Computing the Maximum Order of an Open Irredundant Set in a Tree

Arthur M. Farley
and
Andrzej Proskurowski

Department of Computer and Information Science
University of Oregon
Eugene, Oregon 97403

Abstract

The study of open irredundant sets of vertices in a graph is of interest from both practical and theoretical perspectives. On the one hand, such sets correspond to possible groups of simultaneous senders in broadcast networks. On the other, their maximum size represents a new parameter on graphs. Computing a maximum irredundant set in an arbitrary graph is known to be difficult. This motivates development of an efficient algorithm on tree-structured graphs. Here, we present and prove correct such an algorithm, emphasizing a development technique whereby a general algorithm is specialized for the problem at hand, proven correct, and then refined to reduce its complexity.

1. Introduction

In a broadcast network, messages are transferred between communication sites by being transmitted over a common medium (channel). One site is said to be in range of another site if it can possibly receive the others site's transmissions. A message transmitted by one site is received by another site only if that other site is in range and, at the same time, not transmitting nor within range of another transmitting site. If either of the secondary conditions hold, a transmission collision occurs and no usable (i.e., decipherable) message is received.

According to these constraints, at any time during active operation of a broadcast network N, each communication site can be said to be in one of four possible states:

S -- a sender, having at least one site in range that receives its transmission;

T -- an unsuccessful transmitter, having no site in range not experiencing a collision;

R -- a receiver, being in range of a single sender;

D -- dormant, neither transmitting nor successfully receiving.

A possible network state of a broadcast network is an association of one state with each communication site in a manner consistent with the above state definitions. A network state represents a complete, instantaneous situation that could occur during communication operations in the network. The sites that are in state S are referred to as senders; those in R are receivers. Initial motivation for our study was to characterize and compute maximum sets of senders in a given, arbitrary broadcast network. The maximum number of sites in a sending set represents an upper-bound on the message capacity of a broadcast network, being the maximum number of different messages that can ever be successfully transmitted simultaneously in the network.

We model a broadcast network N by a simple, undirected graph G = (V,E) consisting of a set V of vertices, representing sites of N, and a set E of edges, each being an unordered pair of vertices (uv) such that if (uv) is an edge of G then the sites represented by u and v are within range of each other in N. Vertices u and v are adjacent iff (uv) is in E. Note we assume that the in range relation is symmetric; this allows G to be undirected.

A consistent labeling of G is an assignment of labels S, R, T, and D to vertices of G that corresponds to the states of related sites from a possible network state of N. A consistent labeling directly represents that network state. A set of vertices is a sending set iff there exists a consistent labeling of G such that each vertex in the set is labeled S. Vertices of a sending set represent sites that can simultaneously transfer messages. Possible receivers of those messages are sites represented by vertices adjacent to elements of the sending set. The open neighborhood N(v) of vertex v is the set of vertices adjacent to v. The vertices in N(v) are called the neighbors of v. The closed neighborhood N[v] of v is equal to N(v) + {v}, formed by adding v to its open neighborhood. The composite open neighborhood CN(U) of a set of vertices U is equal to the union of the open neighborhoods of vertices in U; the composite closed neighborhood CN[U] is defined analogously.

In looking for graph-theoretic notions related to sending set, it is natural to consider those that can be defined in terms of the above neighborhood concepts. Independent and dominating sets are two well-known notions. An independent set I is a set of vertices such that if u and v are in I then u is not in N(v) (and vice versa). In other words, no two members of I are neighbors. A dominating set D is a set of vertices such that CN[D] = V, the set of vertices of G. In other words, every vertex of G is either in D or is a neighbor of a vertex in D. A vertex in D is said to dominate itself and its neighbors. Finally, a total dominating set TD is a set of vertices such that CN(TD) = V, as defined in [2]. Every vertex of G is a neighbor of, and thus dominated by, a member of TD. In [5] we demonstrate that these vertex sets are not equivalent to sending sets.

