Scientific Research Publishing

Open Journal of Applied Sciences, 2019, 9, 515-534
http://www.scirp.org/journal/ojapps
ISSN Online: 2165-3925
ISSN Print: 2165-3917

Analysis of Evacuation of Tourists Based on the Louvre's Emergency

Rui Tang, Xiaozhen Luan, Shiwei Xu, Fuliang Lu, Wei Zhang

Linyi University, Linyi, China
Email: 2352935491@qq.com

How to cite this paper: Tang, R., Luan, X.Z., Xu, S.W., Lu, F.L. and Zhang, W. (2019) Analysis of Evacuation of Tourists Based on the Louvre's Emergency. *Open Journal of Applied Sciences*, **9**, 515-534.
https://doi.org/10.4236/ojapps.2019.96041

Received: May 9, 2019
Accepted: June 27, 2019
Published: June 30, 2019

Abstract

Due to France has suffered from many terrorist attacks and the number of visitors to the Louvre has gradually increased in recent years, a good evacuation plan for the Louvre is of vital significance. We use the minimization of the total evacuation time of all tourists as the optimization goal to find an optimal path. For conventional emergencies, a static model is built to evacuate visitors. And then we establish a nonlinear programming model. Using Lingo software, we get the distribution information of the visitors in different exhibition halls. For unconventional emergencies, we establish an adaptive dynamic model of tourist evacuation based on genetic algorithm. The sensitivity analysis of the model is considered by adding new paths. By solving the nonlinear programming problem with the double objective function of maximizing evacuation time and balancing the number of people in every path, we get the evacuation time last 1582.74 s. Finally, according to our result, we built mathematical models for the evacuation after an emergency and analyzed how to adapt and implement our models for other large and crowded structures.

Keywords

NT Model, Linear Programming, Genetic Algorithm, Cellular Automata, M/G/C/C Queuing Network Model, Emergency, Evacuation

1. Introduction

1.1. Background

With the French government's policy of stimulating tourism and the 2015 Paris terrorist attacks, in 2017, a total of more than 8.1 million tourists visited the Louvre. So the security of the visitors and the aftermath of the emergency for

visitors are becoming more and more important. This paper requires the establishment of mathematical models for the evacuation after an emergency.

1.2. Restatement

We divided the task into four parts.

First of all, we built a mathematical model to simulate the situation after an emergency, in order to evacuate visitors from the Louvre and identify bottlenecks.

Secondly, we simulate the possible unexpected events and verify the correctness of the model in the first part. In addition, we discuss how the Louvre implements this model.

Thirdly, through the two-part model described above, we made some suggestions for the Louvre staff. To ensure that visitors can evacuate safely when an emergency occurs.

Finally, we discussed how to adapt and implement our models for other large and crowded structures.

1.3. Analysis

This question requires us to establish an emergency evacuation model. We can classify events into two categories based on the condition of damage to the evacuation path: conventional emergencies and unconventional emergencies. As for regular emergencies, we can build static models to evacuate people. For conventional emergencies, we can build a static model to carry out visitor evacuation allocation. For unconventional emergencies, an adaptive dynamic model can be built. We can simulate the flow of people in different emergency situations and define the bottlenecks of this system.

2. Assumptions

In order to simplify the course of modeling and draw some reasonable conclusions from our model, we have the following assumptions.

- We suppose that tourists from all countries follow the instructions of security guards when a threat happens.
- Other available exit points cannot be used by tourists.
- The Louvre has enough security guard to guide the evacuation of tourists at the emergency.
- Before the evacuation of tourists, tourists are A uniform distribution in each exhibition hall. To simplify our model, we assume that the number of people in each exhibition hall is the same.
- In case of emergency, the elevator will stop and the escalator can be used as ordinary stairs.
- Four main evacuation exits are RICHELIEU corridor, SULLY corridor, DENON corridor and Pyramid corridor. Three of the exits are on the zero level and one on the second floor.

3. Notations

The notation table contains all the notations we use in this paper.

Symbol	Definition	Notes
T_{ij}	the time that the visitor arrived	$i=1,2,\cdots,12;\ j=1,2,\cdots,445$
v	The speed of visitors	
d_{ij}	The shortest path between the exhibition hall j and the path i	$i=1,2,\cdots,12;\ j=1,2,\cdots,445$
t_j	The time for the exhibition hall j to arrive at the paths	
f_j	The task amount of the path	$j=1,2,\cdots,256$
$\overline{s}$	The variable number of paths.	
$\min Z$	Minimize the standard of the tasks	
P_nA	The nth floor of the RICHELIEU	
P_nB	The nth floor of the SULLY	
P_nC	The nth floor of the DENON	
$arrive(t)$	THE number of arrive visitors	
ρ	the density of visitor	
$R(t)$	The total number of people in the Louvre	
$\Delta R(t)$	The variable number of visitors	
S	The total area of the corridors	
q	The corrected speed parameter	
L	The total length of stairs	
Δt	Evacuation time	
u	Throughput	
w	Distance	
D	The area of the exhibition hall	
ρ_s	Service intensity	
c	The maximum bearing capacity of visitors	
D	The serviced accommodate of visitors	

4. The Basic Model of Security Evacuation

In this section, we established a no-wait time model and a queuing-based M/G/C/C model for finding bottlenecks and calculating the longest time.

4.1. How to Plan a Designated Escape Route

4.1.1. The Calculation of the Shortest Path by Floyd Algorithm

The Floyd algorithm calculates the shortest path between exhibition halls and corridors. In the first place, the initial distance matrix $B(i,j)$ is established based on the connection information between some known nodes. Here, a corridor that is not a path is given a sufficiently large value to facilitate updating.

$(i, j = 1, 2, \cdots, n)$

The second step is calculated iteratively. For any two points (i, j), if there is a variable *k*, making $B(i,k) + B(k, j) < B(i, j)$, then update $B(i, j) = B(i,k) + B(k, j)$. Keeping calculating until the distance of all points is no longer updated or stop counting. By doing this, we get the shortest distance matrix $B(i, j), (i, j = 1, 2, \cdots, n)$.

