Broadcasting with the least energy is an NP-complete problem

Wuu Yang, Huei-Ru Tseng, Rong-Hong Jan, and Bor-Yeh Shen
National Chiao-Tung University, Taiwan, Republic of China

Abstract

Energy conservation is an important issue in wireless networks. We propose a method for estimating the least amount of energy needed for broadcasting a message to all nodes in the network. The method can work with any reasonable energy models. We prove that this least-energy problem is NP-complete by showing that the maximum-leaf spanning-tree problem is a special case of the least-energy problem. [1]

Key Words and Phrases: graph theory; least-energy problem; maximum-leaf spanning-tree problem; NP-complete; wireless network

1 Introduction

In wireless networks, devices communicate with radio signal and are powered by battery, which can hold very limited energy. Recharging the battery is usually inconvenient, if not impossible. Many research focuses on conserving battery energy.

A wireless device usually consumes energy on two kinds of work: data transmission and data processing. It is shown that data transmission consumes a major portion of total energy consumption. It is useful to estimate the least energy needed for broadcasting a message in a wireless network. The result could serve as a baseline for comparing power-saving algorithms in wireless transmission.

Conceptually, we attempt to solve the following *coverage* problem:

> *Coverage Problem.* Given a finite number of points on the plane (one of which is designated as the *start point*), a *coverage* is a set of circles such that (1) the circles are centered at (some of) the given points; and (2) the start point directly or transitively *encloses* all other points. We say a point u *encloses* another point v if and only if v is located in a circle centered at u. We wish to find a coverage with the smallest total area.

[1] The work reported in this paper is partially supported by National Science Council, Taiwan, Republic of China, under grants NSC-96-2628-E-009-014-MY3 and NSC96-3114-P-001-002-Y. An earlier, simplified version of this paper will appear in the 2nd International Conference on Multimedia and Ubiquitous Engineering (MUE2008), April 24 - 26, 2008, Busan, Korea.

When all points cluster around the start point, it is advantageous to draw a single circle centered at start point and with a large enough radius to enclose all other points. Alternatively, when all points are located on the same straight line, it would be better to draw several smaller circles to cover all points. In this paper, we will propose a solution to the coverage problem. Our solution is a branch-and-bound approach with clever observations to prune away fruitless search space.

Power management in wireless networks has been a very popular research topic in recent years. Some researchers propose to adjust the transmission power based on the distance of the destination [10, 13, 4]. For the purpose of conserving energy, there are also methods for encoding signals [20], selecting bit rates [11, 14, 5, 7], reducing packet collision [21], dynamically adjusting the sleep periods [6], and routing packets, etc. All these methods are based on heuristics and cannot answer the question of the least amount of energy needed for broadcasting.

There are many books on network and graph algorithms [16, 17, 3]. To our knowledge, there is no discussion of the coverage problem in these works. The coverage problem is generalized into the least-energy problem (in Section 2), which includes the maximum-leaf problem as a special case. It has already been shown that the maximum-leaf problem is NP-complete [2]. We may conclude that the least-energy problem is also NP-complete. (It should be obvious that the least-energy problem belongs to the class of NP.)

There are fast approximation algorithms for the maximum-leaf problem [15, 1, 8, 9]. It would be interesting to adapt these approximation algorithms for the least-energy problem. Salamon and Wiener [12] propose a fast approximation algorithm for the *least*-leaf problem.

The assumptions of our algorithm are similar to those of Wieselthier et al.'s [18]. They propose a heuristic that is based on the Prim's algorithm for finding the minimum spanning tree. Wieselthier et al. [19] also discuss algorithms for multicasting, which is a generalization of broadcasting.

The remainder of this paper is organized as follows: Section 2 will be the algorithm, together with an example and time complexity analysis. Section 3 discusses the issue of NP-completeness. Section 4 concludes this paper.

2 Our Algorithm

In a wireless network, there are n nodes. When a node v wishes to broadcast a message to all other nodes, there are several ways to accomplish the task. For instance, node v can broadcast the message with sufficiently strong transmission power so that every other node can receive the radio signal. A second way is for node v to broadcast with weaker power so that only some nodes can receive the radio signal. A node that receives the radio signal will then re-broadcast the message to the remaining nodes. Different broadcasting schemes consume different amount of energy. We wish to find a broadcasting scheme that consumes the least total energy.

Our energy consumption model is quite simple:

> **Assumption**. The energy required to broadcast a message is equal to d^2, where d is the broadcast distance (or equivalently, the radius of the broadcast circle).

