

# ON MINIMUM CUTS AND THE LINEAR ARRANGEMENT PROBLEM

by

S.B. Horton\*, R. Gary Parker<sup>†</sup> and R. B. Borie<sup>‡</sup>

## Abstract

In this paper we examine the problem of finding minimum cuts in finite graphs with the side constraint that the vertex sets inducing these cuts must be of a given cardinality. As it turns out, this computation is of interest not only from a combinatorial perspective but also from a practical one, pertaining to the linear arrangement value of graphs. We look at some graph classes where these cuts can be efficiently computed (in general this computation is NP-hard) as well as some cases where their value can be determined in closed form.

\*Department of Mathematical Sciences, United States Military Academy, West Point, NY, 10996.

<sup>†</sup>School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA, 30332-0205.

<sup>‡</sup>Department of Computer Science, University of Alabama, Tuscaloosa, AL, 35487-0290.

**Key Words:** Cuts, linear arrangement, bounds.

# 1 INTRODUCTION

Let  $G = (V, E)$  be a finite, simple graph of order  $n$ . Then a *cut* in  $G$  is a partition of  $V$ , say  $\langle A, \bar{A} \rangle$  (i.e.,  $\bar{A} = V \setminus A$ ) and its *size*,  $c(A, \bar{A})$  is the number of edges having exactly one vertex in  $A$  and the other in  $\bar{A}$ . A *minimum cut of order  $i$*  is a cut with  $|A| = i$  that has a least number of edges induced by the respective partition. Denoting these values by  $\sigma_i$ , we have,

$$\sigma_i = \min_{A \subseteq V} \{c(A, \bar{A}) : |A| = i\}.$$

That is,  $\sigma_i$  is the size of a smallest cut in  $G$  induced by exactly  $i$  vertices.

For arbitrary graphs, the problem of determining  $\sigma_i$  values is NP-hard (cf. Garey and Johnson (1976)). On the other hand, there are some special classes where these values are easy to determine. For example, if  $G = C_n$ , a cycle on  $n$  vertices, then we know that  $\sigma_i = 2$  for  $1 \leq i \leq |V| - 1$ ; for paths,  $P_n$ , it is trivial to see that  $\sigma_i = 1$  for  $1 \leq i \leq |V| - 1$ . Other cases where the  $\sigma_i$  values are easy to compute appear in Horton (1997). The paper is organized in the following way. In the next section, we show how to compute successive  $\sigma_i$  values, in polynomial time, on arbitrary recursive graph classes, also known as partial  $k$ -trees. This outcome is meaningful since it is not so obvious that such a computation should be polynomial on these graphs, at least not in the same sense by which we have come to expect similar questions to be resolved on recursive structures (cf. Borie, *et al.* (1992)). We then demonstrate the algorithm on a simple example.

We begin section 3 with a brief examination of the *optimal linear arrangement* problem (OLA). OLA is well-studied and known to be NP-hard in general. We next state and briefly explain some old results that relate the  $\sigma_i$  values to linear arrangements. In section 4, we conclude with a discussion of some issues that are directly motivated by the results reported.

## 2 COMPUTING $\sigma_i$ FOR RECURSIVE GRAPH CLASSES

Often, otherwise NP-hard questions on graphs can be resolved when the latter are restricted to special classes. Classic among these are series-parallel

graphs, Halin graphs, and in general, partial  $k$ -trees. It turns out that this is also the case relative to the determination of  $\sigma_i$  although it is less than immediate that this should be so. We begin with some background.

## 2.1 Recursive Graph Classes

Informally, a *recursive graph class* is one in which any sufficiently large member in the class can be formed by successively joining smaller members in the class at specific vertices called *terminals*. Letting the maximum allowable number of terminals be  $k$ , we sometimes refer to these as  $k$ -terminal graphs, or partial  $k$ -trees. More formally, a  $k$ -terminal graph  $G = (V, T, E)$  has a vertex set  $V$ , edge set  $E$ , and a (possibly ordered) set of distinguished vertices or terminals  $T \subseteq V$  specified such that  $T = \{t_1, t_2, \dots, t_{t(G)}\}$ , where  $t(G) = |T| \leq k$ . For some  $k$ , we let  $U$  be the set of all  $k$ -terminal graphs. Then, a *recursively constructed graph family*,  $F = (B, R)$  in  $U$ , has base elements (graphs)  $B \subseteq U$  and a finite set of recursive *composition* operations  $R = \{f_1, f_2, \dots, f_n\}$  where each  $f_i : U^{p_i} \rightarrow U$ . Here,  $p_i$  denotes the *arity* of the operation  $f_i$ . Generally, we consider only base elements in which all vertices are terminals. In this case, it follows that all such structures decompose in the trivial way into edges, so we can simply take  $B$  to be a singleton consisting of  $K_2$ .

Now we define more precisely the notion of a composition operation  $f$ . For  $1 \leq j \leq m$ , let  $G_j = (V_j, T_j, E_j)$  where  $V_1 - T_1, V_2 - T_2, \dots, V_m - T_m$  are mutually disjoint. Then define  $f(G_1, G_2, \dots, G_m) = G = (V, T, E)$  where  $V = V_1 \cup V_2 \cup \dots \cup V_m$ ,  $E = E_1 \cup E_2 \cup \dots \cup E_m$ , and  $T \subseteq T_1 \cup T_2 \cup \dots \cup T_m$ . Each possible subset  $T$  yields a distinct composition operation.

