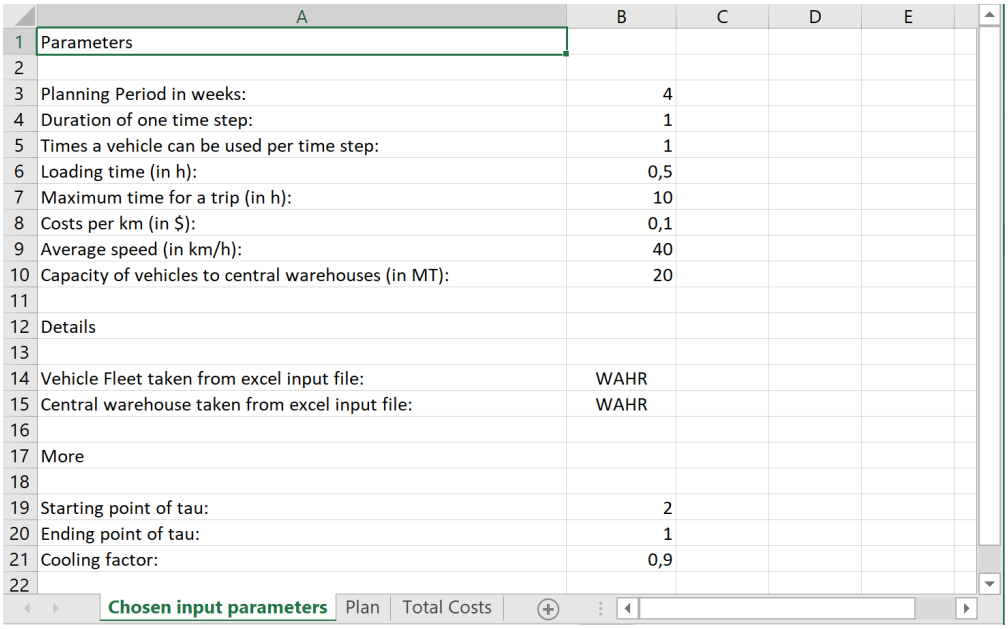


Figure 6: Visualization zoomed in

3.3.2 Output for real world problem

In Figure 7, 8 and 9 one can see an example how the output looks like. There are again three tabs: *Chosen Input Parameters*, *Plan* and *Total Costs*.
In the first tab one can find all parameters that have been inputted in the user interface in order to be able to recall the settings.
The second tab gives a detailed route - including schools and quantities - for a certain vehicle assigned to a timestep and a warehouse. For example in Figure 8 we have in the first week the Renault 6x4 from Warehouse Bujumbura delivering a total quantity of 14.148 MT to 6 schools. In detail it delivers e.g. 2.49 MT to the school Bigwa I.
In the last tab one can find a detailed overview of the different vehicles and their travelled distances and costs.



	A	B	C	D	E
1	Parameters				
2					
3	Planning Period in weeks:	4			
4	Duration of one time step:	1			
5	Times a vehicle can be used per time step:	1			
6	Loading time (in h):	0,5			
7	Maximum time for a trip (in h):	10			
8	Costs per km (in $):	0,1			
9	Average speed (in km/h):	40			
10	Capacity of vehicles to central warehouses (in MT):	20			
11					
12	Details				
13					
14	Vehicle Fleet taken from excel input file:	WAHR			
15	Central warehouse taken from excel input file:	WAHR			
16					
17	More				
18					
19	Starting point of tau:	2			
20	Ending point of tau:	1			
21	Cooling factor:	0,9			
22					

Chosen input parameters | Plan | Total Costs

Figure 7: Excel output: Parameters

	A	B	C	D	E	F
1	**Timestep**	**Point in time (in weeks)**	**Warehouse**	**Vehicle**	**Quantity**	**Route**
2	1	1	BUJUMBURA	RENAULT 6X4	2,49	BIGWA I 2
3					2,124	MWAZA 9
4					2,055	MUTUMBA II 4
5					4,64	BUBAJI I 12
6					1,219	BENGA 0
7					1,62	MUYANGE II 6
8				Total:	14,148	6
9						
10	1	1	NGOZI	RENAULT 6X4	2,066	BIGERA I 1
11					3,612	MWUMBA 10
12					4,672	MUYANGE I 5
13				Total:	10,35	3
14						
15	1	1	GITEGA	RENAULT 6X4	2,736	BIKINGI 3
16					3,392	MUZIMA 8
17					3,285	MUYUGA 7
18					2,204	BUGEGA 14
19					2,54	NDAGO 11
20					1,93	BUBAJI II 13
21				Total:	16,087	6
22						
23	4	4	BUJUMBURA	RENAULT 6X4	0,59	MUYANGE II 6
24					0,056	BENGA 0
25					0,708	MWAZA 9
26				Total:	1,354	3
27						

Chosen input parameters | **Plan** | Total Costs

Figure 8: Excel output: Plan

	A	B	C	D	E	F	G
1	**Timestep**	**Point in time (in weeks)**	**Warehouse**	**Vehicle**	**Quantity of food**	**Distance (km)**	**Cost (dol)**
2	1	1	BUJUMBURA	RENAULT 6X4	14,148	114,979	11,5
3	1	1	NGOZI	RENAULT 6X4	10,35	145,604	14,56
4	1	1	GITEGA	RENAULT 6X4	16,087	93,648	9,36
5	4	4	BUJUMBURA	RENAULT 6X4	1,354	99,424	9,94
6	4	4	NGOZI	RENAULT 4X4	0,074	31,99	3,2
7	4	4	GITEGA	TRAILER	4,242	65,022	6,5
8							
9				Total :	46,255	550,667	55,06
10						All costs (including holding cost) :	55,07

Chosen input parameters | Plan | **Total Costs**

Figure 9: Excel output: Total costs