Consider a set of nodes $v_1, v_2, v_3, \ldots, v_n$ with coordinates $(x_1, y_1), (x_2, y_2), (x_3, y_3), \ldots, (x_n, y_n)$, respectively. Assume that v_1 wishes to broadcast a message to all other nodes. (v_1 is called the *start* node.) We will use an undirected, complete graph to represent the network.

We use a graph G to represent the wireless network. Each node in G represents a node in the wireless network. There are n nodes in G, where n is the number of nodes in the wireless network. Each edge in G represents a direct channel between two nodes in the network. We assume that a node may broadcast with arbitrarily strong power. Hence, there is a direct channel between every pair of nodes and G is, therefore, the complete graph K_n. We also assume that radio transmission is symmetric, that is, node u sends a message to node v in exactly the same way as node v sends a message to u. Therefore, the graph G is undirected.

On each edge (x, y), there is a *cost*, which is the energy needed for node x to broadcast a message to node y. According to our energy model, the cost of an edge is the square of its length. If a different model of energy consumption is adopted, only the edge costs need modification. Our algorithm (shown below) is still applicable.

Our objective is to find a broadcasting scheme with the least total cost. Before we present our algorithm, there are a few observations that we can depend on.

Lemma 1 *Every node broadcasts at most once in an optimal scheme.*

If a node broadcasts the message twice, with power p_1 and p_2, respectively, and assume $p_1 \geq p_2$, the broadcast with power p_2 can be omitted.

Lemma 2 *The upper bound of an optimal broadcast scheme is l^2, where l is the longest distance from the start node to any other node.*

begin *main*
$upperbound := l^2$;
$currentCost := 0$;
$R := \{v_1\}$;
$N := \{v_2, v_3, \ldots, v_n\}$;
$currentScheme := [\quad]$;   /* a null sequence initially */
$bestSchemeUpToNow := [(v_1, v_k)]$,
        where $cost((v_1, v_k)) \geq cost((v_1, v_j))$ for $j = 2, 3, \ldots n$;
$tryNext(R, N, currentCost, currentScheme)$;
print $upperbound$, $bestSchemeUpToNow$;
end *main*

Figure 1. The *main* procedure.

Our algorithm, shown in Figures 1 and 2, is recursive. During each invocation of the $tryNext$ procedure, some of the nodes had broadcast the message before; some other nodes had already received the message but had not re-broadcast it yet; others had neither received nor broadcast the message. Due to Lemma 1, nodes that had already broadcast the message do not need to broadcast or receive the same message again. They can be safely ignored. (This is done in line 14 in the $tryNext$ procedure.)

Thus, the $tryNext$ procedure is concerned with the other two kinds of nodes: R (nodes having received the message but having not re-broadcast it yet) and N (nodes having not received the message). Edges between nodes in R are ignored because nodes in R have already received the message; they do not need to receive the message again. Furthermore, edges between nodes in N are also ignored because nodes in N have not received the message; they cannot send out any message. Only edges from nodes in R to nodes in N need to be considered. The $tryNext$ procedure tries all such edges one by one. Therefore, each recursive call of $tryNext$ starts from a complete bipartite graph (R, N)[2]. In the very first invocation of the $tryNext$ procedure, $R = \{v_1\}$ (i.e., the start node) and $N = \{v_2, v_3, \ldots, v_n\}$.

During each invocation, an edge is selected. This means that (1) the node of the edge that is in R broadcasts the message; (2) the transmission power is the cost of the edge; and (3) all and only nodes in N that fall inside the circle covered by the transmission power will receive the message. The nodes that receive the message (which are the elements of the *moveSet* in the $tryNext$ procedure) will be moved from N to R. At the end of an invocation, we will have a new bipartition, denoted as

[2] A graph, denoted as (R, N), is *complete bipartite* if (1) the nodes of the graph are partitioned into two (disjoint) sets R and N, and (2) there is an edge between every node in R and every node in N. There is no other edge.

```
procedure tryNext(R, N, currentCost, currentScheme)
local var estimatedCost, moveSet, newR, newN, newScheme;
if N = ∅ then begin
    /* found a scheme with cost ≤ upperbound */
    upperbound := currentCost;
    bestSchemeUpToNow := currentScheme;
    return;
    end
for each edge (v_R, v_N) in the bipartite graph (R, N) in the
        order of increasing cost do begin
    estimatedCost := currentCost + cost((v_R, v_N));
    if estimatedCost > upperbound then return;
    moveSet := {w ∈ N | cost((v_R, w)) ≤ cost((v_R, v_N))};
    newR := R − {v_R} ∪ moveSet;
    newN := N − moveSet;
    newScheme := currentScheme || [(v_R, v_N)];
        /* append the edge (v_R, v_N) to currentScheme. */
    tryNext(newR, newN, estimatedCost, newScheme);
end for
end procedure tryNext
```

Figure 2. The *tryNext* procedure.

