Why don't TPPSPs marry the network?
—-City Network Optimization Solutions

September 12, 2011

Abstract

Optimization of junction layouts in network can improve the functionalities of the network to a large extent. A suitable and reasonable Traffic and Patrol Police Service Platforms setting and scheduling can compactly increase the city's resilience against unexpected accidents. However, given the existing city network geographic profile, it is often hard to evaluate the best possible layout scheme for all TPPSPs. We propose, adapt, formulate and test the Distribution Optimization Model (DOM) which incorporates all the considerations in Evaluation Schema. DOM satisfactorily decides the optimal layout of TPPSPs in the city [8]. We also formulate and compare three models to tackle the matching problem between TPPSPs and Access Arteries, Greedy Method, Stable Marriage Matching with Indifference [6] and Hungarian Algorithm Method. The synthesis of the three methods gives us valid and sufficient information about the matching between TPPSPs and Access Arteries. We also construct Marginal Network Circle Model with Inductive Reasoning (MNCMIR), in which model real-life scenarios are mimicked and best strategy which have the highest expectation of success rate are picked by comparing experimental data . The testing results of our models show that DOM is robust with internal parameters and is able to determine the best TPPSP layout for the whole city when combined with greedy method. Hungarian Algorithm in general provides the best matching scheme between TPPSPs and Access Arteries. MNCMIR also gives the strategy of guaranteeing catching the criminal. Our modification to MNCMIR even accelerates the whole process. We conclude the methods we propose above comprises of a robust, stable and effective model.

Contents

Chapter 1

Problem Statement

1. Allocate the jurisdiction region for each TTPSP so that the police can arrive at the locale in time in case of emergency.
2. Match Access Arteries with TPPSPs in case of emergency.
3. Evaluate the reasonableness of the current network layout.
4. When a major criminal is reported to the police to have run by car 3 minutes after he leaves the scene of the crime, location P, we must find an optimal strategy which could make the police arrest the criminal as quickly as possible.

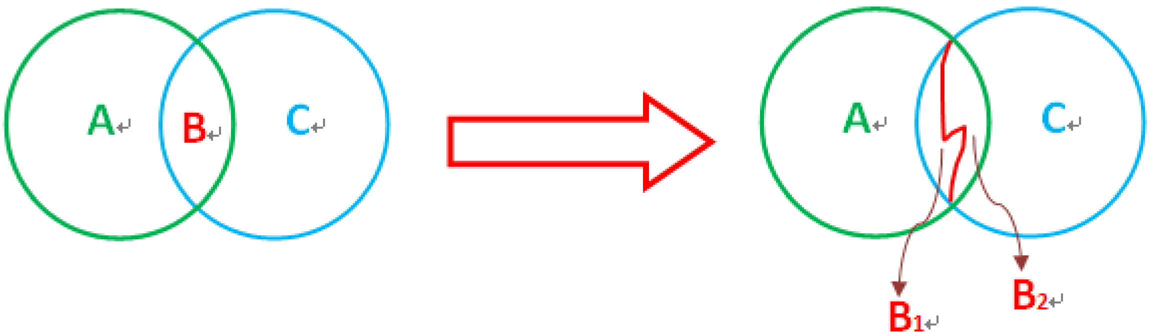


Figure 1 Figure 2

Chapter 2

Assumptions

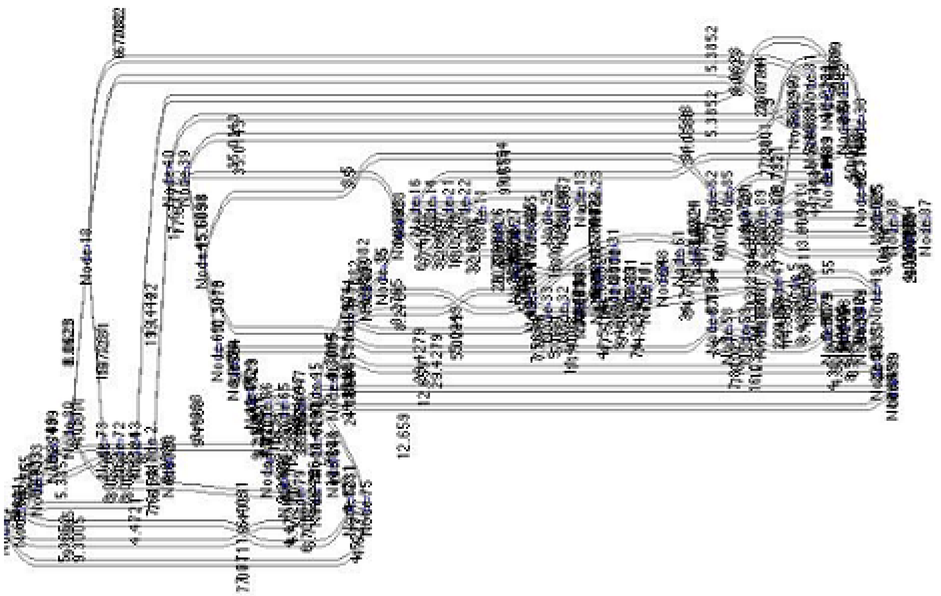
- The velocity of vehicle is constant and other physical factors, like acceleration and friction, are not taken into consideration.
- The police and the criminal use the vehicles of the same kind.
- Traffic lights and toll-gates are omitted.
- Traveling from a junction to another junction can only made possible by car instead of by air, by train or by feet.
- The police force of every TPPSP is modeled as a normal distributed random variable by empirical data.
- Administrative area for each TPPSP is the circular area with centre TPPSP with radius being the place it can reach within 3 minutes. Other areas could be included as the jurisdiction region of each TPPSP, but they are not preferable.
- Assume everyone,including both the police and the criminal, is rational. That is to say, traveling from point A to point B, everyone will choose the shortest path.
- Assume that the city's geometry is Euclidian.

Chapter 3

Methodology

3.1 Basic Idea

In our assumptions, the only way to commute between two junctions is through transportation network, namely roads. Therefore, the method we adopt to simulate the situation is using Dijkstra's Algorithm which is used for finding shortest path in a given graph. We use the variant version of this algorithm which allows the road capacity to be rational. The graph below represents all the shortest path between any two junctions in the given city.



- In order to solve the problem of distributing TPPSP among the city junctions, we quantify the ability of a TPPSP using the notion of “Police Force”. Then the problem is reduced to the distribution problem. Since we model each arc of the network stochastically incorporating other uncertainties such as road conditions, car conditions, weather conditions and so on. Therefore the model is actually analogous to solving a distribution problem with arc of random costs [4].

