B.Sc. in Computer Science and Engineering Thesis

A Graph Theoretic Approach for Maximizing Target Coverage using Minimum Directional Sensors in Randomly Deployed Wireless Sensor Networks

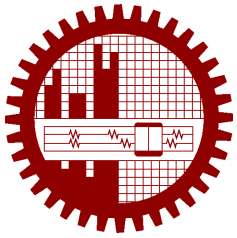
Submitted by

Sakshar Chakravarty
201305002

Laboni Sarker
201305115

Supervised by

Dr. A K M Ashikur Rahman



Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

Dhaka, Bangladesh

October 2018

CANDIDATES' DECLARATION

This is to certify that the work presented in this thesis, titled, "A Graph Theoretic Approach for Maximizing Target Coverage using Minimum Directional Sensors in Randomly Deployed Wireless Sensor Networks", is the outcome of the investigation and research carried out by us under the supervision of Dr. A K M Ashikur Rahman.

It is also declared that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

Sakshar Chakravarty
201305002

Laboni Sarker
201305115

CERTIFICATION

This thesis titled, **"A Graph Theoretic Approach for Maximizing Target Coverage using Minimum Directional Sensors in Randomly Deployed Wireless Sensor Networks"**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering in October 2018.

Group Members:

Sakshar Chakravarty

Laboni Sarker

Supervisor:

Dr. A K M Ashikur Rahman
Professor
Department of Computer Science and Engineering
Bangladesh University of Engineering and Technology

ACKNOWLEDGEMENT

First and foremost, we would like to thank almighty that we could complete our thesis work in time with promising findings. We show our heartfelt gratitude toward our supervisor Dr. A. K. M. Ashikur Rahman, who was very helpful during the entire span of our research work. Without his direction, this work would not have been possible. We would like to thank the Department of Computer Science and Engineering for its support with resources and materials during the research work. We would also like to thank Bangladesh University of Engineering and Technology for its generous support and research grant. The university also provided us with its library facilities and online resource facilities.

Last but not the least, we are thankful to our parents, family and friends for their support and tolerance.

Dhaka
October 2018

Sakshar Chakravarty

Laboni Sarker

Contents

List of Figures

List of Tables

List of Algorithms

ABSTRACT

The coverage maximization by activating minimum number of sensors is one of the most fundamental problems in wireless sensor networks. The use of directional sensors adds new dimension to it. It has great applications in security, tracking, monitoring and surveillance systems. As the problem is known to be NP-hard, most of the previous works attempt different greedy heuristics for achieving a near optimal solution. Unlike other traditional research works, we tackle the problem in a graph theoretic approach and provide novel solutions using different polynomial order graph algorithms. We have first formulated a method to model the problem using *conflict graph*. We have then developed three novel heuristics i.e.; Total Max Conflict Heuristics (TMxCH), Total Min Conflict (TMnCH) and TMxCH with Shadow Edge (TMxCHSE). These heuristics are then simulated on randomly generated deployment scenario. From these simulations, we have evaluated the performances of these heuristics with the traditional greedy heuristics (CGA). These heuristics perform better than the traditional CGA on the basis of three performance metrics namely, Coverage Ratio (CR), Ratio of Active Sensors (RAS) and Target Coverage Per Sensor (TCPS). Use of graph models also significantly reduces the run-time complexity. The graph model helps to localize the search space while removing the targets covered at each iteration leading to significant reduction of run-time. So, our developed heuristics not only provides better performance but also reduces the run-time complexity for achieving the near optimal solution. Since we have used graph model for the first time in this problem domain, this approach can be used to attack other problems related to this field, like k-coverage, balanced k-coverage, heterogeneous k-coverage, etc. Finally, we have developed the solutions considering static deployment scenarios. There is opportunity to extend the work for mobile networks as well.

Chapter 1

Introduction

Coverage problem is one of the fundamental problems in the context of wireless sensor networks. A number of studies have been conducted to achieve maximum target coverage by activating least amount of sensors. More elaborate studies are available in the literature for the so-called *isotropic wireless sensors* which have a circular or disk shaped uniform sensing range. Consequently, different algorithms have been developed for coverage maximization using such sensors. But a limited number of works are available for *directional sensors* having tunable orientations. A directional sensor is able to sense in a particular direction at a time by fixing its orientation. Directional sensor networks have many applications such as surveillance, monitoring and tracking of objects, sensing and environmental monitoring, healthcare etc., to name a few. A very common example of directional sensors is cameras which has become an essential device in the maintenance of security.

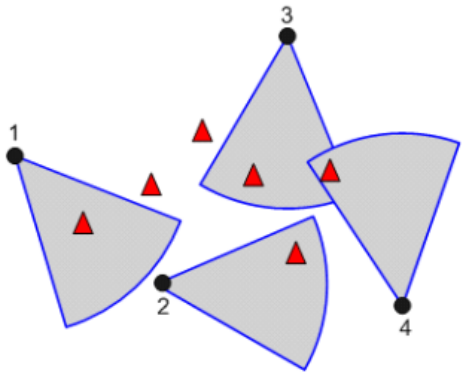


Figure 1.1: A random deployment scenario.

The solution to coverage problem focuses on two aspects; maximization of the coverage area or maximization of the number of discrete targets covered using minimal number of activated sensors at any instance. The process of selecting a sensor usually follows two steps; (i) selecting

a sensor and, (ii) fixing its orientation. The process is proven to be an NP-complete problem. Therefore, to get a near optimal solution in polynomial time, we provide several heuristics in this work. Most of the existing works on directional sensor networks are implemented using Greedy approach. However, although a greedy approach can provide near optimal solution in most of the cases, there are many cases where it fails. Another issue with greedy approach is that the process of selecting a sensor and fixing its orientation is merged together. Splitting these two steps can provide more flexibility in developing many heuristics as we demonstrate in this thesis.

Unlike existing works, in this work we took an unorthodox approach for the problem formulation and in providing necessary solutions. We model the problem using graph theoretic approach so that one can have the opportunities to experiment with already-known graph algorithms having polynomial time complexity. We develop a weighted multi graph from the configurations of the sensors and targets that we term as a conflict graph. Using the information available from the conflict graph, we can easily develop different heuristics for selecting a sensor. Then we fix the orientation of the chosen sensor greedily. We have considered a random deployment environment since deterministic deployment is impractical. In inclement scenario, random deployment (see Figure 1.1) is more suitable. Here, a fixed number of targets and sensors are randomly deployed. The locations of all the targets are known to each sensor. Thus, each sensor knows which targets are covered by it in each orientation. Using this information we can develop a conflict graph where each sensor acts as a vertex and a weighted edge is added between two vertices if there is at least one common target covered by the both sensors for some orientation of each of them. The weight of each edge represents the number of common targets covered by the two sensors at their particular orientations. We then implement different heuristics using this conflict graph to solve the target coverage problem.

Besides, the problem can be viewed from two different perspectives; sensor-oriented and target-oriented. All the existing works are based on either one of them. In the proposed heuristics we have combined the merits of both the approaches to attain better performance.

In greedy approach, each selection affects the problem model in such a way so that it requires sorting of the remaining sensor-orientation pairs based on the number of targets covered by them. But conflict graph approach localizes this effect of sensor selection within the neighboring sensors only. So, conflict graph approach has an upper hand over greedy approach with respect to time complexity. Besides, most visual sensor networks (VSN) are over-provisioned systems. Greedy approach performs well for under provisioned systems, but performs poorly in an over-provisioned system. The proposed conflict graph approach provides a way around this drawback of the greedy approach.

1.1 Our Motivation

All the related works are formulated based on greedy approach. They have designed their solution either in a sensor-oriented or a target-oriented way. Each of the existing algorithm fuse the selection of a sensor and fixing its orientation into a single step. Most of them focused only on target coverage maximization, giving less priority towards sensor minimization. All of the discussed works have another limitation in the process of eliminating targets which get covered at each iteration after the selection of a sensor. To remove targets that are covered at the current selection, they required to search the total set of inactive sensors each time.

In this thesis we are proposing to model the problem in graph theoretic approach which will add a new dimension to this problem. After modeling the problem we can apply well known graph algorithms and use graph properties to find better heuristics as the solution of the problem. Again in the proposed system, the single step of selecting the sensor and orientation is split into two steps of selecting the sensor and then selecting the best orientation of the selected sensor. The traditional methods what we have studied or discussed is either sensor-oriented or target-oriented. When we complete modeling the problem using graphs, we can implement both at the same time depending on different heuristics. As in the previous methods we have no prior information about which targets are covered by more than one sensor, when we have to remove the already covered targets from the target set of the remaining sensor sets after selection of a sensor we have to search all the sensors and each of their orientations. But in our proposed one, this search will be more localized and will involve search only through the neighboring sensors which has potential to reduce the run-time.

1.2 Our Contribution

We have modeled the problem using graph for the first time in this problem domain. It ushered the opportunity to apply different heuristics suitable for graphs to this problem. We have developed three heuristics:

- Total Max Conflict Heuristic: TMxCH (see algorithm 1)
- Total Min Conflict Heuristic: TMnCH (see algorithm 2)
- TMxCH with Shadow Edge: TMxCHSE (section 4.1 and figure 4.3) (see algorithm 3)

We have also implemented an more efficient way to remove the covered targets at each iteration which reduces the run-time significantly without compromising the performance of the heuristics. The graph modeling enables us to perform localized search while removing the covered targets and update the graph after each iteration faster (section 4.3)

We have evaluated the performance of our 3 heuristics with the existing Centralized Greedy Algorithm (CGA) [1] and also compared the results to come the conclusion that our heuristics can out-perform the existing one's based on 3 performance metrics and run-time (see chapter 6). The 3 performance metrics, we have used to compare the different heuristics are:

- Coverage Ratio (CR) (see eq. 6.1)
- Ratio of Active Sensors (RAS) (see eq. 6.2)
- Target Coverage Per Sensor (TCPS) (see eq. 6.3)

Besides, we have split the steps of selecting a sensor and its orientation at each iteration. We have also fused the two approaches i.e.; sensor-oriented and target-oriented to attack the problem.

Chapter 2

Background and Problem Formulation

In this chapter at first we introduce the VSN with relevant parameters. Then we formalize the coverage problem that we are trying to solve in this thesis.

2.1 Visual sensor network description and parameters

The sensing region of a directional sensor (camera) can be characterized by its Field of View (FoV) defined as follows:

Field of View (FoV): The FoV of a camera is the extent of the observable/sensing region that can be captured at any given direction. Some cameras come with fixed-FoV and for some, FoVs are adjustable. The smart cameras used in current VSNs are known as Pan-Tilt-Zoom (PTZ) cameras where FoV can be self-adjusted in three dimensions:

i. horizontal movement in pan

ii. vertical movement or tilt

iii. change in depth-of-field by changing zoom

In this thesis, we limit ourselves to pan-only cameras, i.e., we assume that a camera can move only in horizontal direction and its FoV is only described by its pan.

The pan of a camera is formally defined using the following two parameters:

R_s**:** Maximum coverage range of the camera beyond which a target cannot be detected with acceptable accuracy in a binary detection test.

θ: The maximum sensing/coverage angle of a camera in a certain direction. This angle is also known as Angle of View (AoV).

Thus, when a camera is oriented towards a particular direction, it can cover a circular sector (called a pan) defined by R_s and θ. We assume that every camera possesses a specific number of non-overlapping pans, of which, only one can be selected in a particular deployment. For example: a camera with FoV defined by $\theta = \frac{\pi}{4}$ can pick any one of eight disjoint orientations. fig. 2.1 depicts these parameters of camera coverage. Here, two cameras c_1 and c_2 have eight pans each and can be oriented towards any of these eight pans. We assume that cameras are homogeneous in terms of parameters. Position of a target and a sensor are expressed through Cartesian coordinates (x, y) in a two-dimensional plane. $\vec{d_{ik}}$ is a unit vector which cuts each pan (i.e., the sensing sector) into half representing the orientation of camera c_i towards pan p_k. $\vec{v_{ij}}$ is a vector in the direction from camera c_i to target t_j.