A *decomposition tree* of a  $k$ -terminal graph  $G$  is a rooted tree with vertex labels  $g$  and  $f$  such that

- $g_v = G$  if  $v$  is the root,
- $f_v \in R$  if  $v$  is an interior node,
- $g_v = f_v(g_{v_1}, g_{v_2}, \dots, g_{v_m})$  if interior node  $v$  has children  $v_1, v_2, \dots, v_m$ , and

- $g_v = B$  if  $v$  is a leaf.

Decomposition trees are key in the general problem solving approach on  $k$ -terminal recursive graphs; if we know the solution to a given problem (*i.e.*, vertex cover, dominating set, chromatic number, etc.) on the leaf graphs of a decomposition tree (base graphs), then the postorder traversal of the tree with appropriate recurrence formulae (relevant to a given problem) would produce an efficient (often linear) algorithm for the problem on the given  $k$ -terminal instance.

For example, let  $G = (V, \{t_1, t_2\}, E)$  and let  $G_j = (V_j, \{t_1^{(j)}, t_2^{(j)}\}, E_j)$  for  $j = 1, 2$  be 2-terminal graphs. Define the series operation as  $s(G_1, G_2) = G$  if  $t_1^{(1)} = t_1, t_2^{(1)} = t_1^{(2)}$ , and  $t_2^{(2)} = t_2$ . Define the parallel operation as  $p(G_1, G_2) = G$  if  $t_1^{(1)} = t_1^{(2)} = t_1$  and  $t_2^{(1)} = t_2^{(2)} = t_2$ . Figure 1 demonstrates both of these operations as well as the notion of a decomposition tree.

To develop appropriate recurrence relations, one starts by constructing a *multiplication table*  $f'$  for each composition operation  $f$ . If  $G = f(G_1, G_2, \dots, G_m)$  then the table exhibits the outcome for  $G$  that corresponds to each  $m$ -tuple of compatible subgraph property-tuples for  $G_1, G_2, \dots, G_m$ . It is then straightforward to construct the recurrence relations directly from a complete table. These formulae simply compute the optimal property values from among the possible compositions of compatible pairs. For some simple illustrations of this strategy, the interested reader is directed to any of a host of references among which are Richey and Parker (1985) and Horton (1997). More formal *models* of the methodology appear in Wimer (1986) and Borie, *et al.*(1992).

## 2.2 Calculating $\sigma_i$ on any Recursive Graph Class

For ease of illustration, we will consider only binary composition where  $G = f(G_1, G_2)$ . The extension to the  $m$ -ary case is cumbersome but straightforward. Suppose  $G = (V, T, E)$  is a  $k$ -terminal graph,  $S \subseteq T$  and  $0 \leq i \leq n = |V|$ . Then we shall define  $m(G, S, i)$  to be the smallest number of cut edges that partition  $V$  such that  $i$  vertices are in one component of the bipartition (call this the “blue” side) and  $n - i$  are in the other (“red”), where vertices in  $S$  are entirely blue and those in  $T - S$  entirely red. We further define  $m(G, S, i) = \infty$  if no cutset exists that satisfies the stated conditions.

