For office use only	Team Control Number	For office use only
T1 ________________	1902066	F1 ________________
T2 ________________		F2 ________________
T3 ________________	Problem Chosen	F3 ________________
T4 ________________	D	F4 ________________

2019
MCM/ICM
Summary Sheet

Multi-level evacuation model

Summary

Our goal is to get a large multi-storey building evacuation plan for multiple emergency situations. This paper mainly uses multi-source multi-sink multi-level evacuation model and maximum flow algorithm.

Firstly, we found the optimal evacuation model and route. i) After examining the map of the Louvre's five-story building, we draw an undirected graph with the exhibition hall as the node and the door and the stair as the arc. According to the actual situation of the Louvre and the principle of the maximum flow algorithm, we calculate the maximum loadable flow per arc and the supply and demand of each node during the evacuation process; ii)We combine the adjacent exhibition halls with the same evacuation route into one group. We select the most reasonable 24 schemes and use the multi-layer optimization evacuation model to obtain the optimal evacuation route for each scheme. iii) Based on the above schemes and routes, we use the C++ program written by the maximum flow algorithm to calculate the maximum feasible flow of each scheme. After comparison, the scheme with the largest flow rate is selected. This scheme is the optimal combination of exhibition halls that we need, and it is also the initial optimal evacuation scheme, which has great guiding significance for the evacuation of sub-regional people.

Secondly, we identify potential bottlenecks and discussed various threats. i) In the preliminary optimal evacuation route, we analyze the width and number of doors and corridors in various areas, as well as the number of people in the exhibition halls. Not only we identify potential bottlenecks that may limit the movement of people to the exit, but also find which areas may be the first to experience congestion. ii) We also consider four types of threats: the doors are blocked, the stairs are blocked, the exit is blocked, and the entire floor is blocked. Each type of threat has the potential to change or delete some routing segments. By analyzing these cases, we optimize the evacuation model to make it more robust and stable. iii) In addition, we consider when and how to get staff into the museum as quickly and safely as possible. At the same time we calculate the impact of these activities.

Finally, we present how to implement this model scheme in the Louvre and propose evacuation methods for emergency management and recommendations for crowd evacuation.

Keywords: maximum flow, multi-level evacuation model, multi-source multi-sink, optimal path

Contents

1 Introduction

After the required evacuation notice, the fastest possible safe evacuation must be ensured, and emergency personnel must enter the building for rescue, which can reduce casualties. The five floors of the Louvre, two of which are underground, and the identification of potential bottlenecks that may restrict movement to the exit, make evacuation difficult.

1.1 Outline

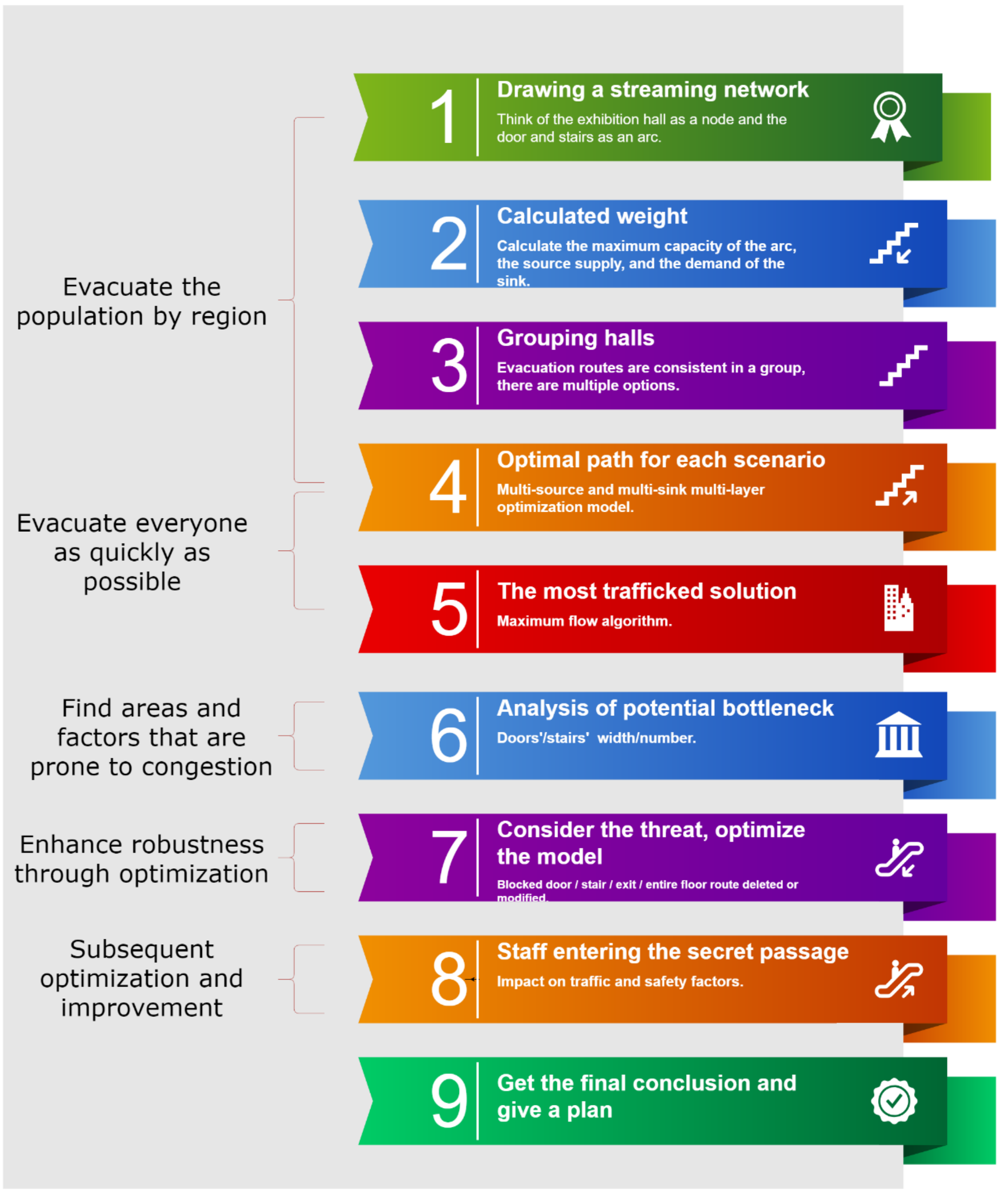
Our goal is to develop an adaptable population evacuation model to address a range of different disasters and implement it in the Louvre. To realize this objective we will proceed as follow:

- **Develop an emergency evacuation model** that allows the museum leaders to explore a range of options to evacuate visitors from the museum, while also allowing emergency personnel to enter the building as quickly as possible.
- Identify **potential bottlenecks** that may limit movement towards the exits.
- **Develope an adaptable model** that can be designed to address a broad set of considerations and various types of potential threats.
- Validate our model(s) and discuss **how the Louvre would implement it**, and propose policy and procedural recommendations for emergency management of the Louvre.

1.2 Main Assumptions

- Assume that the Louvre's three exits except the main exit can carry the same maximum flow of people;
- Assume that the evacuated population is obeying the staff command;
- Assume that the width of the door or stair is proportional to the maximum flow that can be carried;
- Suppose the crowd has the same speed of movement on different door arcs or stair arcs;

1.3 Our works


Evacuate the population by region
Evacuate everyone as quickly as possible
Find areas and factors that are prone to congestion
Enhance robustness through optimization
Subsequent optimization and improvement
1 Drawing a streaming network
Think of the exhibition hall as a node and the door and stairs as an arc.
2 Calculated weight
Calculate the maximum capacity of the arc, the source supply, and the demand of the sink.
3 Grouping halls
Evacuation routes are consistent in a group, there are multiple options.
4 Optimal path for each scenario
Multi-source and multi-sink multi-layer optimization model.
5 The most trafficked solution
Maximum flow algorithm.
6 Analysis of potential bottleneck
Doors'/stairs' width/number.
7 Consider the threat, optimize the model
Blocked door / stair / exit / entire floor route deleted or modified.
8 Staff entering the secret passage
Impact on traffic and safety factors.
9 Get the final conclusion and give a plan

(Graph 1.1 Our works)

1.3 Symbol Description

V: the set of node (an exact exhibition hall on a certain floor)
A: the directed arc set ($M \cup T = A$)

M: the arc of the doors (M_{jk} represents the arc between V_{ij} and V_{jk})

T: the arc of stairs (T_{ibha} represents the arc between V_{ib} and V_{ha}):

V: the weight function on the arc ($M \cup T = U$)

m: $m \to R$ is the weight function on the arc(m_{jk} represents the maximum of arc capacity between V_{ij} and V_{ik})

t: $T \to R$ is the weight function on the arc(t_{ih} represents the maximum of arc capacity between V_{ib} and V_{ha}

D: $V \to R$ is the weight function of the point (d_i is the number of exhibition hall)

N: N= (V, A, U, D) is the flow network, N= (V, A, U) is the uncertain flow network of D

E: the set of points on N (E_{ik} represents layer I, group k)

S: the set of source points on N
I: the set of sink on N

P: the subset of path A(P_{ijhk} represents all paths contain arcs from V_{ij} to V_{hk})

f: $A \to R$ is the weight function on the arc(f_{ijhk} represents rate of flow from V_{ij} to V_{hk})

2 Maximum flow algorithm

Flow network N=(V,A,U,D):

U: the weight function on the arc ($M \cup T = U$) The maximum human flow that can be carried by the doors or stairs and other passageways in this problem. In particular, the lower bound of the capacity of arc (i, j) is set as 0 here;

D: $V \to R$ is the weight function of the point (d_i is the number of exhibition hall), which represents the number of people in each hall at the moment of an emergency in this case.