The program coding of the algorithm is shown in **Appendix I**.

4.1.2. Path Distribution Plan

We divide the plan of distributing an evacuation route for visitors in the exhibition hall into two steps, while introducing a variable T_{ij}. We define variable T_{ij} as the time that the visitor arrived at the corridor.

$(i = 1, 2, \cdots, 12; j = 1, 2, \cdots, 445)$.

In the first place, for the exhibition hall *j* that visitors arrive at the corridor time $T_{ij} < 166$ s, its visitors leave through the corridor *i*.

The second step, for the corridor of arriving at the passage time $T_{ij} > 166$ s, the optimization is based on the principle of balancing the amount of tasks assigned to each corridor as much as possible.

1) Our model plans for the exhibition halls of variable $T_{ij} < 166$ s.

Through the Louvre plane map, there are 445 exhibition halls, which are numbered as $j = 1, 2, \cdots, 445$. There are two stairs on the second floor. There are three stairs on the negative floor and the first floor. There are four stairs on zero floor. We refer to each staircase as a path. There are totals of twelve paths, which are numbered as $i = 1, 2, \cdots, 12$. Our model has a total of 256 exhibition halls by calculation, which visitors arrive at the corridor time $T_{ij} < 166$ s. What has been discussed above, we can easily get the path information in **Table 1**.

Table 1. Path information.

Exhibition halls	Assigned path	Longest theoretical time (s)
100, 101, 104, 105	−1A	138
130, 131, 132, 137	−1B	158
160, 170, 186, 164, 169, 173, 183	−1C	154
206 - 212	0A	152
230 - 236	0A	152
317 - 348	0B	145
417 - 424	0C	136
526 - 548	1A	165
632 - 640	1B	158
720 - 734	1C	146
830 - 852	2A	149
920 - 940	2B	164

Therefore, the 256 corridors are assigned to the nearest path, and the remaining 189 corridors require the minimum distance as little as possible.

2) Our model plans for the exhibition halls of variable $T_{ij} > 166$ s.

Our goal is to minimize the standard deviation of the amount of tasks that assigned to each path. To sum up, we get the following model.

$$\min Z = \sqrt{\frac{\sum_{i=1}^{12}\left(s_i - \overline{s}\right)^2}{n_b - 1}} \tag{4.1}$$

$$s.t \begin{cases} \sum_{j=1}^{12} x_{ij} \geq 1, \quad i = 1,2,\cdots,12 & (1) \\ t_j = \sum_{i=1}^{12} \frac{x_{ij} d_{ij}}{v}, \quad j = 1,2,\cdots,445 & (2) \\ s_i = \sum_{j=1}^{256} x_{ij} f_j, \quad i = 1,2,\cdots,12 & (3) \\ x_{1,28} = 1, x_{1,29} = 1, x_{1,38} = 1, x_{1,39} = 1, x_{1,61} = 1, \cdots & (4) \\ t_j > 166 \text{ s}, \quad j = 1,2,\cdots,445 \text{ and } j \neq 28,29,38,39,61,92,\cdots & (5) \\ \sum_{i=1}^{12} x_{ij} = 1, \quad j = 1,2,\cdots,445 & (6) \\ \overline{s} = \frac{\sum_{i=1}^{12} s_i}{n_b} & (7) \\ x_{ij} = 0 \text{ or } 1, \quad i = 1,2,\cdots,12; j = 1,2,\cdots,445 & (8) \end{cases}$$

(1): $\sum_{j=1}^{12} x_{ij} \geq 1, i = 1,2,\cdots,12$ means that at least one exhibition hall is arranged for each exhibition hall.

(2): $t_j = \sum_{i=1}^{12} \frac{x_{ij} d_{ij}}{v}, j = 1,2,\cdots,445$ means that the time when the exhibition hall *j* arrived at the path by the restrictions.

(3): $s_i = \sum_{j=1}^{256} x_{ij} f_j, i = 1,2,\cdots,12$ means that the task quota that the corridor has been assigned to.

(4): $x_{1,28} = 1, x_{1,29} = 1, x_{1,38} = 1, x_{1,39} = 1, x_{1,61} = 1, \cdots$ means that 266 corridors that have reached the evacuation path less than 166s.

(5): $t_j > 166$ s, $j = 1,2,\cdots,445$ and $j \neq 28,29,38,39,61,92,\cdots$ means that the path allocated by the remaining 189 exhibition halls.

(6): $\sum_{i=1}^{12} x_{ij} = 1, j = 1,2,\cdots,445$ means that the only evacuation path can be assigned to each exhibition hall.

(7): $\overline{s} = \frac{\sum_{i=1}^{12} s_i}{n_b}$ means that the average of the path's task load.

(8): $x_{ij} = 0$ or $1, i = 1,2,\cdots,12; j = 1,2,\cdots,445$ means that the decision variable.

$$x_{ij} = \begin{cases} 0 & \text{The } j\text{-th pavilion is assigned to the } i\text{-th channel} \\ 1 & \text{The } j\text{-th booth is not assigned to the } i\text{-th channel} \end{cases} \tag{4.2}$$

Considering that the objective function is nonlinear, it is not easy to find its solutions. Thus we convert the objective function to the following formula.

$$\min Z = \max_{1 \le i \le 12} s_i \tag{4.3}$$

Then the model is expressed linearly as the following expression.

$$\min Z = TT \tag{4.4}$$

4.1.3. Distribution Result

We will refer to the exhibition hall assigned to the numbered i path as the exhibition area $p_{ij}\ (j = 1, 2, \cdots)$.

According to the equal distribution path model, we distribute the remaining exhibition halls. The final exhibition halls allocation information are as follows. Table 2 is the exhibition area allocations.

