Macroscopic modelling of parking dynamics in urban networks (Part I)

Jin Cao, IVT, ETHZ, Switzerland
Monica Menendez, IVT, ETHZ, Switzerland

Abstract

Parking can have a significant impact on transportation systems. Besides the effects on travel demand, the regulation and usage of urban parking can also directly influence traffic performance. The availability of parking and the demand to park, at any given time, influence the trip completion rate, and correspondingly the traffic conditions in the area.

In our research, we define several vehicle groups, according to their status concerning parking. For example, vehicles in an area can be grouped into "driving", "searching" and "staying (i.e., parking)". To find the quantity of cars within each vehicle group at a future time, we divide the time period into a number of thin time slices, in which the conditions are assumed to be static. Using probability theory, kinematic flow theory and dimensional analysis, a transition matrix can be developed. Based on that, the quantity of cars driving, parking or searching for parking can be updated over time. Therefore, the model enables us to define, simulate, and quantify the dynamics of the parking process. Furthermore, we are able to estimate different traffic parameters / indicators. The ultimate goal is to evaluate the relationship between urban parking and traffic performance in urban networks.

The current study (this paper) is a preliminary part of the research, it quantifies the number of cars that access parking in a time slice analytically (i.e, the number of cars that leave the "searching" group, and join the "staying" group). Numerical examples are given, a program is provided to test the results as well.

Keywords: Urban Parking – Transition Matrix – Traffic

1. General information

Given that this paper is the first part of an ongoing research, we will give a general introduction of the research (part 1.1), then we will introduce the basics of this paper (part 1.2).

1.1 Background

Parking can have a significant impact on the transportation system. Besides the long-term effects on travel demand, the regulation and usage of urban parking can also directly influence traffic performance. For example, in Thompson (1998), based on the review of 16 studies of mostly American and European cities, it was concluded that cars searching for free parking contribute to over 8% of the total traffic in a city, reaching 30% in business areas during rush hour. In Cao (2014), the potential delay caused by on-street parking maneuvers was modeled analytically; showing that when they take place near signalized intersections, the local network can be easily affected. In Montini (2012), GPS data collected in the city of Zurich, showed that a vehicle's speed strongly decreases when approaching the parking space.

In these studies, some "short-term" or "immediate" effects of urban parking on traffic performance are shown. However, to the authors' knowledge, no study has provided yet a generalized methodology to macroscopically model the relation between parking demand, parking availability, and traffic conditions, considering such a temporal scope.

In the research, we develop a parking-state-based transition matrix that aims to model macroscopically a dynamic urban parking system. Basic assumptions for the matrix include a traffic demand over a period of time (e.g., a day), and the distribution of parking durations. The model then provides a continuous approximation of the percentage/number of cars searching for parking, traveling through the system, and staying in parking stalls, as a function of time. These results are useful to estimate both, how traffic performance (e.g., speed, density, flow) affects drivers' ability to find parking; and how parking availability affects traffic performance.

Consider a round-trip going into an urban area as a tour (instead of 2 single trips). The whole tour (within the urban area of interest) can then be divided into four parking-related states separated by five parking-related events (see Figure 1). Notice that we assume the parking maneuvers (access / depart) are instantaneous although in reality they are not.

Figure 1. Parking-related states and parking-related events for a given vehicle.

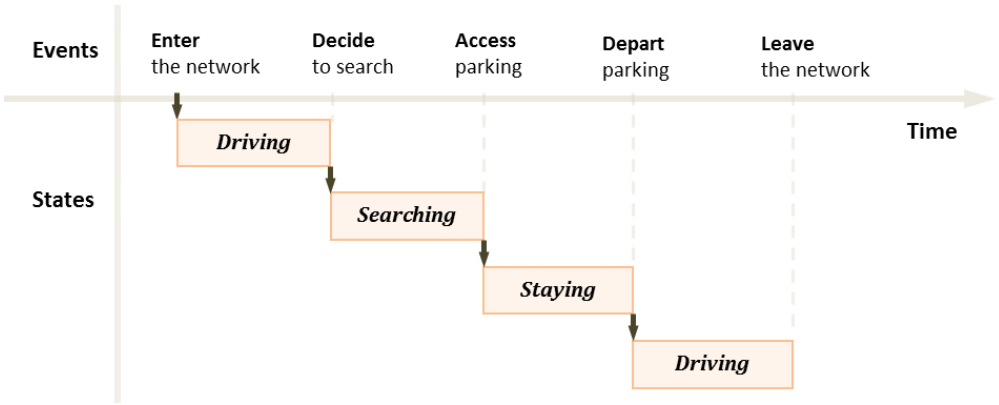
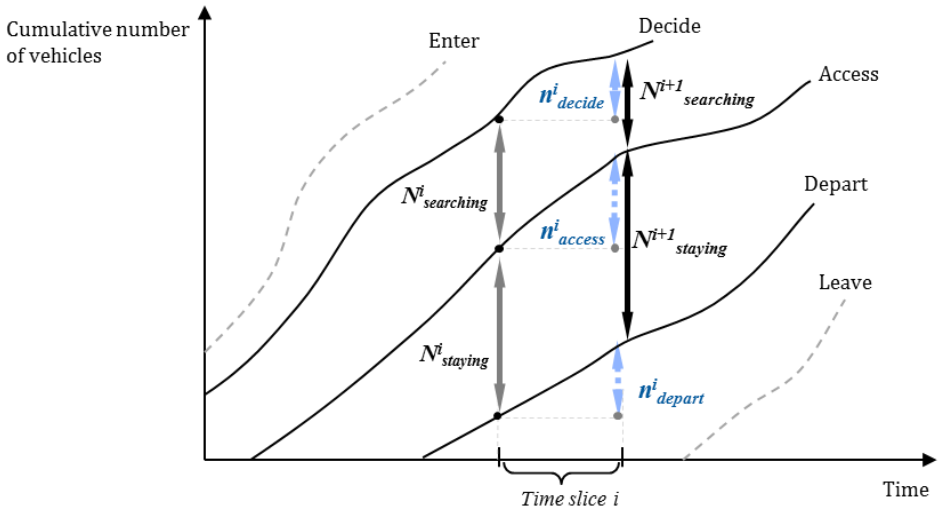


