Team Control Number

For office use only
T1 _____
T2 _____
T3 _____
T4 _____

14124

Problem Chosen

B

For office use only
F1 _____
F2 _____
F3 _____
F4 _____

2012 Mathematical Contest in Modeling (MCM) Summary Sheet
(Attach a copy of this page to each copy of your solution paper.)

Scheduling the Rafting Trip

We are expected to simulate the whole trip routes and schedule an explicit timetable for each individual trip in a specified duration with utilizing the campsites as fully as possible. The number of campsites, the duration of the cycle to be scheduled and the maximum launching trips are thought to be the input parameters of our model while the utilization condition and the trip patterns remain to be optimized.

We go on to formulate the model by devising the data generator, which is proposed to simulate the choice of the tourists and provide initial data for the model. Such data will be scheduled following some rules by the scheduling algorithm, yielding the explicit timetable for each trip and the optimal utilization of campsites with the concerning of minimal contact.

Two algorithms are established and both can simulate the whole trip routes. The basic algorithm schedules the trip routes by avoiding any conflict about the occupied sites or any contact among the trips while the flexible algorithm tries to occupy any possible campsites within the specified scope to maximize the utilization rate of campsites with the tolerance of contact among the trips.

By estimating the results of the applications of the two algorithms, we find that the basic algorithm schedule the trip process with a disappointing utilization condition and limited trip patterns, but no contact is detected in the results. This algorithm could perform in some extreme cases where the rafting trip is so narrow that no contact is allowed regarding the concern of safety. On the contrary, the flexible algorithm schedules the trip routes with all patterns of trips and amazingly high utilization rate of campsites, which is up to 86% approximately in a specified example. However, the times of contact are significant as well. The flexible algorithm can be utilized in calm rivers, where the trip flow is taken as the primary concern.

We test the algorithm to see whether the algorithm is sensitive to some parameters. The results indicate that the flexible algorithm is insensitive to the number of campsites when it is approaching 25, where the maximum efficiency could be achieved. Thus we specify the number of campsites to be 25 and schedule the timetable, which is explicitly presented in the end of our paper, in one simulation cycle taken as 30 days.

Scheduling the Rafting Trip

Team # 14124

February 13, 2012

Abstract

We are expected to simulate the whole trip routes and schedule an explicit timetable for each individual trip in a specified duration with utilizing the campsites as fully as possible. The difficulty of such simulation lies in addressing the conflict happened in the occupied campsites and allocating the vacant campsites properly .

We begin with analyzing the related parameters which may affect the simulation results. The number of campsites , the duration of the cycle to be scheduled and the maximum number of launching trips are thought to be the input parameters of our model while the utilization condition and the trip patterns remain to be optimized . Based on the analysis of the preference and behavior of tourists, we assume that trips ranging from 6 to 18 nights launch randomly every day below a certain threshold and the propulsion(oar and motor) should be regarded as equivalent in the numerical algorithm .

We go on to formulate the model by devising the data generator, which is proposed to simulate the choice of the tourists and provide initial data for the model. Such data will be scheduled following some rules by the scheduling algorithm ,thus yielding the explicit timetable for each trip and the optimal utilization of campsites with the concerning of minimal contact .

Two algorithm are established and both can simulate the trip routes . The basic algorithm schedules the trip routes by avoiding any conflict about the occupied sites or any contact among the trips while the flexible algorithm tries to occupy any possible campsites within the specified scope to maximize the utilization rate of campsites with the tolerance of contact among the trips.

By estimating the results of the applications of the two algorithm , we find that the basic algorithm schedules the trip routes with a disappointing utilization condition and limited trip patterns . But no contact is detected in the results. This algorithm could perform in some extreme cases where the rafting trip is so narrow that no contact is allowed regarding the concern of safety. On the contrary, the flexible algorithm schedules the trip routes with all patterns of trips and amazingly high utilization rate of campsites ,which is up to 86% approximately in a specified exmaple. However, the times of contact are significant as well. The flexible algorithm can be utilized in calm rivers where the trip flow is taken as the primary concern.

We test the algorithm to see whether the algorithm is sensitive to some parameters. The results indicates that the flexible algorithm is insensitive to the number of campsites when it is approaching 25, where the maximum efficiency could be achieved. Thus we specify the number of campsites to be 25 and schedule the timetable ,which is explicitly presented in the end of our paper, in one simulation cycle taken as 30 days. A memo is also prepared for the manager accordingly.

Contents

1 Introduction

The Big Long River, known for its amazing scenery and exciting whitewater rapids, attracts numerous tourists coming from different countries every year. People who want to enjoy it have to take a river trip ,a 225 miles downstream starting at a specified location called First Launch and ending at the Final Exit, with several days of camping.

Given that some tourists may prefer a laborsaving trip and some otherwise, oar-powered rubber rafts and motorized boats are provided for the tourists to choose. Additionally , the trip ranges from 6 to 18 nights of camping, thus the tourists can choose the duration of the trip accordingly. To make the camping available, some campsites are set throughout the river corridor, which distribute fairly uniformly.

With the increase in the quantity of the tourists, a schedule is required to make more trips available. We have been hired by the park managers to propose a scheme with the optimal configuration of trip duration and propulsion to utilize the campsites in the best way and determine the carrying capacity of the river.

The model we try to formulate, if it is complete, should:

- Utilize the campsites as possible as it can.
- Simulate the whole trip process ,propose a practical schedule and trips configuration, if given the initial conditions about the trips.
- Show how sensitive it is to the parameters in the model such as the number of the campsites and the trips configuration.

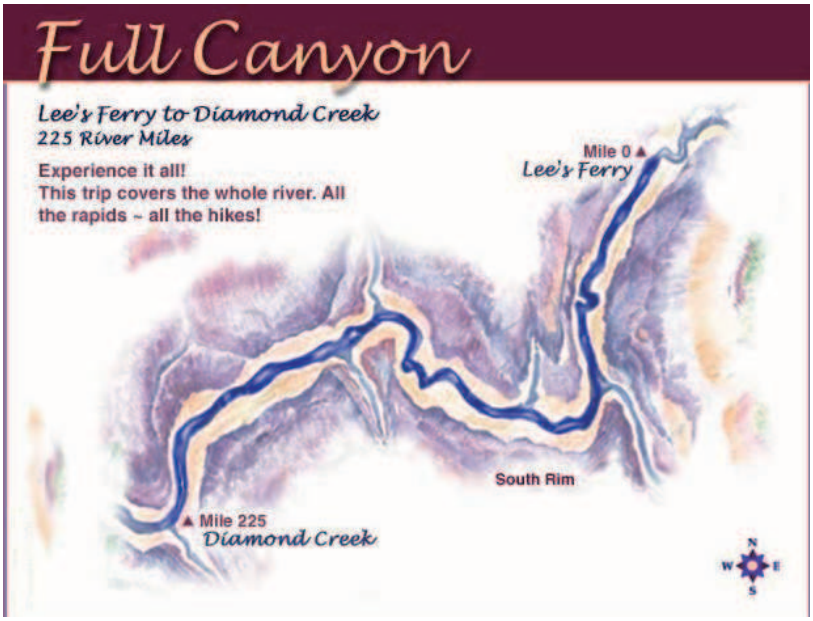

Figure 1: Rafting Trip along the River

2 Prerequisites

2.1 The Maximum Starting Capacity

Considering the limit of labor and outfit of the realistic park, the total number of the trips starting per day could not be infinite. We define the maximum starting capacity , represented by m, to denote the threshold of the number of the trips starting per day which should not be surpassed. This very threshold should not be too high because the government agency demands minimal contact among the trips on the river.

Given the value of Y, the number of campsites, the longest campsite the trip group can reach is given by $\lfloor \frac{Y}{6} \rfloor$,which represents the largest integer that does not exceed $\frac{Y}{6}$,if we assume that the trips would march with a relatively uniform speed. The reason of this assumption will be depicted in the formulation of our algorithm in Section.5. Thus the maximum starting capacity should not exceed $\lfloor \frac{Y}{6} \rfloor$, otherwise there will be more than one trip groups occupy the same site at the same time.

$$m \leq \left\lfloor \frac{Y}{6} \right\rfloor \tag{1}$$

2.2 The Choice of Tourists

The park have provided different trip configurations ,varying from 6 to 18 nights of camping with oar-powered rubber rafts or motorized boats, and tourists can choose which one to take according to their preference. The choice made by tourists would drastically affect the results of our scheduling method , so analyzing the behavior of the tourists is necessary.