- Next, we propose a strategy for the blockade of Access Arteries when emergency happens. The blockade scheme often restricts one TPPSP to one Access Artery since blockade is the continuous effort which requires continuous attention. Three models are proposed, Greedy Method, Stable Marriage with indifference and Hungarian Algorithm. Comparisons are also made among different models in order to produce the most feasible and efficient strategy in the case of emergency.

- We also formulate a model to catch a criminal given the crime spot. The time needed for propagation of information is negligible in this instance. The model makes use of semi-agent-based simulation, which gives the criminal ability to reason given the information attainable. Our Marginal Network Circle Model with Inductive Reasoning (MNCMIR) mimic the process of learning by trial and error based on the simulation result.

- After validating all the models, we implement the simulation methods to get experimental results. This is the last step to verify theoretical prediction and model formulation.

3.2 Softwares used

3.2.1 Primary software used

Matlab, to program, to process data and to generate diagram.
Java, to program.
R, to process data and to generate matrix to be used in Matlab.
Microsoft Excel, to present data neatly and nicely.

3.3 Definition of Variables

In the first part of the report, we first analyze the current structure and layout of the Traffic and Patrol Police Service Platforms (TPPSPs). We define the following parameters as the measures for our evaluation.

- $G(E, V)$: The given map of the city in which E stands for the set of all roads and V represents the set of all junctions.

- O: The set of all junctions.

- A: The set of Access Arteries.

- T: The set of TPPSPs.
- O_i: Junctions, $i \in N$.
- T_i: Traffic and Patrol Police Service Platforms junctions,TPPSPs, $i \in N$.
- A_i: Access arteries, $i \in N$.
- d_i: Density of the i^{th} TPPSP.
- TI : set of indexes representing TPPSPs.
- CI : set of indexes representing junctions.
- R_i : police force of a certain TPPSP (To illustrate the newly introduced variable, R_i is measured by the decrement of accident rate caused by $TPPSP_i$. In our case, we model R_i as $N(\mu, \sigma)$ where μ is the mean of police force).
- c_{ij} : distance cost from T_i to O_j.
- x_{ij} : amount of police force in T_i to O_j.
- r_k : the accident rate in O_k.
- z_k : the decrement of accident rate because of TPPSPs in O_k.
- p_k : penalty cost per unhandled accident in O_k.

3.4 Evaluation Schema

The evaluation scheme is based on the following considerations:

1. For each T_i, based on the computed value of d_i, given the thresholding and capping overlapping rate $f(d_i)$ and $g(d_i)$, if $d_i \in (f(d_i), g(d_i))$, we do not add penalty points, otherwise we add $P_{unit_1} \cdot h(f(d_i) - d_i, d_i - g(d_i))$ to total penalty P, where

$$h(x, y) = \begin{cases} x, & \text{if } x \geq 0 \text{ and } y < 0 \\ y, & \text{if } x < 0 \text{ and } y \geq 0. \end{cases} \tag{3.1}$$

   The rationale for this penalty requirement is that given a certain district, if the density, which is the indication of how likely some accidents are to occur, is huge, then the police force required in that district should be increased accordingly. Since if some accidents occur simultaneously, one TPPSP will not be able to take care of the accidents in all the district. Therefore, the degree of overlapping in terms of the radiated area of TPPSPs should also be big. Note that the cap is to ensure that the overlapping is not too big such that it causes wastes of resources.

2. Police should arrive at where the emergency happens as soon as possible. Therefore, we first assign the circular area each centered with TPPSP and radius within the reasonable distance to each TPPSP as part of there jurisdiction area. However we find that there are still some intersections not covered by any TPPSP. In our evaluation schema, the more such kind of uncovered TPPSPs there are, the more penalty points will be added.The penalty we add is

$$p_k \cdot \frac{n}{N}, \tag{3.2}$$

where n stands for the number of uncovered intersections and N stands for the number of all the intersections.

3. After matching each Access Artery with TPPSPs (details of matching implementation will be provided in the following sections with three methods), we compute the total time t_{reach} required for each TPPSP to reach the Access Artery. We add $w \cdot n \cdot t_{reach}$ to P, where n is the normalized parameter and $w \sim N(1, \sigma)$ which counts for the uncertainty of environmental conditions during the process.
The rationale for this criterion is if the layout of the TPPSPs in the city is such that it takes a long time for them to reach Access Arteries, the possibility of a city being susceptible to certain unexpected accidents will be high.

Chapter 4

Problem Solving Models

4.1 Distribution Optimization Model

Given the evaluation criteria enumerated above, we adapt the algorithm for solving distribution problems [4] for our instance.

Deterministic Parameters

- R_i : police force of T_i (To illustrate the newly introduced variable, R_i is measured by the number of accidents that can be handled by a TPPSP. In our case, we model R_i as $N(\mu, \sigma)$).
- c_{ij} : distance cost for T_i to take care of a O_j
- x_{ij} : amount of police force in T_i distributed to O_j
- r_k : the average number of accidents in a given O_k
- z_k : the number of handled accidents in a certain O_k
- p_k : penalty cost per unhandled accident in a certain O_k

Objective Function $Q(x) = min\{\sum_{i \in TI} \sum_{j \in CI} c_{ij} \cdot x_{ij} + \sum_{k \in CI} p_k(r_k - z_k)\}$ is subject to the constraints:

- $\sum_{j \in CI} x_{ij} = R_i, i \in TI$
- $\sum_{i \in TI} x_{ij} = z_j, j \in CI$
- $z_k \leq r_k, k \in C$
- $0 \leq x_{ij} \leq R_i, i \in TI, j \in CI$

$$c_{ij} = l_{i,j} \cdot Dcost_{i,j} \tag{4.1}$$

where $Dcost_{i,j}$ is the shortest distance between T_i and O_j, and $l_{i,j} \sim N(\mu', \sigma')$ weighting uncertainties.

An efficiency improvement to the objective function to to omit r_k, since the value of r_k has no effect on our minimization process, then we rewrite the objective function as

$$Q(x) = min\{\sum_{i \in TI} \sum_{j \in CI} c_{ij} \cdot x_{ij} - \sum_{k \in CI} p_k \cdot z_k\} \tag{4.2}$$

.

