Feedback Vertex Sets in Rotator Graphs

Chiun-Chieh Hsu[1], Hon-Ren Lin[2], Hsi-Cheng Chang[3], and Kung-Kuei Lin[1]

[1] Department of Information Management,
National Taiwan University of Science and Technology,
Taipei, Taiwan, ROC
[2] Department of Information Management, National Taipei College of Business,
Taipei, Taiwan, ROC
[3] Department of Information Management, Hwa-Shia Institute of Technology,
Taipei, Taiwan, ROC

Abstract. This paper provides an algorithm for finding feedback vertex set in rotator graphs. Feedback vertex set is a subset of a graph whose removal causes an acyclic graph and is developed in various topologies of interconnected networks. In 1992, Corbett pioneered rotator graphs, whose interesting topological structures attract many researchers to publish relative papers in recent years. In this paper, we first develops feedback vertex set algorithm for rotator graphs. Our algorithm utilizes the technique of dynamic programming and generates a feedback vertex set of size $n!/3$ for a rotator graph of scale n, which contains $n!$ nodes. The generated set size is proved to be minimum. Finding a minimum feedback vertex set is a NP-hard problem for general graphs. The time complexity of our algorithm, which finds a minimum feedback vertex set for a rotator graph of scale n, is proved to be $O(n^{n-3})$.

1 Introduction

Rotator graph, which is first proposed in 1992[1], is a family of Cayley graph and has rich topological properties, such as symmetric structure, recursive construction, low diameter, unique shortest path routing, and so on. A rotator graph of scale n, or denoted as an n-rotator, contains $n!$ nodes in which every node has a unique permutation of $123\ldots n$. The generation function g_i inserts first symbol of a permutation to the ith position, where $1 < i \leq n$. A 3-rotator is shown in Fig.1. The bold lines are bi-directional edges. A feedback vertex set, or called FVS, is a vertex subset of a graph whose deletion induces the remaining graph acyclic. The FVS are applied in many applications, such as mutual exclusion [2], data security[3], scheduling[4], optical networks[5][6], and so on. A minimum FVS, or denoted as MFVS, is a FVS that contains smallest number of vertices. Published papers[7][8] proved that finding MFVS is a NP-hard problem in general and bipartite graphs. In recent years, FVS algorithms are also developed in mesh and butterfly[9][10], hypercube[11], star graph[12], and shuffle-based interconnection networks[13]. We first proposed a FVS algorithm in rotator graphs in this paper. The dynamic programming techniques are applied in our algorithm. The main idea is that we utilize the FVS of a smaller scale graph to build that

M. Gavrilova et al. (Eds.): ICCSA 2006, LNCS 3984, pp. 158–164, 2006.

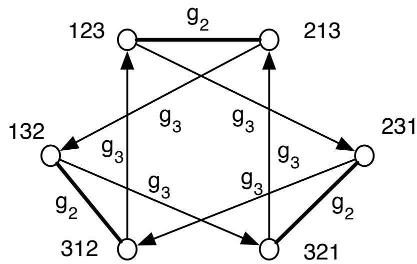

Fig. 1. A 3-rotator

of a larger one. The size of the feedback vertex set generated by our algorithm is proved to be minimum.

2 Definitions

Used lemmas and definitions are introduced in this section. A rotator graph of scale n is denoted as an n-rotator in which every node has a unique permutation of $123\ldots n$. The outgoing edges of one node can be represented as generation functions, g_2, g_3, ..., and g_n. Function g_i inserts first symbol of a permutation to the ith position to form a resultant node. Terms of node, vertex, and permutation are interchangeable in this paper, so are edge and link.

Definition 2.1. Let u be a node in a rotator graph. Node u*g_i denotes the resultant node of applying g_i to node u.

For example, 12345*g_3=23145.

Definition 2.2. Let $\mathrm{V}_{i,j}$ of a rotator graph be the set of all permutations whose ith position is j.

For instance, $\mathrm{V}_{2,1}$ = {2134, 2143, 3124, 3142, 4123, 4132} for a 4-rotator.

Definition 2.3. Let FVS(n) denote a feedback vertex set of an n-rotator.

