The K-Observer Problem in Computer Networks

H.B. Acharya[1], Taehwan Choi[1],
Rida A. Bazzi[2], and Mohamed G. Gouda[1,3]

[1] The University of Texas at Austin, USA
{acharya,ctlight}@cs.utexas.edu
[2] Arizona State University, USA
[3] The National Science Foundation, USA

Abstract. For any non-negative integer K, a *K-observer* P of a network N is a set of nodes in N such that each message, that travels at least K hops in N, is handled (and so observed) by at least one node in P. A K-observer P of a network N is *minimum* iff the number of nodes in P is less than or equal the number of nodes in every K-observer of N. The nodes in a minimum K-observer of a network N can be used to monitor the message traffic in network N, detect denial-of-service attacks, and act as firewalls to identify and discard attack messages. This paper considers the problem of constructing a minimum K-observer for any given network. We show that the problem is NP-hard for general networks, and give linear-time algorithms for constructing minimum or near-minimum K-observers for special classes of networks: trees, rings, L-rings, and large grids.

1 Introduction

Every node in a computer network performs a number of traditional tasks: generating messages, routing and forwarding messages, and consuming messages. Beside these traditional tasks, some nodes are designated, in a network, to perform additional tasks: observing and collecting statistics concerning the message traffic that goes through each designated node and filtering the message traffic that goes through each designated node. We refer to those nodes that are designated, in a computer network, to perform these additional tasks as *network observers*.

This paper discusses the problem of how to select the nodes to be designated network observers in a computer network. We start the discussion by proposing a criterion, named minimum K-observers for some non-negative integer K, which can be used to identify the network observers in a computer network.

A K-observer P of a network N is a set of nodes in N such that each message, that travels at least K hops in N, is handled (and so observed) by at least one node in P.

Clearly, this definition of a K-observer P depends on the chosen value of K. On one hand, if the chosen value of K is small (with respect to the total number

X. Défago, F. Petit, and V. Villain (Eds.): SSS 2011, LNCS 6976, pp. 5–18, 2011.

of nodes n in N), then P is a large set containing most of the nodes in N. For example, if $K = 0$, then P is the set of all nodes in N. On the other hand, if the chosen value of K is large (with respect to the total number of nodes n in N), then P can be a small set containing few nodes in N. For example, if $K = n-1$, then P can be a singleton containing only one node (any node) in N. Also, if $K \geq n$, then P can be the empty set.

Note that if a set P is a K-observer of a network N, then adding more nodes of N to P does not change the status of P of being a K-observer of N. This note suggests that we should be more interested in minimum K-observers, rather than in K-observers, as defined next.

A K-observer P of a network N is *minimum* iff the number of nodes in P is less than or equal the number of nodes in every K-observer of N.

Based on this discussion, in order to identify the network observers in a computer network N, one needs to construct a minimum K-observer P of network N, for some chosen K, and use the nodes in the constructed P as the network observers in N.

The problem of constructing a minimum K-observer in a network is related, though not identical, to several established problems in network design:

1. Constructing a node cover in a network
   The *Node Cover Problem* is to find a minimum number of nodes such that each link is incident to at least one node in the network [9]. The K-observer problem is more generalized than the *Node Cover Problem* such that the Node Cover Problem is the special case of the K-observer problem when $K = 1$. Armbruster [3] discusses the Node Cover Problem for sources and destinations of all paths in the network whereas the K-observer problem generalizes the Node Cover Problem for a path of length K in the network.
2. Creating a backbone for communication
   A very important problem for wireless networks, the *Connected Dominating Set Problem*, is to find a set of nodes that are connected, such that each node in the network either belongs to this set of nodes or is one hop from it [19]. A connected dominating set is used for virtual backbones in wireless networks. Although mobile networks do not have physical backbones, virtual backbones can be formed to help communication in wireless ad-hoc networks [2], [17], [12].
3. Placing guards in an art gallery
   The *Art Gallery Problem* is to determine the number of guards necessary to cover an art gallery, such that every point in the art gallery is guarded by at least one observer [7]. This problem is equivalent to the *Coverage Problem* in the context of wireless sensor networks [10], wireless ad-hoc networks, and wireless sensor ad-hoc networks [13]. Moreover, this problem is equivalent to the *Dominating Set Problem* if guards must be placed on nodes, and only nodes need to be guarded [9].

