Lifetime-Aware Intrusion Detection under Safeguarding Constraints

Ali Iranli, Hanif Fatemi, Massoud Pedram
Dept. of Electrical Engineering
University of Southern California
{iranli, fatemi, pedram}@usc.edu

Abstract: **This paper addresses the problem of maximizing the service lifetime of a distributed battery-powered sensor network in the context of the network interdiction problem under user-specified initial energy and probability of detection constraints. We consider a version of this problem where the probability distribution of selecting paths by the intruder is known to the interdictor. A two-step solution technique is proposed in whereby first the safeguarding constraints are satisfied and then the scheduling problem is solved. Experimental results demonstrate the effectiveness of the proposed two-step approach.**

I. INTRODUCTION

We consider the question of maximizing the sensor network lifetime in the context of the *network interdiction* problem. The network interdiction or *network inhibition* problem models the computation of a strategy to stop attacks on the capacity of a flow network. The simplest version of the network interdiction problem can be formulated as the minimization of the maximum achievable undesirable flow through a physical network subject to constraints on the interdiction resources. More precisely, consider a scenario whereby an adversary wishes to move as much of a single commodity from a source node *s* to a target node *t* in a directed physical network (representing all possible routes and stops between *s* and *t*). Each edge (*i*, *j*) has a capacity of c_{ij} units of commodity and requires from the interdictor an expenditure of r_{ij} units of some resource to stop the flow of undesirable commodity on the edge i.e., "break the edge." The goal of the interdictor is thus to minimize the maximum amount of flow the enemy can push through the network subject to using no more than *R* units of resources. The interdictor achieves this goal by *placing sensors* on different edges of the network. Each sensor samples some portion of the traffic (depending on its *sampling rate*) traversing the designated edge and examining it to determine whether or not the intruder is passing through.

An alternative formulation of the network interdiction problem is to find the minimum number of sensors (or the minimum sum of the sampling rates of all deployed sensors) needed to guarantee that the probability of detection of an intruder moving in the network is at least L_{min}. This version, called *Intrusion Detection under Safeguarding* constraint (IDS) is the version of the problem that we are interested to extend so as to account for the finite capacity of the batteries used as energy sources for the sensor nodes.

The strategy adopted by the interdictor for edge inspection not only changes the intrusion detection probability, but also affects the overall sensor network lifetime. This is due to the fact that the system lifetime is a strong function of the energy consumption rates of the different sensors, which are in turn directly proportional to their sampling rates.

Sensor placement (deployment) is a well researched problem [1]-[6]. Wood [1], McMasters and Mustin [2], Steinrauf [3], and Phillips [4] have studied this problem for military settings and applications to the interdiction of illegal drugs and precursor chemicals. Game-theoretic network interdiction models have also been studied by Wollmer [5], Washburn and Wood [6], and Kodialam et al. [7]. These techniques determine the optimal set of edges in the network for placing sensors so as to cost-efficiently detect an evader surreptitiously moving through a network. As a representative example, the authors of [7] solve the sensor placement problem under a sampling budget constraint in the context of security in communication networks. The authors consider the problem in a game-theoretic framework, where the intruder picks paths to minimize chances of being detected whereas the network operator chooses a sampling strategy to maximize the chances of detecting the intruder. Using min-max games, the authors propose a sampling scheme, which is optimal in this game theoretic setting.

With the rapid increase in the required functionality and performance of sensors, and therefore, high levels of power consumption in the sensors, the lifetime of a sensor network has become a key concern of data gathering systems. This has influenced the design of sensor networks so as to be driven by probability of detection (i.e., the level of safeguarding or "protection" that can be provided against intrusion) as well as network lifetime (i.e., how long can the network be "protected" against enemy intrusion given a set of initial battery capacities for the sensor nodes) considerations. However, the primary focus of the research in network interdiction has been on sensor deployment to minimize the illegal flow in the network. To our knowledge, no researcher has considered network interdiction problem with the target of meeting a sensor network lifetime constraint or maximizing this same metric under a probability of detection constraint.