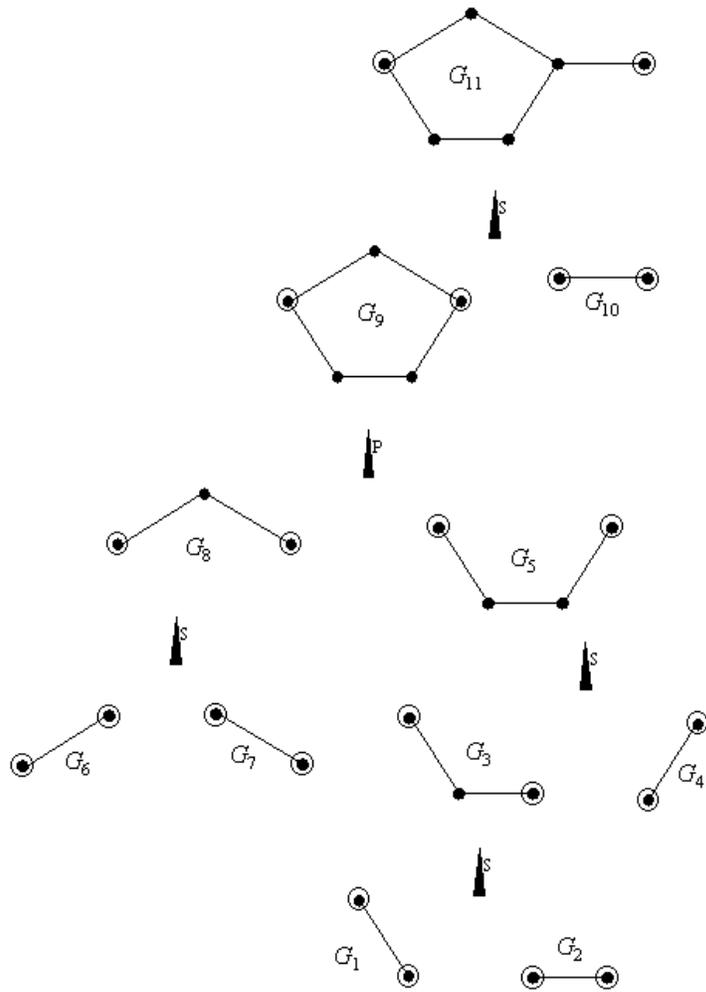


Figure 1: A Decomposition Tree

We can easily compute  $m(G, S, i)$  at the leaves of the decomposition tree, since each corresponding graph is typically either  $K_2$  or, if  $B$  is defined differently, a graph with  $V = T$  and therefore  $|V| = k$ . Now, assume that  $G = f(G_1, G_2)$  where  $G_j = (V_j, T_j, E_j)$  for  $j = 1, 2$ ; then, compute  $m(G, S, i)$  for each non-leaf node by the following procedure:

**Algorithm 2.1**

For each of the  $2^k$  subsets  $S \subseteq T$  do

For each  $i$  such that  $0 \leq i \leq n$  do

Let  $m(G, S, i) = \min\{m(G_1, S_1, i_1) + m(G_2, S_2, i_2)\}$  such that

conditions  $a, b$ , and  $c$  hold.

- condition  $a : S_1 \cap T_2 = S_2 \cap T_1$
- condition  $b : S = (S_1 \cup S_2) \cap T$
- condition  $c : i = i_1 + i_2 - |S_1 \cap S_2|$

Then when  $m(G, \bullet, \bullet)$  is found at the root graph  $G$  of the decomposition tree, we can obtain each  $\sigma_i$  as

$$\sigma_i = \min\{m(G, S, i) : S \subseteq T\}. \tag{1}$$

□

The coloring of  $G$  must be *color compatible* with the colorings of  $G_1$  and  $G_2$ . That is, if some vertex  $v$  of  $G$  appears in both  $G_1$  and  $G_2$ , then it must appear in both  $S_1$  and  $S_2$  (in which case it is blue) or in both  $T_1 - S_1$  and  $T_2 - S_2$  (in which case it is red). Note that condition  $a$  above insures that composition is color compatible, whereas conditions  $b$  and  $c$  describe how  $S$  and  $i$ , respectively, are determined. We can think of each  $m(G_j, \bullet, \bullet)$  as a table with  $2^k$  rows (one for each subset of  $T$ ) and  $n + 1$  columns, where

$n = |V_j|$ . Thus if  $G = f(G_1, G_2)$ ,  $m(G, \bullet, \bullet)$  is completely determined from  $m(G_1, \bullet, \bullet)$  and  $m(G_2, \bullet, \bullet)$ .

It is easy to verify that the running time of Algorithm 2.1 is polynomial in the size of the input graph. Clearly the number of columns of these tables grows linearly with the order of the graph, but the number of rows remains the same. Every composition adds at least one edge, so there are  $O(n)$  nodes in the decomposition tree (for members of any recursive graph class,  $|E| = O(n)$ ). For each node there are  $O(2^k n)$  values of  $m(G, S, i)$  to calculate. For each  $m(G, S, i)$  the recursion involves taking the minimum of  $O(2^k 2^k n) = O(2^{2k} n)$  expressions; to see this, select any  $S_1$  (there are  $2^k$  choices), choose any  $S_2$  (again, there are  $2^k$  choices), then choose  $i_1$  (there are  $O(n)$  choices). Now the value of  $i_2$  is determined, and since each such expression can be computed in  $O(1)$  effort, the total running time is  $O(n^3)$  for fixed  $k$ .

Next, we shall verify correctness. If  $G = f(G_1, G_2)$ , we know that as long as merged terminals of  $G_1$  and  $G_2$  have the same colors (*i.e.*, they are either both blue or both red), then the *status* of any  $e \in E(G)$  is the same as it was in  $G_1$  or  $G_2$ , depending on which child in the decomposition tree it is derived from. Observe that by “status”, we refer to whether or not the edge in question connects a red vertex with a blue one and is hence a cut edge. Condition *a* above insures that this color compatibility holds. Since  $E(G) = E(G_1) \cup E(G_2)$  and  $E(G_1) \cap E(G_2) = \emptyset$ , we can obtain the number of cut edges in  $G$  with a given  $S \subseteq T$  colored blue, by simply adding the cut edges present in  $G_1$  and  $G_2$  under compatible conditions for  $S_1$  and  $S_2$ . But requirements *b* and *c* insure these conditions are maintained.

We have thus established

**Theorem 2.1** *Let  $G = (V, E)$  be a member of any recursive graph family  $F$ . Then in effort bounded by a polynomial function of the size of  $G$ , Algorithm 2.1 will compute every value  $\sigma_i$  for  $i = 1, 2, \dots, |V| - 1$ .*

□

It is worth noting that Algorithm 2.1 is not (so far as we can judge) “anticipated” in the same sense that other fast algorithms are, for problems restricted to recursively constructable instances. That is, it is not evident that the problem of computing  $\sigma_i$  values is expressible in any of the formal contexts that have been developed for graph problems on recursive structures.