2.2.1 An Extreme Case

Imagining the case where all the tourists choose the 6 nights trip with motorized boats, which is considered to be the most laborsaving and rapid way to travel the Big Long River. From the standpoint of the park manager, it is such an encouraging and amazing phenomenon because maximum quantity of tourists could enjoy the Big Long River in per unit duration in this case , bringing the maximum profit to the park . Besides , the only way people choose on which to travel is motorized boats, making the park management significantly convenient. The utilization of campsites seems to be convenient as well on condition that the park manager schedules the starting time and camping time for each trip properly.

The problem seems to be explicit and simple with the extreme initial condition, right? No!

This extreme case will never occur because the variety of the preference and the uncertainty of behavior of people could not be neglected. In reality , the preference and behavior of the tourist is very complex.

2.2.2 The Realistic Behavior of Tourists

Travelers have their own preference and habits. Some may prefer enjoying the idyllic scenery along the river so that they would like to choose a trip by motorized

boat with relatively long duration, such as a trip with 17 or 18 nights of camping. In a contrary manner , some may intend to challenge themselves with high level of difficulty , for instance, a rapid accomplishment of trip without motor . In that case , the adventure of the trip is the primary attraction to them. There are also some people making the neutral decision, they just take the trips with mean nights of camping like 12 nights or 13 nights, etc.

The experienced travelers know their own preference and habits, so they can make some explicit decision accordingly, but when it comes to the inexperienced travelers, the problem goes more complex. Some inexperienced travelers do not know the trip well, like the strength and weakness of oar-powered rubber rafts and the motorized boats respectively and the impact on their emotions due to the duration of the trip, thus the uncertainty of their decision could be significant.

Even if we devise a complete timetable of starting time with specified trip configuration (such as the trip with 7 nights of camping by motorized boat), the behavior of the tourists still remains uncertain. For instance, we provide 2 trips with 6 nights of camping by oar-powered rubber raft and motorized boat respectively and a trip with 16 nights of camping by motorized boat today. The first two trips may be taken by tourists who prefer an exciting trip with short duration soon, with the other one remaining vacant.

Therefore, it would be unwise to devise a model based on a specified timetable and trips configuration (like the previous passage shows). Our model should be flexible enough to deal with random choice of the trip configuration of the tourists.

2.3 The Scheduling Cycle

The Big Long River is available during a six month period each year because the rest of the year is too cold to raft. Thus we assume that the number of tourists would not be affected significantly by the season factors when the Big Long River Park is open. Or rather, the tourists arrival distribute approximately uniformly during the six month period. Note that the uniform distribution of the number of tourists is at the year level. when it comes to the trips launched each day, the behavior of the tourists remains a random state.

We define the scheduling cycle by dividing the six month period when the Big Long River is available into several parts. Each part denotes a cycle during which we try to schedule the whole process of the trips, including arranging the launching time and camping time and determining the corresponding camping site for each trip taken during the cycle . Considering the uniform distribution of the number of tourists throughout the six month period, each scheduling cycle can be taken as uniform.

2.4 The Campsites Efficiency

Every night some campsites accommodate the trip groups while the others remain vacant. The campsites efficiency is defined to denote the utilization rate of the campsites. Since every campsite could be occupied by only one set of campers and the tourists flow is measured in trip group, we can assert that the tourists in the different campsites can be regarded equivalent. Therefore, the utilization rate can be calculated by the radio of occupied campsites to the whole number of campsites per day.

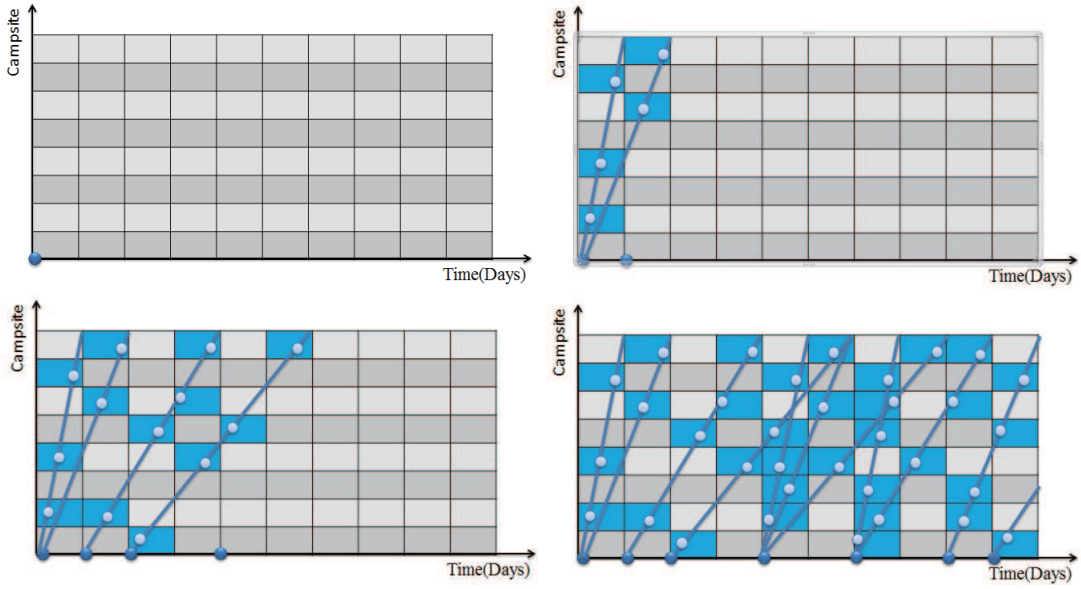

Figure 2: The Plot Showing the Dynamic Nature of Campsites Efficiency

As is shown is Figure.2,the problem is that the utilization rate alters every day if we validate the previous definition because of the initial random behavior of the tourists and the dynamic nature of our scheduling method. Given the uniformly distributed nature of the tourists in each cycle, we can solve the problem by defining the campsites efficiency as:

$$\eta = \frac{\sum_{i=1}^{T} Y_i}{TY} \tag{2}$$

where η represents the campsites efficiency , T represents the number of days in a scheduling cycle, namely, 30 days in this problem,Y represents the total number of the campsites, $Y_i$represents the number of occupied campsites in the i^{th} day in a scheduling cycle.

2.5 The Carrying Capacity

According to the Ref.[1], the carrying capacity of the river is defined as the maximum number that can fit on the site at any given time and still allow people to be able to move. As for the Big Long River, the carrying capacity can be given by the maximum number of the trips on the river at any given day correspondingly.

2.6 The Difference Between The Propulsion

In this problem, the difference between the propulsion appears to lie in the speed. However, this assertion is not sufficient.

The average journey per day can be calculated by the ratio of the total trip length, namely, 225 miles to the days required to accomplish the whole trip. Since the most rapid trip lasts 6 nights, the maximum average journey per day the passengers can

reach is given by the radio of 225miles to 7 days(after the 6^{th} camping night, the passengers can also utilize the next daytime to accomplish the trip),or rather, 32.1 miles per day approximately. This analysis indicates that the passengers who take the trip by oar-powered rubber rafts can accomplish the trip, thought to be easier for those by motorized boats, within the shortest specified duration on condition that they can keep rafting about 8 hours per day.

The propulsion (motor and oar) can be regarded as equivalent in the algorithm if doubling time rafting every day is achievable for the passengers. Since the difference of speed is negligible, we should focus on some other aspect.

As we have analyzed in Section.2.2, the behavior and preference vary with respect to the tourists. For those preferring a laborsaving and leisure trip, the motorized boats are their first choice while for those intending to enjoy an adventure, the oar-powered rubber rafts are more satisfactory. Thus the propulsion is taken randomly as we have discussed in Section.2.2.

2.7 Basic Assumptions

Based on the previous analysis and the information we have gathered from the Internet, we propose the basic assumptions of our problem as follows:

- **No two trips with the same configurations(the same duration and the same propulsion) could be launched at the same day.** This assumption is based on the common practice of some famous rafting park as shown in Ref.[2].
- **Trips ranging from 6 to 18 nights launch randomly every day below a certain threshold** because the preference of the tourists varies and the choice of the trip patterns is random
- **The propulsion (motor and oar) can be regarded as equivalent in the algorithm.**

2.8 Restatement of the Problem

Given these considerations, we can restate the problem as follows:

Given a certain set of initial data(the duration varying from 6 to 18nigths) about the trips in a predefined scheduling cycle, namely ,30days, simulate the whole trips process in the cycle depicting the very schedule of each individual trip with the objective that :

- The campsites efficiency should be maximized .
- Trip flow added to the river should be as much as possible.
- No two trips can occupy the same site at the same time.

3 Formulate the Model

Our model consists of two primary component:

- the data generator, simulating the choice of the tourists and producing the initial data.
- the scheduling system, devising a set of algorithms, simulating the whole trip schedule for each trip in the cycle and determine the optimal timetable and configuration planning.

