INTEGER PROGRAMMING FORMULATIONS AND BENDERS DECOMPOSITION FOR MAXIMUM INDUCED MATCHING PROBLEM

by

Betül Ahat

B.S., Industrial Engineering, Boğaziçi University, 2013

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Industrial Engineering
Boğaziçi University

2016

INTEGER PROGRAMMING FORMULATIONS AND BENDERS DECOMPOSITION FOR MAXIMUM INDUCED MATCHING PROBLEM

APPROVED BY:

Assoc. Prof. Z. Caner Taşkın .....
(Thesis Supervisor)

Assoc. Prof. Tınaz Ekim .....
(Thesis Co-supervisor)

Prof. İ. Kuban Altınel .....

Prof. Necati Aras .....

Asst. Prof. İbrahim Muter .....

DATE OF APPROVAL: 11.01.2016

ACKNOWLEDGEMENTS

First of all, I would like to thank my supervisors Assoc. Prof. Z. Caner Taşkın and Assoc. Prof. Tınaz Ekim for their support and patience throughout my entire graduate study. This research would have been impossible without their guidance and encouragement. I am also very grateful to Prof. İ. Kuban Altınel, Prof. Necati Aras and Asst. Prof. İbrahim Muter for taking part in my thesis committee and for their valuable suggestions. I also want to thank Prof. Refik Güllü, Prof. Taner Bilgiç and Assoc. Prof. Aybek Korugan for their kindness and support.

I want to thank Beste and Ceyda for their encouragement, the great times we spent together as friends and colleagues. My friends Tuğba, Hazan, Başak, Nimet, Sedef, Nurbanu and Vildan played a big role in making my undergraduate years more enjoyable. I also want to thank Oylum, Şeyma, Yücel, Kübra, Banu, Zeynep, Nisa, Gökçe and Cemil for their friendship.

I would like to thank my dear friends Fatma and Büşranur for their invaluable support and friendship since the first years of collage. They always helped me to make right decisions when I felt doubtful. I thank Betül Cansu and Gamze for their patience while listening the difficulties I encountered. I am grateful to my cousin Tuğba for being my closest friend and cheering me up when I feel tired. I want to thank Filiz, with whom I shared the same dorm room. I feel warmth of her love even when she is not with me.

I will never be able to find the correct words to express my gratitudes to my parents and my brother for their love and patience. I am thankful for trusting and encouraging me throughout my life. I am sure that things would be much harder without their support and love .

I also thank TUBITAK for their financial support during my graduate study.

ABSTRACT

INTEGER PROGRAMMING FORMULATIONS AND BENDERS DECOMPOSITION FOR MAXIMUM INDUCED MATCHING PROBLEM

In this thesis, we investigate Maximum Induced Matching problem (MIM), finding an induced matching having the largest cardinality. The problem is NP-hard for general graphs. We develop a binary integer programming formulation with less decision variables compared to other formulations in the literature. Then, we extend the problem for vertex-weighted graphs and introduce Maximum Vertex-Weighted Induced Matching problem (MVWIM). We introduce edge-weighted version of MIM and call it Maximum Edge-Weighted Induced Matching problem (MEWIM). We adapt formulations found in the literature and our formulation to solve MVWIM and MEWIM instances. In generalized version of Maximum Weighted Induced Matching problem (MWIM), we assume both vertices and edges have weights, and give a binary integer programming formulation for it. Since it has many decision variables and constraints, we implement Benders decomposition approach to partition the problem into smaller problems. Then, we add some valid inequalities to our formulation to improve the efficiency of our algorithm. To test the performance of our methodology, we generate random graphs with different densities. By looking at computational results, it can be seen that our approach solves instances with medium and large densities significantly faster than other methods in the literature.