4.2. Identify Bottleneck Model

In this section, we introduce two models for identifying the bottleneck. The first model is an optimal but not practical model (NT model). This is just a comparison and the purpose is to provide a modified version. The second is M/G/C/C queuing network model that simulates the Louvre real evacuation system.

4.2.1. No-Wait Time Model (NT Model)

The Louvre is a whole consisting of escalators, stairs, corridors and exhibition halls. The facilities and connections within the Louvre can be considered as a network. If each node of the network is regarded as a relatively independent queuing system, then the whole Louvre can be regarded as a queuing network.

We divided each floor into three zones. The P_nA, P_nB, P_nC represents the RICHELIEU, the SULLY, and the DENON on the nth floor. Subsequently, we numbered the corridors. The zones A, B, C correspond to the path that the nth floor of the RICHELIEU, the SULLY, and the DENON. The evacuation system is shown in Figure 1.

Table 2. Exhibition area allocations

Exhibition hall	Assigned path	Exhibition area
100 - 106	−1A	P_{-1j}
130 - 137	−1B	P_{-1j}
160 - 187	−1C	P_{-1j}
206 - 236	0A	P_{0j}
300 - 348	0B	P_{0j}
400 - 433	0C	P_{0j}
500 - 564	1A	P_{1j}
600 - 663	1B	P_{1j}
700 - 734	1C	P_{1j}
800 - 864	2A	P_{2j}
900 - 952	2B	P_{2j}

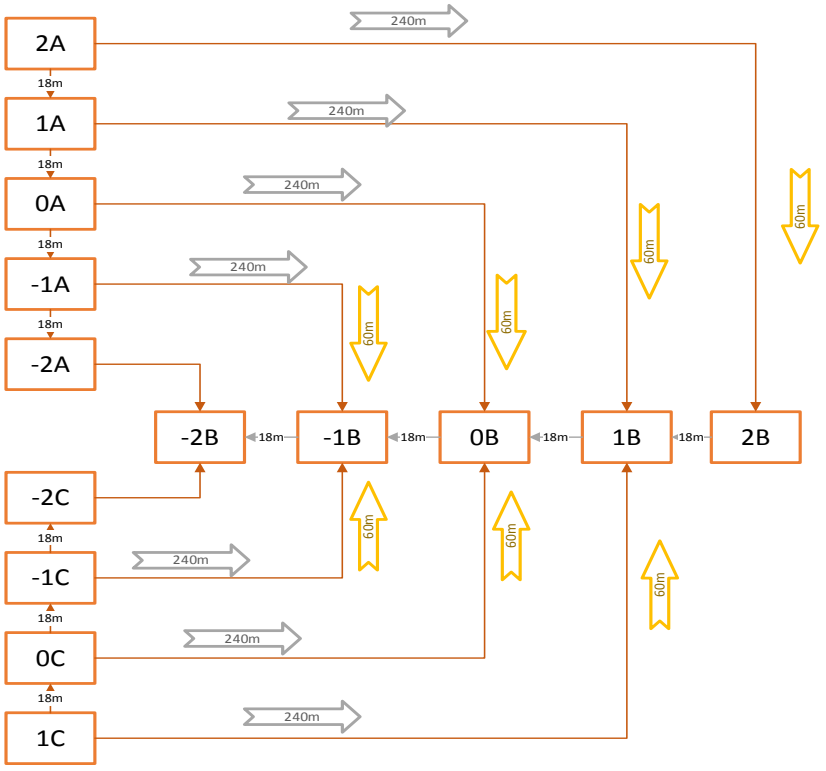

Figure 1. Evacuation system diagram.

In addition, the visitors in a zone are treated as a whole. These people are called visitors to the area. The corridors, stairs, and escalators are set as service organizations to construct the Louvre queuing network system. According to the 4.1 plan of designated escape route, it is stipulated that visitors from each zone can only walk through the corridor of the zone. The network model of the system can be described as this. When the number of people on the evacuation path exceeds the maximum bearing capacity, it enters the congestion state. At this time, the visitor has to enter the queue waiting state. After a series of escalators or corridors of the queuing subsystem, the evacuated visitors can reach the Louvre exit to the safe area. At that moment, evacuation service completed.

In the first place, we propose an ideal model which is an NT model. The model assumes that congestion will not occur. In other words, the waiting time in the model is zero and the model achieves the maximum throughput. Nevertheless, it is not feasible in real life. Later we will compare the performance of the ideal model with the performance of the actual model to evaluate the latter and modify it.

4.2.2. Building the NT Model

By consulting literature, we determined that the number of people arriving at the Louvre obeys the Poisson distribution [1]

$$P(t)=\frac{\lambda^t}{t!}\mathrm{e}^{-t},\ t=1,2,\cdots \tag{4.5}$$

We simulated the number of visitors $arrive(t)$ of the Louvre in any unit time t by MATLAB. Finally the total number of people in the Louvre in any time is $\sum_{t=0}^{t=t_0} arrive(t)$

$$\begin{cases} R(t) = \sum_{t=0}^{t=t_0} arrive(t) \\ \Delta R(t) = \dfrac{R(t)}{n} \\ \rho = \dfrac{\Delta R(t)}{S} \\ v = \dfrac{e^q}{\rho} \\ T_{\max} = \dfrac{L}{v} \end{cases} \tag{4.6}$$

here we assume that the number of visitors in each exhibition hall is subject to Uniform distribution. Denoting the visitor density within the system by ρ. The $R(t)$ represents the total number of people in the Louvre at t time. The variable n is the number of current regional exhibition halls. Denoting the number of visitors in any exhibition hall by variable $\Delta R(t)$. The S represents the total area of the current corridors and stairs in the area. The q represents the corrected speed parameter. The L represents the total length of the passage and stairs in the area.

4.2.3. The Solving of NT Model

In order to show the relationship between the walking speed of visitors and the density of visitors during the evacuation process, we investigated many museum data. Through linear and exponential fitting, it is found that the exponential fitting is better [2]

$$R^2 = 0.998, \quad q = 0.62$$

Looking at the Louvre data, we find $L = 256$ m. [3] Due to the largest number of P_2A exhibition halls, we choose the evacuation time for the whole system as the evacuation time. So, we can get this result $T_{\max} = 342.67$ s.