$V_{im} \to V_{jn}$ is the path from hall m on floor I to hall n on floor j at will.

V_i is the set of all points on floor I, which contains n_i group.

t'_{ajbk} is the maximum capacity from E_{aj} (layer a, group j) to E_{bk} (layer b, group k)

f'_{ajbk} is the maximum rate of flow from E_{aj} (layer a, group j) to E_{bk} (layer b, group k)

$$\max v$$

s.t. $$\sum_{A_{ajbk}\in A} x_{ajbk} - \sum_{A_{bkaj}\in A} x_{bkaj} = \begin{cases} v, V_{aj} = s'' \\ -v, V_{aj} = t'' \\ 0, V_{aj} \neq s'', t'' \end{cases}$$

$$0 \le x_{ajbk} \le m_{ajbk}, t_{ajbk} \qquad \forall A_{ajbk} \in A$$

$$E_{i1} \cup E_{i2} \cup \cdots \cup E_{in_i} = V_i$$

$$V_{im} \to V_{in} \in P \quad \forall V_{im}, V_{in} \in E_{ij} \, (\mathrm{n} \neq \mathrm{m})$$

$$m'_{ijik} = \sum m_{xy} - \sum m_{yx} \quad \forall V_{ix} \in E_{ij} \quad \forall V_{iy} \in E_{ik}$$

$$0 \le f'_{ijik} \le m'_{ijik} \quad 0 \le f'_{ajbk} \le t'_{ajbk}$$

$$t'_{ajbk} = \sum t_{ab} - \sum t_{ba} \qquad \forall V_{am} \in E_{aj} \qquad \forall V_{bn} \in E_{bk}$$

Ford - Fulkerson algorithm

In each lteration of the ford-fulkerrson method, an augmented path p is found, and p is then used to modify (increase) the flow f. We keep looking for an augmented path in the graph and increasing the value of f by that until the graph doesn't contain an augmented path. Pseudo code implementation is as follows:

```
FORD-FULKERSON (G,s,t)
1   for each edge (u,v) ∈ G.E
2       (u,v).f = 0
3   while there exists a path p from s to t in the residual network $G_f$
4       $c_f(p) = \min\{c_f(u,v): (u,v)\ is\ in\ p\}$
5       for each edge (u,v) in p
6           if (u,v) ∈ E
7               $(u,v).f = (u,v).f + c_f(p)$
8           else $(v,u).f = (v,u).f - c_f(p)$
```

The running time of the ford-fulkerson algorithm depends on the method of finding an augmented path, which is also the theoretical basis for edmonds-karps algorithm to improve the basic ford-fulkerson algorithm.

We will use the above algorithm C++ program design and its implementation, at the same time, we also can output any one point to the export path, note that because it is concluded that the results of the maximum flow algorithm is one of the largest evacuation traffic overall evacuation cases, so the results must be the optimal overall evacuation plan, to get the path is in the current situation for the evacuation of the optimal path (the path for an individual is not necessarily the optimal path).

3 Multi-layer evacuation model

3.1 Model establishment

For the evacuation of multi-layer buildings like the Louvre, it is considered how to rationally group each person to be evacuated, and select the appropriate path for evacuation, so that the evacuation network evacuation end time is minimized. Suppose $x_k^w(\tau)$ is the total number of people on the w floor who are leaving the path P_k to reach the exit at time τ. According to the dynamic network flow, we obtain: $x_k^w(\tau) = \sum_{t=0}^{\tau} f_k^w(t)$ (1)

It can be seen from Equation 1 that by the end of the evacuation of the w layer, the number of people evacuated by the trapped person along the path P_k is $\sum_{t=0}^{T^w} f_k^w(t)$.

Based on the above defined symbols and variables, we have established the following model:

$$(P) minT = \max_{w}\{T^w\} \tag{2}$$

$$\text{s.t. } \sum_{w=1}^{W} \sum_{P_k \in K^w} \delta_{ijk}^w f_k^w(t) \le c_{ij}, \ \forall (\mathrm{i,j}) \in \mathrm{E}, \mathrm{t} = 1,2 \dots \mathrm{T} \tag{3}$$

$$\sum_{P_k \in K^w} \sum_{t=0}^{T^w} f_k^w(t) = q^w,\ w = 1,2,\dots,W \tag{4}$$

$$f_k^w(t) \ge 0,\ \forall P_k \in K^w,\ w = 1,2,\dots,W \tag{5}$$

The objective function (2) indicates that the end time of the entire evacuation network evacuation is minimized; the constraint (3) refers to the constraint that the road segment capacity is satisfied at any time, and Equation (4) means that all evacuated personnel at each affected point are safely evacuated, and the constraint (5)indicates the nonnegative conditions path flow.

3.2 Model solution

step 1: Find the shortest path P^w of each layer $s_w \in S$ to the exit, and record the time t of these paths, and sort by their order from largest to smallest $T_{P^{w1}} \leq T_{P^{w2}} \leq \cdots T_{P^{wr}} \leq \cdots \leq T_{P^{wW}}$;

step 2: Enter the initial actual evacuation path $K := \emptyset$, the feasible path $K' := \emptyset$, the floor collection $H := \emptyset$, the dynamic traffic $F := \emptyset$, the layer evacuation time set $T := \emptyset$, let r=1;

step 3: Use the algorithm Dijkstra to find the shortest distance P_k at which the disaster point s reaches the super end point D_0. we had $T_{P_k} = \sum_{e_{ij} \in P_k} t_{ij}$, and let $P = P \cup \{P_k\}$,$TP = TP \cup \{P_k\}$. Calculate the maximum amount of traffic for path P_k , $PC_k = \min\{c_{ij} | e_{ij} \in P_k\}$, $f_k = PC_k$, $F = F \cup \{f_k\}$. Update the maximum capacity of each arc $c_{ij} = \{\begin{matrix} c_{ij} - f_k, e_{ij} \in P_k \\ c_{ij}, e_{ij} \notin P_k \end{matrix}$, if $c_{ij} = 0$,then delete e_{ij} and update the net. If the updated network is not connected, we can get a feasible evacuation k_0^{wr} .

The effect of other paths is not considered when looking for a feasible path, so f_k is a constant. According to the dynamic network flow, the total number of people reaching the super end point at time t is equal to the following equation.

$$x_k(t) = \begin{cases} (t - (T_{P_k} - 1)) f_k, t \geq T_{P_k} \\ 0 \end{cases}$$

Assuming that x is the total number of people to be evacuated, then the existence of path m_1 is satisfied the formula:

$$x \geq \sum_{k=1}^{m_1} (T_{P_{m1}} - (T_{P_k} - 1)) f_k \quad \text{and} \quad x < \sum_{k=1}^{m_1+1} (T_{P_{m1+1}} - (T_{P_k} - 1)) f_k$$

We can determine the actual participation in the evacuation path set according to the above formula. Assuming $P_1, P_2, \dots, P_\gamma$ is the path selected to evacuate personnel, the evacuation time required for all evacuation groups is equal in the optimal evacuation scheme, $T_1 = T_2 = \cdots = T_y = T$.

In the optimal evacuation scheme, all the paths used for evacuation are equal to the time when the last person reaches the super end point. We can conclude:

$$x = \sum_{k=1}^{m_1} x_k(T) = \sum_{k=1}^{m_1} (T - (T_{P_k} - 1)) f_k$$

Which means that the

$$T = \frac{x + \sum_{k=1}^{m_1} T_{p_k} f_k - \sum_{k=1}^{m_1} f_k}{\sum_{k=1}^{m_1} f_k}$$

As long as the actual evacuation path set is determined, the evacuation end time can be determined, and then the number of actually assigned each selected path is calculated according to the equation$x_k(T) = (T - (T_{P_k} - 1)) f_k$, thereby determining the optimal evacuation plan.