In this paper, the energy capacity of each sensor is treated as a new type of network resource, and the effects of the finite energy capacity of each sensor node on the overall system lifetime and probability of detection is studied. More precisely, we address the problem of maximizing the sensor network lifetime subject to 1) initial energy levels for all sensor nodes, E_0, and 2) a lower bound on the level of network "protection" or safeguarding, L_{min}. This problem, which is called *Lifetime-aware Intrusion Detection under Safeguarding* constraints (LIDS), is solved in two steps. In the first step, we generate a set of distinct, non-dominated solutions such that minimum safeguarding constraint is satisfied. Next, we find an optimal scheduling policy to switch between these "sensor placement solutions" with the objective of maximizing the overall network lifetime by specifying which sensor placement solution to use and for how long.

The remainder of this paper is as follows; section II provides necessary background and the problem formulation. Section III presents the proposed solution to the problem and sections IV and V present the experimental results and conclusions.

II. Background and Problem formulation

Consider a directed graph $G(V,E)$, with node set V and edge set E. Let s and t denote the source and the target nodes, respectively. Furthermore, let f_e denote the flow rate of (legal and illegal) traffic on each edge $e \in E$. Note that traffic values do not satisfy the flow conservation constraints because it is possible for a portion of traffic to be generated and/or consumed in an intermediate node. Furthermore, the larger f_e requires a larger sampling rate, s_e, for the sensor so that the illegal traffic can be separated from the legal traffic. For example $G(V,E)$ can represent the corridors and walkways in a bank where s represents the entrance door and t signifies the bank's safe. Moreover, f_e denotes the average number of people per unit of time who use corridor e to move between two points in the bank.

A sensor with an initial energy level of E_0 is placed on every edge. The probability of detecting the intruder on any edge e is a function of f_e and the quality of detection on the edge. The latter in turn depends on the power consumption, pow_e, of the sensor placed on the edge. Therefore, we can write:

$$PoD_e = g_e(pow_e, f_e) \tag{1}$$

where g_e is generally a non-decreasing function of pow_e and a non-increasing function of f_e capturing the characteristics of the sensors used on edge e (cf. Section IV.) Note that function, g_e, takes into account the intruders' speed as he/she is traversing an edge of known length. For example, one may adopt a simple function taken as the ratio of the sampling rate of the sensor, s_e to the traffic flow rate through the edge, that is,

$$PoD_e \equiv \frac{s_e}{f_e} = \frac{pow_e - \gamma_e}{\kappa_e f_e} \tag{1.a}$$

where κ_e and γ_e denote regression coefficients, which are easily obtained from power profiling and regression analysis. From now on, for the sake of simplicity we will assume κ_e and γ_e to be equal for all edges in graph G.

Let $q(\pi_1)$ denote the probability that path π_1 between s and t is chosen by the intruder. The probability that the intruder takes path π_1 and is subsequently detected on that path, PoD(π_1), is given by:

$$PoD(\pi_1) = q(\pi_1).\left[1 - \prod_{e \in \pi_1} (1 - PoD_e)\right] \tag{2}$$

The IDS problem is to find a subset of edges, S, and their corresponding sensor sampling rates (hence, their power consumptions) such that the probability of detecting an intruder moving from source s to target t along any arbitrary path is larger than or equal to some threshold value L_{min}. The objective is to minimize the overall system power consumption. This can be written in mathematical form as:

$$\underset{S \subset E}{argmin} \left(\sum_{e \in S} pow_e \right) \tag{3}$$

such that

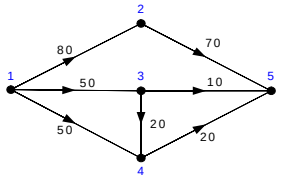
$$\sum_{\pi \in \Pi_{s \to t}} q(\pi).\left[1 - \prod_{e \in \pi \cap S} (1 - PoD_e)\right] \geq L_{min} \tag{3.a}$$

where $\Pi_{s \to t}$ denotes the set of all paths from source s to target t and the product term computes the probability that an intruder is not detected on any of the edges of path π that have sensors on them. The implicit assumption is that PoD_e=0 for any edge e such that $e \in \pi \wedge e \notin S$. This nonlinear mathematical formulation of the problem cannot be solved optimally and efficiently. In fact, it has been shown in [1] that this problem is NP-complete.