4.2.4. NT Model Analysis

NT model is idealized. On the one hand, congestion is possible. And the waiting time cannot be zero. On the other hand, because $P_1 < P_2$, we assume that P_2 is completely evacuated while P_1 is evacuated. This model is impossible due to various emergencies and congestion situations.

4.3. M/G/C/C Queuing Network Model

4.3.1. The Building of the M/G/C/C Model

Now adding waiting time based on no-waiting time to build the M/G/C/C model. The M is the Poisson distribution for the visitor's arrival time. The G indicates

that the service time which is subject to the general distribution. Using this model to simulate the evacuation of the Louvre and identify bottlenecks [4] [5]

- Service Time

First, we need to solve the evacuation time $\Delta t = \frac{\Delta R(t)}{u}$ as the ratio of the number of people in this zone to the throughput of the service system $u = \frac{w\rho\Delta R(t)v}{L}$.

Denoting the evacuation time in the corridor and the escalator by the Δt. Defining v as the evacuation speed of visitors. The variable u is the throughput of the service system. The variable w is the lateral distance of the visitor from the evacuation stairs. Denoting the total area of the exhibition hall by the D.

Subsequently, we solved the change of visitors by evacuation time, and determined the service density by the amount of change of visitors.

- Service Density

$$\begin{cases} c = \dfrac{D}{\rho} \\ \rho_s = \dfrac{\Delta R(t) - \Delta t u}{cu} \end{cases} \tag{4.7}$$

The ρ_s means service intensity. Defining c as maximum visitor density bearing capacity in the current area. In this way, we quantify the bottleneck through ρ_s. When an emergency occurs, the density ρ of visitors who go to each exhibition hall will change. It will cause a change in service intensity ρ_s. We select areas with higher service intensity ρ_s as candidate areas for defining bottlenecks.

- The average waiting time

Now, we calculate the average waiting time W_p for each path of the Louvre. Assume that the path can accommodate the serviced visitors are $c = \frac{D}{\rho}$. Then we divided the visitors into two parts. There are $\frac{D}{\rho}$ visitors is zero. There are $\Delta R(t)\left(1 - \frac{D}{\rho}\right)$ visitors is not zero. Then divide the tourists in this place into k areas.

$$k = \frac{\Delta R(t)\left(1 - \frac{D}{\rho}\right)}{w\rho} \tag{4.8}$$

It is clear that the waiting time of visitors in each area is the following expression.

$$t_\Delta = \frac{L\rho\Delta R(t)}{v} \tag{4.9}$$

Eventually, the following models can be obtained.

$$\begin{cases} t_{\Delta} = \dfrac{L\rho\Delta R(t)}{v} \\ k = \dfrac{\Delta R(t)(1-D\rho)}{w\rho} \\ W_s = t_{\Delta}\sum_{n=1}^{k} n \\ W_p = \dfrac{\sum_{s=1}^{n} W_s}{n} \end{cases} \tag{4.10}$$

The variable W_s is the waiting time of a path. The variable k is an intermediate parameter. The variable t_{Δ} is the middle time.

4.3.2. The Solving of M/G/C/C Model

We use the actual situation and MATLAB to solve the model. And we use the cellular automata to simulate the exhibition hall. We choose the first floor for evacuation simulation. Thinking of each exhibition hall as a cell, and only use the movement speed as a constraint.

The simulation of the cellular automata on the first floor is shown in **Figures 2-4**.

Result Analysis: The red line in the figure indicates the walking path of the visitor. When the slope is zero, it means that the visitor has not moved. It is crowded at that moment. The path of the RICHELIEU on the first floor is the most congested, and the evacuation time is about 1388.4 s.

In short, the path of the RICHELIEU on the first floor is the evacuation bottleneck.

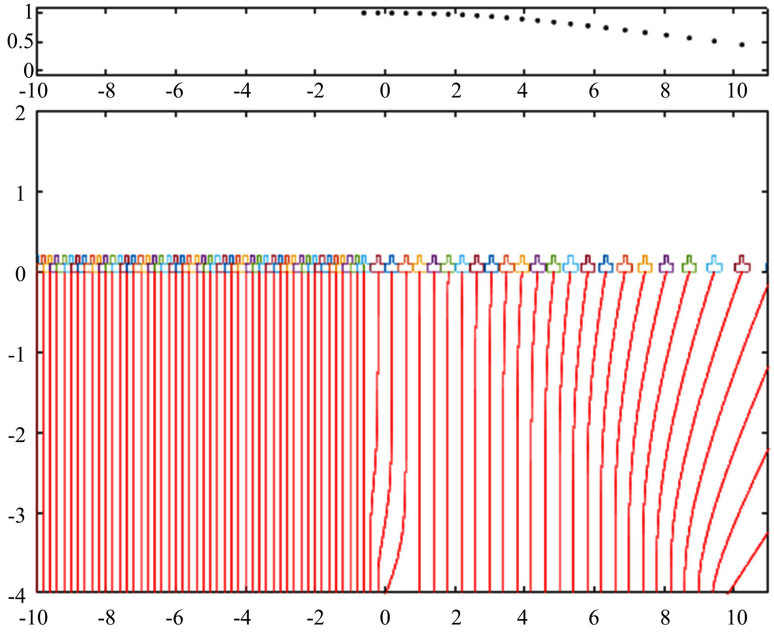

Figure 2. Simulation of the RICHELIEU on the first floor. As we can see from the picture, **the first floor of the RICHELIEU is very crowded.**

