Journal of Engineering and Applied Sciences 12 (Special Issue 1): 5684-5689, 2017
ISSN: 1816-949X

Branch and Bound for the Cutwidth Minimization Problem

[1]Mochamad Suyudi, [2]Mustafa Mamat, [1]Sukono and [1]Sudradjat Supian
[1]Department of Mathematics, Fakultas Matematikedan Ilmu Pengetahuan Alam, Padjadjaran University, Jawa Barat, Indonesia
[2]Fakulti Informatik dan Komputeran, Universiti Sultan Zainal Abidin, Terengganu, Malaysia

Abstract: The cutwidth minimization problem consists of finding a linear layout of a graph so that, the maximum linear cut of edges (i.e., the number of edges that cut a line between consecutive vertices) is minimized. This study, starts by reviewing previous exact approaches for special classes of graphs as well as a linear integer formulation for the general problem. We propose a branch and bound algorithm based on different lower bounds on the cutwidth of partial solutions.

Key words: Cutwidth, branch and bound integer programming, approaches, partial solutions, vertices, graph

INTRODUCTION

Let G = (V, E) be a graph with vertex set V (|V| = n) and edge set E (|E| = m). A labeling or linear arrangement f of G assigns the integers {1, 2, ..., n} to the vertices of in such a way that each vertex v∈V has a different label f(v) (i.e., f(v)≠f(u) for all u, v∈V). The cutwidth of a vertex v with respect to a labeling f, $CW_f(v)$ is given by the number of edges (u, w)∈E in the graph satisfying f(u)≤f(v)<f(w). In mathematical terms:

$$Cw_f(v) = |\{(u, w) \in E: f(u) \le f(v) < f(w)\}|$$

Given a labeling f, the cutwidth of G is defined as:

$$CW_f(G) = \max_{v \in V} CW_f(v)$$

The optimum cutwidth of graph G, CW(G) is defined as the minimum $CW_f(G)$ value over all possible labelings f. In other words, the cutwidth minimization problem consists of finding a labeling f that minimizes $CW_f(G)$ over set Πn of all possible labelings:

$$CW(G) = \min_{f \in \Pi n} CW_f(G)$$

This problem is NP-hard as stated in Gavril even for graphs with a maximum degree of three (Makedon *et al*., 1985). Some special cases have been solved optimally for example (Harper, 1966) solved the cutwidth for hypercubes (Chung *et al*., 1982) presented a $O(\log^{d-2} n)$ time algorithm for the cutwidth oftrees with n vertices and with maximum degree d. Yannakakis (1985) improved the aforesaid results by giving a O(n log n) time algorithm to determine the cutwidth of trees with n vertices. In particular for k-level, t-ary trees $T_{t,k}$ it holds that:

$$CW(T_{t,k}) = \left\lceil \frac{1}{2}(k-1)(t-1) \right\rceil, \forall k \le 3$$

Exact methods to obtain the optimal cutwidth of grids have been proposed in Rolim. Specifically for width, height ≥2 the researchers proved that:

$$CW(L_{width,\, height}) = \begin{cases} 2, \text{ if width} = \text{height} = 2 \\ \min\{\text{width}+1, \text{height}+1\}, \text{ otherwise} \end{cases}$$

Finally, Thilikos *et al*. (2001) presented an algorithm to compute the cutwidth of bounded degree graphs with small tree-width in polynomial time. Figure 1a shows an example of an undirected graph with 6 vertices and 10 edges. Figure 1b shows a labeling, f of the graph in Fig. 1a, setting the vertices in a line in the order of the labeling as commonly represented in the cutwidth problem. In this way, since f(A) = 1 Vertex A comes first, followed by Vertex D(f(D) = 2) and so on. We represent f with the ordering (A, D, E, F, B, C) meaning that vertex A is located in the first position (Label 1), Vertex D is located in the second position (Label 2) and so on. In Fig. 1b, the cutwidth of each vertex is represented as a dashed line with its corresponding value at the bottom. For example, the cutwidth of vertex A is $CW_f(A) = 5$ because the edges (A, D) (A, E) (A, F) (A, B) and (A, C) have an endpoint in A labeled with 1 and the other endpoint in a vertex labeled with a value >1. Similarly, we can compute the cutwidth of vertex B, $CW_f(B) = 4$ by counting the appropriate number of edges ((A, C) (D, C) (F, C) and (D, C)). Then, since the cutwidth of graph G, $CW_f(G)$ is the