Among these is the previously mentioned predicate calculus developed in Borie, *et al.*(1992) (and extended in Borie, *et al.*(1993)) where if a given problem is shown to be expressible in the calculus, then a polynomial-time algorithm for its solution is guaranteed for the problem on *any* recursive graph. Thus, if a legal expression for the given problem can be formed, it is generally straightforward to create a practical algorithm; above all, the algorithm's existence itself is not in question. On the other hand, the more interesting outcome is to find a fast algorithm for a problem whose formal expressibility status is, if not explicitly known to be impossible, at least ambiguous. This is the case with the computation of  $\sigma_i$  thus lending interest to the creation of Algorithm 2.1.

### 2.3 An Example

Now we demonstrate Algorithm 2.1 by describing a procedure for computing the  $\sigma_i$  values for series-parallel graphs. This is a 2 terminal class, and  $|T| = 2$  means there are  $2^2 = 4$  subsets of  $T$  to be accounted for. We shall denote these subsets by  $N, L, R$ , and  $B$  indicating *neither, left, right, or both* terminal vertices are colored blue, respectively. We use the notation  $m(G, S, i)$  as defined in the previous section.

First, to initialize the recursion, we observe that the values of  $m(K_2, \bullet, \bullet)$  are as indicated in Table 1.

Table 1:  $m(K_2, \bullet, \bullet)$

	<b>0</b>	<b>1</b>	<b>2</b>
<b><i>N</i></b>	0	$\infty$	$\infty$
<b><i>L</i></b>	$\infty$	1	$\infty$
<b><i>R</i></b>	$\infty$	1	$\infty$
<b><i>B</i></b>	$\infty$	$\infty$	0

Using these values at the leaf nodes ( $G_1, G_2, G_4, G_6, G_7$ , and  $G_{10}$ ) of the decomposition tree of Figure 1, it is easy to apply Algorithm 2.1 to compute the  $m(G, S, i)$  values for the other graphs in the tree. Table 2 gives the values for  $G_3$  and  $G_8$ , Table 3 for  $G_5$ , Table 4 for  $G_9$ , and Table 5 for  $G_{11}$ .

Table 2:  $m(P_3, \bullet, \bullet)$ 

	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>
<b>N</b>	0	2	$\infty$	$\infty$
<b>L</b>	$\infty$	1	1	$\infty$
<b>R</b>	$\infty$	1	1	$\infty$
<b>B</b>	$\infty$	$\infty$	2	0

Table 3:  $m(G_5, \bullet, \bullet)$ 

	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>N</b>	0	2	2	$\infty$	$\infty$
<b>L</b>	$\infty$	1	1	1	$\infty$
<b>R</b>	$\infty$	1	1	1	$\infty$
<b>B</b>	$\infty$	$\infty$	2	2	0

From  $m(G_{11}, \bullet, \bullet)$  and equation (1), it is easy to obtain the values of  $\sigma_i$  for  $G_{11}$  which are  $\sigma = [\sigma_0, \sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6] = [0, 1, 2, 2, 2, 1, 0]$ .

In the next section, we briefly investigate the celebrated *optimal linear arrangement* problem and its relationship to a graph's  $\sigma_i$  values in order to motivate the discussion that follows. The results of section 3 are certainly not new but merit restating in light of Algorithm 2.1.

Table 4:  $m(G_9, \bullet, \bullet)$ 

	<b>0</b>	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>	<b>5</b>
<b>N</b>	0	2	2	4	$\infty$	$\infty$
<b>L</b>	$\infty$	2	2	2	2	$\infty$
<b>R</b>	$\infty$	2	2	2	2	$\infty$
<b>B</b>	$\infty$	$\infty$	4	2	2	0

Table 5:  $m(G_{11}, \bullet, \bullet)$ 

	0	1	2	3	4	5	6
<b>N</b>	0	2	2	3	3	$\infty$	$\infty$
<b>L</b>	$\infty$	2	2	2	2	1	$\infty$
<b>R</b>	$\infty$	1	2	2	2	2	$\infty$
<b>B</b>	$\infty$	$\infty$	3	3	2	2	0

### 3 CUT VALUES AND LINEAR ARRANGEMENT

Given a simple, finite graph  $G = (V, E)$  of order  $n$ , the *optimal linear arrangement* problem (OLA) seeks a vertex labeling  $f : V \rightarrow \{1, 2, \dots, n\}$  such that  $\sum_{(u,v) \in E} |f(u) - f(v)|$  is minimum over all such labelings. For ease, let us denote the value of an admissible labeling of a graph  $G$ , by  $L(G)$ . Optimal labelings are denoted by  $f^*$  and their values by  $L^*$ . OLA is well-known to be NP-hard in general but solvable on trees following work reported in Shiloach (1979) and more recently, in Chung (1984). The problem is also solved on the class of *outerplanar* graphs; planar graphs without subgraphs homeomorphic to  $K_4$  or  $K_{2,3}$  (Frederickson, *et al.* (1988)). It is interesting to note that the problem's status remains open on the larger class of series-parallel graphs. In fact, the status of OLA is unknown for recursive graphs in general.