Note that to incorporate the "Three-Minute" constraint into the presentation of the optimization formula, we will add time penalty p_{time} to each distance cost c_{ij}. Note that p_{time} can be made arbitrarily large to enforce a TPPSP only to stick to its own "duty"! The computation of the minimum value of the linear programs is introduced in [8] and [4], whose method, Network Recourse Decomposition Method, makes use of the separable approximation of the given linear programs. We simply make use of the already-derived results.

This model takes into consideration the first criteria of the evaluation scheme, namely, the effect of overlapping jurisdiction regions. Instead of considering the effect directly, we proceed in the indirect way. Since for every junction there is need of TPPSP, and the degree of "Police Force" reflects the density of the region and thus the corresponding overlapping degree. The penalty on unsatisfied demand for Police Force to each junction reflects the criteria requiring covering as many junctions as possible. Finally and most obviously, the costs are directly associated with the distance from TPPSPs to junctions and both of them are the minimization objectives.

Given a determined network $G(E, V)$, we could always compare $Q(G(E, V))$ to decide the optimal layout.

4.2 Adding TPPSPs to Obtain Optimal Network

The general method to obtain an optimal Network with TPPSPs is in greedy spirit. Let q be the number of TPPSPs in a network. We use marginal principle and greedy method to decide whether to add another TPPSP. Let G_q be the current network already with q TPPSPs, we now need to decide the location of adding another TPPSP in the network. Choose

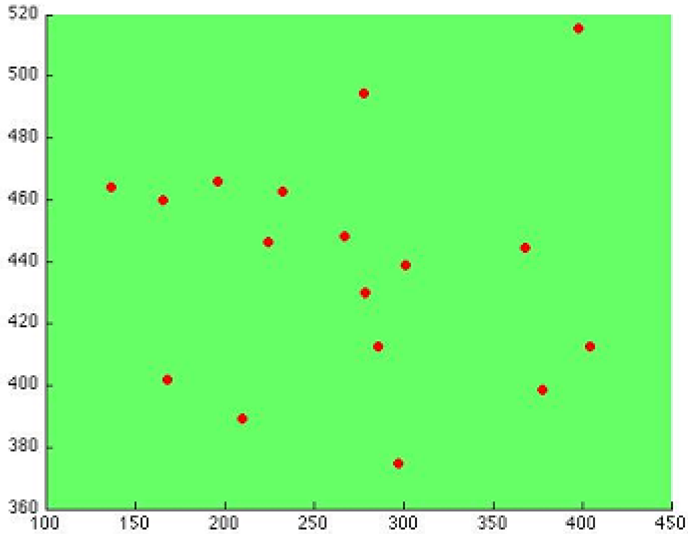
$$O_t = max_{k \in C}\{p_k \cdot (r_k - z_k)\}, \tag{4.3}$$

which is the junction with the highest unsatisfactory measure. After doing this, we compute the gradient

$$g(q+1, q) = Q_{q+1} - Q_q, \tag{4.4}$$

given a thresholding value $g_{threshold}$, if $g(q+1, q) < g_{threshold}$, then we stop the process immediately. Otherwise we proceed to the next iteration.

Proceeding in this manner, it is possible to obtain the optimal layout for a given city.



Relocated TPPSPs in District C

The figure above depicts the relocation of TPPSPs in district C using our proposed algorithm.

4.3 Implementation and Numerical Results

Adding TPPTSs Table

No. of adding TPPTSs	Point newly added	X-coordinate	Y-coordinate	Accident Rate	Objective Value
1	58	357	387	1.1	0.8627
2	81	438	368	1.4	0.7929
3	87	448	381	1.1	0.7858
4	72	418	347	0.8	0.7770
5	85	440	392	1.2	0.7746

The table gives the value of the objective function for the current layout of the city. We implement the greedy optimization algorithm to district A given the number of TPPSPs is fixed, the final result is presented in the table. It shows the current layout of district A is not optimal. Observing the accidents rates on the junctions that we pick, we have found the accident rates on these junctions are relatively high. Furthermore, for junction No.58, No.81 and No.72, the original decrement of accident rates due to the coverage of police force is less than the one-quantile of the junction's own accidents rate, therefore the adding of TPPSPs improves the objective function to a certain degree. While in the current evaluation scheme, it will always decrement the objective function when adding new TPPSP. if we pick our $g_{threshold}$ as $2.5 \cdot 10^{-3}$, which is derived by repeated experiments, then it is not necessary to add the fifth TPPSP. The evaluation of whether to add the fifth TPPSP depends on how we take other factors into consideration, such as the cost of adding a TPPSP, or the potential job opportunities created by TPPSP.

4.4 Matching Schemes

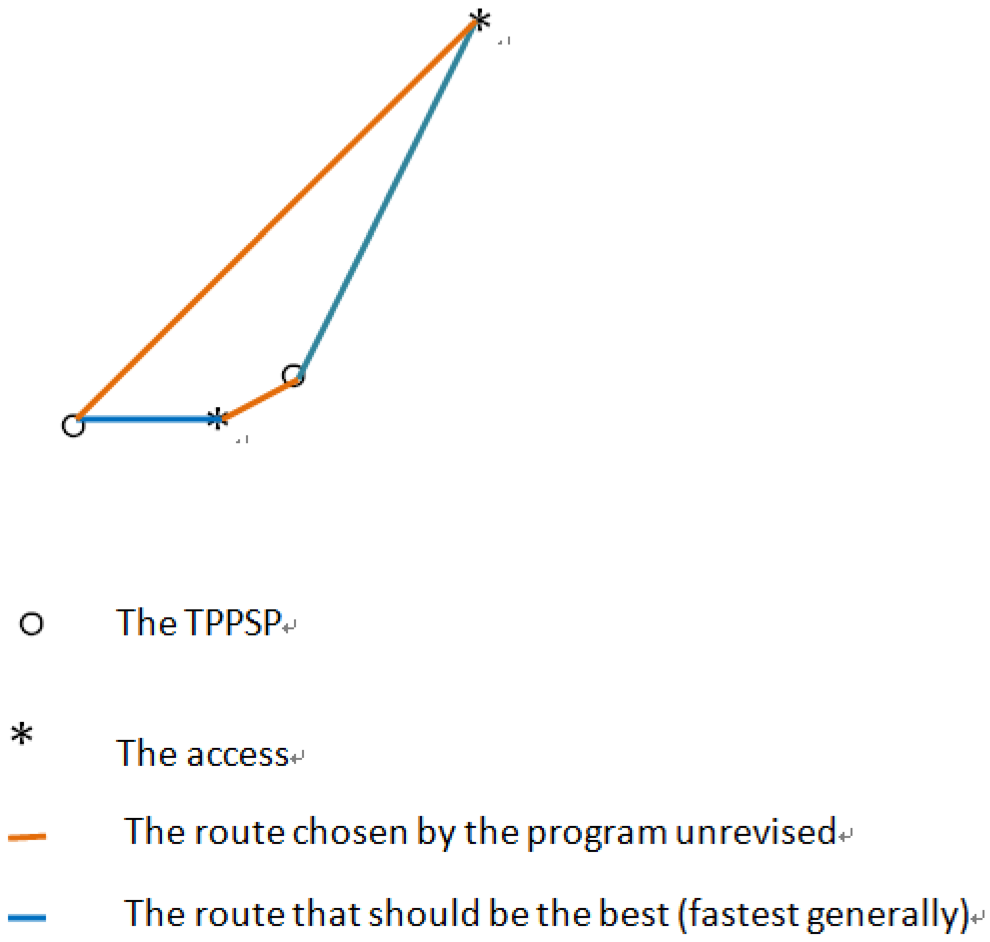
In this section, we are ready to deal with the problem of assigning TPPSPs to Access Arteries when emergency happens. Notice the constraint that each TPPSP can only take care of an Access Artery, we could utilize the following models to solve the corresponding problems.