A recently introduced notion more closely corresponding to that of sending set is the irredundant set [1]. An irredundant set IR is a set of vertices such that if v is in IR then N[v] - CN[IR-{v}] is not empty. The vertices in N[v] - CN[IR-{v}] are called the private vertices of v. If we view a vertex in IR as a member of a sending set, its private neighbors correspond to receivers of its transmission. However, a vertex in IR may have itself as sole private vertex, whereas each member of a sending set must have at least one private neighbor. As such, we define an open irredundant set OIR to be a set of vertices such that if v is in OIR then N(v) - CN[OIR-{v}] is not empty. By this definition, each vertex in OIR is guaranteed to have at least one private neighbor, i.e., a neighbor that is not adjacent to any other member of OIR. This slight change in the definition of an irredundant set, now considering only the open neighborhood of a member, has given us a new graph-theoretic concept and one that is equivalent to our notion of sending set, as proven in [5].

The maximum and minimum orders of maximal open irredundant sets have been compared with those of maximal independent, minimal dominating, and maximal irredundant sets. A maximal open irredundant (independent, irredundant) set is such a set not contained in any other such set. A minimal dominating set is a dominating set such that no strict subset of its vertices is also a dominating set. We define the parameters i(G) (I(G)), d(G) (D(G)), ir(G) (IR(G)), and oir(G) (OIR(G)) to be the minimum (maximum) orders of maximal independent, minimal dominating, maximal irredundant, and maximal open irredundant sets of G, respectively. Cockayne, Hedetniemi, and Miller [1] note the following inequality sequence:

$$ir(G) \leq d(G) \leq i(G) \leq I(G) \leq D(G) \leq IR(G).$$

Cockayne, Favaron, Payan, and Thomason [3] discuss further relationships among these six parameters, describing a class of graphs for which all of the parameter values differ (i.e., where strictly less-than holds). We proved in [5] that OIR(G) is related to this sequence of inequalities as follows:

$$d(G) <= OIR(G) <= IR(G).$$

OIR(G) is equal to the number of communication sites in a largest sending set of a broadcast network corresponding to G. This, in conjunction with the above discussion, motivates the study of open irredundant sets and determination of the largest of these sets for a given graph. In [4] Even, Goldreich and Tong prove that the problem of determining the maximum order of an open irredundant set in an arbitrary graph is computationally difficult (NP-complete, see [9]). Such a result prompts search for efficient algorithms that either determine an approximate solution for an arbitrary graph or that determine an exact solution for a restricted class of graphs. In the following, we present results of pursuing the latter course, wherein we define and prove correct an efficient, exact algorithm solving our problem for trees (i.e., acyclic graphs).

2. An Efficient Algorithm for Trees

We present the development of a solution algorithm in a manner illustrating a paradigm we have found particularly useful for defining and verifying algorithms on trees. In recent years, we have designed efficient algorithms on trees computing parameters that for arbitrary graphs are known to be difficult to determine [6,7,8]. As a result of this experience, we have developed a paradigm for defining such algorithms in which a general, bottom-up tree traversal is specialized with respect to the problem at hand, the resultant algorithm is proved correct, and finally refined to reduce its complexity by taking advantage of properties true of the particular parameter for trees. The general algorithm on trees is as follows:

General Tree Algorithm

Purpose:

To compute a parameter of interest for a given, arbitrary tree T.

Information Structures:

Let the current tree CT be T.
Associate with each vertex v of T a set of state-dependent variables {SDV1(v), ..., SDVk(v)}.

Method:

```
Step 1.]  {Process leaf vertices.}
          Until only one vertex remains do
            (i) Select leaf vertex v from CT.
            (ii) Determine values for
                          SDV1(v), ..., SDVk(v).
            (iii) Remove v from CT.
Step 2.]  {Process final vertex.}
          With v being the remaining vertex of CT do
            (i) Determine values for
                          SDV1(v), ..., SDVk(v).
            (ii) Determine the value of the parameter
                          of interest by considering
                          SDV1(v), ..., SDVk(v).
```

At any time, the current tree CT is the as yet unprocessed subtree of T. A leaf of CT is a vertex having only one neighbor in CT. To specialize the general algorithm, a set of state-dependent variables must be defined. We then must indicate how their values are to be determined for a leaf as it is processed (Step 1) and how the value of the parameter of interest is determined from the state-dependent variables of the last remaining vertex (Step 2).