Although applying g_2 twice to one node makes a routing from itself to its neighbor and back, it does not be considered as a cycle in our discussion. As illustrated in Fig.2, a 3-rotator contains two node disjoint cycles, $123 \rightarrow 231 \rightarrow 312 \rightarrow 123$ and $213 \rightarrow 132 \rightarrow 321 \rightarrow 213$. If we remove node 123 and 132, the remaining graph will contain no cycle. Thus 123, 132 is a FVS of a 3-rotator. In addition, $g_3g_3g_3$ is the smallest cycle in a rotator graph and an n-rotator contains $n!/3$ disjoint cycles of length 3.

3 Algorithm

The FVS algorithm illustrated in this section applies the techniques of dynamic programming. That is, the FVS of an n-rotator, denoted as FVS(n), is

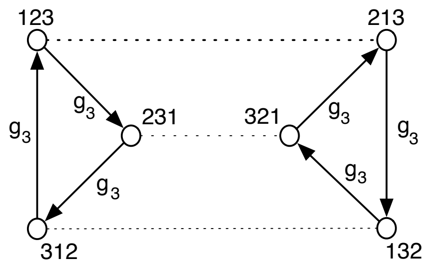

Fig. 2. Two node disjoint cycles in a 3-rotator

obtained by using that of a (n-1)-rotator. FVS(3)={123, 132} is easily observed. The method of finding FVS(4) is described below: Nodes in a 4-rotator can be categorized into $V_{1,1}$, $V_{2,1}$, $V_{3,1}$, and $V_{4,1}$ and each category consists of 6 nodes. FVS(4) is initialized an empty set. We first add $V_{1,1}$ to FVS(4). The remaining graph of removing $V_{1,1}$ from a 4-rotator is denoted as G(4-rotator - $V_{1,1}$). Indeed, G(4-rotator - $V_{1,1}$) does not contain any cycle that includes any node in $V_{2,1}$ or $V_{3,1}$. The reason is that any cycle contains at least one g_k, where $k \geq 3$, and the generation g_k will left shift symbol 1 of the permutations in $V_{2,1}$ or $V_{3,1}$. If cycle exists, symbol 1 must be shifted to the first position eventually and then rotated to a correct position. Because $V_{1,1}$ had already been removed, symbol 1 can not be left shifted to the first position. Thus the cycle does not exist. Therefore, G(4-rotator - $V_{1,1}$) does not include any cycle that contains node in $V_{2,1}$ or $V_{3,1}$. Hence, nodes in $V_{1,1}$ are the only candidates to add to FVS(4). Nodes in $V_{1,1}$ are the form of ***1, where *** represents any legal permutation. Hence, finding cycles in G(4-rotator - $V_{1,1}$) can be viewed as finding cycles in a sub 3-rotator of $V_{4,1}$. Because rotator graphs are node symmetric and the feedback vertex set of a 3-rotator, 123, 132, is already known, the FVS of the sub 3-rotator $V_{4,1}$ can be easily solved by a symbol transformation. For example, FVS(3)={123, 132} implies that 1234, 1324 is a FVS of $V_{4,1}$. By exchanging symbol 1 and 4, we obtain that 4231, 4321 is a FVS of $V_{4,1}$. Therefore, a FVS of a 4-rotator is the union of $V_{1,1}$ and 4231, 4321 = {1234, 1243, 1324, 1342, 1423, 1432, 4231, 4321}.

Lemma 3-1. For an n-rotator, G(n-rotator - $V_{1,1}$) does not contain any cycle that includes any node in $V_{2,1}$ or $V_{3,1}$.

Proof. First, every node in a cycle has a predecessor and a successor node. Because $V_{1,1}$ has been removed, nodes in $V_{2,1}$ do not have any successor node. Thus G(n-rotator - $V_{1,1}$) contains no cycle that includes any node in $V_{2,1}$. Second, every cycle contains at least one g_3 function, which left shift symbol 1 of nodes in $V_{3,1}$ and connects to nodes in $V_{2,1}$. Because $V_{2,1}$ do not have any successor node in G(n-rotator - $V_{1,1}$), there is no cycle that includes any node in $V_{3,1}$. This lemma is verified. □

Lemma 3-2. Let C be a cycle in G(n-rotator - $V_{1,1}$). Every node in C belongs to the same vertex set $V_{i,1}$, where $4 \leq i \leq n$.

