Carnegie Mellon University

Intersection Traffic Light Optimization Problem

Vanessa Jiang, Maggie Liu, Cara Yi, Jessica Zhao

21-393 Operation Research II

Professor Zhang

Introduction

Traffic signals are a common type of intersection control. The goal in signal movement design is to improve overall safety, to decrease average waiting time, and to equalize the level of services for all (or most) traffic flows. Several traffic intersections in Pittsburgh stop traffic from all directions and allow pedestrians to simultaneously cross in all directions, including diagonally, and this type of traffic signal movement is called a pedestrian scramble. First introduced in the 1940s, the pedestrian scramble became less and less popular, because it prioritizes the flow of pedestrians over the flow of cars, leading to moments where no cars nor people can cross an intersection and potentially increasing certain parties' wait times, which may also result in jaywalkers.

Currently, traffic signals operate according to several methods. There are three types of traffic signals: pre-timed, semi-actuated, and fully actuated. Pre-timed intersections operate on a fixed cycle length; semi-actuated intersections have sensors on some or all movements except the main flow(s). Non-detected flows are controlled on a pre-timed policy, and fully actuated intersections have sensors in all directions. Each movement has an initial green light cycle length for queuing vehicles to get through; this cycle length is extended if the sensor detects a car moving through the intersection.

Problem Statement

Because traffic is so notorious in Pittsburgh, we decided to see how we can optimize traffic flow at four-way intersections. Using what we know about the ways traffic signals currently operate, in addition to traffic rules and other intersection properties, how can we effectively analyze and improve the overall traffic in Pittsburgh? This problem is nontrivial because we must consider a variety of factors including car flow, pedestrian flow, car speeds, pedestrian walk speeds, car and pedestrian intended directions, etc. We aim to design a method and choose parameters that will "satisfy" both pedestrians and cars. For example, we would not want to only cater to pedestrians and hence letting the cars wait for longer times. Therefore we consider both the average pedestrian and car wait times when evaluating the performance of our methods.

Some questions we answer through this research are: How often should the lights change for cars and for pedestrians? How does this affect the pedestrian and car wait times? Would having sensors to situationally change light behavior improve traffic? Specifically, which of the three approaches, non-adaptive, time-adaptive, or order-adaptive perform the best? How do other parameters in our model such as the rates of car and pedestrian arrivals affect car and pedestrian wait times?

To answer these questions, we took the following general approach: We first generated sample pedestrian and car flow distributions and designed and implemented three logical algorithms to model the intersection traffic according to a set of basic traffic rules and conditions. We then ran our simulation on a variety of parameters and analyzed how changing each parameter affected car and pedestrian wait times. Lastly, we demonstrate some of our conclusions via graphs and plots.

Mathematical modeling

To simulate the traffic flow, we assume a Poisson arrival process for both pedestrians and cars and test out different intensity parameters for each. The rates we examined range from 10 arrivals /sec to 100 arrivals /sec with an increment of 10.

We have 3 different types of traffic signal movements at the intersection: 1) Horizontal lights are green, which means cars can go East to West and people who want to cross horizontally can go; 2) Vertical lights are green, which is for cars going North to South and pedestrians in that direction; 3) All lights for cars are red, so pedestrians can cross in whatever direction they want, including diagonally.

We also have 3 different algorithms we use to control the traffic lights: 1) The non-adaptive version has the same length for all 3 lights, and goes in order of horizontal green, vertical green, and all red; 2) The time-adaptive version has the same order of lights, but we assume there are sensors that can detect how many pedestrians and cars are at the intersection waiting and can use that to shorten the current light if there is significant flow in another direction; 3) The order-adaptive version has the same fixed length for all lights, but when it is time to switch, we assume there are sensors again and use that to determine which light should come up next, that would be the most efficient choice for the current traffic.

We needed several assumptions for the model to be close to reality. The minimum time it takes for a pedestrian to cross is assumed to be 10 seconds, and at the intersection, only one car is allowed to cross every two seconds during a green light. In addition, there is a 2 second reaction time for cars when there is a light switch, which penalizes switching lights unnecessarily often in the adaptive case, which we will explain in more detail in the solution section.

Each simulation runs for 30 minutes, so there is enough time to mitigate some variation in data and so that the adaptive versions can make enough changes to reflect their advantages and disadvantages. To have a more direct comparison, for each set of parameters, we generate 400 iterations of traffic flow(with 4 different light intervals, and 10 different pedestrian's rate of arrival and car's rate of arrival). We use the same traffic flow for the 3 algorithms, and see which one does better by comparing the average wait time for cars and the average wait time for pedestrians. We also applied linear regression for the above-defined variables and average waiting times from 3 algorithms to see if there is any relationship that could help us compare our algorithms and understand the model further.

Solution

A main part of the project is building a simulator that can closely reflect what would happen in the real world. To start tackling this problem, we first defined two classes: Car and Pedestrian. The attributes stored by a Car object include the arrival time of the car, and which direction the car came from, which could be North, South, West, or East, since we are looking at an intersection. The attributes stored by a Pedestrian object include arrival time of the person, as well as the direction the person wants to travel in, which could be Horizontal, Vertical, or Diagonal. The second attribute of both Car and Pedestrian is randomly chosen from the list of possible values upon initialization of the object, since the arrival time generator mentioned above only generates the arrival time of cars and pedestrians, and not their directions etc.