Figure 2 illustrates the cumulative number of vehicles that have experienced each parking event as a function of time. The vertical distance between the curves from two consecutive events, indicates the number of vehicles in the in-between parking state at any given time, e.g., the vertical distance at the beginning of time slice *i* between the curves "Decide" and "Access" indicates the number of vehicles searching for parking at that specific time. Similarly, the average horizontal distance between two consecutive curves indicates the average time spent in the in-between parking state.

Figure 2. Cumulative number of vehicles that have experienced each parking event.



Note that the dotted line in Figure 2, "enter" and "leave" indicate the cars which enter and leave the area. The number of vehicles going through these two events are obtained based on assumptions on travel demand (i.e., input to the model). For a time slice (e.g., one minute), each state of the system can be updated based on:

- $\boldsymbol{n_{decide}}$: number of new vehicles that start searching for parking during the current time slice.
- $\boldsymbol{n_{access}}$: number of vehicles finding/accessing a parking stall during the current time slice.
- $\boldsymbol{n_{depart}}$: number of vehicles departing a parking stall during the current time slice.
- $\boldsymbol{N_{searching}}$: number of vehicles searching for parking at the beginning of the current time slice.
- $\boldsymbol{N_{staying}}$: number of vehicles staying in a parking stall at the beginning of the current time slice (i.e., parking stalls used). Since $\boldsymbol{N_{staying}} + \boldsymbol{N_{vacant}}$ represents the total number of available parking stalls, $\boldsymbol{N_{vacant}}$ can be easily found at the beginning of each time slice.

In this paper, we are trying to provide an analytical estimation of n_{access}.

1.2 Introduction

In this paper, an analytical and also an experimental (program) estimation of n_{access} are provided. It is not only the number of cars accessing parking (finding parking) during a time slice, but also the number of parking spots being newly occupied during this period. For simplification purposes, we use n instead of n_{access} to represent this value.

In the real world, the road network and distribution of parking facilities can be very detailed and different from one place to another. Therefore, in our model, we focus on the averaged values based on generalized conditions. In this way, the model is kept simple but yet valid enough to represent the basics of urban parking systems and the parking search process. Here we mention the basics of our assumptions for the model, details can be found in the next section.

Imagine the network of interest as a ring road with cars traveling in a single direction. Vehicles entering the network appear uniformly distributed along the ring. The overall parking supply can be (or not) uniformly distributed along the ring; but in each time slice (i.e., in the time period of interest), the vacant parking stalls appear randomly on the road. Vehicles drive on the ring road searching for vacant parking stalls, and access the first one they see. These assumptions allow us to imitate a practical situation with the imbalance between parking availability and demand, as well as the parking search phenomenon. But, as mentioned before, the model neglects the influence of the network shape (by assuming all streets have the same likelihood of being visited), and personal requirements for parking.

2. Model

2.1 Assumptions

The urban network is abstracted as one ring road with cars driving in a single direction. Here are some basic assumptions:

1. At the beginning of a time slice, new searching vehicles appear on the road uniformly, (this constraint can be relaxed in future extensions of this model) while the parking spots appear randomly (notice that the distribution of the parking spots follows a uniform distribution in a longer term).
2. All vehicles drive at the same speed while looking for available parking.
3. A parking spot might be visited by several vehicles, but it can only accommodate the first one that passes by. The others will see it occupied, then continue searching for the next available parking spot.

Figure 3 shows two examples of the situation: (a) 4 cars are searching while only 2 parking spots are available; (b) 4 cars are searching while 5 parking spots are available. Evidently, the results depend on, the number of searchers and parking spots, and the distance a car can reach within a given time slice.

Figure 3. Examples illustrating possible combinations of both the quantity and the location of parking searchers and the available parking spots.

(a) (b)

Through probability theory, the number of vehicles successfully accessing parking spots, n, can be found.

2.2 Variables

Here is a list of all variables that will be used in the model to estimate n.

Table 1. A list of all variables with explanation

v	the average travel speed is v.
L	the size of the network is L (the total length of the ring road). For example, the probability of a parking being generated within $[0, x]$ is $\int_0^x \frac{1}{L} dx$.

N	at the beginning of the period, there are N cars searching for parking.
A	at the beginning of the period, a number of A parking spots are available.
t	the length of the time slice is t.
s	the spacing between each two cars is $s = \frac{L}{N}$.
d	the maximum driving distance of each car is $d = vt$.
m	$m = min\{\lfloor\frac{d}{s}\rfloor + 1, N\}$. A place on the network can be visited by a maximum of m cars. In other words, a parking spot can be visited by a maximum of m cars, but still, it may not be taken by any (happens if all the cars parked before reaching that given spot).
p'	the probability of any specific parking spot being taken when it is located within $[0, s]$.
$\bar{p}$	the average probability of a parking spot being taken, $\bar{p} = N \cdot p'$.

Evidently, $n \leq min\{A, m\}$.

2.3 Results

Denote $\bar{p}$ as the probability for a specific parking spot been taken by any car during the period, regardless of what happens to the other parking spots. As $\bar{p}$ applies to any parking spot, $n = A \cdot \bar{p}$.

A parking spot can be located within any of the spacing between two cars, e.g., $[0, s]$, $[s, 2s]$ …$[(N-1)s, Ns]$, the probability of it being taken is the same no matter which range it is located. Denote p' as this probability, we can see it as the chance of any specific parking spot being taken when it is located within $[0, s]$, then $\bar{p} = N \cdot p'$.