We specialize the general algorithm for the problem of computing the maximum order of an open irredundant set in the following way. As each leaf v of CT is processed in Step 1, it separates from CT a subtree Tv of previously processed vertices of T. Vertex v can be in one of five states with respect to an open irredundant set OIR in T:

```
IB  --  in OIR, with a private neighbor in Tv;
IA  --  in OIR, with a private neighbor not in Tv;
OB  --  out of OIR, required as a private neighbor
            of a vertex of OIR in Tv;
OC  --  out of OIR, as a neighbor (not required as
            private neighbor) of one or more
            vertices of OIR in Tv;
ON  --  out of OIR, not a neighbor of any vertex
            of OIR in Tv.
```

Based on these states, we associate with each vertex v the five state-dependent variables IB(v), IA(v), OB(v), OC(v), and ON(v). As vertex v is processed, each variable is assigned the maximum order of an open irredundant set in Tv, given that v is in the corresponding state. If v is a leaf of the original tree T, IB(v) = OB(v) = OC(v) = ON(v) = 0 and IA(v) = 1. Otherwise, the values of the state-dependent variables are determined in terms of the values associated with the neighbors v1, ..., vk of v in Tv as follows:

$$IB(v) = 1 + \max_i [ON(vi) + \sum_{j \neq i} \max (IB(vj), OC(vj), ON(vj))]$$

$$IA(v) = 1 + \sum_i \max (IB(vi), OC(vi), ON(vi))$$

$$OB(v) = \max_i [IA(vi) + \sum_{j \neq i} \max (OB(vj), OC(vj), ON(vj))]$$

$$OC(v) = \max_i [IB(vi) + \sum_{j \neq i} \max (IB(vj), OC(vj), ON(vj))]$$

$$ON(v) = \sum_i \max (OB(vi), OC(vi), ON(vi)).$$

Finally, when considering the last vertex v (Step 2), the maximum order of an open irredundant set in T is the maximum of IB(v), OB(v), OC(v), and ON(v).

Theorem 1. The General Tree Algorithm, specialized as described above, computes the maximum order of an open irredundant set for a given tree T.

Proof: The graph T-CT is a forest of trees Ti, that is initially empty. Each Ti has a distinguished vertex wi adjacent to a vertex in CT, being the last vertex of Ti removed from CT in Step 1. The invariant relation to be established for Step 1 is that each state-dependent variable value associated with the wi is equal to the maximum order of an open irredundant set in Ti, when wi is in the corresponding state. This is vacuously true prior to execution of Step 1. As a vertex v is processed, it is either a leaf of T or is adjacent to previously removed vertices {v1, ..., vk} of the set of distinguished vertices. In the first case, the assignment of values as indicated above is obviously correct, where IB(v)=0 indicates that v can not be a private neighbor of a previously removed vertex as no such vertex exists; as v is removed from CT, it is added to the set of distinguished vertices. In the second case, each state of v is possible in Tv only if the states of v1, ..., vk are constrained as indicated in the expressions given above. The state of a vertex directly limits the allowable states of neighboring vertices only, as open irredundance is defined in terms of (local) neighborhoods. The expressions given above optimize over the selection of allowable states for neighbors and thus determine the maximum orders of the corresponding open irredundant sets in the newly processed subtree of T. The vertices {v1, ..., vk} are removed from the set of distinguished vertices while v is added to it (as the distinguished vertex of the newly created, processed subtree). Upon completion of Step 1, CT consists of a single vertex v with its neighbors constituting the set of distinguished vertices. First, the values for the state-dependent variables of v are computed. The only impossible state for v is IA; the maximum order of an open irredundant set for T is determined by taking the maximum of the values associated with the other four states. []

Now that we have a correct solution algorithm, we can turn our attention to refinements that can reduce time and space complexity while maintaining correctness. The transformations we make for this problem are all based upon the following property of open irredundant sets in an arbitrary graph.

