交巡警服务平台的设置与调度

摘要：在我国经济社会快速发展进程中，警察的工作任务日益繁重。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。

问题一：

（1）题目要求在城区 A 的 20 个巡警服务台位置确定的情况下，按照尽量 3min 到达案发地的原则为各服务平台分配管辖范围。对于此问题本文建立最大集合覆盖模型，并利用数学软件 MATLAB 进行分配求解，最后得到 A 区现有每个巡警服务台的管辖范围如表 1。

（2）我们对于 13 条交通要道实现快速全封锁的问题，以所用时间最小为目标，引入 0-1 变量，建立该问题的 0-1 规划模型，并借助数学软件 LINGO 进行求解，求解结果表明需要 8.05 分钟可以实现快速封锁。

（3）由问题（1）的分配结果可知，在现有巡警服务台的设置下：①还有 6 个路口在案发时巡警不能在 3min 之内到达，即某些地方出警时间过长；②我们根据巡警服务台的工作量的方差定义工作量不均衡度，结果显示：此时服务台的工作量不均衡度为 8.4314。

为了解决上述出警时间过长与工作量不均衡的问题。我们建立集合覆盖的 0-1 规划模型，求解结果表明：在增加 4 个平台的情况下，可以解决出警时间过长的问题。在此基础上我们优化分配方案：在增加 4 个巡警服务台的情况下，使平台的工作量的不均衡度降为 3.0742。增加的 4 个巡警服务台的路口标号见表 8。

问题二：

（1）本文定义了两个评价原则，原则一：巡警能在 3min 之内到达案发路口；原则二：巡警服务台的工作量均衡度尽量小。根据以上两个原则对该市现有巡警服务台的设置方案的合理性进行评价，评价结果显示：①全市有 138 个路口，在案发时巡警不能在 3min 之内到达；②此时的不均衡度已达 40.3。基于上述两点，现有的巡警服务台设置不合理。

针对现有巡警服务台设置不合理的情况下，本文提出三种方案对设置进行优化调整。方案一：保持现有巡警服务台的个数和位置，再在其他路口增设巡警服务台；方案二：保持现有巡警服务台的个数，但对其位置进行调整；方案三：不考虑现有巡警服务台的设置情况，重新确定全城的最佳巡警服务台数目与位置。

（2）本问题实质是单目标规划问题，我们建立 0-1 规划模型，以巡警围堵时间最短为目标，以成功围堵为条件。对于巡警的成功围堵，可以转化为二部图的完全匹配，利用匈牙利算法，求得最佳围堵方案，原始方案和三种优化方案的求解结果见表 18、表 19、表 20 和表 21。

关键字：最大集合覆盖　0-1 规划模型　　MATLAB 软件　LINGO 软件　二部图

一、 问题重述

警察肩负着刑事执法、治安管理、交通管理、服务群众四大职能。为了更有效地贯彻实施这些职能，需要在市区的一些交通要道和重要部位设置交巡警服务平台。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。试就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：

问题一：

(1)根据附件中的图与数据，为城区 A 中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。

（2）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。

（3）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。

问题二：

（1）针对全市（主城六区 A，B，C，D，E，F）的具体情况，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（参见附件）的合理性。如果有明显不合理，请给出解决方案。

（2）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、 问题分析与建模思路

问题一：

（1）问题要求在城区 A 的 20 个巡警服务台位置确定的情况下，按照 3min 到达案发地的原则为各服务平台分配管辖范围。本文引入经典离散定位理论中的最大集合覆盖模型进行求解。

记 $I=\{1,2\cdots 92\}$ 为城区 A 的所有路口节点集合，$J=\{1,2\cdots 20\}$ 为城区 A 巡警服务台的节点集合，$c_{ij}\left(i\in I,j\in J\right)$ 为巡警服务台 j 到达路口 i 的最短距离。

引入 0-1 变量 $s_{ij}\left(i\in I,j\in J\right)$，当路口 i 分配给巡警服务台 j 管辖是为 1，当路口 i 不分配给巡警服务台 j 管辖是为 0。即：

$$s_{ij}=\begin{cases}1 & \text{，路口}i\text{分配给服务台}j\text{管辖}\\ 0 & \text{，路口}i\text{不分配给服务台}j\text{管辖}\end{cases}$$

由题目的要求可知，当 $c_{ij}\le 3km$ 时，路口 i 可能分配给巡警服务台 j，也可

能分配给其他可在 3min 到达 i 路口的巡警服务台，而不分配给平台 j，故有 $s_{ij}\leq 1$；当 $c_{ij}>3km$ 时，巡警服务台 j 不可能在规定的时间内到达路口 i，故此时路口 i 不能分配给巡警服务台 j 管辖，故此时 $s_{ij}=0$。

根据上述的分配原则及每个路口只由一个巡警服务台进行管辖、每个巡警服务台至少要管辖一个路口，可建立最大集合覆盖模型，并借助数学软件 MATLAB 进行求解。

（2）问题要求调度全区 20 个交巡警服务平台的警力资源，对进出 A 区的 13 条交通要道进行快速全封锁，且每个平台的警力最多封锁一个路口。本文将问题转化为：从 20 个服务平台中选出 13 个对 13 条交通要道进行封锁，且这 13 个平台所用的时间要最小的规划问题。

本文引入 0-1 变量表示一个巡警服务台是否封锁一条交通要道，从而建立这个问题的 0-1 规划模型，并借助数学软件 LINGO 进行求解。

（3）根据问题一（1）的分配方案可知：

当标号为 39、61、28、29、38、92 的路口有案件发生时，标号为 2、7、15、16、20 的巡警服务台的出警时间将超过 3min，即出警时间过长。

此时每个巡警服务台的工作量分别为：

表 1　按问题一（1）的分配方案 20 个巡警服务台的工作量

平台标号	1	2	3	4	5	6	7	8	9	10
工作量	10.3	8.3	5.6	6.6	9.7	2.5	9	5	8.2	1.6
平台标号	11	12	13	14	15	16	17	18	19	20
工作量	4.6	4	8.5	2.5	2.1	3.8	5.3	6.1	3.4	10.7

此时巡警服务台的工作量不均衡度为 8.4314。

由 1），2）可知现有巡警服务台的工作量极其不均衡且有些地方出警时间过长。针上述问题，题目要求再增加 2—5 个巡警服务台来解决上述问题。

本文首先建立集合覆盖的 0-1 规划模型，然后利用 MATLAB 对模型进行求解，可得到初步的分配方案，最后再引入工作量不均衡度，通过计算求解可确定增加巡警服务台的数目与位置。

问题二：

（1）本文定义了两个评价原则：

原则一：巡警能在 3min 之内到达案发路口

原则二：巡警服务台的工作量均衡度尽量小。

根据以上两个原则对该城区现有巡警服务台的设置方案的合理性进行评价。

若现有巡警服务台的设置不合理，本文则提出三种方案对全城的巡警服务台设置进行优化：

方案一：保持现有巡警服务台的个数和位置，再在其他路口增设巡警服务台；

方案二：保持现有巡警服务台的个数，但对其位置进行调整；

方案三：不考虑现有巡警服务台的设置情况，重新确定全城的最佳巡警服务台数目与位置。

（2）当该市某路口发生重大刑事案件时，犯罪嫌疑人已逃跑，由于在案发 3min 后巡警才能接到报警，为了快速搜捕嫌疑犯，将调度全市交巡警服务平台警力围堵嫌疑犯。因为警车相对于嫌疑犯车延迟三分钟行驶，而且巡警不知道嫌疑犯逃跑方向，所以此问题可转化为以下模型：对于任意时间 t，嫌疑犯驾车逃

跑的最大范围为：在$t+3$时间内嫌疑犯所有可能行驶路线所包含路口节点的并集，记为Q，将Q的边界点集记为∂Q。所谓最快围堵方案，即寻找一个最短时间t，适当的调配巡警警力，使其在t时间内能够到达边界点∂Q，这样嫌疑犯就被控制在区域Q中，此时嫌疑犯将无法逃脱。

三、基本假设与符号说明

3.1 基本假设

1.假设每个巡警服务台的职能和警力配备基本相同；
2.假设每个路口只由一个巡警服务台进行管辖；
3.假设每个巡警服务台至少管辖一个路口；
4.假设巡警都按最短路径到达各案发路口；
5.假设犯罪案件都在路口上发生；
6.假设在重大案件发生时，每个平台都有能够封锁一个路口的能力；
7.工作量：每个巡警服务台所管辖范围内的所有路口案发率之和；
8.出警时间：巡警到达案发路口所需时间；
9.假设逃犯的逃跑速度等于警车的行驶速度；
10.假设巡警在接到报案后并不知道逃犯的逃跑方向；