Therefore, $n = A \cdot N \cdot p'$. The results can be drawn depending on three different scenarios, they are described below.

Scenario 1: $d \in [0, s]$

Under this scenario, a parking spot can be visited by a maximum number of m=1 car. Therefore, the probability of a parking spot being taken when it is located within $[0, s]$, p', is based on two conditions:

Condition 1. the parking spot can be reached by the car. Assume the location of the parking spot as x, then this part is the probability of $x \in [0, d]$, can be written as $\int_0^d \frac{1}{L} dx$.

Condition 2. within the rest $A-1$ parking spots, all of them are further away than this one (located at x) from the initial location of the car, so that the car will park at x as it is the first parking it sees. This probability can be written as $\left(\int_x^L \frac{1}{L} dx\right)^{A-1}$.

Therefore, $p' = \int_0^d \frac{1}{L}\left(\int_x^L \frac{1}{L} dx\right)^{A-1} dx$ and $n = ANp' = A \cdot \left\{N \cdot \left[\int_0^d \frac{1}{L}\left(\int_x^L \frac{1}{L} dx\right)^{A-1} dx\right]\right\}$, a simplified equation can be seen in Eq. 1(a).

Scenario 2: $d \in (s, L)$.

Under this condition, the probability of the parking spot to be taken, p', can be summarized by two separate parts:

- p_1: the probability of the parking spot to be taken by the car if it is located within $[0, d-(m-1)s]$. In this scenario, a maximum number of m cars can reach the parking spot (assume they did not park before then).
  - If $A \leq m$, the number of the rest of the parking spots (i.e., $A-1$) can accommodate at most $m-1$ cars. In other words, at least one car still did not park before it arrives at location x. Therefore, the probability of the parking spot (at location x) is
    $p_1 = \int_0^{d-(m-1)s} \frac{1}{L} dx$.
  - If $A > m$, there are more parking spot available on the network than cars, then the probability of the parking spots being taken equals to one minus "the chance that all the cars parked before they arrived at location x". Therefore, the probability of the parking spot (at location x) being taken is

$$p_1 = \int_0^{d-(m-1)s} \frac{1}{L} \left\{ 1 - \left\{ \begin{array}{c} \sum_{i_m=m}^{A-1} C_{A-1}^{i_m} \left(\int_{-(m-1)s}^{x} \frac{1}{L} dx\right)^{i_m} \cdot \left(1 - \int_{-(m-1)s}^{x} \frac{1}{L} dx\right)^{A-1-i_m} \cdot \\ \left\{ \sum_{i_{m-1}=m-1}^{i_m} C_{i_m}^{i_{m-1}} \left(\frac{\int_{-(m-2)s}^{x} \frac{1}{L} dx}{\int_{-(m-1)s}^{x} \frac{1}{L} dx}\right)^{i_{m-1}} \cdot \left(1 - \frac{\int_{-(m-2)s}^{x} \frac{1}{L} dx}{\int_{-(m-1)s}^{x} \frac{1}{L} dx}\right)^{i_m - i_{m-1}} \cdot \right. \\ \left. \begin{array}{c} \ldots \cdot \\ \left[\sum_{i_1=1}^{i_2} C_{i_2}^{i_1} \left(\frac{\int_0^{x_1} \frac{1}{L} dx}{\int_{-s}^{x} \frac{1}{L} dx}\right)^{i_1} \cdot \left(1 - \frac{\int_0^{x_1} \frac{1}{L} dx}{\int_{-s}^{x} \frac{1}{L} dx}\right)^{i_2 - i_1}\right] \end{array} \right\} \end{array} \right\} \right\} dx$$

- p_2: the probability of the parking spot to be taken by the car if it is located within $[d-(m-1)s, s]$. In this scenario, a maximum number of $m-1$ cars can reach the parking spot (assume they did not park before then).
  - If $A < m$, there are more parking spots available on the network than cars, then the probability of the parking spot being taken equals to one minus "the chance that all the cars parked before they arrived at location x". Therefore, the probability of the parking spot (at location x) being taken is
    $p_2 = \int_{d-(m-1)s}^{s} \frac{1}{L} dx$.
  - If $A \geq m$, it is possible that all the $m-1$ cars parked before they arrive at location x, and one minus this chance is, then, the probability of the parking spot at x being taken. Therefore, the probability of the parking spot (at location x) being taken is

$$p_2 = \int_{d-(m-1)s}^{s} \frac{1}{L} \left\{ 1 - \left\{ \begin{array}{c} \sum_{i_{m-1}=m-1}^{A-1} C_{A-1}^{i_{m-1}} \left(\int_{-(m-2)s}^{x} \frac{1}{L} dx \right)^{i_{m-1}} \cdot \left(1 - \int_{-(m-2)s}^{x} \frac{1}{L} dx \right)^{A-1-i_{m-1}} \cdot \\ \left\{ \begin{array}{c} \sum_{i_{m-2}=m-2}^{i_{m-1}} C_{i_{m-1}}^{i_{m-2}} \left(\frac{\int_{-(m-3)s}^{x} \frac{1}{L} dx}{\int_{-(m-2)s}^{x} \frac{1}{L} dx} \right)^{i_{m-2}} \cdot \left(1 - \frac{\int_{-(m-3)s}^{x} \frac{1}{L} dx}{\int_{-(m-2)s}^{x} \frac{1}{L} dx} \right)^{i_{m-1}-i_{m-2}} \cdot \\ \dots \cdot \\ \left[\sum_{i_1=1}^{i_2} C_{i_2}^{i_1} \left(\frac{\int_{0}^{x_1} \frac{1}{L} dx}{\int_{-s}^{x} \frac{1}{L} dx} \right)^{i_1} \cdot \left(1 - \frac{\int_{0}^{x_1} \frac{1}{L} dx}{\int_{-s}^{x} \frac{1}{L} dx} \right)^{i_2-i_1} \right] \end{array} \right\} \end{array} \right\} dx$$