Simplifying the IDS problem, Wood et al. [1] add an additional constraint to the problem which limits the maximum number of sensors on any path from source s to target t to one. By using this additional constraint and assuming a linear characteristic function g for the sensors, one can rewrite constraint (3.a) as $\sum q(\pi).PoD_e \geq L_{min}$, thus transforming the problem into an integer linear programming problem.

In this paper, we attempt to solve the LIDS problem with the objective of maximizing the overall system lifetime subject to a global safeguarding constraint. However, maximization of the system lifetime is not equivalent to minimization of the total power consumption. For example, consider the graph shown in Figure 1 with indicated traffic pattern and initial energy of 100 joules for each sensor. Cuts C1-1 and C1-2 are generated by using cost function 3 for this graph (cf. Table 1). Using these solutions, one can activate solution C1-1 for 1610 seconds before the sensor on edge (2,5) dies. Subsequently, solution C1-2 will be activated for 1,960 seconds before the sensor on edge (4,5) dies. Therefore, these solutions together can maintain the safeguarding constraints for approximately 3,570 seconds. Note that using C1-1 before C1-2 results in the consumption of a portion of the initial energy of the sensor on edge (4,5) in the first period, making it the bottleneck of the system lifetime in the next period. Now, consider another pair of cuts C2-1 and C2-2, which does not minimize cost function (3) but maintains the safeguarding constraints for a longer period of time by including edge (4,5) in C2-1 only. In fact C2-1 and C2-2 cuts can satisfy the safeguarding constraints for approximately 4,000 seconds.

Motivated by the aforementioned example, we seek to identify a revised cost function for the IDS problem so that we can generate a set of "good" cuts to choose from when solving the LIDS problem to achieve longer sensor network lifetime. In the following, we formulate the LIDS problem mathematically. Subsequently, we break the LIDS problem into two sub-problems, a) IDS problem with a revised cost function and b) a concurrent cut selection and scheduling problem.



Path	Probability
{1,2,5}	1/21
{1,3,5}	8/21
{1,4,5}	3/21
{1,3,4,5}	9/21

Figure 1. Example for Lifetime aware Intrusion Detection

Cuts	Edges	Total Power	Max. Power
C1-1	{(2,5), (3,5), (4,5)}	94mW	47mW
C1-2	{(1,2), (3,5), (4,5)}	98mW	55mW
C2-1	{(2,5), (3,5), (4,5)}	94mW	47mW
C2-2	{(1,2), (3,5), (3,4), (1,4)}	155mW	50mW

Table 1. Solution for example 1

LIDS problem can be modeled as follows:

LIDS Problem: Given a directed graph $G(V,E)$ and initial energy E_0 for each sensor on the edges of G, for a required minimum safeguarding level L_{min}, find subsets of edges $S_1, S_2, \ldots, S_n$ and their corresponding time spans $t_1, t_2, \ldots, t_n$ so that

$$Max \sum_{i=1}^{n} t_i \qquad (4)$$

with

$$\sum_{\pi \in \prod_{s \to t}} q(\pi).\left[1 - \prod_{e \in \pi \cap S_i} \left(1 - PoD_e\right)\right] \geq L_{min} \quad \forall i \qquad (4.a)$$

$$\sum_{i:e \in S_i} t_i \cdot pow_e \leq E_0 \quad \forall e \in E \qquad (4.b)$$

Here, the first constraint (4.a) is the minimum required safeguarding constraint in each time period i, similar to constraint (3.a) whereas the second constraint (4.b) captures the effect of limited energy source of the sensor nodes.