Target in Sector (TIS) Test: With TIS test (Ai and Abouzeid, 2006) one can verify whether a target t_j is coverable by a given sensor c_i. To conduct this test, at first we calculate the angle ϕ_{ij} between camera orientation $\vec{d_{ik}}$ of pan p_k and the target vector $\vec{v_{ij}}$.

$$\phi_{ij} = cos^{-1}\left(\frac{\vec{d_{ij}}.\vec{v_{ij}}}{\left|\vec{d_{ij}}\right| |\vec{v_{ij}}|}\right) \tag{2.1}$$

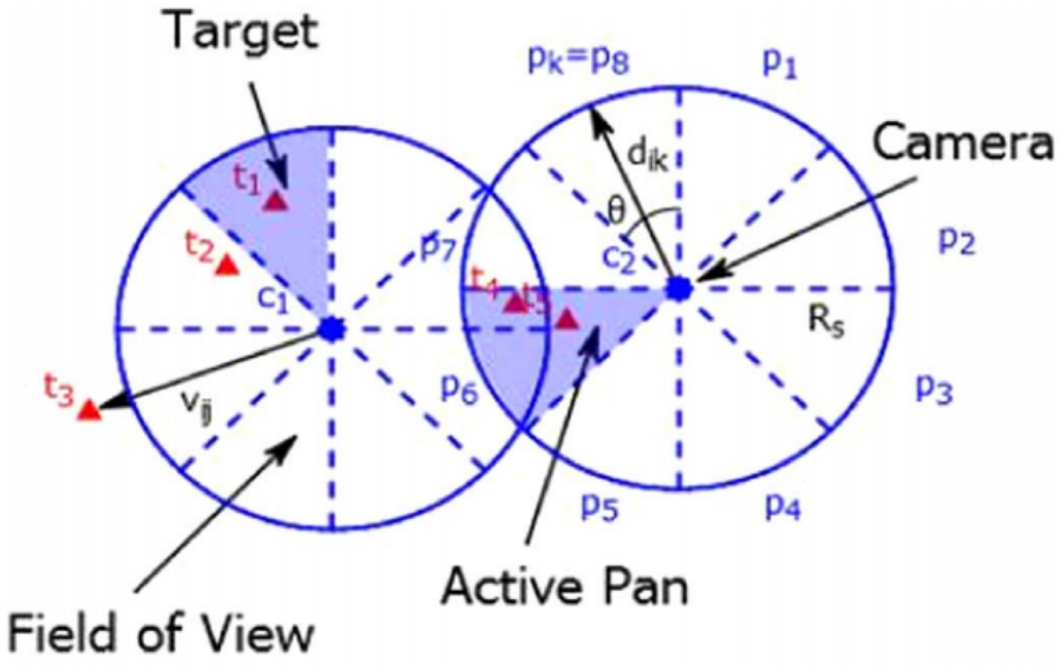


Figure 2.1: Coverage parameters of a directional sensor

A target is coverable by a camera's FoV if the span of its FoV contains the target and the target is located within the sensing range of the camera. Geometrically, $\vec{d_{ik}}$ divides the pan p_k into two equal halves and if a target is located in either of them, it is coverable by that camera on the pan p_k. Thus, the angle ϕ_{ij} needs to be less than or equal to half of the AoV, i.e., $\frac{\theta}{2}$. The other condition requires that the target has to be inside the maximum sensing range of the camera, i.e., $|\vec{v_{ij}}| \leq R_s$.

Conducting TIS tests over every pan p_k of camera c_i and every target t_j, we can build a binary coverage matrix $A^M_{N\times Q}$ of the network comprising of M targets and N cameras with Q pans where an entry in the matrix can be calculated as:

$$a^j_{ik} = \begin{cases} 1 & \text{if target } t_j \text{ is covered by camera } c_i \text{ at pan } p_k \\ 0 & \text{otherwise} \end{cases} \tag{2.2}$$

2.2 Problem formulation

In this section we provide a formal description of the problem. Suppose we have a set of M randomly deployed targets $\tau = \{t_1, t_2, ..., t_M\}$ and a set of N homogeneous directional cameras $C = \{c_1, c_2, ..., c_N\}$, each of which can be (self) orientated towards one of the Q possible pans $\rho = \{p_1, p_2, ..., p_Q\}$. Let us define a tuple $\langle c_i, p_k\rangle$ which is a camera-pan pair denoting a camera c_i oriented towards pan p_k for any $1 \leq i \leq N$ and $1 \leq k \leq Q$. Define a function $F(\langle c_i, p_k\rangle)$ which returns the set of targets covered by the tuple $\langle c_i, p_k\rangle$. Suppose δ is the set of tuples that can cover at least one target, i.e.,

$$\delta = \{\langle c_i, p_k\rangle : |F(\langle c_i, p_k\rangle)| > 0\} \quad i\epsilon[1, N]\,; k\epsilon[1, Q]$$

The goal is to find a subset $\delta^{'} \subseteq \delta$ such that $\left|\delta^{'}\right|$ is the minimum among all cardinalities of possible subsets of δ, and the quantity $\left|\cup_{\langle c_i,p_k\rangle\epsilon\delta'} F(\langle c_i, p_k\rangle)\right|$ gets maximized.
In other words, we seek to find orientations of minimum number of cameras to cover maximum number of targets which is also traditionally known as *Maximum Coverage with Minimum Sensors* (MCMS) problem.

Chapter 3

Literature Review

Extensive works have been done over the past years in the area of wireless sensor networks for isotropic sensors. Only a few works are done with directional sensors. Generally, directional sensors are considered as visual sensors since they can sense only along a particular direction at any time. The primary goals of the research works in the area of VSN are design issues related to optimal placement and orientation of the visual sensors, energy efficient scheduling techniques to increase network lifetime, cost-effective transmission system for obtained sensor data, and fault tolerant network design to attain secured connectivity of the network. Yap and Yen (2014) [2] have worked on these design issues. Liu et al. (2016) [3] have done a comprehensive survey on different design aspects of VSN. There has been some works regarding optimal placement and orientation of sensors to achieve some predefined goals. Chakrabarty et al. (2002) [4] investigated on optimal placement problem considering the deployment area as a grid of coordinates. Only one visual sensor can be placed at a coordinate within the grid. They designed an Integer Linear Programming (ILP) model for solving small scale instances of the problem and for solving large scale instances, followed a Divide and Conquer approach. Hrster and Lienhart (2006) [5] devised a rank based greedy heuristic for placing the sensors where rank of sensor was calculated based on its cost and achievable target coverage. Sensor placement is a very important problem for VSNs. But it this thesis, we only focus on the problem of selection and orientation of the randomly deployed sensors. In a randomly deployed environment, it is required to keep the number of active sensors at a time to a minimum due to limitation on energy consumption. So, the most important question in this context is how to select the sensors in an appropriate manner? The solution to this problem can be viewed from two directions: - (i) sensor-oriented approach, and (ii) target-oriented approach. There has been several works focusing on these two approaches. All of them aimed at maximizing target coverage with minimum number of sensors. Ai and Abouzeid (2006) [1] proposed an ILP formulation that gave an exact optimal solution to this MCMS problem. But it is not suitable for large-scale scenario due to its high computational complexity and low computational capa-

bilities of the deployed sensors. As a solution to this issue they proposed a Centralized Greedy Algorithm (CGA) and a Distributed Greedy Algorithm (DGA). DGA was introduced to achieve scalability in place of CGA. But CGA had some shortcomings in case of resolving tie between sensors. To come round the shortcomings of CGA, Munishwar and Abu-Ghazaleh (2013) [6] proposed another heuristic based on greedy approach. They called it Centralized Force-directed Algorithm (CFA). The modification they made was that if a sensor covered some targets in only one pan, that sensor is selected first. So, they created a way to assign priority to each sensor pans. Instead of activating a sensor having maximum number of uncovered targets in some pans, they decided to activate the sensor having targets only in a single pan first and cover those targets. To find out such sensors, they introduced a concept, force which is the measurement of priority assigned to each pan. The force of a sensor-pan pair is the ratio of coverable targets in that pan of the sensor to the total number of uncovered targets available in all pans of that sensor. The higher the value of force, the higher is the priority. The value of force equal to 1 indicated that the sensor covered targets only in that pan and subsequently got selected first. Munishwar et al. (2011) [7] also suggested a distributed algorithm to achieve coverage maximization. Here, they overlooked the idea of minimizing active sensors. Initially, all sensors assign a unique priority value by themselves. Then each sensor detects the total number of targets it can cover in each direction and orients itself to the direction of maximum target coverage. This orientation information is exchanged among the sensors located within twice of the sensing range. If there is more than one sensor covering a particular region, the orientation of the highest priority sensor prevails. For assigning priority they proposed two approaches: - (i) an area based approach, and (ii) a target based approach. In area based approach higher priority is assigned to the sensor with lower degrees of overlap in coverage with other sensors. In target based approach, a sensors priority is higher if it has lower number of useful pans, in other words, it has coverable targets in fewer pans than other sensors. All research works discussed till now are based on sensor oriented approach. There has been works on target oriented approach too. H. Zannat et al. (2016) [8] proposed three different heuristics for target oriented approach. The proposed algorithms are Greedy Target Oriented Heuristic (GTOH), Pure Target Oriented Heuristic (PTOH), and Hybrid Target Oriented Heuristic (HTOH). This approach differs from sensor oriented approach in a very vital aspect. In sensor oriented approach, priority is assigned to sensor and all targets are treated equally. But here, targets are assigned some priority and in each iteration, higher priority targets are selected and a sensor-pan pair is assigned to it. In GTOH, targets coverable by a single sensor (lonely targets) is given the highest priority. However, targets coverable by more than one sensors are assigned equal priority without any consideration of number of sensors covering them. PTOH differs from GTOH in the weight assignment of targets and the sensor selection criteria at each iteration. HTOH combines the ideas of both GTOH and PTOH. The weight assignment of targets and rank calculation of sensor-pan pairs are similar to PTOH. But the rule for selecting sensor-pan pairs follows GTOH. There has been a few works (Fusco and Gupta, 2009 [9]; Lu and Yu, 2014 [10]; Costa et al., 2014 [11])

focusing on k-coverage problem in directional sensor networks. These works are designed to allow redundant sensor activation so that multiple coverage can be ensured to all or prioritized targets. Costa et al. (2014) [11] proposed an algorithm to achieve a high level of monitoring redundancy for some critical targets by concurrently viewing them using more than one visual sensor. The optimal solution to k-coverage problem has been proved to be NP-hard by Fusco and Gupta (2009) [9]. They provided a centralized greedy solution to the problem assuming that each sensor has overlapping pans instead of discrete pans. They also proposed approximation algorithms to some related problems on directional sensors: (i) orient all the given sensors in order to maximize coverage, (ii) place and orient a minimum number of sensors in order to cover the given area, and (iii) place and orient the given number of sensors to maximize the area covered. Malek et al. (2016) [12] identified a novel issue called coverage balancing in k-coverage problem and provided centralized solutions to jointly solve k-coverage and coverage balancing. The solutions to k-coverage problems address fault tolerance issue which is beyond the scope of this paper and we limit ourselves only to single coverage of each target.

Chapter 4

Methodology

In this chapter, we elaborately discuss our methodology explaining different heuristics which we have implemented. Here, we discuss the proposed algorithm which is based on conflict graphs. The section 4.1 begins with the definition of conflict graphs.

4.1 Conflict Graph

Using graph theoretic approach at first we build a *conflict graph*. The nodes of this graph are the deployed sensors. The edges between two sensors are created on the basis of the targets in conflict. The targets which are covered by more than one sensors are called conflicted targets. For each pair of orientations between two sensors, there will be an edge if there are one or more conflicted targets. That means each edge actually represents the conflict or coverage overlap between a pair of orientations of two sensors. Thus the graph becomes a weighted multi-graph as there could be more than one edge between two nodes indicating conflicts in multiple orientation pairs between them. The weight of the edges are the total number of conflicted targets involved in an orientation pair. So, we name the graph a *conflict graph*. Let us explain the conflict graph using an example. In figure 4.1 there are three sensors and fifteen targets that are randomly deployed. Sensor, s_1 has six and one target in range along pans, p_1 and p_5 respectively. Sensor, s_2 has seven and three targets in range along pans, p_3 and p_7 respectively. Lastly, sensor, s_3 has two, one and five targets in range along pans, p_1, p_4 and p_5 respectively. In *conflict graph* of the given scenario of figure 4.1 there are exactly three nodes that represent three sensors. We observe that there are four overlapping pair of pans where there exist conflicted targets. We add an edge with weight 5 between nodes, s_1 and s_2 since there are five conflicted targets in pan pair, p_1 and p_3 of those sensors respectively. Similarly, we have an edge with weight 4 between s_2 and s_3 in pans, p_3 and p_5 of them respectively. There are conflicted targets in two pairs of pans between nodes, s_1 and s_3. Therefore, we add two edges with weights 5 and

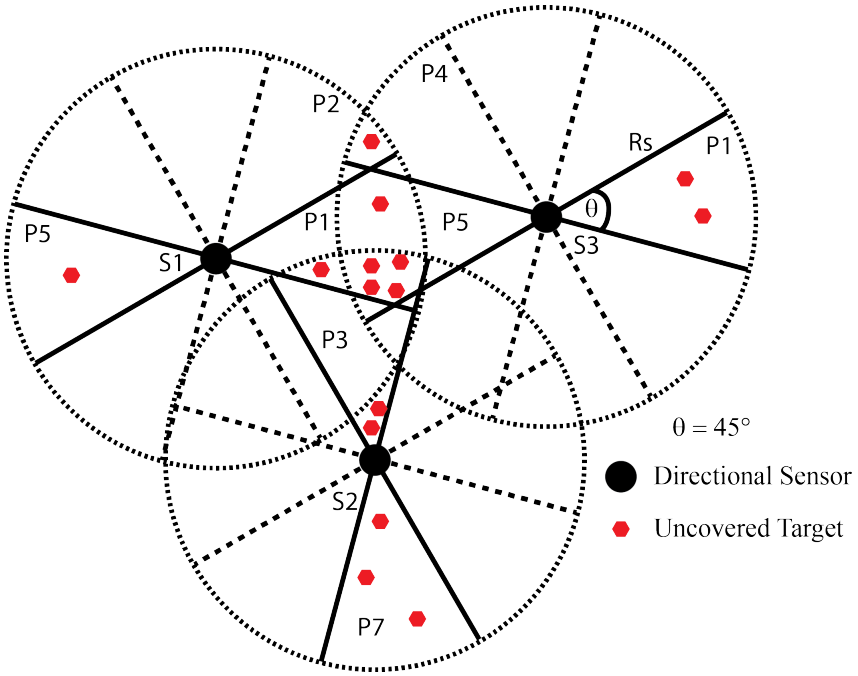


Figure 4.1: An example scenario

1 between them in pan-pairs, $\langle p_1, p_5 \rangle$ and $\langle p_2, p_4 \rangle$ respectively (see figure 4.2). Here, the sum of weights of edges connected to node, s_1 is 11. It means that there are total eleven conflicted targets with other sensors in different pan-pairs of the associated sensor and the other. Similarly, there are total 9 and 10 conflicts associated with sensors, s_2 and s_3 respectively. Besides edges