Corresponding Author: Mochamad Suyudi, Department of Mathematics, Fakultas Matematikedan Ilmu Pengetahuan Alam, Padjadjaran University Indonesia, Jawa Barat, Indonesia

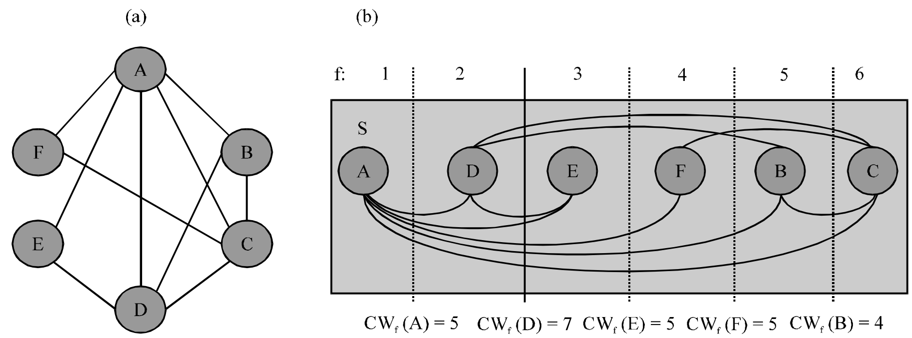

Fig. 1: a) Graph example and b) Cutwidth of for a labeling f

maximum of the cutwidth of all vertices in G in this particular example, we obtain $CW_f(G) = CW_f(D) = 7$, represented in the figure as a bold line with the corresponding value at the bottom.

Luttamaguzi *et al*. (2005) proposed the following linear integer formulation to solve the cutwidth minimization problem. Min b (s.t.):

$$x_i^k \in \{0, 1\} \tag{1}$$

$$i, k \in \{1, ..., n\} \tag{2}$$

$$\sum\nolimits_{k\in\{1, ..., n\}} x_i^k = 1, \forall i \in \{1, ..., n\} \tag{3}$$

$$\sum\nolimits_{i\in\{1, ..., n\}} x_i^k = 1, \forall k \in \{1, ..., n\} \tag{4}$$

$$y_{i,j}^{k,l} \le x_i^k \tag{5}$$

$$y_{i,j}^{k,l} \le x_j^l \tag{6}$$

$$x_i^k + x_j^l \le y_{i,j}^{k,l} + 1 \tag{7}$$

$$\sum_{\substack{k\le c<l\vee \\ l\le c<k}} y_{i,j}^{k,l} \le b, \forall c \in \{1,...,n-1\} \tag{8}$$

where, x_i^k is a decision binary variable whose indices are i, k∈{1, 2, ..., n}. This variable specifies whether i is placed in position k in the ordering. In other words, for all x_i^k (i, k∈{1, 2, ..., n}) they take on value 1 if and only if i occupies the position k in the ordering; otherwise x_i^k takes on value 0. Constraints (Eq. 3 and 4) ensure that each vertex is only assigned to one position and one position is only assigned to one vertex, respectively. Consequently, constraints (Eq. 1-4) together imply that a solution of the problem is an ordering. The decision binary variableis $y_{i,j}^{k,l} \in \{0, 1\}$ is defined in terms of x_i^k and x_j^l as follows:

$$y_{i,j}^{k,l} = x_i^k \wedge x_j^l$$

where, i, j∈{1, 2, ..., n} $(v_i, v_j) \in E$ and k, l∈{1, 2, ..., n} the labels associated to vertex v_i and v_j, respectively. In the linear formulation above this conjunction is computed with constraints (Eq. 5-7).