III. Solution Technique

In this section an approach will be proposed to approximately, yet efficiently, solve the LIDS problem. Consider a generic solution to the LIDS problem, which we will call, the LIDS_*Algorithm* (Figure 2).

The LIDS_Algorithm has two steps. First, it generates all feasible subsets S_i of E, which we will denote by Σ_E (line 4-8). This is done by calculating the minimum total power consumption of all sensors that are placed on edges of S_i so that inequality (4.a) becomes active (line 6).

Then, in the second step, for each subset S_i from step 1, it calculates t_i such that constraint (4.b) is satisfied (line 9-12) and the system lifetime as given in (4) is maximized.

It is easy to see that a solution generated for the LIDS problem by using this two-step approach is optimal. This is due to the fact that in step 1, the algorithm generates all feasible subsets; whereas in step two, it calculates the best scheduling of all the feasible solutions. Obviously, the disadvantage of this two-step approach is that the cardinality of Σ_E is an exponential function of the graph size. Therefore, the proposed brute-force solution is generally impractical. In the following, a method for generation of some promising S_i's will be described so that step one becomes computationally manageable.

A. Generating feasible solutions

To do this, we only generate subsets of E that satisfy the safeguarding constraints and simultaneously optimize some cost function. This cost function is chosen such that it results in generation of a small number of feasible subsets of E that are likely to be present in the optimal solution to the LIDS problem. The remaining questions are a) what this cost function is, and b) how to iteratively find next optimal solution.

```
   Algorithm LIDS_Algorithm (Graph G, E0, Lmin)
   begin
1.    ΣE = Find_Feasible_Solutions (G, Lmin);
2.    Schedule = Find_Schedule (ΣE, E0 );
3.    return Schedule;
   end
   ====================================
   Algorithm Find_Feasible_Solutions (Graph G, Lmin)
   begin
4.    feasible_subsets=∅;
5.    foreach subset Si⊂E s.t. constraint (4.a) can be
      satisfied;
      begin
6.       Pmin=Find_minimum_power (Si);
7.       feasible_subsets= feasible_subsets∪{( Si, P-
         min)};
      end
8.    return feasible_subsets;
   end
   ====================================
   Algorithm Find_Schedule (ΣE, E0)
   begin
9.    Schedule=∅;
10.   foreach (Si, Pi)∈ ΣE
      begin
11.      Find ti s.t. (4) is maximized and constraint
         (4.b) is satisfied;
12.      Schedule= Schedule∪{(ti, Si, Pi)};
      end
13.   return Schedule;
   end
```

Figure 2. *LIDS_Alogorithm*

1) Choice of the Cost function

In this section use of two different cost functions for step 1 of the LIDS algorithm is investigated.

If we adopt the *Min-Sum power* cost function for the LIDS problem, then we basically have the IDS problem as stated in (3), where the total power consumption of the sensors in the solution is minimized. The solution to this problem is similar to that presented in [1]. The integer linear programming formulation can be written as,

$$min \sum_{\forall i,j} pow_{ij} \qquad (5)$$

such that

$$\begin{aligned} & PoD_{ij} = \frac{pow_{ij} - \gamma}{\kappa \cdot f_{ij}} \\ & PoD_{ij} + \alpha_i \leq 1 \qquad \forall ij \in E \\ & PoD_{ij} - \alpha_j \leq 0 \\ & \alpha_i \in \{0,1\} \qquad \forall i \in V \\ & \alpha_s = 0, \alpha_t = 1 \end{aligned} \qquad (5.a)$$

$$PoD_{ij} \times \sum_{\substack{\pi \in \prod_{s \to t} \\ (i,j) \in \pi}} q(\pi) \geq L_{min} \qquad (5.b)$$

In formulation (5), α_i is a binary variable for each node i that show in which partition node i is located, α_i=0 if node i is on source node's side and α_i=1 if node i is on target node side. Therefore, an s-t cut is identified with a set of edges (i, j) such that α_i=0 and α_j=1 (eqn. 5.a). Pow_{ij} denotes the power consumption of the sensor on edge (i, j) and the first constraint in (5.a) captures the characteristics of that sensor. Three remaining constraints in (5.a) guarantee that only edges (i, j) for which α_i=0 and α_j=1 can have active sensors that consume power, i.e., $Pow_{ij} \neq 0$. The minimum safeguarding constraint is rewritten in equation (5.b) and the total power consumption cost function is shown in equation (5).

