2011 高教社杯全国大学生数学建模竞赛

承　诺　书

我们仔细阅读了中国大学生数学建模竞赛的竞赛规则。

我们完全明白，在竞赛开始后参赛队员不能以任何方式（包括电话、电子邮件、网上咨询等）与队外的任何人（包括指导教师）研究、讨论与赛题有关的问题。

我们知道，抄袭别人的成果是违反竞赛规则的，如果引用别人的成果或其他公开的资料（包括网上查到的资料），必须按照规定的参考文献的表述方式在正文引用处和参考文献中明确列出。

我们郑重承诺，严格遵守竞赛规则，以保证竞赛的公正、公平性。如有违反竞赛规则的行为，我们将受到严肃处理。

我们参赛选择的题号是（从 A/B/C/D 中选择一项填写）：B

我们的参赛报名号为（如果赛区设置报名号的话）：

所属学校（请填写完整的全名）：上海交通大学

参赛队员 (打印并签名) ：1. 杜若飞

2. 凌宇霄

3. 田金

指导教师或指导教师组负责人 (打印并签名)：数模指导组

日期：2011 年 9 月 11 日

赛区评阅编号（由赛区组委会评阅前进行编号）：

2011 高教社杯全国大学生数学建模竞赛

编 号 专 用 页

赛区评阅编号（由赛区组委会评阅前进行编号）：

赛区评阅记录（可供赛区评阅时使用）：

评阅人										
评分										
备注										

全国统一编号（由赛区组委会送交全国前编号）：

全国评阅编号（由全国组委会评阅前进行编号）：

交巡警服务平台的设置与调度

摘　要

本文建立了全城区路口以及交巡警平台的无向图、二分图模型，该模型有助于在遵循交巡警服务平台的原则和任务的基础上，解决交巡警服务平台的分配问题，进而大幅减少因交巡警服务平台分布不均匀带来的社会不安定因素。在城区路口、交巡警平台、要道、犯罪嫌疑人作案点的坐标给定的情况下，我们根据所提出的问题和给定的数据，通过二分法、匈牙利算法、调整算法以及评价模型给出了交巡警平台的完整调度方案和围堵方案。

对于问题一，我们对某市城区地图及交巡警平台构造了无向图模型。利用 Floyd 算法，我们得出全城区两点间的最短路径。为了解决最短时间将有限的警力分配到城区出入口，我们对交巡警平台和城区交通要道构建了二分图。通过二分法和匈牙利算法，可以得出交巡警平台警力的最佳调度方案。最终，我们在 8 分钟内封锁了城区交通要道。为了评价交巡警平台设置的合理性，我们提出了基于负载指数和不安指数的评价模型。为了解决现有服务平台工作量不均衡，我们在 A 区内增加了 1 个平台；为了解决有些地方出警时间过长的实际情况，我们在 A 区内增加了 4 个平台，最终实现了 A 区全部路口可在 3 分钟之内到达，且全城路口不安指数较低，服务平台负载指数较低的目标。

对于问题二，我们依然利用上述评价模型，加上调整算法，按照设置交巡警平台的原则和任务，综合考虑现有服务平台工作量不均衡，出警时间过长的情况，给出了新的平台配置方案。该方案保证了全城所有路口可在 8 分钟以内到达。最后，对于实际抓捕犯罪嫌疑人，我们设置了城市包围圈和城区包围圈，给出了调度全市警力资源的围堵方案，在 13 分钟之内完成任务。

最终，我们利用 C++编程，输出了所用方案，并验证了其合理性。通过对建立模型的方法、假设进行评价与分析，提出了进一步改进的方法。

关键词：交巡警平台　调度　配置　匈牙利算法　不安指数

一、 问题重述

1.1 背景资料与条件

每个交巡警服务平台的职能和警力配备基本相同。由于警务资源是有限的，如何根据城市的实际情况与需求合理地设置交巡警服务平台、分配各平台的管辖范围、调度警务资源是警务部门面临的一个实际课题。