Constraint Eq. 8 computes for each position C in the ordering the number of edges whose origin is placed in any position k ($1 \le k<c$) and destination in any Position l ($c<l\le n$). The cutwidth problem consists of minimizing the maximum number of cutting edges in any position c∈{1, ..., n-1} of the labeling. Therefore, the objective function b must be larger than or equal to this quantity. In this study, we propose a branch and bound algorithm for the cutwidth minimization problem.

Lower bounds for partial solutions: Given a subset S of V with k<n vertices and an ordering $g \in \Pi_k$ assigning the integers {1, 2, ..., k} to the vertices in S, we define a partial solution as the pair (S, g). A complete solution of the cutwidth problem in the graph G = (V, E) can be obtained by adding n-k elements from V\S to S, assigning them the integers {k+1, k+2, ..., n}. Therefore, the elements in S ordered according to g can be viewed as an incomplete or partial solution of the cutwidth problem in G. We define U as the set of unlabeled vertices (U = V\S) and S_g as the setof all complete solutions of the problem in G obtained by adding ordered elements to S. Figure 2 shows the partial solution (S, g) of the example introduced in Fig. 1a where the vertices in S = {A, D, E} have been labeled with g (g (A) = 1, g (D) = 2 and g (E) = 3). Vertices B, C and F remain unlabeled and therefore belong to set U.

Given a partial solution (S, g) with S⊂G and $g \in \Pi_k$, we consider the graph $G_S = (S, E_S)$ where, S is the set of labeled vertices and $E_S \subset E$ is the set of edges among them. In the example depicted in Fig. 2, S = {A, D, E}, E_S = {(A, E), (D, E)} and Sg = {(A, D, E, F, C, B), (A, D, E, C, B, F), (A, D, E, B, F, C), (A, D, E, F, B, C), (A, D, E, C, F, B), (A, D, E, B, C, F)}.

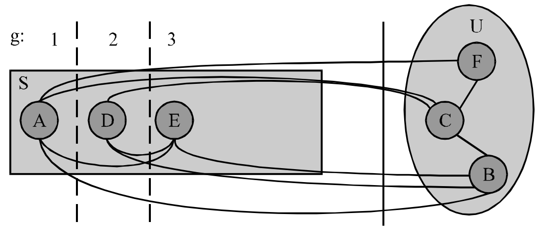

Fig. 2: Partial solution

Particularizing the expression to compute the cutwidth shown in section 1 to a partial solution (S, g), we can calculate the cutwidth of each labeled vertex in G_S with respect to the ordering g and the edges in E_S, $CW_g(v)$ as follows:

$$CW_g(v) = \left|\{(u, w) \in E_S: g(u) \leq g(v) < g(w)\}\right|$$

In the example in Fig. 2, we have CWg(A) = 1, CWg(D) = 2 and CWg(E) = 0. It is clear that the cutwidth values in the partial solution provide a lower bound of their corresponding values than in any complete solution f∈Sg. In this example, if f is a complete solution (with 4-6 assigned to C, B and F), we have $CW_f(A) \geq CWg(A) = 1$, $CW_f(D) \geq CWg(D) = 2$ and $CW_f(E) = CWg(E) = 0$. We can therefore conclude that the cutwidth of the graph CW_f () is larger than max {CWg(A), CWg(D), CWg(E)} = 2 and say that this maximum is a lower bound of the cutwidth. In mathematical terms for any f∈Sg:

$$Cwf(G) \geq LB(S, g) = \max_{v \in S} CWg(v)$$

In this study, we propose 5 lower bounds, LB_1-LB_5 to the value of $CW_f(G)$ for f∈Sg thus improving this trivial lower bound, LB(S, g). LB_1 is based on the degree of the vertices in G, LB_2 computes the edges between the labeled and unlabeled vertices, LB_3 is a refinement of LB_1, LB_4 considers the best vertex to be labeled next in the partial solution and LB_5 is based on the distribution of the edges in minimizing the cutwidth.