$(newR, newN)$, on which another invocation will be perform. $tryNext$ stops when N becomes an empty set.

In summary, our algorithm consists of recursive calls. During each invocation, we will choose a node in R and suitable transmission power (i.e., suitable radius) and draw a circle around the chosen node. All nodes covered by the circle are moved to R.

Our approach to choosing a node and suitable radius is branch and bound. We will try every edge from a node in R to a node in N, in the order of increasing costs. (It is equally fine to use any other convenient order.)

There is a trivial upper bound on the total cost according to Lemma 2. Further search along a path is pruned off whenever the estimated cost exceeds the current upper bound. When a scheme is found, the upper bound is replaced by the cost of the scheme (which should be no more than the current upper bound). Since we aim at a scheme with the least cost, we need to examine all possible schemes. According to Lemma 2, the initial value of the upper bound is l^2, where l is the longest distance from the start node to any other node.

A *scheme* is represented as a sequence of edges such as $[(u_1, u_2), (u_3, u_4), \ldots, (u_5, u_6)]$. This scheme means that u_1 first broadcasts the message with transmission power equal to the cost of the edge (u_1, u_2). Then u_3 broadcasts the message with transmission power equal to the cost of the edge (u_3, u_4). Similarly for other edges in the scheme. Finally, u_5 broadcasts the message with transmission power equal to the cost of the edge (u_5, u_6). We adopt the convention that, when an edge (u, v) is included in a scheme, it is the first node, that is, node u, who broadcasts the message.

The two variables *upperbound* and *bestSchemeUpToNow* are global variables. They are used in both the `main` procedure and the *tryNext* procedure.

Due to the choice of the edge (v_R, v_N) (line 9), the set *moveSet* (which contains the set of nodes that will receive the radio signal and hence should be moved from N to R) contains at least one node (i.e., v_N). Therefore, *newN* is a proper subset of N. This means that the second argument (i.e., N) of the *tryNext* procedure gets smaller and smaller in deeper and deeper nested recursive calls. Eventually, N becomes an empty set. Therefore, the recursion is finite. Actually, the recursion can be at most $n - 1$ levels deep, where n is the number of nodes in the wireless network.

Lemma 3 *The procedure tryNext always terminates for finite wireless networks.*

Example. Suppose that there are four nodes in a wireless network whose coordinates are $v_1(0, 0), v_2(1, 0), v_3(1, 1), v_4(3, 0)$, respectively. Figure 3 shows the various recursive calls of *tryNext*. In the beginning, $R = \{v_1\}$, $N = \{v_2, v_3, v_4\}$, and *upperbound* $= 9$. In the first call to *tryNext*, we will choose the edge (v_1, v_2) (whose cost is 1). Then $newR = \{v_2\}$ and $newN = \{v_3, v_4\}$. In the second call to *tryNext*, we will choose (v_2, v_3) (whose cost is 1). Then $newR = \{v_3\}$ and $newN = \{v_4\}$. In the third call to *tryNext*, there is only one edge (v_3, v_4) (whose cost 5). At this time, a scheme is found: $[(v_1, v_2), (v_2, v_3), (v_3, v_4)]$ (whose cost is 7).

After finding the scheme, we will backtrack to the second call and choose anther edge (v_2, v_4) (whose cost is 4). Then $newR = \{v_3, v_4\}$ and $newN = \emptyset$. At this time, a better scheme is found: $[(v_1, v_2), (v_2, v_4)]$ (whose cost is 5).

After finding the scheme, we will backtrack to the first call and choose anther edge (v_1, v_3) (whose cost is 2). Then $newN = \{v_2, v_3\}$ and $newN = \{v_4\}$. In the fourth, we will choose the edge (v_2, v_4) (whose cost is 4). At this time, the estimated cost is $2 + 4$, which exceeds the lowest cost of the schemes already found (which is 5). Further search along this path is pruned off.