For the actual simulator itself, we relied heavily on queues to keep track of how many cars and pedestrians are waiting at the intersection at any given time. Our simulator uses seconds as its unit, so at every second we do several things: First, we check if there are people or cars that

arrive at this second, and add them to the corresponding queue based on their direction or location. We mentioned in the previous paragraph that the direction and location are attributes to the class objects, so it is simple to retrieve these values, which makes our choice of structure advantageous for clean implementation. Second, we check which lights are currently on, and let pedestrians and cars pass accordingly. One thing to note is pedestrians cannot start walking if the time remaining is less than the minimum walk time. Finally, we check if it is time to switch lights, and if it is, we update relevant parameters to reflect this change in state. The last step is different for adaptive and non-adaptive approaches, which will be discussed next, but for the basic case, it uses a variable that keeps track of how long the current light has lasted, and compares it to how long the light should last.

For the adaptive case, we focused on developing two different approaches, and will discuss their advantages and disadvantages in the result section in detail. We called the first approach "time-adaptive", as the basic idea is that light durations may be shortened depending on the current traffic flow. For example, if the lights are green in the horizontal direction, but there is significantly more traffic in some other direction, we can shorten the current light so that lights in another direction would turn green sooner.

One problem that we faced initially while working on the time-adaptive approach was: if we just add a condition that shortens the current light and immediately changes the light, it could lead to pedestrians being stranded in the middle of the road as they cross. This is because a pedestrian would start crossing the road if they think the light will last longer than the minimum walk time, but if we alter the light lengths like this, there will be no way to know how much time is left until the next light switch. The solution we came up with was adding a variable called "flashing", which is analogous to the count down you would see at a real pedestrian crossing

light. So when the condition of shortening a light is met, we change "flashing" to true and keep a tracker of how long the light has been flashing for. No pedestrian should cross when the lights are flashing, if they have not started crossing already. We spent some time thinking about what the condition should be, to best reflect the traffic flow and the best decision. This is not trivial because pedestrians can cross together, but cars have a certain speed they can go at and they also have to wait in line to go through an intersection, so a buildup of cars should be worse than a buildup of pedestrians, since each car in the traffic jam will wait for even longer. We decided to use a factor that is the number of cars that can pass per second, which in the model is set to be some reasonable constant, and to scale cars to be comparable to pedestrians, we divide the number of cars by this factor. As we will later show in the results, this scheme works pretty well, but it could probably be improved and made more sophisticated, which is what can be worked on in the future.

The second of the two approaches is called the "order-adaptive" approach, where we fix the length of each light interval but make the order of the lights flexible, instead of going in a cycle between the 3 lights. To achieve this, we first calculate the total traffic flow(cars and pedestrians), where cars are scaled in the same way as the time-adaptive version, for all 3 light states. So for the horizontal green lights, the total traffic flow would be cars in the East and West direction and people who want to cross horizontally; for vertical green lights it would be North and South, and for 4-way red lights it would be all the pedestrians. Then, we find which state has the heaviest traffic flow, and we make that light state the next one. Initially, we did not check for all 3 states, and simply compared the two lights that are not currently on. However, the results were not good, so we proceeded to alter the algorithm and check for all 3 states, which led to significant improvements in the outcome.

Results & Discussion

By running a correlation test, we discover that by doing our first approach, the non-adaptive method, the pedestrian waiting time is positively correlated with the light interval, and the car waiting time is positively correlated with the car's rate of arrival. These two correlations are reasonable since we restrict the number of cars passing by the intersection while we have no restrictions on the number of people walking during green light. This property can be also seen in the correlation between car waiting time and car's rate of arrival in the other two algorithms, whereas pedestrian rate of arrival has little influence on any of the pedestrian waiting times. In the order-adaptive and non-adaptive approaches, pedestrian waiting times are correlated with the light interval. However, in the time-adaptive approach pedestrian waiting time has very little correlation with all parameters while maintaining a low waiting time. Therefore, the shorter the interval, the better the adaptive and order-adaptive algorithms perform on pedestrian waiting time, and the car waiting time in all three is mostly affected by the car's rate of arrival.

Among the three approaches, each model has its own advantages and disadvantages depending on the traffic conditions. In general, the 30 second interval order-adaptive approach performs better when cars have heavier traffic, and time-adaptive approach performs extremely well for all circumstances for pedestrians (Appendix B). We speculate that since the pedestrian wait time depends on the light interval, the time-adaptive approach having the option to shorten the interval plays heavily into their favor and therefore greatly shortens the waiting time of pedestrians. The order-adaptive approach performs well on cars. Specifically, when there is medium traffic(40 to 80 cars per minute), it does significantly better than the non-adaptive and time-adaptive cases. This is probably due to the fact that it prioritizes longer lines, and switches to the corresponding light, which in turn achieves an overall reduction of average wait time for

cars. It is also important to note that when the car rate is around 90 to 100 per minute, the wait time of cars is over 200 seconds for all the algorithms, which indicates that when there is too much traffic, the traffic jam and therefore longer wait time is unavoidable. However, we should remember that these are the results we got under our assumptions about the environment, which includes the penalty we add for switching lights as well as the car speed we use to track car traffic. If a different set of assumptions is used, such as having a heavier penalty for switching lights often, we could get different results that may indicate a longer time interval is advantageous.