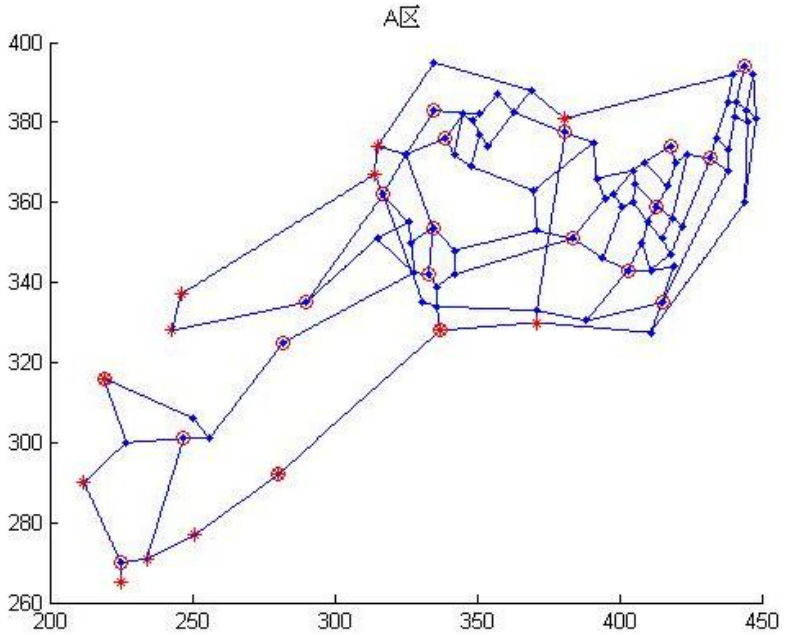

图 1：A 区的交通网络与平台设置的示意图

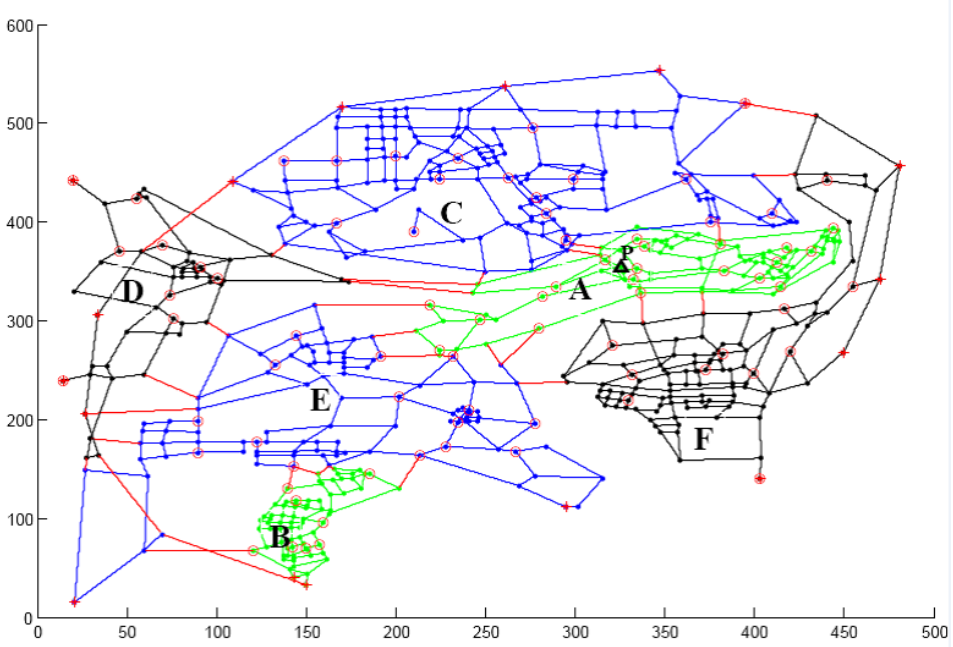

图 2：全市六区交通网络与平台设置的示意图

说明：

（1）图中实线表示市区道路；红色线表示连接两个区之间的道路；
（2）实圆点“ ”表示交叉路口的节点，没有实圆点的交叉线为道路立体相交；
（3）星号“*”表示出入城区的路口节点；
（4）圆圈“○”表示现有交巡警服务平台的设置点；
（5）圆圈加星号“ ”表示在出入城区的路口处设置了交巡警服务平台；
（6）附图 2 中的不同颜色表示不同的区。

1.2 需要解决的问题

根据上述某市设置交巡警服务平台的相关情况，建立数学模型分析研究下面的问题：

（1）图 1 给出了该市中心城区 A 的交通网络和现有的 20 个交巡警服务平台的设置情况示意图，具体数据已给定。请为各交巡警服务平台分配管辖范围，使其在所管辖的范围内出现突发事件时，尽量能在 3 分钟内有交巡警（警车的时速为 60km/h）到达事发地。

对于重大突发事件，需要调度全区 20 个交巡警服务平台的警力资源，对进出该区的 13 条交通要道实现快速全封锁。实际中一个平台的警力最多封锁一个路口，请给出该区交巡警服务平台警力合理的调度方案。

根据现有交巡警服务平台的工作量不均衡和有些地方出警时间过长的实际情况，拟在该区内再增加 2 至 5 个平台，请确定需要增加平台的具体个数和位置。

（2）图 2 给出了全市（主城六区 A，B，C，D，E，F）的示意图，按照设置交巡警服务平台的原则和任务，分析研究该市现有交巡警服务平台设置方案的合理性。如果有明显不合理，请给出解决方案。

如果该市地点 P（第 32 个节点）处发生了重大刑事案件，在案发 3 分钟后接到报警，犯罪嫌疑人已驾车逃跑。为了快速搜捕嫌疑犯，请给出调度全市交巡警服务平台警力资源的最佳围堵方案。

二、 问题分析

2.1 问题的重要性分析

随着人口的不断增长和城镇化进程的不断加快，社会生活中突发事件（比如犯罪，交通事故等）的发生率也在不断提高。正所谓“有问题找警察”，面对突发时间，警察机关必须在第一时间赶到现场做出处理。但是即使在科技高度发达的今天，面对许多突发事件的时候，人们还是赶到束手无策。“交巡警平台”是近年来兴起的一种警务模式，他们拥有高精尖的设备仪器，肩负着保护人民生命财产不受损失的重大责任。因此，如何合理地设置和调度交巡警平台是一个非常重要的课题。

同时，通过解决这个问题所建立的一些数学模型和方法也可以推广到许多类似的领域中进行应用，从而可以解决一系列的实际生产生活问题。因此，研究“交巡警服务平台的设置于调度”这个问题具有实际意义，因此显得尤为重要。

2.2 问题的思路分析

该问题给出交通网络，很容易联想到图论模型。这样我们便将实际问题数学

化，该问题属于确定性模拟与数据分析问题，计算量较大，但是由于计算机编程软件的使用，可以使手工计算量大大减少，具有求解可行性。

交巡警平台和路口之间存在单射关系，容易联想到二分图最大匹配模型。故可采用二分法结合匈牙利算法解决，实用性较强。

三、 基本假设

3.1 评价模型假设

1. 一个交巡警的警力最多封锁一个路口；
2. 警车的时速（60km/h）不会因其他情况（如堵车，转弯等）而改变；
3. 警车在 3 分钟之内能够到达案发路口可以保证及时处理突发事件或者罪犯没有逃离该路口。

