Rodwin Luke Gillamac ALVAREZ <ralvarez.2016>

GONG Zian <zian.gong.2017>

Yusuke MINAMI <yminami.2017>

1. Abstract

For bike-sharing businesses, it is a critical part in their operation to address the imbalance in the distribution of the bikes in the bike-sharing systems, which is caused by the uneven nature of bike usage by the users. In this project, two different rebalancing strategies are explored using agent-based simulation and their effectiveness is evaluated. The group makes use of the multiagent tool NetLogo to create a simulation environment to effectively compare strategies such as the two strategies implemented in finding out which performs better with different parameter settings.

2. Introduction

There are two types of bike-sharing systems. One kind, the dock-less system as seen in Singapore, lets users rent the bikes when they can find one and return the bikes they have rented simply by locking the bikes at the end of their trips. In other words, no stations or docks are set up as extra infrastructure for the system. The other kind, the docked system, uses docks to store the available bikes. Users of such systems can only rent a bike from bike-sharing stations and return the bikes to other stations near where they want to go. For such systems, docks and other pieces of equipment are infrastructure they must invest in before they can start the business.

In this project, we focus on rebalancing strategy for the docked bike-sharing system. User trips in such systems usually show an uneven pattern, causing imbalance in distribution of bikes. Addressing such imbalance is an important task for such systems, and rebalancing strategies become a critical part in the operation. Computation to find the optimal rebalancing strategy by solving optimization problem is computationally expensive, thus computationally less expensive methods such as Agent-based simulation are required.

The system we investigated in our project is the Boston Hubway bike sharing system. For this system, we built a simulator using NetLogo to experiment on two different rebalancing strategies and evaluated their effectiveness. The two strategies are forecast-based strategy and status-based strategy.

In the forecast-based strategy, we calculate the hourly forecast for each station in each hour for the month and use that forecast to allocate resources (rebalancing vehicles) to move bikes from low-demand, high-supply stations to high-demand, low-supply stations.

In the status-based strategy, we use the station's status (full / empty and capacity information) to guide the trucks to transfer bikes. We then run experiments with different parameter settings to evaluate the effectiveness of the strategies.

Furthermore, the simulation model can be used to investigate other strategies that the group has not included in the scope to determine a good rebalancing solution for the environment in a cost-effective manner.

It is also worth noting that cost of rebalancing is an important area of consideration when identifying strategies to address this optimization problem. This creates an additional dimension to consider when finding the optimal rebalancing model with regards to the resources or trucks to be used. However, because of the lack of data to incorporate this dimension, the group decided to leave it out of scope for this study and focus on the comparison of rebalancing models through simulation without factoring direct costs.

In this project, the following terminology is used:

- Let station's "capacity" := max # of bikes that can be potentially stored
- Let # of "bikes" := # of bikes stored at a timestep
- Let # of "docks" := (# of station's capacity) - (# of "bikes" stored at a timestep)
- Let "trucks" := the transport vehicles carrying bikes with a capacity of 20
- Let "forecast" := estimated arrival – estimated departure (all at next time step)

3. Literature Review

There were several literatures that tackled rebalancing problem in Bike-Sharing System (BSS). Ghosh et al (2017) focused on rebalancing of bikes using vehicles through optimization and routing algorithms considering bike forecasted demands. The challenge was computation time. They reported that they could only solve the optimization problems with at most 60 base stations, 38 time steps and 5 vehicles (called "trucks" in this project) within a threshold time of 12 hours even after applying the Lagrangian dual decomposition (LDD), thus aggregated stations by geographical proximity based clustering. In this project, we overcame computation time problem by applying agent-based simulation approach.

According to them, previous works in inventory management have represented and verified the random arrival of customers. More importantly, earlier works in bike sharing have also represented the random arrival of customers at each station and at each time step using a Poisson distribution and assumed that customers choose their destination station with a certain probability. In a similar way, we also represent the arrival of customers at a base station in a time step using a Poisson distribution. Since we can only know about the satisfied demand from the data sets, the mean of the Poisson distribution is the average served demand (outgoing flow) in that time step. We adopted this way to simulate user trips.

C. Diez et al (2017) investigated on incentivizing users to slightly deviate from origins and destinations to drive user-driven balancing. We did not adopt this approach as how people change their behavior is complexed.

4. A statement of the regularities to explore

Our model can experiment results based on the different rebalancing method options that we implement, and it is interesting to find out which set of parameters is the best for the system.