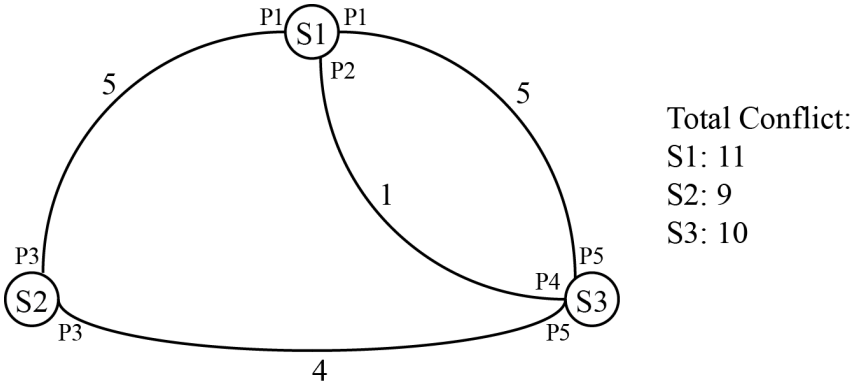


Figure 4.2: Conflict graph (Weighted multi-graph)

repsenting conflicts, there will be another set of self-edges in all sensors in the graph. We name them *shadow* edges which will be created for each orientations of a sensor. The weight of these shadow edges will be different from the other edges. The weight here will represent the total number of non-conflicted or lonely targets in each orientation of a sensor. In target-oriented method, the lonely targets are looked into at first whereas in our approach without looking at the targets we can keep track of the lonely targets. In that sense, we can incorporate the merits of target-oriented method in our approach. Here, we add shadow edges to the *conflict graph* of

figure 4.2 and get the graph as of figure 4.3. From figure 4.1, we can see that there is a lonely target in pan, p_5 of sensor, s_1. So, we add an edge with weight 1 between node, s_1 and the the shadow node of itself. Similarly, there is an edge with weight 2 between shadow node of s_3 and itself as there are two lonely targets in pan, p_1 of sensor, s_3. Finally, we add two edges with weights 2 and 3 for pans, p_3 and p_7 of sensor, s_2 for the lonely targets falling in those two pans respectively.

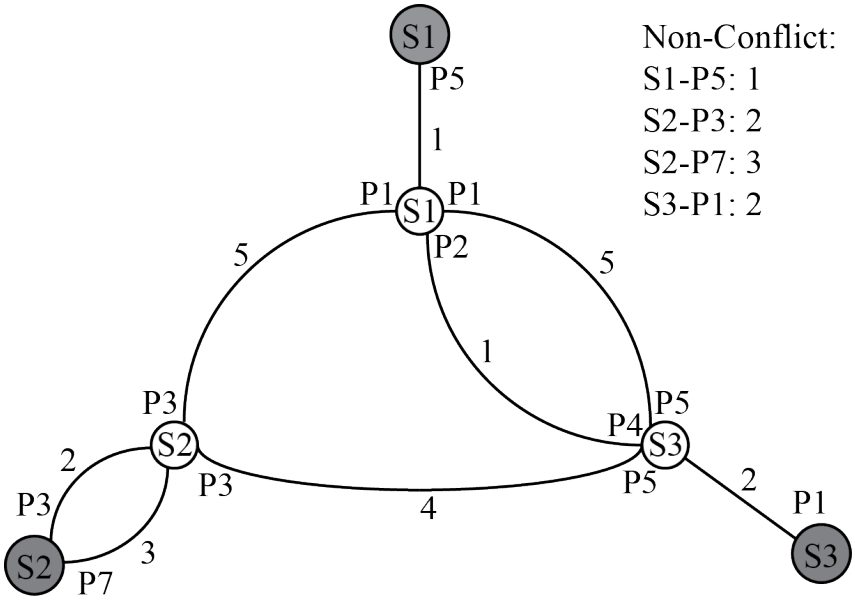


Figure 4.3: Conflict graph with shadow edges

4.2 Heuristics

After modeling the graph, we get the information of total conflicts of all the sensors and also the total number of non-conflicted or lonely targets. Unlike traditional methods, we are spitting the selection of sensor and its orientation in two steps. At first we will select the sensor from the modeled graph in one step and next, we will select the orientation from that selected sensor. For selecting the sensor, we can use the heuristic of total maximum or minimum number of conflicts. We can also select that sensor which covers the maximum lonely targets (4.3). If we select the sensors with the total maximum conflict, we are actually resolving the conflicts among the conflicted sensors which will allow the remaining sensors involved in the conflict to focus more on the lonely targets. Again if the total number of targets covered by a sensor is much more than the number of conflicts, then the sensor is covering mostly the lonely targets. If we select those sensors, this will be more likely target-oriented. After selection of the sensor, for selecting the orientation of that sensor we can greedily take that orientation which covers the most targets. Then the selection process of orientation will be in sensor-oriented greedy

approach. When we have the conflict graph, we can apply more different heuristics to find out better solutions for different scenarios.

4.2.1 Total Max Conflict Heuristics (TMxCH)

In Algorithm 1 we show how to implement TMxCH from a given scenario. The basic assumption is that both sensors and targets are static. At first, a *conflict graph* is developed from the given co-ordinates of sensors and targets using the coverage parameters of the sensors namely, Range (R_s) and AoV (θ) with the help of **TIS** test. At each iteration, firstly, an unused sensor is selected that is causing the maximum conflict with other sensors. Then, the orientation of that selected sensor in which it covers maximum number of uncovered targets is selected. At the end of an iteration, the *conflict graph* is updated (see 4.3).

In Algorithm 1 we show how to implement Total Max Conflict Heuristics (TMxCH).

Algorithm 1 Total Max Conflict Heuristics (TMxCH)

Require: C = set of unused cameras; T = set of uncovered targets; P = set of discrete pans
Ensure: Z = set of sensor-orientation pairs given by TMxCH

model the conflict graph, $G = (V, E)$ where V = {set of nodes (sensors)}; E = {set of weighted edges}
while $C \neq \phi$ **or** $T \neq \phi$ **do**
  for each $c_i \in C$ **do**
    sum_i = sum of conflicts with all $p_k \in P$ of $c_j \in C$ for $\forall j \neq i$
  end for
  $c_i = max(sum_i \; \forall \; c_i \in C)$ {select the sensor with maximum conflicts}
  for each $p_k \in P \; of \; c$ **do**
    $p_k = max(|t_k|)$ {find the orientation covering maximum targets}
  end for
  τ = set of targets covered by $\langle c_i, p_k \rangle$ sensor-orientation pair
  $C = C \backslash \{c_i\}$; $T = T \backslash \{\tau\}$
  $Z = Z \; \cup \; \{\langle c_i, p_k \rangle\}$
  $V^{'} = V \backslash \{c_i\}$
  $E^{'} = E \backslash$ {set of edges connected to c_i}
  update weight of the edges of the neighboring nodes of c_i
  $G = (V^{'}, E^{'})$
  if $t_{ik} = \phi \; \forall \; p_k \in P \; of \; c_i \in C$ **then**
    $break$
  end if
end while

4.2.2 Total Min Conflict Heuristics (TMnCH)

In Algorithm 2 we similarly form the *conflict graph*. The only change from the TMxCH is that while selecting the sensor, we choose the one causing minimum number of conflicts with other sensors. Finally, the *conflict graph* is updated at the end of each iteration in the same way as well.

In Algorithm 2 we show how to implement Total Min Conflict Heuristics (TMnCH).

Algorithm 2 Total Min Conflict Heuristics (TMnCH)

Require: C = set of unused cameras; T = set of uncovered targets; P = set of discrete pans
Ensure: Z = set of sensor-orientation pairs given by TMnCH
1: model the conflict graph, $G = (V, E)$ where V = {set of nodes (sensors)}; E = {set of weighted edges}
2: **while** $C \neq \phi$ **or** $T \neq \phi$ **do**
3: **for** each $c_i \in C$ **do**
4: sum_i = sum of conflicts with all $p_k \in P$ of $c_j \in C$ for $\forall j \neq i$
5: **end for**
6: $c_i = min(sum_i \; \forall \; c_i \in C)$ {select the sensor with minimum conflicts}
7: **for** each $p_k \in P \; of \; c$ **do**
8: $p_k = max(|t_k|)$ {find the orientation covering maximum targets}
9: **end for**
10: τ = set of targets covered by $\langle c_i, p_k \rangle$ sensor-orientation pair
11: $C = C \backslash \{c_i\}; \; T = T \backslash \{\tau\}$
12: $Z = Z \; \cup \; \{\langle c_i, p_k \rangle\}$
13: $V^{'} = V \backslash \{c_i\}$
14: $E^{'} = E \backslash$ {set of edges connected to c_i}
15: update weight of the edges of the neighboring nodes of c_i
16: $G = (V^{'}, E^{'})$
17: **if** $t_{ik} = \phi \; \forall \; p_k \in P \; of \; c_i \in C$ **then**
18: $break$
19: **end if**
20: **end while**

4.2.3 Total Max Conflict Heuristics with Shadow Edge (TMxCHSE)

In Algorithm 3 from the given *conflict graph with shadow edges* we check for any valid shadow edge. If there is a shadow edge, it means that there are targets which are covered by only one sensor in a particular orientation. We call these targets *non-conflicting or lonely* targets. First, these targets are covered. When all the *lonely* targets are covered, the algorithm follows TMxCH.

In Algorithm 3 we show how to implement Total Max Conflict Heuristics with Shadow Edge (TMxCHSE).

Algorithm 3 TMxCH with Shadow Edge (TMxCHSE)

Require: C = set of unused cameras; T = set of uncovered targets; P = set of discrete pans
Ensure: Z = set of sensor-orientation pairs given by TMxCHSE

model the conflict graph, $G = (V, E, S)$ where V = {set of nodes (sensors)}; E = {set of weighted edges}; S = {set of shadow edges}
while $C \neq \phi$ **or** $T \neq \phi$ **do**
  if $S \neq \phi$ **then**
    $\langle c_i, p_k \rangle$ = the sensor-orientation pair covering most non-conflicting targets
  else
    for each $c_i \in C$ **do**
      sum_i = sum of conflicts with all $p_k \in P$ of $c_j \in C$ for $\forall j \neq i$
    end for
    $c_i = max(sum_i \; \forall \; c_i \in C)$ {select the sensor with maximum conflicts}
    for each $p_k \in P \; of \; c$ **do**
      $p_k = max(|t_k|)$ {find the orientation covering maximum targets}
    end for
  end if
  τ = set of targets covered by $\langle c_i, p_k \rangle$ sensor-orientation pair
  $C = C \backslash \{c_i\}; \; T = T \backslash \{\tau\}$
  $Z = Z \; \cup \; \{\langle c_i, p_k \rangle\}$
  $V^{'} = V \backslash \{c_i\}$
  $E^{'} = E \backslash$ {set of edges connected to c_i}
  $S^{'} = S \backslash$ {set of shadow edges connected to c_i}
  update weight of the edges of the neighboring nodes of c_i
  $G = (V^{'}, E^{'}, S^{'})$
  if $t_{ik} = \phi \; \forall \; p_k \in P \; of \; c_i \in C$ **then**
    $break$
  end if
end while

4.3 Updating conflict graph

It is already known that which sensor is selected and which targets are covered at this iteration. The selected sensor is removed from the set of unused sensors and also the node representing that sensor in the *conflict graph* is removed from the graph. The tricky part comes afterwards. This removal of a sensor (node) from the current graph has an impact on the new *conflict graph* which will be passed on to the next iteration. In all the existing literatures, for removing the targets which are covered at any iteration, it was required to traverse through all the unused sensors since there was no efficient data structures which could tell that the sensors which had the same targets in their different pans. So, the complexity of this removal process was of $O(n)$. But, we have the *conflict graph*. It holds the information of overlapping coverage for a target between two sensors with the help of an edge between them. So, when a sensor is getting selected, to update the graph for the next iteration, we only need to check the neighboring nodes (sensors). If there is no edge between two sensors, it means that they have no overlapping targets

in any of their pans. That is why removing any target under the coverage of any one of them will have no impact to the edges associated with the other one. We only update the weight of the edges of the neighboring nodes (sensors), when we remove a node (sensor) from the graph. This reduces the time complexity to $O(\Delta)$ (Δ means the average number of neighbors of a node in the *conflict graph*).