3.2 围堵方案假设

1. 一个交巡警的警力最多封锁一个路口；
2. 警车的时速（60km/h）不会因其他情况（如堵车，转弯等）而改变。

四、 符号说明

G：城区建立的无向图
W：图的权值矩阵
V：图的顶点集
E：图的边集
α：发案率

$w_{u,v}$：节点 u,v 之间的初始权值

$d_{u,v}$：节点 u,v 之间的最短距离

n_{u}：3 分钟之内交巡警平台 u 能够到达的路口数目
m_v：3 分钟之内能够到达路口 v 的交巡警平台的数目
ψ_u：交巡警平台 u 的负载指数
β_u：交巡警平台 u 的权值
ζ_{v}：路口 v 的不安指数

五、 模型的建立与求解

5.1 问题一

5.1.1 问题一概述

问题一分为三个问题：分配管辖区域，使交巡警能够及时赶到案发地；给出能够快速封锁 13 条交通要道的方案；增加若干交巡警服务平台。利用图论和评价模型可以解决以上问题。

5.1.2 模型一的建立与求解

首先，对于第一小问，将路口节点抽象为无向图中的节点，道路抽象为无向

图的边，边$< i,j >$的权值$w(i,j)$设定为道路的长度，由此对 A 城区建立无向图模型$G(V,E,W)$，并得到图的权值矩阵W。

这样运用 Floyd 算法[1]，可得出两点间的最短距离，算法过程如下：

1. $k=1$, 对于边(u,v)，最短路 $d(u,v)=w(u,v)$
2. $k \geq 1,$，对于所有顶点对 (u,v)，若 $d(u,k)+d(k,v)>d(u,v)$，则令 $d(u,v)=d(u,k)+d(k,v)$。
3. 若$k=1$，结束；否则转 2。

算法的时间复杂度：$O(n^3)$

接下来，我们可以求解对于路口 i，距离其最近的交巡警平台 j 的编号及其距离。令平台 j 负责路口 i，我们便为各交巡警服务平台分配了管辖范围，且保证了路口 i 出现紧急事故，巡警可在最短时间内到达该处。

交巡警平台编号	管辖的路口编号
A1	1 67 68 69 71 73 74 75 76 78
A2	2 39 40 43 44 70 72
A3	3 54 55 65 66
A4	4 57 60 62 63 64
A5	5 49 50 51 52 53 56 58 59
A6	6
A7	7 30 32 47 48 61
A8	8 33 46
A9	9 31 34 35 45
A10	10
A11	11 26 27
A12	12 25
A13	13 21 22 23 24
A14	14
A15	15 28 29
A16	16 36 37 38
A17	17 41 42
A18	18 80 81 82 83
A19	19 77 79
A20	20 84 85 86 87 88 89 90 91 92
3 分钟之内不能到达的路口：28 29 38 39 61 92	

第二小问是调度全区 20 个交巡警服务平台对出入该区的 13 条交通要到实现快速全封锁，我们为此建立了如下的二分图模型：

[1] 算法证明请参见参考文献[1]

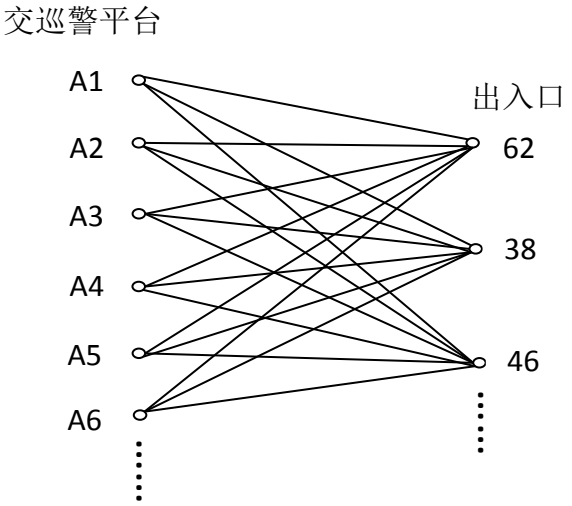

图 3

由于实际一个平台的警力最多封锁一个路口，所以在二分图模型中，左侧节点仅能匹配一个右侧节点。设交巡警平台点集为X，路口点集为Y，二分图匹配中选中的点集为X', Y'; $i \in X, j \in Y$ 则我们要求

$$\max\{d(i,j), i \in X', j \in Y'\}$$

最小。

如果枚举边的权值，则时间复杂度为$O(\max\{d(i,j) \times |X|^2|Y|, i \in X, j \in Y\}$。

为此，我们采用二分算法结合匈牙利算法解决这个问题，过程如下：

1. 令 $l = 0, r = \max\{d(i,j), i \in X, j \in Y\}$
2. 若$l > r$程序终止。令 $\text{mid} = \frac{\text{l+r}}{2}$ ，二分图邻接矩阵中，所有权值大于 mid 的边设为 0，否则设为 1
3. 进行匈牙利算法[2]，设置空点集 S，深度优先搜索找出增广路径 P，若能找到，则有更大的匹配点集 S' 代替S，否则输出匹配结果。
4. 若最大匹配数目等于路口点个数，则令 $\text{r} = \text{mid} - 1$; 否则 $\text{l} = \text{mid} + 1$，回到 2

这样，我们便得到最大边权值最小的一个二分图匹配结果，在 8 分钟之内迅速封锁了 A 区全部出入口，分配方案如下：

交巡警平台编号	负责封锁路口	最短路径长度/m	用时/min
A1	62	4885.2167	4.8552
A2	38	3982.1859	3.9822
A3	16	6025.5657	6.0255
A4	48	7395.8694	7.3958