5. Model description

The model consists of 2 submodels: user trips and rebalancing trucks.

To simulate users' bike trips, real-world data from Boston Hubway system downloaded from http://hubwaydatachallenge.org/ was used. The datasets include the following 3 tables.

- Bike trip table including timestamp & station ID for each user trip in the period
- Station location table including station ID, latitude & longitude information
- Station status table including station ID, number of bikes and docks and station capacity

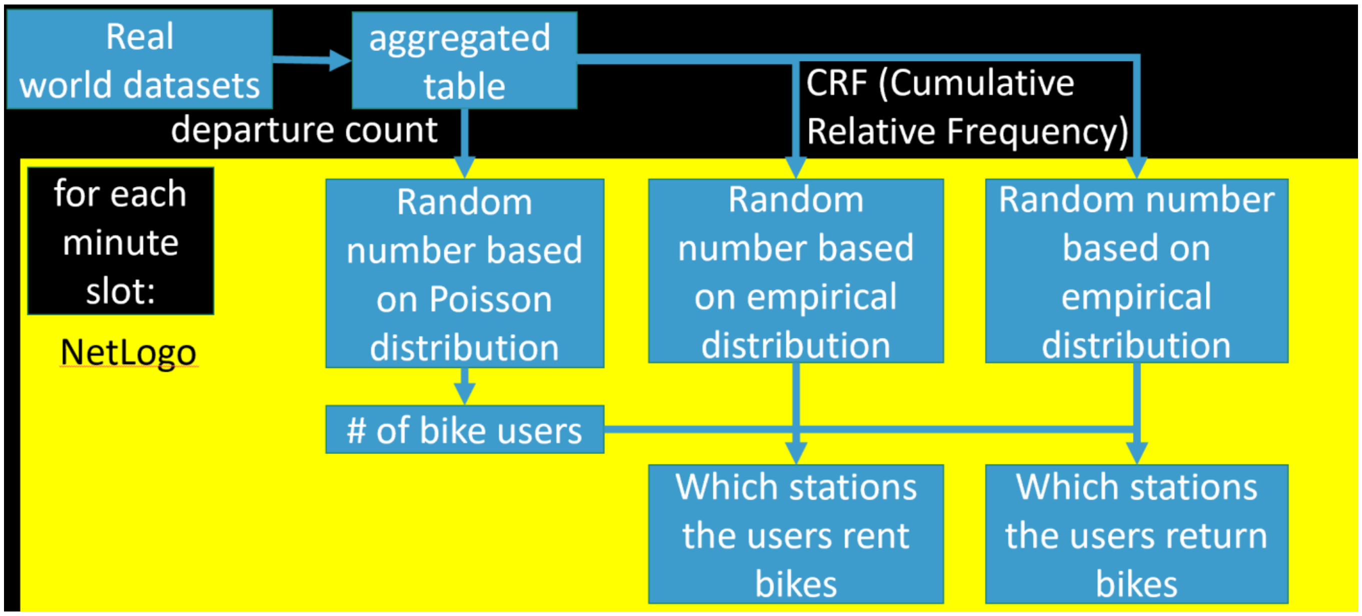
Using these 3 tables, the following 3 CSV files were generated for use in simulation in NetLogo.

- CRF (Panel data): the numbers of departure (renting a bike) and arrival (returning a bike) for each station were aggregated by hour (0 - 23), and CRF (Cumulative relative frequency) values were calculated
- Interstation table: the distance between stations using longitude and latitude using GeoPy
- Forecast table: difference of the aggregated number of arrivals and number of departures.

It is worth noting that the group has implemented a very crude forecasting methodology as the focus of this study was never to employ machine learning algorithms to produce a more accurate forecast. We do feel however that with our model and setup, it will be easy for future work to incorporate better forecasts and test it in our simulation model since the input is a plain csv file of the forecasts.

User Trips Submodel

For each time step (a minute), behavior of user trips was simulated as follows.