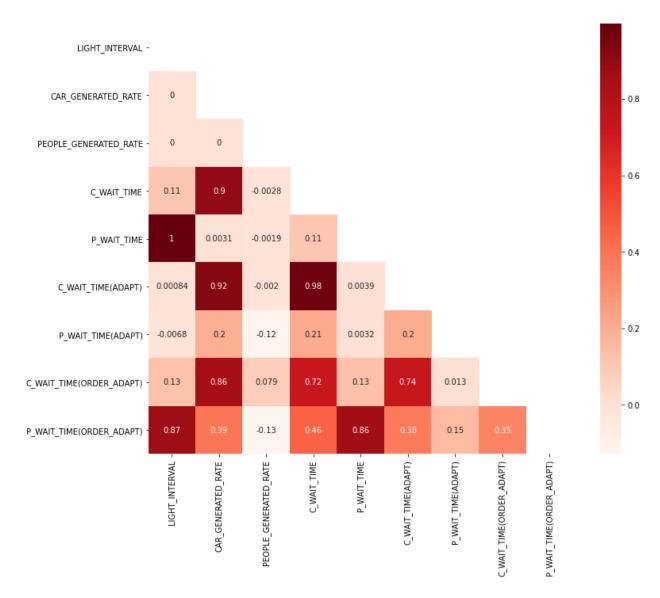
For future work, we can develop a new adaptive algorithm that incorporates both of the ones we have developed. As we have analyzed above, the two adaptive algorithms work well in different scenarios and have different advantages. We could combine the two and have an algorithm that can adapt in both ways, which could potentially reduce the wait time for both pedestrians and cars in a meaningful way. Also as mentioned before, we could do more rigorous testing and find a good conversion factor to take into account the speed of cars so that the car count can be comparable to number of pedestrians when we work on adaptive algorithms.

In addition, we can build a more complicated model by adding additional features. We can change some existing fixed features into hyperparameters, such as the penalty for switching lights too quickly, the constant used to balance the pedestrian and car buildup in time adaptive approach, and the travel speed of cars. Some of the ones we have considered include adding left and right turns for cars, which would not only interact with the pedestrians crossing, but also potentially hold up traffic behind the turning cars. Also, if we look at the Murray & Forbes intersection, which has a bus stop, we can take that into account and have buses that have to stop there regardless of the traffic light, which could hold up traffic and people who come off of the

bus could add to those that are waiting at the intersection to cross. To incorporate these features, we can continue to use the framework we have now, but add an attribute to class Car to indicate which way it is going, and have additional conditions and variables that simulate the more complicated traffic interaction.

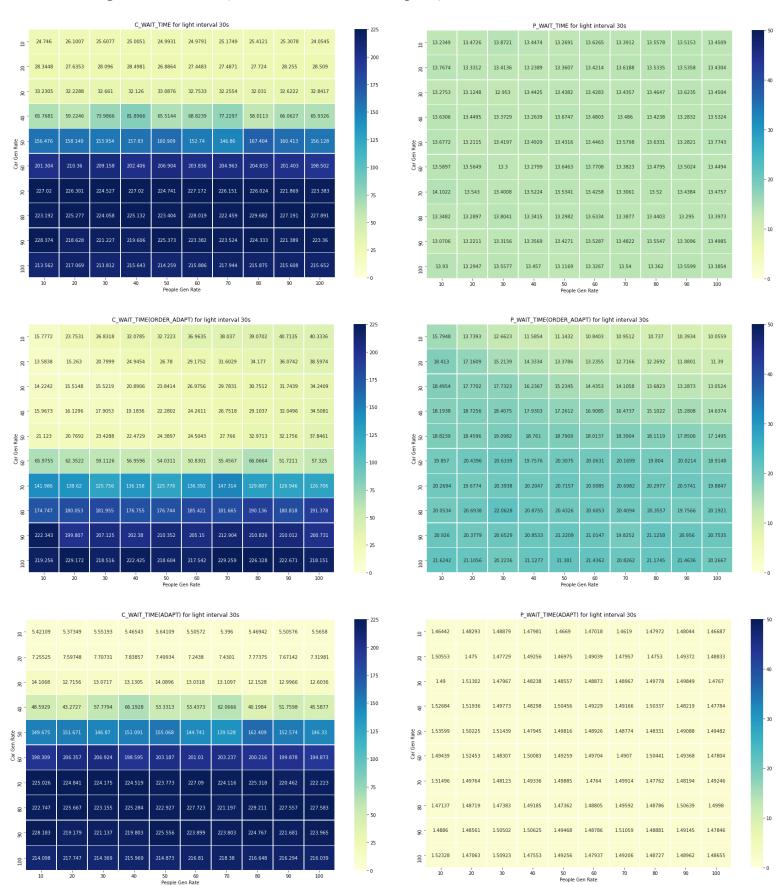
Appendix A

Correlation Graph



Appendix B

Light Interval = 30 (for normal and order adaptive)



Light Interval = 60(for normal and order adaptive)