Chapter 5

Comparison of Different Heuristics

In this chapter, we will give a comparative analysis among the existing heuristics and one of our proposed heuristics with the help of an example scenario.

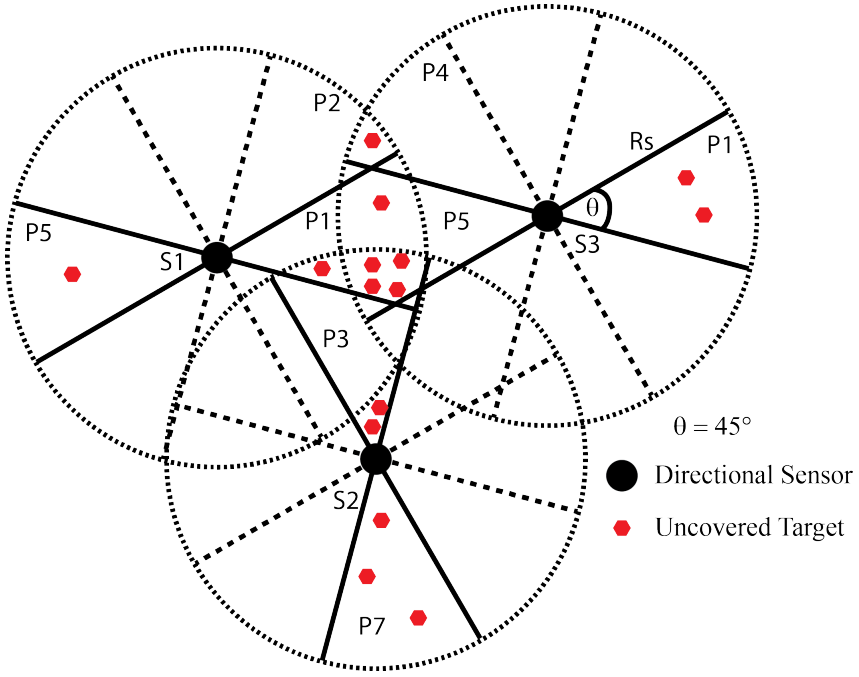


Figure 5.1: An example scenario for the comparison of different heuristics

Here, three sensors s_1, s_2, and s_3 are deployed. The sensing range is R_s. AoV is fixed at 45 degree. So, each sensor has eight disjoint pans. Total fifteen targets are randomly deployed. The size of the coverage set of each sensor-orientation pair is as follows:

- $\langle s_1, p_1 \rangle = 6$
- $\langle s_1, p_2 \rangle = 1$
- $\langle s_1, p_5 \rangle = 1$

- $\langle s_2, p_3 \rangle = 7$
- $\langle s_2, p_7 \rangle = 3$
- $\langle s_3, p_1 \rangle = 2$
- $\langle s_3, p_4 \rangle = 1$
- $\langle s_3, p_5 \rangle = 5$

5.1 Greedy Solution

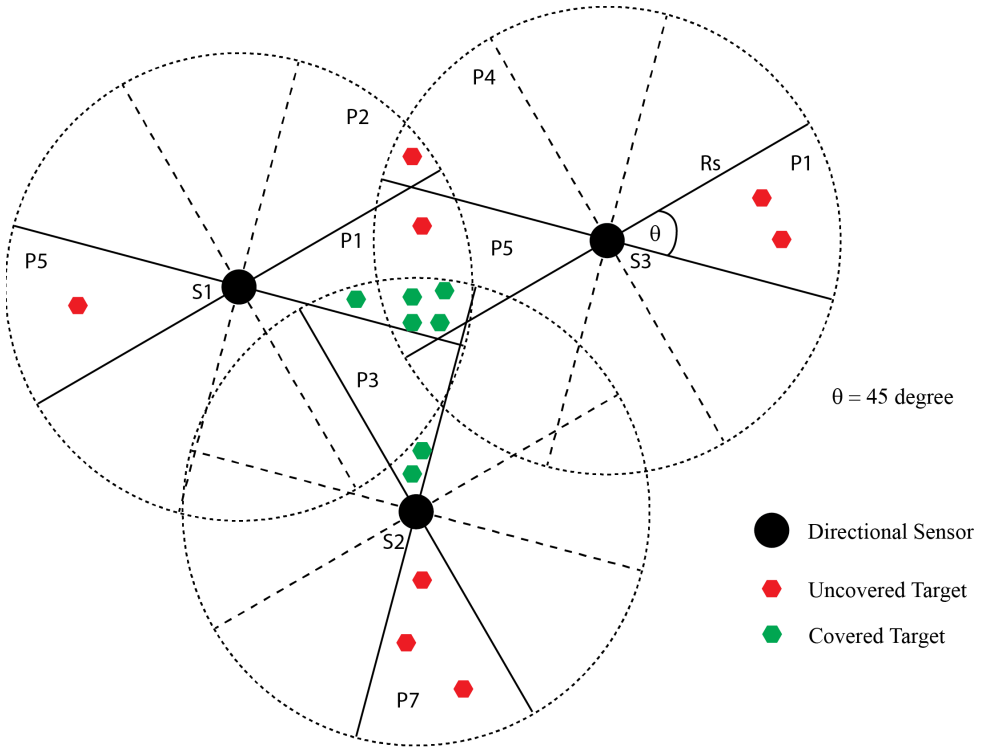
In greedy approach, sensor-orientation pair covering the maximum number of targets is chosen at each iteration. Here, at figure 5.2a we can see that $\langle s_2, p_3 \rangle$ covers maximum 7 targets. So, it is selected first. In the next iteration, we are left with sensors, s_1 and s_3. Sensor-orientation pairs, $\langle s_1, p_1 \rangle$, $\langle s_1, p_2 \rangle$, $\langle s_1, p_5 \rangle$ and $\langle s_3, p_5 \rangle$ cover 1 target each and $\langle s_3, p_1 \rangle$ covers 2 targets. So, at figure 5.2b, we can see that $\langle s_3, p_1 \rangle$ is selected. At the last iteration, we are only left with sensor, s_1. It covers a single target in pans, p_1, p_2 and p_5 each. So, any of the pans can be selected. In figure 5.2c, we see that $\langle s_1, p_1 \rangle$ is selected. So, following greedy heuristic, in the given scenario of figure 5.1 in total 10 targets are covered by the 3 sensors out of the 15 targets.

5.2 Target-oriented Solution

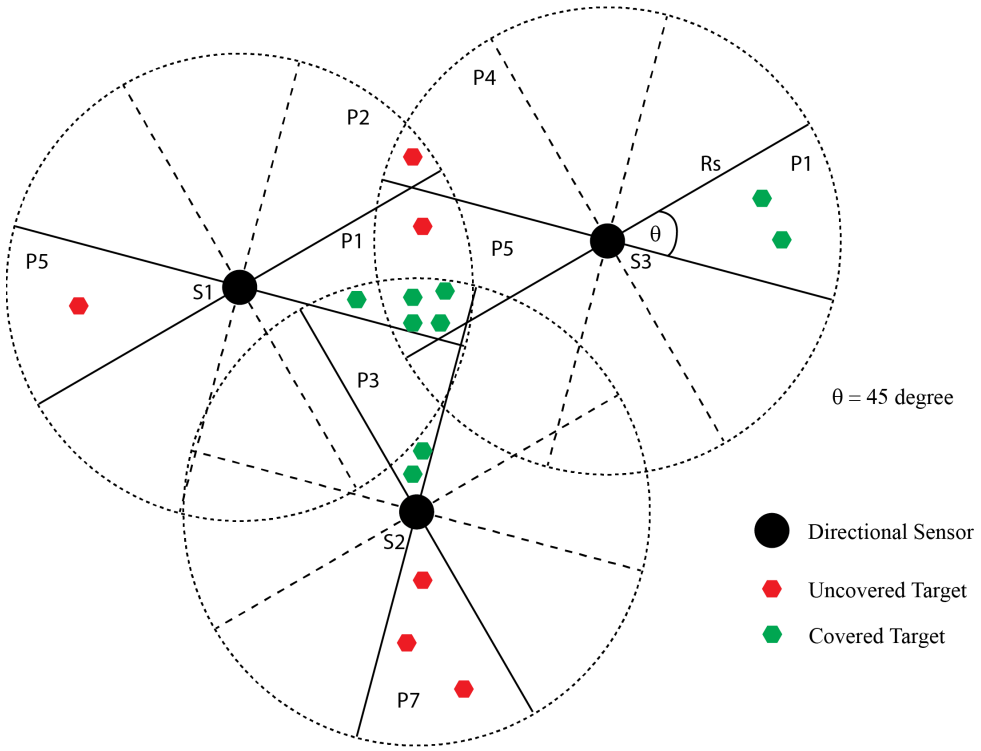
In target-oriented approach, selection of a sensor is done from the point of view of a target. Here, priority is given to lonely targets, the targets which are covered by a single sensor. In figure 5.1, we can see that maximum 3 lonely targets are in $\langle s_2, p_7 \rangle$. So, it is selected first. Then $\langle s_3, p_1 \rangle$ has maximum of 2 lonely targets and lastly, $\langle s_1, p_5 \rangle$ has a single lonely target. So, from figure 5.3, we can see that total 6 targets are covered by the 3 sensors out of the 15 targets.

5.3 Conflict Graph: Using TMxCH

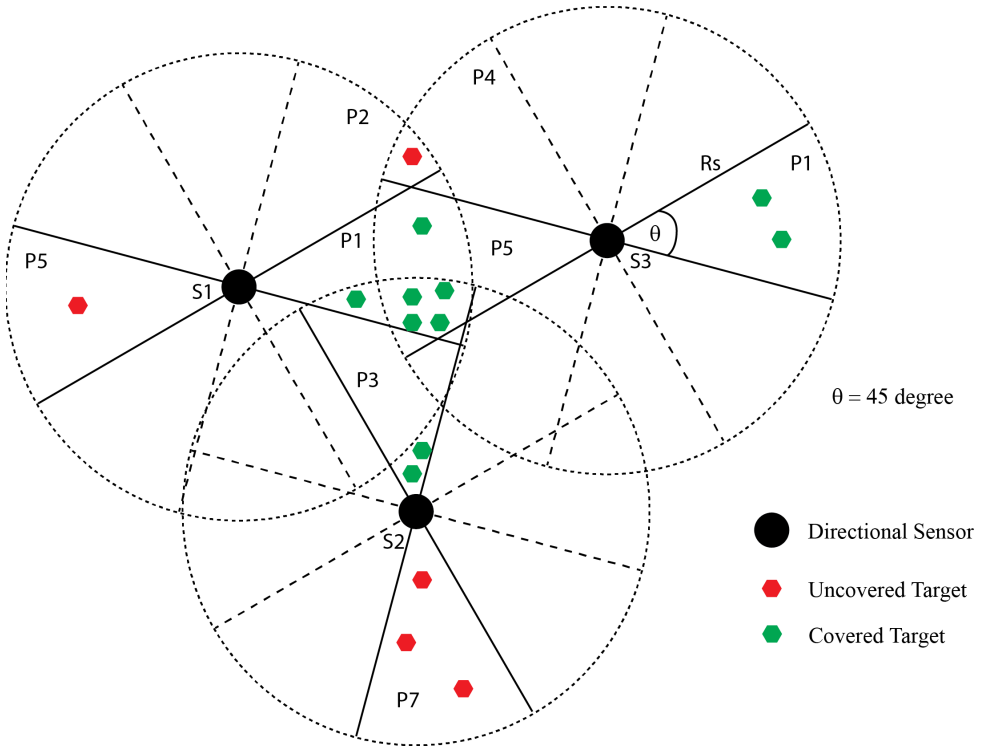
At first, we generate the *conflict graph* of figure 5.4 using the scenario of figure 5.1. From the *conflict graph*, we see that sensor, s_1 has total 11 conflicted targets with the other two sensors. Sensor, s_2 and s_3 have 9 and 10 conflicted targets respectively. If we use algorithm 1, sensor, s_1 will be selected first since it has the maximum conflicted targets. Now among the pans of sensor, s_1, p_1 covers the maximum of 6 targets. So, at the first iteration, sensor-orientation pair, $\langle s_1, p_1 \rangle$ will be selected as seen in figure 5.5a. After this the *conflict graph* is updated by removing the already used sensor, s_1 and targets covered by its pan, p_1. After the first iteration,