In this section, we present some old results that show how the  $\sigma_i$  values are related to  $L(G)$ . We do this to facilitate and motivate the discussion that follows. To begin, let  $S_i$  be a subset of vertices of order  $i$  and as defined before,  $\sigma_i$  denotes the size of a smallest cardinality cut formed over all choices for  $S_i$ , *i.e.*, by the respective  $\langle S_i, V \setminus S_i \rangle$ . Then, we will call a graph  $\sigma$ -good if there exist sets  $S_1, S_2, \dots, S_n$  where each produces the respective  $\sigma_i$  and  $S_1 \subset S_2 \subset \dots \subset S_n$ . The subsets formed in this way are said to be *nested*. The graph shown on the left in Figure 2 is  $\sigma$ -good; admissible sets  $S_i$  are indicated by the given labeling, *i.e.*,  $S_1 = \{1\}$ ,  $S_2 = \{1, 2\}$ ,  $S_3 = \{1, 2, 3\}$ , etc. Alternately, the graph to the right is not  $\sigma$ -good since clearly  $\sigma_2 = \sigma_3 = 1$  but the only subsets realizing these values are  $S_2 = \{a, b\}$  and  $S_3 = \{d, e, f\}$  but  $S_2 \not\subset S_3$ .

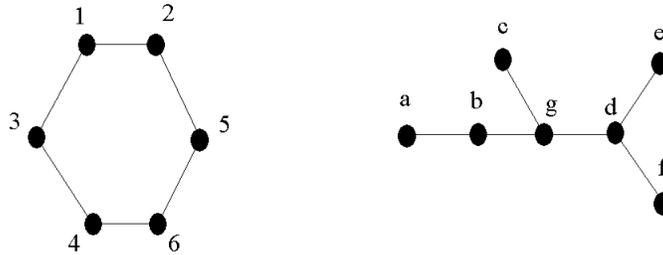


Figure 2: Graphs Demonstrating the Concept of  $\sigma$ -Goodness

Observe that for a cycle,  $C_n$ , we have  $\sigma_i = 2$  for  $1 \leq i \leq |V| - 1$  and thus

$$\sum_{1 \leq i \leq n-1} \sigma_i = 2(n-1)$$

which is precisely  $L^*(C_n)$ . On the other hand, for the tree in Figure 2, the reader can easily verify that  $\sigma_1 = \sigma_2 = \dots = \sigma_6 = 1$ . It is also easy to see, in this case, that

$$L^* = 8 > \sum_{1 \leq i \leq n-1} \sigma_i = 6.$$

Following, we show that these outcomes are not unanticipated.

**Theorem 3.1** *Let  $G = (V, E)$  be an arbitrary graph having an optimal labeling  $f^*(G)$  producing value  $L^*(G)$ . Then,  $L^*(G) \geq \sum_i \sigma_i$ .*

*Proof:* The proof is easy; simply map  $f^*(G)$  to the integer line and form the implied sets  $S_i$  in the natural way, *i.e.*,  $v \in S_i$  whenever  $i \geq f(v)$ . But obviously, the cardinalities of the cutsets induced by these  $S_i$  are bounded

from below by the respective optimal cut values  $\sigma_i$ , produced over all sets of the stated cardinalities. This is sufficient to establish the relationship and the proof is complete.

□

From Theorem 3.1, we have that  $\sum_i \sigma_i$  provides a lower bound on the cost of *any* arrangement value and hence on an optimal value, for any graph. The next result underscores a stronger relationship for graphs that are  $\sigma$ -good.

**Theorem 3.2** *Let  $G$  be an arbitrary finite graph. Then  $G$  is  $\sigma$ -good if and only if there exists an optimal labeling,  $f^*(G)$  such that  $L^*(G) = \sum_i \sigma_i$ .*

*Proof:* ( $\Leftarrow$ ) As indicated in the proof of Theorem 3.1, any labeling of  $G$  defines nested sets  $S_1 \subset S_2 \dots \subset S_n$  where  $v \in S_i$  whenever  $i \geq f(v)$ . Summing the sizes of the cut sets implied by these  $S_i$ , yields the arrangement value. Then for  $f^*(G)$  and the induced sets  $S_i$ , suppose we have

$$L^*(G) = \sum_{1 \leq i \leq n-1} c(S_i, \overline{S_i}) = \sum_{1 \leq i \leq n-1} \sigma_i.$$

But then it must be the case that  $c(S_i, \overline{S_i}) = \sigma_i, \forall i$ , and thus, it follows that  $G$  is  $\sigma$ -good.

( $\Rightarrow$ ) Alternately, suppose that  $G$  is  $\sigma$ -good. Then there are nested subsets  $S_1, S_2, \dots, S_n \subseteq V$  and that moreover, define an ordering of  $G$  given as  $f(v) = i$  where  $v \in S_i \setminus S_{i-1}$  for  $1 \leq i \leq n$  with  $S_0 = \emptyset$ . But the value of this labeling is

$$\sum_{1 \leq i \leq n-1} c(S_i, \overline{S_i}) = \sum_{1 \leq i \leq n-1} \sigma_i$$

which must be an optimal value following the inequality of Theorem 3.1. This is enough to establish the result and we are done.