[2] 匈牙利算法，匈牙利数学家 Edmonds 于 1965 年提出。具体请参见参考文献[2]

A5	24	3182.9326	3.1829
A7	29	8015.4568	8.0155
A10	22	7707.9177	7.7079
A11	12	3791.3528	3.7913
A12	24	3591.6300	3.5916
A13	23	500	0.5
A14	21	3264.9655	3.2650
A15	28	4751.8417	4.7518
A16	14	6741.6615	6.7417

程序运行结果显示，有 6 个路口不能保证巡警在 3 分钟之内到达。

第三小问是确定需要增加平台的具体个数和位置。

我们建立一个评价模型，基于模型假设，我们创造了两个评价指数——负载指数和不安指数。

负载指数：用于评价交巡警服务平台的工作量，工作量由出警次数决定。

负载指数的计算：

首先，选定编号为 u 的交巡警服务平台，我们将从该服务平台到达事发地时间没有超过 3 分钟的路口节点确定为与其相关的路口节点（节点总数为 n_u）。然后，我们考虑到，从交巡警服务平台到达事发地的时间（最多 3 分钟）与处理事件的时间相比，到达事发地的时间完全可以忽略不计。所以，我们不考虑服务平台到路口节点距离的因素，仅根据路口节点 v 的发案率 α_v（次数）和路口节点与 3 分钟范围内的交巡警服务平台个数m_v，计算节点权值。基于假设 3——警车在 3 分钟之内能够到达案发路口可以保证及时处理突发事件或者罪犯没有逃离该路口，我们认为出入城区的路口节点与一般路口节点在此处没有区别。最后，利用公式计算出编号为 i 交巡警服务平台的负载指数 ψ_u。

$$\psi_u = \sum_{\mathrm{v}=1}^{\mathrm{n_u}} \frac{\alpha_v}{m_v} \quad (\mathrm{I})$$

不安指数：用于评价路口节点处的发案情况与案发后是否有交巡警能尽快到达的安全状况。不安指数数值越大，表示次路口节点越不安全。

不安指数的计算：

首先，选定一个路口节点v，按照求负载指数方法中的定义，找出与其相关的交巡警服务平台。然后，根据服务平台 i 的负载指数 ψ_u 与服务平台与路口节点之间的距离 $d_{u,v}$ 的乘积计算服务平台权值 β_i。即：

$$\beta_u = \psi_u \cdot d_{u,v} \quad (\mathrm{II})$$

最后运用公式Ⅰ、Ⅱ，对路口 v 求得不安指数 ζ，其中 u 是 3 分钟内可达 v 的服务平台：

$$\zeta_{\mathrm{v}} = \frac{1}{\sum_{u=1}^{m_u} \frac{1}{\beta_u}} \quad (\mathrm{III})$$

我们只需限定增设服务平台数，对每一套可以清除所有 3 分钟内不可到达的路口节点的增设方案进行运算，当所有路口节点的不安指数和取最小值时为最佳

增设方案。

根据上述想法，我们编写程序进行运算，并得到结果如下：

路口编号	原始不安指数	建平台后不安指数
29	1000000000.000000	4006301988.447613
28	1000000000.000000	4006302937.130911
61	1000000000.000000	5006214672.900473
39	1000000000.000000	4006169640.566601
92	1000000000.000000	5005921815.677416
38	1000000000.000000	4006177358.894831
54	290673.465856	6005747210.137786
53	181477.160915	6005559738.596028
21	170623.791576	6006092343.698461
40	167993.849681	4006503067.340547

通过比较不同的增设交巡警平台的方案得到的总不安指数，可以得到最佳的增设方案：

增设平台方案	总不安指数
28 38 61 92	5747817.909411
29 38 61 92	5746869.226113
28 39 61 92	5740099.581181
29 39 61 92	5739150.897883