4. The problem of facility location in a network
   The *Facility Location Problem* is to find a place for a facility such that the distances from customers are minimized [18]. This problem can be regarded as the *Set Cover Problem* [11]; it has been discussed in various contexts such as the placement of monitoring nodes [4], web server replicas [14], and overlay nodes [16].

The rest of the paper is organized as follows. We begin by proving that our K-observer problem is NP hard for general networks in Section 2. We then explore solutions for some special cases: tree networks in Section 3, ring networks in Section 4, and grid networks in Section 5. We go on to discuss some possible applications for the K-observer problem, and conclude with a few remarks.

2 The K-Observer Problem

A *network* N is an undirected graph (V, E), where V is a nonempty set of *nodes* and E is a set of undirected *links*. Each *link* in E is a set of two distinct nodes in V. A link $\{u, v\}$ in a network N is said to be *incident* at nodes u and v in N.

A *path* in a network N is a nonempty sequence $(u_1, u_2, \ldots, u_r)$ of distinct nodes in N such that each pair $\{u_i, u_{i+1}\}$ of consecutive nodes in the sequence constitutes a link in network N.

The *length* of a path $(u_1, u_2, \ldots, u_r)$ in a network is $r - 1$. For example, the length of the path (u_1) is 0, the length of the path (u_1, u_2) is 1, and so on.

Let P denote a set of nodes in a network N and let K be a non-negative integer. Set P is called K-*observer* of N iff every path of length at least K in N has at least one node in P.

A K-observer P of a network N is called *minimal* iff for each node u in P, N has a path q of length at least K such that P and q share only one node and their shared node is u.

A minimal K-observer P of a network N is called *minimum* iff for every minimal K-observer Q of network N, the number of nodes in Q is at least the number of nodes in P.

Let K be a non-negative integer. The K-*observer problem* is to design an algorithm that takes as input any network N and produces as output a minimum K-observer of network N.

An algorithm that solves the K-observer problem, when $K = 0$, can be designed as follows. This algorithm takes any network N as input and produces the set of all nodes in N as output. The correctness of this algorithm is based on the observation that the only 0-observer (and so the only minimum 0-observer) of a network N is the set of all nodes in N.

Unfortunately, the following theorem states that the K-observer problem, for any given constant $K > 0$, is NP-hard, so any algorithm to solve this problem is very likely to be expensive.

Theorem 1. *The K-observer problem, for any given constant value of K that is greater than 0, is NP hard.*

Proof. We begin our proof by noting that, for the case where $K = 1$, the K-observer problem reduces to choosing a minimum number of nodes such that every path of 1 or more hops i.e. every edge or simple path in the graph has at least one chosen node. This is the well-known minimum vertex cover problem, which is of course NP-hard [9].

We will now prove the NP-hardness of the general case by contradiction. If the K-observer problem is solvable in polynomial time for any value of K greater than 1 (say $K = k$), then minimum vertex cover is also solvable in polynomial time.

Suppose the K-observer problem is polynomial-time solvable for $K = k$. Given a network N, we can find its minimum vertex cover as follows.

To each node u of N, we attach a "string" of nodes $u_1, u_2, ...u_{k-1}$. These nodes have no edges incident on them, except the edges that link them to form the string. We call this modified network, consisting of N as well as the new nodes and edges added to form strings, network N'. The node on a string that belongs to N is called the base node of the string.

We now run the polynomial-time algorithm to find the minimum k-observer of N'. Note the size of N' is k times, i.e. a constant times the size of N, so this runs in time polynomial in the size of N also.