	C_WAIT_TIME for light interval 60s															P WAIT	_TIME for lig	nht interva	l 60s				
o -	43.1987	43.5913	44.8707	44.4152	43.8072	45.1584	43.976	45.2719	43.0453	45.608	225 g -	- 19	.3269	19.8872	19.6541	19.6326	19.8501	19.915	19.5146	19.7956	20.0008	19.893	- 40
20 1	49.5554	50.5728	49.4114	47.2898	48.0591	49.7513	48.0374	48.9497	49.2659	48.2368	- 200 8 -	- 19	.4676	19.4407	20.2618	20.0851	20.253	19.5698	19.8733	19.7252	19.7504	19.5343	- 35
R -	54.9796	54.4639	55.6922	54.1848	54.4051	55.572	55.0217	55.2415	53.8333	55.1469	- 175 - R	- 19	.3507	20.0609	19.6846	19.5043	20.1728	19.6579	19.9684	19.5407	20.0602	19.7205	- 30
9-	85.5554	94.4027	87.1984	89.6663	87.0112	102.514	87.5278	92.3488	75.0794	87.5967	- 150 \(\frac{1}{2}\)	- 20	1.9907	19.8334	19.8444	19.2945	19.6785	19.9357	19.7708	19.5494	19.3816	19.8574	- 25
Rate 50	168.203	172.356	168.284	159.599	165.984	178.84	174.794	175.849	170.398	162.275	- 125 ag S -	- 19	.6536	20.1458	19.7575	19.9146	19.7805	19.8825	19.8216	19.9127	19.6201	19.9785	25
Car Gen 60	213.892	209.672	206.61	207.505	212.665	209.47	210.06	211.318	210.254	210.466	- 100 G 8 -	- 20	1.1485	19.5617	20.0208	19.4178	19.0291	19.7793	19.6461	19.6159	19.7097	19.7032	- 20
07	226.607	222.942	225.187	226.777	221.195	227.06	222.421	218.852	222.158	224.415	- 75 R -	- 20	.3161	20.0625	19.2165	19.4296	19.5735	20.4584	19.9047	19.9449	19.5596	20.234	- 15
80 -	220.515	221.431	229.475	223.163	222.78	223.608	220.077	226.423	222.701	223.899	- 50	- 20	1.2273	19.3341	20.3585	19.1977	19.4625	20.173	19.7681	19.9296	19.7774	19.4983	- 10
o6 ⁻	219.388	222.264	221.126	216.804	221.849	221.265	221.762	218.422	222.139	217.695	- 25	- 19	.8192	20.1736	19.6506	19.3709	20.1893	19.236	19.8509	19.7478	19.4637	19.9569	- 5
100	215.688	217.069	214.63	216.654	211.677	213.58	214.814	215.012	212.622	210.397	- o			20.0132		20.0447	20.081	19.7318	19.7806	19.7154	19.4818	19.3482	-0
	10	20	30	40	50 People (60 Gen Rate	70	80	90	100			10	20	30	40	50 People Ge	60 n Rate	70	80	90	100	
			C_'	WAIT_TIME(ORDER_AD	APT) for lig	ht interval 6	50s			225				P_	WAIT_TIME	(ORDER_A	DAPT) for li	ght interva	al 60s			
g ⁻	24.5705	36.5191	44.4987	46.9254	58.7392	59.2306	70.7382	71.2857	68.4576	68.489		요 -		20.7826	18.8533	19.1304	17.3885	16.7078	14.7675	15.222	7 15.2899	15.3059	
20 -	21.4301	23.2655	29.9335	37.784	42.5919	46.5427	49.0488	53.4363	56.8322	61.3838		g -	27.1137	26.6855	25.3599	22.6956	21.6517	19.73	19.3823	18.396	2 17.6587	17.0546	
R -	23.5426	22.7337	23.0012	25.4985	31.9028	36.8829	47.2661	48.4835	51.5329	54.7784	- 175	R -		27.9953	27.954	26.9884	25.0926	23.5222	21.572	20.756	20.3023	19.2708	
8 -	26.1134	26.5442	26.9617	26.6959	27.5076	28.7044	37.3312	40.1232	50.5592	57.6159	- 150 S	육 -		27.6127	27.9423	27.6575	28.0606	27.7503	25.801	24.765	3 22.2282	22.0041	