MATERIALS AND METHODS

Lower bound LB_1: Let N(v) be the set of adjacent vertices to vertex v and let E(v) be the edges with an endpoint in v. Consider a solution f and the vertex u in position f(v)-1 (i.e., u precedes v in the ordering f). If an edge in N(v) is adjacent to a vertex w with f(w)<f(v) then it contributes to $CW_f(u)$; otherwise, it contributes to $CW_f(v)$ (the edge is computed in the cutwidth of the vertex). Then, $CW_f(u)+CW_f(v) \geq |N(v)|$. Therefore:

$$\max\{CW_f(u), CW_f(v)\} \geq |N(v)|/2$$

Considering that the cutwidth of the graph $CW_f(G)$ is the maximum of the cutwidths of all its vertices, we conclude that |N(v)|/2 is a lower bound on $CW_f(G)$:

$$CW_f(G) \geq LB_1 = \max_{v \in V} \left\lceil \frac{|N(v)|}{2} \right\rceil$$

In the example in Fig. 2, we obtain $LB_1 = 3$. Note that this bound is independent of the labeling f and it actually provides a lower bound on the optimum cutwidth of the graph CW(G).

Lower bound LB_2: Given a partial solution (S, g) and a complete solution f in Sg, the cutwidth of a vertex v∈S with respect to f, $CW_f(v)$ can be computed as:

$$CW_f(v) = CW_g(v) + \sum_{\substack{u \in S \\ 1 \leq g(u) \leq g(v)}} |N_U(u)| \quad (9)$$

where, $N_U(u)$ is the set of unlabeled adjacent vertices to u. The first term in this expression, CWg(v), corresponds to the cutwidth of v in $G_S = (S, E_S)$. The second term computes the number of edges with an endpoint in a vertex u labeled with g(u)≤g(v) (i.e., previous to v in the ordering g) and the other endpoint in aunlabeled vertex w. Note that f(w)>g(v) for all w in U and any labeling (solution) f in Sg. This is why we include all the edges with an endpoint in the unlabeled vertices w in the computation of $CW_f(v)$.

Given that Eq. 9 provides an expression of $CW_f(v)$ for all v in S⊆V and that $CW_f(G)$ is the maximum of $CW_f(v)$ for all v in V, we can conclude that:

$$CW_f(G) \geq LB_2(v) = \max_{v \in S} \{CWg(v) + \sum_{\substack{u \in S \\ 1 \leq g(u) \leq g(v)}} |N_U(u)|\}$$

In the partial solution shown in Fig. 2, the value of the cutwidth of any solution f in Sg, $CW_f(G)$, satisfies:

$$CW_f(G) \geq \max\{CW_f(A), CW_f(D), CW_f(E)\} = \max\{4, 7, 6\} = 7$$

$$CW_f(A) = CW_g(A) + |N_U(A)| = 1+3 = 4$$

$$CW_f(D) = CW_g(D) + |N_U(A)| + |N_U(D)| = 2+3+2 = 7$$

$$CW_f(E) = CW_g(E) + |N_U(A)| + |N_U(D)| + |N_U(E)| = 0+3+2+1 = 6$$

Lower bound LB_3: Given a partial solution (S, g) and an unlabeled vertex u∈U, let $N_S(u)$ be the set of labeled adjacent vertices to u. Let v_k be the vertex in S with the largest label (i.e., $g(v_k) = k = |S|$). It is clear that for any f in Sg and any v in S, $f(v) \le f(v_k) < f(u)$. Then, $Cw_f(v_k) \ge |N_S(u)|$. On the other hand, we can also apply the same argument to the vertices in U as in LB_1, obtaining an improved lower bound LB_3 for the vertices in U:

$$CW_f(G) \ge LB_3 = \max_{u \in U}\{\left\lceil \frac{1}{2}|N(u)| \right\rceil, N_S(u)\}$$