4 The Data Generator

The data generator is proposed to simulate the behavior of the tourists, which has been discussed explicitly in Section.2.2. Given the maximum starting capacity, the data generator chooses a trip ranging from 6 to 18 nights of camping randomly. This generating loop will terminate after it has executed the maximum starting capacity times.

Note that the generator regards the oar-powered rubber rafts and motorized as equivalent, the reasons have been depicted specifically in Section.2.6

5 The Scheduling System

5.1 The Basic Algorithm

Figure 3: The One-dimension Description of the Problem

The rafting process is an one-direction motion on the river, the Big Long River can be simplified as a set of sequential nodes, which represents the campsites along the river. Thus the trip with specified duration can be depicted as a leaping motion starting from the launching node to the exit node with the temporary stay on some nodes of the nodes set.

Considering that this one-direction motion , depicted with spacial dimension, is dynamic and various sets of motion spawn every day , we can take the time dimension into account as is shown in Figure.2.

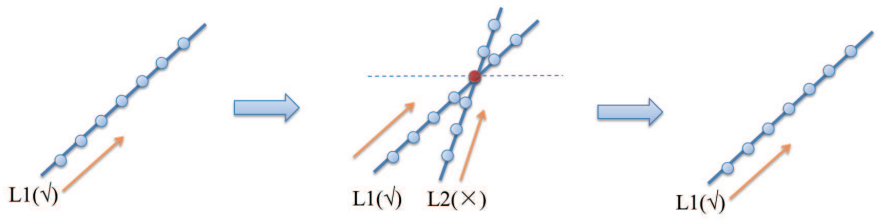

Figure 4: The Mechanism of the Basic Algorithm

The plot shown in Figure.2 of the nodes(measured in campsites) v.s time(measured in days) depicts the solution basis of this very 2-dimensional problem. The

maximum of nodes is the number of campsites plus 1(considering the exiting node, which is not included in the campsites node),namely,$Y + 1$.

The simulation of the trip process can be specified as follows: In the i^{th} day of the cycle , each trip intended to launch at the i^{th} day spawns at the coordinate (0,i) and the trip schedule(the camping time and corresponding camping node) is predefined following some specified principals. Thus the trip schedule could be denoted by a set of nodes whose y-coordinate increases with the increment of x-coordinate. Different patterns of trips end at the same point $Y + 1$ with various results of trip duration.

The basic algorithm

```
while still running do
    extract the trips data in the i^th day from data generator
    for each trip do
        predefine the trip schedule
        if the trip schedule occupies the site having been occupied then
            cancel this trip
        end if
    end for
    increment i
end do
```

Considering the restriction that no two trips could occupy the same site at the same time, we can validate our basic algorithm by specifying some rules to avoid such time conflict as follows: **During the simulation process, if two trips occupy the same site, the trip launched relatively later should be canceled.**

5.1.1 A Simple Example

We test our basic algorithm by applying it into a simple example. In this example, the maximum starting capacity is set to be 3 while$Y$ is set to be 20 . Only one pattern of trip is available (take the trip with 10 days for instance) with 5 different predefined schedule every day which spawns randomly as shown in Table.1.

Table 1: The Predefined Schedule of the Example

	Marching step every day							
Route 1	7	3	4	5	6	8	2	5
Route 2	5	8	2	3	1	8	9	4
Route 3	4	2	7	4	5	8	2	8
Route 4	1	4	9	2	8	5	7	4
Route 5	1	8	1	5	7	3	10	5

We test the basic algorithm iteratively and the simulation results vary due to the randomness of the spawning of predefined schedule .The results are shown in Figure.5.

According to the results of our simulation ,we assert that the conflict can be avoid in the basic algorithm because no route occupies the same node at the same day as is shown in Figure.5.

However , the results indicate that withdrawing the trips which would conflict with the trips launched before in campsites affects the campsites efficiency negatively. Considering that where the conflict occurs, the adjacent node may be vacant, resulting

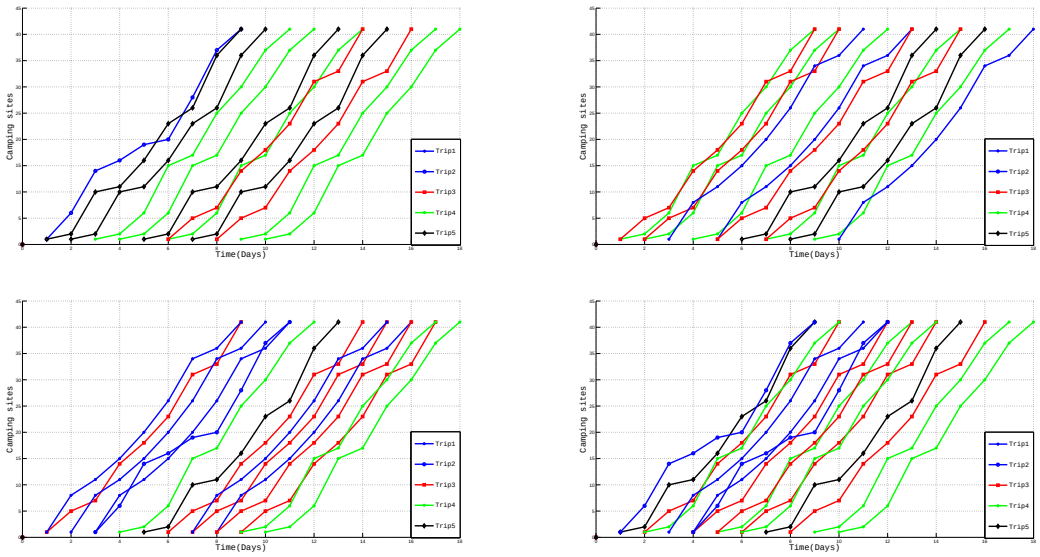

Figure 5: The Simulation Results of the Example

in low rate of utilization of campsites, we propose the flexible algorithm to fully utilize the vacant node by modifying certain steps of the basic algorithm.

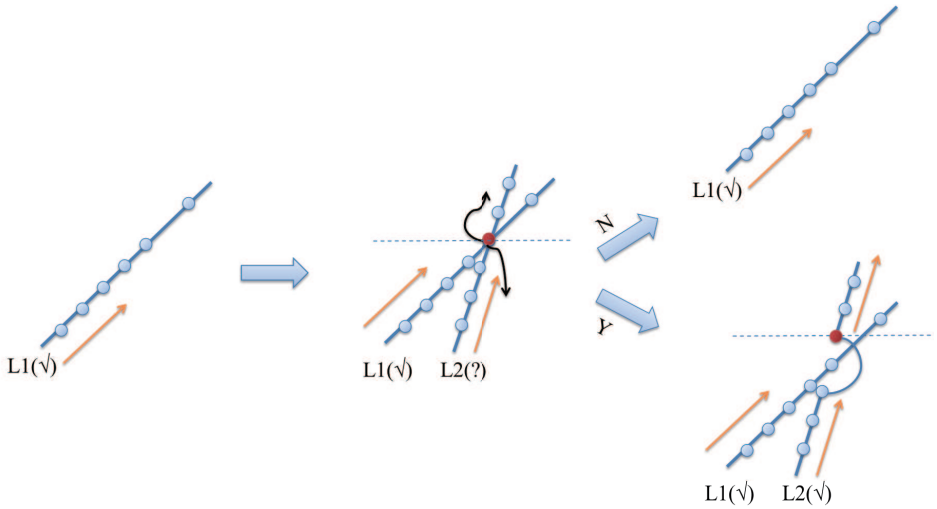

Figure 6: The Mechanism of the Flexible Algorithm

5.1.2 Predefined Schedule

Note that in this example, the predefined schedule is determined randomly. This randomness is negligible because we just want to test the ability to avoid conflict of the algorithm in this case. But we should think over the rules of the predefinition of the schedule so that it can fit the actual situation. We assume that the journey per day stays relatively invariable. However , the journey is measured in the number of campsites(integer)while the journey per day might not be divided by the distance

between two adjacent nodes without remainder. So rounding to integer is necessary. Given such analysis , the predefined schedule could be given as:

$$y_i = \begin{cases} \left[\frac{iY}{d}\right] & , \frac{iY}{d} - \left[\frac{iY}{d}\right] \leq 0.5 \\ \left[\frac{iY}{d}\right] + 1 & , \frac{iY}{d} - \left[\frac{iY}{d}\right] > 0.5 \end{cases} \tag{3}$$

where y_i is the y-coordinate of the trip in the i^{th} day ,d is the duration of the trip and$Y$ is the number of campsites.