Gen Rate 50	31.0217	33.4729	33.4493	30.9442	32.8763	37.2167	34.8288	33.6329	42.4976	45.0094	- 172 at t	S -	28.2822	28.6446	28.2442	28.1316	27.9165	28.1563	27.4054	27.092	5 26.4082	26.6729	
. 8 c	69.2197	69.1245	60.2282	61.7278	64.4821	54.4254	70.0899	68.768	62.0864	71.5999	-100 💆 8	8 -		29.7026	28.7131	27.9596	29.7132	27.6143	29.2003	27.728	1 28.3518	30.2023	
07	135.399	129.493	144.694	142.444	141.549	145.305	144.049	119.92	128.521	139.241	- 75 §	۶ -	31.3254	32.5039	28.237	30.8552	29.4371	31.4217	29.6173	29.337	9 29.9572	30.6172	
88	173.254	184.253	183.904	179.395	183.286	179.796	178.531	188.18	183.967	181.512	- 50	8 -	32.1584	30.8222	30.6884	31.079	30.5775	31.3895	30.2103				
06	208.558	209.375	208.637	205.439	209.257	213.353	207.179	200.308	210.983	205.784	- 25	8 -	28.9396	28.671	30.2248	30.5112	29.3385	30.976	32.1043		+		
100	222.695	226.972	221.951 30	227.568 40	217.617	215.333	224.215 70	221.704	214.635 90	215.579	-0	100	30.4437	34.2981	31.1829	30.3256 40	29.8139	33.3185	29.511 70	30.711 80	90	31.4553	
		People Gen Rate People Gen Rate P_WAIT_TIME(ADAPT) for light interval 60s																					
	5.46749	5.37285	5.51197) for light in		5.52969	5.53583	5.5005	225	음 -	1.47053	1.46341	1.49508		1.47648	1.48402	147136		2 1.46583	1.47387	
20 10		7.50713	7.81405	7.42401	7.26052	7.19324	7.70651	7.82565	7.49998	7.61006	- 200	g -	1.49817	1.48798	1.49009	1.48771	1.48479	1.47677	1.47692	1.4743	3 1.48671	1.4878	
	13.3858	12.3229	13.5374	12.3231	12.6716	14.2259	13.7971	12.9836	13.513	12.519	- 175 g	R -	1.5133	1.48748	1.50042	1.47895	1.49349	1.48801	1.48225	1.4748	2 1.50187	1.48861	
8 -	51.607	56.0876	54.2693	51.9288	54.8102	68.169	51.0129	58.948	36.6361	52.3267	- 150	8 -	1.47925	1.50359	1.47935	1.49063	1.5014	1.49975	1.48932	1.4871	1 1.47525	1.49387	
Rate 50 4	150.106	156.179	154.628	143.724	151.801	161.657	158.17	157.03	153.763	150.09		os -	1.47137	1.54127	1.50094	1.49672	1.4905	1.48437	1.48707	1.4896	1.49732	1.49421	
Car Gen R	207.691	200.777	200.312	205.064	205.407	201.589	204.952	204.049	203.709	205.17	- 100	8 -	1.4888	1.48889	1.494	1.49098	1.51547	1.46614	1.48867	1.4871	1.49411	1.49799	
07	224.968	221.492	224.735	225.439	221.126	224.8	219.11	216.558	221.696	221.553	- 75	۶-	1.52712	1.49658	1.48538	1.48461	1.4881	1.50854	1.49094	1.4958	1.49926	1.48991	
98 -	220.8	220.694	229.238	222.664	222.439	223.839	220.287	225.391	222.036	224.356		8 -	1.47555	1.49366	1.47764	1.47716	1.49414	1.49538	1.49149	1.4848	1.49489	149193	
o -	219.658	223.28	223.358	217.951	223.183	222.657	223.164	218.597	222.611	218.537	8	g -	1.46435	1.50398	1.49672	1.47719	1.47609	1.49371	1.47715	1.4949	1.48589	1.50948	
100	217.359	218.48	216.552	217.944	212.849	215.616	216.033	217.295	214.904	212.316		100	1.47546	1.48481	1.50555	1.49075	1.4893	1.48127	1.47532				
	10	20	30	40	50 People (60 Gen Rate	70	80	90	100	-0		10	20	30	40	50 People	60 Gen Rate	70	80	90	100	