Step 4:

if(r==W) { $H := H \cup \{w_r\}$ $K' \coloneqq K' \cup K_0^{wr}$ goto step 7 }

step 5:

if ($T^{w_r} < T_{p^{w_{r+1}}}$) { goto step 7 } else { $H := H \cup \{w_r\}$ $K' := K' \cup K_0^{w_r}$ $K^{w_r} := \emptyset$ $r := r + 1$ go to step 3 }

Step 6:

$T^{w_r} := T^{w_r}$ $K^{w_r} := \cup_k$ $\{P_k^{w_r} | P_k^{w_r} \leq T^{w_r}\, and\, P_k^{w_r} \in K_0^{w_r}\}$ $T_0 := T_0 - \{T^{w_r}\}$

$F_0 := F_0 - \{f_k^{wr}\}$ If ($t \leq T^{w_r} \& t \geq T_{p_k^{w_r}}$) $f_k^{w_r}(t) := 0$ else { goto step 11 }

Step 7:

sort-small(H, T^{w_r}) if ($w_r \in H$){ $T^{w_r} \in H'$ } $T^{w_r} := max\{H'\}$

Step 8:

$P_k^{w_r} := \min\{K'\}$ for ($P_l^{w_r} in (K' \cup K)$) { if($\{e_{ij} | e_{ij} \in P_k^{w_r} \cap P_l^{w_y}\} = \emptyset$) { if ($t \geq T_{p_k^{w_r}}$) {$f_k^{w_r}(t) = \min\{c_{ij}(t) | e_{ij} \in P_k^{w_r}\}$} else{$f_k^{w_r}(t) = 0$} goto step 10 } else { goto step 9 }}

Step 9:

$V = P_k^{w_r}[0]$ If($P_l^{w_r} \in K' \cup K \& P_L^{w_\gamma} \cup P_k^{w_r} == V$) {if($t < \min\{T_{P_k^{w_r}}, T_{p_L^{w_\gamma}}\}$)

{ $f_k^{w_r}(t) = 0$, $f_l^{w_r}(t) = 0$ elseif({ $f_k^{w_r}(t) = 0$ $f_k^{w_r}(t) = \min\{c_{ij}(t) | e_{ij} \in P_k^{w_r}\}, t \geq T_{p_k^{w_r}}\}$ else if ($T_{P_l^{w_r}} \leq t \leq T_{p_k^{w_r}}$) $f_k^{w_r}(t) = 0$ {$f_l^{w_\gamma}(t) = \min\{c_{ij}(t) | e_{ij} \in P_k^{w_r}\}$} else if ($\max\{T_{P_k^{w_r}}, T_{P_l^{w_\gamma}}\} \leq t \leq T^{w_r}$){{f(t) = $\min\{c_{ij}(t) \in P_k^{w_r}\}$

$$c_{ij}(t) = \begin{cases} c_{ij}(\mathrm{t}) e_{ij} \notin P_k^{w_r} \\ c_{ij}(t) - f_k^{w_r}(t) e_{ij} \in P_k^{w_r} \end{cases} \quad f_l^{w_\gamma}(t) = \min\{c_{ij}(\mathrm{t}) e_{ij} \in P_l^{w_\gamma}\}\}$$

else if(t>T^{w_r}) {$f_r^{w_\gamma}(t) = \min\{f_k^{w_r}(T^{w_r}) + f_l^{w_\gamma}(T^{w_r}),\ f_k^{w_r} = 0\}$

Step 10:

$K' := K'\{P_k^{w_r}\}$ $K^{w_r} := K^{w_r} \cup \{P_k^{w_r}\}$ $F_0 = F_0 - \{f_k^{w_r}\}$ If($K' \cap \left(\bigcup_k P_k^{w_r}\right) == \emptyset$ $\{T^{w_r} := T^{w_r}$ goto step 11} else { goto step 8 }

Step 11:

$K := K \cup K^{wr}$ $F := F \cup \{f_k^{wr}(t)\}$ $\Gamma := \Gamma\{T^{wr}\}$ $H := H - \{w_r\}$

Step 12:

If (H:=ϕ) { goto step 13 } else { update (T^{wr}) goto step 7}

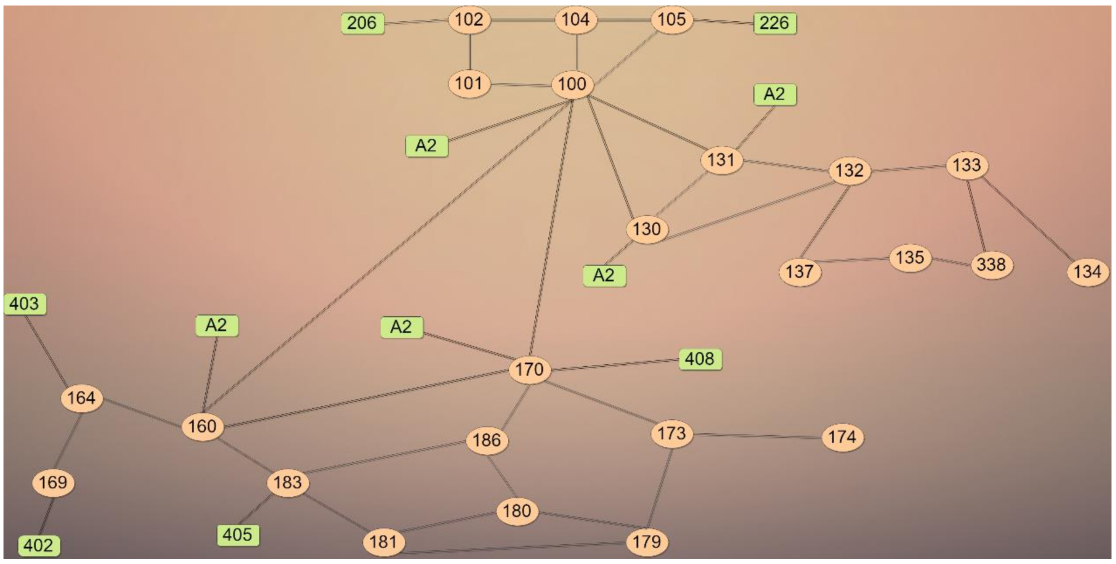
Output:

Output path set, path dynamic traffic set, evacuation time set of each layer.

4 Solve problems with models

Pre-preparation: calculation and drawing

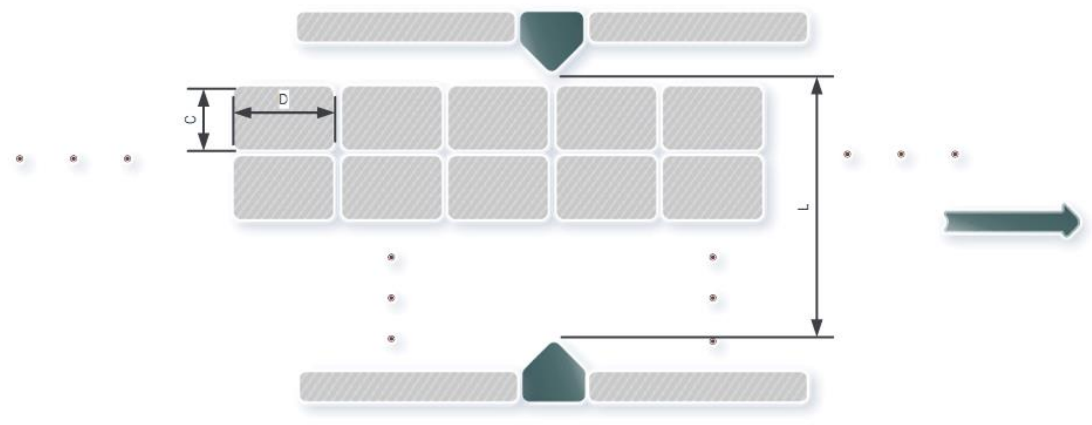
In this question, according to the relevant algorithm of the maximum flow problem, we regard each exhibition hall as a "point" in the mathematical sense, and the door or staircase is regarded as an "arc" in the mathematical sense. Since it is considered here that population evacuation occurs in the event of an emergency, we do not consider a passage such as an elevator that may be unsuitable for escape in an emergency to be considered a valid "arc". According to the museum map of the Louvre official website(https://www.louvre.fr/sites/default/files/medias/medias_fichiers/fichiers/pdf/louvre-plan-information-english.pdf), we have drawn all the main exhibition halls and doors, stairs and other channels in the five floors of the Louvre as an undirected picture, floor -1 as shown below:



(Graph 4.1 map of floor -1)

In this picture, each circle corresponds to an exhibition hall in the Louvre, each rectangle corresponds to a staircase in the Louvre (connecting the upper and lower escape passages), the number marked in the circle or rectangle corresponds to the unique label of the exhibition hall or passage on the map of the Louvre Museum, and each arc marks the maximum flow of people that the path can bear (single). Bit: person/second.