On the other hand if we adopt the *Min-Max power* cost function, then we obtain a version of the IDS problem where the objective is to minimize the maximum of power consumption of the sensors. Mathematical program formulation of this version of the problem can be written as:

$$min \;\; P_{\max} \qquad (6)$$

With constraints (5.a) and (5.b) and the addition of:

$$pow_{ij} \leq P_{\max} \qquad \forall ij \in E \qquad (6.a)$$

The solution to this problem can be found using standard ILP solvers.

2) *Finding the next optimal solution iteratively*

To generate the next optimal solution after finding the current solution, we must modify the underlying network graph by prohibiting the use of a subset of edges. This is necessary to make sure that a new solution can be generated and that the effect of previously generated solutions is taken into account. Two different heuristics for the modification of the network is considered: the *maximum power threshold heuristic*, and the *number of repetitions heuristic*.

In maximum power threshold heuristic, network edges in the previous solution with sensors that have power consumption more than h percent of the maximum power sensor in that solution, are eliminated from the network and thus will not be considered as part of any solution generated in a subsequent solution. The rationale is that high power consuming sensors are less likely to be present in the optimal solution of the subsequent LIDS problem formulations since they are more likely to result in lower system lifetime.

In the number of repetitions heuristic, If a sensor has been used in at most $N_{\max}$ previously generated solutions then this sensor cannot be present in a subsequent solution, and hence, it is removed from the network. This constraint basically helps to increase the utilization of the sensors in the network in generation of solution set. Now, we turn our attention to solving step two of the LIDS_Algorithm.

B. *Scheduling Algorithm*

In step two of LIDS_Algorithm we are required to solve the following problem,

Scheduling Problem: Given a set of k solutions to the IDS problem $S_1, \ldots, S_k$, solve the following:

$$T \equiv Max \sum_{i=1}^{n} t_i \qquad (7)$$

Such that

$$\sum_{i \mid e \in S_i} t_i \cdot pow_e \leq E_0 \quad \forall e \in E \qquad (7.a)$$

It is easy to see that Problem 2 is a linear programming problem, and hence, it can optimally be solved in polynomial time. To further increase the efficiency, one can introduce a dominance relation between different sensors and thus consider constraint (7.a) only for those sensors that are most likely to exhaust their energy first as described below.

Definition: Let $SPS(e_i)$ denote the *set of parent sets* of edge e_i, i.e.,

$$SPS(e_i) = \left\{ S_j \,\middle|\, e_i \in S_j \right\} \qquad (8)$$

We say edge e_j is dominated by edge e_i , denoted by $e_j \subset e_i$, if $SPS(e_j) \subset SPS(e_i)$ and $\forall \; S_k \in SPS(e_j)$, $pow_{e_j}^{S_k} \leq pow_{e_i}^{S_k}$ where pow_e^S denotes the power consumption of the sensor on edge e when solution S is active.

Definition: Given a collection of sets S_k, $k=1\ldots m$, set Θ is a *minimal* dominating *set* if

$$\forall e_j \in \bigcup_{k=1}^{m} S_k : \exists e_i \in \Theta \wedge e_j \subset e_i \qquad (9)$$

and $|\Theta|$ is minimum.

