A new Integer Linear Programming and Quadratically Constrained Quadratic Programming Formulation for Vertex Bisection Minimization Problem

Submitted: 11th July 2015; accepted: 10th February 2016

Pallavi Jain, Gur Saran, Kamal Srivastava

DOI: 10.14313/JAMRIS_1-2016/9

Abstract:

Vertex Bisection Minimization problem (VBMP) consists of partitioning a vertex set V of graph G = (V, E) into two sets B and B′ where |B| = $\lfloor |V|/2 \rfloor$ such that vertex width (VW) is minimized where vertex width is defined as the number of vertices in B which are adjacent to at least one vertex in B′. It is an NP-complete problem in general. VBMP has applications in fault tolerance and is related to the complexity of sending messages to processors in interconnection networks via vertex disjoint paths. In this paper, we have proposed a new integer linear programming (ILP) and quadratically constrained quadratic programming (QCQP) formulation for VBMP. Both of them require number of variables and constraints lesser than existing ILPs and QCQP. We have also implemented ILP and obtained optimal results for various classes of graphs. The result of the experiments with the benchmark graphs shows that the proposed model outperforms the state of the art. Moreover, proposed model obtains optimal result for all the benchmark graphs.

Keywords: *Vertex Bisection Minimization, Integer Linear Programming, Quadratic Programming*

1. Introduction

Vertex Bisection Minimization problem (VBMP) consists of partitioning a vertex set of a graph $G = (V, E)$ where $|V| = n$ into two sets B and B' where $|B| = \lfloor n/2 \rfloor$ such that vertex width (VW) is minimized where vertex width is defined as the number of vertices in B which are adjacent to at least one vertex in B'. Mathematically, $VW = |\{u \in B : \exists v \in B' \wedge (u,v) \in E(G)\}|$ [1]. It can also be defined as a graph layout problem. Graph layout problems are a class of combinatorial optimization problems whose goal is to find a layout of an input graph G to optimize a certain objective function [2]. A linear layout or layout of an undirected graph $G = (V, E)$ is the bijective function ϕ: V → $[n] = \{1,2,\dots, n\}$ [2, 3]. Set of all layouts is denoted by $\phi(G)$. Vertex Bisection Minimization Problem consists of finding a layout $\Phi^* \in \Phi(G)$ of a graph $G = (V, E)$ which minimizes the vertex width (VW) where $VW = \delta(\lfloor |V|/2 \rfloor, \Phi, G)$ for a layout φ where, $\delta(i, \phi, G) = |\{u \in L(i, \phi, G) : \exists v \in R(i, \phi, G) \wedge (u,v) \in E(G)\}|$, $L(i, \phi, G) = \{u \in V : \phi(u) \le i\}$ and $R(i, \phi, G) = \{u \in V : \phi(u) > i\}$ [2,3]. Vertex Bisection Minimization Problem is relevant to fault tolerance and is related to the complexity of sending messages to processors in interconnection networks via vertex disjoint paths [2]. Brandes and Fleisher [1] proved that Vertex Bisection minimization problem is NP-complete in general but vertex bisection minimization is polynomially solvable for trees and hypercubes. Fraire *et al.* [4] have proposed two ILP models and one QCQP for VBMP. In this paper, we have proposed a 0-1 ILP model and two QCQP models which requires fewer number of variables and constraints than the one proposed by [4].

When solving a problem using Integer Programming (IP) where all the involved variables are integer variables, we first need to develop an IP formulation involving: (a) an objective function which must be either maximized or minimized, (b) definition of the variables used and (c) a set of the associated constraints [5]. In ILP, the objective function and the constraints are restricted to be linear. 0-1 integer linear programming is a special case in which unknowns are binary [6]. A large number of problems have been formulated as ILP such as Travelling salesman, Vertex Cover and other covering problems, Set Packing and other Packing problems, Max Bisection Minimization problem, Vertex Separation Minimization problem, Cutwidth Minimization Problem, bandwidth minimization problem, MinLA, etc. [5–10].