4.4.1 Scheme 1: Greedy Method

For every Access Artery A_i, we search for the T_j such that

$$dis(A_i, T_j) = min_{T_k \in T}\{dis(A_i, T_k)\} \tag{4.5}$$

in the whole city, then they are made into a pair. After this, we delete (A_i, T_j) from the graph. Then we apply the same procedure to the remaining graph until there is not Access Artery left. This method is straightforward, however, it has some vital fatalities. For example, consider the following graph.



4.4.2 Scheme 2: Stable Marriage Problem with Indifference

We introduce the notion of stable matching from the famous stable marriage problem[1]. To adapt it to our use, we make use of one of the invariants of stable marriage problem[2], stable marriage problem with indifference.

Original Problem Stable marriage problem with indifference refers to the problem where each man or woman is indifferent among several members on his or her list. A matching is said to be strongly stable if there does not exist a pair (m, w) (we denote m's partner to be $p(m)$, and w's partner to be $p(w)$) such that

- m prefers w to $p(m)$, w prefers m to $p(w)$ or w is indifferent between m and $p(w)$;
- w prefers m to $p(w)$, m prefers w to $p(m)$ or m is indifferent between w and $p(m)$.

The "Men'' set M in this problem instance is the set of all Access Arteries, while the "$Woman''$ set W consists of all TPPSPs. The objective is to find a matching such that each Access Artery is finally matched with some TPPSP. How do we construct preference lists for every member participating in this matching scheme? First of all, we introduce a parameter "Ambiguity Measure" A_M such that each vertex, regardless of which set it is in, ranks each vertex in another set by the order of distance with respect to A_M, for example, for $v \in M$

$$Rank(u_1) > Rank(u_2), \text{ if } dis(v, u_2) + A_M < dis(u_1, u_2), \tag{4.6}$$

$$u_1 = u_2, \text{ if } |dis(v, u_1) - dis(v, u_2)| \leq A_M, \tag{4.7}$$

$$Rank(u_1) < Rank(u_2), \text{ if } dis(v, u_2) + A_M > dis(u_1, u_2). \tag{4.8}$$

In this way, a preference list possibly containing indifference is created. Since $|M|$ may not equal $|W|$, considering if $|M| > |W|$ there is no solution, if $|M| < |W|$, we simply add some dummy "men" to M such that for each dummy Access Artery, it is indifferent among any TPPSP and none of the TPPSPs prefers any dummy "man" to any normal "man".
Next, we provide the revised algorithm.

Adapted version of stable marriage problem with indifference

assign each node to be free
repeat
  PHASE 1: remove any pair causing instability from both sets
  PHASE 2:
  if the engagement graph does not contain a perfect matching **then**
    find the critical set Z;
    delete every pair at the tail of the preference list of nodes in the critical set
  end if

until (some Access Artery's list is empty) or (every node is engaged)
if everyone is engaged **then**
any perfect matching in the engagement graph is a stable matching;
else
return any maximal matching
end if

Theoretically, this algorithm may not find a perfect matching at times since the constraint is strong stability. We could always modify this constraint to be weak stability[5] to ensure there always exists a result. Nonetheless, empirical studies show that this method is generally robust since only 6.78% of our tests do not generate perfect matching.

4.4.3 Scheme 3: Hungarian Algorithm

This method was developed and published by Harold Kuhn in 1955 [3]. We formalize the problem in our specific instance. Let indication variable

$$x_{ij} = \begin{cases} 1, & \text{if } ith \text{ node is assigned to } jth \text{ node,} \\ 0, & \text{otherwise..} \end{cases} \tag{4.9}$$

We adopt the same method of appending dummy nodes to the end of the set of Access Arteries such that we could make both set equal in cardinality. It is assumed that every A_i is assigned to only one T_j and vice versa, therefore,

$$\begin{cases} \sum_{1\le j\le n} x_{ij} = 1 \text{ for } 1 \le i \le n \\ \sum_{1\le i\le n} x_{ij} = 1 \text{ for } 1 \le j \le n. \end{cases} \tag{4.10}$$

At the same time we construct a matrix (a_{ij}) where any a_{ij} represents the distance from A_i to T_j. Our objective is to choose $x_{i,j}$ such that it minimizes the total distance

$$\sum_{1\le i\le n} \sum_{1\le j\le n} a_{ij}x_{ij}. \tag{4.11}$$

Next we apply Hungarian's algorithm then we will be able to find the solution that minimizes the sum of distances from TPPSPs to Access Arteries.

4.4.4 Comparison of three schemes after implementation

Comarison of the three schemas			
	Greedy Method	Stable Marriage Method	Hungarian Algorithm
success rate	100%	93%	100%
average distance	50.9843	30.6493	26.9214
max distance	337.1218	156.3849	149.3582
min distance	0.0000	0.0000	0.0000
standard deviation	83.4939	49.8273	51.7130

Here is a table that compares the result using three different models. Success rate measures whether a method could generate a perfect matching. Greedy method and Hungarian Algorithm ensure the existence of perfect matching since there are no other forbidding constraints. Stable matching may not be able to output a perfect matching as mentioned above. Since Hungarian Algorithm aims at minimizing the total distance costs, therefore the no other method could produce a result whose average distance is shorter than that produced by Hungarian Algorithm. Considering the possible scenario that may happen in the previous section about the Greedy Method, it has a large standard deviation, which reveals the instability of this method.