Real world datasets
aggregated table
departure count
CRF (Cumulative Relative Frequency)
for each minute slot:
NetLogo
Random number based on Poisson distribution
Random number based on empirical distribution
Random number based on empirical distribution
# of bike users
Which stations the users rent bikes
Which stations the users return bikes

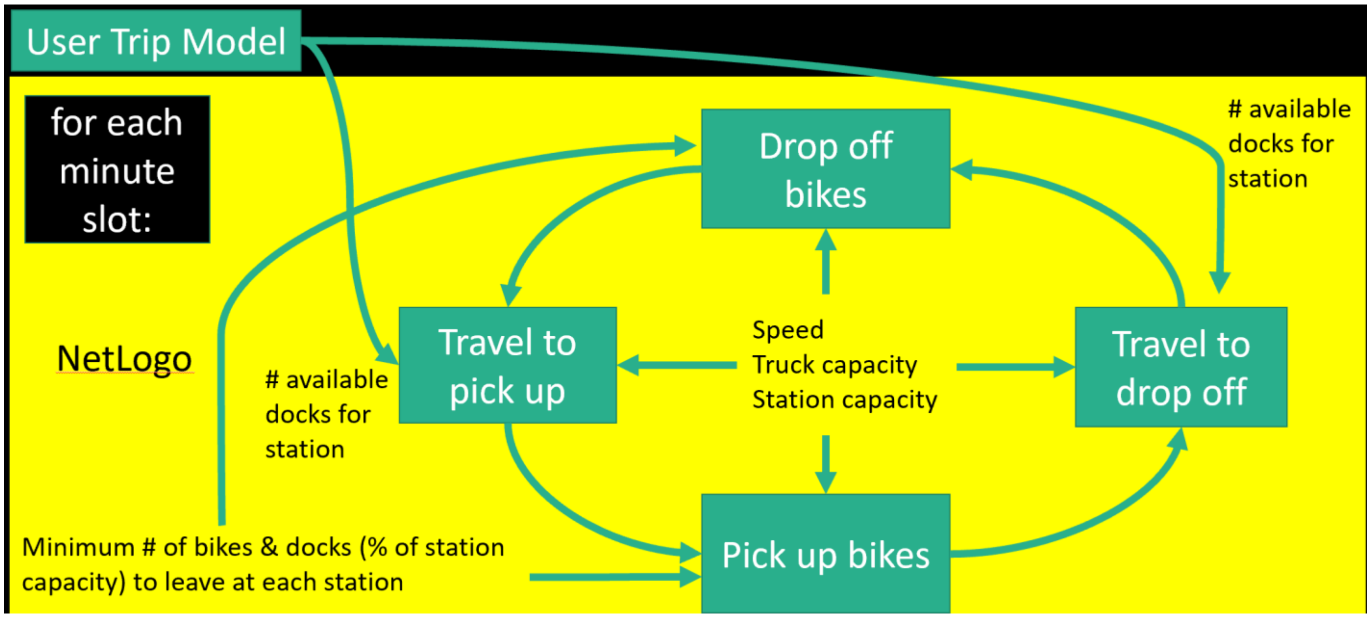
[1: Determine the number of user trips] A random number was generated based on Poisson distribution setting the mean value to the corresponding value (hourly value / 60) from the panel data to determine the number of user trips in a time step.

[2: Determine the stations from which users depart] A random number was generated based on uniform distribution between 0 and 1 to compare with each station’s cumulative relative frequency read from the panel data and determine the departure station. If the determined station has no bike, then the nearest station with a bike is searched using the interstation distance table, which requires additional computation.

[3: Determine the stations at which users arrive] Arrival stations were determined likewise.

Rebalancing Trucks Submodel

In each time step (a minute), the trucks were in states of either traveling to pick-up a station, picking up bikes, traveling to a drop-off station, or dropping off bikes, and the order of transitions was fixed in this project and it took around 30 minutes per cycle depending on the distance between the assigned stations.



To determine the urgency of stations to visit, forecast-based and status-based were implemented.