As $p' = p_1 + p_2$, $n = ANp' = A \cdot N \cdot (p_1 + p_2) =$

$$\begin{cases} AN \left(\int_0^{d-(m-1)s} \frac{1}{L} dx + \int_{d-(m-1)s}^{s} \frac{1}{L} dx \right), & \text{if } A < m \\ AN \left\{ \begin{array}{l} \int_0^{d-\left\lfloor \frac{d}{s} \right\rfloor s} \frac{1}{L} dx \\ \int_{d-(m-1)s}^{s} \frac{1}{L} \left\{ 1 - \left\{ \begin{array}{c} \sum_{i_{m-1}=m-1}^{A-1} C_{A-1}^{i_{m-1}} \left(\int_{-(m-2)s}^{x} \frac{1}{L} dx \right)^{i_{m-1}} \cdot \left(1 - \int_{-(m-2)s}^{x} \frac{1}{L} dx \right)^{A-1-i_{m-1}} \cdot \\ \left\{ \begin{array}{c} \sum_{i_{m-2}=m-2}^{i_{m-1}} C_{i_{m-1}}^{i_{m-2}} \left(\frac{\int_{-(m-3)s}^{x} \frac{1}{L} dx}{\int_{-(m-2)s}^{x} \frac{1}{L} dx} \right)^{i_{m-2}} \cdot \left(1 - \frac{\int_{-(m-3)s}^{x} \frac{1}{L} dx}{\int_{-(m-2)s}^{x} \frac{1}{L} dx} \right)^{i_{m-1}-i_{m-2}} \cdot \\ \dots \cdot \\ \left[\sum_{i_1=1}^{i_2} C_{i_2}^{i_1} \left(\frac{\int_{0}^{x_1} \frac{1}{L} dx}{\int_{-s}^{x} \frac{1}{L} dx} \right)^{i_1} \cdot \left(1 - \frac{\int_{0}^{x_1} \frac{1}{L} dx}{\int_{-s}^{x} \frac{1}{L} dx} \right)^{i_2-i_1} \right] \end{array} \right\} \end{array} \right\} dx + \end{array} \right\}, & \text{If } A = m \\ AN \left\{ \begin{array}{l} \int_0^{d-(m-1)s} \frac{1}{L} \left\{ 1 - \left\{ \begin{array}{c} \sum_{i_m=m}^{A-1} C_{A-1}^{i_m} \left(\int_{-(m-1)s}^{x} \frac{1}{L} dx \right)^{i_m} \cdot \left(1 - \int_{-(m-1)s}^{x} \frac{1}{L} dx \right)^{A-1-i_m} \cdot \\ \left\{ \begin{array}{c} \sum_{i_{m-1}=m-1}^{i_m} C_{i_m}^{i_{m-1}} \left(\frac{\int_{-(m-2)s}^{x} \frac{1}{L} dx}{\int_{-(m-1)s}^{x} \frac{1}{L} dx} \right)^{i_{m-1}} \cdot \left(1 - \frac{\int_{-(m-2)s}^{x} \frac{1}{L} dx}{\int_{-(m-1)s}^{x} \frac{1}{L} dx} \right)^{i_m-i_{m-1}} \cdot \\ \dots \cdot \\ \left[\sum_{i_1=1}^{i_2} C_{i_2}^{i_1} \left(\frac{\int_{0}^{x_1} \frac{1}{L} dx}{\int_{-s}^{x} \frac{1}{L} dx} \right)^{i_1} \cdot \left(1 - \frac{\int_{0}^{x_1} \frac{1}{L} dx}{\int_{-s}^{x} \frac{1}{L} dx} \right)^{i_2-i_1} \right] \end{array} \right\} \end{array} \right\} dx \\ + \int_{d-(m-1)s}^{s} \frac{1}{L} \left\{ 1 - \left\{ \begin{array}{c} \sum_{i_{m-1}=m-1}^{A-1} C_{A-1}^{i_{m-1}} \left(\int_{-(m-2)s}^{x} \frac{1}{L} dx \right)^{i_{m-1}} \cdot \left(1 - \int_{-(m-2)s}^{x} \frac{1}{L} dx \right)^{A-1-i_{m-1}} \cdot \\ \left\{ \begin{array}{c} \sum_{i_{m-2}=m-2}^{i_{m-1}} C_{i_{m-1}}^{i_{m-2}} \left(\frac{\int_{-(m-3)s}^{x} \frac{1}{L} dx}{\int_{-(m-2)s}^{x} \frac{1}{L} dx} \right)^{i_{m-2}} \cdot \left(1 - \frac{\int_{-(m-3)s}^{x} \frac{1}{L} dx}{\int_{-(m-2)s}^{x} \frac{1}{L} dx} \right)^{i_{m-1}-i_{m-2}} \cdot \\ \dots \cdot \\ \left[\sum_{i_1=1}^{i_2} C_{i_2}^{i_1} \left(\frac{\int_{0}^{x_1} \frac{1}{L} dx}{\int_{-s}^{x} \frac{1}{L} dx} \right)^{i_1} \cdot \left(1 - \frac{\int_{0}^{x_1} \frac{1}{L} dx}{\int_{-s}^{x} \frac{1}{L} dx} \right)^{i_2-i_1} \right] \end{array} \right\} \end{array} \right\} dx \end{array} \right\}, & \text{If A} > m \end{cases}$$

, a simplified equation can be seen in Eq. 1(b).

Scenario 3: d ∈ [L, ∞).

When $d \in [L, \infty)$, each car can drive virtually around the whole network at least once, and the result is $n = \min\{A, N\}$. The equation can be seen in Eq. 1(c).