3.2 符号说明

1. c_{ij}：为巡警服务台 j 到达路口 i 的最短距离，其中：$i=1,2\cdots92$，$j=1,2\cdots20$；
2. u_{ij}：为路口 i 与路口 j 之间的最短距离，其中：$i=1,2\cdots92$， $j=1,2\cdots92$
2. $s_{ij}=\begin{cases}1 & \text{，路口}i\text{分配给服务台}j\text{管辖}\\0 & \text{，路口}i\text{不分配给服务台}j\text{管辖}\end{cases}$，其中：$i=1,2\cdots92$，$j=1,2\cdots20$；
3. $k_{ij}=\begin{cases}0 & u_{ij}>3km\\1 & u_{ij}\le 3km\end{cases}$，其中$i=1,2\cdots20$， $j=1,2\cdots92$
3. $x_{ij}=\begin{cases}1\text{，服务台}i\text{对要道}j\text{进行封锁}\\0\text{，服务台}i\text{不对要道}j\text{进行封锁}\end{cases}$，其中$i=1,2\cdots20$， $j=1,2\cdots92$；
4. . c_j： j 巡警服务台的工作量，其中 $j=1,2\cdots24$；

5. $\bar{C}$：平均工作量
6. In_i： i 路口的发案次数，其中$i=1,2\cdots92$；
7. ρ：工作量不均衡度；
8. $\begin{cases}\text{A类路口：只由一个服务台进行管辖；}\\\text{B类路口：可被多个服务台进行管辖；}\\\text{C类路口；不能被任何服务台进行管辖；}\end{cases}$
9. $I_{isolated}$：C 类路口的集合；
10. Q_{t+3}：嫌疑犯在$(t+3)\min$内行驶的最大区域；

11. ∂Q_{t+3}：嫌疑犯在$(t+3)\min$内行使的最大区域边界点集；

四、模型的建立与求解

4.1 问题一（1）：管辖区域的确定---最大集合覆盖模型

4.1.1 模型建立：

最大覆盖函数：根据问题一（1）的分析确定最大覆盖函数为：

$$f_1=\max_{\substack{1\le i\le 92\\1\le j\le 20}}\sum_{i\in I}\sum_{j\in J}s_{ij}\text{。}$$

满足条件：

1）因为每个路口只由一个巡警服务台管辖，所以$\sum_{j\in J}s_{ij}\le 1(i\in I)$；

2）根据实际情况每个巡警服务台管辖的路口数至少等于 1，所以$\sum_{i\in I}s_{ij}\ge 1(j\in J$）。

综上所述，得到最大集合覆盖模型为：

$$f_1=\max_{\substack{1\le i\le 92\\1\le j\le 20}}\sum_{i\in I}\sum_{j\in J}s_{ij}$$

满足：

$$\text{s.t.}\begin{cases}s_{ij}\le 1 & (c_{ij}\le 3km)\\ s_{ij}=0 & (c_{ij}>3km)\\ \sum_{j\in J}s_{ij}\le 1 & (i\in I)\\ \sum_{i\in I}s_{ij}\ge 1 & (j\in J)\\ s_{ij}=0\text{或}1\end{cases}\tag{1}$$

4.1.2 模型求解：

1. 最短路径矩阵$U_{92\times 92}$的建立

本文选用 Warshall 算法确定城区 A 任意两个路口之间的最短路径矩阵$U_{92\times 92}$。Warshall 算法为：任意两点(i,j)之间的最短路径（记为D_{ij}）等于从i出发到达j点的以任一点为中转点所有可能方案中，距离最短的一个。即：$D_{ij}=\min(D_{ij},D_{ik}+D_{kj},\cdots),1\le k\le n$，$n$为节点总数。

计算方法为：

$u_{ii}^{(1)}=0$，

$u_{ij}^{(1)}=w_{ij}\ i\ne j$，

$u_{ij}^{(k+1)}=\min\{u_{ij}^{(k)},u_{ik}^{(k)}+u_{kj}^{(k)}\},i,j,k=1,\cdots,n$

在上式中，临时标号$u_{ij}^{(k)}$是不通过$k,k+1,\cdots,n$节点（i,j除外）时从节点i到节点j的最短路径。

通过上述算法，利用数学软件 MATLAB 计算出各节点的最短路径，组成一个最短路径矩阵$U_{92\times 92}$。矩阵中，元素u_{i_1j}为从路口i_1到路口j的最短距离，其中

$i_1 = 1 \cdots 92$， $j = 1 \cdots 92$。

2. 集合覆盖矩阵 $K_{92\times 20}$ 的建立

以巡警服务台能否在 3min 内到达案发路口为分配标准，将上述最短路矩阵转化为集合覆盖问题，即 0-1 覆盖问题。转化方法为：

$$k_{ij} = \begin{cases} 0 & u_{ij} > 3km \\ 1 & u_{ij} \le 3km \end{cases} \tag{2}$$

此时将得到集合覆盖矩阵 $K_{92\times 20}$。此时从矩阵中可以得到巡警服务台在只考虑能在规定时间内到达的初始分配情况。

3. 最终分配方案的确定

从上述得到的集合覆盖矩阵中，我们可以清楚的看到如下问题：

1）在仅满足分配标准时有些路口可同时被多个巡警服务台管辖；

2）在仅满足分配标准时标号为 28、29、38、39、61、92 的路口无巡警服务台进行管辖；

那么此时并不满足模型的要求，必须对 $K_{92\times 20}$ 进行处理，以得到满足要求的最终分配方案。

首先解决 1）中出现的问题，此过程我们利用数学软件 MATLAB 进行处理，相应的程序见附录。

步骤一：

由集合覆盖矩阵 $K_{92\times 20}$ 将 92 个路口分为 A、B、C 三类：

A 类：已只由一个巡警服务台进行管辖；

B 类：可被多个巡警服务台进行管辖；

C 类；还不能被任何巡警服务台进行管辖；

步骤二：

将 A 类中的路口直接分配给对其进行管辖的唯一的巡警服务台。

步骤三：

将 B 类中的路口按最近原则分配给距离它最近的巡警服务台。

此时得到初始分配方案如下：

表 2　未考虑 C 类路口的初始分配方案

巡警服务台	管辖路口	巡警服务台	管辖路口
1	1,67,68,69,71,73,74,75,76,78	**11**	11,26,27
2	2,40,43,44,70,72	**12**	12,25
3	54,55,3,65,66	**13**	13,21,22,23,24
4	4,57, 60,62,63,64	**14**	14
5	49,53,5, 50,51,52,56,58,59	**15**	15
6	6	**16**	16,36,37
7	30,7,32,47,48	**17**	41,17,42
8	8,33,46	**18**	18,80,81,82,83
9	9,31,34,35,45	**19**	19,77, 79
10	10	**20**	86,20,84,85,87,88,89,90,91

然后解决 2）中的问题。将标号为 28、29、38、39、61、92 的 6 个任何巡警服务台都不能在规定时间内到达的路口按就近原则进行分配。

最后得到最终的分配方案如下：

表 3　最终分配方案

巡警服务台	管辖路口	巡警服务台	管辖路口
1	1,67,68,69,71,73,74,75,76,78	**11**	11,26,27
2	2,40,43,44,70,72,**[39]**	**12**	12,25
3	54,55,3,65,66	**13**	13,21,22,23,24
4	4,57, 60,62,63,64	**14**	14
5	49,53,5, 50,51,52,56,58,59	**15**	15,**[28]**,**[29]**
6	6	**16**	16,36,37,**[38]**
7	30,7,32,47,48,**[61]**	**17**	41,17,42
8	8,33,46	**18**	18,80,81,82,83
9	9,31,34,35,45	**19**	19,77, 79
10	10	**20**	86,20,84,85,87,88,89,90,91,**[92]**

4.2 问题一（2）：警力合理调度方案— 0-1 整数规划模型

4.2.1 模型建立：

记 20 个巡警服务台分别为$i=1\cdots20$，记 13 条交通要道分别为 $j=1\cdots13$。记巡警服务台 i 与要道 j 间的距离为 c_{ij}。

决策变量：引入 0-1 变量 x_{ij}，若选择巡警服务台 i 对要道 j 进行封锁，记 $x_{ij}=1$，否则记 $x_{ij}=0$，即：

$$x_{ij}=\begin{cases}1，服务台i对要道j进行封锁\\0，服务台i不对要道j进行封锁\end{cases} \tag{3}$$

此为问题的决策变量，共 260 个。

目标函数：本题要求对 13 条要道进行快速封锁，即要求巡警服务台对 13 条交通要道进行全部封锁所需时间最短的调度方案。在假设警车行驶速度相同的条件下，可转化为求巡警服务台与要道最大距离最短的调度方案。则本题目标函数为 $f_2=\max\left(c_{ij}x_{ij}\right)$，其中 $i=1\cdots20$，$j=1\cdots13$。

约束条件：根据问题的要求，每个交通要道必须有一个巡警服务台对其进行封锁，即对于 $j=1\cdots13$，应有：$\sum_{i=1}^{20}x_{ij}=1$，对于 $i=1\cdots20$，应有：$\sum_{j=1}^{13}x_{ij}\le1$。

综上所述，此问题的优化模型为：

$$\min\ f_2=\max_{\substack{1\le i\le20\\1\le j\le13}}\left(c_{ij}x_{ij}\right)$$

$$s.t.\ \begin{cases}\sum_{i=1}^{20}x_{ij}=1,\ \ j=1\cdots13\\\sum_{j=1}^{13}x_{ij}\le1,\ \ i=1\cdots20\\x_{ij}=0或1\end{cases} \tag{4}$$