5.2 The Flexible Algorithm

We devise the flexible algorithm on the basis of the basic algorithm by revising the way on which it avoids the conflict.

The flexible algorithm avoids the conflict by searching for the adjacent vacant node for camping. We define the searching rules as follows:

Check to see if the most adjacent node is occupied first, if no adjacent campsite is available, gradually increase the searching step within the given searching scope until a vacant node is found. The scope can be defined as a set of nodes starting from the node occupied in the previous day to the booked node in the following day(this two very nodes are not included). Considering the extreme case that no vacant node is available within the given scope, we have to cancel the trip.

The flexible algorithm

```
while still running do
    extract the trips data in the i^th day from data generator
    for each trip do
        define flag:=0
        predefine the trip schedule
        define index:=the index of the booked node
        while the trip schedule occupies the site having been occupied do
            increment flag
            if (index + flag) exceed the specified scope then
                cancel this trip
                break
            else if the (index + flag)^th or (index − flag)^th node is vacant then
                    occupy the node
                    break
                end if
            end if
        end do
    end for
    increment i
end do
```

6 The Application of our Model

Before we apply our model to solve the Big Long River problem , we have to specify the exact value of some parameters.

For the value of Y , the number of campsites, we have noticed that the Grand Canyon is 255 miles long , which resembles the length of the Big Long River. So we can refer to the arrangement of the Grand Canyon.we assume that there are 25 campsites along the river. Thus the maximum starting capacity can be set to be 4 correspondingly based on the Equation.1.Besides , we devise the value of cycle to be 6 months. That is to say , the model will be utilized to simulate the trip process for the whole six month period.

Table 2: Parameters and the Corresponding Values

Parameter	Value	description
Y	25	number of campsites
m	4	maximum starting capactity
T	180	scheduling cycle

6.1 Application of Basic Algorithm

By utilizing the basic algorithm , the trip process in the cycle is simulated . The simulation result is depicted in Figure.7.

Note that no trip launches after the 162^{th}day because the trips launches in the162^{th}day with the duration of 18 nights couldn't reach the destination within the cycle . Thus we devise that no trip could launch after the 162^{th}day.

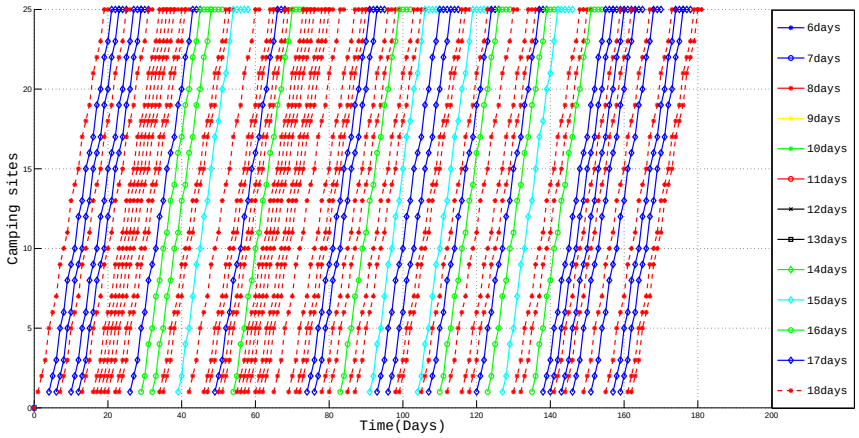

Figure 7: The Simulation Results of the Basic Model

The result surprisingly shows that only 4 types of trip (the trips with 15, 16, 17, 18 nights of camping) are available , this would be a disaster for the tourists who intend for an exciting adventure on the Big Long River because no trip with relatively short duration ,which is the prerequisite of an exciting trip, can be provided. Given the random nature of the choice of the trip configurations and the lack in the trip configurations which could be provided, only a small proportion of the requirement could be satisfied . In other words , the basic algorithm fail to add more trips to the Big Long River. On the contrary , the basic algorithm may cause a decrease in the trip flow by ignoring a large proportion of requirements of the tourists.

The total trips provided by the algorithm during the whole rafting season are merely 76. That means no more than one trip is launched during any two days on average. This result is unacceptable for the park manager.

Table 3: Simulation Result of the Basic Algorithm

Total Days	180
Total Flow	76
Efficiency	0.3221
Carrying Capacity	8

Let us turn to the analysis of the campsites efficiency. By utilizing the Equation. 2,we figure out the campsites efficiency to be 32.21%, which is disappointingly low. Such value indicates that only 8 campsites are occupied with the other 17 campsites remaining vacant on average . Such a waste in the utilization of the campsites along the river makes the basic algorithm unacceptable.

Possibly the only consolation is the fact that the basic algorithm can avoid any contact among the trips on the river because no intersection among the trip routes, denoting the contact among trips in the actual situation, is detected .

6.2 Application of flexible algorithm

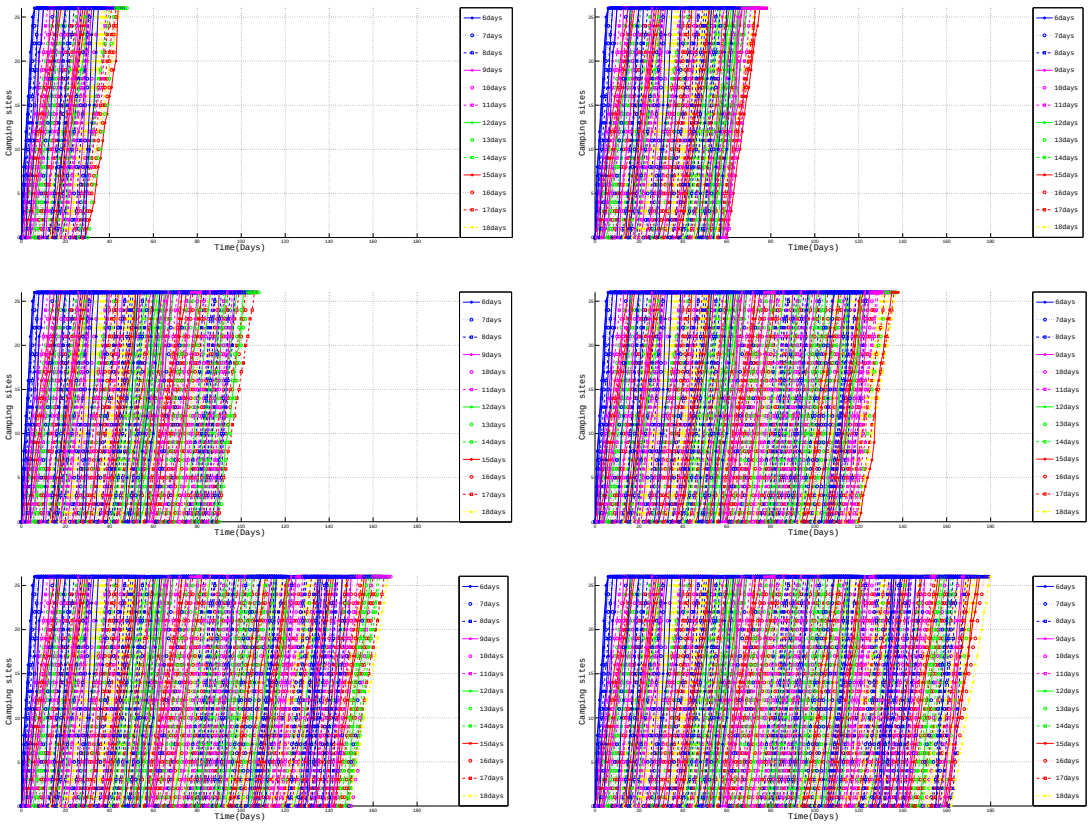

Figure 8: The Simulation Results of the Flexible Model with Cycle to be 180Days

We utilize the flexible algorithm to simulate the trip process correspondingly. Figure.8 shows the dynamic scheduling process for each individual trip and Table.4 shows the outputs of our model.

Table 4: Simulation Result of the Flexible Algorithm with Cycle to be 180Days

Total Days	180
Total Flow	340
Efficiency	0.8662
Carrying Capacity	21

We find that , contrary to the performance of basic algorithm, all the trip patterns (ranging from 6 to 18 nights of camping) are available in this simulation,indicating that the requirements of tourists could be fully satisfied . Additionally , the total trips provided during the whole rafting season is drastically increased when compared with those provided by basic algorithm . Thus more people have the access to the trips on the Big Long River. This is a piece of good news for both the tourists and the park managers.