For the evacuated people, we took the following physical quantities according to the search paper and the actual research: the average moving speed of the passing gate of the normal person in the evacuation is 7.2 km/h (2 m/s), and the average moving speed of the stairway is 1.2 m / s, the average speed of the stairs is 1.5 m / s, the average space occupied by a person is 0.75 m, the space width of a person is 0.5 m, and the average width of the door in the Louvre is 2 m. The average corridor width is 4 meters. When passing through a door or stairs, the following top view is shown:



(Graph 4.2 door)

Where c is the width of a person's space, d is the length of a person's space, l is the width of the door or stairs, v is the average moving speed of the crowd, the dotted line is the door or stairs, the solid line is the wall, a small rectangle Represents the space occupied by a person. Then, c = 1 m, d = 0.75 m, for the gate, v = 2 m / s, l = 2 m, so the maximum flow of the crowd through the door is:

$$INT\left(\frac{1}{c}*\frac{1}{\frac{d}{v}}\right)=INT\left(\frac{2}{0.5}*\frac{1}{\frac{0.75}{2}}\right)=INT(10.666666)=10$$

In the same way, the maximum flow of the stairs to the crowd is 12 people/second, and the maximum flow of the stairs down the stairs is 16 people/second.

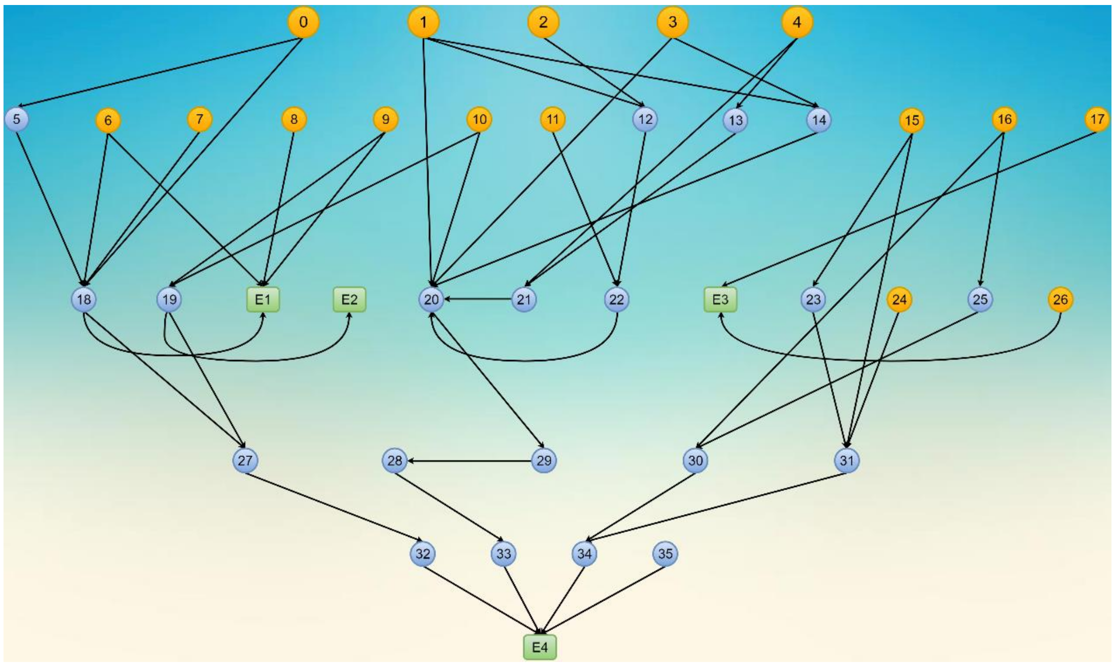
After doing the preparatory work as shown above, we established an evacuation model based on the multi-layer optimization evacuation model and the maximum flow algorithm.

Step 1: Group and find the optimal path for each scenario

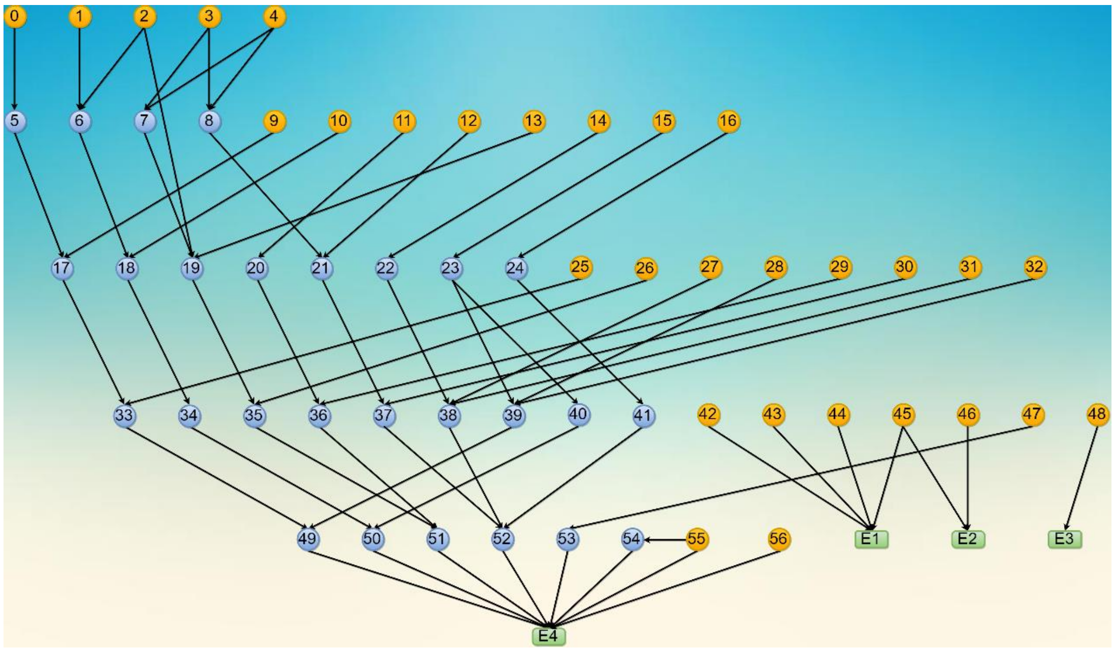
In order to make it easier to analyze the evacuation route of an area, and to control the crowd more conveniently and reasonably, for example, the staff can conduct guidance or broadcast to tell people how to leave the museum as soon as possible. We merged the adjacent exhibition halls with the same evacuation route as a point in the picture. Of course, there exist many different grouping schemes, but the number of reasonable grouping schemes can be much smaller. The basic conditions for multiple exhibition halls can be divided into one group:

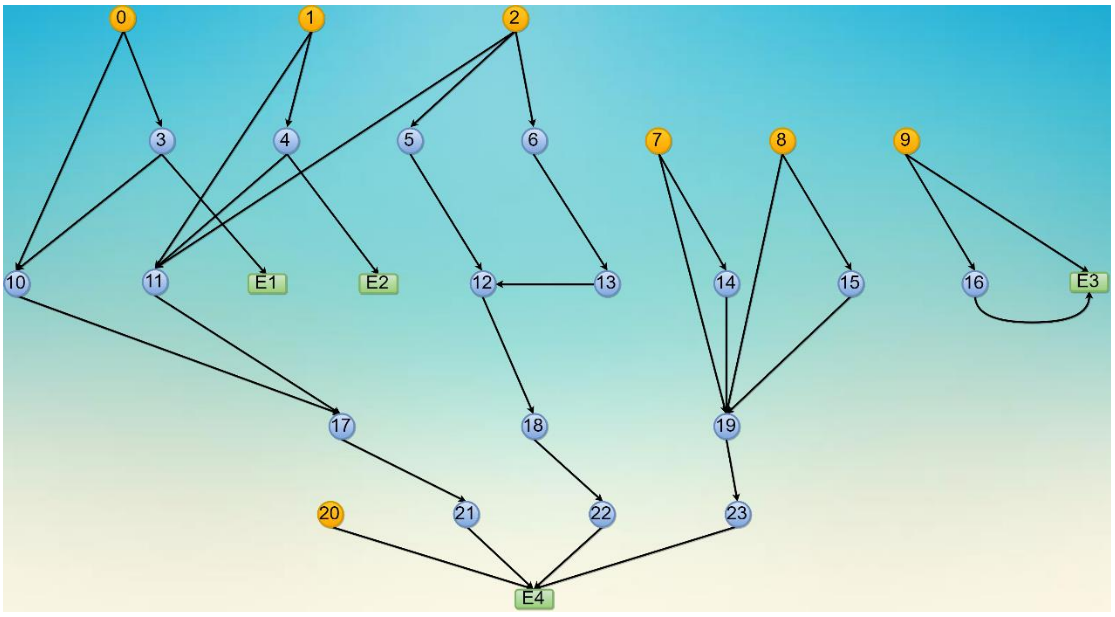
-The evacuation routes of these exhibition halls are consistent.
- Any two exhibition halls of these exhibition halls can be reached only by the arcs in the group.