4.2.2 模型的求解：

本文利用 MATLAB 和 Lingo 进行编程求解，程序见附录，具体步骤如下：

1.求解 c_{ij}。整理附件 2 中的数据，根据 Floyd-Warshall 算法，利用 MATLAB 编程，得到 20 个巡警服务台距离 13 条交通要道的最短距离 c_{ij}。

2.按照附件 2 中 20 个巡警服务台和 13 条交通要道的顺序进行编号，引入决策变量 x_{ij}，根据已经建立的模型中的约束条件和目标函数，利用 Lingo9.0 求得全局最优解。

3.求解结果显示，目标函数的最小值为 8.0155，即封锁 13 条交通要道的最少时间为 8.0155 分钟。下表列出了 A 区交巡警服务平台警力合理的调度方案。

表 4　A 区交巡警服务平台警力封锁 13 条交通要道的调度方案

出入A区的路口标号	12	14	16	21	22	23	24	28	29	30	38	48	62
交巡警平台位置标号	10	16	6	11	12	14	13	15	7	8	19	4	20
交巡警到达路口的时	7.5866	6.7417	6.259	5.0723	6.8825	6.4733	2.3854	4.7518	8.0155	3.0608	7.6393	7.3959	6.4489

4.3 问题一（3）确定增加平台的个数与位置——集合覆盖模型

4.3.1 模型建立与问题求解：

1.初步分配方案的确定

同样运用问题一（1）中的方法可得到：距离 C 类各个路口小于 3km 的路口集合，如下表：

表 5　距离 C 类各个路口小于 3km 的路口集合

C 类路口标号	**28**	**29**	**38**	**39**	**48**	**92**
集　　合	{28,29}	{28,29}	{38,39,40}	{38,39,40}	{48,61}	{87,88,89,90,91,92}

对上表中的 6 个集合求并，得到需要增加巡警服务台的路口的候选集 $Q=\{28,29,38,39,40,48,61,87,88,89,90,9\}$。本文将要在候选集 Q 中选择 2~5 个路口设置巡警服务台，使需求集 $I_{isolated}=[28,29,38,39,61,92]$ 中的所有路口在案发生时均有巡警在 3min 之内能赶到。

集合覆盖模型的建立

首先，建立覆盖矩阵 $T_{6\times13}$，其元素：

$$t_{ij}=\begin{cases}1, & \text{路口}i\text{在3min之内可到达}j\\ 0, & \text{路口}i\text{在3min之内不可到达}j\end{cases} \tag{5}$$

$i=1,2\cdots6$，　$j=1,2\cdots13$。

其次，建立集合覆盖模型：

$$f_3=\min_{j\in Q}\sum_{j\in Q}x_j$$

满足：

$$s.t.\begin{cases}\sum_{j\in Q}t_{ij}x_j\geq1 \ ,\forall i\in I_{isolated}\\ x_j=0\text{或}1\end{cases} \tag{6}$$

其中：$x_j=\begin{cases}1 & ,\text{路口}j\text{设置巡警服务台}\\ 0 & ,\text{路口}j\text{不设置巡警服务台}\end{cases}$

最后，利用 MATLAB 运用搜索法得到：至少从候选集 Q 中选出 4 个路口来设置巡警服务台，才能解决出警时间过长的问题。此时共有 48 种可能的分配方案。分别如下表所示：

表 6　48 种分配方案

28,38,48,87	**28,38,61,87**	**28,39,48,87**	**28,39,61,87**	**29,38,48,87**	**29,38,61,87**	**29,39,48,87**	**29,39,61,87**
28,38,48,88	28,38,61,88	28,39,48,88	28,39,61,88	29,38,48,88	29,38,61,88	29,39,48,88	29,39,61,88
28,38,48,89	28,38,61,89	28,39,48,89	28,39,61,89	29,38,48,89	29,38,61,89	29,39,48,89	29,39,61,89
28,38,48,90	28,38,61,90	28,39,48,90	28,39,61,90	29,38,48,90	29,38,61,90	29,39,48,90	29,39,61,90
28,38,48,91	28,38,61,91	28,39,48,91	28,39,61,91	29,38,48,91	29,38,61,91	29,39,48,91	29,39,61,91
28,38,48,92	28,38,61,92	28,39,48,92	28,39,61,92	29,38,48,92	29,38,61,92	29,39,48,92	29,39,61,92

2.最终分配方案的确定

1）为每种方案中的 24 个巡警服务台分配管辖范围。

步骤一：

同样按照问题一（1）中的求解过程 1 和 2 可得到有 24 个巡警服务台的集合覆盖矩阵 $K_{92\times24}$。

步骤二：

此时由上述集合覆盖矩阵可将城区 A 的 92 个路口分为 A、B 两类：

A 类：已只由一个巡警服务台进行管辖；

B 类：可被多个巡警服务台进行管辖；

将 A 类中的路口直接分配给对其进行管辖的唯一的巡警服务台。对于 B 类的路口，在综合距离最近与工作量平均的情况下来进行分配。即：首先选择距离路口 i 最近的巡警服务台 $j(j=1,2\cdots24)$，然后利用公式 $c_j=\sum_{i=1}^{92}s_{ij}in_i$ 计算巡警服务台 j 的工作量 c_j，若 $c_j\le\bar{C}$ 则将路口 i 分配给巡警服务台 j 管辖，否则选择次短距离的巡警服务台进行同样考虑。最后得到每种分配方案中 24 个巡警服务台的管辖范围。

步骤三：

根据平局工作量公式 $\bar{C}=\frac{\sum_{i=1}^{92}in_i}{24}$ 与工作量不均衡度公式 $\rho=\sum_{j=1}^{24}\left(c_j-\bar{C}\right)^2$，利用 MATLAB 分别对 48 中分配方案中巡警服务台的工作量不均衡度进行计算。得到下表：

表 7　48 中分配方案对应的工作量不均衡度

3.4890	**4.7194**	**3.0742**	**3.3046**	**3.4890**	**4.7194**	**3.0742**	**4.3046**
3.4890	4.7194	3.0742	4.3046	3.4890	4.7194	3.0742	4.3046
3.4890	4.7194	3.0742	4.3046	3.4890	4.7194	3.0742	4.3064
3.4890	4.6490	3.0742	4.2498	3.4890	4.6490	3.0742	4.2498
3.4890	4.7194	3.0742	4.3046	3.4890	4.7194	3.0742	4.3046
3.4916	4.6672	3.0768	4.2524	3.4916	4.6672	3.0768	4.2524

由表中数据可得：最小不均衡度为 3.0742，有 8 中分配方案。如下表所示：

表 8　满足题目一（3）要求的 4 个巡警服务台的路口标号

28,39,48,87	**28,39,48,88**	**28,39,48,89**	**28,39,48,90**	**28,39,48,91**
29,39,48,87	29,39,48,88	29,39,48,89	29,39,48,90	29,39,48,91

本文仅给出其中方案一（在路口标号为 28、39、48、87 处增加巡警服务台）对应的城区 A 的 24 个巡警服务台的管辖范围与每个巡警服务台对应的工作量，如下表：

表 9　方案一中 24 个巡警服务台的管辖范围

服务台	管辖范围	服务台	管辖范围
1	1,67,68,69,71	**13**	13,21,22,23,24
2	2,43,44,70,72	**14**	14
3	54,55,3,65,66	**15**	15
4	4,57,60,62,63	**16**	16,36,37
5	53,5,49,50,51	**17**	41,17,42
6	6,52,56,58,59	**18**	18,74,80,81,82
7	7,30,32	**19**	19,75,76,77,78
8	8,33,45,46	**20**	20,85,86,90,91
9	9,31,34,35	**28**	28,29
10	10	**39**	38,39,40
11	11,26,27	**48**	61,67,48
12	12,25	**87**	92,83,84,87,88

表 10　方案一中每个巡警服务台对应的平均工作量

服务台	1	2	3	4	5	6	7	8	9	10	11	12	平均工作量
工作量	6.5	6.6	5.6	6.6	6.6	5.6	6	6.4	6.8	1.6	4.6	4	**5.1875**
服务台	**13**	**14**	**15**	**16**	**17**	**18**	**19**	**20**	**28**	**39**	**48**	**87**	**不均衡度**
工作量	8.5	2.5	2.1	3.8	5.3	6.3	6.1	6.3	2.7	4.3	3.6	6.1	3.0742

4.4 问题二（1）

4.4.1 现有巡警服务台设置的合理性的讨论

本文定义了两个评价原则：

原则一：巡警能在 3min 之内到达案发路口

原则二：巡警服务台的工作量均衡度尽量小。

依据问题分析中的两个评价原则，分别对现有巡警服务台的设置方案进行评价。

讨论现有设置方案是否满足原则一

全城六区 A，B，C，D，E，F 现有个 80 巡警服务台、582 个路口，运用问题一（1）中的算法，得到全城 C 类路口的数目与位置，如下表：

表 11　C 类路口的位置标号

28	29	38	39	61	92	122	123	124	151
152	153	183	199	200	201	202	203	205	206
207	208	209	210	215	238	239	240	247	248
251	252	253	257	259	261	262	263	264	268
269	285	286	287	288	299	300	301	302	303
304	312	313	314	315	316	317	318	319	329
330	331	332	336	337	339	344	362	369	370
371	387	388	389	390	391	392	393	395	407
408	409	411	412	413	417	418	419	420	438
439	443	445	446	451	452	455	458	459	464
469	471	474	486	487	505	506	507	508	509
510	512	513	514	515	516	517	518	519	522
523	524	525	526	527	529	533	540	541	559
560	561	566	569	574	575	578	582		