When we turn to the campsites efficiency , the superiority of the flexible algorithm becomes more obvious. We figure out the campsites efficiency as the previous application do and find that its value is up to 86.62% , presenting a high utilization rate of the campsites during the rafting season. Imagine how excited the park manager would be if he notes that more than 21 campsites are occupied on average every day.

However , the contact among the trips , measured by the intersection point in the plot , increases significantly. Besides , the timetable is more difficult to determine because of the incredible complexity of the simulation results while the flexible algorithm accomplishes the allocation of campsites for the huge number of trips in each given day . In scheduling the trip process at a system level, we call upon a cycle with relatively shorter duration.

- Revise the Cycle Down to 30 Days

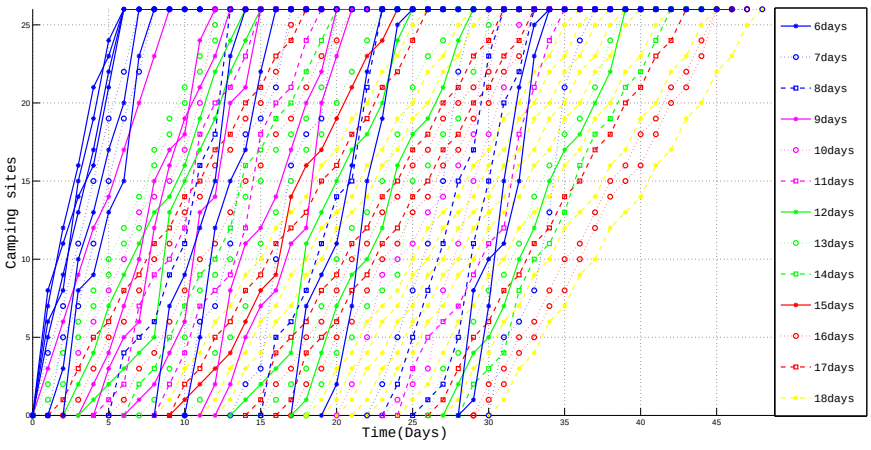

Figure 9: The Simulation Results of the Flexible Model with Cycle to be 30Days

Revising the values of cycle down to 30, yields a new set of simulation results shown in Figure.9. The variety of the trips do not change when compared to the results simulated with cycle to be 180days. To estimate the results ,we calculate the campsites efficiency to be 75.6%, which does not alter significantly. we compare the trip flows by detecting the total trips in the whole rafting season. So the total trip flow is approximately $71 \times 6 = 426$,which means the performance of flexible algorithm remains stable when the value of cycle is revised down to 30days. But scheduling the trip routes is achievable with the shorter cycle duration.

The timetable will be presented following the memo at the end of our paper.

Table 5: Simulation Result of the Flexible Algorithm with Cycle to be 30Days

Total Days	30
Total Flow	71
Efficiency	0.7560
Carrying Capacity	21

7 Sensitivity Analysis and Model Testing

We test the algorithms by changing the value of some parameters and detecting the corresponding variance in the output. Figure.10 shows the campsites efficiency and average trip flow for a given schedule cycle which ranges from 30 to 180 with a step of 30 days. The sensitivity of efficiency decreases slowly and turns to be stable ultimately while the sensitivity of average flow decreases at first, which is followed by an increase in the end.

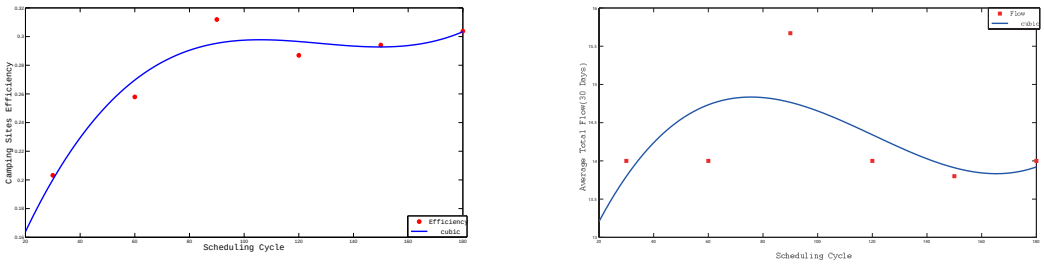

Figure 10: Testing the Basic Algorithm by Varying the Scheduling Cycle

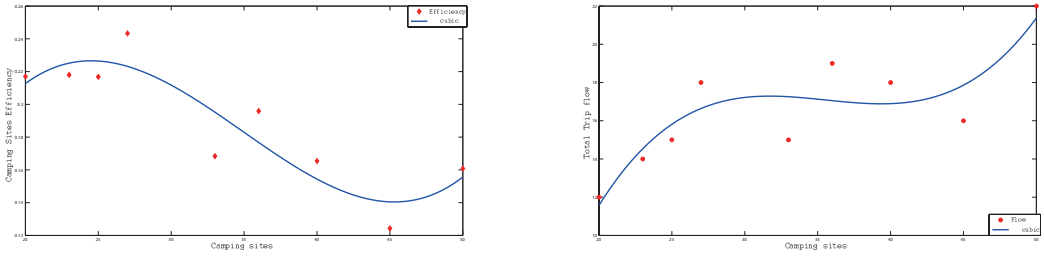

Figure 11: Testing the Basic Algorithm by Varying the Campsites

We can estimate the sensitivity of efficiency and total flow in terms of campsites as shown in Figure.11in the similar way to analysis in Figure.10 as well.

Figure.12 shows the campsites efficiency and average trip flow for a given schedule cycle which ranges from 30 to 180 with a step of 30 days. The set of discrete points denotes the corresponding efficiency or average trip flow for a given cycle while the curve depicts the approximate trend of the data.

We can also estimate the efficiency and total trip flow in terms of campsites as shown in Figure.13.

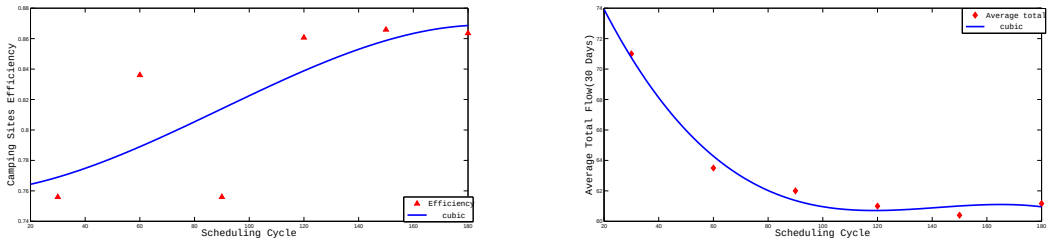

Figure 12: Testing the Flexible Algorithm by Varying the Scheduling Cycle

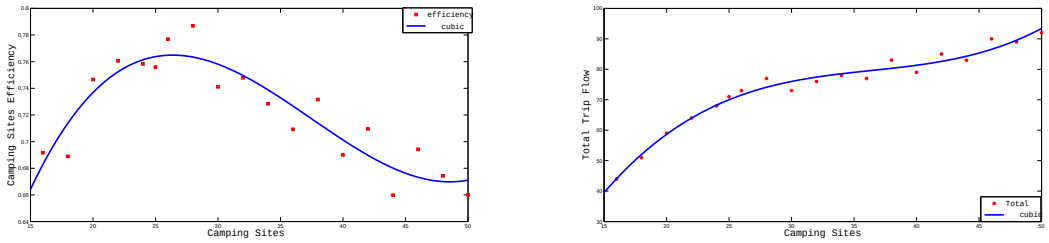

Figure 13: Testing the Flexible Algorithm by Varying the Campsites

Note that when the number of campsites is approximately 25, the slope of the cubic curve trends to zero and the campsites efficiency reaches a maximum. This fact validates our plan to schedule the timetable where the number of campsites is set to be 25 because it is the very case that the campsites can be fully utilized. We estimate the optimal trip configurations by determining the number of campsites with which the maximum campsites efficiency appears.

8 Analysis of Strength and Weakness

8.1 The Basic Algorithm

Strength: The basic algorithm observes the expectation of the government agency that minimal contact occurs on the river because there is no contact (intersection point of two given route) detected in the whole simulation process.

Weakness: The basic algorithm provides limited trip configurations (4 types only in the example in Section) , holds very few trips on the river and utilizes the campsites with extremely low efficiency.

8.2 The Flexible Algorithm

Strength: The flexible algorithm satisfies various requirements of the tourists for the trip configurations (all the 13 types in the example in Section) and allows a huge trip flow on the river with high campsite efficiency.

Weakness: A significant amount of contact can be detected in the scheduling cycle . Besides, the flexible algorithm is relatively more computationally expensive.

9 Possible Extension and Further Consideration

The factors we have considered in our analysis are thought to be crucial. In fact, many factors which we do not incorporate to our analysis should not be neglected.