(a) Iteration 1



(b) Iteration 2



(c) Iteration 3

Figure 5.2: Greedy approach

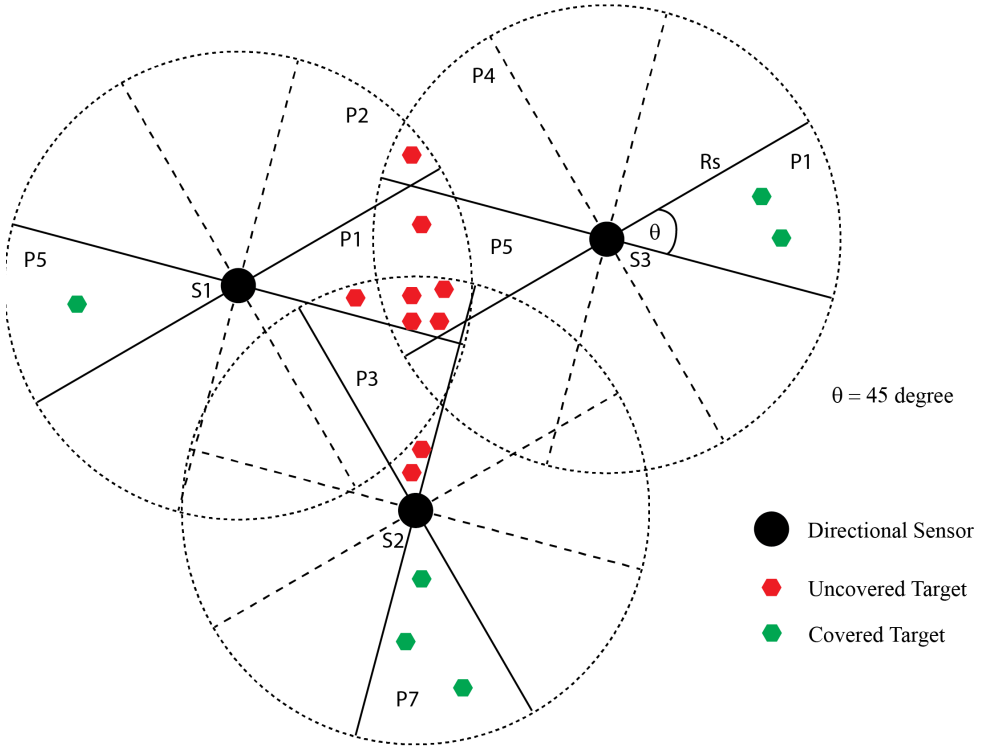


Figure 5.3: Target-oriented approach

the remaining two sensors, s_2 and s_3 have no conflicted targets between them. So, these two sensors can be selected greedily without any requirement of *conflict graph*. From figure 5.5b, we can see that $\langle s_2, p_7 \rangle$ and $\langle s_3, p_1 \rangle$ covering 3 and 2 targets respectively are selected at next two iterations. By using this *conflict graph heuristic*, we achieve a coverage of 11 targets using the 3 sensors out of 15 targets.

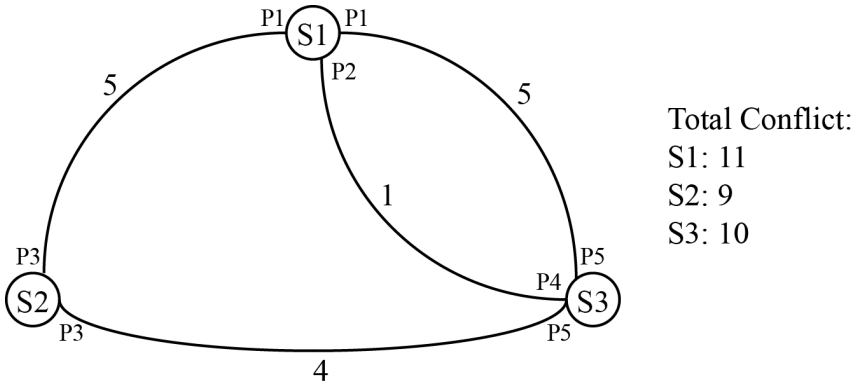


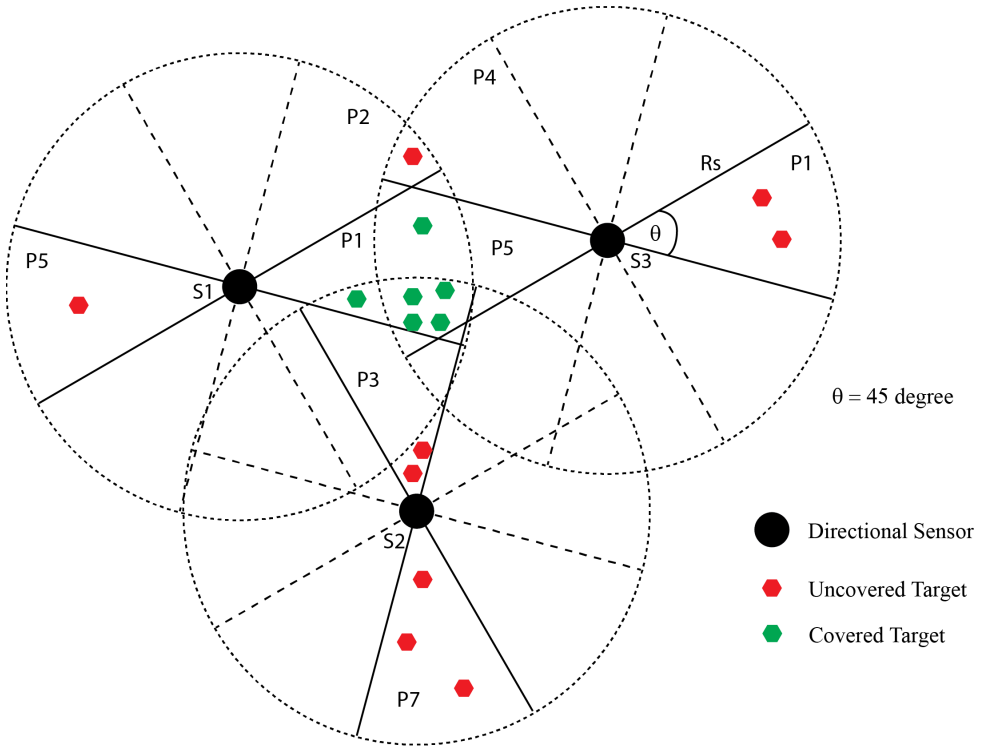
Figure 5.4: Conflict graph of the example scenario

5.4 Performance Comparison

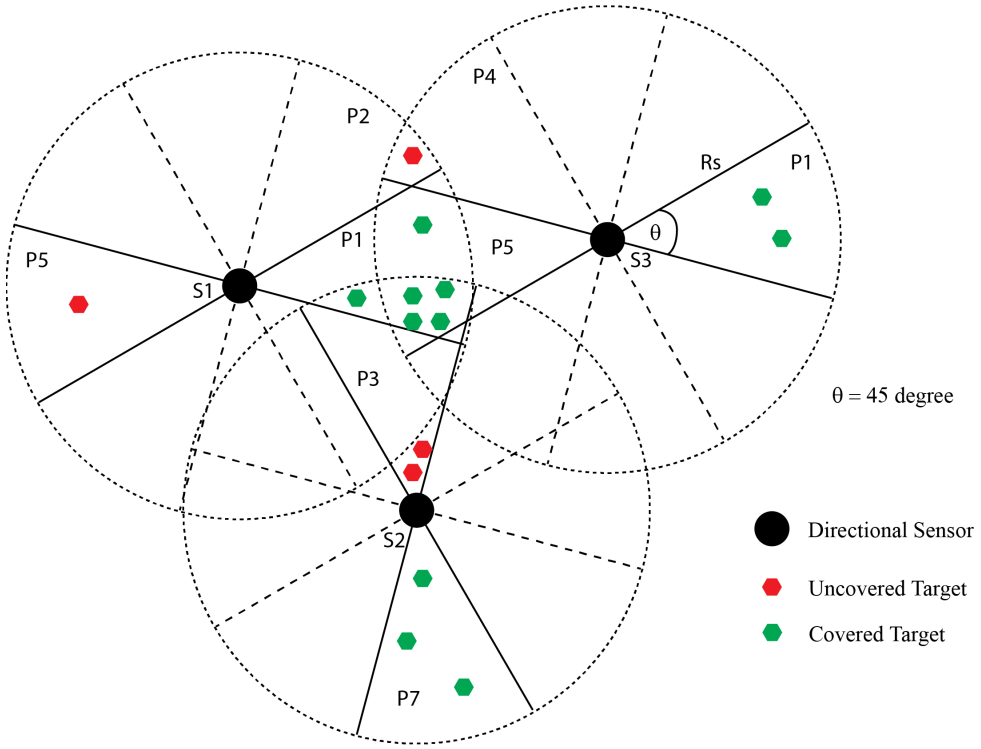
Now, let us compare the performance of the aforesaid three heuristics on the same example scenario of figure 5.1. We can see that one of the *conflict graph heuristic* that is TMxCH has performed the best among them. It covered 11 targets whereas greedy could cover 10 and target-oriented heuristic could cover only 6 targets out of a total of 15 targets. From this, we can infer that in a scenario with higher ratio of conflicted targets, *conflict graph heuristic* can out-perform all other existing heuristics like CGA [1], PTOH [8], etc.

Table 5.1: Comparing the performance of different heuristics

Heuristic	Coverage (out of 15 targets)
TMxCH	11
Greedy	10
Target-oriented	6



(a) Iteration 1



(b) Iteration 2 and 3

Figure 5.5: Conflict graph approach: using TMxCH

Chapter 6

Performance Evaluation

In this chapter we compare the performance of different *conflict graph heuristics* (CHs) with centralized greedy algorithm (CGA) of Ai and Abouzeid (2006) [1] through various simulations.

Table 6.1: Acronyms used in the plots

Acronym	*Actual term*	*Reference to work*
CGA	Centralized Greedy Algorithm	Ai and Abouzeid [1]
TMnCH	Total Minimum Conflict Heuristics	This thesis
TMxCH	Total Maximum Conflict Heuristics	This thesis
TMxCHSE	TMxCH with Shadow Edge	This thesis

6.1 Simulation Environment

We consider a fixed area of 1000×1000 square unit, where N cameras and M targets were randomly deployed. Position of cameras and targets are defined as 2-D points (x, y) in Cartesian co-ordinate system. The sensing range, R_s is assumed to be 100 unit, and the Angle Of View (AoV) is set to $\theta = \frac{\pi}{4}$. With an AoV of $\frac{\pi}{4}$, the number of discrete pans, Q becomes 8 ($\frac{2\pi}{\frac{\pi}{4}} = 8$).

6.1.1 Scenario generation

We generate two types of static scenarios: (i) sensor-only scenarios, and (ii) target-only scenarios.

In order to generate sensor-only scenarios, we vary the number of sensors over a fixed deployment area of 1000×1000 square unit. Number of sensors ranges from N=25 to N=400 in different scenarios with an increment of 25 sensors from one scenario to the other. As we

increase the number of sensors, the larger set includes the sensors from the previous scenario. This ensures a consistent evaluation of the impact of the enlarged population of cameras by retaining all the features of the previous environment and simply making it better.

In order to generate target-only scenarios, 25225 targets were randomly spread over a 1000×1000 square region with an increase of 25 targets at each step. Similar strategy of including targets from the smaller sets to the larger sets was adopted while generating the target-only scenarios. Finally, we combine one sensor-only scenario (for a certain N) with one target-only scenario (for a certain M) to create a visual sensor network (VSN). Then the proposed heuristics along with the existing algorithm are simulated over these VSNs. For each VSN size (i.e., for a certain N and M), we have generated 5 instances. Performance measures are reported as an average of these 5 random VSNs (unless explicitly stated otherwise).

We have adopted two different approaches to capture the performance during simulation: (i) by varying the number of sensors N while keeping the number of targets M fixed, and (ii) by varying the number of targets M while keeping the number of sensors N fixed.

6.2 Performance Metrics

We analyze the performance of the different VSNs using three performance criteria: *coverage ratio* (CR), *ratio of active sensors* (RAS), and *target coverage per sensor* (TCPS). These performance metrics are defined as follows:

6.2.1 Coverage Ratio (CR)

CR is the fraction of targets covered by the selected sensors after running an algorithm, i.e.:

$$CR = \frac{Number\ of\ targets\ covered}{Total\ number\ of\ targets} \tag{6.1}$$

We assume that the Optimal Solution covers all the targets, the metric CR indicates the proximity of performance of a heuristic with respect to the optimal solution. The CR of the Optimal Solution is always 1. For any heuristic, the higher value of CR indicates that the performance of that heuristic is better.

6.2.2 Ratio of Active Sensors (RAS)

RAS is the fraction of sensors selected by an algorithm:

$$RAS = \frac{Number\ of\ sensors\ activated}{Total\ number\ of\ sensors} \tag{6.2}$$

While CR is a measure of target coverage, RAS is a measure of sensor usage. The *smaller* the ratio, the better is the performance.