计算结果表明：582 个路口中共有 138 个 C 类路口，即在案发时巡警不能在 3min 到达此路口，约占全城总路口数的 1/4。

讨论现有设置方案是否满足原则二

运用问题一（3）中的方法，为每个巡警服务台分配管辖范围，并计算工作量及巡警服务台的工作不均衡度。结果如下：

表 12　现有配置下每个巡警服务台的工作量

巡警服务台	工作量	巡警服务台	工作量	巡警服务台	工作量	巡警服务台	工作量
1	10.3	93	2.1	178	4.5	378	2.6
2	9.7	94	11.3	179	13	379	7.4
3	5.6	95	9.5	180	13	380	2.5
4	17.1	96	11.5	181	6.2	381	6.2
5	9.7	97	5.6	182	12.2	382	10.3
6	2.5	98	12.1	320	8.7	383	10
7	40.4	99	4.3	321	12	384	8.3
8	5	100	4.5	322	4.4	385	9.1
9	8.2	166	3.8	323	4.2	386	6.3
10	1.6	167	8.3	324	7.9	475	13.1
11	4.6	168	4.7	325	2.2	476	13
12	10.3	169	3.4	326	5.1	477	10.7
13	39.6	170	12.9	327	7.6	478	9.5
14	7.2	171	12.4	328	6.7	479	8.7
15	12.9	172	8.3	372	5.2	480	4.7
16	28.4	173	11.5	373	4.1	481	7.2
17	5.3	174	10.1	374	5.5	482	4.4
18	6.1	175	8.7	375	6.1	483	3.3
19	3.4	176	8.1	376	2.6	484	3.8
20	11.5	177	2.2	377	4.2	485	3.3

结果分析：

由表中数据可得：在现有配置下，其中有个 7 巡警服务台的工作量已达到 40.4，而其中还有 10 巡警服务平台的工作量仅为 1.6，所有巡警服务台的工作量

不均衡度为 40.3。可见，此时巡警服务平台的工作量极其不均衡。

根据上述 1、2 的讨论可知：现有巡警服务台的设置极其不合理。

4.4.2 巡警服务台设置的优化

由现有巡警服务台的分配不合理的情况，我们提出三种优化方案。

方案一——“静态增加”优化方案

此方案的优化思想为：在不改变现有巡警服务台的位置的情况下，适当增加巡警服务台的数目，从而使城区 A 中无 C 类路口且每个巡警服务台的工作量尽量均衡。

由上题 4.4.1 中的计算结果可知，全城共有 138 个 C 类路口。将其组成需求点的集合，同样利用求解问题一（3）中的方法与步骤，得到新增加的最小巡警服务台数目与位置、此种方案下巡警服务台的工作量和工作量的不均衡度。

表 13　新增加的 54 个巡警服务台的路口标号

新增加的平台的节点位置					
486	575	421	344	239	149
505	578	442	362	248	38
509	582	454	371	252	61
522	387	458	184	253	89
539	389	472	201	263	
541	393	330	204	268	
549	403	332	208	288	
560	407	333	210	313	
567	419	338	216	315	
574	420	340	237	104	

表 14　新增加 54 个巡警服务台后每个巡警服务台的工作量

巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小
1	10.3	**98**	12.1	**325**	2.2	**481**	5.2	**421**	3.3	**268**	5.8
2	8.3	**99**	2.6	**326**	5.1	**482**	2.9	**442**	3.5	**288**	8.2
3	5.6	**100**	3.8	**327**	3.7	**483**	3.3	**454**	11.1	**313**	7.7
4	6.6	**166**	3.8	**328**	2.6	**484**	2.4	**458**	3.3	**315**	7.3
5	9.7	**167**	8.3	**372**	5.2	**485**	2.1	**472**	5.8	**104**	7.6
6	2.5	**168**	4.7	**373**	4.1	**486**	2.5	**330**	2.7	**149**	3.5
7	9	**169**	3.4	**374**	4.4	**505**	3.7	**332**	0.2	**38**	4
8	5	**170**	12.2	**375**	6.1	**509**	3.1	**333**	6	**61**	0.6
9	8.2	**171**	11	**376**	2.6	**522**	8.7	**338**	3.2	**89**	7
10	1.6	**172**	8.3	**377**	4.2	**539**	6.2	**340**	3		
11	4.6	**173**	7.5	**378**	2.6	**541**	0.1	**344**	1.1		
12	4	**174**	10.1	**379**	5.7	**549**	6.3	**362**	0.1		
13	5.7	**175**	8.7	**380**	1.8	**560**	2.7	**371**	2.7		
14	2.5	**176**	6.7	**381**	6.2	**567**	5.3	**184**	2.6		
15	2.1	**177**	2.2	**382**	5.8	**574**	0.6	**201**	2.1		
16	3.8	**178**	1.7	**383**	6.2	**575**	0.6	**204**	2.2		
17	5.3	**179**	11.9	**384**	5	**578**	1.8	**208**	4.1		
18	6.1	**180**	10.7	**385**	8	**582**	0.4	**210**	1.5		
19	3.4	**181**	4	**386**	4.7	**387**	1.1	**216**	2.6		
20	4.5	**182**	10.8	**475**	12	**389**	1	**237**	5.1		
93	2.1	**320**	8.7	**476**	11.8	**393**	3.6	**239**	5.1		
94	7.6	**321**	12	**477**	8.4	**403**	8.3	**248**	3.2		
95	8.4	**322**	4.4	**478**	6.3	**407**	3.3	**252**	2.4		
96	11.1	**323**	4.2	**479**	7.5	**419**	2.5	**253**	1.9		
97	5.6	**324**	7.9	**480**	3.9	**420**	1.6	**263**	3.3		

结果分析：

由上表可知：标号为 170 的巡警服务台工作量最大，为 12.2，541 巡警服务台工作量最小，为 0.1，此方案不均衡度降为 9.4078。

方案二——“动态”优化方案

此方案的优化思想为：在不改变现有巡警服务台数目的情况下，对 80 个巡警服务台的位置进行重新定位，从而使城区 A 中的 C 类路口数目尽量少且巡警服务台的工作量尽量均衡。

模型的建立

根据此方案的优化思想，首先确定候选集为：$J=\{1,2,\cdots 582\}$，需求点集为：$I=\{1,2,\cdots 582\}$。然后，建立最大集合覆盖模型，从候选集中选出 80 个点以满足使城区 A 中的 C 类路口数目尽量少且巡警服务台工作量尽量均衡。模型如下：

$$f_4=\max_{\substack{i\in I\\ j\in J}}\sum_{i\in I}\sum_{j\in J}k_{ij}x_j$$

满足：

$$s.t.\begin{cases}\sum_{j\in J}x_j=80\\ x_j=0\text{或}1\end{cases}\tag{7}$$

其中：$x_j=\begin{cases}1 & ,\text{路口}j\text{设置巡警服务台}\\ 0 & ,\text{路口}j\text{不设置巡警服务台}\end{cases}$，其中 $j\in J$ （8）

$k_{ij}=\begin{cases}1 & ，\text{路口}i\text{在}3\min\text{内可达到路口}j\\ 0 & ,\text{路口}i\text{在}3\min\text{内不可达到路口}j\end{cases}$，其中 $i\in I$，$j\in J$ （9）

模型求解：

利用 LINGO 对上述模型进行求解，结果如下：

表　80 个巡警服务台的工作量

新设置的平台位置	每个平台工作量	新设置的平台位置	每个平台工作量	新设置的平台位置	每个平台工作量	新设置的平台位置	每个平台工作量
3	8.6	185	8.1	330	2.7	440	3.5
21	6.3	190	4.7	333	8.6	448	9.4
24	8.7	201	2.1	338	8.6	453	16.1
27	7.8	204	4.7	352	24.6	472	9.2
31	8.3	208	4.1	362	0.1	474	3
42	8.1	210	1.5	363	10.1	485	2.1
46	12.3	216	18.1	369	3.4	486	6.2
48	13.9	237	13.8	373	4.1	488	4.4
57	12.2	239	5.1	374	11.6	499	8.6
70	18.6	240	4.5	381	5.6	509	4.6
90	17.6	250	3.9	382	10.8	511	3.3
98	19.1	254	3.4	386	5.5	515	7.7
115	14.3	261	6.6	389	1	532	15.9
127	15.9	263	1.4	393	3.6	539	11.1
135	22.7	271	16.6	397	1.6	550	5.3
150	13.4	276	17.6	407	5	561	6.6
167	4.5	289	12.9	416	7.9	565	17.2
174	12.1	305	9.1	419	2.5	570	7.4
175	8.7	314	5.4	421	3.3	573	4.4
181	9.5	315	7.3	423	8.3	581	6.1

结果分析：在以上 80 个路口设置巡警服务平台，可以使全城 C 类路口为 20 个，远小于原来的 138 个，其工作量不均衡度降到 29.3736，相对于原来的 40.4 下降了 27.3%。