Theorem: To satisfy constraint (7.a) for all edges in the network, it is sufficient to satisfy it only for edges in Θ . ◆

Proof: If edge e_j is dominated by edge e_i then the system lifetime T is determined by the lifetime of edge e_i and not e_j. This is due to the fact that not only for every time instance, t where e_j is active e_i is active (because SPS of e_j is subset of SPS of e_i), but also e_i is consuming more power than e_j when it is active because $\forall \; S_k \in SPS(e_j)$, $pow_{e_j}^{S_k} \leq pow_{e_i}^{S_k}$. Knowing that both e_i and e_j had same initial energies would result that for all t remaining energy of e_i is less than or equal to that of e_j and hence when e_i satisfies (7.a) implies that e_j would satisfy the same condition. ◆

To find the minimal dominating set Θ of the selected solutions $S_1, \ldots, S_m$, one can construct a directed graph H in which each edge e_i in E is a node u_i, and there is a directed edge from u_j to u_i exactly if edge e_i dominates edge e_j. Then, all directed cycles in the resulting graph will be removed by using the operation shown in Figure 3. Note that due to the definition of dominance relation, a directed cycle represents an

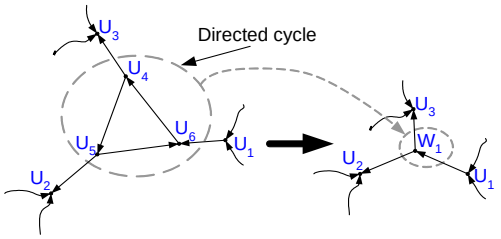


Figure 3.Cycle removal operation

equivalence class; therefore, one can remove all the nodes (which are the edges on the actual underlying network) in a cycle and collapse them into one node. Of course, all edges connecting the original cycle to other nodes must now be incident to the collapsed node. (See Figure 3). After removing the cycles in graph *H*, the resulting graph will be a forest of rooted, directed trees. The set of leaf nodes of all trees in *H* is the minimal dominating set Θ. Therefore, linear program (7) can be solved with constraints written only for the members of the minimal dominating set, further reducing the size of the problem.

IV. Experimental Results

To generate our experimental results we used the web-server based camera from *STARDOT* Technologies as the sensor [9]. This camera can take up to 35 snap shots per second from its area of coverage. Probability of detection for this sensor is proportional to the number of frames per second taken from the coverage area. This number is in turn related to the sensor's power consumption. Based on actual current measurements, the probability of detection was related to the camera's power consumption as follows:

$$PoD_e = \frac{(pow_e - 100_{mW})}{9_{mW}} \cdot \frac{1}{f_e} \qquad (10)$$

Initial energy values are assumed to be 100 Joules.

The underlying network graphs considered are two graphs from Wood et al. [6] depicted in Figure 6 and Figure 7, and five other randomly-generated graphs from graph benchmark database of Rutgers University [8]. Key information about these graphs is provided in Table 2. The traffic values (flow rates) for the edges in these five networks are generated using a random number generator (i.e., the *random(seed)* function of the linux kernel.) Furthermore, a preprocessing program is used to generate all paths starting from the source node *s* and ending in target node *t* and assigning a randomly generated probability of intrusion to each path. All random generators are assumed to be uniformly distributed. Using this information as input data, we proceed to solve the LIDS problem. Figure 5 presents the flow used for generation of experimental results.

There are three phases for generating experimental results. In the first phase, the input networks, their traffic values, and path probability values are generated as described above. In the second phase, two cost functions of Section 3.1 are used to generate the solutions to the IDS problem. In this phase, an attempt to generate a solution that satisfies the safeguarding constraints is made. If any feasible solution is found, then the input graph is modified by using one of heuristics of Section 3.2 and a new iteration is initiated. This procedure continues until there are no more possible solutions left. In the third and final phase, three different schedules are generated based on three different solution sets available: the solution set from the min-sum power cost function, the solution set from the min-max power function, and the solution set generated by creating the conjunction of the min-sum and min-max power solutions. In each case, we solve the linear programming problem of (5), which results in an optimal schedule for the given set of solutions of IDS problem.