- Considering the effect of the climate and season , the choice of the tourists could not be fairly random.
- The contact can be further avoided.
- The launching of the trips with the same configuration(duration and propulsion) may be allowed by allocating the exact launching time properly.

10 Memo

To: Big Long River Management Team
Data: February 13, 2012
Subject: Some Significant Suggestions for Big Long River Management

Dear Big Long River Management Team:

I am glad to report our researching findings to you, since we have been hired to schedule the optimal trip process for the Big Long River.

We have developed two algorithms to address the problem by concerning different objective or restrictions. The basic algorithm can schedule the trip process with no contact among the trips while the flexible algorithm can utilize the campsites in the best way .

We have applied our model to a certain rafting canyon and the following data would be referred to when necessary. Either the basic algorithm or the flexible algorithm schedule each individual trip for the whole rafting season respectively . The campsites efficiency calculated by the basic algorithm is 32.21% while the one calculated by the flexible algorithm is up to 86.62%.The total number of trips launch in the basic algorithm during the whole rafting season is merely 76 while the one of flexible algorithm is up to 340. Note that in the basic algorithm running , no contact occurs in the whole duration. We have scheduled a timetable for a month simulated by the flexible algorithm, whose corresponding campsites efficiency reaches 75.6%, if this case fits the situation of the Big Long River, you can refer to the timetable in the following page.

Recommendations on the choice of algorithms:

Based on the previous data, we recommend that :

For the narrow rafting river ,like the Granite Narrows whose narrowest point is merely a tight squeeze at 76 feet[3],any contact among the trips may lead to crashing or other hazard . So we recommend that the basic algorithm should be utilized considering the consequence of contact.

For the calmer rafting river , like the Family Rafting provided by villmarkskompaniet[4] where the effect of contact can be ignored since the rafting is just a safe slow moving, the flexible algorithm can be proposed to full utilize the campsites.

For the river combined with narrow rafting and calm rafting sections, we recommend that the river should be divided into several trips according to the difficulty grade and the basic and flexible algorithm should be utilized for the corresponding section respectively.

Other suggestions:

We regard the two types of propulsion(oar and motor) to be equivalent in the algorithm, but the preference of the tourists varies, so the difference between them should be considered here. Considering the travel time and travel distance in every single day, we advice the managers to provide motorized boats for the tourists who select the trips range from 6 to 9 nights of camping on the river while providing oar-powered rubber rafts for those who select the trips range from 15 to 18 nights. For the rest cases , either propulsion could be provided .

May all of these suggestions and the timetable presented in the following pages be helpful to you.

The schedule of 13 types of trips in per month (Y=25)

Range / Time	6D	7D	8D	9D	10D	11D	12D
1	√√√√	√		√			
2	√	√					
3	√				√		√
4				√			√
5				√		√	
6			√				
7				√			
8							
9	√					√	
10							
11	√	√					
12				√			
13				√			
14		√					√
15							
16			√				
17							
18	√						√
19							
20	√						
21					√		
22							
23		√					
24			√		√		
25						√	
26			√				
27							
28							√
29	√√						
30							

Range / Time	13D	14D	15D	16D	17D	18D
1	√√					
2					√	
3				√		
4						
5						
6				√		
7		√				
8	√					
9						√
10			√		√	
11	√					
12						√
13						√
14						
15					√	
16	√			√		
17					√	
18				√		
19	√					√
20						√
21						√
22						√
23						√
24						
25						√
26						
27	√				√	
28						√
29		√				
30				√√		

Time (Days)	Trip Range	Day 0	Day 1	Day 2	Day 3	Day 4	Day 5	Day 6	Day 7	Day 8	Day 9	Day 10	Day 11	Day 12	Day 13	Day 14	Day 15	Day 16	Day 17	Day 18
1	6D	0	5	9	13	17	22	26												
1	6D	0	6	8	15	19	24	26												
1	6D	0	8	11	14	16	21	26												
1	6D	0	7	12	16	21	23	26												
1	7D	0	4	7	11	15	19	22												
1	9D	0	3	6	9	12	14	17	20	23	26									
1	13D	0	2	4	6	8	10	12	14	16	18	20	22	24	26					
1	13D	0	1	2	5	7	9	11	12	17	19	21	23	25	26					
2	6D	0	3	10	13	17	20	26												
2	7D	0	5	7	11	15	19	22	26											
2	17D	0	1	3	5	6	8	9	11	12	14	15	17	18	20	21	23	24	26	
3	6D	0	8	9	13	15	23	26												
3	10D	0	4	6	8	10	13	14	15	17	20	26								
3	12D	0	2	4	7	9	11	13	14	16	19	22	24	26						
3	16D	0	1	3	5	6	8	10	11	13	14	16	17	19	20	22	25			
4	9D	0	2	4	7	10	15	17	18	24	26									
4	12D	0	1	2	3	4	5	13	15	17	19	22	24	26						
5	9D	0	3	5	6	12	16	19	21	23	26									
5	11D	0	1	2	7	9	10	12	18	20	21	23	26							
6	8D	0	4	5	6	9	11	16	18	26										
6	16	0	1	3	4	6	8	10	11	13	15	16	18	19	21	23	24	26		
7	9D	0	1	2	4	6	13	14	20	21	26									
7	14D	0	2	3	5	7	9	10	12	16	17	19	20	22	24	26				
8	13D	0	1	3	5	8	9	10	14	15	17	18	19	21	26					

9	6D	0	7	9	12	15	23	26												
9	11D	0	2	4	7	8	9	12	18	20	21	24	26							
9	18D	0	1	3	4	6	7	9	10	12	13	14	16	17	19	20	22	23	25	26
10	15D	0	1	2	3	4	6	8	9	14	16	17	19	21	23	24	26			
10	17D	0	2	3	5	6	8	9	11	12	13	15	16	18	19	21	22	24	26	
11	6D	0	5	12	15	17	22	26												
11	7D	0	6	7	11	18	19	21	26											
11	13D	0	1	4	5	7	11	13	15	17	18	20	22	24	26					
12	9D	0	2	8	11	12	14	17	20	22	26									
12	18D	0	1	3	4	6	7	9	10	12	13	14	16	17	19	20	22	23	25	26
13	9D	0	2	5	7	8	11	12	20	23	26									
13	18D	0	1	3	4	6	7	9	10	12	13	14	16	17	19	20	22	23	25	26
14	7D	0	2	3	10	16	18	19	26											
14	12D	0	1	2	3	4	11	13	15	17	18	20	24							
15	17D	0	1	4	5	6	8	9	11	12	14	15	17	18	20	21	23	24	26	
16	8D	0	5	6	8	11	14	15	21	26										
16	13D	0	1	2	3	7	10	12	13	15	18	21	23	24	26					
16	16D	0	2	3	5	6	8	10	11	13	14	16	17	19	20	21	22	26		
17	17D	0	1	4	5	6	8	9	11	12	14	15	17	18	20	21	23	24	26	
18	6D	0	7	9	11	16	22	26												
18	12D	0	1	4	7	9	10	12	16	18	19	21	24	26						
18	16D	0	2	3	5	6	8	10	11	13	14	16	17	19	20	22	23	26		
19	13D	0	2	4	5	7	8	9	15	16	18	19	22	25	26					
19	18D	0	1	3	4	6	7	8	10	12	13	14	16	17	19	20	22	23	25	26
20	6D	0	2	7	15	19	25	26												
20	18D	0	1	3	4	6	7	9	10	12	13	14	16	17	19	20	22	23	25	26

21	10D	0	2	5	9	10	11	13	14	16	18	26								
21	18D	0	1	3	4	6	7	9	10	12	13	14	16	17	19	20	22	23	25	26
22	18D	0	1	3	4	6	7	9	10	12	13	14	16	17	19	20	22	23	25	26
23	7D	0	2	5	8	11	15	22	26											
23	18D	0	1	3	4	6	7	9	10	12	13	14	16	17	19	20	22	23	25	26
24	8D	0	2	5	8	11	15	17	23	26										
24	10D	0	1	2	3	8	11	15	18	21	25	26								
25	11D	0	3	5	6	7	9	11	12	18	21	24	26							
25	18D	0	1	2	4	6	7	9	10	12	13	14	16	17	19	20	22	23	25	26
26	8D	0	1	2	5	11	15	20	22	26										
27	13D	0	3	4	6	8	9	11	14	16	18	19	21	24	26					
27	17D	0	1	3	5	6	8	9	11	12	14	15	17	18	20	21	23	24	26	
28	12D	0	2	4	5	7	10	12	15	17	18	20	22	26						
28	18D	0	1	3	4	6	7	9	10	12	13	14	16	17	19	20	22	23	25	26
29	6D	0	1	7	15	21	25	26												
29	6D	0	8	10	11	15	23	26												
29	14D	0	2	3	4	8	10	11	13	16	18	19	21	22	24	26				
30	16D	0	2	3	5	6	8	10	11	13	15	16	18	19	21	22	24	26		
30	16D	0	1	2	4	5	7	8	10	12	14	15	16	18	20	21	23	26		