（3）方案三——“动态增加”优化方案

此方案的优化思想为，不考虑城区 A 现有的巡警服务台设置，重新在 582 个路口中确定最优巡警服务台数目与位置，从而使城区 A 中无 C 类路口且每个巡警服务台的工作量尽量均衡。

根据此方案的优化思想，首先确定候选集为：$J=\{1,2,\cdots 582\}$，需求点集为：$I=\{1,2,\cdots 582\}$，然后建立与问题一（3）中同样的集合覆盖模型，最后利用 LINGO 与 MATLAB 求解，得到此优化方案下的最优巡警服务台的数目与位置、每个巡警服务台的工作量，如下表所示：

表 15　方案三得到的 96 个巡警服务台的路口标号与工作量

巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小
3	8.6	174	12.1	289	11	386	5.5	509	4.6
10	1.6	175	8.7	305	9.1	387	1.1	511	3.3
14	2.5	177	2.2	314	5.4	389	1	515	7.7
21	6.3	181	9.5	315	7.3	393	3.6	532	15.9
24	7.1	185	8.1	330	2.7	397	1.6	539	11.1
25	4	190	4.7	332	0.2	407	5	541	0.1
27	4.6	201	2.1	333	8.6	416	7.9	550	5.3
31	8.3	204	4.7	338	8.6	419	2.5	561	5.1
42	5.3	208	4.1	344	1.1	420	1.6	562	3
46	12.3	210	1.5	352	22.9	421	3.3	565	17.2
48	13.9	216	18.1	361	8.4	423	8.3	570	7.4
57	12.2	237	13.8	362	0.1	440	3.5	573	4.4
70	19.8	239	5.1	363	10.1	448	9.4	574	0.6
90	17.6	240	4.5	369	3.4	453	16.1	575	0.6
98	11.4	250	3.9	371	2.7	472	9.2	581	6.1
115	11.4	254	3.4	373	4.1	474	3	582	0.4
127	10.8	261	6.6	374	11.6	485	2.1		
135	18.6	263	1.4	378	2.6	486	6.2		
150	10.1	271	16.6	381	5.6	488	4.4		
167	4.5	276	19.5	382	10.8	499	8.6		

分析结果：在以上 96 个路口设置巡警服务台，就可以使全城无 C 类路口，但是其不均衡度为 27.0268，工作量并不均衡。于是对结果做进一步优化，在确定优化方案前，定义相关概念如下：

节点增加条件：当某巡警节点的实际工作量大于 c 倍的平均工作量时（本文令 $c=1.8$），该巡警节点满足节点增加条件。当工作量大于 c 倍的平均工作量小于 1+c 倍平均工作量，增加一个节点；当工作量大于 1+c 倍工作量小于 2+c 倍工作量，增加 2 个节点.....

巡警节点优化算法如下：

第一步：在所有巡警节点中寻找工作量最大的那个巡警节点；

第二步：判断最大节点是否满足节点增加条件，如果满足条件，进入第三步，否则，终止算法，现有的节点分配即为最优解；

第三步：更具节点增加条件，计算增加的节点数 i，并在该最大巡警服务台管辖范围内搜索 i 个节点最为新增的巡警平台位置，更新巡警节点集，进入第四步；

第四步：计算更新的巡警节点集的最优分配，得出所有巡警节点新的工作量

分配，转入第一步；

利用 MATLAB 编程求解，总公迭代 9 次，得到最优解，其每次迭代的结果如下：

表 16　迭代结果

迭代次数	增加的节点	不均衡度	迭代次数	增加的节点	不均衡度
1	350,351	22.9431	6	462	14.0915
2	227,231	21.6037	7	85	13.9
3	225,283	19.5819	8	6	13.7
4	75,78	16.2185	9	142	12.3798
5	268,293	14.9694	10	544	11.7574

所以最终的节点个数为 96+15=111 个节点，其每个巡警节点的工作量为：

表 17　111 个巡警服务台的工作量

巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小	巡警节点	工作量大小
3	5.9	174	10.1	289	9.6	386	5.5	509	4.6	225	9.1
10	1.6	175	8.7	305	9.1	387	1.1	511	3.3	283	3.7
14	2.5	177	2.2	314	5.4	389	1	515	7.7	75	7.4
21	6.3	181	8.1	315	7.3	393	3.6	532	11.4	78	9.8
24	7.1	185	8.1	330	2.7	397	1.6	539	7	268	3.5
25	4	190	4.7	332	0.2	407	5	541	0.1	293	10.2
27	4.6	201	2.1	333	8.6	416	7.9	550	5.3	462	7.5
31	8.3	204	4.7	338	8.6	419	2.5	561	5.1	85	4.5
42	5.3	208	4.1	344	1.1	420	1.6	562	3	6	2.5
46	12.3	210	1.5	352	6.3	421	3.3	565	12.4	142	12
48	11.4	216	4	361	6.7	423	8.3	570	7.4	544	12
57	10.8	237	10.1	362	0.1	440	3.5	573	4.4		
70	9.4	239	5.1	363	10.1	448	9.4	574	0.6		
90	10.4	240	3.3	369	3.4	453	8.6	575	0.6		
98	11.4	250	3.9	371	2.7	472	9.2	581	6.1		
115	11.4	254	3.4	373	4.1	474	3	582	0.4		
127	10.8	261	6.6	374	11.6	485	2.1	350	11.5		
135	9.6	263	1.4	378	2.6	486	6.2	351	6.8		
150	7.1	271	6.5	381	5.6	488	4.4	227	6.9		
167	4.5	276	7.9	382	10.8	499	8.6	231	12.1		

4.5 问题二（2）：围堵方案的确定

4.5.1 模型建立

按照问题分析，本文定义相关概念如下：

Q_{t+3}：嫌疑犯在$(t+3)\min$内行驶的最大区域；

∂Q_{t+3}：嫌疑犯在$(t+3)\min$内行使的最大区域边界点集；

P：　现有所有巡警服务台的集合

$$T<\partial Q_{t+3},P>=\begin{cases}1 & \text{可以适当分配P中的警力，在t时间内到达}\partial Q_{t+3}\text{中的所有路口点}\\ 0 & \text{否则}\end{cases}$$

本文以时间t为目标，建立优化模型如下：

$$\min\ t$$

$$s.t.\ T<\partial Q_{t+3},P>=1$$

模型的限制条件中，对于是否可以适当分配 P 中的警力，可以在 t 时间内到

达∂Q_{t+3}中所有点，可以抽象为图论中二部图的完全匹配问题，如果可以分配警力使其和∂Q_{t+3}中的边界点一一匹配，则$T<\partial Q_{t+3}, P>=1$。：

4.5.2 模型求解

初始化$t=0$；

第一步：令$t=t+1$；

第二步：计算嫌疑犯在$t+3$时间内的覆盖区域Q_{t+3}；

第三步：计算Q_{t+3}的边界点集∂Q_{t+3}；

第四步：计算得到∂Q_{t+3}与P之间的关系矩阵；

第四步：将分配P中的警力，在t时间内到达∂Q_{t+3}中的所有路口点的问题，抽象为图论中二部图的完全匹配问题。运用匈牙利算法，得到P到∂Q_{t+3}的最大匹配。如果是完全匹配，则计算终止，最佳围堵时间为t，最佳围堵方案即为完全匹配的结果；否则转到第一步继续计算；

（1）没有优化前的 80 个巡警位置，本文按照算法计算最佳围堵方案和时间如下：

表 18　在巡警服务台原有配置下的围堵方案

边界点	被分配的巡警节点	巡警的行走路径	巡警的围堵时间
40	1	$1\rightarrow 69\rightarrow 70\rightarrow 2\rightarrow 40$	3.8132
44	2	$2\rightarrow 44$	0.9487
60	4	$4\rightarrow 62\rightarrow 60$	1.7392
171	170	$170\rightarrow 227\rightarrow 228\rightarrow 171$	4.1911
230	171	$171\rightarrow 230$	0.8944
242	172	$172\rightarrow 227\rightarrow 228\rightarrow 229\rightarrow 230\rightarrow 243\rightarrow 242$	3.8247
243	173	$173\rightarrow 232\rightarrow 231\rightarrow 244\rightarrow 243$	3.6906

结果分析：由表中数据可知，巡警接到报警后只需 4.1911min 就可以完全围堵嫌疑犯。

（2）在优化方案一的配置下，有巡警服务台 134 个，相应的最佳围堵方案和最少围堵时间如下：

表 20　在方案三的配置下对应的围堵方案表

边界点	被分配的巡警节点	巡警的行走路径	巡警的围堵时间
3	1	$3\rightarrow 69\rightarrow 68\rightarrow 67\rightarrow 66\rightarrow 65\rightarrow 3$	3.8839
10	10	$10\rightarrow 10$	0
38	2	$2\rightarrow 40\rightarrow 39\rightarrow 38$	3.9822
39	16	$16\rightarrow 38\rightarrow 39$	3.7059
54	3	$3\rightarrow 55\rightarrow 54$	2.2709
57	4	$4\rightarrow 57$	1.8682
58	5	$5\rightarrow 50\rightarrow 51\rightarrow 59\rightarrow 58$	2.3019
231	171	$171\rightarrow 231$	0.7211
238	173	$173\rightarrow 236\rightarrow 237\rightarrow 238$	3.9609
244	172	$172\rightarrow 227\rightarrow 228\rightarrow 171\rightarrow 231\rightarrow 244$	3.5289
246	216	$216\rightarrow 171\rightarrow 230\rightarrow 243\rightarrow 242\rightarrow 246$	3.9096
560	560	$560\rightarrow 560$	0