Graph No.	Graph Name	# nodes	# edges
1	Wood-1	37	58
2	Wood-2	38	51
3	Johnson8-4-4	70	1855
4	Johnson8-2-4	28	210
5	MANN_a9	45	918
6	Mulsol-i-5	186	3973
7	DSJC125-1	125	1472

Table 2. Some statistics about benchmarks

In order to evaluate our heuristics, the *exact* solution is also obtained by generating the complete set of feasible subsets of *E* in step 1 of the *LIDS-Algorithm*. Different solutions are compared in Figure 4. For example, in case of *MANN_a9* graph when L_{min}=95% the system lifetime is 10.80, 9.75, and 10.85 seconds for Min-Sum, Min-Max, and hybrid cost functions, respectively; whereas the optimal solution for this case was 11.83 seconds. Note that all of different heuristics are within 10% of the optimal solution.

Moreover, a closer look at the graphs shown in Figure 4 reveals that as the safeguarding constraint is lowered the lifetime of the system is increase as expected. Moreover, for tighter requirements on the safeguarding constraint the Min-Sum heuristic seems to follow the optimal solution very closely. This is due to the fact that for higher minimum required safeguarding limit, i.e., larger L_{min}, different solutions have very similar power consumption characteristics and therefore, the optimal solution set is very similar to the solution set from the Min-Sum heuristic.

The relative degradation of lifetime of the network based on hybrid heuristic and relative runtime normalized to runtime of optimal solution, i.e., the optimal solution's runtime is assumed to be 100 units of time, is also reported in Table 3. Note that the overall error in network lifetime due to application of heuristic approach is upper-bounded by less than 10% while the time required to generate the solution is reduced by 300%.

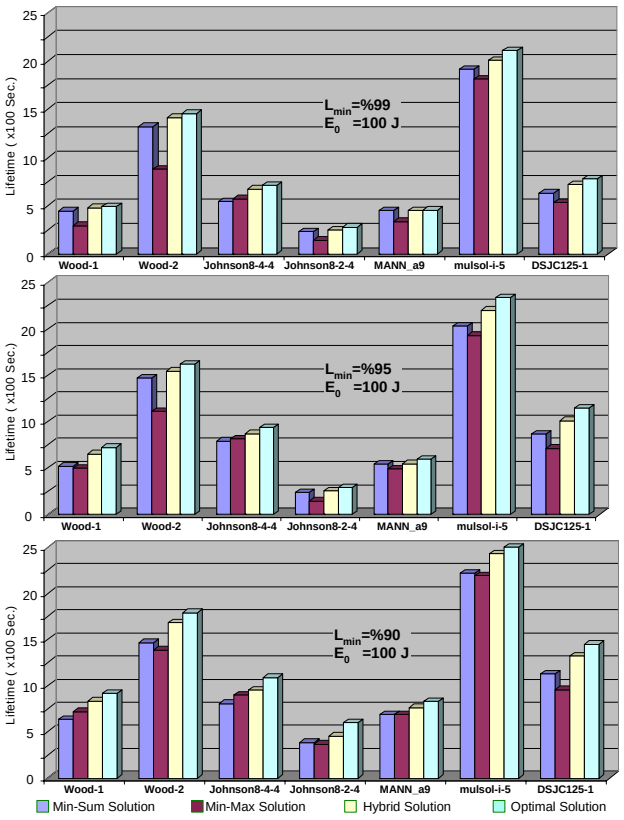


Figure 4. Lifetime vs. Safeguarding Level (L_{min})

	Relative degradation in network lifetime	Runtime
Optimal Solution	0.00%	100.00
Hybrid Heuristic (Lmin=99%)	4.89%	30.26
Hybrid Heuristic (Lmin=95%)	8.62%	31.51
Hybrid Heuristic (Lmin=90%)	8.98%	33.58