Quadratically constrained quadratic programming (QCQP) is an optimization problem in which both the objective functions and the constraints are quadratic functions [5].

In this paper, Section 2 describes the 0-1 integer linear programming model for VBMP. Section 3 is devoted to the results of VBMP obtained using ILP for various classes of graphs such as *Small* graphs (*http://www.optsicom.es/cutwidth*), 2-dimensional ordinary meshes, 2-dimensional cylindrical meshes and 2-dimensional toroidal meshes. We have also proposed two quadratically constrained quadratic programming formulation for VBMP which is described in Section 4. Section 5 is devoted to conclusions.

2. 0-1 Integer Linear Programming for VBMP

We first define the variables. These variables take value 0 or 1.

2.1. Variable x_i

This binary variable indicates whether the vertex is placed in B or B'.

$$x_i = \begin{cases} 1, & if\ i \in B \\ 0, & if\ i \in B' \end{cases}, \quad \forall i \in V(G)$$

2.2. Variable v_{ij}

This variable indicates that for an edge (i,j) whether $i \in B \wedge j \in B'$ or not. v_{ij} = 1 if i is placed in B and j is placed in B' otherwise 0. It is defined as follows:

$$v_{ij} = x_i \wedge \overline{x}_j = \begin{cases} 1 & \text{if } i \in B \wedge j \in B' \\ 0 & \text{if } (i,j \in B) \vee (i,j \in B') \vee (i \in B' \wedge j \in B) \end{cases},$$

$$\forall (i,j) \in E(G)$$

2.3. Variable z_i

Variable z_i indicates that whether the vertex i contribute to the vertex width or not. z_i = 1 if v_{ij} = 1 for some $j \in \{1,2,\ldots, n\} \backslash i$ where n is the order of graph G otherwise 0, $\forall i \in V(G)$. It is defined as follows:

$$z_i = \bigvee_{(i,j)\in E(G)} v_{ij} = \begin{cases} 1 & \text{if } \exists (i,j) \in E(G) : v_{ij} = 1 \\ 0 & \text{otherwise} \end{cases}, \ \forall i \in V(G)$$

2.4. ILP Formulation

The following integer linear programming formulation is proposed for VBMP.

$$z_i, x_i, v_{ij} \in \{0,1\}$$

$$min \sum_{i=1}^{n} z_i$$

$$\sum_{i=1}^{n} x_i = \left\lfloor \frac{n}{2} \right\rfloor \qquad (1)$$

$$v_{ij} \leq x_i \qquad \forall (i,j) \in E(G) \qquad (2)$$

$$v_{ij} \leq 1 - x_j \qquad \forall (i,j) \in E(G) \qquad (3)$$

$$x_i - x_j \leq v_{ij} \qquad \forall (i,j) \in E(G) \qquad (4)$$

$$z_i \geq v_{ij} \qquad \forall i \in V(G) \qquad (5)$$

$$z_i \leq \sum_{(i,j)\in E(G)} v_{ij} \qquad \forall i \in V(G) \qquad (6)$$

Constraint (1) guarantees that $|B| = \lfloor n/2 \rfloor$. Constraints (2) – (4) represents that $v_{ij} = x_i \wedge \overline{x}_j$. If x_i = 0, then v_{ij} = 0 using constraint (2). If x_i =1 and x_j = 0, then from using constraints (2) and (3) v_{ij} may be 0 or 1 but constraint (4) guarantees that v_{ij} = 1 not 0. If x_i =1 and x_j = 1, v_{ij} = 0 using constraint (3). Constraints (5) – (6) represents $z_i = \bigvee_{(i,j)\in E(G)} v_{ij}$. If v_{ij} = 0 for all $(i, j) \in E(G)$, where i is fixed, then constraint (5) gives z_i = 0 or 1 but constraint (6) gives that z_i = 0 not 1. If v_{ij} = 1 for some $(i, j) \in E(G)$, where i is fixed, then constraint (5) gives z_i = 1 and equation (6) also satisfies it. Let n and m be the order of graph G and the number of edges in G respectively. It can be observed that there are $2(n+m)$ 0-1 variables and $8m+n$ constraints. In the ILP1 and ILP2 proposed by Fraire *et al.* [4], number of variables and constraints are $O(mn^2)$ and $O(mn^2)$, respectively.