□

So, given any graph, if we can constructively verify  $\sigma$ -goodness we will have also solved OLA on the instance, since the nested subsets  $S_1, S_2, \dots, S_n$

define an optimal labeling. Also, there are graph classes for which OLA can be efficiently solved on all instances accordingly, but where the latter may not be  $\sigma$ -good in general. For example, all trees are not  $\sigma$ -good yet we know that OLA is solved on trees. This is also the case for outerplanar graphs as well as for certain, restricted *Halin* graphs (*cf.* Easton, *et al.* (1996)).

Unfortunately, verifying  $\sigma$ -goodness is not easy since the problem of determining  $\sigma_i$  values themselves is NP-hard in general. So, while  $\sum_i \sigma_i$  might be useful in bounding  $L^*(G)$  as per Theorem 3.1, it is not likely that we can evaluate the explicit bound value efficiently for arbitrary graphs.

As mentioned above, the ideas captured by Theorems 3.1 and 3.2 have been reported in the research of others. Harper (1966) appears to have been the first to describe the relevant relationships; others include Adolphson and Hu (1973), Liu and Vannelli (1995) and Bezrukov (1996).

## 4 DISCUSSION

### 4.1 Using $\sum_i \sigma_i$

By the results described in Section 2, we are able to compute  $\sum_i \sigma_i$  for any recursive graph  $G$ . However, it is not at all clear that this outcome helps us find a labeling that solves OLA for these graphs. Of course, if we can find a labeling with value equal to  $\sum_i \sigma_i$ , then by Theorem 3.1 we are done since no arrangement can have a value smaller than this. On the other hand, if we can find sufficient evidence to conclude that  $G$  is *not*  $\sigma$ -good, then we know from Theorem 3.2, that the  $\sum_i \sigma_i$  bound is not attainable and hence, an optimal labeling must have value at least 1 greater than this.

While we should not hope to be able to decide if an arbitrary graph is  $\sigma$ -good, it is open as to whether or not this issue is resolvable for restricted classes. To illustrate a case where this can be done, consider a particularly restricted (albeit infinite) class of Halin graphs. The latter are planar graphs having the property that the respective edge sets can be partitioned into a tree,  $T$  no vertex of which has degree 2 and a cycle,  $C$  which spans the pendant vertices of  $T$ . Let vertices and edges of  $T$  and  $C$  be given by  $V(T)$ ,  $E(T)$ ,  $V(C)$ , and  $E(C)$  respectively. Now, consider a subclass of Halin graphs that are regular with degree 3 and where  $T$  is a *caterpillar*, *i.e.*, a tree (of at least 4 vertices) where the elimination of its pendant vertices leaves a path.

An example is shown in Figure 3;  $T$  is denoted in bold.

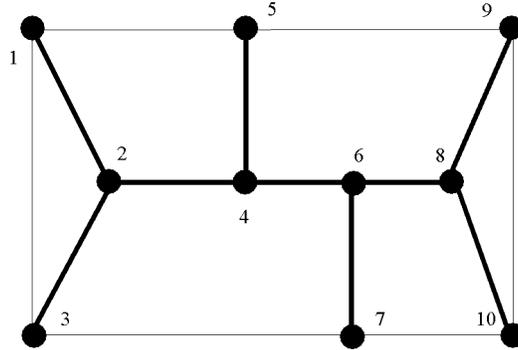


Figure 3: A 3-Regular Caterpillar Halin Graph

First we establish a lemma about these graphs.

**Lemma 4.1** *For every member of the class of 3-regular Halin graphs where  $T$  is a caterpillar,*

$$\sigma_i = \begin{cases} 3 & \text{for } i \text{ odd} \\ 4 & \text{for } i \text{ even} \end{cases}$$

for  $1 \leq i \leq |V| - 1$ .

*Proof:* First we establish an upper bound on  $\sigma_i$ . For  $i$  odd, we can establish that  $\sigma_i \leq 3$  by finding  $A \subseteq V$  such that  $c(A, \overline{A}) = 3$  and  $|A| = i$ . To do this, select  $e \in E(T)$  such that  $T \setminus e$  contains two components, one having exactly  $i$  vertices. We call these vertices  $A$ . Since  $T$  is a caterpillar, this is possible for every odd  $i$ . Then  $\langle A, \overline{A} \rangle$  is as desired since  $c(A, \overline{A})$  counts precisely one edge from  $E(T)$  and two from  $E(C)$ . For  $i$  even, we can establish that  $\sigma_i \leq 4$  inductively from the (odd) case of  $i - 1$  simply by adding to  $A$  any vertex adjacent to any member of  $A$ .

To establish the lower bound, consider an arbitrary cut  $c(A, \overline{A})$ . Define  $X = A \cap C$ . First suppose  $X \neq \emptyset$  and  $X \neq C$ . Then  $c(A, \overline{A})$  counts at least 2 edges in  $E(C)$ . If  $|A|$  and  $|\overline{A}|$  are odd (observe  $|V|$  is always even for these

graphs), then  $c(A, \overline{A})$  also counts at least 1 edge in  $E(T)$ , so  $c(A, \overline{A}) \geq 3$ . If  $|A|$  and  $|\overline{A}|$  are even, then  $c(A, \overline{A})$  also counts at least 2 edges in  $E(T)$ , and  $c(A, \overline{A}) \geq 4$ .