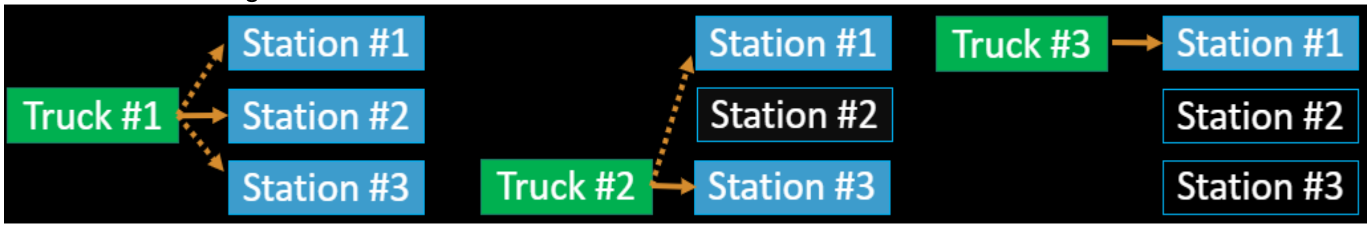
Rebalancing method	Forecast-based	Status-based
What data to use?	Use the aggregated **historical** numbers of bikes rented & returned at each station **for each hour**	Use the number of available bikes & docks at each station **in the beginning of each hour**
Which station to pick up?	Prioritize stations **anticipated** to have more bikes (arrival - departure) at a future time step	Prioritize stations with less docks
Which station to drop off?	Prioritize stations **anticipated** to have least bikes (arrival - departure) at a future time step	Prioritize stations with less bikes
How many to pick up or drop off?	Based on the forecasted pickup or drop off at next time step	Up to the percentage to the station capacity specified by the slider control (e.g. 30%)

For the forecast information, aggregated number of arrivals and number of departures are subtracted together. This net value represents the forecast value for that given time step. When the value is positive, there is an oversupply in the station while if it is negative, then there is an undersupply.

The forecast data is then picked-up by NetLogo and assigned to each station through a hash table extension in NetLogo with the keys as the military time and the values as the forecasted value. The forecast-based simulation will then pick the forecast value of all stations at the next hourly time step to determine the forecasted value and select the stations accordingly for pickup or drop off.

Status-based method simply referred to the number of bikes and docks at each station to determine the urgency.

Target stations for pickup or drop-off were assigned to each truck considering both urgency and proximity. Each truck chooses the closest station among the top (number of trucks: 3 for example) most urgent stations. Urgency is prioritized first so the most urgent station is assigned to the truck closest to it, then the next station is assigned to the truck closest to it and so on.



6. Model parameters

There are a total of 12 parameters we used in our model which can be categorized into three groups namely model wide, tunable and aesthetic parameters. High level descriptions of the parameters as follows:

Model Wide Parameters:

- Simulation time step - Used to determine what time step is used in the simulation (set to 1 minute)
- Rebalancing Method - Used to select the method of the model i.e. forecast- or status-based or none
- Days to simulate - Duration of the experiment (Set to 7 days)

Major Tunable Parameters:

- Number of Trucks – The number of trucks rebalancing bikes to stations
- Truck Capacity – The capacity of each truck (fixed to 20; tunable in future work)
- Pct station capacity to move – The slider to determine how many bikes, in terms of the percentage to the station capacity, to pickup/drop off in the status-based method (e.g. If this value is set to 10% and the station capacity is 20, then the trucks will pick up or drop off 2 bikes.)

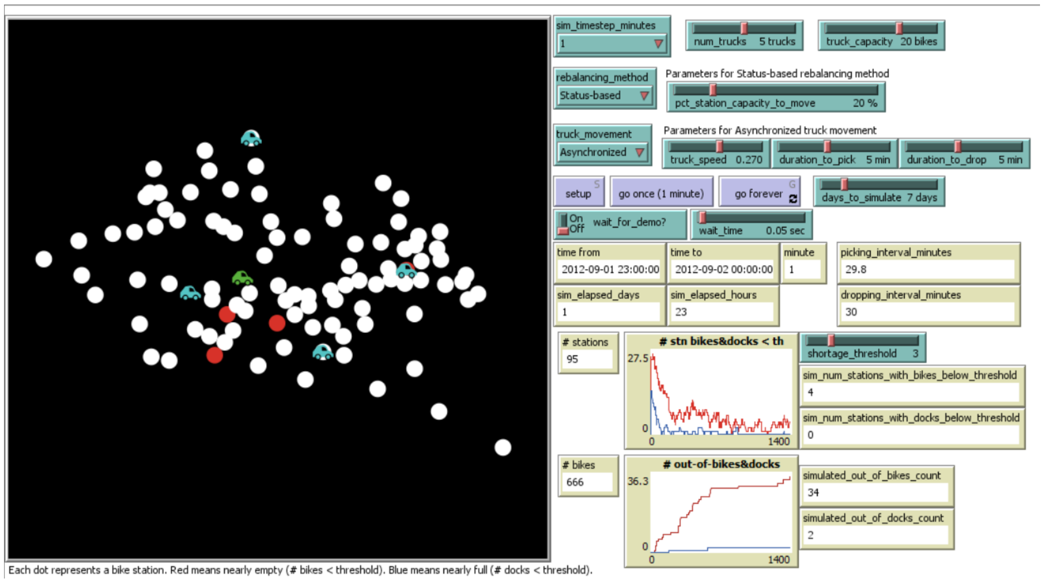
Minor Parameters:

- Truck Movement – Whether trucks rebalance simultaneously (synchronously) or independently (asynchronously)
- Truck Speed – The speed as to which the trucks move around to rebalance
- Pickup and drop-off duration – the duration of the pickup or drop-off action of truck

Aesthetic Parameters:

- Wait for Demo – Switch to slow down speed for demo purposes by adding wait time specified

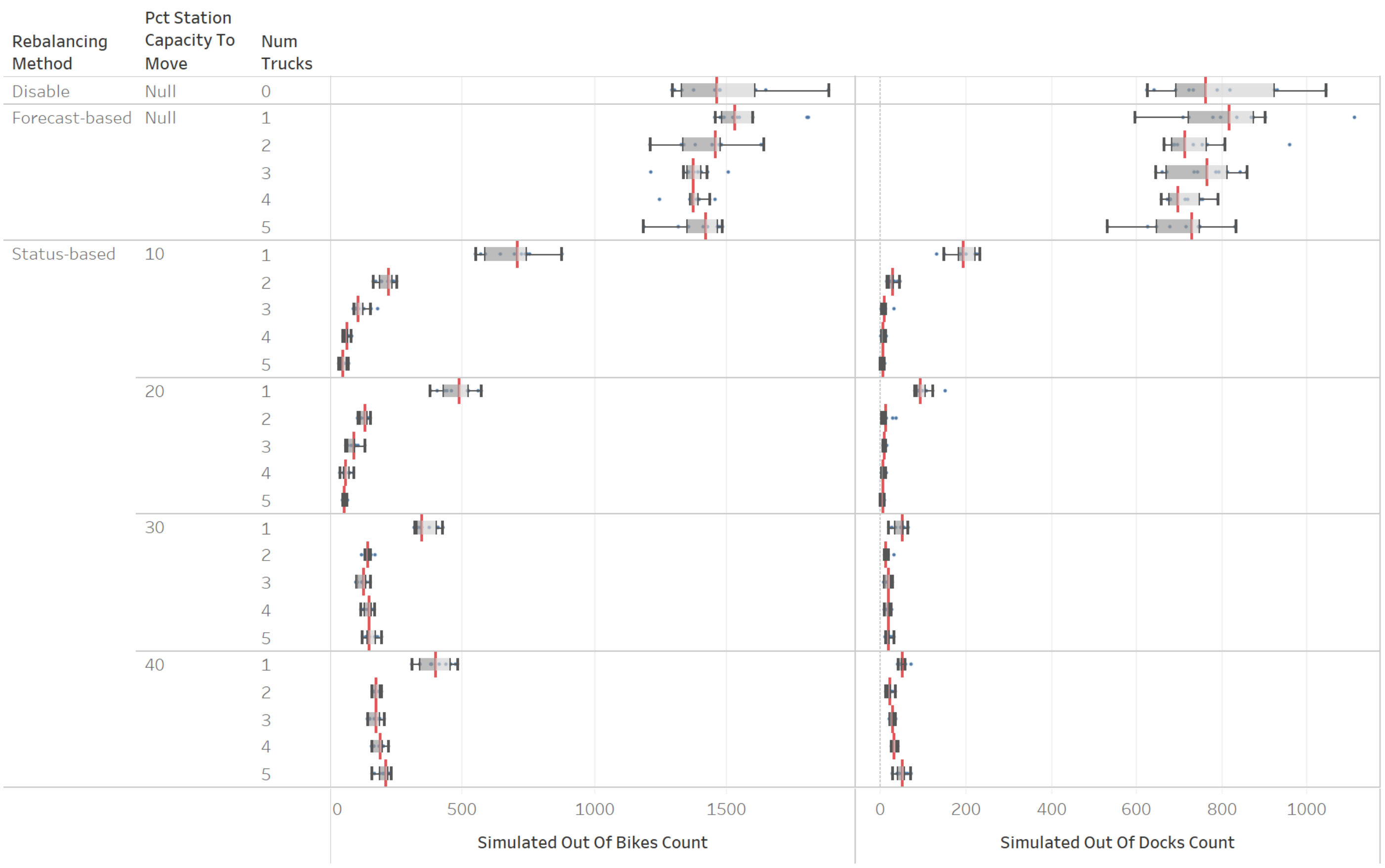
- Wait time – Wait time in between truck movements
- Shortage Threshold – The threshold parameter to show when stations change in color