Proof. Suppose cycle C contains nodes S_1 and S_2, where $S_1 \in V_{i,1}$, $S_2 \in V_{j,1}$, and $j \neq i$. First consider the case that $i > j$. The path from S_2 to S_1 must pass at least one node $\in V_{1,1}$. Because G(n-rotator - $V_{1,1}$) does not contain any node $\in V_{1,1}$, the path from S_2 to S_1 does not exist. Second, if $i < j$, the path from S_1 to S_2 does not exist with the same reason. This lemma is proved. □

By lemma 3-2, in the remaining graph of removing vertex set $V_{1,1}$ from a rotator graph, every node in a cycle belong to the same sub rotator graph $V_{i,1}$, where $i > 3$. That is, in G(n-rotator - $V_{1,1}$), nodes in a cycle have a property that symbol 1 in the same position. In order to indicate node properties, we divide a permutation into two parts, head sequence and tail sequence. Suppose that symbol 1 of permutation S is in the ith position. Head sequence of S is the first $i-1$ symbols and tail is the ith to the last symbols. For example, for node 43125, head sequence is 43 and tail sequence is 125. In addition, we define length of head/tail sequence is the number of symbols in head/tail sequence. If symbol 1 is in the first position for a permutation, of course, this node has no head sequence. A property can be observed: An n-rotator contains one sub $(n-1)$-rotator graph in which every node has tail length is 1. In addition, n-rotator also contains $n-1$ node-disjoint sub $(n-2)$-rotator in which every node has tail length 2. For example, in a 5-rotator graph there is one sub 4-rotator with tail length 1. Nodes in the sub 4-rotator are of the form ****1, where * represents any legal symbol. In addition, this 5-rotator also has four sub 3-rotator graphs with tail length 2. These four sub 3-rotator are ***12, ***13, ***14, and ***15. A general expression of the property is shown in lemma 3-3.

Lemma 3-3. An n-rotator contains P^{n-1}_{n-k-1} disjoint sub k-rotator graph in which every node has tail length $n-k$, where $k \geq 3$.

Proof. Every node in an n-rotator graph has permutation length n. Nodes in a sub k-rotator have the same tail sequence whose length is $n-k$. Because the first symbol of the tail sequence is 1, the number of distinct k-rotator is therefore $(n-1)(n-2)\ldots(k+1) = P^{n-1}_{n-k-1}$. In addition, because nodes in different sub rotator graphs have different tail sequences, these sub rotators are node disjoint. The lemma is verified. □

The steps of finding a FVS in an n-rotator is described in the following: First, we add the set $V_{1,1}$ of an n-rotator to FVS(n). Second, by lemma 3-3, the remaining graph can be divided into a number of isolated sub rotator graphs. The feedback vertex set of the whole graph can be obtained by joining the feedback vertex set of these distinct sub graphs. Since a rotator graph is node symmetric, the feedback vertex set of a rotator graph can be easily transferred to that of identical size of sub rotator graphs. For example, a feedback vertex set of a 3-rotator is 123, 132. Obviously, 1234, 1324 is the feedback vertex set of sub 3-rotator graph with the form ***4. The feedback vertex of sub rotator graph ***1, ***2, and ***3 is obtained by exchanging symbol (1,4), (2,4), and (3,4) of node 1234 and 1324, respectively. Thus the feedback vertex set of ***1, ***2, and ***3 are 4231, 4321, 1432, 1342, and 1243, 1423, respectively.