结果分析：由表中数据可知，巡警在接到报警后只需 3.9822min 就可以完全围堵嫌疑犯。

（3）在优化方案二的配置下，其巡警节点为 80 个，相应的最佳围堵方案和最少围堵时间如下：

表 21　在方案二的配置下对应的围堵方案表

边界点	被分配的巡警节点	巡警的行走路径	巡警的围堵时间
73	3	$3\to44\to2\to43\to72\to73$	4.5241
79	42	$42\to43\to70\to69\to1\to78\to79$	3.9175
80	57	$57\to4\to63\to64\to76\to77\to19\to79\to80$	7.4463
170	174	$174\to224\to223\to225\to170$	4.0934
189	70	$70\to69\to68\to67\to66\to65\to$ $64\to63\to4\to62\to190\to189$	8.0172
190	175	$175\to194\to193\to192\to189\to190$	4.8251
215	48	$48\to235\to173\to232\to231\to171\to216\to215$	8.8238
219	210	$210\to211\to212\to213\to174\to219$	8.0959
224	204	$173\to236\to237\to238$	3.9609
226	216	$204\to178\to222\to223\to224$	7.3676
253	237	$237\to236\to245\to246\to241\to240\to253$	7.4846
482	486	$486\to487\to489\to482$	4.7434
532	488	$488\to560\to549\to548\to532$	7.4322
548	499	$499\to498\to477\to501\to530\to531\to532\to548$	5.5625
550	509	$509\to506\to505\to518\to521\to$ $529\to533\to532\to548\to549\to550$	8.0708
558	532	$532\to548\to549\to550\to559\to558$	5.2028
562	485	$485\to571\to569\to567\to480\to562$	8.6921

结果分析：由表中数据可知，巡警在接到报警后只需 8.8238min 就可以完全围堵嫌疑犯。

（4）在优化方案三的配置中，有巡警服务台 111 个，相应的最佳围堵方案和最少围堵时间如下：

表 19　在方案三的配置下对应的围堵方案表

边界点	被分配的巡警节点	巡警的行走路径	巡警的围堵时间
40	3	$3\to44\to2\to2\to40$	4.0261
44	42	$42\to43\to2\to44$	2.5549
60	6	$6\to59\to58\to57\to60$	3.9481
171	216	$216\to171$	0.9605
230	227	$227\to228\to229\to230$	2.2035
242	231	$231\to244\to243\to242$	2.3049
243	237	$237\to236\to245\to244\to243$	3.9153

结果分析：由表中数据可知，巡警接到报警后只需 4.0261min 就可以完全围堵嫌疑犯。

五、模型的评价与推广

1.优点：

采用离散定位模型作为城区巡警服务台优化布局方法的应用基础，结合相关的影响因素（发案率）,能很好地解决实际问题。

本文把实际问题抽象成集合模型、规划模型和图论模型，完整准确的描述了实际问题。

本文所用算法，效率好，精度高，解决实际问题方便快捷。

2.缺点：

本文对工作量的定义只考虑路口的发案率，没有考虑不同区的人口密度对交巡警工作量的影响。

本文对所有区都是以 3 分钟赶到作为标准，较少考虑不同区的路口的发案率相差加大，导致交巡警工作量难以均衡。

3.模型推广：

本模型不仅对巡警服务平台适用，而且可以广泛运用于消防站、医院等应急服务设施的布局。

六、参考文献

[1] 谢金星，优化模型与 LINDO/LINGO 软件，北京：清华大学出版社，2006 年。

[2] 王沫然，MATLAB 与科学，北京：电子工业出版社，2008 年。

[3] 方世昌，离散数学，西安：西安电子科技大学出版社，2009 年

[4] 吴美文，基于离散定位模型的城市消防站优化布局方法*，系统仿真技术，2006 年 1 月第 2 卷 第一期：58-62 页。

[5]陈驰 任爱珠，消防站布局优化的计算机方法[J]，清华大学学报：自然科学版，2003,43（10）：1390~1393。

[6]陈艳艳 郭国旗，城市消防站的优化布局[J].消防科技，1999，（1）：26~28

[7]吴军，消防站优化布局方法与技术研究，消防科学与技术 2006 年 1 月第 25 卷第 1 期

七、 附录

%问题一（1）：管辖区域的确定——最大集合覆盖模型的求解程序

```
function Mcm1.1
disp(sprintf('正在载入相关数据...'));
Node_data=xlsread('cumcm2011B.xls',1,'b2:c93');       %载入A区路口节点的左边数据
Routine_data=xlsread('cumcm2011B.xls',2,'a2:b144');   %载入路线节点标号数据
Record_data = cell(92,1);                             %创建包体，用来保存92个节点，每点的最大
覆盖区域
count = 0;
%更急路线节点标号数据创建邻接矩阵
for i = 1 :92
    Node_index = Routine_data(find(Routine_data(:,1)==i),2)
    Node_index = [Routine_data(find(Routine_data(:,2)==i),1);Node_index];
    Node_index = Node_index(find(Node_index <=92));
    n = length( Node_index);
    count = count + n;
    Record_data{i} = zeros(n,2);
    for j = 1 : n
        Record_data{i}(j,1) = Node_index(j);
        Record_data{i}(j,2) = 100*sqrt((Node_data(i,1) -
Node_data(Node_index(j),1))^2+(Node_data(i,2) - Node_data(Node_index(j),2))^2);
    end
end

Adjoin_matrix = zeros(count,3);      % 邻接矩阵
index_adj = 1;
for i = 1 :92
    [n1,n2] = size(Record_data{i});
    n = n1;
    for j = 1 : n
        Adjoin_matrix(index_adj,:) = [i,Record_data{i}(j,1),Record_data{i}(j,2)];
        index_adj = index_adj + 1;
    end
end

%根据邻接矩阵数据创建图论的稀疏矩阵
a1=Adjoin_matrix(:,1)';
a2=Adjoin_matrix(:,2)';
a3=Adjoin_matrix(:,3)';
DG=sparse(a1,a2,a3)%建立稀疏矩阵，图论求解

Patrol_range=cell(20,1);
for i=1:20
    dist=graphshortestpath(DG,i);%求图中任意两个节点之间的最短距离
    for j=1:92
        if(dist(j)<=3000)
            Patrol_range{i}=[Patrol_range{i},j];
        end
    end
end
Patrol_distribution=Patrol_range;    %复制原始数据
Patrol_cover=cell(92,1);             %定义交集
Cover=[];
Isolated=[];                         %定义孤立点
for i=1:92
```

```
    c=[];
    for j=1:20
        m=length(Patrol_range{j});
        for l=1:m
            if(Patrol_range{j}(l)==i)
                c=[c,j];            %保存i节点所对应的所有可能的交通巡警点
            end
        end
    end
    m=length(c);
    if(m>1)                         %如果大于1，说明有交集，先去除，不分配
        Cover=[Cover,i];
        Patrol_cover{i}=c;          %保存交集
        for k=1:m
            find(Patrol_distribution{c(k)}~=i);

Patrol_distribution{c(k)}=Patrol_distribution{c(k)}(find(Patrol_distribution{c(k)}~=i));%预
分配只属于自己的交通节点
        end
    end
    if(m==0)
        Isolated=[Isolated,i];
    end
end

Patrol_xin=Patrol_distribution;                                    %进行B类节点的的分
配
for i=1:92
    m=length(Patrol_cover{i});
    Distance_linshi=[];
    if(m>=1)
        dist=graphshortestpath(DG,i);
        for j=1:m
            Distance_linshi(j)=dist(Patrol_cover{i}(j));
        end
          Patrol_xin{Patrol_cover{i}(A(j,2))}=[Patrol_xin{Patrol_cover{i}(A(j,2))},i];
    end
end

m=length(Isolated);                                                %对孤立点进行分配
for i=1:m
    dist=graphshortestpath(DG,Isolated(i));
    D=dist(1:20);
    [m0,m1]=min(D);
    Patrol_xin{m1}=[Patrol_xin{m1},Isolated(i)];
end
save Patrol_xin.mat;
```

问题一（2）：求解围堵13条要道的方案

```
model:
!求解围堵13条要道的方案;
sets:
AA/1..20/;
cross/1..13/;
links(AA,cross): dis, x;
Endsets
!数据的定义部分;
data:
```

```
dis = @FILE(C:\Users\Administrator\Desktop\2011math\data.txt);
enddata
!目标函数;
min=@max(links(i,j):x(i,j)*dis(i,j));
!需求约束;
@for(cross(j):@sum(AA(i): x(i,j))=1);
@for(AA(i):@sum(cross(j): x(i,j))<=1);
!整数约束;
@for(links(i,j):@bin(x(i,j)));
```