Light Interval = 90(for normal and order adaptive)

	C_WAIT_TIME for light interval 90s									225				P_WA	IT_TIME for	light interv	al 90s				_	- 40		
8.	65.4944	64.1098	65.1456	62.685	62.0881	66.5791	65.8972	65.4984	63.9847	64.6434		25	- 25.0423	25.4684	25.1656	25.4149	25.3551		25.2527	25.7394	25.7239	25.6062		
92	72.1853	71.0797	74.2882	69.6523	70.8748	71.1411	70.9416	68.5046	71.9919	71.4934		200	- 25.9113	25.2413	25.1097	25.5198	25.6754	24.5955	26.0122	25.3627	25.6785	25.3515		- 35
Я.	79.8041	78.0354	78.9896	82.3384	79.2374	80.0771	80.0944	77.48	79.7693	80.2977	-	175 R	- 25.4202	25.3126	24.8122	25.1185	25.2143	25.7972	25.5884	25.6407	25.313	25.4886		- 30
8	122.091	109.632	119.212	104.17	117.071	113.323	110.006	112.454	111.881	109.403	-	120 8	- 25.6564	24.9732	26.1138	25.5573	24.5793	25.6306	25.2885	25.5103	26.1841	25.5776		- 25
Rate 50	188.434	185.489	180.827	188.973	179.654	193.302	189.265	186.685	179.269	191.611	-	125 gt 221	- 26.2906	25.5123	25.5195	26.1864	25.657	25.8031	25.3923	24.9574	25.4181	25.4893		
Car Gen 60	218.976	224.495	218.313	220.591	228.922	223.557	215.728	217.754	226.131	229.184		100 G	- 24.2875	26.5621	25.6116	25.4034	25.0305	25.3572	25.9905	24.669	24.6901	25.6861		- 20
8	232.421	234.988	228.266	234.36	230.563	232.108	234.903	234.705	231.434	233.221	-	75 R	- 25.5966	24.8649	25.2434	25.5111	25.4298	25.6094	25.3626	25.4788	25.2301	25.3119		- 15
8	230.617	236.25	230.159	229.643	237.087	233.518	231.881	228.64	232.462	233.466	_	50	- 24.9543	26.4972	25.1004	25.4114	25.4447	25.2786	25.3083	25.6611	25.1011	25.4819		- 10
8.	230.973	230.529	228.726	230.171	229.709	226.165	228.38	229.061	230.145	223.995	_	S 25	- 26.0182	25.5923	25.5992	25.4283	25.5894	24.9971	25.1347	25.177	25.1508	25.6614		- 5
100	224.854	222.969	220.649	223.791	223.936	220.09	222.22	225.451	219.126	218.217		100		24.8092	25.5979	25.5659	25.6701	24.7339	25.805	25.2789	25.3885	25.3261		-0
	10	20	30	40	50 People (60 Gen Rate	70	80	90	100	-	0	10	20	30	40	50 People (60 Gen Rate	70	80	90	100		-0
01	30.455	48.2421	C_1 60.6373	WAIT_TIME(63.2976	ORDER_AD. 78.03	APT) for ligi 86.2919	95.6406	96.9861	100.685	100.823		225	- 33.6893	28.6727	P_ 26.7582	WAIT_TIME(24.3946	ORDER_AD 22.6409	APT) for ligi 21.4261	nt interval 9	20.1884	19.7807	19.9279		- 40
20 1	29.7373	29.8116	37.4235	44.1924	60.3636	64.9049	69.2834	76.6779	76.0258	80.0205	-	200		36.0854	34.4286	31.2726	27.4412	26.1062	25.881	23.9445	23.5281	23.1869		- 35
8 ·	33.4361	33.0081	33.0464	33.0327	33.6633	45.307	55.1688	65.6099	71.0284	77.89	-	175		36.9485	36.8248	36.0859	36.2973	33.3489	30.9078	27.9233	26.5867	25.3655		- 30
8.	36.75	37.2498	35.6411	37.0535	37.0914	36.5929	41.2993	47.7897	57.472	67.6682	-	150		36.1173	37.5291	36.9153	36.1214	37	35.9828	33.953	32.2507	29.5147		
Rate 50	43.6067	42.8231	42.7769	43.4684	43.1974	43.2916	47.2296	46.1481	54.1454	57.1514	-	125 gg S	- 36.1758	36.7162	36.4296	36.7082	36.6049	36.4327	35.5885	35.9079	36.0166	35.3828		- 25
Car Gen Ra	65.1127	80.1115	75.0548	77.0552	78.5261	72.8569	72.6875	67.605	76.209	80.9178	-	S Car Gen Ri		39.5736	35.9708	38.4398	37.2643	37.624	37.1443	37.7342	36.1966	36.4452		- 20
02	139.432	137.679	144.085	137.122	144.048	137.519	148.268	144.431	143.113	138.437		75		36.6677	39.3776	37.7996	38.021	40.2407	37.0873	37.3477	37.2703	37.9307		- 15
88	183.562	182.11	178.563	179.304	192.164	183.565	179.526	182.213	180.837	192.732		9	- 41.2365	41.8562	40.2958	37.6937	38.2951	41.0528	39.7912	39.9288	38.547	41.8896		- 10
8.	210.186	215.088	204.512	207.082	210.481	207.873	207.675	211.221	215.351	204.149		50 %	- 38.8544	37.3163	38.9188	43.4255	39.8795	39.3063	41.9369	40.4116	38.9551	39.4521		- 5
100	230.386	220.346	220.389	224.916	226.809	211.901	221.331	231.923	221.708	220.437		25	- 38.285	41.2407	38.7517	41.4002	40.3977	37.7052	43.49	41.1535	43.5542	40.5779		- 3
	10	20	30	40	50 People (60 Gen Rate	70	80	90	100			10	20	30	40	50 People	60 Gen Rate	70	80	90	100		-0
	respire doct-mate																							
					IME(ADAPT							225					IME(ADAPT							40
	5.66287	5.39759	5.50666	5.50813	5.61272	5.46015	5.69022	5.49677	5.566	5.32613	-	200	- 1.45334	1.45692	1.48178	1.48518	1.48722	1.46846	1.48943	1.49645	1.46278	1.4747		- 35
20	7.51202	7.29637	7.80232	7.24999	7.55003	7.48173	7.30511	7.02408	7.26259	7.5503	-	175	- 1.50233	1.49588	1.50571	1.48509	1.50722	1.48668	1.48366	1.47813	1.47843	1.4833		
8	13.836	12.9853	14.265	15.3961	12.6721	12.2767	14.4476	12.2332	13.1536	13.5093			- 1.47783	1.50563	1.51648	1.48188	1.50971	1.48379	1.46859	1.4996	1.47941	1.48047		- 30
8.	63.7684	53.0535	55.5541	46.7572	60.6588	61.3192	52.0323	54.0054	54.2947	52.5926		4	- 1.51275	1.46903	1.47891	1.49187	1.50612	1.49092	1.49269	1.47953	1.50427	1.48664		- 25
r Gen Rate 50		152.85	151.784	157.643	147.812	159.862	155.879	157.984	150.559	155.445		Gen	- 1.48292	1.48584	1.48403	147744	1.49106	1.50416	1.47963	1.51135	1.49804	1.4784		- 20
. 8		205.592	202.069	202.964	209.874	205.055	201.877	199.287	209.094	211.43			- 1.49249	1.47724	1.47193	1.48778	1.50314	1.48714	1.48761	1.4931	1.49043	1.48659		- 15
0.		222.381	215.979	221.891	218.927	219.648	223.54	225.916	218.651	221.044	-	75 R	- 1.48704	1.4974	1.49486	1.49181	1.49804	1.49646	1.50228	1.49452	1.48419	1.49048		
8		225.981	222.383	223.256	227.058	224.409	221.068	218.491	223.713	225.436		50 8		1.51938	1.47689	1.49882	1.4879	1.50088	1.47954	1.48655	1.48299	1.49964		- 10
8.		223.871	222.45	222.754	223.903	218.197	220.83	221.916	223.288	218.096	-	25	- 1.48874	1.512	1.48844	1.50002	1.48858	1.47955	1.50215	1.48016	1.48377	1.49613		- 5
100	219.158	217.001	216.405	218.367	217.614	214.625	217.039 70	218.462 80	213.325 90	212.032	_		1.49551	1.49857 20	1.49264 30	1.47903	1.48153 50	1.48333	1.48764	1.4818	1.47752 90	1.49972		-0
					People (Gen Rate												Gen Rate						