Again we will backtrack to the first call and choose the last edge (v_1, v_4) (whose cost is 9). Since the estimated cost is 9, which exceeds the lowest cost of the schemes already found (which is 5), further search along this path will be pruned off. Since all edges are explored in the first call, the algorithm stops. □

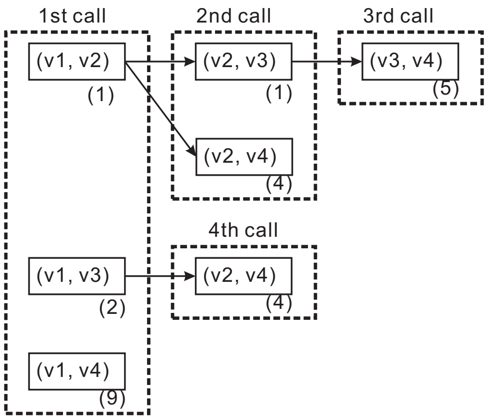

Figure 3. The call tree. A dashed rectangle denotes an invocation. The small solid rectangles inside a dashed rectangle denote iterations of the for loop.

Complexity analysis. To analyze the complexity of the above algorithm, we consider the call tree similar to that in Figure 3 (in which the root is on the left and leaves are on the right). The degrees of the nodes in the call tree is equal to the numbers of edges in the `for` loop of the $tryNext$ procedure, which is the number of edges in the complete bipartite graph (R, N), which is at most $k^2/4$ if $|R| + |N| \leq k$. Note that, in the first recursive call of $tryNext$, $|R| + |N| = n$. Whenever the recursion goes one level deeper, $|R| + |N|$ is reduced by 1 (due to the exclusion of v_R in line 14 of $tryNext$ in Figure 2). The depth of the recursive calls of the $tryNext$ procedure is at most n. Therefore, the total number of invocations of the $tryNext$ procedure is at most $\Pi_{k=1}^{n} \frac{k^2}{4}$, which is $O((n!)^2/4^n)$. (There is also additional time needed for enumerating all edges *in increasing order* (line 9) and for computing the $moveSet$ (line 13). The additional time is comparatively very small and is hence ignored.) It would be interesting to investigate more efficient algorithms for the coverage problem.

The above time complexity is grossly over-estimated. The number of recursive invocations can be significantly reduced if we take into account the fact that when node u can receive radio signals from node v, all other nodes u' that are closer to v than u can also receive the same radio signals.

The depth of the call tree is at most $n - 1$. On each level of recursive calls, we need to maintain a bipartite graph (R, N), which contains at most $\frac{n^2}{4}$ edges. Hence, the total space needed is $O(n^3)$.

2.1 Practical Variations

In practice, the energy of a single broadcast operation is limited. This means that there is an upper bound on the distance a node can reach via radio broadcast. It is easy to accommodate this constraint in our algorithm: When two nodes are too far apart, there will be no edges connecting them. In other words, we will start from a subgraph of the complete graph K_n.

Simple wireless devices may lack the ability to dynamically adjust the broadcasting power. They always broadcast with the same energy. For this constraint, we can draw a graph in which there is an edge between two nodes if and only if their distance is no more than the radius of the circle determined by the fixed broadcasting power. All edges will carry the same cost.

2.2 NP-Completeness of the Least-Energy Problem

In summary, the above algorithm aims at the *least-energy problem* which is defined as follows:

> *Least-Energy Spanning Tree Problem.* Consider an undirected simple graph in which the cost of an edge is a positive real number. One of the nodes is designated as the *start node*. For each spanning tree of this graph, the *energy* of a node is defined as the maximum of the costs of all the edges incident from that node. (Note that, in a spanning tree, an edge is considered to be incident *from* a parent *to* its child.) By convention, the energy of a leaf node is always 0. The *energy* of a spanning tree is defined as the sum of the energy of all the nodes. The objective is to determine whether the energy of a *least-energy spanning tree* rooted at the designated start node is no more than k, where k is a given constant.

If the cost of every edge in the graph is always 1, the energy of a spanning tree is equal to the number of non-leaf nodes in the tree. A least-energy spanning tree is a spanning tree with the least number of non-leaf nodes. Equivalently, it is a spanning tree with the maximum number of leaves. Therefore, in this case, the *least-energy problem* becomes the *maximum-leaf spanning-tree problem.* It has already been shown that the maximum-leaf problem is NP-complete [2]. We may conclude that the least-energy problem is also NP-complete. (It should be obvious that the least-energy problem belongs to the class of NP.)

3 Conclusion

Since energy conservation is important in wireless networks, it is useful to establish a lower bound of the energy consumed in transmission. This paper formalized the problem as the least-energy problem and proposed a solution for it. Since the time complexity of our algorithm is quite high, it would be interesting to investigate efficient approximation algorithms for the least-energy problem in the future.