Theorem 2. {Duality Property for Open Irredundant Sets}
Given an open irredundant set OIR = {v1, ..., vk} of vertices in a graph G such that each vertex vi has private neighbors PN(vi), the set OIR′ = {v1′, ..., vk′} such that each vi′ is a member of PN(vi) is also an open irredundant set of vertices in G.

Proof: From the definition of private neighbor, no vertex vi of OIR has a vertex vj′ of OIR′ as neighbor, for i≠j. Hence, in OIR′ each vertex vi′ has vi as a private neighbor. []

We refer to the action of forming a new open irredundant set OIR′ from a given open irredundant set OIR by selecting private neighbors of vertices in OIR as a shift operation, since each vertex in OIR is replaced by a neighbor. Note that OIR′ is of the same order as OIR. The Duality Property allows us to establish the following relationships among the values of the five state dependent variables.

Theorem 3. For a vertex v that has been processed in Step 1 of the specialized General Tree Algorithm,

(i) OB(v) = IB(v);
(ii) OC(v) = ON(v);
(iii) IA(v) - 1 = ON(v);
(iv) ON(v) ≤ IB(v) ≤ ON(v) + 1.

Proof: If v is a leaf of T, then the relationships all hold. Otherwise, as v is processed, it is the root of subtree Tv of processed vertices with processed neighbors v1, ..., vk. Given an open irredundant set associated with one of the variable values, we are able to apply one or more shift operations in Tv or in subtrees rooted at the processed neighbors to realize an open irredundant set associated with a related value. In each of the cases below, we transform a set associated with the value on the left-hand side of the relation into the one on the right-hand side. The opposite direction is realized by direct inverse of this transformation.

(i) Apply a shift to Tv, selecting v as the private neighbor (of its only neighbor in the given set) to be in the new set.

(ii) For any processed neighbor vi of v that is in the given set (there must be at least one such vi, as v is OC), shift the subtree rooted by vi.

(iii) Remove v from the given set and for any processed neighbor of v that is also in the given set, shift its subtree.

(iv) Consider the case where v is in state ON. If there exists an open irredundant set associated with the value ON(v) for which not all the neighbors of v are in state OB, then IB(v) = ON(v) + 1 by shifting subtrees rooted at all OB neighbors to make them IB neighbors and adding v to the given set, claiming one of the other neighbors as private (possibly using relation ii and its transformation described above). Otherwise, v can be added to the set only by claiming one of its OB neighbors as private and losing a vertex from the set in the subtree rooted at that neighbor. []

The above relationships allow us to significantly simplify expressions to determine state-dependent variable values. First, we can eliminate IA(v), OC(v), and OB(v) as being redundant. The expressions for determining IB(v) and ON(v) now are:

$$IB(v) = 1 + \max_i \; [ON(vi) + \sum_{j \neq i} IB(vj)];$$

$$ON(v) = \sum_i IB(vi).$$

The expression for ON(v) appears counterintuitive, but follows as IB(v) = OB(v) and ON(v) = OC(v) and IB(v) $\geq$ ON(v). As a final refinement, we realize that IB(v) will be one more than ON(v) in those cases where ON(vi) = IB(vi) for some processed neighbor vi of v. We can select such a neighbor to be the private neighbor of v (assuming it to be in state ON) without losing a vertex from the set in that processed subtree. As such the following process computes the values of IB(v) and ON(v) from the values associated with {v1, ..., vk}. The boolean variable selected indicates whether neighbor vi was found with IB(vi) = ON(vi), allowing us to add 1 to IB(v).

```
procedure Compute-Values
  begin on := 0;
        selected := false
        for i := 1 to k do
          begin on := on + IB(vi);
                if IB(vi) = ON(vi)
                    then selected := true
          end;
        ON(v) := on;
        if selected then IB(v) := on + 1
                    else IB(v) := on
  end
```

Theorem 4. The specialized tree algorithm as refined above requires time and space linearly related to the order of the input tree.