We provide the detail of the implementation of Hungarian Algorithm in the Appendix and the following is the assigning scheme.

$$(12, 14, 16, 9, 11, 13, 10, 15, 8, 7, 2, 5, 4), \tag{4.12}$$

where the sequence describes the sequence of TPPSP assigned to corresponding Access Artery, for example, TPPSP number 12 is assigned to the first Access Artery under the natural numbering system of the given information.

4.5 Marginal Network Circle Model with Inductive Reasoning

In this section, we address the problem of finding a containment strategy to catch a criminal who committed a crime at some spot of the city. The virtual scenario we create to model the reality is based on infinitesimal discrete time system, namely, ϵ is a time unit and any other representation of time in the defined system is the multiple of ϵ [7]. We abbreviate $a\epsilon$ as a in the following paragraph. The following notions and assumptions are needed in the description of the method.

Notations and Assumptions

- The criminal is caught if and only if given the current location of the criminal, there is no way he could possibly go without bumping into any TPPSP;

- Every TPPSP is only able to catch a criminal at some junction;
- The time for TPPSP in different part of the city to receive the report slightly differs due to propagation effect. Define $w(d)$ to be a delay coefficient which is a function with respect to the distance from the location of the current TPPSP to the crime spot O_c, then the time for T_i to receive report is $3 + w(d(T_i, O_c))$;
- β stands for the time span within which the criminal will remain inside the city;
- Let the initial time be t_{init} when TPPSPs receive the report (after around 3 minutes). In the following paragraphs when we refer to time t, it means the relative time elapse with respect to t_{init};
- Let Re_t be the set of junctions such that the criminal could reach at time t;
- The junctions in a city are uniquely numbered;
- Assume the speed of the criminal the same as the police.

4.5.1 Step 1: Emergency Block

Let the $D(i,j)$ store the shortest path to move from junction i to junction j. It is easy to derive $D(i,j)$ using Dijkstra's Algorithm given the network layout. Given a thresholding value d_{th}, if the distance $d(C, CE)$ between a crime spot C and a city exit CE is greater or equal to d_{th}, then we immediately block those exits in the susceptible area. The rationale of this action is to prevent the criminal from escaping the city, since if it happens it would not be possible for the police in this city to catch the criminal anymore.

4.5.2 Step 2: Computation of the Marginal Network Circle

At every time step, we mark the junctions that have been reached before, and let $AlreadyReached$ be the collection set. Suppose at time t_k, let $Reach_{t_k}$ be the set of marginal junctions Re_i's that the criminal is likely to reach in t_k but not $t_k - 1$, each of which is equipped with probability p_i. For each $Re_i \in Reach_{t_k}$, we consider the set $Connect_{Re_i}$ that contains all junctions Re_l that are directly connected with Re_i. For each element $Re_j \in Connect_{Re_i} - AlreadyReached$, we compute the shortest distance $Path_{Re_i,Re_j}$ between Re_i and the nearest city exit $CE_{RE_i,Rdj}$ through Re_j. Then the probability of embarking on the road that leads to Re_j given the criminal is at Re_i is

$$Pr(Re_j, Re_i|Re_i) = 1 - \frac{Path_{Re_i,Re_j}}{\sum_{Re_j \in Connect_{Re_i} - AlreadyReached} Path_{Re_i,Re_j}} \tag{4.13}$$

At time t_k, for each $Re_l \notin AlreadyReached$ and $Re_l \in Connect_{Re_l} \cap Reach_{t_k}$, $Re_l \in Reach_{t_k+1}$. Therefore,

$$p_l = \sum_{Re_i \in Connect_{Re_l} \cap Reach_{t_k}} Pr(Re_l, Re_i | Re_i) \cdot p_i \tag{4.14}$$

In this inductive manner, we could compute all $Reach_t$ from time t_{init} to β.Note that at t_{init}, the criminal has been running for 3 minutes, therefore p_0 does not equal 1. In fact, we could extend the indexes of probability to negative numbers such that $p_{-t_{init}} = 1$.

4.5.3 Step 3: Besiege the Criminal

The principle of besieging a criminal is to block the junctions where the criminal is likely to appear at time t. By Step 2, we already know that we only need to ensure all the junctions in $Reach_t$ are blocked by time t. Let us formulate a linear program to solve the problem.

Variables and Constraints:

- O: set of junctions in $Reach_t$ and are reachable by some TPPSP, while O_i is the subset of O that contains the junctions reachable by T_i with elements O_{ij}
- x_{ij} is defined if and only if O_{ij} exists and is well defined

$$x_{ij} = \begin{cases} 1, & T_i \text{ is assigned to } O_i j \\ 0, & \text{otherwise} \end{cases} \tag{4.15}$$

- $\sum_{T_i \in T} x_{i,j} \leq 1$ for any $O_{ij} \in O_j$, for any $O_j \in O$
- $\sum_{O_{ij} \in O_j, O_j \in O} x_{i,j} = 1$ for any $T_i \in T$

The probability of successfully besieging the criminal at time t is

$$Pr(Besiege) = \sum_{T_i \in T, O_j \in O} p_j \cdot x_{ij} \tag{4.16}$$

The goal is to maximize $Pr(Besiege)$. We use a linear programming (LP)-based branch-and-bound algorithm [9].