Further, we note that as this is the minimum k-observer, it will not contain more than one member of each 'string' $(u, u_1, ...u_{k-1})$. (If there is more than one member, the one farther away from the base node i.e. u can be unmarked without breaking the condition that all paths of length k or more have at least one marked node, so the solution is not minimal and not minimum.) We now apply a simple linear time transformation: if any node in a string is marked, we unmark it and mark the base node of the string.

The set of marked nodes is a minimum K-observer of N', because it is still a K-observer (any path using edges from the original N that had a marked node on it, still has a marked node on it, and any path not using edges from N has a maximum length of $k-1$) and it has the same number of nodes as the minimum K-observer found by our polynomial-time algorithm. It is also a vertex cover of N (because if there is an edge $\{u, v\}$ in N such that neither u nor v is marked, then there is a path $\{u, v, ...v_{k-1}\}$ of length k in N' without any marked nodes, which is impossible). Thus, we have a vertex cover of size m', where m' is the size of the minimum k-observer of N'.

Now we note that any vertex cover of N is a k-observer of N', as any path of length k or more must use at least one edge in N and thus contain a marked node. So if the size of the minimum vertex cover of N is m and the minimum k-observer of N' is m', we have $m \geq m'$. (In other words, the minimum vertex cover of N being a k-observer of N' is at least as large as the minimum k-observer of N'.)

Hence the vertex cover we compute, being of size m' and thus $\leq m$, is in fact the minimum vertex cover (because it is a vertex cover, upper bounded

by the size of the minimum vertex cover). Thus, if the K-observer problem is solvable for general graphs for $K = k > 1$, we have a polynomial time algorithm for the minimum vertex cover problem, which is NP hard; in other words, the K-observer problem is NP hard for all $K > 0$.

Having shown that the K-observer problem is NP hard for general networks, we present next polynomial time (in fact linear time) algorithms for solving this problem for special classes of networks such as tree networks, ring networks, and grid networks.

3 *K*-Observers of Tree Networks

In this chapter, we solve the K-observer problem for tree networks that have no cycles. Figure 1 shows an example of a tree network T that has 13 nodes, named node 0 to node 12, and 12 links.

Next, we describe an algorithm that computes a minimum K-observer for a tree network. This algorithm takes as input a tree network T and a positive integer K and returns as output a minimum K-observer P of network T. This algorithm consists of the following four steps:

Step 1:
Choose any node in T to be the root and add directions to the links in T to make T a directed tree where the root is a sink node.
Step 2:
Define for each node x in T, a variable named len_x whose range of values is $0 \ldots K$.
Step 3:
For each node x in T, where the values of the len variables of all predecessor nodes of x have already been computed, compute the value of len_x as follows:

$len_x := 0$	if x has no predecessor y whose $len_y < K$
$:= len_y + 1$	if x has exactly one predecessor y whose $len_y < K$
$:= len_y + 1$	if x has two or more predecessors $\{y, z, \ldots\}$ whose $len's < K$ and len_y is the maximum len among those predecessors and len_z is the second maximum len among those predecessors and $(len_y + len_z + 2) < K$
$:= K$	if x has two or more predecessors $\{y, z, \ldots\}$ whose $len's < K$ and len_y is the maximum len among those predecessors and len_z is the second maximum among those predecessors and $(len_y + len_z + 2) >= K$

Step 4:
A minimum K-observer of tree network T is the set of every node x in T where $len_x = K$.

Theorem 2. *If this algorithm is applied to a tree network T, and a positive integer K, then the computed set by this algorithm is a minimum K-observer of network T.*

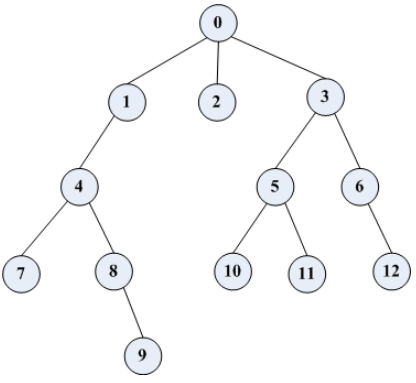


Fig. 1. Example of a tree network T

To see how this algorithm works, consider the tree network T in Figure 1. In step 1, we choose node 5 to be the root and we make the root a sink node such that T becomes a directed tree as shown in Figure 2(a). In step 2, we define a variable named len_x for every node x in T. In step 3, we compute len_x for every node x in T. In step 4, we show that the 2-observer of minimal cardinality of T is $\{4, 0, 3, 5\}$ in Figure 2(b). In addition to that, we show that the 3-observer of minimal cardinality of T is $\{4, 3\}$ in Figure 2(c).