In the example in Fig. 2, we can see that the value of LB_3 for vertices B, C and F:

$$LB_3(B) = \max\{\left\lceil \frac{1}{2}|N(B)| \right\rceil, N_S(B)\} = \max\{2,3\} = 3$$

$$LB_3(C) = \max\{\left\lceil \frac{1}{2}|N(C)| \right\rceil, N_S(C)\} = \max\{2,2\} = 2$$

$$LB_3(F) = \max\{\left\lceil \frac{1}{2}|N(F)| \right\rceil, N_S(F)\} = \max\{1,1\} = 1$$

Therefore, LB_3 will be for this graph:

$$LB_3 = \max\{LB_3(B), LB_3(C)$$
$$LB_3(F)\} = \max\{3,2,1\} = 3$$

Lower bound LB_4: As in the previous case, consider a partial solution (S, g) an unlabeled vertex $u \in U$ the vertex v_k in S with the largest label and a solution f in Sg. If the vertex u is labeled in f with k+1 (i.e., u follows vk in the ordering f) its cutwidth can be computed as:

$$CW_f(u) = CW_g(v_k) - (|N_S(u)| - |N_U(u)|)$$

where, $N_S(u)$ is the set of labeled adjacent vertices to u and $N_U(u)$ is the set of unlabeled adjacent vertices to u. We can then compute a lower bound of the CW_f-value for the vertex in position k+1 by computing the maximum of the term $|N_S(u)|-|N_U(u)|$ for all $u \in U$. Thus we obtain:

$$CW_f(G) \ge LB_4 = CW_g(v_k) - \max_{u \in U}(|N_S(u)| - |N_U(u)|)$$

Figure 3a shows a partial solution (S, g) of the example given in Fig. 1 where, S = {E, F}, g(E) = 1, g(F) = 2 and U = {A, B, C, D} with CWg(F) = 4. Figure 3b shows the value of $|N_S(u)|-|N_U(u)|$ for each vertex u in U. According to the definition given above, we select the Vertex A , giving a value of $LB_4 = 4-(-1) = 5$. This means that independently of the labeling of the vertices in U, the value of the final solution is ≥ 5.

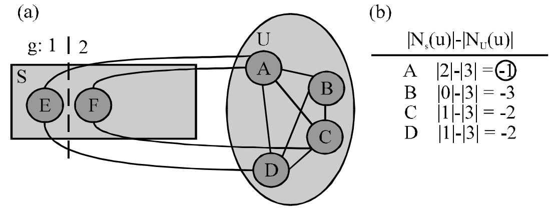

Fig. 3: a) Partial solution and b) $|N_S(u)|-|N_U(u)|$ values for every $u \in U$

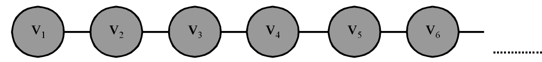

Fig. 4: Graph G' with m = n-1 edges (path)

Lower bound LB_5: Given a graph with n vertices and m edges we compute the lower bound LB_5 of its cutwidth CW(G) by constructing an auxiliary graph G' with n vertices and m edges distributed in such a way that it has minimum cutwidth. In other words, we "put" the edges in G' between the appropriate vertices to obtain a minimum cutwidth. In this way, the cutwidth of G' is a lower bound of the cutwidth of G for any labeling of its vertices (it is in fact a lower bound of the cutwidth of any graph with n vertices and m edges).

Consider the case in which m<n, we construct the auxiliary graph G' as a path (Fig. 4) in which some vertices may eventually be disconnected (when m = n-1 it is a connected path).

The cutwidth of G' = 1 and it is clear that regardless how the edges are distributed in, given that it has m edges for any labeling f its cutwidth $CW_f(G)$ will be equal to or larger than CW(G') = 1. Moreover, if we have m = n, we need to add an extra edge to the connected path G' and it necessarily results in a vertex with cutwidth 2; therefore in this case $CW(G') = 2 \le CW_f(G)$ for any labeling f of the vertices in G.