4.5.4 Implementation

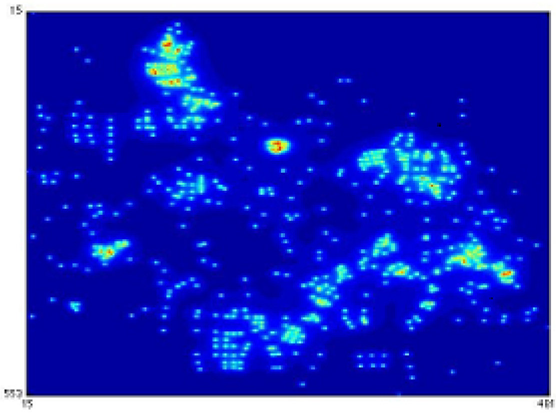


Figure 1

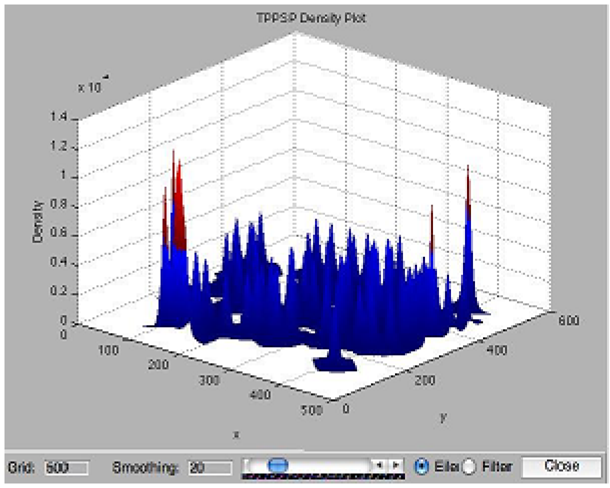


Figure 2

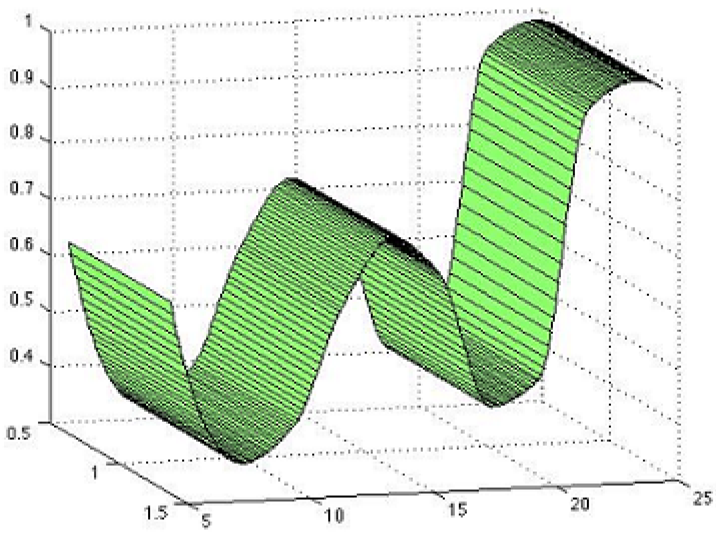
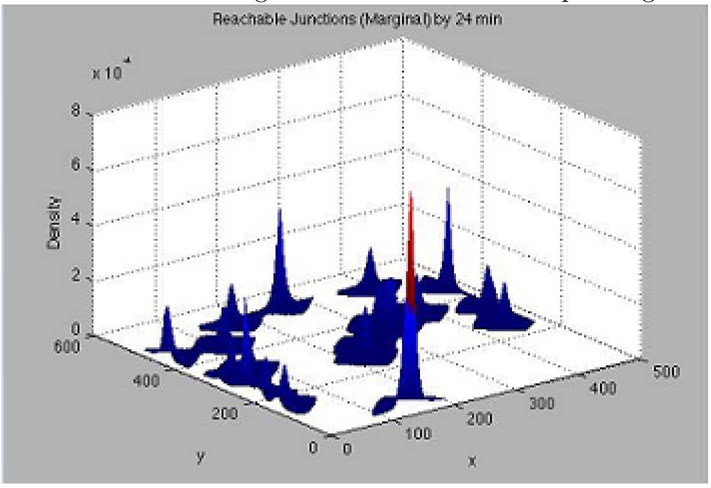
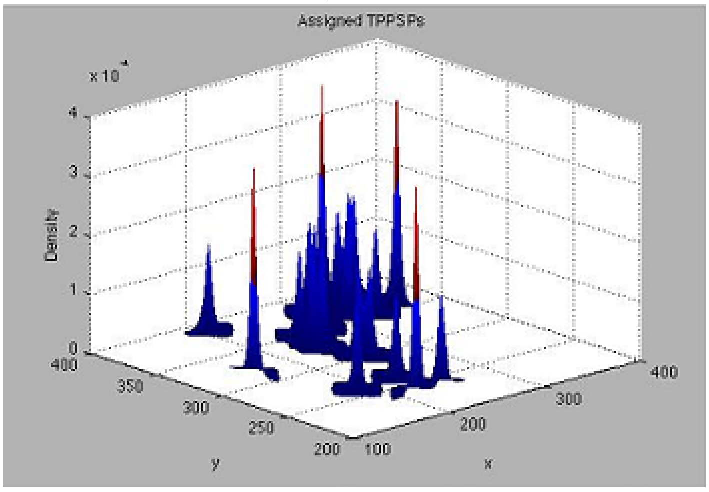


Figure 3

Figure 3 is the resulting curve we obtain with x-axis representing time and y-axis representing probability. We choose the integer points in the interval [5,25], which represents the multiples of ϵ. The figure is the graph after cubic interpolation. To explain there are two sudden drops in probability after around 8 minutes and 17 minutes, see Figure 1 that plots the density of the junctions in the city and Figure 2 that plots the distribution of TPPSPs in district A. It can be observed from the density plot of this city that the marginal network circle that the criminal reaches after 8 minutes and 17 minutes covers the high density region in the city, where the ratio of number of TPPSPs to number of junctions is about 1:10. Thus, TPPSPs are less likely to cover the possible junctions the criminal may reach at that time. Given the data we obtain in the implementation, we give two suggestions. (The following graphs describe the junctions likely to be reached by the criminal after 24 minutes and the original location of the corresponding TPP-



SPs that block those junctions.)



- Block the junctions within the marginal network circle as the criminal is predicted to reach after 14 minutes. The probability of catching the criminal after 14 minutes is 73.68%, which is the local maxima. The advantage of this strategy is that it aims to catch the criminal in a short time while it makes sure the probability is high enough. The disadvantage of this strategy is that there are still chances that the criminal will escape.

- Block the junctions within the marginal network circle as the criminal is predicted to reach after 24 minutes. In this case, the success rate of catching a criminal is close to 1. However, the disadvantage of this strategy is it takes a longer time to catch the criminal.

