1. Does the distance between the buses keep uniform? If not, what should be done to ensure it is uniform?

A: The distance between buses doesn’t keep uniform, if we decided to allowed buses can exceed former buses, then the distance will keep uniform.

2. What is the average size of a waiting queue at each stop (and what are its maximum and minimum)? (You may provide this information on an hourly [simulation time] base.)

A: My experiment assumption is:

1. 12 stops.
2. 4 buses.
3. driving time between two contiguous time is 5s.
4. person generate rate is 1, means time is 1 * rand(0, 1)
5. boarding time is 0.04s
6. Total running time is 30s.

The average size of each queue is [4.7, 7.1, 5.1, 3.2, 5.0, 7.7,4.1, 4.6,6.2,4.1,7.4, 7.1], based on stop id from 0 ~ 11.

The max size of each queue is [15, 14, 15, 18, 13, 16, 13, 14, 16, 13, 14, 15], the min size of each queue is [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0], which might cause by the person generate time is too long.

Main Code:

1. person event generate, it will push a new person event based on the same stop id to event lists.

```
function person(event){
	//update stop queue
	stopId = event.stop;
	stops[stopId] += 1;

	//gernate new person event
	newEvent = new Event();
	newEvent.time = event.time + Math.random() * passagerRate;
	newEvent.type = 'person';
	newEvent.stop = stopId;

	console.log('New '+ newEvent.type +' event: time:'+ newEvent.time + ' | stop: '+
newEvent.stop);

	return newEvent;
}
```

2. boarder event, it will check stop waiting person number, if exists, it will update stop queue by minus one, and push a new boarding event, if not, it will push arrival event.

```
function boarder(event){
	stopId = event.stop;

	// while loop doing boarder event
	if(stops[stopId]){
		stops[stopId] -= 1;
		//gernate new boarder event
		newEvent = new Event();
		newEvent.time = event.time + boardingTime;
		newEvent.type = 'boarder';
		newEvent.stop = stopId;
		newEvent.busId = event.busId;

		console.log('New '+ newEvent.type +' event: time:'+ newEvent.time + ' |
stop: '+ newEvent.stop + ' | bus id: '+ newEvent.busId);
		return newEvent;
	} else {
		// generate arrival event
		newEvent = new Event();
		newEvent.time = event.time + boardingTime;
		newEvent.type = 'arrival';
		newEvent.stop = (stopId + 1) % stopsNum;
		newEvent.busId = event.busId;

		console.log('New '+ newEvent.type +' event: time:'+ newEvent.time + ' |
stop: '+ newEvent.stop + ' | bus id: '+ newEvent.busId);
		return newEvent;
	}
}
```

3. arrival event, it checks stop person number, if exists, it will generate boarder event, otherwise generate arrival event to next stop

```
function arrival(event){
	stopId = event.stop;

	if(stops[stopId] == 0){
		//gernate new arrival event
		newEvent = new Event();
		newEvent.time = event.time + drivingTime;
		newEvent.type = 'arrival';
		newEvent.stop = (stopId + 1) % stopsNum;
		newEvent.busId = event.busId;

		console.log('New '+ newEvent.type +' event: time:'+ newEvent.time + ' |
stop: '+ newEvent.stop + ' | bus id: '+ newEvent.busId);
		return newEvent;
	} else {
```

```
                //gernate new board event
                newEvent = new Event();
                newEvent.time = event.time + boardingTime;
                newEvent.type = 'boarder';
                newEvent.stop = stopId;
                newEvent.busId = event.busId;

                console.log('New '+ newEvent.type +' event: time:'+ newEvent.time + ' |
stop: '+ newEvent.stop + ' | bus id: '+ newEvent.busId);
                return newEvent;
        }
}
```

Note: All the code is written by JavaScript, the zip file contain main.html file, it allows you run it on any modern browser, and check its “develop tool”, the console.log() info will display in there.