We have drawn the most reasonable 24 programs, some of which are shown below:



(Graph 4.3 program1)


(Graph 4.4 program2)


(Graph 4.5 program3)

Then, we used the multi-level optimization evacuation model for each scheme, and wrote a program to solve the problem. The optimal path of each scheme was obtained. For one of the schemes, the operation results are as follows:

Node	Optimal route
0	0->18->E1
1	1->20->29->28->33->E4
2	2->12->22->20->29->28->E4
3	3->20->29->28->33->E4

4	4->21->20->29->28->33->E4
5	5->18->E1
6	6->E1
7	7->18->E1
8	8->E1
9	9->E1
10	10->19->E2
11	11->22->20->29->28->33->E4
12	12->22->20->29->28->33->E4
13	13->21->20->29->28->33->E4
14	14->20->29->28->33->E4
15	15->31->34->E4
16	16->30->34->E4
17	17->E3
18	18->E1
19	19->27->32->E4
20	20->29->28->33->E4
21	21->20->29->28->33->E4
22	22->20->29->28->33->E4
23	23->31->34->E4
24	24->31->34->E4
25	25->30->34->E4
26	26->E3
27	27->32->E4
28	28->33->E4
29	29->28->33->E4
30	30->34->E4
31	31->34->E4
32	32->E4
33	33->E4
34	34->E4
35	35->E4

(Chart 4.1 route)

Through the above results, under the condition that everyone can safely leave the building as quickly as possible, we can see the route of all the areas to the exit.

Step 2: Find the best solution

After obtaining the complete flow network and evacuation route, we used the C++

program written by the maximum flow model to calculate each solution and obtained their maximum flow numerical solution, as shown in the following figure:

program	maxflow value	Percentage of the maxflow value of the optimal solution
1	52	0.825396825
2	51	0.80952381
3	53	0.841269841
4	52	0.825396825
5	49	0.777777778
6	54	0.857142857
7	58	0.920634921
8	51	0.80952381
9	52	0.825396825
10	48	0.761904762
11	53	0.841269841
12	55	0.873015873
13	46	0.73015873
14	50	0.793650794
15	54	0.857142857
16	48	0.761904762
17	53	0.841269841
18	52	0.825396825
19	50	0.793650794
20	48	0.761904762
21	42	0.666666667
22	44	0.698412698
23	43	0.682539683
24	47	0.746031746
maxflow value of the optimal solution		63

(Chart 4.2 programs)

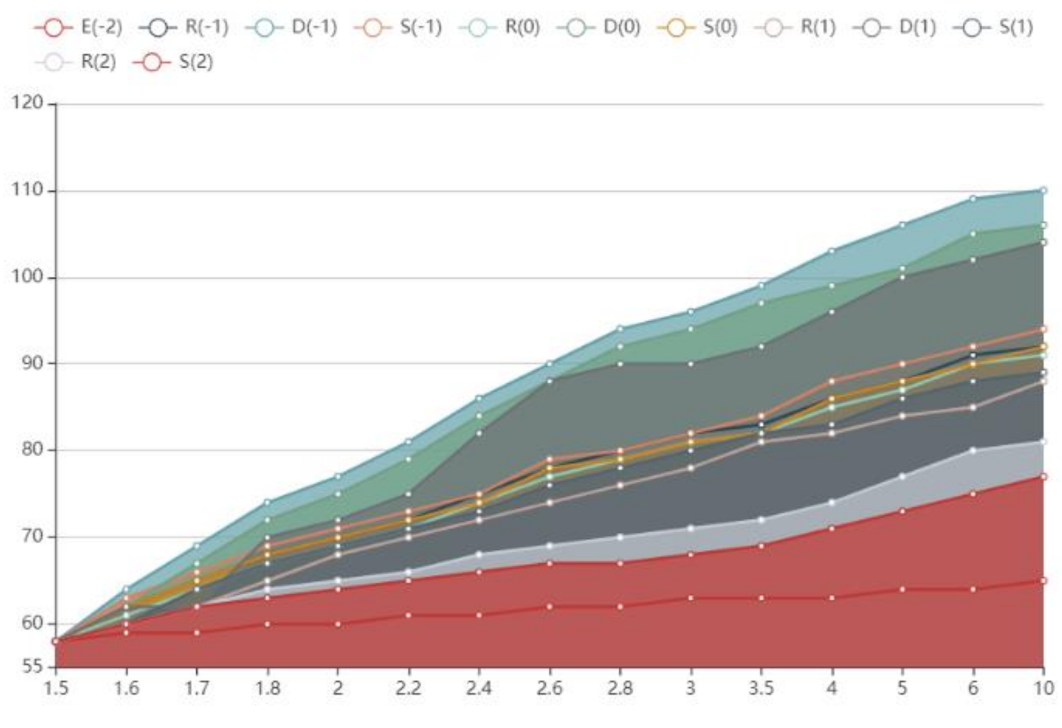
After that, we sorted each solution according to the numerical solution of the maximum flow, and got the one with the largest flow value. This plan was the grouping method and evacuation route for the exhibition hall that we were looking for, which is the scheme 7 in the above figure.

So far, we have made a reasonable grouping of exhibition halls. Sub-regional evacuation has been achieved, and we have derived the best evacuation route for each region, laying the foundation for a reasonable, rapid and safe evacuation of people. The above model is a very idealized model. Then we could consider the various realistic factors and constraints to locally optimize the model.

Step 3: Find potential bottlenecks

After analyzing the actual Louvre, we found that the factors that hindered the evacuation of the crowd were: the width of the door of the exhibition hall, the width of the stairs, the number of doors, the number of stairs and the number of people in the exhibition hall. The above factors can be easily reflected and changed in our flow network diagram: the width of the door and the stairs correspond to the maximum capacity of the arc and is linear, the number of doors and stairs corresponds to the number of arcs, and is linear The number of people in the exhibition hall corresponds to the capacity of the point. We used the control variable method to analyze each of the above five factors and draw some conclusions.

It is noticeable that we have not only changed the width or number of doors or stairs, but changed the sub-regions. The advantage is that we can know exactly when we get the threshold of the width of the door. The width of the door in which area has a greater impact on the final result, that is, the width of the door in this area is a potential bottleneck. For the width of the gate of a certain area, we took the values of 1.5, 1.6, ..., 10, etc., and then according to the flow calculation formula of the above preparation stage, we obtained the maximum flow of the arc, adjusted the value of the flow network, and applied the maximum flow algorithm. Similarly, for the width of the stairs, the maximum flow of the arc in the flow network is also adjusted; for the number of doors and stairs, the number of arcs in the flow network needs to be adjusted. At this time, the maximum flow algorithm is used to calculate the result; for the number of people in the exhibition hall, What needs to be adjusted is the supply of source points in the streaming network. Finally, the above data is plotted, as shown below for the bottleneck analysis of the width of the door:

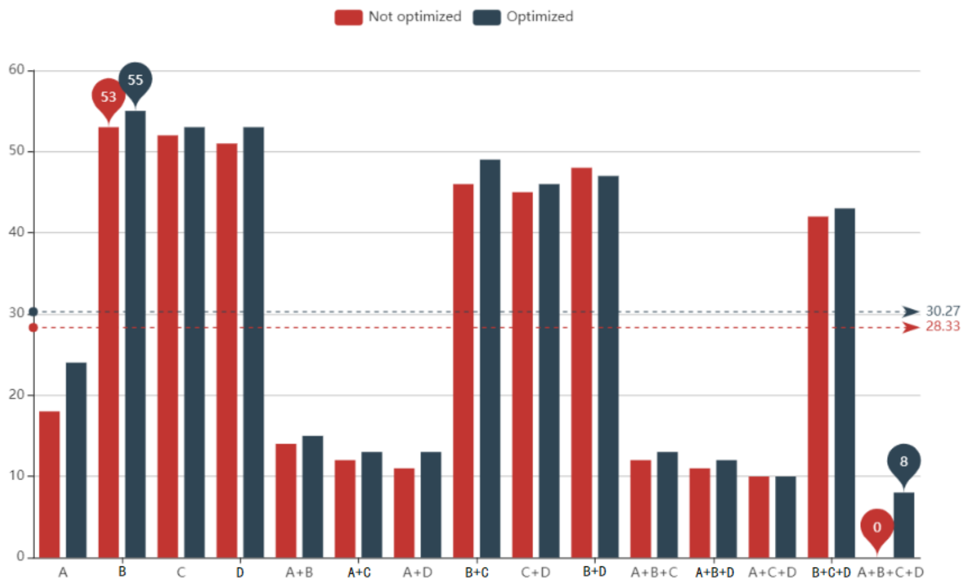


(Chart 4.3 Width of the door in different areas - flow)