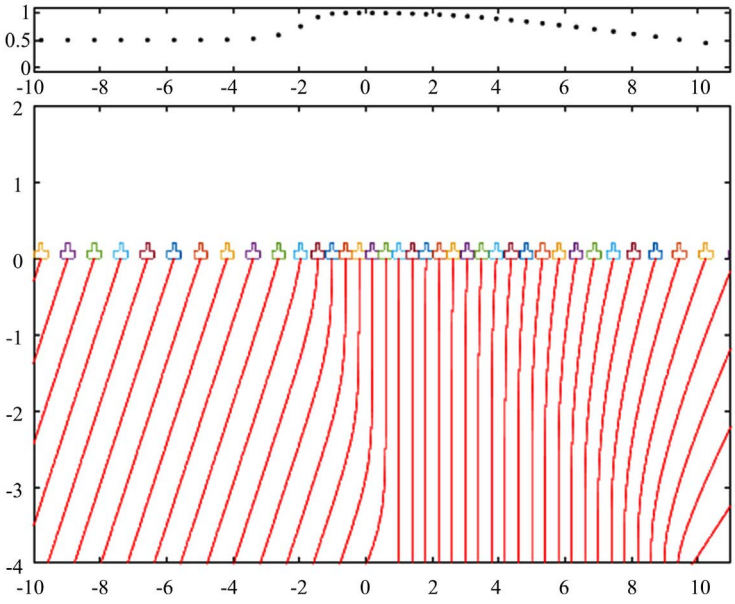

Figure 3. Simulation of the SULLY on the first floor. As we can see from the picture, **the first floor of the SULLY is relatively crowded.**

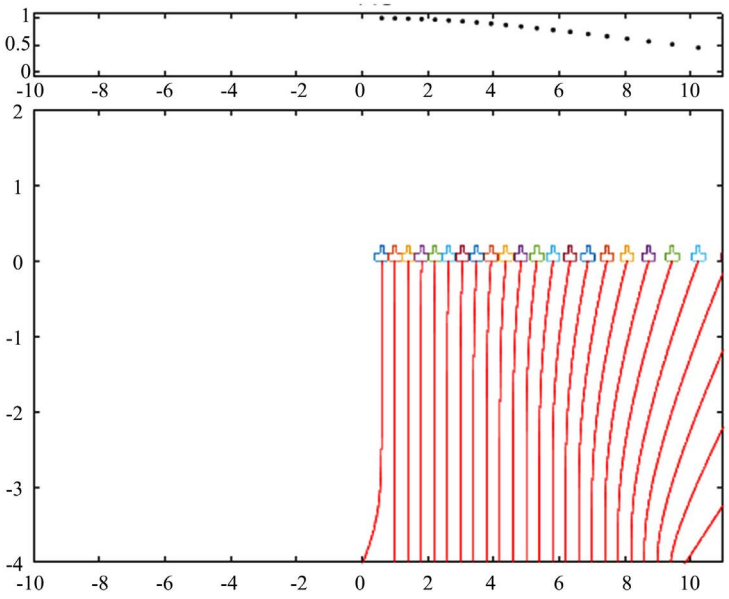

Figure 4. Simulation of the DENON on the first floor. As can be seen from the figure, **the DENON on the first floor is relatively uncrowded.**

4.4. Adaptive Genetic Algorithm Model

4.4.1. Improved Genetic Algorithm to Solve the Problem of TSP

Application of Improved Genetic Algorithm to Solve TSP Problem in Dynamic Evacuation Model. We make the following improvements to the traditional model:

- The result of the traditional TSP problem is a loop. We improved the genetic algorithm and now we are looking for a path from the starting point to the exiting point.

- Traditional TSP issues require that each city can only be visited once. The improved model can traverse the nodes repeatedly.
- The traditional TSP problem uses the shortest path between two nodes as the fitness function. The improved model uses "equivalent length" as a fitness function.

$$L_d = Lf \tag{4.11}$$

The variable L is the geometric length. The variable f is the impact index.

$$f = \Delta R_n(t) \tag{4.12}$$

The variable $\Delta R(t)$ is the number of people at the n node at the t time.

4.4.2. Improved Genetic Algorithm Steps

We do the flow chart design of genetic algorithm in Figure 5. Traverse all nodes, we can get the optimal route matrix for each node at present.

The calculation steps of the improving genetic algorithm for the traditional TSP problem are as follows: [6] [7].

1) Gene setting. First, we number each path $\{1,2,\cdots,14\}$. In addition, for the convenience of solving, we turn the multi-source and multi-end (multi-exit) problem into a multi-source and single-end problem. Then set the ending point number to 15, connect negative two layers of "A" area (number 1) path negative two layers of "B" area (number 6) path and negative two layers of "C" area (number 11) path. And set $d_{ij} = 0$.

2) Initial population. Through genetic algorithms, we want to find the shortest "equivalent length" path of each node at different times. So we decompose the multi-source and single-end point problem into fourteen (fourteen paths) single-source and single-end point problems. The traditional genetic algorithm randomly generates a feasible solution as the initial population when initializing the population. So in order to ensure that the starting point and the end point are fixed, we only let the middle thirteen columns be randomly generated. The first column of the loop sets a fixed number $(1,2,\cdots,14)$. The last column is fixed as $N\text{ind}\times 1$. $N\text{ind}\times 1$ is a matrix that fixed to 15 for all of them. In addition, when generating random numbers, we generate repeatable numbers through the "randi" function in MATLAB.