Interval = 120 (for normal and order adaptive)

		C_WAIT_TIME for light interval 120s												P_WAIT_TIME for light interval 120s										 - 40
or .	74.6605	77.218	80.5084	77.0526	74.3956	76.1322	78.0081	77.0354	78.8629	80.0321				_음 - 31.658	9 32.8763	31.9375	32.586	32.6151	32.2528	31.5716	32.3242	32.4503	32.3851	
8 -	79.4398	83.7601	85.774	82.9	85.3581	84.8968	83.6311	86.0736	85.1203	82.1124		- 200		R - 32.71	33.1655	33.0975	32.2659	33.1082	32.7243	32.1398	32.46	32.9503	32.4941	- 35
R -	90.3655	93.904	90.9598	93.971	94.1851	91.0994	92.344	96.5076	92.4051	93.0303		- 175		_유 - 32.29	3 31.6682	32.2973	32.9013	32.458	33.3518	33.0105	32.6441	32.7889	32.0322	- 30
9 .	119.693	117.106	123.796	118.215	128.969	131.516	118.259	115.793	126.282	119.615		- 150		_육 - 32.129	2 32.0988	31.9312	32.626	33.0061	32.3386	32.3199	32.6571	32.5999	32.7682	- 25
So .	187.295	181.393	176.943	167.405	176.693	174.651	168.264	179.114	176.623	178.135		- 125	Rate	ලු - 33.058	6 31.8905	34.4289	32.0919	32.6936	32.5978	32.865	32.8677	32.8184	32.7212	
Car Gen 60	204.241	203.465	209.161	202.4	202.386	204.862	208.03	201.907	208.907	205.304		- 100	Car Gen		32.1963	32.5545	32.4438	32.516	32.0897	32.4987	32.8002	32.4668	32.8554	- 20
8.	213.979	211.48	216.973	212.271	210.899	210.632	212.061	211.687	215.149	210.234		- 75		g - 33.088	6 32.66	31.8684	32.457	33.3432	32.8456	33.3334	32.9196	32.4923	32.5079	- 15
8 -	217.079	214.318	211.834	213.147	217.676	211.933	213.018	213.04	211.858	210.418		- 50		8 - 31.777	6 32.3707	33.5118	31.6841	32.9074	32.2983	32.2302	32.3483	33.1604	32.4386	- 10
8.	210.411	208.428	210.713	208.279	209.166	210.293	207.244	208.122	205.531	208.972		- 25		g - 32.839	9 32.434	32.6474	33.3324	32.6315	32.6782	32.4471	32.3912	33.0569	32.0897	- 5
100	201.425	199.77	202.313	198.442	202.127	201.822	201.965	203.223	205.845	200.278				00 - 35.396	4 32.4738	31.4916	32.1224	32.9945	33.3767	32.5742	32.9442	32.8181	33.0361	
	10	20	30	40	50 People (60 Gen Rate	70	80	90	100		-0		10	20	30	40	50 People	60 Gen Rate	70	80	90	100	-0
			C_V	VAIT_TIME(APT) for ligh	t interval 1	20s				225				P_	WAIT_TIME			ht interval :	120s			 - 40
g ·	34.1226	65.9022	73.9965	81.3987	97.9865	110.112	124.393	122.136	128.482	129.999				g - 44.705	1 36.2824	32.6341	30.1036	28.3787	26.2855	23.9985	23.1208	22.8767	23.5324	
20	36.6715	36.3387	42.6993	55.1343	67.6991	85.3172	84.8997	94.0366	95.0642	95.3178		- 200		R - 46.220	2 45.4615	43.1707	40.3916	36.9302	31.7366	31.4416	29.3915	29.3897	29.4979	- 35
R -	41.2743	41.9645	40.565	40.9561	42.4071	53.7032	72.7398	75.3562	86.1168	94.0724		- 175		유 - 44.640	3 45.9528	44.8518	44.7088	45.5165	42.1319	37.1889	35.0114	32.5449	31.0135	- 30
8 .	46.3474	45.9142	46.4314	46.9413	46.4901	48.3583	51.0762	62.1123	64.2743	84.1287		- 150		Q - 44.949	4 43.5277	44.6627	45.6443	45.4263	45.4154	43.1933	41.9102	41.3762	37.5083	- 25
n Rate 50	56.2854	55.7919	53.5517	52.5062	53.0357	53.7674	52.648	55.9933	52.342	56.9791		- 125	n Rate	ලු - 45.693	7 44.2454	46.4833	45.6149	44.5343	45.1231	45.2572	45.314	45.5074	44.372	- 20