References

[1] http://en.wikipedia.org/wiki/Tourism_carrying_capacity, Accessed Feb.10, 2012

[2] http://rafting.allaboutrivers.com/whitewater_rafting/Arizona_River_Runners-pol8 Accessed Feb.10, 2012

[3] http://www.wlu.edu/x11683.xml ,Accessed Feb.11, 2012

[4] http://www.villmarkskompaniet.no/en/Activities/Rafting/?article_ID=226, Accessed Feb.11, 2012

Appendix

- **Matlab Program:**

```
%Agent
function a=Agent(Y)
for i=6:18
    a(i-5,1)=1;
    for j=1:i
     a(i-5,j+1)=fix(j*Y/i)-fix((j-1)*Y/i);
    end
    s=sum(a(i-5,1:(i+1)));
    a(i-5,i+2)=Y-s+1;
    a(i-5,(i+3):19)=0;
end
```

```
%The Example of Basic Model
trips=3;
time=9;
Y=40;
count=0;
hold on;
perdistance=[1 7 3 4 5 6 8 2 5
             1 5 8 2 3 1 8 9 4
             1 4 2 7 4 5 8 2 8
             1 1 4 9 2 8 5 7 4
             1 1 8 1 5 7 3 10 5];
m=size(perdistance);
for i=1:m(1)
    for j=1:m(2)
    distance(i,j)=sum(perdistance(i,1:j));
    end
end
judgement=zeros(Y,time+m(2));
plot(0,0,'*-','LineWidth',3);
plot(0,0,'o-','LineWidth',3);
plot(0,0,'rs-','LineWidth',3);
plot(0,0,'g*-','LineWidth',3);
plot(0,0,'kd-','LineWidth',3);
legend('Trip1','Trip2','Trip3','Trip4','Trip5');
for i=0:time
    r=rand(1,m(1));
    [dm,rm]=sort(r);
    rd=rm(1:trips);
    for j=1:trips
        flag=0;
        for k=2:m(2)-1
            if judgement(distance(rd(j),k),i+k)==1;
                flag=1;
            end
        end
        if flag==0
            if rd(j)==1
            plot(i+1:i+m(2),distance(rd(j),1:m(2)),'*-','LineWidth',3);
            end
            if rd(j)==2
```

```
            plot(i+1:i+m(2),distance(rd(j),1:m(2)),'o-','LineWidth',3);
            end
            if rd(j)==3
            plot(i+1:i+m(2),distance(rd(j),1:m(2)),'rs-','LineWidth',3);
            end
            if rd(j)==4
            plot(i+1:i+m(2),distance(rd(j),1:m(2)),'g*-','LineWidth',3);
            end
            if rd(j)==5
            plot(i+1:i+m(2),distance(rd(j),1:m(2)),'kd-','LineWidth',3);
            end
        for k=2:m(2)-1
            judgement(distance(rd(j),k),i+k)=1;
        end
    count=count+1;
        end
        flag=0;
    end
end
count
grid on;
```

```
%Basic Model
clear;
clc;
time=30;
Y=25;
count=0;
s=0;
trips=fix(Y/6);
hold on;
perdistance=Agent(Y-1);
m=size(perdistance);
   plot(0,0,'*-','LineWidth',1.5);
   plot(0,0,'o-','LineWidth',1.5);
   plot(0,0,'r*-','LineWidth',1.5);
   plot(0,0,'y*-','LineWidth',1.5);
   plot(0,0,'g*-','LineWidth',1.5);
   plot(0,0,'ro-','LineWidth',1.5);
   plot(0,0,'kx-','LineWidth',1.5);
   plot(0,0,'ks-','LineWidth',1.5);
   plot(0,0,'gd-','LineWidth',1.5);
   plot(0,0,'cd-','LineWidth',1.5);
   plot(0,0,'go-','LineWidth',1.5);
   plot(0,0,'bd-','LineWidth',1.5);
   plot(0,0,'r*--','LineWidth',1.5);
legend('6days','7days','8days','9days','10days','11days','12days','13da
ys','14days','15days','16days','17days','18days');
 for i=1:m(1)
    for j=1:m(2)
    distance(i,j)=sum(perdistance(i,1:j));
    end
end
judgement=zeros(Y,time+m(2));
distance2=distance;
for i=0:time
    r=rand(1,m(1));
```

```
    [dm,rm]=sort(r);
    rd=rm(1:trips);
    for j=1:trips
        flag=0;
        for k=2:m(2)-1
            if judgement(distance(rd(j),k),i+k)==1;
                flag=1;
            end
        end
        if flag==0
            if rd(j)==1

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'*-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==2

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'o-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==3

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'r*-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==4

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'y*-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==5

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'g*-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==6

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'ro-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==7

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'kx-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==8

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'ks-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==9

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'gd-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==10
```

```
plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'cd-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==11

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'go-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==12

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'bd-','LineWidth',1.5);
                s=s+rd(j)+5;
            end
            if rd(j)==13

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'r*--','LineWidth',1.5);
                s=s+rd(j)+5;
            end
        for k=2:m(2)-1
            judgement(distance(rd(j),k),i+k)=1;
        end
        count=count+1;
        end
        flag=0;
    end
end
count
rate=s/((Y-1)*(time+m(2)-1))
grid on;
```

```
%Flexible Model(30Days)
time=30;
Y=26;
trips=fix((Y-1)/6);
count=0;
count2=0;
percount=zeros(13,1);
flag=0;
flag2=0;
location=0;
schedule=zeros(13,time);
hold on;
perdistance=Agent(Y-1);
m=size(perdistance);
for i=1:m(1)
    for j=1:m(2)
    distance(i,j)=sum(perdistance(i,1:j));
    end
end
judgement=zeros(Y,time+m(2));
axis([1 time+m(2) 1 Y]);
grid on;
for i=0:time-1
    distance2=distance;
    pause(0.01);
    r=rand(1,m(1));
    [dm,rm]=sort(r);
```

```
    rd=rm(1:trips);
    for j=1:trips
        flag=0;
        flag2=0;
        for k=2:(m(2)-1)
            if judgement(distance(rd(j),k),i+k)==1;
                if distance2(rd(j),k)<Y&&distance2(rd(j),k)>1
                    flag=1;
                    for q=0:(perdistance(rd(j),k+1)-1)
                        if distance2(rd(j),k)+q==Y
                          break;
                        end
                         if judgement(distance2(rd(j),k)+q,i+k)==0
                            location(1)=distance2(rd(j),k)+q;
                            location(2)=i+k;
                           judgement(distance2(rd(j),k)+q,i+k)=1;
                            if location(1)>1&&location(1)<Y
                                flag2=1;
                                flag=1;
                                distance2(rd(j),k)=distance2(rd(j),k)+q;
                                break;
                            end
                         end
                    end
                    for q=0:(perdistance(rd(j),k-1)-1)
                        if distance2(rd(j),k)-q==1
                          break;
                        end
                         if judgement(distance2(rd(j),k)-q,i+k)==0
                            location(1)=distance2(rd(j),k)-q;
                            location(2)=i+k;
                            judgement(distance2(rd(j),k)-q,i+k)=1;
                            if location(1)>1&&location(1)<Y
                                flag2=1;
                                flag=1;
                                distance2(rd(j),k)=distance2(rd(j),k)-q;
                                break;
                            end
                         end
                    end
                end
                if flag2==1
                    judgement(location(1),location(2))=1;
                    flag2=0;
                    flag=0;
                    count2=count2+1;
                end
            end
        end
        if flag==0
            percount(rd(j))=percount(rd(j))+1;
            for k=2:m(2)-1
                judgement(distance2(rd(j),k),i+k)=1;
            end
            if rd(j)==1

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'*-','LineWidth',1.5);
```

```
            end
            if rd(j)==2

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'o-','LineWidth',1.5);
            end
            if rd(j)==3

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'r*-','LineWidth',1.5);
            end
            if rd(j)==4

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'y*-','LineWidth',1.5);
            end
            if rd(j)==5

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'g*-','LineWidth',1.5);
            end
            if rd(j)==6

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'ro-','LineWidth',1.5);
            end
            if rd(j)==7

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'kd-','LineWidth',1.5);
            end
            if rd(j)==8

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'ks-','LineWidth',1.5);
            end
            if rd(j)==9

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'gd-','LineWidth',1.5);
            end
            if rd(j)==10

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'cd-','LineWidth',1.5);
            end
            if rd(j)==11

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'go-','LineWidth',1.5);
            end
            if rd(j)==12

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'yd-','LineWidth',1.5);
            end
            if rd(j)==13

plot(i+1:i+m(2),distance2(rd(j),1:m(2)),'c*-','LineWidth',1.5);
            end
            count=count+1;
            schedule(rd(j),i+1)=schedule(rd(j),i+1)+1;
        end
        if flag==1
            flag=0;
        end
    end
end
```

```
judgement
schedule
count
count2
percount
rate=sum(sum(judgement))/((Y-1)*(time+m(2)-1))
grid on;
```

```
%Draw pictures
function drawpictures(data)
m=size(data);
times=m(2)-3;
hold on;
grid on;
Y=26;
TIME=180;
axis([0 TIME+times+1 0 Y]);
       plot(0,0,'b*-','LineWidth',1.5);
       plot(0,0,'bo:','LineWidth',1.5);
       plot(0,0,'bs--','LineWidth',1.5);
       plot(0,0,'m*-','LineWidth',1.5);
       plot(0,0,'mo:','LineWidth',1.5);
       plot(0,0,'ms--','LineWidth',1.5);
       plot(0,0,'g*-','LineWidth',1.5);
       plot(0,0,'go:','LineWidth',1.5);
       plot(0,0,'gs--','LineWidth',1.5);
       plot(0,0,'r*-','LineWidth',1.5);
       plot(0,0,'ro:','LineWidth',1.5);
       plot(0,0,'rs--','LineWidth',1.5);
       plot(0,0,'y*--','LineWidth',1.5);
legend('6days','7days','8days','9days','10days','11days','12days','13da
ys','14days','15days','16days','17days','18days');
for i=1:m(1)
    if data(i,2)==6

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'b*-','LineWidth',1.5);
    end
    if data(i,2)==7

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'bo:','LineWidth',1.5);
    end
    if data(i,2)==8

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'bs--','LineWidth',1.5)
;
    end
    if data(i,2)==9

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'m*-','LineWidth',1.5);
    end
    if data(i,2)==10

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'mo:','LineWidth',1.5);
    end
    if data(i,2)==11

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'ms--','LineWidth',1.5)
```

```
;
    end
    if data(i,2)==12

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'g*-','LineWidth',1.5);
    end
    if data(i,2)==13

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'go:','LineWidth',1.5);
    end
    if data(i,2)==14

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'gs--','LineWidth',1.5)
;
    end
    if data(i,2)==15

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'r*-','LineWidth',1.5);
    end
    if data(i,2)==16

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'ro:','LineWidth',1.5);
    end
    if data(i,2)==17

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'rs--','LineWidth',1.5)
;
    end
    if data(i,2)==18

plot(data(i,1):(data(i,1)+times),data(i,3:m(2)),'y*--','LineWidth',1.5)
;
    end
end
```