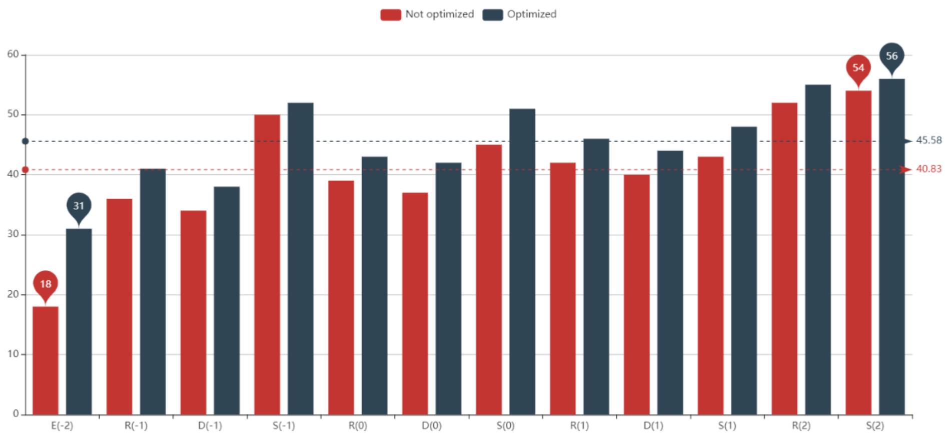
According to the above chart, D(0) is a very obvious potential bottleneck. In the event of an emergency requiring evacuation, these areas need special attention, because if congestion occurs, these areas can be the first to be congested, and the staff needs priority. Pay attention to these places and give priority to evacuating people from these places. Next we would optimize the existing model based on several typical possible emergencies.

Step 4: Optimize the model based on possible emergencies

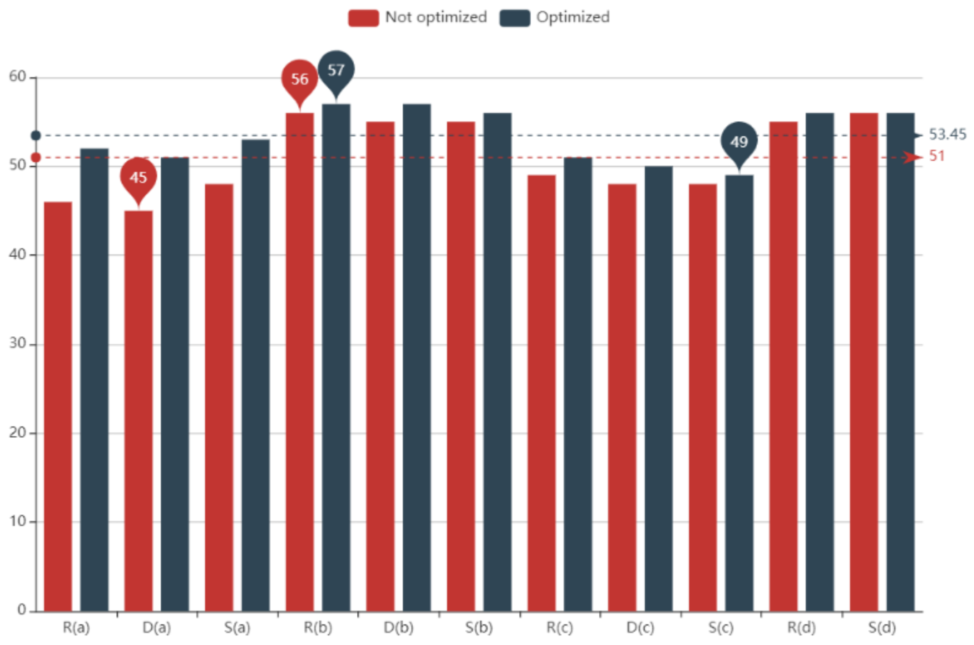
We optimize and improve the existing model route, which is essentially a small adjustment of the route of the convection network, and then calculate the maximum flow of the network at this time, and compare the two, the results are as follows:



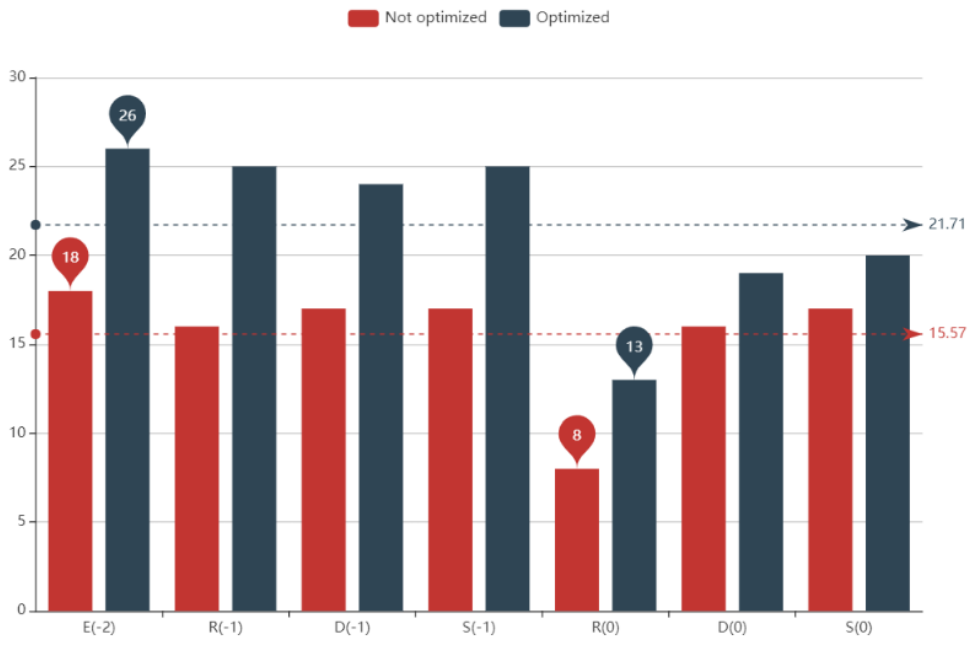
(Chart 4.4 Before and after optimization-1)



(Chart 4.5 Before and after optimization-2)



(Chart 4.6 Before and after optimization-3)

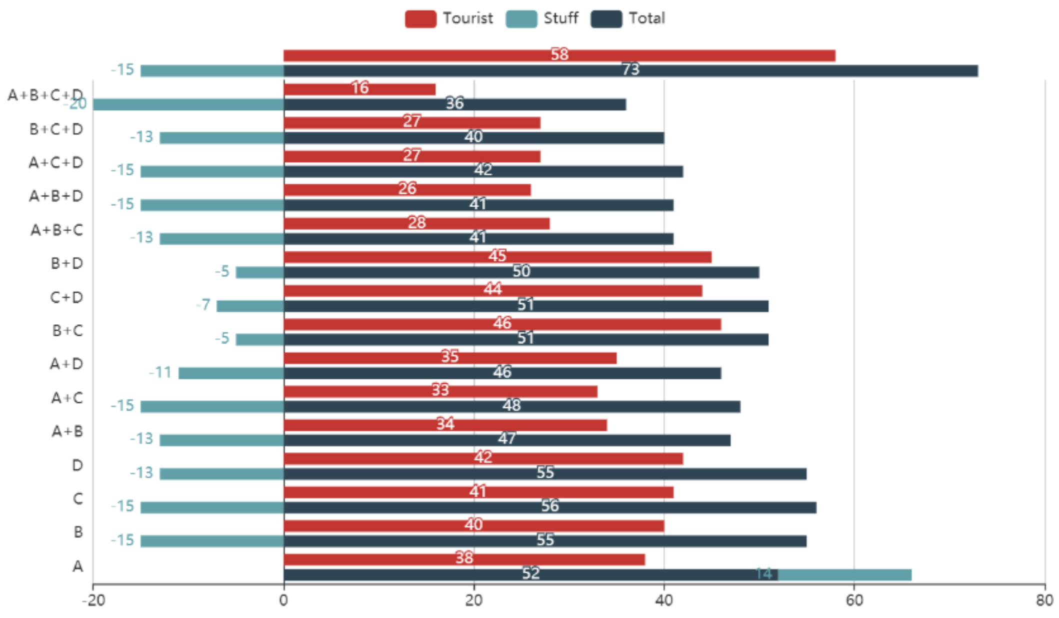


(Chart 4.7 Before and after optimization-4)

It can be seen that our model has been well adapted to a variety of emergencies and has good robustness. For each different emergency type, different adjustment methods are needed. Therefore, when the model is actually implemented, whether an emergency occurs, different optimization strategies need to be adopted according to different emergency types to obtain a larger. The maximum flow of the network means that all people can be evacuated at a faster rate.