6.2.3 Target Coverage Per Sensor (TCPS)

TCPS is the ratio of the number of covered targets and the number of sensors activated, i.e.:

$$TCPS = \frac{Number\ of\ targets\ covered}{Number\ of\ sensors\ activated} \tag{6.3}$$

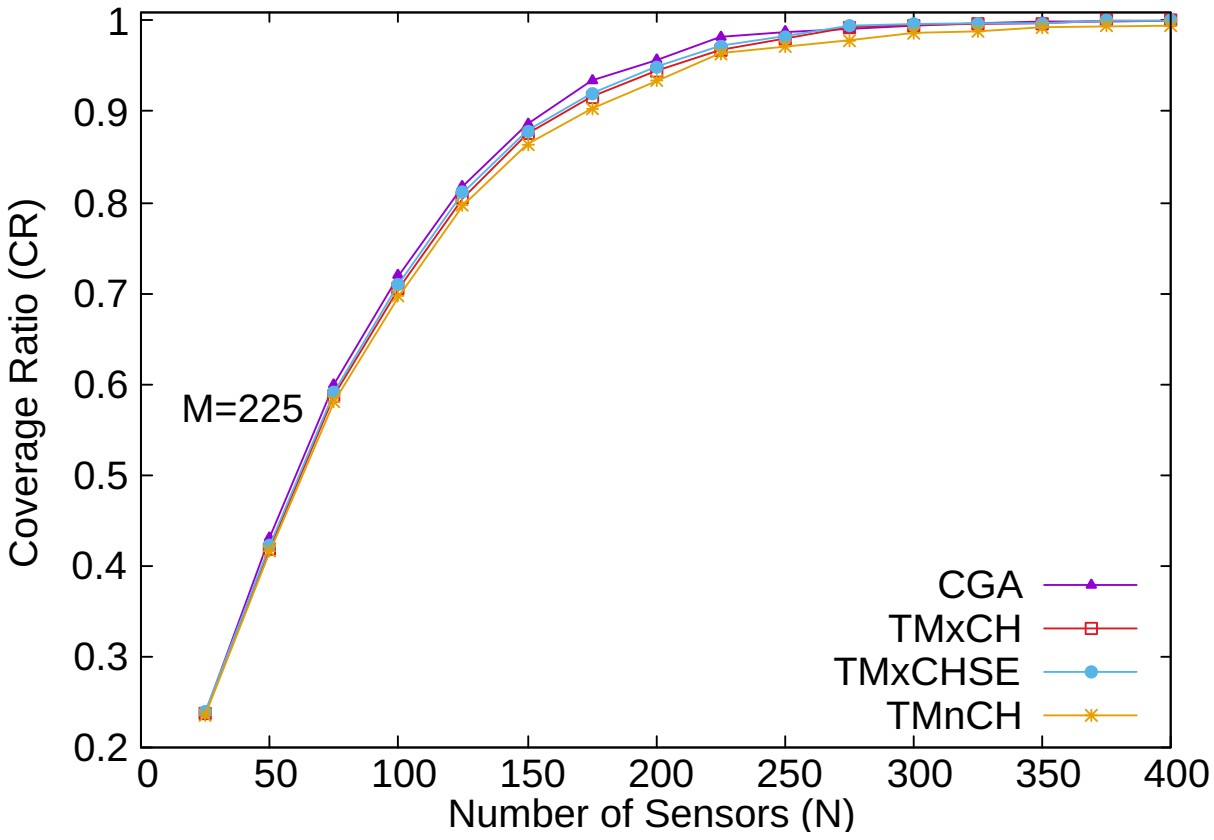
It is to mention that TCPS provides average number of targets covered per sensor. This is an important metric since a particular algorithm may achieve higher RAS value activating a large number of sensors or lower CR value by not covering all the targets. Thus, the better performance of one metric may not reflect the poorer performance of the other metric. Considering this fact, TCPS is introduced which can combine the effect of both sensor usage and target coverage (i.e., considers CR and RAS simultaneously).

6.3 Results

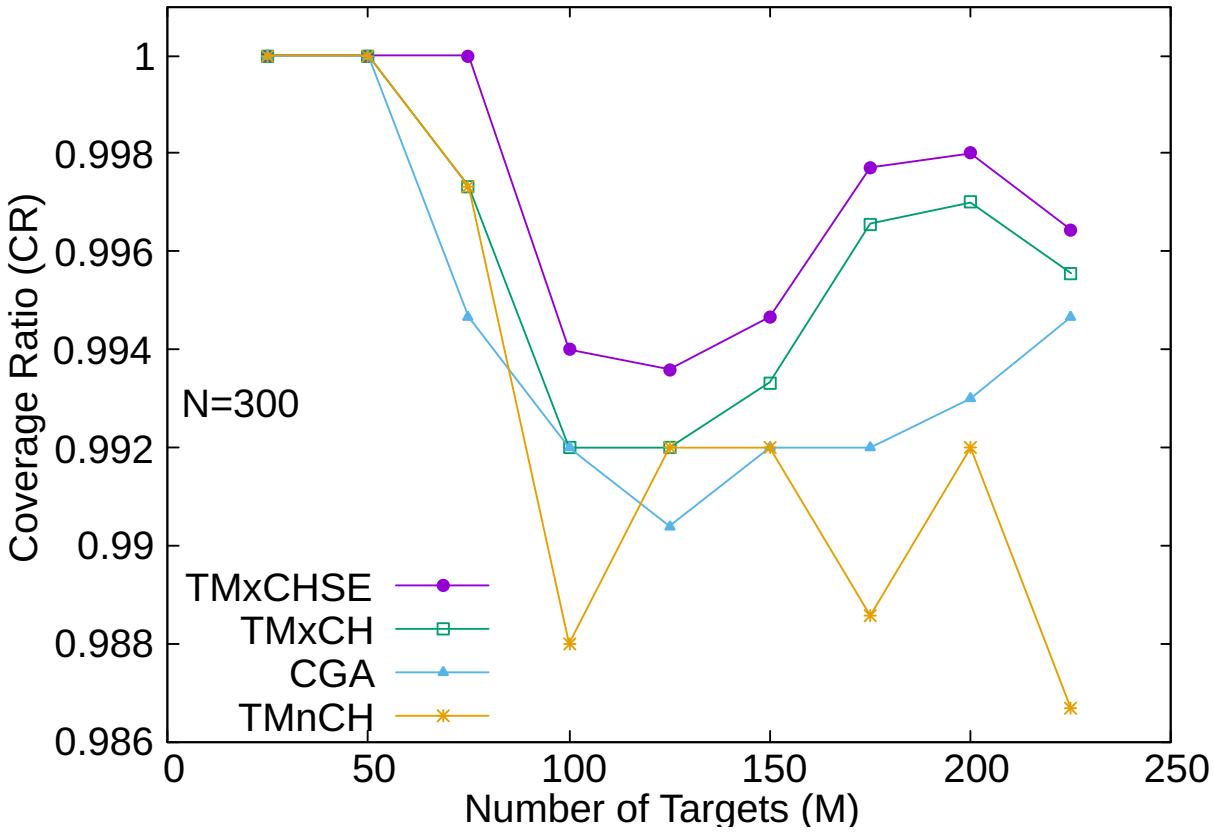
In this section we provide the performance comparison graphs for each of the proposed *conflict graph heuristics* (CHs) along with centralized greedy algorithm (CGA) of Ai and Abouzeid (2006) [1]. Table 6.1 summarizes all the acronyms used in the plots.

6.3.1 Target Coverage

At first, we measure the coverage ratio by keeping the number of targets fixed at M=225 and varying the number of sensors N from 25 to 400. The result is shown in figure 6.1a. All the algorithms converge to the optimal coverage when number of sensors becomes high enough to cover all 225 targets, i.e., the system moves into an over-provisioned one. This happens when the number of cameras crosses 300. Next we plot CR in figure 6.1b by keeping the number of sensors fixed at N=300 and varying the number of targets M from 25 to 225. As the number of sensors is lot more than the number of targets, clearly the system is over-provisioned and



(a)



(b)

Figure 6.1: Coverage performance comparison of CGA, TMnCH, TMxCH and TMxCHSE heuristics

all approaches have enough sensor to cover all the targets and thereby nearly converge to the optimal coverage.

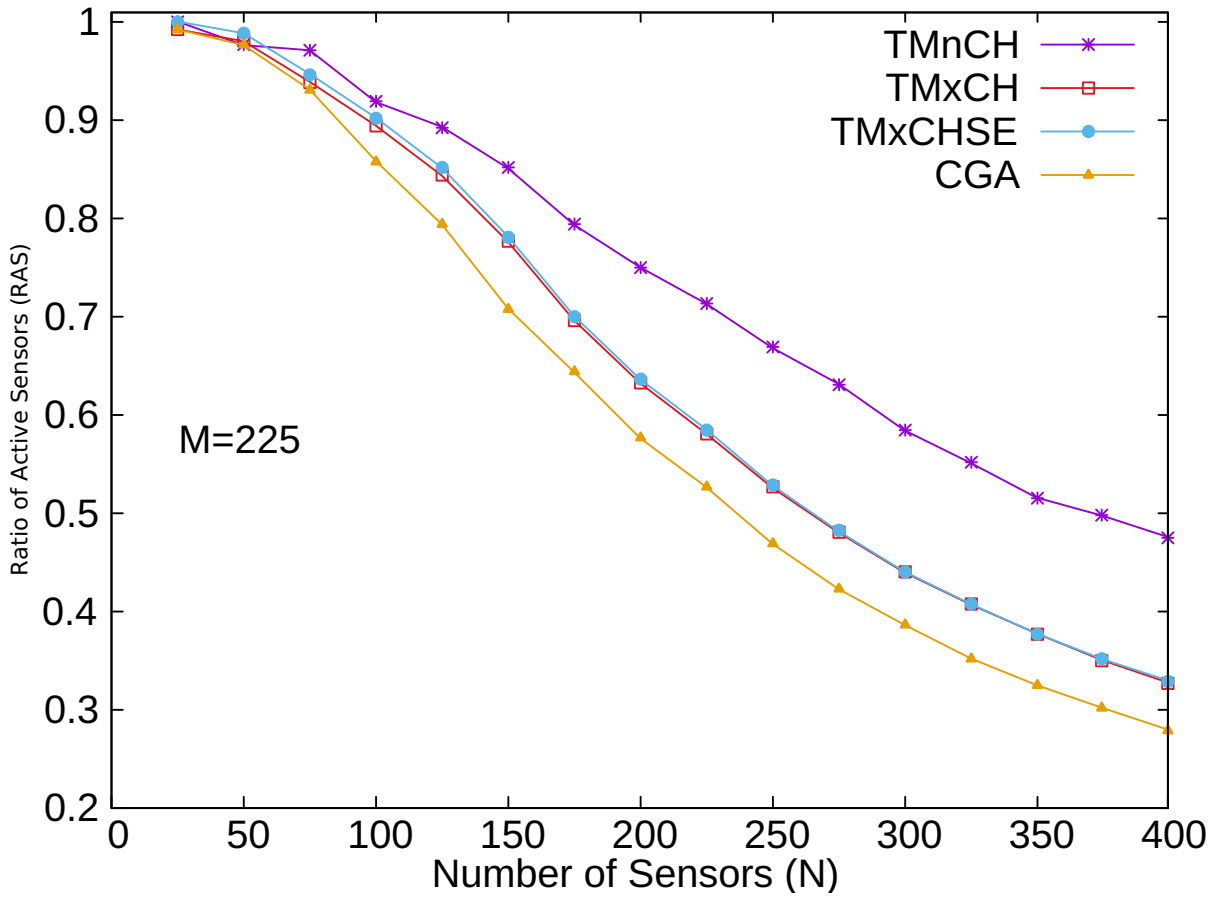
In figure 6.1a, all the algorithms perform very close to each other. However, CGA has better CR values when the number of sensors are less w.r.t. the number of targets that is, the system is under-provisioned. But gradually, TMxCHSE and TMxCH out-perform CGA when the system becomes over-provisioned. TMnCH has the worst performance among all. It works like a pure target-oriented algorithm. It would perform better when the ratio of lonely targets is very high in the VSN. TMxCH is the opposite of TMnCH. It selects the sensor involved in the most conflicts at first. That is, it tries to resolve conflicting targets first and conflicting targets are just the opposite of lonely targets. TMxCHSE combines the benefits of both TMxCH and TMnCH. While selecting a sensor, it takes both the number of conflicting and non-conflicting targets into account and chooses the best option between the two.

In figure 6.1b, all the heuristics give almost optimal coverage. Here, the number of sensors is fixed at 300 and the number of targets are varied. TMxCHSE performs the best here as well. TMxCH, TMnCH and CGA follow it respectively. It is to observe that with the increase of the number of targets, the system becomes less over-provisioned and CGA starts to perform slightly better than TMnCH. But still, TMxCHSE and TMxCH remains above CGA in CR values.