Simplified Equations/Results

The results according to the three scenarios are summarized as below. The equations are simplified, therefore it might be harder to understand, but they are meant to be used/ calculated more easily.

Scenario 1: when $d \in [0, s]$

$$n = N \cdot \left[1 - \left(1 - \frac{d}{L}\right)^A\right] \quad \text{Eq 1}$$

Scenario 2: when $d \in (s, L)$

- 2(a). $if\ A < m$

$$n = A \quad \text{Eq 2(a)}$$

- 2(b). if A = m

$$n = A - \frac{AN}{L^A} \cdot \left(\sum_{\substack{i_{m-1} \\ =m-1}}^{A-1} C_{A-1}^{i_{m-1}} \cdot \sum_{\substack{i_{m-2} \\ =m-2}}^{i_{m-1}} C_{i_{m-1}}^{i_{m-2}} \cdot \ldots \cdot \sum_{i_1=1}^{i_2} C_{i_2}^{i_1} \right) \cdot s^{(i_{m-1} - i_1)} \cdot \int_{d-(m-1)s}^{s} [(N-m+2)s - x]^{A-1-i_{m-1}} \cdot x^{i_1} dx \quad \text{Eq 2(b)}$$

- 2(c). if A > m

$$n = A - \frac{AN}{L^A} \cdot \left\{ \begin{array}{l} \left(\sum_{i_m=m}^{A-1} C_{A-1}^{i_m} \cdot \sum_{\substack{i_{m-1} \\ =m-1}}^{i_m} C_{i_m}^{i_{m-1}} \ldots \sum_{i_1=1}^{i_2} C_{i_2}^{i_1} \right) \cdot s^{(i_m - i_1)} \cdot \int_0^{d-(m-1)s} [(N-m+1)s - x]^{A-1-i_m} \cdot x^{i_1} dx + \\ \left(\sum_{\substack{i_{m-1} \\ =m-1}}^{A-1} C_{A-1}^{i_{m-1}} \cdot \sum_{\substack{i_{m-2} \\ =m-2}}^{i_{m-1}} C_{i_{m-1}}^{i_{m-2}} \cdot \ldots \cdot \sum_{i_1=1}^{i_2} C_{i_2}^{i_1} \right) \cdot s^{(i_{m-1} - i_1)} \cdot \int_{d-(m-1)s}^{s} [(N-m+2)s - x]^{A-1-i_{m-1}} \cdot x^{i_1} dx \end{array} \right\} \quad \text{Eq 2(c)}$$

Scenario 2: when $d \in [L, \infty)$

$$n = min\{A, N\} \quad \text{Eq 3}$$

2.4 Numerical Examples

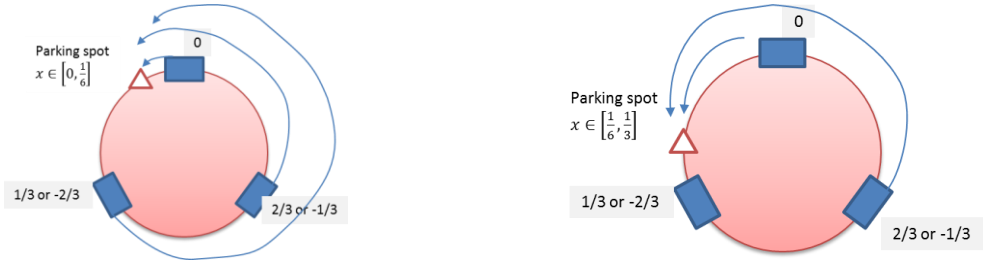
Scenario 1: $d \in [0, s]$.

Assume the length of the ring road is L=1km; a number of A=4 parking spots are available; a number of N=3 cars are searching for parking spots, the spacing between two cars is s=1/3 km; each car can drive a maximum distance of d =1/5 km. Then $n = 4 \cdot \left\{3 \cdot \left[\int_0^{0.2} \frac{1}{1} dx \left(\int_x^1 \frac{1}{1}\right)^3\right]\right\} = 1.77$. During the period, an average value of 1.77 cars out of 3 find parking, 1.77 parking spots out of 4 are newly occupied.

Scenario 2: $d \in (s, L)$.

Assume the same values as before (i.e., L=1km; A=4; N=3; s=1/3km) but a different value of d=5/6 km, $A > m = 3$. In this case, within each spacing of distance s, the parking spot located within $[0, d + (m-1) \cdot s]$ can be reached by m=3 cars virtually, the parking spot located within $[d + (m-1) \cdot s, s]$ can be reached by m-1=2 cars virtually. p_1 and p_2 are analyzed below.

Figure 4. Examples supporting the illustration of p_1 and p_2.



(a) if $x \in [0, \frac{1}{6}]$, a number of m=3 cars can reach x. (b) if $x \in [\frac{1}{6}, \frac{1}{3}]$, a number of m-1=2 cars can reach x.