Interestingly, the algorithm works irrespective of the choice of the root. Even if a leaf node is chosen as the root, the algorithm still correctly computes minimum K-observers in tree networks.

The time complexity of this algorithm is linear such that it is proportional to the number of nodes in the input tree network.

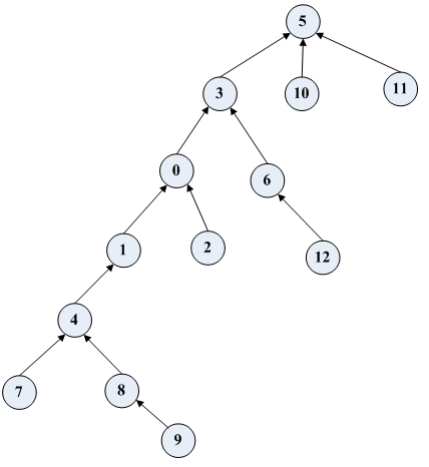
This algorithm as described above is centralized. But a distributed version of this algorithm can be described as follows:

Step 1:
Each node x, that has exactly one neighboring node y in the tree network T, knows that it is a leaf in T and so it assigns its variable len_x the value 0 and sends the value of its len_x to node y.

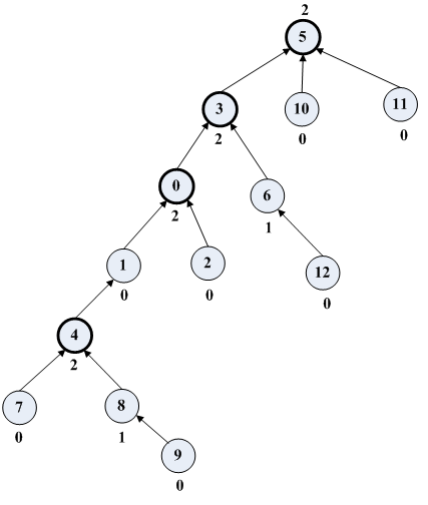
Step 2:
After a node y receives the value of len_x from every neighboring node x except one, say node z, then node y computes the value of its variable len_y (as described in the centralized version of the algorithm) and sends the computed value of len_y to node z.

Step 3:
If a node z receives the value of len_y from every neighboring node y, then node z recognizes that it is the root and computes the value of its variable len_z(as described in the centralized version of the algorithm).

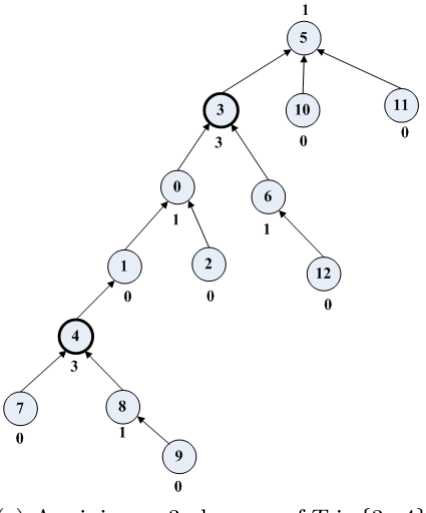
Step 4:
Every node x, where the value of len_x in K, knows that it is in the computed minimum K-observer P of the tree network T.