%问题一（3）：管辖区域的确定——集合覆盖模型并使工作量最均衡的求解程序

```
function Mcm1.3
Node_data=xlsread('cumcm2011B.xls',1,'b2:c93');          %载入A区路口节点的左边数据
Routine_data=xlsread('cumcm2011B.xls',2,'a2:b144');      %载入路线节点标号数据
load B.mat;
Record_data = cell(92,1);                                %创建包体，用来保存92个节点，每点的最
大覆盖
count = 0;
for i = 1 :92
    Node_index = Routine_data(find(Routine_data(:,1)==i),2);
    Node_index = [Routine_data(find(Routine_data(:,2)==i),1);Node_index];
    Node_index = Node_index(find(Node_index <=92));
    n = length( Node_index);
    count = count + n;
    Record_data{i} = zeros(n,2);
    for j = 1 : n
        Record_data{i}(j,1) = Node_index(j);
        Record_data{i}(j,2) = 100*sqrt((Node_data(i,1) -
Node_data(Node_index(j),1))^2+(Node_data(i,2) - Node_data(Node_index(j),2))^2);
    end
end

Adjoin_matrix = zeros(count,3);                          % 邻接矩阵
index_adj = 1;
for i = 1 :92
    [n1,n2] = size(Record_data{i});
    n = n1;
    for j = 1 : n
        Adjoin_matrix(index_adj,:) = [i,Record_data{i}(j,1),Record_data{i}(j,2)];
        index_adj = index_adj + 1;
    end
end
%创建图论的稀疏矩阵及其图论的求解
a1=Adjoin_matrix(:,1)';
a2=Adjoin_matrix(:,2)';
a3=Adjoin_matrix(:,3)';
DG=sparse(a1,a2,a3);                                     %建立稀疏矩阵，图论求解

Patrol_range=cell(24,1);

D_24=B(13,:);                                            %B为可能的分配情况，共有48中，每次从中选
取1中可能，本次选取的事第13中可能性

for i=1:24
    dist=graphshortestpath(DG,D_24(i));                  %求图中任意两个节点之间的最短距离
    for j=1:92
        if(dist(j)<=3000)
```

```
            Patrol_range{i}=[Patrol_range{i},j];
        end
    end
end
save Patrol_range;

%求解交集和预分配问题
load Patrol_range.mat;                  %载入数据
Patrol_distribution=Patrol_range;       %复制原始数据
Patrol_cover=cell(92,1);                %定义交集
Cover=[];
Isolated=[];                            %定义孤立点
for i=1:92
    c=[];
    c2=[];
    for j=1:24
        m=length(Patrol_range{j});
        for l=1:m
            if(Patrol_range{j}(l)==i)
                c=[c,j];                %保存i节点所对应的所有可能的交通巡警点
                c2=[c2,D_24(j)];
            end
        end
    end
    m=length(c);
    if(m>1)                             %如果大于1，说明有交集，先去除，不分配
        Cover=[Cover,i];
        Patrol_cover{i}=c2;             %保存交集
        for k=1:m
            find(Patrol_distribution{c(k)}~=i);

Patrol_distribution{c(k)}=Patrol_distribution{c(k)}(find(Patrol_distribution{c(k)}~=i));%预
分配只属于自己的交通节点
        end
    end
    if(m==0)
        Isolated=[Isolated,i];
    end
end
save Patrol_distribution.mat;           %完成预分配，对于交集和孤立交点另外考虑
save Patrol_cover.mat;                  %保存交集所对应的可能交通巡警点

load Patrol_cover.mat;
load Patrol_distribution.mat;
load Patrol_range.mat;
%初始化预分配中每个交通巡警点的发案次数
Occurrence=xlsread('cumcm2011B.xls',1,'e2:e93');              %A区每个交通节点的发案次数
Standard_occurrence=sum(Occurrence)/24
Patrol_occurrence=zeros(24,1);
for i=1:24
    m=length(Patrol_distribution{i});
    a=0;
    if(m>=1)
        for j=1:m
            a=a+Occurrence(Patrol_distribution{i}(j));
        end
        Patrol_occurrence(i)=a;
    end
end
```

```
Patrol_xin=Patrol_distribution;                                    %进行交集的分配
for i=1:92
    m=length(Patrol_cover{i});
    Distance_linshi=[];
    if(m>=1)
        dist=graphshortestpath(DG,i);
        for j=1:m
            Distance_linshi(j)=dist(Patrol_cover{i}(j));
        end
        A=Sort_vector(Distance_linshi);                            %记录最小值的相对位置
        h=length(Distance_linshi);
        for j=1:h
            linshi_canshu=find(D_24==Patrol_cover{i}(A(j,2)));
            Patrol_occurrence(linshi_canshu);
            Patrol_cover{i}(A(j,2));
            if(Patrol_occurrence(linshi_canshu)<=(Standard_occurrence+0.62))
                Patrol_xin{linshi_canshu}=[Patrol_xin{linshi_canshu},i];

Patrol_occurrence(linshi_canshu)=Patrol_occurrence(linshi_canshu)+Occurrence(i);
                break;
            end
            if(j==h)
                i
            end
        end
    end
end

Patrol_occurrence
c=var(Patrol_occurrence);
save Patrol_xin.mat;
%初始化所有可能情况，构建矩阵，共后来计算使用
function chuli
a=[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20];
I=[28 29];
J=[38 39];
K=[48,61];
L=[87 88 89 90 91 92];
m=0
for i=1:2
    for j=1:2
        for k=1:2
            for l=1:6
                m=m+1;
                B(m,:)=[a,[I(i) J(j) K(k) L(l)]];
                DD(m,:)=[I(i) J(j) K(k) L(l)];
            end
        end
    end
end
save  B;
function A=Sort_vector(X)    %创建子函数，供调用使用
a=length(X);
for i=1:a
    [m0,weizhi]=min(X);
    A(i,1)=m0;
    A(i,2)=weizhi;
    X(weizhi)=inf;
```

```
end
```

%问题二（1.1）计算现有节点工作量，不均衡度和C类节点个数，以判断合理性

```
function Mcm2.1
Node_data=xlsread('cumcm2011B.xls',1,'b2:c583');
Routine_data=xlsread('cumcm2011B.xls',2,'a2:b929');
load dongzenjia.mat;
Record_data = cell(582,1);
count = 0;
for i = 1 :582
     Node_index = Routine_data(find(Routine_data(:,1)==i),2);
     Node_index = [Routine_data(find(Routine_data(:,2)==i),1);Node_index];
     n = length( Node_index);
     count = count + n;
     Record_data{i} = zeros(n,2);
     for j = 1 : n
         Record_data{i}(j,1) = Node_index(j);
         Record_data{i}(j,2) = 100*sqrt((Node_data(i,1) -
Node_data(Node_index(j),1))^2+(Node_data(i,2) - Node_data(Node_index(j),2))^2);
     end
end
Adjoin_matrix = zeros(count,3);      % 邻接矩阵
index_adj = 1;
for i = 1 :582
    [n1,n2] = size(Record_data{i});
    n = n1;
    for j = 1 : n
        Adjoin_matrix(index_adj,:) = [i,Record_data{i}(j,1),Record_data{i}(j,2)];
        index_adj = index_adj + 1;
    end
end

%创建图论的稀疏矩阵及其图论的求解
a1=Adjoin_matrix(:,1)';
a2=Adjoin_matrix(:,2)';
a3=Adjoin_matrix(:,3)';
DG=sparse(a1,a2,a3);%建立稀疏矩阵，图论求解
load budongzj.mat;
% Weizhi_all=xlsread('cumcm2011B.xls',3,'b2:b81');
% e=budongzj'
% Weizhi_all=[Weizhi_all;e]
%  Weizhi_all=[weizhi]
% Weizhi_all=[Weizhi_all;[350;351;227;231;225;283;75;78;268;293;462;85;6;142;544]];
Weizhi_all=xlsread('cumcm2011B.xls',7,'a1:a80');
a=length(Weizhi_all);
Patrol_range=cell(a,1);
for i=1:a
    dist=graphshortestpath(DG,Weizhi_all(i));%求图中任意两个节点之间的最短距离
    for j=1:582
        if(dist(j)<=3000)
            Patrol_range{i}=[Patrol_range{i},j];
        end
    end
end
save Patrol_range;
load Patrol_range.mat;              %载入数据
Patrol_distribution=Patrol_range;   %复制原始数据
Patrol_cover=cell(582,1);            %定义交集
Cover=[];
```

```
Isolated=[];                          %定义孤立点
a=length(Weizhi_all);
for i=1:582
    c=[];
    c2=[];
    for j=1:a
        m=length(Patrol_range{j});
        for l=1:m
            if(Patrol_range{j}(l)==i)
                c=[c,j];           %保存i节点所对应的所有可能的交通巡警点
                c2=[c2,Weizhi_all(j)];
            end
        end
    end
    m=length(c);
    if(m>1)                       %如果大于1，说明有交集，先去除，不分配
        Cover=[Cover,i];
        Patrol_cover{i}=c2;      %保存交集
        for k=1:m
            find(Patrol_distribution{c(k)}~=i);

Patrol_distribution{c(k)}=Patrol_distribution{c(k)}(find(Patrol_distribution{c(k)}~=i));%预
分配只属于自己的交通节点
        end
    end
    if(m==0)
        Isolated=[Isolated,i];
    end
end
save Patrol_distribution.mat;%完成预分配，对于交集和孤立交点另外考虑
save Patrol_cover.mat;%保存交集所对应的可能交通巡警点
load Patrol_cover.mat;
load Patrol_distribution.mat;
load Patrol_range.mat;

%初始化预分配中每个交通巡警点的发案次数
Occurrence=xlsread('cumcm2011B.xls',1,'e2:e583');              %A区每个交通节点的发案次数
a=length(Weizhi_all);
Standard_occurrence=sum(Occurrence)/a
sum(Occurrence);
Patrol_occurrence=zeros(a,1);


for i=1:a
    m=length(Patrol_distribution{i});
    a=0;
    if(m>=1)
        for j=1:m
            a=a+Occurrence(Patrol_distribution{i}(j));
        end
        Patrol_occurrence(i)=a;
    end
end


Patrol_xin=Patrol_distribution;                                    %进行交集的分配
for i=1:582
    m=length(Patrol_cover{i});
    Distance_linshi=[];
    if(m>=1)
```

```
        dist=graphshortestpath(DG,i);
        for j=1:m
            Distance_linshi(j)=dist(Patrol_cover{i}(j));
        end
        A=Sort_vector(Distance_linshi);                          %记录最小值的相对位置
        h=length(Distance_linshi);
        for j=1:h
            linshi_canshu=find(Weizhi_all==Patrol_cover{i}(A(j,2)));
            Patrol_occurrence(linshi_canshu);
            Patrol_cover{i}(A(j,2));
            if(Patrol_occurrence(linshi_canshu)<=(Standard_occurrence+8.5))
                Patrol_xin{linshi_canshu}=[Patrol_xin{linshi_canshu},i];

Patrol_occurrence(linshi_canshu)=Patrol_occurrence(linshi_canshu)+Occurrence(i);
                break;
            end
            if(j==h)
                i
            end
        end
    end
end
%对孤立点进行分配
m=length(Isolated);
for i=1:m
    dist=graphshortestpath(DG,Isolated(i));
    D=dist(1:20);
    Isolated(i);

    [m0,m1]=min(D);
    m1;
    Patrol_xin{m1}=[Patrol_xin{m1},Isolated(i)];
    Patrol_occurrence(m1)=Patrol_occurrence(m1)+Occurrence(Isolated(i));
end
Patrol_occurrence;                                  %每个警力点的工作量
length(Patrol_occurrence);
var(Patrol_occurrence)
[a,b]=max(Patrol_occurrence);
zuidazhi=a;
b;
Weizhi_all(b);
save Patrol_xin.mat;
```