3. Computational Experiments for VBMP

This section describes the computational experiments performed to test the efficacy of 0-1 integer linear programming formulated by us for VBMP. We have solved 0-1 ILP using CPLEX 12.6 using multiple threads on an Intel Xeon 12 core CPU.

We have carried out experiments on the benchmark graphs in [4] which consists of *Small* graphs, Grid, Trees and Harwell-Boeing graphs. These instances are available at *www.optsicom.es/vsp/#instances*. Table 1-4 shows the optimal results achieved and the time taken to obtain the result. As can be seen in the table, the results are obtained in a very short time for all the graphs.

4. Quadratic Programming Formulation

In the QP formulation, x_i is defined in the similar manner as in Section 2. Variable v_i indicates that whether the vertex $i \in V(G)$ contribute to the vertex width or not. It is defined as follows:

$$v_i = \bigvee_{(i,j)\in E(G)} x_i . \overline{x}_j = \begin{cases} 1, if\ \exists (i,j) \in E(G) : i \in B \wedge j \in B' \\ 0, \qquad otherwise \end{cases}$$

4.1. {0,1} Quadratic Programming Formulation for VBMP

$$\boldsymbol{x_i}, \boldsymbol{v_i} \in \{\boldsymbol{0}, \boldsymbol{1}\}$$

$$\max \sum_{i=1}^{n} v_i$$

$$\sum_{i=1}^{n} x_i = \lfloor n/2 \rfloor \qquad (1)$$

$$v_i \geq x_i(1 - x_j) \quad \forall (i,j) \in E(G) \qquad (2)$$

$$\sum_{(i,j)\in E(G)} \left(x_i \cdot (1 - x_j) \right) \geq v_i \qquad (3)$$

In above quadratic programming, equation (1) guarantees that $|B| = \lfloor n/2 \rfloor$. Equation (2) and (3) represents that $v_i = \bigvee_{(i,j)\in E(G)} x_i \cdot \overline{x}_j$. If x_i = 0, then v_i = 0 or 1 from equation (2) but since left hand side of equation (3) is 0, therefore equation (3) gives that v_i = 0 not 1. If x_i = 1 and x_j = 0, then from equation (2) v_i = 1. If x_i = 1 and x_j = 1, then from equation (2) v_i = 0 or 1. If x_i = 1 and x_j = 0 for some $(i,j) \in E(G)$, then equation (3) gives that v_i = 1. If x_i = 1 and x_j = 1 for all $(i,j) \in E(G)$, then equation (3) gives that v_i = 0. Let n and m be the order of graph G and the number of edges in G respectively. It can be observed that there are $O(n)$ 0-1 variables and $O(n.\Delta(G))$ constraints while in Fraire *et al.* [4] number of variables and constraints both are equal to $O(m.n^2)$.