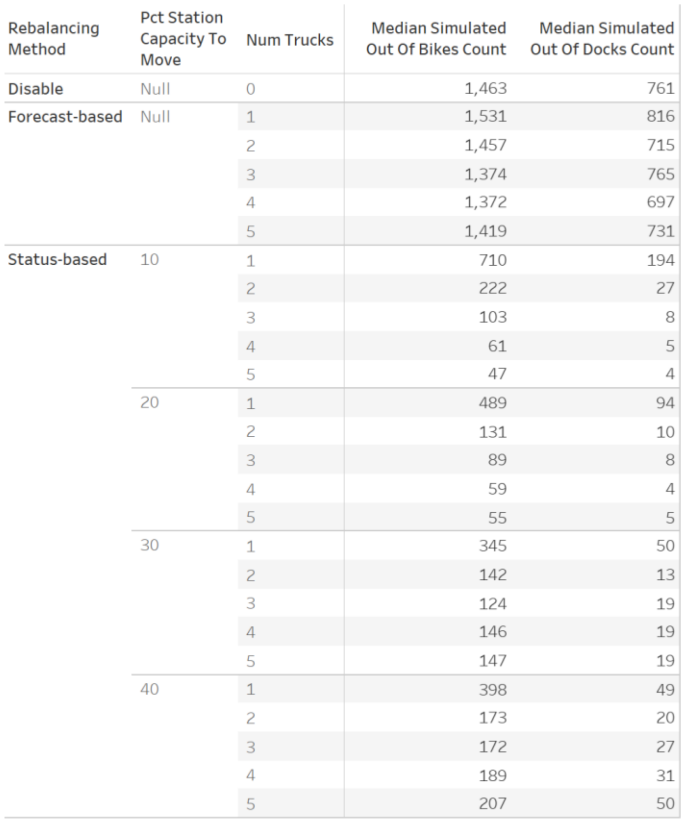

sim_timestep_minutes: 1
num_trucks 5 trucks
truck_capacity 20 bikes
rebalancing_method: Status-based
Parameters for Status-based rebalancing method
pct_station_capacity_to_move 20 %
truck_movement: Asynchronized
Parameters for Asynchronized truck movement
truck_speed 0.270
duration_to_pick 5 min
duration_to_drop 5 min
setup
go once (1 minute)
go forever
days_to_simulate 7 days
On Off wait_for_demo?
wait_time 0.05 sec
time from: 2012-09-01 23:00:00
time to: 2012-09-02 00:00:00
minute: 1
picking_interval_minutes: 29.8
sim_elapsed_days: 1
sim_elapsed_hours: 23
dropping_interval_minutes: 30
# stations: 95
shortage_threshold 3
sim_num_stations_with_bikes_below_threshold: 4
sim_num_stations_with_docks_below_threshold: 0
# bikes: 666
simulated_out_of_bikes_count: 34
simulated_out_of_docks_count: 2

Each dot represents a bike station. Red means nearly empty (# bikes < threshold). Blue means nearly full (# docks < threshold).

7. Results, Verification and Validation

Experiments of running simulation for 7 days (1st to 7th Sep 2012) were run 10 repetitions for each of parameter settings (Rebalancing disabled, Forecast-based, and Status-based), and the 2 performance metrics, out-of-bikes (cases when a user cannot find a bike at the station originally assigned) and out-of-docks count (cases when a user cannot find a dock at the station originally assigned), were plotted below as boxplots with red lines representing the median values (numeric table follows).





Rebalancing Method	Pct Station Capacity To Move	Num Trucks	Median Simulated Out Of Bikes Count	Median Simulated Out Of Docks Count
Disable	Null	0	1,463	761
Forecast-based	Null	1	1,531	816
		2	1,457	715
		3	1,374	765
		4	1,372	697
		5	1,419	731
Status-based	10	1	710	194
		2	222	27
		3	103	8
		4	61	5
		5	47	4
	20	1	489	94
		2	131	10
		3	89	8
		4	59	4
		5	55	5
	30	1	345	50
		2	142	13
		3	124	19
		4	146	19
		5	147	19
	40	1	398	49
		2	173	20
		3	172	27
		4	189	31
		5	207	50

Compared with the rebalancing submodel disabled (2 performance metrics' median values: 1463 and 761, respectively), forecast-based rebalancing did not perform well in terms of the 2 performance metrics. Status-based rebalancing, on the other hand, performed much better. The parameter setting (47 and 4) was Status-based rebalancing by 5 trucks moving 10 % of each station capacity. However, the more trucks deployed, the costlier it is. Depending on the cost such as drivers' salary and fuel, settings such as 2 trucks moving 20% (131 and 10) are more cost efficient.