(a) Making T a directed tree by choosing node 5 to be the root



(b) A minimum 2-observer of T is $\{0,\ 3,\ 4,\ 5\}$



(c) A minimum 3-observer of T is $\{3,\ 4\}$

Fig. 2. Computing minimum K-observers of T

It is possible that during the execution of this distributed algorithm, two adjacent nodes y and z compute the values of their respective len variables and send their computed values to one another. Thus, each of these two nodes first sends its len value to the other node then receives the len value of the other node. In this case, the two nodes y and z behave differently depending on whether or not index y is larger than index z as follows:

i. The node with the larger index, say node y, ignores the value of variable len_z that it receives from $node_z$ and keeps the value of its variable len_y unchanged.
ii. The node with the smaller index, node z, recognizes that it is the root and uses the value of variable len_y received from node y to re-compute the value of its variable len_z.

4 *K*-Observers of Ring Networks

In this chapter, we solve the K-observer problem for ring networks. Figure 3 shows a *ring network* with n nodes, named $u_1, u_2, \ldots, u_n$, and n links.

Next, we describe an algorithm that computes a minimum K-observer for a ring network. This algorithm takes as input a ring network R and a positive integer K and returns as output a minimum K-observer P of network R. This algorithm consists of the following three steps:

Step 1:
Initially, $P := \emptyset$ (the empty set)
Step 2:
if R has at most K nodes **then**
return the empty K-observer P
terminate the algorithm
else
/* R has at least $K + 1$ nodes */
continue the algorithm
end if
Step 3:
remove any node, say node u, and its two incident links from network R
/* the resulting network is a tree */
apply the algorithm in Chapter 3 to compute a minimum K-observer Q of the resulting tree network
$P := Q \cup \{u\}$
return P and terminate the algorithm

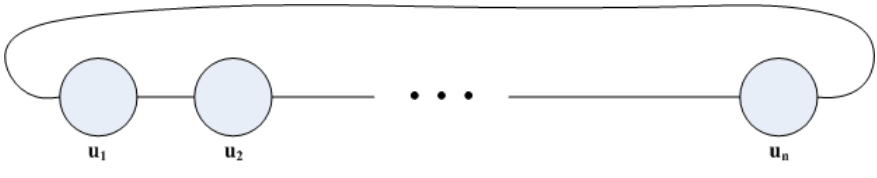


Fig. 3. A ring network of n nodes

Theorem 3. *If this algorithm is applied to a ring network R, and a positive integer K, then the computed set P by this algorithm is a minimum K-observer of network R.*

Note that the time complexity of this algorithm is linear in the number of nodes in the input ring network R.

Note also that if Step 2 is removed from the above algorithm, then the computed K-observer P of ring R may no longer be minimum. In this case, however, the number of nodes in P is no more than one over the number of nodes in a minimum K-observer of R. This suggests the following definition.

Let L be a non-negative integer. A K-observer P of a network N is called *L-bounded* iff the number of nodes in P is no more than L of the number of nodes in a minimum K-observer of network N. (Note that a minimum K-observer of a network N can now be regarded as a 0-bounded K-observer of N.)

Next, we describe a linear time algorithm that can be used to construct an L-bounded K-observer for a special class of networks called L-rings.

Let L be a positive integer. A network RR is called an *L-ring* iff RR has exactly L ring sub-networks. Note that if every ring sub-network in an L-ring is collapsed into a single super node then the resulting network is a tree.

An algorithm, for constructing an L-bounded K-observer P for any given L-ring RR, is as follows. First, identify a node in every ring sub-network in the given L-ring RR. Let $u_1, \ldots, u_x$ denote the identified nodes, where $x \leq L$. Second, remove the identified nodes and their incident links from the given L-ring RR. Note that the resulting network is a tree. Third, apply the algorithm in Chapter 3 to compute the minimum K-observer Q of the resulting tree. Compute the L-bounded K-observer P of the given L-ring RR as follows: $P := Q \cup \{u_1, \ldots, u_L\}$.

5 *K*-Observers of Large Grid Networks

In this chapter, we discuss how to construct minimal K-observers for large grid networks.