从上述表格中可以看到，增设的平台编号是：29，39，61，92。

运用 Excel 对每个节点的不安指数进行数值上的处理，作图如下：

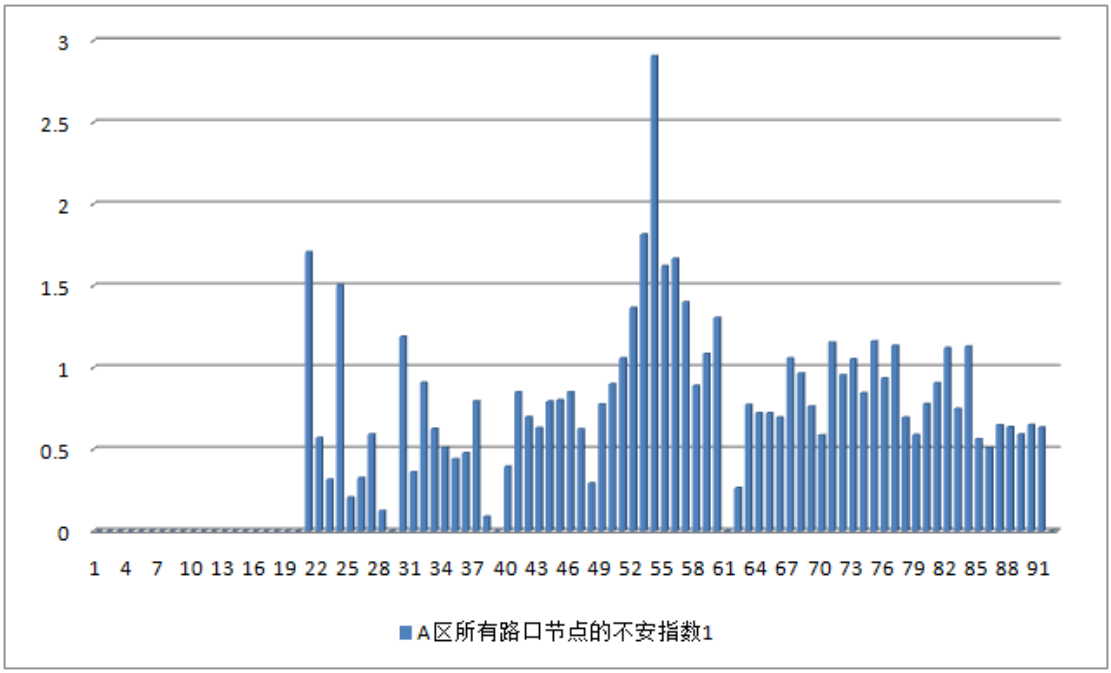

图 4

我们发现存在一个节点数值明显偏高，于是考虑此路口节点（编号 54）增设服务平台。

做同样处理，作图如下：

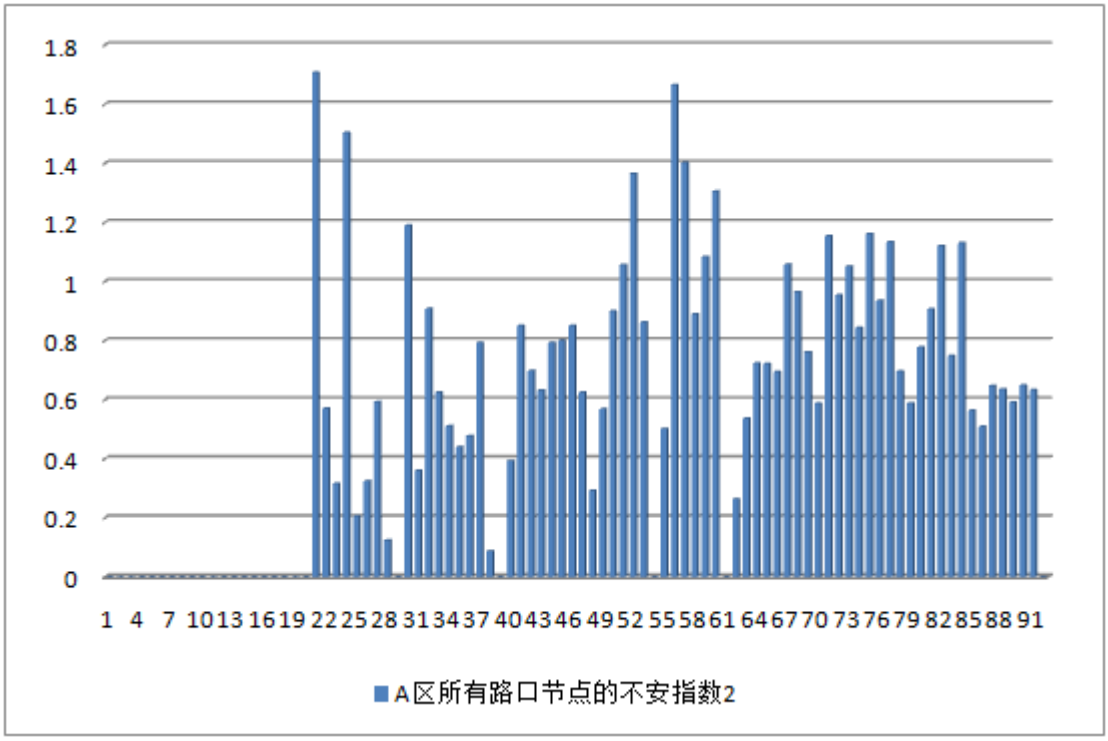

图 5

很明显，安全状况得到改善。所以，我们得出最终结论，需要增设的平台编号为：29，39，61，92，54。

5.2 问题二

5.2.1 问题二概述

问题二分为两个问题：评价该市目前交巡警服务平台设置方案的合理性，如果有明显不合理，请给出解决方案；罪犯逃跑，给出最佳围堵方案。第一小问是全局配置问题；第二小问是资源分配问题，要注意把握好时间最短和最有效封堵等原则。

5.2.2 模型二的建立与求解

同问题一，我们对全城区建立无向图模型$G(V,E)$，并得到图的权值矩阵M，再一次运用 Floyd 算法，得出两点间的最短距离。考虑到全市人口密度 0.23（万人/平方公里）比 A 区人口密度 2.73（万人/平方公里）小很多，为了评价更为准确，我们将评价模型假设中的 3 分钟改为 8 分钟，经过处理，得到全城区的不安指数柱状图：