The computation time was measured and analyzed below.

Frequency to find another station in user trips submodel	Rebalancing trucks submodel	Parameter combinations	Repetitions	Simulated period [days]	Total computation time [min]	Computation time per run [min]	Computation speed per real-time
Often	(None)	1	10	7	8	0.800	12600
Often	Forecast-based	5	10	7	40	0.800	12600
Rarely	Status-based	20	10	7	117	0.585	17231

The simulation without rebalancing ran 12600 times the real-time. Simulation with forecast rebalancing ran as fast as simulation without rebalancing. This implies that computation time for forecast-based submodel was negligible and the bottleneck of computation is the user trips submodel. The computation for Status-based rebalancing is similar to Forecast-based, thus Status-based rebalancing can be assumed negligible as well. The measured computation speed for Status-based rebalancing ran even faster than simulation without rebalance. This can be explained that Status-based rebalancing could drastically reduce the out-of-bike and out-of-docks events which requires additional computation to find a nearest station with an available bike or dock.

The time complexity of the 2 submodels was analyzed below.

Submodel	Time Complexity	Description
User Trips	O(#BikeUsages * #Stations)	For each of bike usage, station table is searched.

Rebalancing Trucks	O(#Trucks)
	
	If sorting stations is the bottleneck:
	O(#Stations * log(#Stations))
		Each of truck does rebalancing jobs. Stations needs sorting in the order of urgency (forecast or status) every hour.

The time complexity of both sub models was not computationally expensive. The user trips submodel, the one thought to be the bottleneck, is only linear to the number of bike usages and number of stations. Thus, the developed model will finish in reasonable time even more shared bikes are used or more stations are built.

8. Conclusion

An agent-based simulation model, which would run much faster than solving the computationally expensive optimization problem, could be developed. In addition, the model created serves as a good vaulting platform for testing and comparison for future better forecasting and optimization strategies in the bike sharing space.

It is observed that a forecast based rebalancing method requires much better forecasting to be able to compete with the performance of status-based rebalancing. Though the focus of this project was not to explore and find the best forecasting approach to these problems, the model generated would be able to compare these approaches on how it fares with the baseline of the status-based approach. This work then paves way for other future work in trying to test out further optimization solutions through better forecasts to find a more robust solution for bike-sharing.

9. Acknowledgements

The authors greatly thank Prof Cheng Shih-fen for his suggestions.

10. References

- Hubway Data Visualization Challenge. Retrieved October 17, 2018 from http://hubwaydatachallenge.org/
- GHOSH, Supriyo; VARAKANTHAM, Pradeep; ADULYASAK, Yossiri; and JAILLET, Patrick. Dynamic repositioning to reduce lost demand in Bike Sharing Systems. (2017). Journal of Artificial Intelligence Research. 58, 387-430. Research Collection School Of Information Systems.
- Diez C., Sanchez-Anguix V., Palanca J., Julian V., Giret A. (2017) A Multi-agent Proposal for Efficient Bike-Sharing Usage. In: An B., Bazzan A., Leite J., Villata S., van der Torre L. (eds) PRIMA 2017: Principles and Practice of Multi-Agent Systems. PRIMA 2017. Lecture Notes in Computer Science, vol 10621. Springer, Cham
- CHENG, Shih-fen. (2018, September 24). CS603: MULTI-AGENT SYSTEMS WEEK 5: MODELING UNCERTAINTIES. Retrieved September 24, 2018 from Singapore Management University's Intranet
- EDMUNDS, Doug. (2017, February). Sequential timeseries using CSV. Retrieved November 7, 2018, from http://modelingcommons.org/browse/one_model/4990#model_tabs_browse_info
- user299791. (2014, November 14). Find lists intersection in NetLogo. Retrieved November 5, 2018, from https://stackoverflow.com/questions/26928738/find-lists-intersection-in-NetLogo