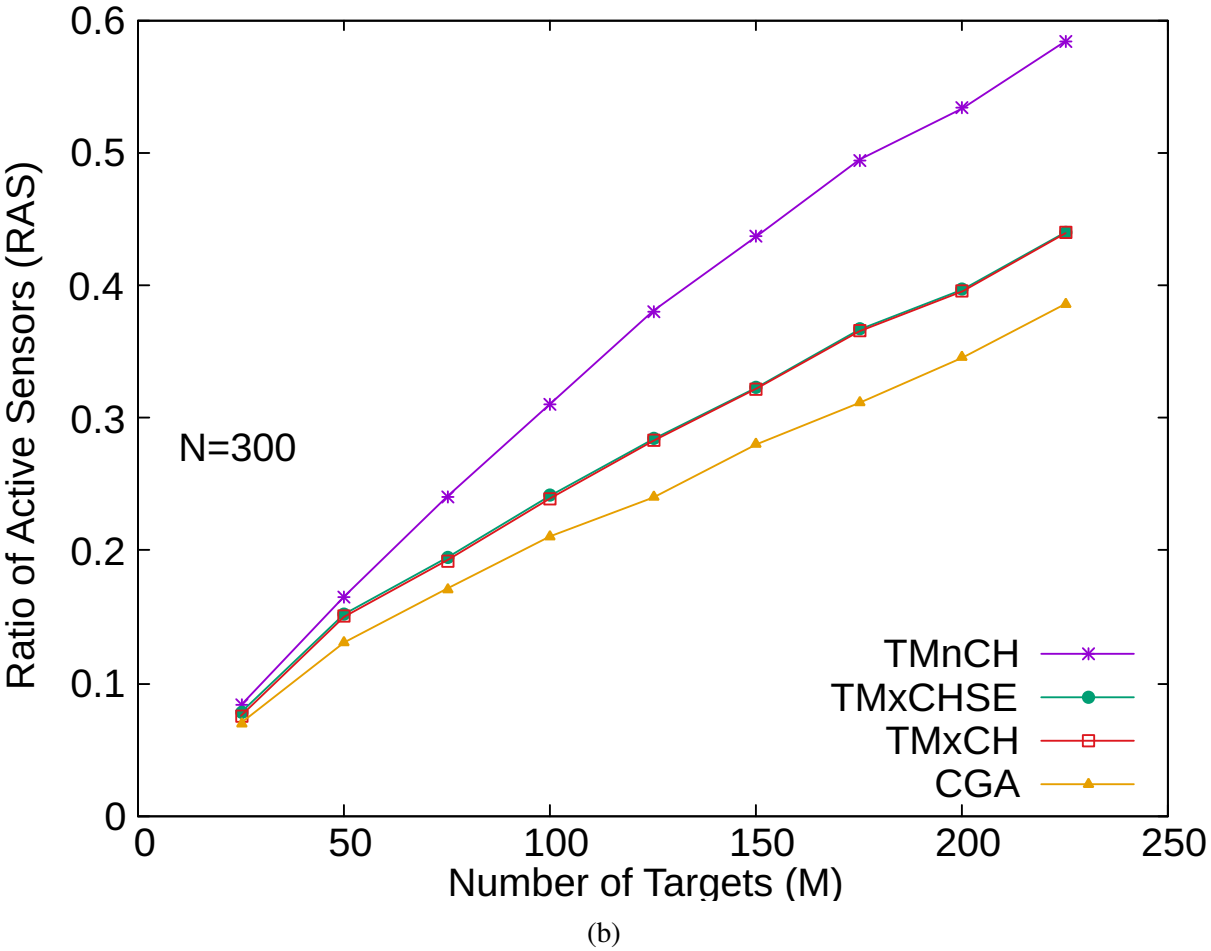
6.3.2 Sensor Usage

The ratio of active sensor (RAS) values are shown in figure 6.2 where we vary both the number of sensors and the number of targets. In figure 6.2a the number of targets M is kept fixed at 225 and the number of sensors N varies from 25 to 400. As we increase N the system approaches to an over-provisioned one. In terms of sensor usage, CGA of Ai and Abouzeid (2006) [1] performs close to optimal as it greedily activates sensors but does not cover all of the targets. Next better performance is given by TMxCH. TMxCHSE performs almost similar to TMxCH, but it activates a little bit more sensors than TMxCH. We can allow this since TMxCHSE gives better coverage than TMxCH. TMnCH uses more sensors than TMxCHSE. Actually, TMnCH works in a target-oriented approach where it selects the sensor with least conflict first. In other words, TMnCH selects the sensors covering more lonely targets at first. That is why it activates more sensors than other heuristics to achieve the best coverage.

Figure 6.2b shows the RAS values when we keep the number of sensor N fixed at 300 and vary the number of targets from 25 to 225. When we increase the number of targets while keeping the number of sensors fixed the system becomes less over-provisioned and the ratio of active sensors linearly increases. The performance comparison of different heuristics shows similar behavior as of figure 6.2a.



(a)



(b)

Figure 6.2: Sensor usage comparison of CGA, TMnCH, TMxCH and TMxCHSE heuristics

6.3.3 Target Coverage Per Sensor

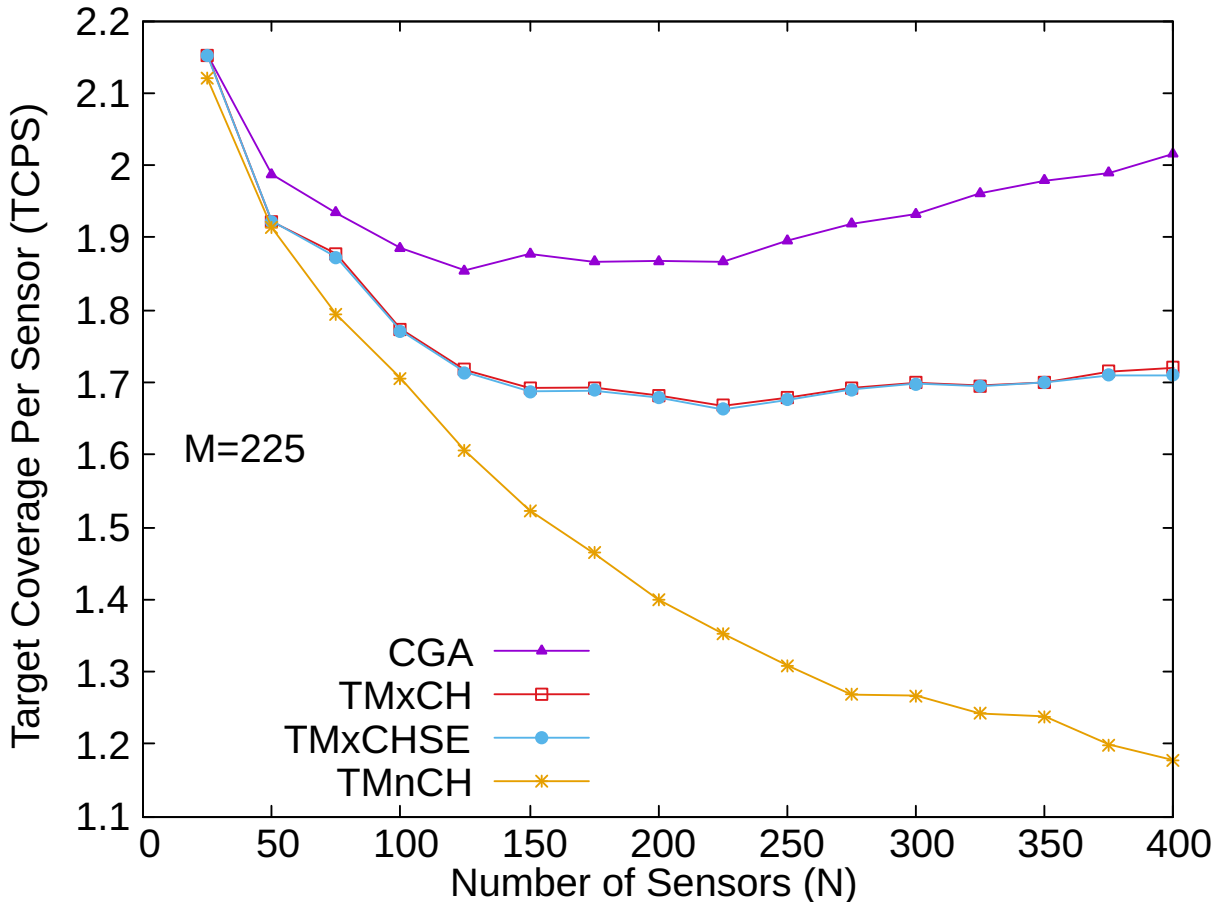
Finally, the *conflict graph heuristics* (TMxCHSE and TMxCH) come out as the best one's when we look at figure 6.3a and 6.3b. A question may arise since CGA shows higher values in TCPS plots. Though it covers more targets per sensor, it loses to both TMxCHSE and TMxCH in ensuring higher coverage to an over-provisioned VSN. From both of the figures, it is clearly evident that for different number of sensors, TCPS of TMxCHSE is convergent to the optimal as it provides higher coverage using fewer sensors. Between TMxCH and TMxCHSE, TMxCH activates less sensors as its RAS value is better as shown in figure 6.2a but it has poorer coverage w.r.t. TMxCHSE as shown in figure 6.1a. When we scale the coverage measure to camera usage measure, i.e., consider *coverage/camera usage* as the metric of interest (i.e., the TCPS), performance of TMnCH is not as good as TMxCHSE and TMxCH.

6.4 Run-time Analysis

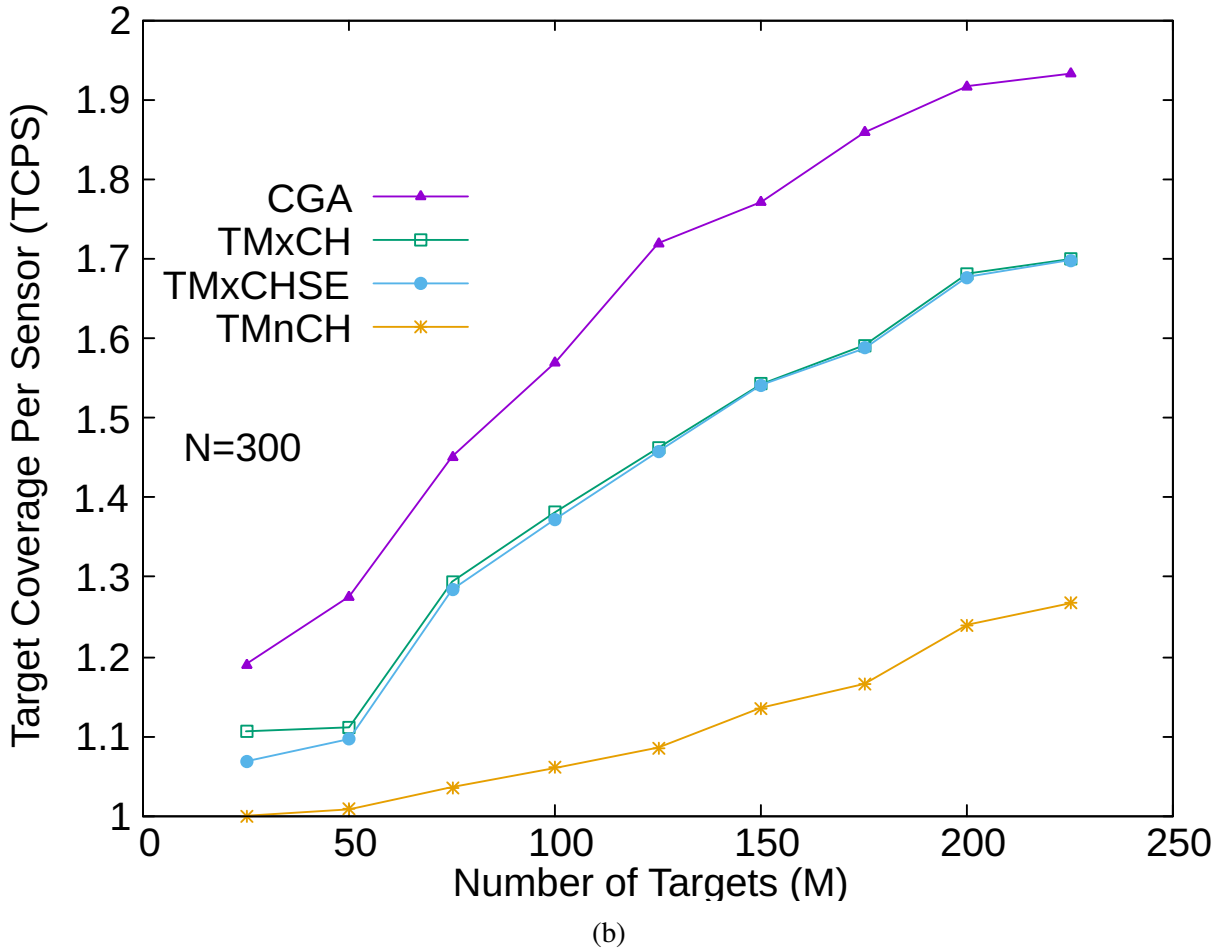
The simulation is run on a machine with both hardware and software specifications as follows:

- Hardware
  - CPU: core i5-7th Generation
  - RAM: 8GB DDR4
- Software
  - OS: windows10
  - Environment: Java Development Kit (JDK) 8.0

From the simulation we can find significant achievement in case of run-time of the heuristics developed for *conflict graph* model. Here, we have simulated by both varying the number of sensors (N) from 150 to 400 with an increment of 25 keeping the number of targets (M) fixed at 225 and varying the number of targets (M) from 25 to 225 with an increment of 25 keeping the number of sensors (N) fixed at 300. In the bar chart 6.4a, we see that with the increase of the number of sensors the run-time of CGA [1] increases following an exponential curve whereas TMxCH, TMnCH and TMxCHSE takes almost same run-time over the span of varying number of sensors. This result supports our claim that *conflict graph heuristics* (CHs) can very well localize the search at the step of removing the covered targets at each iteration. Since the average number of neighbors (Δ) of a node (sensor) in a uniformly distributed system of nodes (sensors) remains almost the same over the whole span of varying sensors, the run-time of the CHs also remains almost unchanged.