Step 5: Consider staff and other exits

Based on the above model, consider how the staff will enter the building in a timely and safe manner after an emergency and with minimal impact on the evacuation of the population. The approximate maximum flow value corresponding to the entry of the staff from the existing four entrances and from other secret passages as shown in the chart below:



(Chart 4.7 Staff entering the building)

It can be seen from the figure that if the staff enters from the existing four entrances, they will collide with the people who are evacuated, which is not only not conducive to the staff entering the building more quickly and safely, but will make the evacuated people become crowded and Slow, and entering the building from other secret passages, at least will not affect the people evacuated. Therefore, it is best for the staff to enter the building from a secret passage.

Next, we consider the impact of opening a few secret passages after the staff enters the secret passage. As shown in the chart below, what effect does the network have on the maximum flow when several secret channels are opened? On different floors, the optimal number of open channels is different, but for each floor, there is a relatively small number of "specific quantities", when the number of open secret channels is greater than this "specific number" At the time, the contribution to the maximum flow of the network was not large or even contributed. Moreover, due to the poor security conditions of the secret passages, there are some security risks in the open secret passages. The more open secret passages, the greater the security risks. Therefore, when a worker enters a building, a small number of secret passages can be opened to enable the crowd to evacuate more quickly while ensuring safety.

5 Improving the Model

Improvement point 1: Accuracy of population distribution

When the actual map of the Louvre is transformed into a stream network in the mathematical sense, there is no more reasonable arrangement regarding the number of people and the distribution of the people in the Louvre because of the imperfect information. The number of people in each exhibition halls is not the same because of their different popularity. The existing model is based on the total number of people in the Louvre and then average into each exhibition hall, which has a certain sense of irrationality. When the model is improved, it is more reasonable to grasp the distribution of the people in the Louvre and draw a heat map of the population distribution. Then, using the knowledge of statistics, after obtaining enough people in each exhibition hall at each time point, the average number of people in each exhibition hall can be obtained by averaging each exhibition hall.
Assuming that the number of people in the N_ij^k exhibition hall on the i floor is j at the time of the k sampling. When the Louvre is open, N_ij^k is obtained at random times every day, and there is the following formula:

$$\begin{cases} \frac{1}{k}\sum_{1}^{n} N_{ij}^{k} = N_{ij} \\ n \geq 100 \\ 0 < N_{ij}^{k}\ \forall k \in [1, n] \end{cases}$$

According to this formula, the average number of people in all the halls of all floors can be obtained, and the number of people thus obtained is very close to the reality. Using this value can get a better and more accurate model. In addition, without this opportunity to count the exact number of people, it is also possible to use a statistical poisson distribution and the actual situation of the Louvre's popular exhibition hall to arrive at a statistically reasonable number.

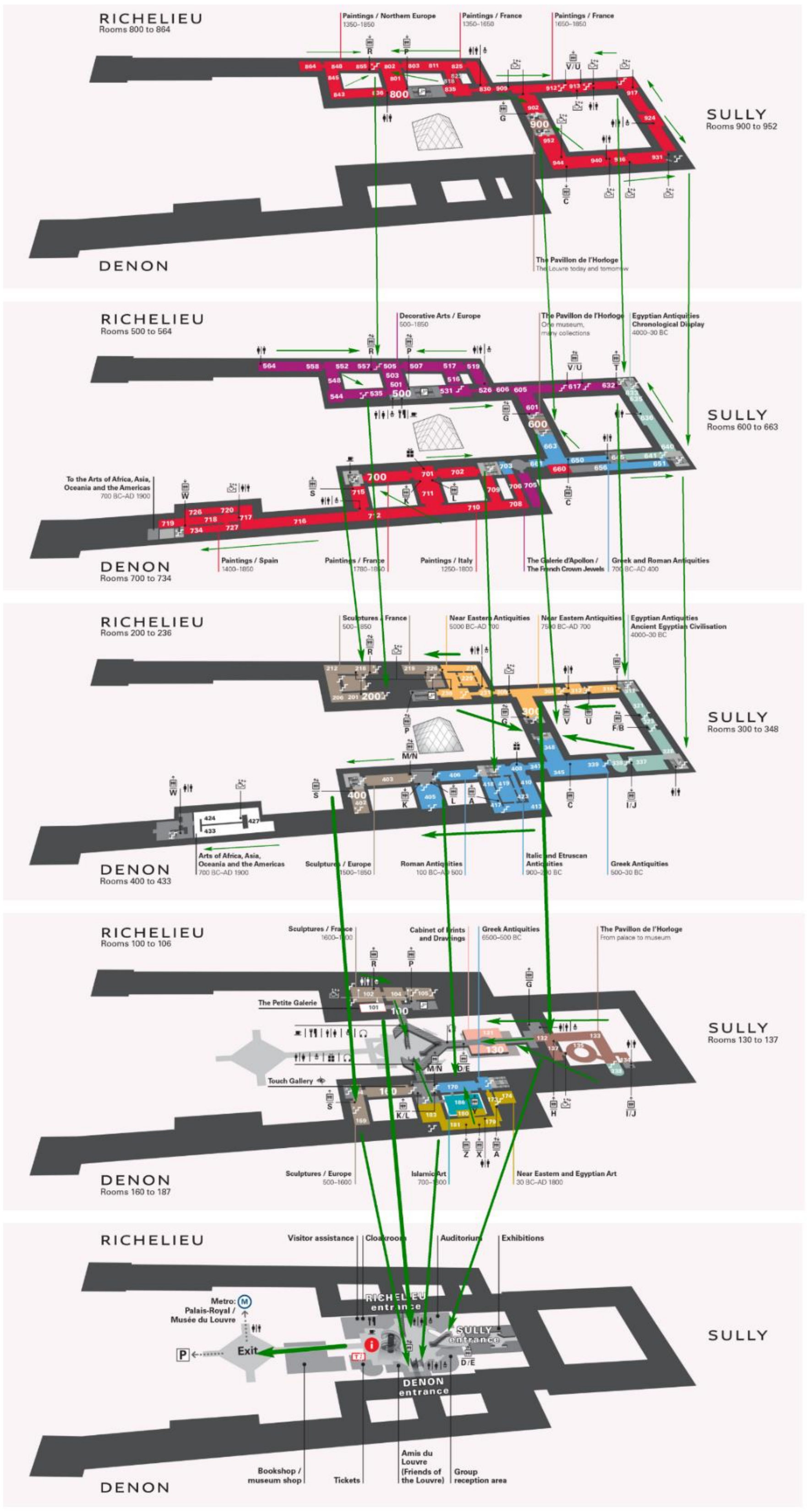
Improvement point 2: Optimization of the maximum capacity and flow network

In this model, we built a flow network and used this as a basis to solve and optimize a series of subsequent situations. In the flow network, the maximum capacity between nodes and nodes is a theoretical solution through a series of physical parameters of the Louvre's door or stairs and the crowd. Although we have tried our best to restore the evacuation site and try to solve it with near-realistic values, it is undeniable that the

reality is complex and variable, and the maximum capacity value is closely related to the final result, especially the arc near the exit. Maximum capacity, its impact on the results is very large. The calculation error of the maximum capacity of the arc will be amplified in the later model calculations, and even the result will have a large deviation. Therefore, this must be taken seriously in the improvement of the model.
The main factor causing the error in the maximum capacity of the arc is the complexity of the actual evacuation situation, which is difficult to accurately measure with a limited number of physical quantities. Therefore, there are two ways to improve the model given here: field test and large-scale calculation in statistical sense. The best way to improve is to count as much historical information as possible about the evacuation site (not necessarily in the Louvre, as long as the site is close), and then based on the statistical data to more reasonably calculate the maximum capacity of the arc.
In addition, for the optimization part of the flow network, the flow network in our model in the later stage is all network simplified by grouping. Although we have conducted many explorations and put forward 24 kinds of solutions, the combination is too much. We can't exhaust all combinations in a limited time to find the best combination. This can be improved by taking a big data analysis based on the route of the people visiting the Louvre and finding out which part of the population has a similar "next" route. This part of the exhibition hall can be divided into one group. At the same time, due to the suddenness of the emergency, most people may make their own judgments and flee in a certain direction before the crowd is notified by the museum. Therefore, it is obviously more reasonable to divide the combination according to the big data that the people have been used to.

6 Advice

Hello! Regarding the evacuation of people in the Louvre, we have given the best evacuation route under ideal conditions after the model is built. Please refer to the appendix for details. For the potential bottleneck, we found that the number of stairs is the main bottleneck, and the width of the stairs between the -1 and -2 floors is also a very important bottleneck, and the evacuated people tend to be the first to be congested here. When threats such as fires, floods, etc. occur, the best way to divert them is to prioritize the crowded exhibition halls or people near the passages to nearby passages and notify others that they are no longer evacuated to the blocked areas. For the staff, it is best to enter the building from other entrances, then open a certain number of exits and guide the crowd to actively evacuate.



(Graph 6.1 Solution)

References

[1] Keith Christensen, Yuya Sasaki. Agent-based emergency evacuation simulation with individuals with disabilities in the population [J]. Journal of Artificial Societies and Social Simulation (S1460-7425), 2008, 11(3): 9-21.