Criminal Containment for 14 minute matching					
Junction	X-coordinate	Y-coordinate	TPPSP	X-coordinate	Y-coordinate
12	219	316	45	342	342
14	280	292	35	336	339
16	337	328	5	339	376
21	251	277	37	331	335
22	234	271	49	342	372
23	225	265	6	335	383
24	212	290	7	317	362
28	243	328	8	334.5	353.5
29	246	337	46	342	348
30	314	367	34	328	342.5
38	371	330	31	315	351
153	150	33	30	314	367
202	347	553	247	276	361
264	109	441	235	298.5	378

Criminal Containment for 24 minute matching					
Junction	X-coordinate	Y-coordinate	TPPSP	X-coordinate	Y-coordinate
317	170	516.5	372	232.5	264
362	34	306	50	345	382
23	225	265	59	351	382
27	250.5	306	61	335	395
31	315	351	7	317	362
32	326	355	237	296	372
203	261	537.5	8	334.5	353.5
48	315	374	28	243	328
62	381	381	471	155	316
151	143	40	26	256	301
572	470	342	27	250.5	306
177	395	520	29	246	337
541	450	268	54	370	363
578	481	457	238	276	352
153	150	33	169	210	390
264	109	441	235	298.5	378
325	20	442	6	335	383
328	15	240	456	214	235
332	27	206	457	244	238
153	150	33	460	188	261
326	74	326	5	339	376
329	28	161	239	250	350

In conclusion, we recommend the first strategy if the time factor for this anti-crime action is weighted higher than accuracy. We recommend the second strategy in the other case.

4.6 Sensitivity Analysis

Distribution Optimization Model involves some randomly chosen variables such as Police Force z_k and Penalty p_k. In order to determine the effect the variation of those parameters have on our models, we implement the Distribution Optimization Model with

$$z_k \sim N(2.3, 0.01), N(2.5, 0.02), N(3, 0.01), N(4, 0.05) \tag{4.17}$$

Not surprisingly, if the z_k is larger, the decrement of accident rates of junctions that it may cause will be larger. However, although the absolute values alter, the relative relationship between different sets of results with other variables unchanged remain the same. We go through the same verification procedures for other models, those models generally show the robustness.

4.7 Strengths and Weaknesses

Strengths

- **Compact formulation of the problem.** Distribution Optimization Model makes use of the variant of distribution problem which can be used to solved similar problems. Three matching models have wide application in similar applications. Marginal Network Circle Model with Inductive Reasoning mimics the process of human reasoning, which is an essential principle for logical validation [**?**].

- **Various alternatives.** We use three methods to tackle the matching problem. It is noteworthy that the application of stable marriage problem with indifference to this problem is a novel approach. We compare and choose from the alternatives such that the final method we decide to adopt will be optimal.

- **Easily computable and applicable.** Due to the use of Dijkstra's Algorithm, the final results of the above mentioned models can be computed with modest computational resources.

- **Consistent behavior with simple assumptions and criteria.** In Distribution Optimization Model, we satisfy the criteria in the evaluation schema, such as there should be more overlapping of TPPSP administrative region when a point resides at a high-density area, without intensionally trying to do so.

Weaknesses

- **Cunning criminal.** The criteria of catching a criminal specified in Marginal Network Circle Model with Inductive Reasoning is to besiege the criminal. However, besieging a criminal does not equal catching a criminal.

- **Probability assignment.** The probability weight for each direction in Marginal Network Circle Model with Inductive Reasoning is solely based on the consideration of distance to the nearest city. In real life, other factors like alliances may also be taken into consideration.
- **Possible synergy.** The greedy method we use to decide the layout of TPPSPs in a city focuses on marginal effect a newly established TPPSP might bring. Synergy may arise in terms of functionality improvement.

4.8 Conclusion

In this report, we propose, formalize and validate several models to tackle three major problems:

- Assignment of TPPSPs
- Matching TPPSPs to Access Arteries
- Criminal Chasing Strategy

Primarily tested on district A, Distribution Optimization Model gives a solution that every junction will have an optimal share of police force. The greedy method ensures the optimality and the results obtained are satisfactory. Comparing three matching schemes that match TPPSPs to Access Arteries in case of emergency, we choose Hungarian Algorithm as the prime strategy for its performance is the most accurate and compact. Marginal Network Circle Model with Inductive Reasoning uses agent-based simulation which expects each individual to be rational and independent in decision making. The result shows that there is a trade-off between probability and time. We give two recommendations for those who have different perspectives in weighing the two factors.

Appendix A

Hungarian Algorithm

```
function [assignment,cost] = Hungarian(costMat)
% Hungarian Algorithm for Distribution Problem.
%
% [ASSIGN,COST] = Hungarian(COSTMAT) returns the optimal column indices,
% ASSIGN assigned to each row and the minimum COST based on the assignment
% problem represented by the COSTMAT, where the (i,j)th element represents the cost t
% job to the ith worker.


assignment = zeros(1,size(costMat,1));
cost = 0;

costMat(costMat~=costMat)=Inf;
validMat = costMat<Inf;
validCol = any(validMat,1);
validRow = any(validMat,2);

nRows = sum(validRow);
nCols = sum(validCol);
n = max(nRows,nCols);
if ~n
    return
end

maxv=10*max(costMat(validMat));
```

```
dMat = zeros(n) + maxv;
dMat(1:nRows,1:nCols) = costMat(validRow,validCol);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%   STEP 1: Subtract the row minimum from each row.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
minR = min(dMat,[],2);
minC = min(bsxfun(@minus, dMat, minR));

%**************************************************************************
%   STEP 2: Find a zero of dMat. If there are no starred zeros in its
%           column or row start the zero. Repeat for each zero
%**************************************************************************
zP = dMat == bsxfun(@plus, minC, minR);

starZ = zeros(n,1);
while any(zP(:))
    [r,c]=find(zP,1);
    starZ(r)=c;
    zP(r,:)=false;
    zP(:,c)=false;
end

while 1
%**************************************************************************
%   STEP 3: Cover each column with a starred zero. If all the columns are
%           covered then the matching is maximum
%**************************************************************************
    if all(starZ>0)
        break
    end
    coverColumn = false(1,n);
    coverColumn(starZ(starZ>0))=true;
    coverRow = false(n,1);
    primeZ = zeros(n,1);
    [rIdx, cIdx] = find(dMat(~coverRow,~coverColumn)==bsxfun(@plus,minR(~coverRow),mi
    while 1
        %**************************************************************************
        %   STEP 4: Find a noncovered zero and prime it.  If there is no starred
```

```
%            zero in the row containing this primed zero, Go to Step 5.
%            Otherwise, cover this row and uncover the column containing
%            the starred zero. Continue in this manner until there are no
%            uncovered zeros left. Save the smallest uncovered value and
%            Go to Step 6.
%*************************************************************************
cR = find(~coverRow);
cC = find(~coverColumn);
rIdx = cR(rIdx);
cIdx = cC(cIdx);
Step = 6;
while ~isempty(cIdx)
    uZr = rIdx(1);
    uZc = cIdx(1);
    primeZ(uZr) = uZc;
    stz = starZ(uZr);
    if ~stz
        Step = 5;
        break;
    end
    coverRow(uZr) = true;
    coverColumn(stz) = false;
    z = rIdx==uZr;
    rIdx(z) = [];
    cIdx(z) = [];
    cR = find(~coverRow);
    z = dMat(~coverRow,stz) == minR(~coverRow) + minC(stz);
    rIdx = [rIdx(:);cR(z)];
    cIdx = [cIdx(:);stz(ones(sum(z),1))];
end
if Step == 6
    % *************************************************************************
    % STEP 6: Add the minimum uncovered value to every element of each covere
    %         row, and subtract it from every element of each uncovered colum
    %         Return to Step 4 without altering any stars, primes, or covered
    %*************************************************************************
    [minval,rIdx,cIdx]=outerplus(dMat(~coverRow,~coverColumn),minR(~coverRow)
    minC(~coverColumn) = minC(~coverColumn) + minval;
    minR(coverRow) = minR(coverRow) - minval;
```

```
            else
                break
            end
        end
        %***********************************************************************
        % STEP 5:
        %  Construct a series of alternating primed and starred zeros as
        %  follows:
        %  Let Z0 represent the uncovered primed zero found in Step 4.
        %  Let Z1 denote the starred zero in the column of Z0 (if any).
        %  Let Z2 denote the primed zero in the row of Z1 (there will always
        %  be one).  Continue until the series terminates at a primed zero
        %  that has no starred zero in its column.  Unstar each starred
        %  zero of the series, star each primed zero of the series, erase
        %  all primes and uncover every line in the matrix.  Return to Step 3.
        %***********************************************************************
        rowZ1 = find(starZ==uZc);
        starZ(uZr)=uZc;
        while rowZ1>0
            starZ(rowZ1)=0;
            uZc = primeZ(rowZ1);
            uZr = rowZ1;
            rowZ1 = find(starZ==uZc);
            starZ(uZr)=uZc;
        end
    end

    % Cost of assignment
    rowIdx = find(validRow);
    colIdx = find(validCol);
    starZ = starZ(1:nRows);
    vIdx = starZ <= nCols;
    assignment(rowIdx(vIdx)) = colIdx(starZ(vIdx));
    cost = trace(costMat(assignment>0,assignment(assignment>0)));

    function [minval,rIdx,cIdx]=outerplus(M,x,y)
    ny=size(M,2);
    minval=inf;
    for c=1:ny
```

```
    M(:,c)=M(:,c)-(x+y(c));
    minval = min(minval,min(M(:,c)));
end
[rIdx,cIdx]=find(M==minval);
```