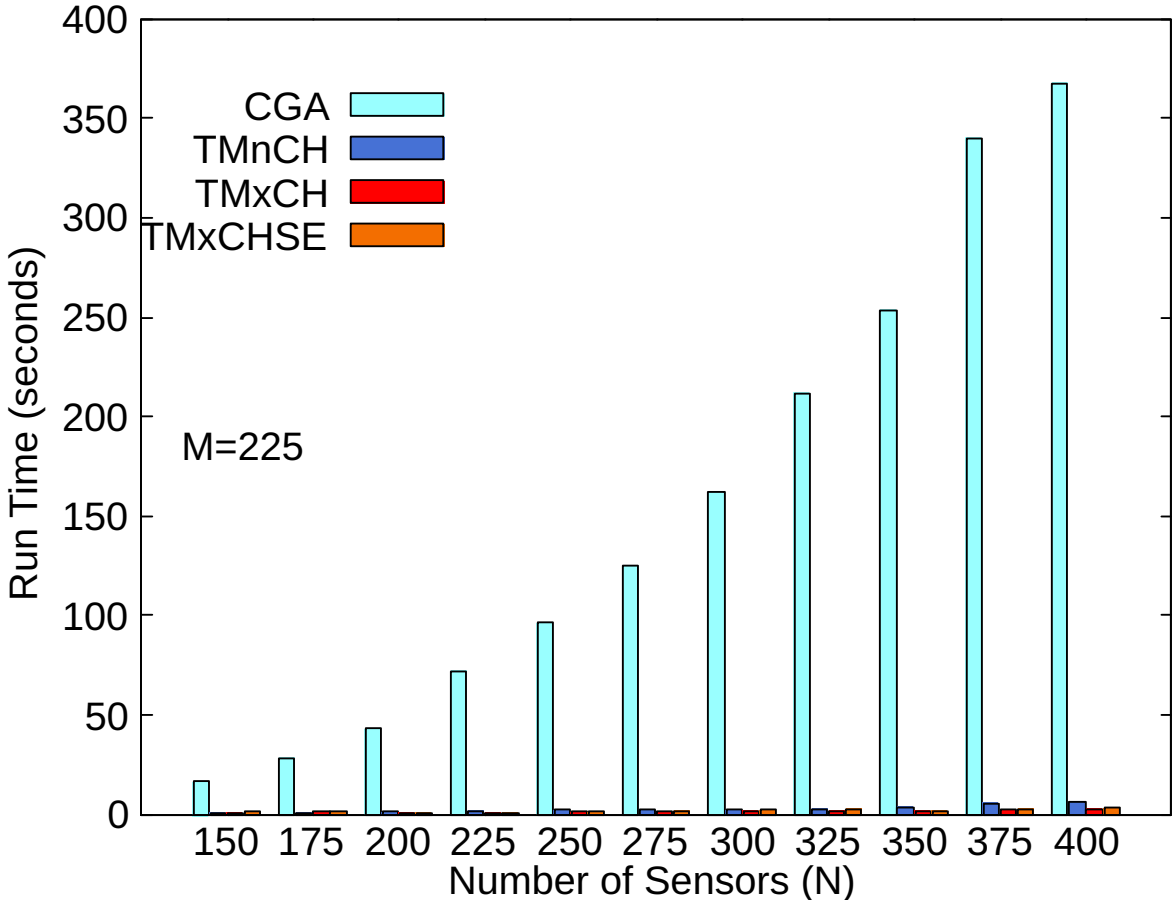


(a)

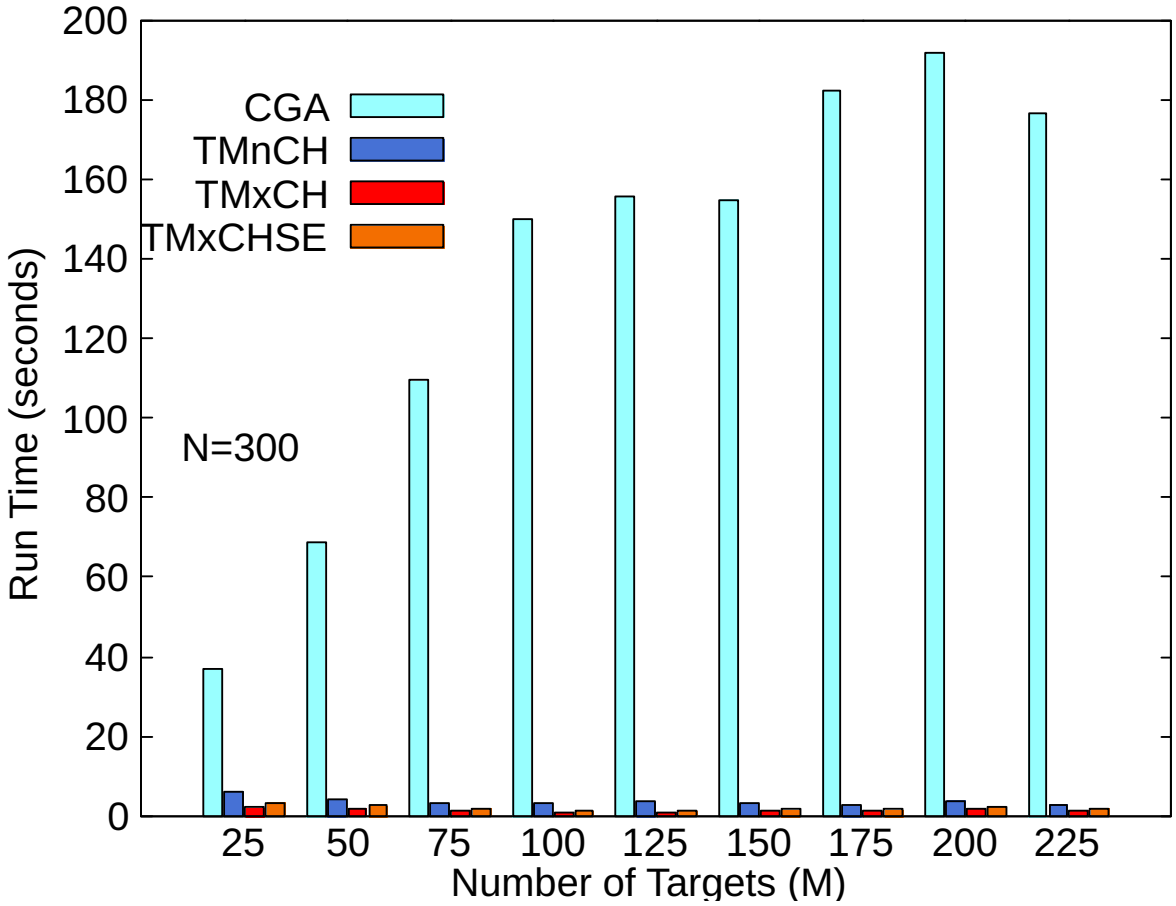


(b)

Figure 6.3: Target coverage per sensor (TCPS) comparison of CGA, TMnCH, TMxCH and TMxCHSE heuristics



(a)



(b)

Figure 6.4: Run-time comparison of CGA, TMnCH, TMxCH and TMxCHSE heuristics

We have got similar findings related to the run-time while varying the number of targets in bar chart 6.4b. Here, we see that CGA shows similar trend of what we have seen in 6.4a. But the run-time of the CHs slightly decreases with the increase of targets. The reason behind this decrement in run-time with the increment of the number of targets is that more the number of targets (M) more connected the graph model will be or there will be higher probability of conflicts among the sensors over target coverage. As the CHs perform better when there is higher amount conflicted targets, the run-time also decreases with better performance.

Finally, it can be stated that after evaluating on the basis of three different performance metrics and run-time, CHs can out-perform the existing heuristics. The greatest achievement is that CHs have kept the run-time checked very nicely in over-provisioned system and also provided better performance in target coverage.

Chapter 7

Conclusion

We have presented a graph theoretic approach to coverage problem of Visual Sensor Networks (VSNs) that provides near-optimal performance to achieve maximum target coverage using minimum number of sensors. We have formulate the problem using a *conflict graph* and describe our proposed heuristics along with their implementation. We have proposed three heuristics in this paper each of which tries to tackle the problem in different way. TMxCH can be interpreted as a sensor-oriented approach. TMnCH is quite the opposite of TMxCH. It attacks the problem in a target-oriented way by giving higher priority to the sensors involved in least conflicts. Lastly, TMxCHSE introduces the concept of shadow edge which enables us to look into the problem in both sensor-oriented and target-oriented ways at the same time. Then we have presented the performance evaluation of our heuristics varying the number of targets while keeping the number of sensors fixed and vice versa. We have simulated performance of CHs and CGA and taking Coverage Ratio (CR), Ratio of Active Sensors (RAS), and Target Coverage Per Sensor (TCPS) as metrics. The results show that CHs cover more targets achieving higher CR, and activating less number of sensors having lower values of RAS. It is also mentionable that CHs produce higher TCPS ratio in over-provisioned systems which supports our claim that *conflict graph heuristics* is a better solution for such systems.

There are yet some open problems to answer. For this study, we limit ourselves to *pan-only* sensors i.e., we allowed only horizontal movements of the sensors. Orientations in other dimensions (like tilt and zoom) are also possible. We only consider uniform distribution of the sensors and targets. It will be interesting to see how the proposed heuristics perform under other kinds of distribution systems. Our future work plan includes applying this graph theoretic approach to other variants of this problem; such as k-coverage [9], balanced k-coverage [12], heterogeneous k-coverage, etc.

References

[1] J. Ai and A. Abouzeid, "Coverage by directional sensors in randomly deployed wireless sensor networks," *Journal of Combinatorial Optimization*, vol. 11, no. 1, pp. 21–41, 2006.

[2] F. G. Yap and H. H. Yen, "A survey on sensor coverage and visual data capturing/processing/transmission in wireless visual sensor networks," *Sensors*, vol. 14, no. 2, pp. 3506–3527, 2014.

[3] J. Liu, S. Sridharan, and C. Fookes, "Recent advances in camera planning for large area surveillance: a comprehensive review," *ACM Computing Surveys*, vol. 49, no. 1, pp. 6:1–6:37, 2016.

[4] K. Chakrabarty, S. S. Iyengar, H. Qi, and E. Cho, "Grid coverage for surveillance and target location in distributed sensor networks," *IEEE Transactions on Computers*, vol. 51, no. 12, pp. 1448–1453, 2002.

[5] E. Hrster and R. Lienhart, "On the optimal placement of multiple visual sensors," in *Proceedings of the 4th ACM International Workshop on Video Surveillance and Sensor Networks*, pp. 111–120, ACM, 2006.

[6] V. P. Munishwar and N. B. Abu-Ghazaleh, "Coverage algorithms for visual sensor networks," *ACM Transactions on Sensor Networks*, vol. 9, no. 4, p. 45, 2013.

[7] V. P. Munishwar and N. B. Abu-Ghazaleh, "Target-oriented coverage maximization in visual sensor networks," in *Proceedings of the 9th ACM International Symposium on Mobility Management and Wireless Access*, pp. 175–178, ACM, 2011.

[8] H. Zannat, T. Akter, M. Tasnim, and A. Rahman, "The coverage problem in visual sensor networks: A target oriented approach," *Journal of Network & Computer Applications*, 2016.

[9] G. Fusco and H. Gupta, "Selection and orientation of directional sensors for coverage maximization," in *Proceedings of the 6th Annual IEEE Communications Society Conference on, IEEE Sensor, Mesh and Ad Hoc Communications and Networks*, pp. 1–9, IEEE, 2009.

[10] S. Lu and S. Yu, "A fuzzy k-coverage approach for rfid network planning using plant growth simulation algorithm," *Journal of Network & Computer Applications*, vol. 39, pp. 280–291, 2014.

[11] D. G. Costa, I. Silva, L. A. Guedes, P. Portugal, and F. Vasques, "Enhancing redundancy in wireless visual sensor networks for target coverage," in *Proceedings of the 20th Brazilian Symposium on Multimedia and the Web*, pp. 31–38, ACM, 2014.

[12] S. M. B. Malek, M. M. Sadik, and A. K. M. Rahman, "On balanced k-coverage in visual sensor networks," *Journal of Network and Computer Applications*, vol. 72, pp. 72–86, 2016.

Generated using Undegraduate Thesis LaTeX Template, Version 1.4. Department of Computer Science and Engineering, Bangladesh University of Engineering and Technology, Dhaka, Bangladesh.

This thesis was generated on Saturday 20th October, 2018 at 1:06pm.