Next suppose  $X$  is empty. If  $|A| = 1$ , then  $c(A, \overline{A})$  counts exactly 3 edges in  $E(T)$ . If  $|A| > 1$ , then  $c(A, \overline{A})$  counts at least 4 edges in  $E(T)$ . The case of  $X = C$  can be treated analogously by interchanging  $A$  and  $\overline{A}$ .

□

Using this result we can calculate  $\sum_i \sigma_i = 3(n - 1) + \lfloor \frac{n-1}{2} \rfloor$  which, by Theorem 3.1, bounds from below the value of any admissible labeling. Now, consider the following labeling strategy. First, embed the instance graph in the plane and identify the subgraph corresponding to  $T$ . Denote a longest path in  $T$  by  $P$  and let  $x_i \in V(T)$  be the internal vertices on  $P$ . Next, assign the label 1 to one pendant of  $P$  and  $n$  to the other. Then beginning with the vertex labeled 1, assign labels to the vertices  $x_i$  in strictly increasing order using only and all the even integers between 1 and  $n$ . Complete the total labeling of the graph by assigning the remaining (odd) integers to the unlabeled vertices  $u \in V(C)$  such that  $f(u) = f(x_i) + 1$  if  $(x_i, u) \in E$ . The labeling shown in Figure 3 was formed this way.

Now, the value of a labeling conforming to this scheme is easy to write in closed form. There are always three edge-disjoint paths connecting the vertices labeled 1 and  $n$ . This follows as a property of Halin graphs. One of these paths is  $P$  and the other two define  $C$ . Since the labels on each of these paths increase monotonically, the total value of the arrangement on these paths is  $3(n - 1)$ . Further, if these three paths are removed from  $G$ , a set of  $\frac{n}{2} - 1$  independent edges remain and each is labeled by consecutive integers per the stated procedure. Hence, the cost of the total labeling is  $3(n - 1) + \frac{n}{2} - 1$  but since 3-regularity requires that  $n$  is even, this is equal to  $3(n - 1) + \lfloor \frac{n-1}{2} \rfloor$ . This agrees with the value of  $\sum_i \sigma_i$  determined from Lemma 4.1, so the given labeling is optimal. In addition, it is easy to see that the stated labeling always induces nested sets and we have thus established

**Theorem 4.1** *Any member in the class of 3-regular Halin graphs where  $T$  is a caterpillar is  $\sigma$ -good.*

□

Actually, since we can compute  $\sigma_i$  values on any recursive graph, it is

natural to wonder if such results can be extended in some way in order to resolve the general  $\sigma$ -goodness recognition issue accordingly. Unfortunately, the characterization of  $\sigma$ -goodness given in Theorem 3.2 does not seem in a practical sense to extend the class of  $\sigma$ -good graphs, since in every case we have examined where we can show a labeling with  $L^*(G) = \sum_i \sigma_i$ , the graph can easily be shown to be  $\sigma$ -good directly from the definition. Beyond the issue of  $\sigma$ -goodness recognition, extreme good fortune might suggest a way to modify the  $\sigma_i$ -finding computation in order to resolve OLA on recursive graphs altogether. Unfortunately, however, our pursuits in this regard have not yielded the desired outcome. Still, some insights have been produced even though the overall issue remains unresolved. The interested reader is directed to Horton (1997).

Finally, it may also be that  $\sum_i \sigma_i$  can provide some meaningful insight (*e.g.*, bounds) relative to labeling problems other than OLA. For example, suppose we seek a labeling that minimizes  $\max_{(u,v) \in E} |f(u) - f(v)|$ . This is the well-known *bandwidth* problem and in contrast to its OLA counterpart, remains NP-hard even on trees. Although we make no claim regarding the strength of the relationship, it is clear that  $\sum_i \sigma_i$  also yields a lower bound on a graph's bandwidth value. Letting this be  $\alpha(G)$ , we have

**Theorem 4.2** *Let  $G = (V, E)$  be an arbitrary graph. Then*

$$\alpha(G) \geq \left\lceil \frac{\sum_i \sigma_i}{|E|} \right\rceil.$$

*Proof:* We have from Theorem 3.1 that  $L^*(G) \geq \sum_i \sigma_i$ . But certainly,  $|E|\alpha(G) \geq L^*(G)$  and the result follows.

□

To illustrate, consider again the graph in Figure 3. The  $\sum_i \sigma_i$  (and  $L^*(G)$ ) value is 31 and  $|E| = 15$  so after rounding,  $\alpha(G)$  must be at least 3. The labeling shown in Figure 4 achieves this value and is thus an optimal bandwidth labeling.