Car Gen 60	81.3054	72.0345	89.3585	77.4633	79.074	87.4823	79.5015	76.3521	88.7094	84.3198		- 100	Car Gen	B - 45.862	5 44.2978	45.4338	44.5484	44.3273	45.2426	44.8926	45.2697	44.8988	45.0895	20
ο.	143.508	144.108	149.973	153.799	138.191	145.372	148.773	136.992	153.251	141.886		- 75		g - 46.23	5 45.3715	48.067	44.7303	46.6884	49.1614	45.6635	44.4781	45.5513	49.8155	- 15
8 -	190.555	195.472	183.728	188.134	196.706	194.36	189.004	200.088	191.272	186.641		- 50		g - 46.514	9 45.2967	47.8867	46.0338	47.5827	45.0553	47.2846	43.2285	47.5074	44.494	- 10
8 -	216.315	213.504	218.163	212.373	219.587	219.257	209.179	212.485	208.404	214.552		- 25		g - 54.347	1 46.0399	46.6247	46.9931	49.6029	49.287	48.4987	50.9137	50.7171	44.6872	- 5
100	229.335	223.481	226.185	221.432	223.516	225.452	231.238	227.023	229.939	221.93		-0		00 - 49.300		50.0388	53.0749	49.1029	52.7812	50.2308	50.2867	48.9019	46.6165	-0
	10	20	30	40	50 People (60 Gen Rate	70	80	90	100				10	20	30	40	50 People	60 Gen Rate	70	80	90	100	Ü
				C_WAIT_TI	ME(ADAPT)	for light in	terval 120s					- 225					P_WAIT_TIM	E(ADAPT) f	or light inte	rval 120s				40
01	5.67948	5.55501	5.41982	5.60052	5.44892	5.32375	5.57432	5.5187	5.46517	5.5915		- 200		- 1.48531	1.49797	1.47493	1.47161	1.48142	1.46733	1.4872	1.47608	1.4752	1.46568	
20	7.41011	7.4305	7.49254	7.78965	7.4586	7.56492	7.55648	7.33435	7.46507	7.07743				- 1.4906	1.47767	1.52543	1.49579	1.47701	1.49159	1.49956	1.47365	1.47458	1.47191	- 35
R -	12.3866	12.7936	14.0453	12.5579	13.3456	13.7255	13.6236	13.2881	12.4492	13.2157		- 175	æ	- 1.48794	1.48682	1.4925	1.50479	1.50753	1.49831	1.49149	1.47176	1.48413	1.49535	- 30
8 .	55.1089	48.0959	52.2304	46.625	62.0305	64.9673	49.4921	44.8217	53.8055	52.4662		- 150	4	- 1.46638	1.47545	1.47438	1.49425	1.49021	1.49196	1.47588	1.51953	1.48879	1.49144	- 25
Gen Rate 50	158.967	159.213	146.815	136.122	146.357	144.637	139.574	158.307	147.541	152.784		- 125	Gen Rate	- 1.49211	1.50013	1.48535	1.49699	1.4979	1.49302	1.51601	1.48906	1.50482	1.48269	- 20
. 8 c	206.566	196.568	212.045	201.74	202.111	205.09	202.426	202.061	209.765	206.09		- 100		- 1.53186	1.5	1.48566	1.50418	1.47501	1.48668	1.51203	1.49129	1.49314	1.49736	
0,	222.509	216.488	223.652	220.852	219.415	220.631	221.328	220.754	224.831	217.266		- 75	92	- 1.51003	1.47321	1.49939	1.48543	1.4926	1.47483	1.49291	1.48049	1.49822	1.49765	- 15
80	231.301	228.241	222.749	224.55	227.83	224.38	224.941	226.212	225.188	223.252		- 50	8	- 1.47395	1.49477	1.48047	1.47814	1.48812	1.48994	1.48727	1.47098	1.50269	1.49306	- 10
8 -	224.622	223.098	227.172	222.556	223.52	225.009	222.275	223.146	220.163	224.906		- 25	8	- 1.50892	1.4887	1.48698	1.49546	1.48687	1.49944	1.49928	1.49867	1.5017	1.4881	- 5
100	216.772	215.608	216.398	213.075	215.885	217.163	217.059	218.981	221.041	216.73		-0	100	- 1.51565	1.48049	1.47004	1.4758	1.49568	1.48703	1.50224	1.49007	1.48925	1.49097	-0
	10	20	30	40	50 People (60 Gen Rate	70	80	90	100				10	20	30	40	50 People G	60 en Rate	70	80	90	100	