We illustrate the steps of acquiring the feedback vertex set of a 5-rotator graph. In the beginning, FVS(3)={123, 132} and FVS(4)={2134, 3124, 2143, 4132, 4132, 3142, 2341, 2431} is already known. This assumption is reasonable because FVS(4) is only a joint of 4 isolated sub 3-rotator graph and FVS(3) is quite simple to be observed. We initialize FVS(5) = $V_{1,1}$ ={12345, 12354, 12435, 12453, 12534, 12543, 13245, 13254, 13425, 13452, 13524, 13542, 14235, 14253, 14325, 14352, 14523, 14532, 15234, 15243, 15324, 15342, 15423, 15432}. By lemma 3-3, a 5-rotator contains one 4-rotator graph with tail length 1 and the sub 4-rotator is of the form ****1. We use FVS(4) to obtain the FVS of the sub 4-rotator ****1. The FVS of the sub 4-rotator {25341, 35241, 25431, 45321, 45321, 35421, 23451, 24351} is added to FVS(5). In addition, a 5-rotator also contains four isolated sub 3-rotator graph with tail length 2, which are ***12, ***13, ***14, and ***15. Each of these 3-rotator contains a FVS of size 2. These FVS of these sub 3-rotator are easily obtains from FVS(3), {123, 132}. The FVS of ***12, ***13, ***14, and ***15 are {34512, 35412}, {24513, 25413}, {23514, 25314} and {23415, 24315}, respectively. These four sets are also add to FVS(5). In summary, FVS(5) contains 24+8+2*4=40 elements.

The algorithm of finding feedback vertex set of a rotator graph is given below:

Algorithm 3-1. Feedback vertex set finding in rotator graphs.

Input: the scale of rotator graph, n.

Output: the feedback vertex set of an n-rotator, FVS(n).

Steps:

1. Initialize FVS(n) = $V_{1,1}$.
2. for $k = 3$ to $n-1$ do

   Add FVS of P^{n-1}_{n-k-1} units of sub k-rotator graph to FVS(n).

   loop

Lemma 3-4. The output FVS(n) of Algorithm 3-1 is correct.

Proof. The first step of Algorithm 3-1 adds $V_{1,1}$ to FVS(n). By lemma 3-2, symbol 1 of every node in a cycle in G(n-rotator - $V_{1,1}$) must be in the same position. In other words, if symbol 1 is in the ith position, the generations in the cycle can only be g_2, g_3, ..., and g_{i-1}. This cycle is therefore being limited in a sub (i-1)-rotator. Our algorithm joins feedback vertex sets of these sub rotators to acquire the feedback vertex set of the whole graph. Step 2 adds all feedback vertex sets of sub graphs whose node has tail sequence length $n-3$, $n-4$, ..., and 1 to the FVS(n). All sub rotators are considered in our algorithm, the aggregation of the feedback vertex set is the feedback vertex set of the whole graph. □

Lemma 3-5. The size of FVS(n) generated by Algorithm 3-1 is $n!/3$.

Proof. The number of nodes in FVS(n) is denoted as |FVS(n)| . We prove this theorem by induction.

1. Because FVS(3)={123, 132}, |FVS(n)| =$n!/3$ holds for $n = 3$.

2. We assume that |FVS(n)|=$n!/3$ is true when $n = k$.

3. When $n = k + 1$,

|FVS(n)|=$k!$ + P_0^k * |FVS(k)| +P_1^k* |FVS(k-1)| + ...+P_{k-3}^k * |FVS(3)|

= $k!$ + P_0^k * $k!/3$ +P_1^k* (k-1)!/3 + ...+ P_{k-3}^k * 3!/3

= $k!$ + $k!/3$ * (k-2)

= (k+1)*$k!/3$

= (k+1)!/3

= $n!/3$. This lemma is proved. □

Lemma 3-6. The size of FVS(n) generated by Algorithm 3-1 is minimum.

Proof. An n-rotator contains $n!/3$ disjoint cycles of size 3 and these cycles are formed $g_3g_3g_3$. Therefore, to eliminate all possible cycles need to remove at least $n!/3$ nodes from an n-rotator. By Theorem 3-2, the size of FVS(n) generated by Algorithm 3-1 is $n!/3$. Thus, it is minimum. □

Lemma 3-7. The time complexity of Algorithm 3-1 is $O(n^{n-3})$, where n is the size of rotator graph.