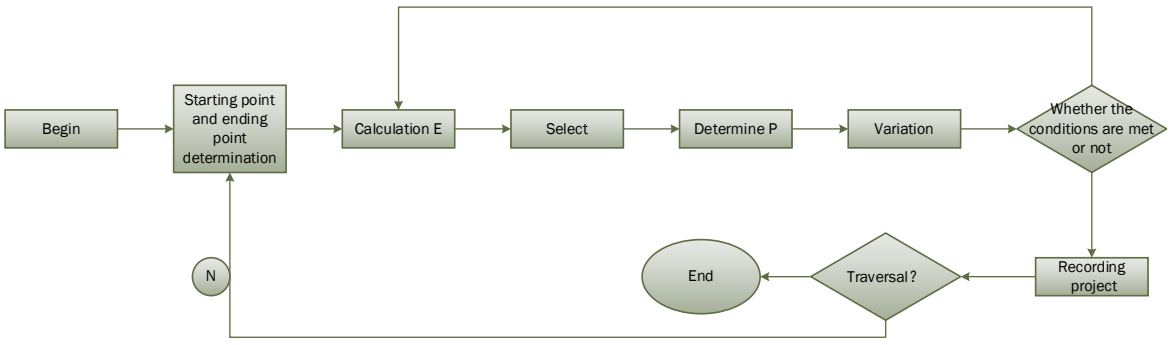

Figure 5. Flow chart design of genetic algorithm.

1) Select operation. Using a deterministic selection strategy to select individuals with greater fitness functions to evolve to the next generation. In order to ensure that the superior genes of the father are passed on to the offspring. Here the fitness function is "equivalent length " $L_d = Lf = L\Delta R_n(t)$.

2) Crossover operation. In the case of ensuring that the genes are not repeated, the starting point and ending point are unique. Simultaneously, the first column and the last column do not participate in the crossover operation. So we only randomly generate intersections p in the middle thirteen columns. Copying the left code of the parent code intersection to the left of the child code. The right side of the children's intersection one is sequentially selected from the parent two to select the code that is not used by the child. And the code of the child two is also generated.

3) Mutation operation. Variation operation can realize population diversification. It is easier to pick out the local optimal solution. And this is the guarantee of global optimization. Therefore, setting the mutation rate is significant, and the Coefficient of Variation is $P_m = 0.05$.

4) If conditions are met, then the algorithm is an end. Recording the optimal solution and putting it as a row vector. If the conditions are not met, we should repeat the Step 3 to Step 5.

4.4.3. The Design of Personnel Evacuation System

1) Thinking

When the terrorist attack, fires, earthquakes, occur, this can lead to road blockage. It causes road impassability. Therefore, it is not feasible to rely solely on the M/G/C/C model. And in order to solve the problem, we use the genetic algorithm to reach fitness function $L_d = L\Delta R_n(t)$. Thus, the optimal path matrix of the current node is obtained. Because of the number of visitors keeps changing, the function is keeping changing. At the same time, the optimal path matrix is keeping changing too. To address this situation, we design the algorithm.

Step 1: for the nodes with incompatible input paths, inputting the number of customers to the current system.

Step 2: we get the optimal path by genetic algorithm.

Step 3: by judging if the number of people at the last node is equal to the number of people in the current system. If so, we end the algorithm. Otherwise, looping through the row vectors in the matrix. In other words, it is traversing each node.

Step 4: determining if the next node is the end point.

Step 5: if it is not the end point, then go to step 6. Otherwise, the node loop ends.

Step 6: determining if the speed is greater than 1.2 m/s. If the conditions are met, we proceed to step 7. We record the track list as $[i,i]$.

Step 7: then moving to the next node, updating the number of nodes, and record the track list as $[i,j]$.

2) Flowchart

We do the Personnel evacuation system algorithm flowchart in Figure 6.

3) Solution

We assume that an incident occurred at the node with layer 1 number 4 and solved by the above model. And the movement path of each node can be seen in Table 3.

4) Algorithm analysis

If an emergency occurs at four nodes, then the adjacent node path is set to infinity. Through the results of the algorithm, it is easy to see that the node movement path at the negative point, zero point and two point are consistent with the escape route we specified in 4.1.

Only in layer 1 and 2 did the moving path change. This fully illustrates the applicability of our static model in slight emergencies.

5. Sensitivity Analysis

In this section, we are based on the imbalance of task load in existing paths and the actual situation of congestion in some places. We want to try to add some paths and study the effect of increasing the number of paths on throughput. We assumed that k paths are added, in this way, there are $14+k$ paths which have added in the Louvre.

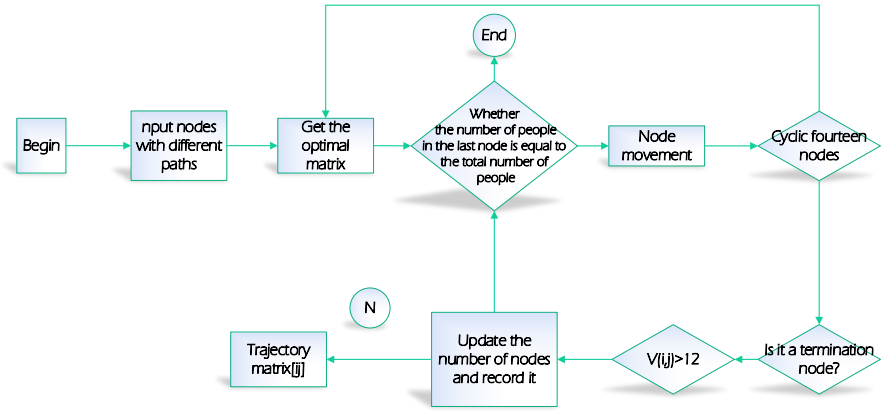

Figure 6. Personnel evacuation system algorithm flowchart.

Table 3. Movement path of each node.