Table 1. Small Graphs

Graphs	Optimal Result	Time (ms)	Graphs	Optimal Result	Time (ms)
p17_16_24	3	57	p59_20_23	2	47
p18_16_21	2	60	p60_20_22	2	53
p19_16_19	2	55	p61_21_22	2	55
p20_16_18	2	46	p62_21_30	3	61
p21_17_20	2	83	p63_21_42	5	200
p22_17_19	2	39	p64_21_22	2	60
p23_17_23	2	84	p65_21_24	2	77
p24_17_29	3	57	p66_21_28	3	86
p25_17_20	2	91	p67_21_22	2	70
p26_17_19	2	51	p68_21_27	3	62
p27_17_19	2	46	p69_21_23	2	53
p28_17_18	2	58	p70_21_25	3	55
p29_17_18	1	51	p71_22_29	3	61
p30_17_19	2	78	p72_22_49	5	198
p31_18_21	2	63	p73_22_29	2	64
p32_18_20	2	57	p74_22_30	3	81
p33_18_21	3	54	p75_22_25	2	48
p34_18_21	2	33	p76_22_30	2	61
p35_18_19	2	64	p77_22_37	4	155
p36_18_20	2	90	p78_22_31	3	47
p37_18_20	2	39	p79_22_29	3	53
p38_18_19	2	71	p80_22_30	3	55
p39_18_19	2	66	p81_23_46	6	175
p40_18_32	4	60	p82_23_24	2	59
p41_19_20	1	63	p83_23_24	1	49
p42_19_24	3	66	p84_23_26	2	58
p43_19_22	2	42	p85_23_26	1	74
p44_19_25	3	82	p86_23_24	2	40
p45_19_25	2	61	p87_23_30	3	108
p46_19_20	2	61	p88_23_26	2	82
p47_19_21	2	52	p89_23_27	3	71
p48_19_21	2	36	p90_23_35	3	60
P49_19_22	2	52	p91_24_33	3	57
P50_19_25	2	49	p92_24_26	2	45
p51_20_28	4	103	p93_24_27	2	49
p52_20_27	2	87	p94_24_31	3	115
p53_20_22	2	45	p95_24_27	2	61
p54_20_28	3	69	p96_24_27	2	65
p55_20_24	2	63	p97_24_26	2	55
p56_20_23	3	68	p98_24_29	2	69
p57_20_24	2	47	p99_24_27	2	72
p58_20_21	2	55	p100_24_34	3	94

Table 2. Grids

Graphs	Optimal Result	Time (ms)	Graphs	Optimal Result	Time (ms)
Grid3	3	53	Grid4	4	82
Grid5	5	247	Grid6	6	258
Grid 7	7	348			

Table 3. Trees

Graphs	Optimal Result	Time (ms)	Graphs	Optimal Result	Time (ms)
1rot_Tree_22_3_rot1.mtx	2	94	2rot_Tree_22_3_rot1.mtx	2	57
1rot_Tree_22_3_rot2.mtx	2	100	2rot_Tree_22_3_rot2.mtx	2	68
1rot_Tree_22_3_rot3.mtx	2	61	2rot_Tree_22_3_rot3.mtx	2	52
1rot_Tree_22_3_rot4.mtx	2	87	2rot_Tree_22_3_rot4.mtx	2	67
1rot_Tree_22_3_rot5.mtx	2	55	2rot_Tree_22_3_rot5.mtx	2	95
3rot_Tree_22_3_rot1.mtx	2	72	3rot_Tree_22_3_rot3.mtx	2	67
3rot_Tree_22_3_rot2.mtx	2	91	3rot_Tree_22_3_rot4.mtx	2	68
3rot_Tree_22_3_rot5.mtx	22	52			

Table 4. Harwell-Boeing Graphs

Graphs	Optimal Result	Time (ms)	Graphs	Optimal Result	Time (ms)
bcspwr01	3	98	bcspwr02	2	92
bcsstk01	12	1909	can24	4	88

4.2. {-1,1} Quadratic Programming Formulation for VBMP

$$x_k, v_k \in \{-1, 1\}$$

$$min \sum_{i=1}^{n} v_i$$

$$\sum_{i=1}^{n} x_i \leq 0 \quad (1)$$

$$\sum_{i=1}^{n} x_i \geq -1 \quad (2)$$

$$v_i \leq x_i \quad (3)$$

$$v_i \leq \sum_{(i,j)\in E(G)} \left(\frac{1-x_j}{2}\right) \quad (4)$$

$$v_i \geq x_i \left(\frac{1-x_j}{2}\right) - 1 \ \ \forall (i,j) \in E(G) \quad (5)$$

$x_i = 1$ if $i \in B$ otherwise $x_i = -1$

Equation (1) and (2) guarantees the bipartition of the vertex set. If $x_i = 1$ and $x_j = 1$, (3) gives $v_i = 1$ or -1. If for all $(i, j) \in E(G)$ where i is fixed, $x_j = 1$ then $v_i = -1$ using (4) and (5) also satisfies it. If $x_i = 1$ and $x_j = -1$, (3) gives $v_i = 1$ or -1. (4) also implies $v_i = 1$ or -1 and (5) ensures that $v_i = 1$. If $x_i = -1$ and $x_j = 1$, (3) gives $v_i = -1$. (4) and (5) also satisfies it. If $x_i = -1$ and $x_j = -1$, (3) gives $v_i = -1$. (4) and (5) also satisfies it. Let n and m be the order of graph G and the number of edges in G respectively. It can be observed that in this formulation also there are $O(n)$ {-1,1} variables and $O(n.\Delta(G))$ constraints.