## 4.2 Further Research

There are various issues that deserve additional study. Among these is a closer examination of the value of the  $\sum_i \sigma_i$  bound as it relates to  $L^*(G)$ . It

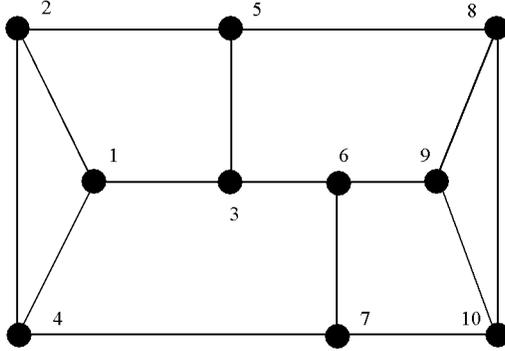


Figure 4: An Optimal Bandwidth Labeling

turns out that the bound is (in the limit) quite close for some graph classes. For example, in Mitchison and Durbin (1986), an optimal labeling scheme for the  $n \times n$  discrete torus is demonstrated. In Horton(1997), it is shown that this labeling has value

$$\frac{n(431n^2 + 350n - 900)}{250}$$

for  $n \equiv 0 \pmod{10}$ . But Horton (1997) also shows that the value of  $\sum_i \sigma_i$  for  $n \equiv 0 \pmod{2}$  is

$$\left\lfloor \frac{n(10n^2 + 9n - 16)}{6} \right\rfloor$$

which yields a ratio, with  $L^*$  for this class of graphs, of 1.0344 for  $n \equiv 0 \pmod{10}$ .

On the other hand, we can also create graphs, even fairly primitive ones such as series-parallel graphs, where this ratio is substantially distant from 1. And of course, the gap between the two values notwithstanding, it remains that we need to be able to compute the relevant  $\sigma_i$  values in the first place.

An intriguing issue that remains is the recognition question, particularly for graph classes where the  $\sigma_i$  values can be efficiently computed: given a

graph  $G$  (belonging to such a class), is  $G$   $\sigma$ -good? To date, we know of no fast algorithm to answer this question, but no complexity result is evident either. A deeper discussion of this issue appears in Horton (1997).

## 5 REFERENCES

1. D. Adolphson and T.C. Hu, Optimal linear ordering, SIAM J. Appl. Math 25 (1973) 403-423.
2. S. Arnborg, J. Lagergren and D. Seese, Easy problems for tree-decomposable graphs, Journal of Algorithms 12 (1991) 308-340.
3. S.L. Bezrukov, Edge isoperimetric problems on graphs, Graph Theory and Combinatorial Biology, Bolyai Soc. Math. Stud. 7, L. Lovasz, A. Gyarfás, G.O.H. Katona, A. Recski, L. Szekely eds., Budapest, (1999) 157-197.
4. R.B. Borie, R.G. Parker and C.A. Tovey, Deterministic decomposition of recursive graph classes, SIAM Journal of Discrete Mathematics 4 (1991) 481-501.
5. R.B. Borie, R.G. Parker and C.A. Tovey, Automatic generation of linear-time algorithms from predicate calculus descriptions of problems on recursively constructed graph families, Algorithmica 7 (1992) 555-581.
6. R.B. Borie, Generation of polynomial time algorithms for some optimization problems On tree-decomposable graphs, Algorithmica 14 (1995) 123-137.
7. F.R.K. Chung, On optimal linear arrangements of trees, Comp. & Maths. with Appls. 10 (1984) 43-60.
8. B. Courcelle and M. Mosbah, Monadic second-order evaluations on tree-decomposable graphs, Graph-Theoretic Concepts in Computer Science, 17th International Workshop, Springer-Verlag (1991).

9. T. Easton, S.B. Horton and R.G. Parker, A solvable case of the linear arrangement problem on Halin graphs, *Congressus Numerantium* 119 (1996) 3-17.
10. P. Fishburn, P. Tetali and P. Winkler, Optimal linear arrangements of a rectangular grid, to appear in *Discrete Mathematics*.
11. G.N. Frederickson and S.E. Hambrusch, Planar linear arrangements of outerplanar graphs, *IEEE Transactions on Circuits and Systems* 35 (1988) 323-332.
12. M.R. Garey and D.S. Johnson, *Computers and Intractability* (W.H. Freeman and Company, New York, 1979).
13. L.H. Harper, Optimal numberings and isoperimetric problems on graphs, *Journal of Combinatorial Theory* 1 (1966) 385-393.
14. S.B. Horton, The optimal linear arrangement problem: algorithms and approximation, Ph.D. Thesis, Georgia Institute of Technology (1997).
15. M. Juvan and B. Mohar, Optimal linear labelings and eigenvalues of graphs, *Discrete Applied Mathematics* 36 (1992) 153-168.
16. W. Liu and A. Vannelli, Generating lower bounds for the linear arrangement problem, *Discrete Applied Mathematics* 59 (1995) 137-151.
17. G. Mitchison and R. Durbin, Optimal numberings of an  $N \times N$  array, *SIAM Journal of Algebraic Discrete Methods* 7 (1986) 571-582.
18. D.O. Muradyan and T.E. Piliposjan, Minimal numberings of vertices of a rectangular lattice (in Russian), *Akad. Nauk. Armjan. SSR. Dokl.* 70 (1980).
19. M.B. Richey and R.G. Parker, On finding spanning eulerian subgraphs, *Naval Research Logistics Quarterly* 32 (1985) 443-455.
20. T.V. Wimer and S.T. Hedetniemi, K-terminal recursive families of graphs, *Proceedings of the 250th Anniversary Conference on Graph Theory*, Fort Wayne, IN (1986).