Layer number	Movement path of each node
−1	**2→1→0, 7→6**
0	**3→2→1, 8→7→6**
1	**4→9→8→3→2→1**
1	**9→8→7→6**
2	**5→10→9→8→3→2→1**
2	**10→9→8→7→6**

Then we built the multi-goal programming model.

$$\min Z_1 = \max_{1\le j\le 445} T_j \tag{5.1}$$

$$\min Z_2 = \sqrt{S/(n+k-1)} \tag{5.2}$$

$$s.t.\begin{cases} \sum_{i=1}^{445} x_{ij} = 1,\ j = 1,2,\cdots,445 & (9) \\ \sum_{i=1}^{445} y_i = 14+k & (10) \\ w_i = \sum_{j=1}^{445} x_{ij} f_j, i = 1,2,\cdots,445 & (11) \\ \overline{w} = \dfrac{\sum_{i=1}^{445} w_i}{14+k} & (12) \\ S = \sum_{i=1}^{445} w_i^2 - (14+k)\cdot \overline{w}^2 & (13) \\ T_j = \sum_{i=1}^{445} \dfrac{x_{ij} d_{ij}}{v},\ \ j = 1,2,\cdots,445 & (14) \\ T_j \le TT,\ \ j = 1,2,\cdots,445 & (15) \\ y_i = 1,\ \ i = 1,2,\cdots,14 & (16) \\ x_{ij} = 0 \text{ or } 1,\ \ i, j = 1,2,\cdots,445 & (17) \\ y_i = 0 \text{ or } 1,\ \ i = 21,\cdots,445 & (18) \end{cases}$$

(9): $\sum_{i=1}^{445} x_{ij} = 1,\ j = 1,2,\cdots,445$ means the jth exhibition hall that needs to use.

(10): $\sum_{i=1}^{445} y_i = 14+k$ means the total number of the path.

(11): $w_i = \sum_{j=1}^{445} x_{ij}\cdot f_j, i = 1,2,\cdots,445$ means the amount of tasks that the ith path needs to process.

(12): $\overline{w} = \dfrac{\sum_{i=1}^{445} w_i}{14+k}$ means average task load every path.

(13): $S = \sum_{i=1}^{445} w_i^2 - (14+k)\cdot \overline{w}^2$ means the squares of the sum about task load.

(14): $T_j = \sum_{i=1}^{445} \dfrac{x_{ij} d_{ij}}{v}, j = 1,2,\cdots,445$ means the time of police arrive time.

(15): $T_j \le TT, j = 1,2,\cdots,445$ means less than reaction time.

(16): $y_i = 1, i = 1,2,\cdots,14$ means the original fourteen path.

(17): $x_{ij} = 0$ or $1, i, j = 1,2,\cdots,445$ means if the x_{ij} visitors can pass.

(18): $y_i = 0$ or $1, i = 21,\cdots,445$ means whether to choose the intersection near the exhibition hall as a path.

The model implementation program by LINGO is b2011_3.lg4 in **Appendix II**.

When the number of paths is increased by $k = 2$, the added path is the second floor exhibition hall at A area. The shortest time is 1582.74 s. It reduces 194.34 s

than the M/G/C/C model. This shows that increasing the paths can reduce the evacuation time obviously. But increasing the number of path, not only will increase spending, but also increase security personnel. Museum leaders can choose this appropriately.

6. Suggestions to the Security Manager

According to museum plane map and model simulation, we find that Mona Lisa and Nike lie in the first floor of the Louvre. At the same time, there are many exhibition halls on the first floor. And the M/G/C/C model simulates bottleneck locating on RICHELIEU. Therefore, we give the following suggestions [8].

- Reducing the number of visitors by limiting the number of people entering the exhibition halls on the first floor.
- It is recommended to increase the security investment in the first floor of the exhibition hall to ensure that the staff can evacuate the visitors quickly when an emergency occurs.
- It is proposed that the museum leaders install a continuous electric indicator on the floor of the exhibition halls to ensure that visitors can quickly find an evacuation path after an emergency occurs.
- Use the software of Affluences to determine the number of queues at the entrance to the exhibition hall. When an emergency occurs, staff members firstly guide visitors away from the entrance with a smaller queue.
- Museum leaders can post announcements to popularize the correct evacuation method for visitors.

7. The Discussion of Other Crowded Places

The Adaptive Genetic Algorithm Model is suitable for other scenarios, for example: Cinema, auditorium, gymnasium, etc. They have some common characteristics, there are a lot of visitor flow rates in these places, and people feel it is strange, so it is very significant to develop an accurate evacuation path. This is a flow chart given to different buildings in **Figure 7**: [8].

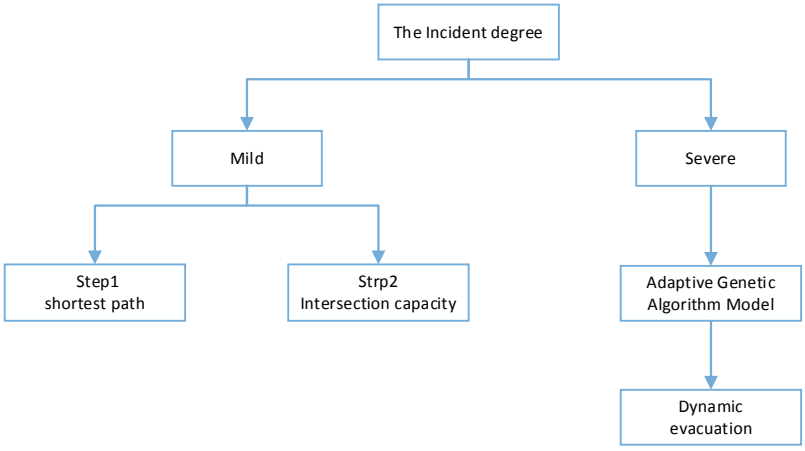

Figure 7. Simulate evacuation according to conditions.

8. Strengths and Weaknesses

- We built the ideal model and the actual model. The comparison shows the accuracy of the actual model.
- We improve the genetic algorithm, so we can get more accurate results.
- We did a sensitivity examine. It can ensure applicability and accuracy for our model.

Weaknesses

- We lack some data about the stairs.
- Using an imprecisely estimated parameter from a small set about date.
- Some hidden channels are not considered.

Conflicts of Interest

The authors declare no conflicts of interest regarding the publication of this paper.

References

[1] Meisling, T. (1958) Discrete-Time Queuing Theory. *Operations Research*, **6**, 96-105. https://doi.org/10.1287/opre.6.1.96

[2] Yang, G., Min, Z., Hang, Z. and Yue, L. (2016) Research on Dynamic Allocations of Airport Security Check Resources. *Aeronautical Computing Technology*, **46**.