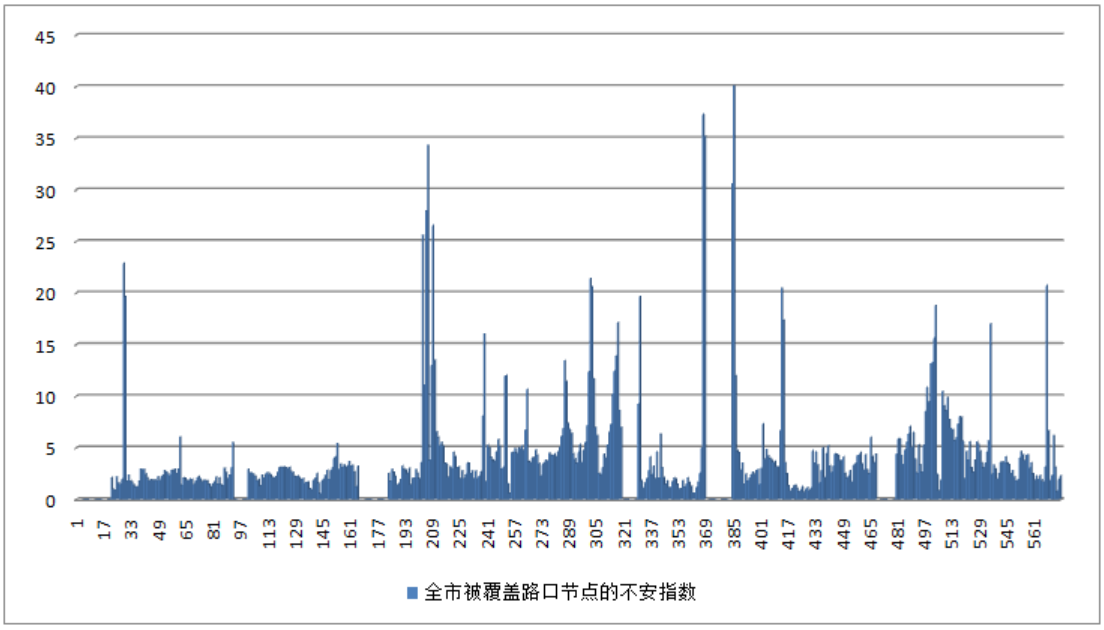

图 6

我们发现存在不少不安指数极高的路口节点，并且存在不能被覆盖[3]（8 分钟）的路口节点。

因此，我们设计了调整算法设法降低不安指数，算法过程如下：

1. 设置迭代次数 X
2. 若迭代完成，则退出，否则进行 3
3. 随机选取当前交巡警平台 u，随机选取其邻接点 v
4. 将 i 调整到 j，重新计算节点 u,v 所覆盖[4]路口的不安指数，若不安指数变小，则确认调整，否则将 i 还原到曾经的位置。
5. 回到 2。

这样，我们得到调整后的结果（增加了 1 个交巡警平台），并在不安指数极高的 3 个点分设了新的交巡警平台，得到新的方案如下：

区域编号	调整之前的交巡警平台位置编号	调整之后的交巡警平台位置编号
A	1,2,3,4,5,6,7,8,9,10,11,12, 13,14,15,16,17,18,19,20	6,11,13,14,15,32,34,39,40, 44,49,52,55,64,72,81,89
B	93,94,95,96,97,98,99,100	110,112,116,130,139,154,163
C	166,167,168,169,170,171,172,173, 174,175,176,177,178,179,180,181,182	169,185,192,201,206,217,224,240,243 250,254,263,277,287,297,301,307,313
D	320,321,3222,323,324,325,326,327,328	327,332,334,344,349,351,352, 362,363,364,365,370
E	372,373, 374, 375, 376, 377, 378, 379, 380,381,382,383,384,385,386	372,374,377,384,388,392,398,401, 407,419,420,422,423,453,462,473

[3] 这里的“不能被覆盖”节点特指 8 分钟以内最近的交巡警平台无法到达的节点

[4] 这里的“所覆盖”节点指 8 分钟以内从u,v出发可以到达的节点

F	475,476,477,478,479,480,481, 482,483,484,485	476,479,483,488,495,503,511,534, 540,552,556,563,571,574
总数	80	84
新增交巡警平台：332,169,254,574		