Appendix B

Distribution Model Evaluation Scheme

```
% Function DistributionEvaluation evaluate the best junction to add
% a new TPPSP
function [result score] = DistributionEvaluation(TPPSP,accidentrate)

% Resize the original matrix and align the coefficients for
% the unknowns. The penalty is set to be 0.4.

[police ordinary]=size(TPPSP);
TPPSP=tocol(TPPSP);
penalty=(-0.4)*ones(ordinary,1);
TPPSP=[TPPSP' penalty']';

% Set the equality constraint, where Aeq(X)=beq
% X stands for the unknowns. Here we set
% the police force to be a random variable with
% mean 3.0 and standard deviation 0.01

Aeq=generator(police,ordinary);
beq=normrnd(3,0.01,police+ordinary,1);
beq(police+1:police+ordinary)=zeros(ordinary,1);

% Set the inequality constraint, where A(X) <= b
% For the detailed constraints please refer to the
% distribution Model
```

```
A=eye(police*ordinary+ordinary);
b=zeros(police*ordinary+ordinary,1);
b(police*ordinary+1:police*ordinary+ordinary)=accidentrate;
b(1:police*ordinary)=(3+3*0.01).*ones(1,police*ordinary);
lb=zeros(police*ordinary+ordinary,1);

% Make use of Matlab API to compute the minimum value
% of our objective function

result=linprog(TPPSP,A,b,Aeq,beq,lb);
score=mean(TPPSP.*result);
end

function X = tocol( X )
%
% TOCOL Converts a vector or a matrix into a column vector.
%   If input is already a column vector, it is returned with no change.
%   If input is a row vector, it is converted into a column vector and
%   returned.
%   If input is a matrix, each row is converted into a column, and all
%   resulting columns are placed in series into a single column which is
%   returned.

    % check if input is a vector
    [ m, n ] = size(X);
    if m*n==m
        return % input is already a column vector with n rows
    elseif m*n==n
        X = X'; % input is converted from row vector to column vector
    elseif (m*n>n) || (m*n>m)
        X = X';
        X = X(:); % input is converted from matrix to column vector by row
    else
        X = []; % input is unknown and an empty output is returned
    end

end
```

```
function y = generator( row,col )
% Function generator generates Aeq which is tailored to the variables
% format required for this linear program.

A=zeros(row+col,row*col+col);
record=1;
for i=1:row
    for k=1:col
        A(i,record+k-1)=1;
    end
    record=record+col;
end
record=1;
minus=row*col+1;
for j=row+1:(row+col)
    for t=1:col:(row*col)
        A(j,record+t-1)=1;
    end
    A(j,minus)=-1;
    minus=minus+1;
    record=record+1;
end
y=A;
end
```

Bibliography

[1] D. Gale and L. S. Shapley, *College admissions and the stability of marriage*, Amer. Math. Monthly 69 (1962), 9-15.

[2] Robert W. Irving *Stable marriage and indifference*, Discrete Applied Mathematics 48 (1994) 261-272.

[3] Harold W. Kuhn, *The Hungarian Method for the assignment problem*, Naval Research Logistics Quarterly, 2:83∼97.

[4] Raymond K,M,Cheung, Warren B. Powell *Models and Algorithms for Distribution Problems with Uncertain Demands*, `http://castlelab.princeton.edu/Papers/dist_rev2.pdf`

[5] David F. Manlove, *Stable marriage with ties and unacceptable partners*, University of Glasgow Technical Report, TR-1999-29, January 1999.

[6] David F. Manlove, *The structure of stable marriage with indifference*, Discrete Applied Mathematics, 122 (2002) 167-181.

[7] Gershenfeld, Neil A, *The Nature of mathematical Modeling*, Cambridge University Press, 1999, P18

[8] W.B.Powell, R,K.-M. Cheung, *Stochastic programs over trees with random arc capacities* Networks, 24:161-175, 1994

[9] A. H. Land and A. G. Doig (1960), *An automatic method of solving discrete programming problems*, Econometrica 28 (3): pp. 497-520