References

[1] P.S. Bonsma, T, Brueggemann, and G. J. Woeginger, "A Faster FPT Algorithm for Finding Spanning Trees with Many Leaves," in *Proceedings of Mathematical foundations of computer science* (MFCS03, Bratislava, 2003), LNCS 2747, pp. 259-268, 2003.

[2] M.R. Garey and D.S. Johnson, Computers and Intractability, W.H. Freeman and Co., San Francisco, 1979.

[3] A. Gibbons, Algorithmic Graph Theory Cambridge UP, Cambridge, England, 1985.

[4] S. Gitzenis, N. Bambos,"Efficient data prefetching for power-controlled wireless packet networks," in *Proceedings of The First Annual International Conference on Mobile and Ubiquitous Systems: Networking and Services*, pp. 64-73, August 2004.

[5] C. S. Hsu, J. P. Sheu, Y. C. Tseng,"Minimize waiting time and conserve energy by scheduling transmissions in IEEE 802.11-based ad hoc networks," in *Proceedings of 10th International Conference on Telecommunications*, vol. 1, pp.393-399, Feb 2003.

[6] S.-C. Huang, R.-H. Jan, and W. Yang, "ALPS: A Loading-aware Self-configuration Power-saving Schedule Algorithm for Wireless Ad Hoc Networks," in preparation, 2007.

[7] J. Jelitto, A. N. Barreto, H. L. Truong,"Power and rate adaptation in IEEE 802.11a wireless LANs," in *Proceedings of Proceedings of 57th IEEE Semiannual Vehicular Technology Conference*, vol. 1, pp.413-417, April 2003.

[8] H.-I Lu and R. Ravi, "The power of local optimization: Approximation algorithms for maximum-leaf spanning tree," CS-96-05, Computer Science Department, Brown University, January 1996.

[9] H.-I Lu and R. Ravi, "Approximating Maximum Leaf Spanning Trees in Almost Linear Time," Journal of Algorithm, vol. 29, pp. 132-141, 1998.

[10] C. A. S. Oliveira, P. M. Pardalos,"A distributed optimization algorithm for power control in wireless ad hoc networks," in *Proceedings of Proceedings of 18th International Parallel and Distributed Processing Symposium*, pp. 177, April 2004.

[11] G. Razzano, L. Andreani, R. Cusani, "Wireless LANs: an adaptive algorithm to reduce power consumption," in *IEEE International Conference on Communications (ICC '03)*, vol. 2, pp. 1096-1100, May 2003.

[12] Gábor Salamon and Gábor Wiener, "On finding spanning trees with few leaves," Information Processing Letters, vol. 105, no. 5, pp. 164-169, 2008.

[13] A. Sheth, R. Han,"Adaptive power control and selective radio activation for low-power infrastructure-mode 802.11 LANs," in *Proceedings of 23rd International Conference on Distributed Computing Systems Workshops*, pp. 812-818, May 2003.

[14] S.T. Sheu, J. Chen,"MR^2RP: the multi-rate and multi-range routing protocol for ad hoc wireless networks," in *Proceedings of 7th International Symposium on Computers and Communications*, pp. 435-440, July 2002.

[15] R. Solis-Oba, "2-Approximation Algorithm for Finding a Spanning Tree with Maximum Number of Leaves," in *Proceedings of the 6th Annual European Symposium on Algorithms* (ESA), LNCS 1461, pp. 441-452, 1998.

[16] R.E. Tarjan, Data Structure and Network Algorithms, SIAM, Philadelphia, Penn., 1983.

[17] W.T. Tutte, Graph Theory, Addison-Wesley, Reading, MA, 1984.

[18] J.E. Wieselthier, G.D. Nguyen, and A. Ephremides, "On the Construction of energy-Efficient Broadcast and Multicast Trees in Wireless Networks," in *IEEE Infocom'2000.*

[19] J.E. Wieslthier and G.D. Nguyen, "Algorithms for Energy-Efficient Multicasting in Static Ad Hoc Wireless Networks, " Mobile Networks and Applications vol. 6, pp. 251V263, 2001.

[20] M.L. Wu, K.A. Wen, C.W. Huang, "Low power pseudo noise codec design for spread spectrum wireless communication," in *Proceedings of 42nd Midwest Symposium on Circuits and Systems*, vol. 1, pp. 92-95, Aug 1999.

[21] W. Ye, J. Heidemann, D. Estrin,"Medium access control with coordinated adap-

tive sleeping for wireless sensor network," in *IEEE/ACM Transactions on Networking*, vol. 12, Issue 3, pp. 493-506, June 2004.