5. Conclusions

In this paper, a 0-1 Integer linear programming model and two quadratically constrained quadratic programming models for the Vertex Bisection problem have been proposed. These formulations involve fewer number of variables and constraints than the earlier ILPs and QP which have been proposed in the literature. We have also experimented with the ILP for the Vertex Bisection Minimization problem. It can be observed that, optimal results are achieved very quickly for all the benchmark graphs that were experimented on.

ACKNOWLEDGMENTS

We thank Dr. Volkmar Sauerland, Christian-Albrechts-Universitat zu Kiel for CPLEX runs. The first author also acknowledges UGC for MANF fellowship [MANF-2014-15-JAI-UTT-38830].

AUTHORS

Pallavi Jain*, Gur Saran, Kamal Srivastava – Department of Mathematics, Dayalbagh Educational Institute, Agra, India. Tel: +91-9690000126.
E-mail: pallavijain.t.cms@gmail.com.

*Corresponding author

REFERENCES

[1] Brandes U., Fleischer D., "Vertex Bisection is Hard, too", *Journal of Graph Algorithms and Applications*, vol. 13, no. 2, 2009, 119–131. DOI: 10.7155/jgaa.00179.

[2] Diaz J., Petit J., Serna M., "A Survey of Graph Layout Problems", *ACM Computing Surveys*, vol. 34, no. 3, 2002, 313–356. DOI: 10.1145/568522.568523.

[3] Petit J., "Addenda to the Survey of Layout Problems", *Bulletin of the European Association for Theoretical Computer Science* (EATCS), no. 105, 2011, 177–201.

[4] Fraire H., David J., Villanueva T., et al., "Exact Methods for the Vertex Bisection Problem". In: *Recent Advances on Hybrid Approaches for Designing Intelligent Systems*, ed. by: O. Castillo, P. Melin, W. Pedrycz, J. Kacprzyk, chapter 40, series: Studies in Computational Intelligence vol. 547, 2014, 567–577. DOI: 10.1007/978-3-319-05170-3_40.

[5] Vazirani V.V., *Approximation Algorithms*, Springer 2003. DOI: 10.1007/978-3-662-04565-7.

[6] Diaby M., "The Travelling Salesman Problem: Linear Programming Formulation", *WSEAS Transactions on Mathematics*, no. 6, 2007, 745–754.

[7] Frieze A., Jerrum M., "Improved Approximation Algorithms for Max k-cut and Max Bisection", *Algorithmica*, vol. 18, no. 1, 1997, 67–81. DOI: 10.1007/BF02523688.

[8] Duarte A., Escudero L.F., Marti R., et al., "Variable Neighborhood Search for the Vertex Separation Problem", *Computers and Operations Research*, vol. 39, no. 12, 2012, 3247–3255. DOI: 10.1016/j.cor.2012.04.017.

[9] Loces M.C.L., Garcia N.C., Huacuja H.J.F., et al., "A New Integer Linear Programming Model for the Cutwidth Minimization Problem of a Connected Undirected Graph", In: *Recent Advances on Hybrid Approaches for Designing Intelligent Systems*, ed. by: O. Castillo, P. Melin, W. Pedrycz, J. Kacprzyk, chapter 40, series: Studies in Computational Intelligence vol. 547, 2014, 509–517. DOI: 10.1007/978-3-319-05170-3_35.

[10] Luttamaguzi J., Pelsmajer, M., Shen, Z., Yang, B., *Integer Programming Solutions for Several Optimization Problems in Graph Theory. Technical report*, DIMACS 2005.