ÖZET

MAKSİMUM TETİKLENMİŞ EŞLEŞTİRME PROBLEMİ İÇİN TAMSAYI PROGRAMLAMA FORMÜLASYONU VE BENDERS AYRIŞTIRMASI

Bu tezde Maksimum Tetiklenmiş Eşleştirme problemini (MIM), en büyük boyuta sahip tetiklenmiş eşleştirme bulma, inceledik. Bu problem genel çizgeler için NP-zor'dur. Literatürde bulunan formülasyonlardan daha az karar değişkenine sahip tamsayı programlama formülasyonu geliştirdik. Daha sonra, bu problemi düğüm-ağırlıklı çizgeler için genişleterek Maksimum Düğüm-Ağırlıklı Tetiklenmiş Eşleştirme problemini (MVWIM) tanıttık. MIM'in kenar-ağırlıklı versiyonunu tanıttık ve buna Maksimum Kenar-Ağırlıklı Tetiklenmiş Eşleştirme problemi (MEWIM) adını verdik. Literatürde bulunan formülasyonları ve bizim formülasyonumuzu MVWIM ve MEWIM örneklerini çözmek için uyarladık. Maksimum Ağırlıklı Tetiklenmiş Eşleştirme probleminin (MWIM) genelleştirilmiş versiyonunda hem düğümler hem de kenarların ağırlıklı olduğunu varsaydık ve bunun için bir tamsayı programlama formülasyonu verdik. Bu formülasyonda çok fazla karar değişkeni ve kısıt olduğundan, problemi daha küçük problemlere bölmek için Benders parçalama yaklaşımı uyguladık. Daha sonra, algoritmamızın verimliliğini geliştirmek için formülasyonumuza bazı geçerli eşitsizlikler ekledik. Metodumuzun performansını test edebilmek için, farklı yoğunluklara sahip rassal çizgeler ürettik. Sayısal sonuçlara bakarak, yaklaşımımızın literatürde bulunan diğer formülasyonlara göre orta ve büyük yoğunluğa sahip örnekleri belirgin derecede daha hızlı çözdüğü görülebilir.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii
ABSTRACT iv
ÖZET v
LIST OF FIGURES vii
LIST OF TABLES viii
LIST OF SYMBOLS ix
LIST OF ACRONYMS/ABBREVIATIONS x
1. INTRODUCTION AND LITERATURE REVIEW 1
2. PROBLEM FORMULATIONS 16
2.1. Maximum Induced Matching Problem (MIM) 16
2.2. Maximum Vertex-Weighted Induced Matching Problem (MVWIM) . . 19
2.3. Maximum Edge-Weighted Induced Matching Problem (MEWIM) . . . 19
3. BENDERS DECOMPOSITION FOR GENERALIZED MWIM 22
4. ALGORITHMIC IMPROVEMENTS 29
4.1. Valid Inequalities 29
4.2. Formulation Tightening 30
4.3. Single Branch-and-Bound Tree 31
5. COMPUTATIONAL RESULTS 33
6. CONCLUSION AND FURTHER RESEARCH 44
REFERENCES 46

LIST OF FIGURES

Figure 1.1. An example of matching that is not an induced matching 1

Figure 1.2. An example of induced matching 1

Figure 1.3. An example of subgraph (G') and induced subgraph (G'') 2

Figure 1.4. A graph with a unique maximum induced matching $\{(1,2),(4,7)\}$ of size 2 3

Figure 1.5. An example of induced matching in G which corresponds to an independent set in $L(G)^2$ 4

Figure 1.6. An example of a graph whose square of line graph is not claw-free 5

Figure 1.7. An example of wireless ad-hoc network 6

Figure 1.8. An example of a tree T where $L(T)^2$ is not a tree 10

Figure 1.9. A strong edge coloring of Petersen graph with five colors 12

Figure 4.1. An example of construction of maximum weighted induced matching problem for vertex i 31

LIST OF TABLES

Table 5.1. Comparison of formulations for solving MIM 35

Table 5.2. Effect of formulation tightening method described in Section 4.2 for MIM formulation 36

Table 5.3. Effect of different θ_{ij} selection in the solution of $DSP(\hat{x})$ 37