Table 3. Relative error and runtime of hybrid heuristic

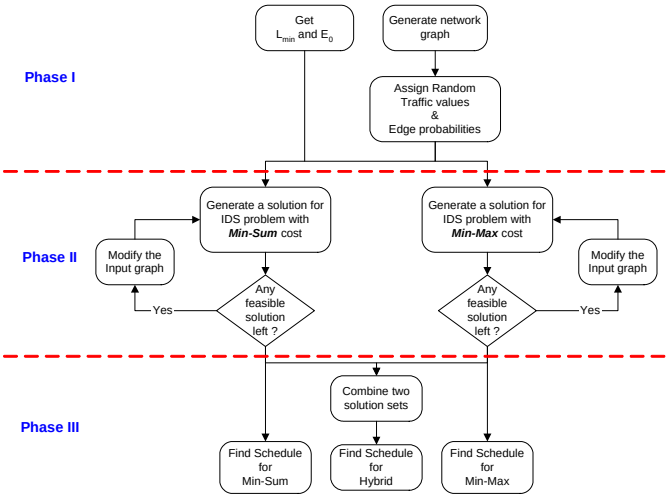


Figure 5. Flow of solution generation for LIDS problem

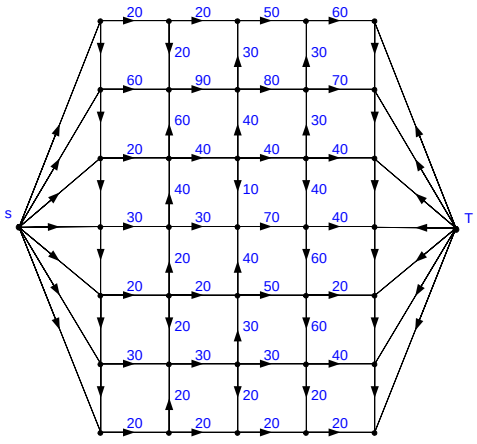


Figure 6. 7×5 example from Wood et al. [6]

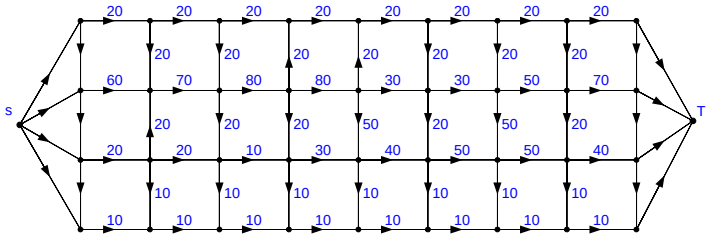


Figure 7. 4×9 example from Wood et al. [6]

V. Conclusions

The problem of maximizing the service lifetime of a distributed sensor network in the context of the network interdiction problem under user-specified initial energy and probability of detection constraints is addressed. The offline version of this problem is considered. Different heuristic solutions based on partitioning of problem in two parts, i.e., placement and scheduling, are proposed. Experimental results to show the effectiveness of the approach are presented.

References:

[1] K.J. Cormican, D.P. Morton, R.K. Wood, "Deterministic Network Interdiction," *Journal of Mathematical and Computer Modeling*, Vol.17, No.2, 1993.

[2] A.W. McMasters and T.M. Mustin, "Optimal Interdiction of a Supply Network," *Naval Research Logist.* Vol.17, 1970.

[3] R.L. Steinruf, "A Network Interdiction Model," *M.S. Thesis*, Naval Postgraduate School, Monterey, CA, 1991.

[4] C.A. Phillips, "The Network Destruction Problem," Sandia National Lab. Albuquerque, NM, 1992.

[5] R.D. Wollmer, "Two-Stage linear programming Under Uncertainty with 0-1 Integer First-Stage Variables," *Journal of Mathematical Prog.*, Vol.19.

[6] A.R. Washburn, R.K. Wood, "Stochastic Network Interdiction," *Operations Research*, Vol.46. No.2, Mar. 1998.

[7] M. Kodialam and T.V. Lakshman, "Detecting Network Intrusion via Sampling: A Game Theoretic Approach," *Proc. of Infocom*, Apr. 2003.

[8] ftp://dimacs.rutgers.edu/pub/challenge/graph/benchmarks/

[9] http://www.stardot-tech.com/