Let us now consider the case in which m>n. The best way to distribute the m edges in a graph with n vertices in order to reduce its cutwidth is as follows: we place the first n-1 edges joining "consecutive" vertices in the graph (we call them edges of length 1) as shown in Fig. 4 (between v_i and v_{i+1} for any i). Then, we can add a few extra edges increasing the cutwidth by only one unit. Specifically, we can add (n-1)/2 edges between "alternated" vertices (v_i and v_{i+2}) as shown in Fig. 5, keeping the cutwidth of' with value 2. We shall denote them edges of length 2. Therefore, the cutwidth of a graph with n vertices and m edges with $n \le m \le n-1+\lfloor (n-1) \rfloor/2$ satisfies $CW(G') = 2 \le CW_f(G)$ for any labeling f of the vertices in G. Any extra edge would result in a cutwidth of 3.

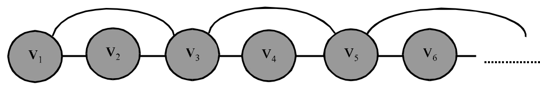

Fig. 5: Graph G' with a length 1 and 2 edges

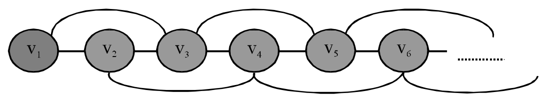

Fig. 6: Graph G' with cutwidth 3

Figure 6 shows how can we add $\lfloor(n-2)\rfloor/2$ edges to the graph in Fig. 5 keeping the cutwidth of G′ with value 3. Then, following the same argument described above, the cutwidth of a graph G with n vertices and m edges with $(n-1)+\lfloor(n-1)\rfloor/2<m\leq(n-1)+(n-2)$ satisfies $3\leq CW_f(G)$ for any labeling f of its vertices (It is easy to see that $\lfloor(n-1)\rfloor/2+\lfloor(n-2)\rfloor/2 = n-2$).

Generalizing this incremental construction of G' we observe that there is a maximum of n-k edges of length k (between v_i and v_{i+k} for any i) that can be added to G' (in which we have previously added all the edges with lengths t from t = 1 to k-1). The first $\lfloor(n-1)\rfloor/k$ edges increase the cutwidth of' by one unit the second $\lfloor(n-2)\rfloor/k$ by another unit the third $\lfloor(n-3)\rfloor/k$ in another unit and so on until the n-k edges of length k have been added and the cutwidth of G' increases by k units. The cutwidth of graph G' provides a bound of the cutwidth of any graph with the same number of vertices and edges.

Initial upper bound: In this study, we propose a heuristic approach to obtain an upper bound for the cutwidth problem based on GRASP methodology (Feo *et al*., 1994). Each GRASP iteration involves constructing a trial solution and then applying a local search from the constructed solution. Algorithm 1 showsa Pseudo-code of our GRASP construction method for the cutwidth problem.

Algorithm 1 (Pseudo-code of the constructive method Procedure constructive:

Let S and U be the sets of labeled and unlabeled vertices of the graph, respectively
Initially S = Ø and U = G
Select a vertex u from U randomly
Assign the label k = 1 to u. S = {u}, U = U\{u}
While (U≠Ø)
k = k+1
Construct CL = {v∈U/(w, v)∈E∀w∈S}
Let $N_S(v)$ and $N_U(v)$ be the set of adjacent labeled and unlabeled vertices to v, respectively
Compute $e(v)= |N_S|-|N_U(v)|\forall v$ in CL
Construct RCL = {v∈CL /e (v)≥th}
Select a vertex u randomly in RCL
Label u with the label k
U = U\{u}, S = SU{u}

RESULTS AND DISCUSSION

The constructive method starts by creating a list of unlabeled vertices U (initially U = V). The first vertex v is randomly selected from all those vertices in U and labeled with 1. In subsequent construction steps a candidate list CL is formed by all the vertices in U that are adjacent to at least one labeled vertex. For each vertex u in CL we compute its evaluation e(u) as:

$$e(u) = N_S(v)|-|N_U(v)|$$

Where:
$N_S(u)$ = The set of labeled adjacent vertices to u
$N_U(u)$ = The set of unlabeled adjacent vertices to u

Note that in this step a greedy selection would label the vertex u* having the maximum e-value with the next available label which would be the minimum $CW_f(u)$ value. However, by contrast, the GRASP methodology computes a restricted candidate list, RCL with good candidates and selects one at random. Specifically, RCL = {v∈CL/e(v)≥th} where the parameter th is a threshold to establish the "good" elements for selection as shown in Algorithm 1.