Proof: Assuming unit costs for storing integers of the same order as the input tree, a constant overhead per vertex implies linear storage requirements. As for time complexity, the two values at each vertex are referenced twice: once when the vertex is processed and the values are established and again, after being processed, when its sole neighbor in CT is being processed. Assuming that assignment and reference to these values require constant time, the computation time is linearly related to the order of the input tree. []

3. Conclusion

In this paper, we have discussed a problem motivated by the study of broadcast networks. The problem is that of determining maximum sets of senders in such networks. We define a graph-theoretic model of the problem, introducing the new notion of open irredundant sets of vertices. We compare the orders of open irredundant sets with other graph parameters and establish a duality property for open irredundant sets that is useful in computing their maximum orders. Finally, we define an efficient algorithm computing the maximum order of such sets in trees, illustrating an important algorithm development technique in the process.

An irredundant set is defined by existence of a non-empty difference between the neighborhood of any vertex in the set and the composite neighborhood of all other vertices in the set. Considering an open or a closed neighborhood in either case gives us four classes of irredundancy. Closed-closed corresponds to the original concept as studied by Cockayne et al.; open-closed corresponds to open irredundancy as defined here. Open-open would correspond to sets of senders in broadcast networks where a sender could ignore (subtract out) its own transmissions, allowing it to receive while transmitting. Finally, closed-open is the largest of the irredundant sets in any given graph. The maximum orders of the four classes of irredundant sets in an arbitrary graph are related, due to class inclusion, as follows:

open-closed <= (closed-closed and open-open) <= closed-open.

No necessary inequality sequence holds between the orders of closed-closed and open-open irredundant sets. A comparative study of the different classes of irredundant sets represents an interesting topic for future research as does the definition of algorithms to compute their maximum orders.

References

[1] Cockayne, E.J., Hedetniemi, S.T., and Miller, D.J. "Properties of Hereditary Hypergraphs and Middle Graphs", Canadian Math. Bulletin, 21(4), 1978, 461-468.

[2] Cockayne, E.J., Dawes, R.M., and Hedetiniemi, S.T. "Total Domination in Graphs", Networks, 10, 1980, 211-219.

[3] Cockayne, E.J., Favaron, O., Payan, C., and Thomason, A. "Contributions to the Theory of Domination, Independence, and Irredundance in Graphs", Discrete Mathematics, 33(3), 1981, 249-258.

[4] Even, S., Goldreich, O., and Tong, P. "On the NP-Completeness of Certain Network-Testing Problems", Technical Report 230, Computer Science Department, Technion, Haifa, Israel, 1981.

[5] Farley, A.M. and Schacham, N. "Senders in Broadcast Networks: Open Irredundancy in Graphs", presented at the Thirteenth Southeastern Conference on Combinatorics, Graph Theory, and Computing, in Congressus Numerantium, 38, Utilitas Math., 1983, 47 - 57.

[6] Farley, A.M. and Proskurowski, A. "Computation of the Center and Diameter of Outerplanar Graphs", Discrete Applied Math., 2, 1980, 185 - 191.

[7] Farley, A.M., Hedetniemi, S.T. and Proskurowski, A. "Partitioning Trees: Matching, Domination, and Maximum Diameter", Int. J. Computer and Information Science, 10, 1981, 55 - 61.

[8] Farley, A.M. and Proskurowski, A. "Broadcasting in Trees with Multiple Originators", SIAM Journal of Algebraic and Discrete Methods, 2, 1981, 381 - 386.

[9] Garey, M. and Johnson, D.S. Computers and Intractability: A Guide to the Theory of NP-Completeness, Freeman:San Francisco,CA, 1979, 1 - 50.