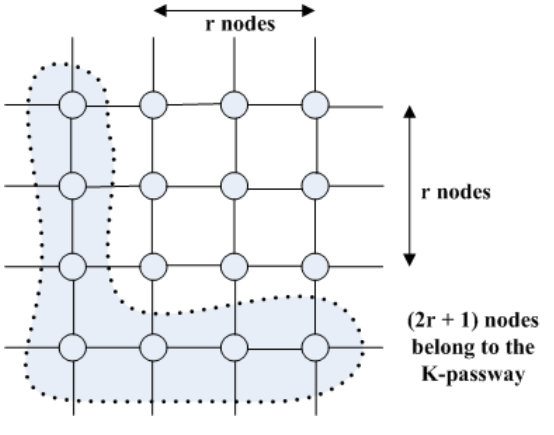


Fig. 4. A section of $(r + 1) \times (r + 1)$ nodes in a large grid network

Let d be a large positive integer. A *d-large grid network* is a network that has d^2 nodes partitioned into 4 *corner nodes*, $4d-8$ *border nodes*, and *middle nodes*. Each corner node has 2 incident links, each border node has 3 incident links, and each middle node has 4 incident links.

Next, we present a construction of a minimal K-observer of a d-large grid network D. An interesting point about this construction is that the number of nodes in the constructed K-observer is $O(d)$ when the value of K is $O(d^2)$.

Our construction takes as input a d-large grid network D and a positive integer K and produces as output a minimal K-observer P of network D. The number of nodes in the constructed P is $O(d^2/\lfloor\sqrt{K}\rfloor)$. The construction proceeds in the following three steps.

Step 1:
Initially, P is the empty set.
Step 2:
Partition the nodes in network D into sections, where each section has $(r+1)^2$ nodes as shown in Figure 4. In Step 3 below, we argue that the value of r is $\lfloor\sqrt{K}\rfloor$. From each section, add the $2r+1$ nodes, that form an L-shape in the section, into set P. Finally, remove the corner node of the L-shape from P.
Step 3:
Any path in network D, whose nodes are not in set P, must be confined to a single section in D. Thus, the length of the longest path in D, whose nodes are not in P, is r^2-1. Therefore, in order to make P a K-observer of D, the value of r needs to be $\lfloor\sqrt{K}\rfloor$.

Theorem 4. *If this algorithm is applied to a d-large grid network D and a positive integer K, then the computed set P by this algorithm is a minimal K-observer of network D.*

Note that the time complexity of this algorithm is linear in the number of nodes in the input d-large grid network.

So far, we could only prove that the computed K-observer P of network D is minimal (not minimum). Luckily, we show next that the number of nodes in the computed P is relatively small when K is $O(d^2)$. But first we need to adopt the following notation.

$|P|$ is the number of nodes in computed set P.
$|D|$ is number of nodes in network D.
$section.P$ is the number of nodes from one section in computed P.
$section.D$ is the number of nodes in one section in network D.

Hence,

$$\begin{aligned} |P| &= (section.P * |D|)/section.D \\ &= ((2r+1) * d^2)/(r+1)^2 \\ &\approx O(d^2/r) \\ &\approx O(d^2/\lfloor\sqrt{K}\rfloor) \end{aligned}$$

Therefore, if K is $O(d^2)$, for example $K = d^2/10$, then $|P|$ is $O(d)$.

6 Applications

The concept of minimal K-observers of a network is a fundamental idea, as it is a generalization of the concept of node cover. This is illustrated by the many applications for which this concept can be put to good use. We mention some of these applications here.