$\boldsymbol{p_1}$ represents the probability of the parking spot (which locates at $x \in [0, \frac{1}{6}]$) being taken, given that m=3 cars can actually pass by the location, they start at location $-\frac{2}{3}$, $-\frac{1}{3}$ and 0 respectively. It can be seen in the equation below:

- Condition 1 represents the probability that, within the rest of the $A-1$ available parking spots, a number of $i_3 \in [3, A-1]$ spots exist between $-\frac{2}{3}$ and x. Therefore, the car that started from $-\frac{2}{3}$ can be accommodated before it reaches x;
- Condition 2 represents the probability that, within these i_3 parking spots mentioned in condition 1, a number of $i_2 \in [2, i_3]$ spots are actually located between $-\frac{1}{3}$ and x. Therefore, the car that started from $-\frac{1}{3}$ can be accommodated before it reaches x;

- Condition 3 represents the probability that, within these i_2 parking spots mentioned in condition 2, a number of $i_1 \in [1, i_2]$ spots are actually located between 0 and x. Therefore, the car that started from 0 can be accommodated before it reaches x.

$$p_1 = \int_0^{\frac{1}{6}} \frac{1}{1} \left\{ 1 - \underbrace{\left\{ \overbrace{\sum_{i_3=3}^{3} C_3^{i_3} \left(\int_{-\frac{2}{3}}^{x} \frac{1}{1} dx \right)^{i_3} \cdot \left(1 - \int_{-\frac{2}{3}}^{x} \frac{1}{1} dx \right)^{3-i_3}}^{condition\ 1} \cdot \left\{ \overbrace{\sum_{i_2=2}^{i_3} C_{i_3}^{i_2} \left(\frac{\int_{-\frac{1}{3}}^{x} \frac{1}{1} dx}{\int_{-\frac{2}{3}}^{x} \frac{1}{1} dx} \right)^{i_2} \cdot \left(1 - \frac{\int_{-\frac{1}{3}}^{x} \frac{1}{1} dx}{\int_{-\frac{2}{3}}^{x} \frac{1}{1} dx} \right)^{i_3 - i_2}}^{condition\ 2} \cdot \underbrace{\left[\sum_{i_1=1}^{i_2} C_{i_2}^{i_1} \left(\frac{\int_0^{x_1} \frac{1}{1} dx}{\int_{-\frac{1}{3}}^{x} \frac{1}{1} dx} \right)^{i_1} \cdot \left(1 - \frac{\int_0^{x_1} \frac{1}{1} dx}{\int_{-\frac{1}{3}}^{x} \frac{1}{1} dx} \right)^{i_2 - i_1} \right]}_{condition\ 3} \right\} \right\}}_{probability\ of\ the\ parking\ spot\ not\ been\ taken.} \right\} dx \quad \text{Eq 6}$$

Therefore, all three cars that possibly can reach location x are accommodated before then, and the parking spot at x won't be taken. Correspondingly, one minus this probability is the chance that the parking spot is taken.

$\boldsymbol{p_2}$ represents the probability of the parking spot (which is located at $x \in \left[\frac{1}{6}, \frac{1}{3}\right]$) being taken, given that only 2 (i.e., $\left\lfloor \frac{d}{s} \right\rfloor$) cars can actually passed by the location, one of them started driving at location of $-\frac{1}{3}$ and the other starts at location 0.

It can be seen in the equation below:

- Condition 1 represents the probability that, within the rest of the $A - 1$ available parking spots, a number of $i_2 \in [2, A-1]$ spots exist between $-\frac{1}{3}$ and x. Therefore, the car that started from $-\frac{1}{3}$ can be accommodated before it reaches x;
- Condition 2 represents the probability that, within these i_2 parking spots, a number of $i_1 \in [1, i_2]$ spots are actually located between 0 and x. Therefore, the car that started from 0 can be accommodated before it reaches x.

Therefore, both cars that possibly can reach to location x are accommodated before then, and the parking spot at x won't be taken. Correspondingly, one minus this probability is the chance that the parking spot is taken.

$$p_2 = \int_{\frac{1}{6}}^{\frac{1}{3}} \frac{1}{1} \left\{ 1 - \underbrace{\left\{ \overbrace{\sum_{i_2=2}^{3} C_3^{i_2} \left(\int_{-\frac{1}{3}}^{x} \frac{1}{1} dx \right)^{i_2} \cdot \left(1 - \int_{-\frac{1}{3}}^{x} \frac{1}{1} dx \right)^{3-i_2}}^{condition\ 1} \cdot \underbrace{\left[\sum_{i_1=1}^{i_2} C_{i_2}^{i_1} \left(\frac{\int_0^{x_1} \frac{1}{1} dx}{\int_{-\frac{1}{3}}^{x} \frac{1}{1} dx} \right)^{i_1} \cdot \left(1 - \frac{\int_0^{x_1} \frac{1}{1} dx}{\int_{-\frac{1}{3}}^{x} \frac{1}{1} dx} \right)^{i_2 - i_1} \right]}_{condition\ 2} \right\}}_{probability\ of\ the\ parking\ spot\ not\ been\ taken.} \right\} dx \quad \text{Eq 7}$$

Therefore, $n = A \cdot N \cdot (p_1 + p_2) = 2.9$. During the period, an average value of 2.9 cars out of 3 find parking, 2.9 parking spots out of 4 are newly occupied.

Scenario 3: $\boldsymbol{d} \in [\boldsymbol{L}, \infty)$.

Assume the same values as before (i.e., L=1km; A=4; N=3; s=1/3km) but a different value of d=2km. Then $n = min\{A, N\} = 3$. During the period, all 3 cars find parking, 3 parking spots out of 4 are newly occupied.

3. Validation

Our analytical model, although restricted in many aspects, provides us a basic approach to describe the "process and success" of searching for parking. For the validation part, it is currently impossible to validate the equations with real data, but a matlab program is provided to run experiments with random inputs. The code for both the model and the experiment can be seen in the appendix. We used the results obtained in the experiments to validate the model (Eq. 1-3). In the experiment, each set of input can be ran for 1000 times (or more) to obtain the averaged value in order to be compared with the analytical results.

4. Summary

This paper was the starting of the development of the parking-state-based transition matrix, which aims to model macroscopically a dynamic urban parking system. The model and equations are drawn based on quite restrictive conditions, nevertheless, the results do can represent a real life parking searching system. Besides, the results have been tested/ validated by programmed experiments. The next step would be to model the departures from the parking facilities, then the matrix can be built.

5. References

Arnott, R., & Inci, E. (2006). An integrated model of downtown parking and traffic congestion. Journal of Urban Economics, 60(3), 418-442.