Once a solution has been constructed we apply an improving phase based on a local search procedure. Our local search method for the cutwidth problem is based on insertion moves. Given a labeling f, we define the insertion move MOVE (f, j, v) consisting of deleting v from its current position f (v) and inserting it in position j. This operation results in the ordering f′ as follows.

If f (v) = i>j then the vertex v is inserted just before the vertex vj in position j. In mathematical term from $f = (\ldots, v_{j-1}, v_j, v_{j+1}, \ldots, v_{i-1}, v, v_{i+1}, \ldots)$, we obtain the new ordering $f' = (\ldots, v_{j-1}, v, v_j, v_{j+1}, \ldots, v_{i-1}, v_{i+1}, \ldots)$.

If f(v) = i<j the vertex v is inserted just after the vertex vj in position j. Therefore, from the ordering $f = (\ldots, v_{i-1}, v, v_{i+1}, \ldots, v_{j-1}, v_j, v_{j+1}, \ldots)$ we obtain $f' = (\ldots, v_{i-1}, v_{i+1}, \ldots, v_{j-1}, v_j, v, v_{j+1}, \ldots)$.

We define the set of critical vertices CV as those with a cutwidth value equal or close to the cutwidth of the graph. These vertices determine the value of the objective function or are considered likely to do so in subsequent iterations. In each iteration, our local search method selects a vertex v in CV and performs the first improving move MOVE (f, j, v) where the meaning of improving is not limited to the objective function (which provides little information in this problem). The position j in the move is computed as the median of the positions (according to f) of the adjacent vertices to v (a search mechanism explores only positions close to j). An improving move is the one that either reduces $CW_f(G)$ or the number of vertices in CV. When a move is performed, the associated vertex is removed from CV. When the set becomes empty, we recalculate it. The method cuts off when there is no improving move associated with the vertices in CV.

CONCLUSION

We have developed an exact procedure based on the branch and bound methodology to provide solutions for the cutwidth minimization problem. We have introduced the partial solution as the set of solutions that share some vertices and we have proposed several approaches to computing lower bounds on partial solutions. These bounds allow us to explore a relatively small portion of the nodes in the search tree when implementing our branch and bound procedure.

REFERENCES

Chung, M.J., F. Makedon, I.H. Sudborough and J. Turner, 1982. Polynomial time algorithms for the min cut problem on degree restricted trees. Proceedings of the 23rd Annual Symposium on Foundations of Computer Science, November 03-05, 1982, ACM, Washington, DC, USA., pp: 262-271.

Feo, T.A., M.G. Resende and S.H. Smith, 1994. A greedy randomized adaptive search procedure for maximum independent set. Oper. Res., 42: 860-878.

Harper, L.H., 1966. Optimal numberings and isoperimetric problems on graphs. J. Comb. Theory, 1: 385-393.

Luttamaguzi, J., M. Pelsmajer, Z. Shen and B. Yang, 2005. Integer programming solutions for several optimization problems in graph theory. Proceedings of the 20th International Conference on Computers and their Applications (CATA 2005), March 16-18, 2005, ISCA, New Orleans, Louisiana, pp: 1-1.

Makedon, F.S., C.H. Papadimitriou and I.H. Sudborough, 1985. Topological bandwidth. SIAM. J. Algebraic Discrete Methods, 6: 418-444.

Thilikos, D.M., M.J. Serna and H.L. Bodlaender, 2001. A polynomial time algorithm for the cutwidth of bounded degree graphs with small treewidth. Proceedings of the 9th Annual European Symposium on Algorithms, August 28-31, 2001, Springer, Berlin, Germany, pp: 380-390.

Yannakakis, M., 1985. A polynomial algorithm for the min-cut linear arrangement of trees. J. ACM., 32: 950-988.