First, minimal K-observers can be used in constructing optimal connectivity paths in wireless sensor networks, wireless ad-hoc networks, and wireless ad-hoc sensor networks. For example, sensor nodes in wireless sensor networks need to maintain connectivity, but use as little energy as possible [6]. A solution to the problem would be to designate some particular nodes as targets, so the other nodes need only get a message to the nearest target node; these special nodes may have access to high power or bandwidth, and can ensure the message is rapidly delivered to its final destination. If we select nodes that constitute a minimal K-observer of the network, and deploy target nodes at these positions, we can guarantee that we choose an optimal number of target nodes to ensure that every "low-power" node has a nearby target (within K hops). Similarly, it may be interesting to choose nodes constituting the minimal K-observer of a content distribution network or of a disruption tolerant network, and place caches at these nodes; each cache can serve the nodes in its "zone", so with a small number of caches we ensure that every node in the network has access to a nearby cache.

Secondly, a minimal K-observer of a network can be used to measure and monitor the network traffic, as it can be considered to form a good number of well spread-out points at which to measure the parameters of the network, such as packet loss and packet delay. Such an arrangement of monitors can also be used for fine-tuning a network; for example, a service provider might deploy monitors at the nodes of a K-observer in order to identify usage accounting or bottlenecks.

Thirdly, minimal K-observers can be used to detect and prevent malicious attacks in the Internet. Thus, if we solve the K-observer problem for a computer network, it can help us to find the optimal places to place firewalls or filters in the network [1], to prevent IP prefix hijacking [15], or to defend against distributed denial of service (DDOS) attacks [3]. Placing "sentry" nodes forming a minimal K-observer ensures that no localized attack can cascade through the network without being detected. Determining the minimal K-observer of a network helps set up many countermeasures to limit the damage done by an attacker. Moreover, it can be used for digital forensics, to track down the adversary after an attack.

It is natural to ask, given the scope of the K-observer problem, whether there is any way to get around the fact that it is intractable in general. In this regard, we note that there exist interesting network topologies (such as trees, rings, and grids) for which the problem is tractable, so we argue that it should be possible to take advantage of various such special cases and compute minimal K-observers (or good approximations thereof) for many networks of practical importance. The problem of identifying more networks for which finding the

minimal K-observer is tractable, as well as finding good heuristics and approximation algorithms, leaves scope for considerable future research.

The other question that must be answered for practical use of the K-observer problem is, "what is the proper choice of K for this application?" It is clear that, the smaller the value of K, the better the coverage (outreach, sensitivity etc.) will be, but as it will require more observer nodes, the solution will be more expensive. The choice of K is dependent upon the domain. In our future work, we intend to extend our study to 'adaptive' systems, where there are many observers but only a small number (K-observer for large K) are active by default; in case an interesting phenomenon is detected, more and more observers are activated (reducing the value of K), improving the resolution precisely when better coverage is required.

7 Concluding Remarks

This paper introduces the concept of minimum K-observers of computer networks. Our primary idea is that the nodes in a minimum K-observer of a network N can be employed as observers of network N. The problem of constructing a minimum K-observer for a general network is NP-hard (Section 2), but we demonstrate that this problem can be solved in linear-time for some special classes of networks: trees (in Section 3), rings (in Section 4), and large grids (in Section 5). Whether the K-observer problem can be solved for Internet-like networks, as specified in [8] and [5], remains an open, but important, question. The K-observer problem is useful in deciding the placement of firewalls, sentry nodes to detect attacks, and so on, and it would be an important contribution to develop algorithms to solve the K-observer problem (at least near-optimally) for networks in the Internet.

There is scope for considerable further work in identifying interesting classes of network for which the K-observer problem is tractable. An important open question that follows from our work is how to devise good heuristics and approximation algorithms for the K-observer problem. Furthermore, it may be noted that we have developed our theory for the K-observer problem under the assumption that the network is stable in time and does not change; it would be interesting to investigate solutions to the problem – and, indeed, to check if the problem is solvable – for time-dependent networks, such as mobile and vehicular networks.

We find that the potential applications for the K-observer problem can be extended from the optimal connectivity for energy efficiency in wireless networks to the efficient node placement for traffic and to the detection and the prevention of network attacks in Section 6. We foresee that the K-observer problem will be helpful to understand and solve many problems in computer networks.

Acnowledgements. The authors would like to express their gratitude to the anonymous reviewer who helped find two major improvements to the paper, and to Mr Keshav Dhandhania of MIT for his help.