经过处理，得到全城区的不安指数柱状图：

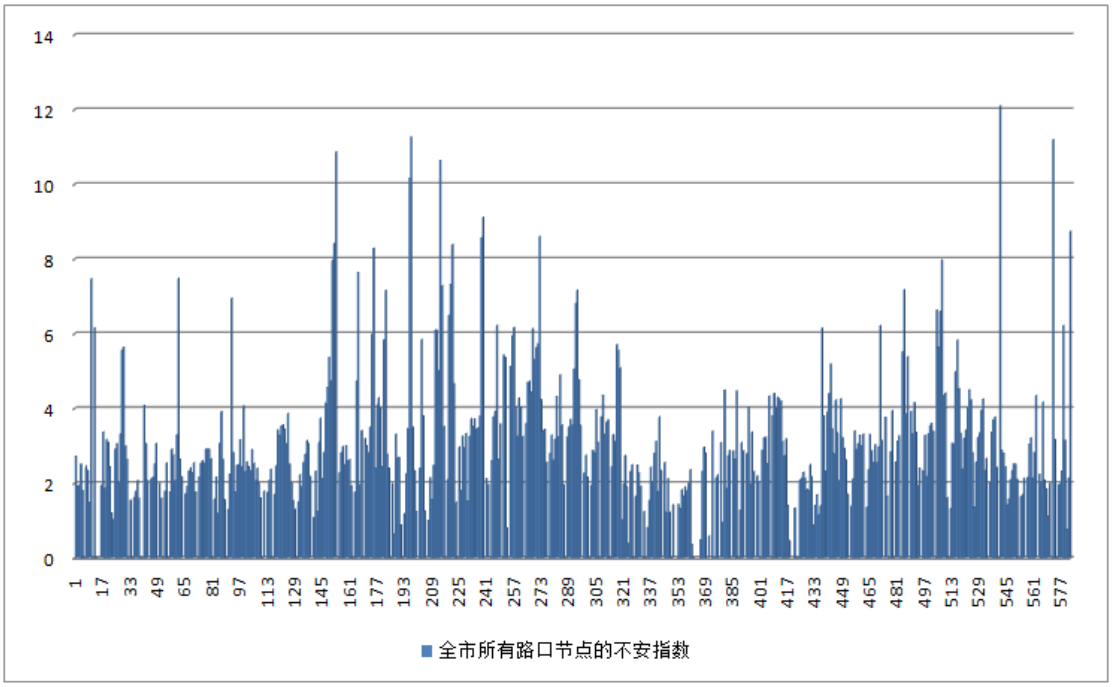

图 7

不安指数较小且分布相对平稳，且相对之前的柱状图有大幅降低，方案合理。

为了围堵犯罪嫌疑人，我们采用了“双包围圈”的战术，为了确保能将犯罪分子控制在城市和各个区域中，分别围堵了各个城区的各个出入口。

同问题一的第二小问解决方案，我们将全区的交巡警平台、城区出入口了构建了二分图，二分最大时间，用匈牙利算法求得最大匹配，

具体方案如下：

交巡警平台	应封锁路口编号	最短路径长度/m	用时/min
A1	561	8183.1078	8.1831
A2	560	9828.3678	9.8284
A3	30	8199.5603	8.1996
A4	183	11561.817	11.562
A5	239	10259.727	10.26
A6	14	13000.213	13
A7	235	2987.9947	2.988
A8	21	12698.909	12.699
A9	486	13489.623	13.49
A10	470	10814.627	10.815

A11	12	3791.3528	3.7914
A12	471	6400	6.4
A13	383	3801.5148	3.8015
A14	23	6473.2797	6.4733
A15	371	11755.412	11.755
A16	459	10996.071	10.996
A17	38	4755.703	4.7557
A18	62	6734.3619	6.7344
A19	48	11998.607	11.999
A20	190	8751.0529	8.7511
B1	392	13086.616	13.087
B2	388	12395.77	12.396
B3	151	6560.172	6.5602
B4	389	11162.587	11.163
B5	100	4054.171	4.0542
B6	93	5255.8378	5.2558
B7	153	4470.3116	4.4703
B8	387	12680.274	12.68
C1	369	10082.224	10.082
C2	370	9657.4108	9.6574
C3	581	11364.604	11.365
C4	237	13610.135	13.61
C5	29	13373.597	13.374
C6	28	10626.031	10.626
C7	203	12139.697	12.14
C8	16	7679.0614	7.6791
C9	0	0	0
C10	177	9021.3593	9.0214
C11	582	12645.298	12.645
C12	578	11021.759	11.022
C13	202	13186.331	13.186
C14	317	9756.3365	9.7563
C15	264	11913.951	11.914
C16	248	11048.834	11.049
C17	264	12055.059	12.055
D1	447	13025.407	13.025
D2	362	10753.595	10.754
D3	474	11551.11	11.551
D4	340	11385.077	11.385
D5	361	7953.044	7.953
D6	325	0	0
D7	328	11303.05	11.303
D8	444	10765.962	10.766

D9	329	11107.348	11.107
E1	377	11265.164	11.265
E2	101	9619.2008	9.6192
E3	418	13199.66	13.2
E4	161	10220.69	10.221
E5	382	9864.5582	9.8646
E6	165	3511.41	3.5114
E7	415	13487.856	13.488
E8	101	12188.146	12.188
E9	390	8573.1721	8.5732
E10	330	12099.667	12.1
E11	331	13611.426	13.611
E12	24	4632.5448	4.6325
E13	339	12902.52	12.903
E14	332	12913.253	12.913
E15	332	7619.8101	7.6198
F1	541	12077.219	12.077
F2	458	11303.356	11.303
F3	372	13145.162	13.145
F4	483	10938.625	10.939
F5	177	12670.611	12.671
F6	572	8405.7782	8.4058
F7	22	12659.294	12.659

我们的最大用时只有 13.4 分钟，而逃犯若以 60km/h 的速度计算逃离城市的最短用时也需要 21.8 分钟，即使考虑 3 分钟的报警时间，我们的方案也完全可行，进一步的分析请见模型的分析、检验。

六、 模型的分析、检验

6.1 假设的合理性分析

首先，一个交巡警的警力最多封锁一个路口和警车的时速（60km/h）不会因其他情况（如堵车，转弯等）而改变，这两条假设是题目的基本假设。为了便于研究，这两条假设很好的化简了现实中原有的复杂问题。

其次，警车在 3 分钟之内能够到达案发路口可以保证及时处理突发事件或者罪犯没有逃离该路口，这条假设是基于题目中给出尽量能在 3 分钟内有交巡警到达事发地的要求。根据题目的暗示提出假设合情合理，同样也化简了问题的复杂程度。

6.2 评价模型的可靠性分析

问题一第三小问中，不安指数柱状图 4 中 54 节点的不安指数明显偏高，而在 54 路口节点增设交巡警平台后得出的图 5 中发现 54 节点不安指数明显下降。

证明了评价模型的可靠性。同样的结论在问题二第一小问中也可以得到证明。而且，我们运用次评价模型很好得解决了实际问题。

6.3 围堵方案的合理性分析

首先，我们要确保能将犯罪分子控制在城市中，否则无法开展搜捕行动。因为我们无法预知逃犯的逃跑速度和路线，所以警方在最短的时间内将城市最外围的所有出入口封锁必须被最先考虑。

接下来，对于其他暂时未分配参与围堵的交巡警平台，我们考虑将其分配至逃犯可能经过的概率最大的路口。我们分析了逃犯从 P 点逃跑的最佳路径，发现区与区之间的出入口大多出现在逃跑的最佳路径上。又考虑到，如果能将逃犯围堵在某个区里，那么将更加便于开展快速搜索。综合以上两点的考虑，我们第二步考虑封锁区与区之间的出入口。当然，以上两步同时执行。剩下的警力可以作为机动部队参与动态围堵。