%问题二（1.2）求解能满足三分钟到达目前不能三分钟到达的138个路口节点的交巡警服务平台的最少个数和位置；

```
model:
!求解能完全满足在三分钟到达目前不能三分钟到达的138个路口节点的交巡警服务平台的最少个数和位
置;
sets:
AA/1..138/;
cross/1..502/:x;
links(AA,cross): a;
Endsets
!数据的定义部分;
data:
a = @FILE(C:\Users\Administrator\Desktop\2011math\data1.txt);
@TEXT('result1.txt') = x;
enddata
```

```
!目标函数;
min=@sum(cross(j):x(j));
!需求约束;
@for(AA(i):@sum(cross(j): a(i,j)*x(j))>=1);
!整数约束;
@for(cross(i):@bin(x(i)));
```

%问题二（1.3）交巡警平台可以改变位置，数量不变;

```
model:
!交巡警服务平台可以改变位置,数量不变;
sets:
!count/1/:c;
AA/1..582/;
cross/1..582/:x;
links(AA,cross): a;
Endsets
!数据的定义部分;
data:
a = @FILE(C:\Users\Administrator\Desktop\2011math\new3\data3.txt);
@TEXT('result7.txt') = x;
!c = 0;
enddata
!目标函数;
min =@sum(AA(i):@if(@sum(cross(j):a(i,j)*x(j))#eq#0,1,0));
!需求约束;
@sum(cross(j): x(j))=80;

!整数约束;
@for(cross(i):@bin(x(i)));
```

%问题二（1.4）交巡警平台可以改变位置，数量不变;

```
model:
!求解在交巡警服务平台可以改变位置及数量的情况下的完全覆盖;
sets:
AA/1..582/;
cross/1..582/:x;
links(AA,cross): a;
Endsets
!数据的定义部分;
data:
a = @FILE(C:\Users\Administrator\Desktop\2011math\new3\data3.txt);
@TEXT('result5.txt') = x;
enddata
!目标函数;
min=@sum(cross(j):x(j));
!需求约束;
@for(AA(i):@sum(cross(j): a(i,j)*x(j))>=1);
!整数约束;
@for(cross(i):@bin(x(i)));
```

%问题二（2）最佳围堵方案;

```
% function weidu
%找出最佳的围堵方案
clear;
load DG.mat;
load Xunjinwz1.mat;
xun_gs = length(Xunjinwz);
```

```
dist=graphshortestpath(DG,32);%求图中任意个节点到案发点的最短距离
for t = 6:30
    Anfadian=[];
    for j = 1 :582
        if(dist(j) <= t*1000 & (t - 1)*1000 <=dist(j) )
            Anfadian=[Anfadian,j];
        end
    end
    n = length(Anfadian);      %罪犯可能到达点的集合
    A1 = zeros(n,582);
    for k = 1:n
        dist2 = graphshortestpath(DG,Anfadian(k));
        count = 0;
        for kk = 1 :xun_gs    %搜素罪犯到达点集合旁边的巡逻点
            if(dist2(Xunjinwz(kk)) < (t-3)*1000 )
                A1(k,Xunjinwz(kk)) = 1;
                count = count + 1;
            end
        end
        if(count == 0)
            break;
        end
    end
    [m,n] = size(A1);
    if(m < n)
        pp = Pipei(A1);
        [m,n] = size(pp);
        if(rank(pp) == m)
            pipei =zeros(m,2);
            for i =1:m
                [row,coloum] = find(pp(i,:)==1);
                pipei(i,1) = Anfadian(i);
                pipei(i,2) = coloum;
            end
            break;
        end
    end
end



function pip =Pipei(A)
%求二部图的最大匹配问题
[m,n] = size(A);
M(m,n)=0;
for(i=1:m)
    for(j=1:n)
        if(A(i,j))
            M(i,j)=1;
            break;
        end
    end %求初始匹配 M
    if(M(i,j))
        break;
    end
end %获得仅含一条边的初始匹配 M
while(1)
    for(i=1:m)
        x(i)=0;
```

```
    end %将记录X 中点的标号和标记*
    for(i=1:n)
        y(i)=0;
    end %将记录Y 中点的标号和标记*
    for(i=1:m)
        pd=1;    %寻找X 中 M 的所有非饱和点
        for(j=1:n)
            if(M(i,j))
                pd=0;
            end;
        end
        if(pd)
            x(i)=-n-1;
        end
    end %将X 中 M 的所有非饱和点都给以标号0 和标记*,  程序中用 n+1 表示0 标号,  标号为负数时
表示标记*
    pd=0;
    while(1)
        xi=0;
        for(i=1:m)
            if(x(i)<0)
                xi=i;
                break;
            end
        end %假如 X  中存在一个既有标号又有标记*的点,  则任 取X 中一个既有标号又有标记*的点
xi
        if(xi==0)
            pd=1;
            break;
        end %假如X 中所有有标号的点都已去掉了标记*,  算法终止
        x(xi)=x(xi)*(-1); %去掉xi 的标记*
        k=1;
        for(j=1:n )
            if(A(xi,j)&y(j)==0)
                y(j)=xi;
                yy(k)=j;
                k=k+1;
            end
        end %对与 xi 邻接且尚未给标号的 yj 都给以标号i
        if(k>1)
            k=k-1;
            for(j=1:k)
                pdd=1;
                for(i=1:m)
                    if(M(i,yy(j)))
                        x(i)=-yy(j);
                        pdd=0;
                        break;
                    end
                end %将yj 在 M 中与之邻接的 点xk (即xkyj ∈M),  给以标号j 和标记*
                if(pdd)
                    break;
                end
            end
            if(pdd)
                k=1;
                j=yy(j); %yj 不是 M 的饱和点
                while(1)
                    P(k,2)=j;
```

```
                    P(k,1)=y(j);
                    j=abs(x(y(j))); %任取 M 的一个非饱和点 yj,  逆向返回
                    if(j==n+1)
                        break;
                    end %找到X 中标号为0 的点时结束,  获得 M-增广路 P
                    k=k+1;
                end
                for(i=1:k)
                    if(M(P(i,1),P(i,2)))
                        M(P(i,1),P(i,2))=0; %将匹配 M 在增广路 P 中出现的边 去掉
                    else M(P(i,1),P(i,2))=1;
                    end
                end %将增广路 P 中没有在匹配 M 中出现的边加入 到匹配 M 中
                break;
            end
        end
    end
    if(pd)
        break;
    end
end %假如X 中所有有标号的点都已去掉了标记*,  算法终止
pip = M ;   %显示最大匹配 M,  程序结束
```