[3] de Lange, R., Samoilovich, I. and van der Rhee, B. (2013) Virtual Queuing at Airport Security Lanes. *European Journal of Operational Research*, **225**, 153-165. https://doi.org/10.1016/j.ejor.2012.09.025

[4] Bevilacqua, M. and Ciarapica, F.E. (2010) Analysis of Check-In Procedure Using Simulation: A Case Study. 2010 *IEEE International Conference on Industrial Engineering and Engineering Management*, Macao, 7-10 December 2010, 1621-1625. https://doi.org/10.1109/IEEM.2010.5674286

[5] He, X. (2010) Stochastic Process and Queuing Theory. Hunan University Press, Changsha.

[6] Xu, X.Y., Liu, J. and Li, H.Y. (2012) Channel Capability Calculation Based on M/G/C/C State Dependent Queuing Model. *Logistics Technology*, **31**, 187-189.

[7] Meng, Y.C., Yang, S.N. and Shi, P.J. (2014) Multi-Objective Optimization of Emergency Evacuation Using Improved Genetic Algorithm. *Geomatics and Information Science of Wuhan University*, **39**, 201-205.

[8] Wang, J.H. and Lu, S.X. (2006) Calculation of Safety Evacuation Reliability of Buildings in Fire Environment Based on Genetic Algorithm. *China Engineering Science*, **8**, 58-61.

Appendices

Appendix I

```
for k=1:n
for i=1 :n
      for j=1:n
            t=B(i, k) + B(k, j);
            if t<B(i,j)    B(i,j)=t; end
      end
 end
end
```

Appendix II

1.Main.m

```
clear
clc
close all
D=[0 18 inf inf inf 300 inf inf inf inf inf inf inf inf 0;
18 0 18 inf inf inf 300 inf inf inf inf inf inf inf inf;
inf 18     0 18 inf inf inf 300 inf inf inf inf inf inf inf;
inf inf    18 0 18 inf inf inf 300 inf inf inf inf inf inf;
inf inf    inf 18      0 inf inf inf 300 inf inf inf inf inf;
inf inf    inf inf     inf 0 18 inf 300 inf inf inf inf inf 0;
inf inf    inf inf     inf 18      0 18 inf inf inf inf inf inf inf;
inf inf    inf inf     inf inf     18 0 18 inf inf inf inf inf inf;
inf inf    inf inf     inf inf     inf 18      0 18 inf inf inf inf;
inf inf    inf inf     inf inf     inf inf     18 0 inf inf inf inf;
inf inf    inf inf     inf 300     inf inf     inf inf     0 18 inf inf 0;
inf inf    inf inf     inf inf     300 inf     inf inf     18 0 18 inf inf;
inf inf    inf inf     inf inf     inf 300     inf inf     inf 18 0 18 inf;
inf inf    inf inf     inf inf     inf inf     300 inf     inf inf     18 0 inf;
0 inf inf inf inf 0 inf inf inf inf 0 inf inf inf    0
];
N=size(D,1);
NIND=100;
MAXGEN=200;
Pc=0.9;
Pm=0.05;
GGAP=0.9;
Chrom=InitPop(NIND,N);
disp('random:')
OutputPath(Chrom(1,:));
 Rlength=PathLength(D,Chrom(1,:));
disp(['distence：',num2str(Rlength)]);
```

```
disp('~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~')
gen=0;
figure;
hold on;box on
xlim([0,MAXGEN])
title('Optimization process')
xlabel('Algebra')
ylabel('The optimal value')
ObjV=PathLength(D,Chrom);
preObjV=min(ObjV);
while gen<MAXGEN

    ObjV=PathLength(D,Chrom);
    line([gen-1,gen],[preObjV,min(ObjV)]);pause(0.0001)
    preObjV=min(ObjV);
    FitnV=Fitness(ObjV);
    SelCh=Select(Chrom,FitnV,GGAP);
    SelCh=Recombin(SelCh,Pc);
    SelCh=Mutate(SelCh,Pm);
    SelCh=Reverse(SelCh,D);
       Chrom=Reins(Chrom,SelCh,ObjV);
    gen=gen+1 ;
end
ObjV=PathLength(D,Chrom);
[minObjV,minInd]=min(ObjV);
disp('zuiyoujie:')
p=OutputPath(Chrom(minInd(1),:));
```

2.Distance.m

```
function D=Distanse(a)
row=size(a,1);
D=zeros(row,row);
for i=1:row
    for j=i+1:row
        D(i,j)=((a(i,1)-a(j,1))^2+(a(i,2)-a(j,2))^2)^0.5;
        D(j,i)=D(i,j);
    end
end
```

3. dsxy2figxy.m

```
function varargout = dsxy2figxy(varargin)
if length(varargin{1}) == 1 && ishandle(varargin{1}) ...
                            && strcmp(get(varargin{1},'type'),'axes')
    hAx = varargin{1};
```

```
        varargin = varargin(2:end);
else
        hAx = gca;
end;
if length(varargin) == 1
        pos = varargin{1};
else
        [x,y] = deal(varargin{:});
end
axun = get(hAx,'Units');
set(hAx,'Units','normalized');
axpos = get(hAx,'Position');
axlim = axis(hAx);
axwidth = diff(axlim(1:2));
axheight = diff(axlim(3:4));
if exist('x','var')
        varargout{1} = (x - axlim(1)) * axpos(3) / axwidth + axpos(1);
        varargout{2} = (y - axlim(3)) * axpos(4) / axheight + axpos(2);
else
        pos(1) = (pos(1) - axlim(1)) / axwidth * axpos(3) + axpos(1);
        pos(2) = (pos(2) - axlim(3)) / axheight * axpos(4) + axpos(2);
        pos(3) = pos(3) * axpos(3) / axwidth;
        pos(4) = pos(4) * axpos(4 )/ axheight;
        varargout{1} = pos;
end
```