[2] Xuwei Chen, F Benjamin Zhan. Agent-based modeling and simulation of urban evacuation: Relative effectiveness of simultaneous and evacuation strategies [J]. Journal of the Operational Research Society (S0160-5682), 2008, 59(1): 25-33.

[3] Chuanjun Ren, Chenghui Yang, Shiyao Jin. Agent-based modeling and simulation on emergency evacuation [J]. Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (S1867-8211), 2009, 5(1): 1451-1461.

[4] Hamacher H W, Tjandra S A. Mathematical modeling of evacuation problems—a state of the art [C]// edited by Michael S, Som D S, Pedestrian and Evacuation Dynamics, Berlin, Germany: Springer, 2002: 227-266.

[5] Hamza-Lup G L, Hua K A, Peng R. Leveraging e-transportation in real-time traffic evacuation management [J]. Electronic Commerce Research and Applications (S1567-4223), 2007, 6(4): 413-424.

[6] Chen Pohan, Feng Feng. A fast flow control algorithm for real-time emergency evacuation in large indoor areas [J]. Fire Safety Journal (S0379 -7112), 2009, 44(5): 732-740.

Appendix

//Source.cpp

```
/*
Memory 328K
Time  438MS
*/

#include "head.h"
#define MAXV 300
#define INF 1<<29
#define min(a,b) (a>b?b:a)


map<int, int> hash1;
map<int, int> hash2;

int n, c[MAXV][MAXV], r[MAXV][MAXV], source, sink;
int dis[MAXV], maxflow, gap[MAXV];


void bfs()
{
    int v, i;
    queue <int>q;
    memset(dis, 0, sizeof(dis));
    memset(gap, 0, sizeof(gap));
    gap[0]++;
    q.push(sink);
    while (!q.empty())
    {
        v = q.front(); q.pop();
        for (i = 0; i <= sink; i++)
        {
            if (!dis[i] && c[i][v]>0)
            {
                dis[i] = dis[v] + 1;
                gap[dis[i]]++;
                q.push(i);
            }
        }
    }
}


void sap() {
    int top = source, pre[MAXV], i, j, low[MAXV];
    bfs();                              //锟街诧拷

    memset(low, 0, sizeof(low));               //锟斤拷
锟斤拷路锟斤拷锟斤拷锟斤拷小锟斤拷锟斤拷
    while (dis[source]<n)
    {
        low[source] = INF;
        for (i = 0; i <= sink; i++)
        {           //锟揭碉拷一锟斤拷锟斤拷锟斤拷
            if (r[top][i]>0 && dis[top] == dis[i] + 1
&& dis[i] >= 0) break;
        }
        if (i <= sink)
        {                   //锟揭碉拷锟斤拷

            low[i] = min(r[top][i], low[top]);      //锟
斤拷锟斤拷锟斤拷小锟斤拷锟斤拷

            pre[i] = top; top = i;              //锟斤拷
录锟斤拷锟斤拷路锟斤拷
            if (top == sink)
            {               //锟揭碉拷一锟斤拷锟斤拷锟斤拷
路锟斤拷锟斤拷锟较诧拷锟斤拷
```

```
                cout << "!!!!!!!!!!!!!!!!!!!!!!!!!!!!"
<< endl;
                maxflow += low[sink];
                j = top;
                stack<int> st;
                while (j != source)
                {
                    i = pre[j];
                    cout << "(" << i-1 << "," << j-1 <<
")" << low[sink] << endl;
                    st.push(hash2[j - 1]);
                    r[i][j] -= low[sink];
                    r[j][i] += low[sink];
                    j = i;
                }
                st.push(hash2[i - 1]);
                while (!st.empty())
                {
                    cout <<"   "<< st.top() << "   "  <<
"->";

                    st.pop();
                }
                cout << endl;
                top = source;                 //锟劫达拷头
锟斤拷一锟斤拷锟斤拷锟斤拷路锟斤拷
                memset(low, 0, sizeof(low));
            }
        }
        else
        {                             //锟揭诧拷锟斤拷锟斤
拷锟斤拷一锟斤拷锟斤拷锟斤拷锟斤拷锟较撅拷锟斤拷锟斤拷锟斤拷
            int mindis = INF;
            for (j = 0; j <= sink; j++)
            {
                if (r[top][j]>0 && mindis>dis[j] + 1 &&
dis[j] >= 0)
                    mindis = dis[j] + 1;
            }
            gap[dis[top]]--;
            if (gap[dis[top]] == 0)
                break;
            gap[mindis]++;
            dis[top] = mindis;
            if (top != source)
                top = pre[top];
        }
    }
}

int nedges, power_stations, consumers;

int main()
{
    recode();
    int i;
    int x, y, z;
    char ch;
    //while (scanf_s("%d%d%d%d", &n, &power_stations,
&consumers, &nedges) != EOF)
    //{
        //Init
    cout << "n" << n << "power_stations" <<
power_stations << "consumers" << consumers << "nedges"
<< nedges << endl;
        memset(r, 0, sizeof(r));
        memset(c, 0, sizeof(c));
        source = 0; sink = n + 1; n += 2; maxflow = 0;
        //Read Gragh
        for (i = 1; i <= nedges; i++)
        {           //锟斤拷锟矫讹拷图锟斤拷 1 锟斤拷始
            cin >> ch >> x >> ch >> y >> ch >> z;
            c[x + 1][y + 1] = r[x + 1][y + 1] = z;
        }
        //Build Gragh
        //锟斤拷锟斤拷锟斤拷锟斤拷源锟斤拷指锟斤拷锟斤拷锟叫
的凤拷锟斤拷站
        for (i = 1; i <= power_stations; i++)
        {
            cin >> ch >> x >> ch >> y;
            c[source][x + 1] = r[source][x + 1] = y;
        }
        //锟斤拷锟斤拷锟斤拷锟斤拷锟斤拷悖  癸拷锟斤拷锟斤拷
锟斤拷锟斤拷锟街革拷锟斤拷锟
        for (i = 1; i <= consumers; i++)
        {
            cin >> ch >> x >> ch >> y;
            c[x + 1][sink] = r[x + 1][sink] = y;
        }
        sap();
        printf("%d\n", maxflow);
        /*for (int i = 0; i < MAXV; i++)
        {
            for (int j = 0; j < MAXV; j++)
            {
                if(r[i][j])
                    cout << "(" << i-1 << "," << j-1 <<
")" << r[i][j] << endl;
            }
        }
        cout << endl;
        for (int i = 0; i < MAXV; i++)
        {
            for (int j = 0; j < MAXV; j++)
            {
                if (c[i][j])
```

```
                    cout << "(" << i - 1 << "," << j -
1 << ")" << c[i][j] << endl;
            }
        }*/
        //break;
    //}
    return 0;
}
```

//Record.cpp

```
#include "head.h"

struct node
{
    int x;
    int y;
    int z;
};

extern map<int, int> hash1;
extern map<int, int> hash2;
vector<node> record;
int flag = 0;



void pre1(int x)
{
    if (hash1.find(x) == hash1.end())
    {
        pair<int, int>temp;
        temp.first = x;
        temp.second = flag;
        pair<int, int>temp1;
        temp1.first = flag;
        temp1.second = x;
        flag++;
        hash1.insert(temp);
        hash2.insert(temp1);
    }
}

void pre2(int x,int y,int z)
{
    struct node temp;
    temp.x = hash1[x];
    temp.y = hash1[y];
    temp.z = z;
    record.insert(record.end(), temp);
}

extern int n, nedges, power_stations, consumers;

int recode()
{
    int i;
    int x, y, z;
    char ch;
    while (scanf_s("%d%d%d%d", &n, &power_stations,
&consumers, &nedges) != EOF)
    {
        for (i = 1; i <= nedges; i++)
        {
            cin >> ch >> x >> ch >> y >> ch >> z;
            pre1(x);
            pre1(y);
            pre2(x, y, z);
        }


        ofstream out("d:\\out.txt");
        if (out.is_open())
        {
            for (int i = 0; i < record.size(); i++)
            {
                out << '(' << record[i].x << ',' <<
record[i].y << ')' << record[i].z << endl;
```

```
            }
        }
        else
        {
            cout << "something is wrongA" << endl;
        }

        for (i = 1; i <= power_stations; i++)
        {
            cin >> ch >> x >> ch >> y;
            out << '(' << hash1[x] << ')' << y << endl;
        }
        for (i = 1; i <= consumers; i++)
        {
            cin >> ch >> x >> ch >> y;
            out << '(' << hash1[x] << ')' << y << endl;
        }
        pair<int, int>temp;
        temp.first = -1;
        temp.second = -1;
        hash2.insert(temp);
        temp.first = flag;
        temp.second = -2;
        hash2.insert(temp);
        break;
    }
    return 0;
}
```