可以发现，静态的围堵部署与动态的机动部队相结合组成了我们的最佳围堵方案。

七、模型评价与优化

7.1 模型的优缺点分析

7.1.1 模型的优点

(1)采用了较为合理的假设，充分利用了题目和附件中给出的信息和数据，并且联系了实际，灵活应用了数学建模的理论和方法，建立了模型；

(2)该模型采用的方法经典，经得起推敲，具有科学性和合理性；

(3)该模型建立思路清晰，方法明确，具有很强的可操作性；

(4)该模型经过了一定的验证与分析，具有可靠性。

总之，该模型可以给予实际中的交巡警服务平台的设置一定的参考，可以使得平台的设置更加合理，围堵和追捕罪犯时的效率更加高效等。

7.1.2 模型的缺点

(1)为了得到较为简单的评价模型，化简了一些条件，使得模型不够精确。

(2)评价模型假设中时间常量的选取需要更多数据的支撑，否则模型不够完善。

7.2 模型的优化

(1)实际中，无法保证警察在接到报警后 3 分钟之内到达案发地点时罪犯没有逃离该路口。如果案发地点在出入城区的路口，罪犯很可能逃出该区，增加了搜捕难度，所以，出入城区的路口节点相比一般路口节点会更加重要。根据这种情况，我们可以在计算负载指数的过程中，对出入城区的路口节点增加权重，比如对原有的节点权值乘以一个大于 1 的系数以达到目的。

(2)实际中，警车时速会受到路况的影响。根据每条道路的交通状况得出不同的警车时速，这样原先模型中与服务平台相关的路口节点会发生改变，更加贴近现实。

(3)如果已知每个路口处理案件时间的统计数据，我们可以利用双权重系数法（发案率 α 和处理案件时间 T ）计算路口节点权值，这样更加合理。
(4)为了计算出更加精确的不安指数，在计算不安指数时，在选取与路口节点相关的服务平台时，我们采用更长的时间标准（大于原先的 3 分钟）。

八、模型的推广

本模型较好的解决了交巡警服务平台的设置与调度问题，可以推广到实际生产生活中的很多问题。比如：城市消防救援平台的设置调度问题，重大安全生产事故的处理问题等等，都可以使用本模型或者其拓展模型进行处理。同时，本模型的思想与算法也可以在许多其他领域中得到应用。

九、参考文献

[1]（美）Thomas H. Cormen，Charles E. Leiserson，Ronald L. Rivest，Clifford Stein,《算法导论》,北京：机械工业出版社，2006.9
[2]（美）Douglas B. West,《图论导引》,北京：机械工业出版社，2006.2

附录

核心代码：
1. 为各交巡警服务平台分配管辖范围：
Floyd 算法求最短路：

```
for (int k = 1; k <= n; ++k)
for (int i = 1; i <= n; ++i)
for (int j = 1; j <= n; ++j)
if (dist[i][k] + dist[k][j] < dist[i][j])
{
    dist[i][j] = dist[i][k] + dist[k][j];
}
```

2. 二分算法及匈牙利算法：

```
//深度优先搜索找增广路
bool SearchPath(int u){
    int v;
    for (v = 0; v < vN; ++v)
    if(a[u][v] <= mid/100 && !chk[v])
    {
        chk[v] = true;
        if(yM[v] == -1 || SearchPath(yM[v]))
        {
            yM[v] = u; xM[u] = v;
            return true ;
        }
```

```
        }
        return false ;
}

//匈牙利算法：
int MaxMatch(){
        int u, ret = 0 ;
        memset(xM, -1, sizeof (xM));
        memset(yM, -1, sizeof (yM));
        for (u = 0; u < uN; u++)
                if (xM[u] == -1){
                        memset(chk, false, sizeof (chk));
                        if(SearchPath(u)) ret++;
                }
        return ret;
}
//读入权值矩阵
        for (int i = 0; i < uN; i++)
                for (int j = 0; j < vN; j++){
                        scanf("%lf", &a[i][j]);
                        if (a[i][j] > mr) mr = a[i][j];
                }

//二分答案
        l = 0; r = (int)(mr * 100); ans = r;

        while (l < r){
                mid = ((l+r)>>1);
                if (MaxMatch() == 75) {
                        if (mid < ans) ans = mid;
                        r = mid-1;
                } else {
                        l = mid+1;
                }
    }
```

3. 不安指数的计算：

```
double compute_nervous(){
        //计算不安系数
        double ans = 0;
        for (int i = 1; i <= n_a_total; ++i ) { nervous[i] = 0;    overload[i] = 0;    }

        for (int j = 1; j <= n_a_total; ++j ){
                min_dist[j] = oo;
```

```
        for (int i = 1; i <= police_a_total; ++i )
        {
            int k = police_pos[i];
            if (dist[k][j] < min_dist[j]){
                min_dist[j] = dist[k][j];
            }
        }
    }

    //计算负载指数
    for (int i = 1; i <= n_a_total; ++i )
        for (int j = 1; j <= police_a_total; ++j )
        if (dist[police_pos[j]][i] < speed * 3 && police_pos[j] != i) {
            overload[j] += occur[i];
        }

    //计算不安指数
    for (int i = 1; i <= n_a_total; ++i ){
        for (int j = 1; j <= police_a_total; ++j ) {
            if (dist[police_pos[j]][i] < speed * 3)
                nervous[i] += 1 / (overload[j] * dist[police_pos[j]][i]);
            }
        if (nervous[i] == 0) nervous[i] = oo; else nervous[i] = 1 / nervous[i];
        ans += nervous[i];
    }
    return ans;
}
```