References

1. Acharya, H.B., Joshi, A., Gouda, M.G.: Firewall modules and modular firewalls. In: Proceedings of the 18TH IEEE International Conference on Network Protocols, ICNP 2010 (2010)
2. Alzoubi, K.M., Wan, P.-J., Frieder, O.: Message-optimal connected dominating sets in mobile ad hoc networks. In: Proceedings of the 3rd ACM International Symposium on Mobile ad Hoc Networking & Computing, MobiHoc 2002, pp. 157–164. ACM, New York (2002)
3. Armbruster, B., Smith, J.C., Park, K.: A packet filter placement problem with application to defense against spoofed denial of service attacks. European Journal of Operational Research (2007)
4. Breslau, L., Diakonikolas, I., Duffield, N., Gu, Y., Hajiaghayi, M., Johnson, D.S., Karloff, H., Resende, M., Sen, S.: Disjoint-path facility location: Theory and practice. In: Proceedings of the Thirteenth Workshop on Algorithm Engineering and Experiments, ALENEX (2011)
5. Calvert, K., Doar, M.B., Nexion, A., Zegura, E.W.: Modeling internet topology. IEEE Communications Magazine 35, 160–163 (1997)
6. Chen, B., Jamieson, K., Balakrishnan, H., Morris, R.: Span: an energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks. Wirel. Netw. 8, 481–494 (2002)
7. Chvátal, V.: A combinatorial theorem in plane geometry. Journal of Combinatorial Theory, Series B, 39–41 (1975)
8. Faloutsos, M., Faloutsos, P., Faloutsos, C.: On power-law relationships of the internet topology. In: Proceedings of the Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, SIGCOMM 1999, pp. 251–262. ACM, New York (1999)
9. Garey, M.R., Johnson, D.S.: Computers and Intractability; A Guide to the Theory of NP-Completeness. W. H. Freeman & Co., New York (1990)
10. Huang, C.-F., Tseng, Y.-C.: The coverage problem in a wireless sensor network. In: Proceedings of the 2nd ACM International Conference on Wireless Sensor Networks and Applications, WSNA 2003, pp. 115–121. ACM, New York (2003)
11. Karp, R.: Reducibility among combinatorial problems. In: Miller, R., Thatcher, J. (eds.) Complexity of Computer Computations, pp. 85–103. Plenum Press, New York (1972)
12. Lee, S.-J., Su, W., Gerla, M.: Wireless ad hoc multicast routing with mobility prediction. Mob. Netw. Appl. 6, 351–360 (2001)
13. Meguerdichian, S., Koushanfar, F., Potkonjak, M., Srivastava, M.B.: Coverage problems in wireless ad-hoc sensor networks. In: Prodeedings of IEEE INFOCOM, pp. 1380–1387 (2001)
14. Qiu, L., Padmanabhan, V.N., Voelker, G.M.: On the placement of web server replicas. In: Proceedings of IEEE INFOCOM, pp. 1587–1596 (2001)
15. Qiu, T., Ji, L., Pei, D., Wang, J., Xu, J.: Towerdefense: Deployment strategies for battling against ip prefix hijacking. In: Proceedings of the 18th IEEE International Conference on Network Protocols, ICNP (October 2010)
16. Roy, S., Pucha, H., Zhang, Z., Hu, Y.C., Qiu, L.: Overlay node placement: Analysis, algorithms and impact on applications. In: International Conference on Distributed Computing Systems, p. 53 (2007)

17. Wan, P.-J., Alzoubi, K.M., Frieder, O.: Distributed construction of connected dominating set in wireless ad hoc networks. Mob. Netw. Appl. 9, 141–149 (2004)
18. Weber, A.: Theory of the Location of Industries. The University of Chicago Press, Chicago (1909)
19. Wu, J., Li, H.: On calculating connected dominating set for efficient routing in ad hoc wireless networks. In: Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, DIALM 1999, pp. 7–14. ACM, New York (1999)