Proof. Let $t(k)$ be the time complexity of finding FVS(k) by using Algorithm 3-1. We assume $t(3) = 1$ because FVS(3) can be easily observed. The time complexity $t(n)$ is expressed follows:

$$t(n) = \mathrm{P}_0^{n-1} * t(n-1) + \mathrm{P}_1^{n-1} * t(n-2) + \ldots + \mathrm{P}_{n-4}^{n-1} * t(3). \quad (1)$$

$$t(n+1) = \mathrm{P}_0^{n} * t(n) + \mathrm{P}_1^{n} * t(n-1) + \ldots + \mathrm{P}_{n-3}^{n} * t(3). \quad (2)$$

From (1) and (2):

$$\frac{t(n+1)}{n} = \frac{t(n)}{n} + \mathrm{P}_0^{n-1} * t(n-1) + \mathrm{P}_1^{n-1} * t(n-2) + \ldots + \mathrm{P}_{n-4}^{n-1} * t(3). \quad (3)$$

From (2) and (3):

$$\frac{t(n+1)}{n} = \frac{t(n)}{n} + t(n) \quad (4)$$

From (4): $t(n)$= $\prod_{k=4}^{n}$ =$O(n^{n-3})$, $n \geq 3$. □

4 Conclusions

This paper provides an algorithm for finding minimum feedback vertex sets for rotator graphs. Finding minimum Feedback vertex set is a NP-hard problem for general graph. For an n-rotator, our algorithm generates a feedback vertex set of size $n!/3$ in $O(n^{n-3})$. The size of the set is proved to be minimum.

Acknowledgement

This work was partially supported by the National Science Council of the Republic of China under Grant NSC94-2213-E-011-053.

References

1. Peter F. Corbett, Rotator Graphs: An Efficient Topology for Point-to-Point Multiprocessor Networks, IEEE Transactions on Parallel and Distributed System, Vol. 3, No. 5. (1995) 622–626
2. C. Chen, D.P. Agrawal and J.r. Burke, dBCube: A New Class of Hierarchical Multiprocessor Interconnection Networks with Area Efficient Layout, IEEE Trans. on Parallel and Distributed Systems, Vol. 4. (1993) 1332–1344
3. Caragiannis, C. Kaklamanis, P. Kanellopoulos: New Bounds on The Size of The Feedback Vertex Set on Meshes and Butterflies, Information Processing Letters, Vol. 83, No. 5. (2002) 275–280
4. M.M. Flood: Exact and Heuristic Algorithms for The Weighted Feedback Arc Set Problem: A special case of the skew-symmetric quadratic assignment problem, Networks, Vol. 20. (1990) 1–23
5. R. Floyd: Assigning Meaning to Programs, In Proceedings of Symposium on Applied Mathematics. (1967) 19–32
6. R. Focardi, F.L. Luccio, D. Peleg: Feedback Vertex Set in Hypercubes, Information Processing Letters, Vol. 76, No. 1-2. (2000) 1–5
7. M.R. Garey, D.S. Johnson: Computers and Intractability, Freeman, San Francisco, CA. (1979)
8. M. Yannakakis, Node-Deletion Problem on Bipartite Graphs, SIAM Journal on Computing, Vol. 10. (1981) 310–327
9. F.L. Luccio: Exact Minimum Feedback Vertex Set in Meshes and Butterflies, Information Processing Letters, Vol. 66, No. 2. (1998) 59–64
10. I. Caragiannis, C. Kaklamanis, P. Kanellopoulos: New Bounds on The Size of The Feedback Vertex Set on Meshes and Butterflies, Information Processing Letters, Vol. 83, No. 5. (2002) 275–280
11. R. Focardi, F.L. Luccio, D. Peleg: Feedback Vertex Set in Hypercubes, Information Processing Letters, Vol. 76, No. 1-2. (2000) 1–5
12. F.H. Wang, Y.L. Wang, J.M. Chang: Feedback Vertex Sets in Star Graphs, Information Processing Letters, Vol. 89, No. 4. (2004) 203–208
13. R. Kralovic, P. Ruzicka: Minimum Feedback Vertex Sets in Shuffle-based Interconnection Networks, Information Processing Letters, Vol. 86, No. 4. (2003) 191–196