Arnott, R., & Inci, E. (2010). The stability of downtown parking and traffic congestion. Journal of Urban Economics, 68(3), 260-276.

Axhausen, K. W., & Polak, J. W. (1991). Choice of parking: stated preference approach. Transportation, 18(1), 59-81.

Axhausen, K. W., Polak, J. W., Boltze, M., & Puzicha, J. (1994). Effectiveness of the parking guidance system in Frankfurt/Main. Traffic Engineering and Control, 35(5), 304-309.

Cao, J., M. Menendez and V. Nikias (2014) The effect of on-street parking on traffic throughput at nearby intersections. 93rd Annual Meeting of the Transportation Research Board, Washington, D.C., January 2014.

Ellis, R. H., Bennett, J. C., & Rassam, P. R. (1974). Parking systems analysis: an overview. ACM SIGSIM Simulation Digest, 6(1), 42-50.

Feeney, B. P. (1989). A review of the impact of parking policy measures on travel demand. Transportation Planning and Technology, 13(4), 229-244.

Gillen, D. W. (1978). Parking policy, parking location decisions and the distribution of congestion. Transportation, 7(1), 69-85.

Montini, L., Horni, A., & Rieser-Schüssler, N. (2012). Searching for parking in GPS data. Eidgenössische Technische Hochschule Zürich, IVT, Institute for Transport Planning and Systems.

Nourht, C. C., El-Reedy, T. Y., & Ismail, H. K. (1981). A COMBINED PARKING AND TRAFFIC ASSIGNMENT MODEL. Traffic Engineering and Control, 22(HS-032 693).

Polak, J. W. and K. W. Axhausen (1990) Parking search behaviour: A review of current research and future prospects, Working Paper, 540, Transport Studies Unit, University of Oxford, Oxford.

Shoup, D. C. (2005). The high cost of free parking (Vol. 7). Chicago: Planners Press, American Planning Association.

Thompson, R. G., & Richardson, A. J. (1998). A parking search model. Transportation Research Part A: Policy and Practice, 32(3), 159-170.

Van der Goot, D. (1982). A model to describe the choice of parking places. Transportation Research Part A: General, 16(2), 109-115

Young, W., Thompson, R. G., & Taylor, M. A. (1991). A review of urban car parking models. Transport Reviews, 11(1), 63-84.

Appendix

1. Analytical model

```
function model(A,L,N,d)
A=4;
L=1;
N=3;
d=5/6;

s=L/N;
m=min(N,floor(d/s)+1);
%%
if d<=s
    display 'Scenario 1: d<=s'
    naccess=N*(1-(1-d/L).^A)
end
%%
if and(d>s,d<L)
    display 'Scenario 2: s<d<L'
    %%
    if A<m
        display 'Scenario 2(1): s<d<L and A<m'
        naccess=A
    end
    %%
    if A==m
        display 'Scenario 2(2): s<d<L and A=m'
        sumcorefirstpart=0;
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        matrixsecondpart(1,1)=A-1;
        for j=2:m
            matrixsecondpart(1,j)=m-j+1;
        end
        matrixsecondpart;
        i=1;
        while matrixsecondpart(i,1)~=matrixsecondpart(i,m)
            matrixsecondpartnewline=matrixsecondpart(i,:);
            j=m;
            while matrixsecondpart(i,:)==matrixsecondpartnewline(1,:)
                if matrixsecondpart(i,j)~=matrixsecondpart(i,j-1)

matrixsecondpartnewline(1,j)=matrixsecondpartnewline(1,j)+1;
                    if j<m
                        for k=j+1:m

matrixsecondpartnewline(1,k)=matrixsecondpart(1,k);
                        end
                    end
                    matrixsecondpart(i+1,:)=matrixsecondpartnewline;
                else
                    j=j-1;
                end
            end
            i=i+1;
        end
        matrixsecondpart
        [k l]=size(matrixsecondpart)
```

```
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        combinationsecond(1:k,1)=1;
        for i=1:1:k
            for j=1:1:l-1
                c=nchoosek(matrixsecondpart(i,j),matrixsecondpart(i,j+1));
                combinationsecond(i,1)=combinationsecond(i,1)*c;
            end
        end
        combinationsecond
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        coresecondpart(1:k,1)=1;
        for i=1:1:k
            fun = @(x) (((N-m+2).*s-x).^(A-1-
matrixsecondpart(i,2)).*(x.^(matrixsecondpart(i,m))));

coresecondpart(i,1)=combinationsecond(i,1).*s.^(matrixsecondpart(i,2)-
matrixsecondpart(i,m)).*integral(fun,(d-(m-1)*s),s);
        end
        coresecondpart
        sumcoresecondpart=sum(coresecondpart(1:k))
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        core=sumcorefirstpart+sumcoresecondpart
        naccess=A*(1-N/(L.^A)*core)
    end
    %%
    if A>m
        display 'Scenario 2(3): s<d<L and A>m'
        matrixfirstpart(1,1)=A-1;
        for j=2:m+1
            matrixfirstpart(1,j)=m-j+2;
        end
        matrixfirstpart;
        i=1;
        while matrixfirstpart(i,1)~=matrixfirstpart(i,m+1)
            matrixfirstpartnewline=matrixfirstpart(i,:);
            j=m+1;
            while matrixfirstpart(i,:)==matrixfirstpartnewline(1,:)
                if matrixfirstpart(i,j)~=matrixfirstpart(i,j-1)

matrixfirstpartnewline(1,j)=matrixfirstpartnewline(1,j)+1;
                    if j<m+1
                        for k=j+1:m+1

matrixfirstpartnewline(1,k)=matrixfirstpart(1,k);
                        end
                    end
                    matrixfirstpart(i+1,:)=matrixfirstpartnewline;
                else
                    j=j-1;
                end
            end
            i=i+1;
        end
        matrixfirstpart
        [k l]=size(matrixfirstpart)
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        combinationfirst(1:k,1)=1;
        for i=1:1:k
            for j=1:1:l-1
                c=nchoosek(matrixfirstpart(i,j),matrixfirstpart(i,j+1));
                combinationfirst(i,1)=combinationfirst(i,1)*c;
```

```
            end
        end
        combinationfirst
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        corefirstpart(1:k,1)=1;
        for i=1:1:k
            fun = @(x) (((N-m+1).*s-x).^(A-1-
matrixfirstpart(i,2)).*(x.^(matrixfirstpart(i,m+1))));

corefirstpart(i,1)=combinationfirst(i,1).*s.^(matrixfirstpart(i,2)-
matrixfirstpart(i,m+1)).*integral(fun,0,(d-(m-1)*s))
        end
        corefirstpart
        sumcorefirstpart=sum(corefirstpart(1:k))
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        matrixsecondpart(1,1)=A-1;
        for j=2:m
            matrixsecondpart(1,j)=m-j+1;
        end
        matrixsecondpart;
        i=1;
        while matrixsecondpart(i,1)~=matrixsecondpart(i,m)
            matrixsecondpartnewline=matrixsecondpart(i,:);
            j=m;
            while matrixsecondpart(i,:)==matrixsecondpartnewline(1,:)
                if matrixsecondpart(i,j)~=matrixsecondpart(i,j-1)

matrixsecondpartnewline(1,j)=matrixsecondpartnewline(1,j)+1;
                    if j<m
                        for k=j+1:m

matrixsecondpartnewline(1,k)=matrixsecondpart(1,k);
                        end
                    end
                    matrixsecondpart(i+1,:)=matrixsecondpartnewline;
                else
                    j=j-1;
                end
            end
            i=i+1;
        end
        matrixsecondpart
        [k l]=size(matrixsecondpart)
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        combinationsecond(1:k,1)=1;
        for i=1:1:k
            for j=1:1:l-1
                c=nchoosek(matrixsecondpart(i,j),matrixsecondpart(i,j+1));
                combinationsecond(i,1)=combinationsecond(i,1)*c;
            end
        end
        combinationsecond
        %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
        coresecondpart(1:k,1)=1;

        for i=1:1:k
            fun = @(x) (((N-m+2).*s-x).^(A-1-
matrixsecondpart(i,2)).*(x.^(matrixsecondpart(i,m))));

coresecondpart(i,1)=combinationsecond(i,1).*s.^(matrixsecondpart(i,2)-
matrixsecondpart(i,m)).*integral(fun,(d-(m-1)*s),s);
```

```
        end
        coresecondpart
        sumcorefirstpart
        sumcoresecondpart=sum(coresecondpart(1:k))
%%
        core=sumcorefirstpart+sumcoresecondpart
        naccess=A*(1-N/(L.^A)*core)
    end
end
%%

if d>=L
    display 'Scenario 3: d>=L'
    naccess=min(A,N)
end
```