- **C Program:**

```
////////////////////////////////////////////////////////////////////////////
/*
the Solution of Flexible Model
date:2012.2.12
vison 1.0
*////////////////////////////////////////////////////////////////////////////
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
#include <math.h>
#define N 200000 //the Max number of boats
#define GAP 1000 //the maximum quantity of tested boat
#define Y 25 //the number of campe sites
#define Day 180 //cycle
#define MAX 9 //the number of camping sites passed by

int day[N];//船只行驶天数
int path[N][20];//path[i][1..N]表示第 i 支船的路径；path[i][0]表示天数
```

```
bool boat[N]; //船只是否可以行驶
int kk[19];
int t[19][19];//t[i][j]表示露营 I 天的船行驶到第 J 天应该到哪个营地
int a[Y+1][Day+21];//a[i][j]表示第 j 天 i 号营地停留船只 a[i][j]号
int start[Day+21][30]; //start[k][0]表示第 k 天出发船只数量；start[i][j]表示第 i 天出发的第 j 支船的编号

//产生船只的随机生成模块
bool generate_rand()
{
	srand(time(NULL));
	for (int i = 0; i < N; i++)
		day[i] = rand() % 13 + 6;
}

//产生行驶 6-18 天船只在平均情况下每天应该到达的露营点编号
bool generate_t()
{
	for (int i = 6; i <= 18; i++)
	{
		for (int j = 1; j < i; j++)
		{
			float p = 1000; int q = 1000;
			for (int k = 1; k <= Y; k++)
			{
				if (fabs(225.0*(float)j/(float)i-225.0*(float)k/(float)(Y+1)) < p)
				{	p = fabs(225.0*(float)j/(float)i-225.0*(float)k/(float)(Y+1));
					q = k;
				}
			}
			t[i][j] = q;
		}
		t[i][i] = 26;
	}
}

//算法实现模块
bool drive()
{
	//初始化
	int l = 0, m = 0;
	memset(a, sizeof(a), 0);
	for (int j = 0; j <= Day; j++)
		a[0][j] = -1;
	memset(start, sizeof(start), 0);
	memset(boat, sizeof(boat), 0);
	for (int i = 0; i < N; i++)
		boat[i] = false;
      for (int i = 0; i <= Day; i++)
	{
		m++;
		for (int j = m*GAP; j < (m+1)*GAP; j++)
```

```
        {
            //安排每只船
            if (boat[j] == false)
            {
                        path[j][0] = 0;
                        bool flag = true, flag2;
                        for (int k = 1;k < day[j]; k++)
                        {
                            if (a[t[day[j]][k]][i+k] == 0 && (t[day[j]][k] - path[j][k-1] < MAX)
                                && (t[day[j]][k+1] - t[day[j]][k]) < MAX)
                            {
                               path[j][k] = t[day[j]][k];
                               continue;
                            }
                            else //解决冲突
                            {
                               int l = 0,r = 0;
                               if (k == 1) l = 1;
                               else l = path[j][k-1]+1;
                               r = t[day[j]][k+1]-1;

                               flag2 = false;

                               //使用蒙特卡罗算法解决冲突
                               int step = 1000;
                               int tt;
                               while (step--)
                               {
                                        if (r-l > 0)
                                            tt = rand() % (r-l+1) + l;
                                        else tt = r;

                                        if (a[tt][i+k] == 0 && (tt - path[j][k-1] < MAX) && (t[day[j]][k+1] - tt) < MAX)
                                        {
                                            path[j][k] = tt;
                                            flag2 = true;
                                            break;
                                        }
                               }
                            }
                            if (k< day[j]-1 && flag2) continue;
                            flag = false;
                            if (flag2 == true)    flag = true;
                            break;
                        }
                        if (flag)
                        {
                                    //该船可以行驶，记录该船的路径
                                    boat[j] = true;
                                    for (int k = 1; k < day[j]; k++)
                                    {
```

```
                                                a[path[j][k]][i+k] = j;
                                            }
                                        }
                    path[j][0] = i;//假设这天发该船
                    }
                }
            }
}

//主模块
int main()
{
    freopen("out.txt", "w", stdout);
    generate_rand();
    generate_t();
    memset(kk, sizeof(kk), 0);
    for (int i = 6; i <= 18; i++)
    {
        for (int j = 1; j < i; j++)
        {
            printf("(%d,%d)=%d; ",i,j,t[i][j]);
        }
        printf("\n");
    }
    drive();
    int zy = 0;
    for (int i = 0; i < Y; i++)
    {
        for (int j = 0; j < Day; j++)
        {
            if (boat[a[i][j]])
            {
                printf("%4d", a[i][j]);
                if (a[i][j] != 0) zy++;
             }
             else
                 printf("%3d", 0);
        }
        printf("\n");
    }
    int tot = 0;
    for (int i = 0; i < N; i++)
    {
        if (boat[i])
        {
                    tot++;
                    kk[day[i]]++;
        }
    }
    printf("\ntot = %d          %f", tot, (float)zy/(float)(Day*(float)Y));
    for (int i = 6; i <= 18; i++)
        printf("\n%dDay：%d;", i, kk[i]);
    printf("\n");
```

```
    //打印
    printf("day\tpath");
    for (int d = 0; d <= Day; d++)
    {
        for (int j = 1; j <= N; j++)
        {
            if ((boat[j] == true) && (path[j][0] == d))
            {
              printf("\n%d %d          0 ", d, day[j]);
              for (int k = 1; k < day[j]; k++)
                  printf("%d ", path[j][k]);
              for (int k = 0; k <= 18 - day[j]; k++)
                  printf("%d ",Y+1);
              }
        }
    }
    return 0;
}
```