2. Experiments

```
function experimentrepeat(A,N,L,V,T)
A=4;
N=3;
L=1;
V=30;
T=5/6/30;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
repeattimes=10000;
final(repeattimes,1)=5555555555;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if A<N
    X=A;
else
    X=N;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
for z=1:repeattimes
    repeattimes%display
    %below is to generate the location of parking spots starting location
of the cars
    parklocation=rand(A,1)*L%display
    for j=1:N
        carlocation(1,j)=(j-1)*L/N;
    end
    carlocation%display
    %below is to locate the parking spots regards to the car locations,
    %e.g.,this parking spot is located between car 2 and car 3. This helps
to
    %measure the driven distance needed for each car to each parking spot.
    for i=1:A
        parklocation(i,2)=ceil(parklocation(i,1)/L*N);
        parklocation(i,3)=parklocation(i,2)+1;
    end
    parklocation%display the location of the parking spot with comparison
to the car's locations

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
    %below is to find distance/driving time needed for each car to reach
any of the parking
    %spot.
```

```
    for i=1:A
        for j=1:N
            if parklocation(i,3)==N+1
                dneeded(i,j)=parklocation(i,1)-carlocation(j);
            elseif parklocation(i,3)>j
                dneeded(i,j)=parklocation(i,1)-carlocation(j);
            else
                dneeded(i,j)=parklocation(i,1)-carlocation(j)+L;
            end
        end
    end
    dneeded%display
    %above is the distance for each car to reach the parking spots, number
in row i column j, means, the driven distance needed by car j to arrive at
parking spot i.
    %below is to find the minimum distance for any car to reach any parking
spot.
    dmin(1:X,1)=L;
    dmin(1:X,2:3)=0;
    for x=1:X
        for i=1:A
            for j=1:N
                if dneeded(i,j)<dmin(x,1)
                    dmin(x,1)=dneeded(i,j);
                    dmin(x,2)=i;
                    dmin(x,3)=j;
                end
            end
        end
        dneeded(dmin(x,2),:)=L;
        dneeded(:,dmin(x,3))=L;
    end
    dmin%display
    %dmin is the driven distance needed by the first cars that are reaching
the parking spots.
    tmin=dmin;
    tmin(:,1)=dmin(1:X,1)/V;
    tmin%display
    %tmin is the driven time needed by the first cars that are reaching the
parking spots.
    if T<tmin(1,1)
        nreal=0;
    elseif T>tmin(X,1)
        nreal=X;
    else
        for j=1:X-1
            if T>tmin(j,1)& T<tmin(j+1,1)
                nreal=j;
            end
        end
    end
    nreal;
    final(z,1)=nreal;
end

final
averagefinal=0;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
for z=1:repeattimes
    averagefinal=averagefinal+final(z,1);
```

```
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%
averagefinal=averagefinal/repeattimes
end
```