Table 5.4. Effect of generating multiple cuts in each iteration 38

Table 5.5. Comparison of formulations for solving MWIM 40

Table 5.6. Comparison of formulations for solving MWIM 41

Table 5.7. Comparison of formulations for solving MWIM 42

Table 5.8. Comparison of formulations for solving MWIM 43

LIST OF SYMBOLS

c_i	weight of vertex i
D	density of a graph
$deg(v)$	degree of vertex v
E	edge set of graph G
G'	subgraph of graph G
G^k	kth power of graph G
$L(G)$	line graph of G
N_{ij}	the set of edges that are adjacent to edge (i, j)
$N(x)$	open neighborhood of vertex x
$N[x]$	closed neighborhood of vertex x
$OOIR(G)$	maximum cardinality of an oo-irredundant set of vertices in G
V	vertex set of graph G
w_{ij}	weight of edge (i, j)
θ_{ij}	proportion of w_{ij} that α_{ij} will take
ρ	approximation ratio

LIST OF ACRONYMS/ABBREVIATIONS

ACK	Acknowledgement
APX	Approximable
AT	Asteroidal triple
CTS	Clear-to-send
CV2013	Christou and Vassilaras' formulation for MIM in [1]
DSP	Dual of Subproblem
D2EMIS	Maximum Distance-2 Matching Problem
GRASP	Greedy Randomized Adaptive Search Procedures
LAT	Line-asteroidal triple
LexBFS	Lexicographic Breadth-First Search
LB	Lower Bound
LP	Linear Programming
MAC	Media Access
MEWIM	Maximum Edge-Weighted Induced Matching Problem
MIM	Maximum Induced Matching Problem
MIP	Mixed Integer Programming
MP	Master Problem
MVWIM	Maximum Vertex-Weighted Induced Matching Problem
MWIM	Maximum Weighted Induced Matching Problem
NP	Nondeterministic Polynomial Time
RTS	Request-to-send
SP	Subproblem
UB	Upper Bound
VC2011	Vassilaras and Christou's formulation for MIM in [2]
VC2011 MWIM	Reformulation of Vassilaras and Christou's model in [2] for MWIM

1. INTRODUCTION AND LITERATURE REVIEW

For a given graph $G = (V, E)$, two edges are called adjacent if they have a common end point. A matching is a set of edges such that no two of which are adjacent [3]. An induced matching in a graph G is a matching such that no two edges in the matching are joined by an edge of G [4]. In Figure 1.1, the edge set $\{(1, 5), (2, 6), (4, 8)\}$ forms a matching since it does not contain two edges that are adjacent. However, edges $(1, 5)$ and $(2, 6)$ have a common adjacent edge $(2, 5)$. Therefore, it is not an induced matching. In Figure 1.2, the edge set $\{(1, 5), (6, 7), (4, 8)\}$ is an induced matching since no two edges in the set are joined by another edge of G.

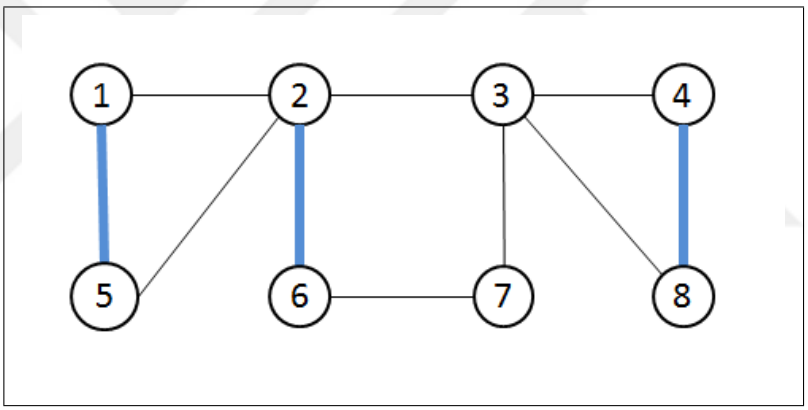

Figure 1.1. An example of matching that is not an induced matching

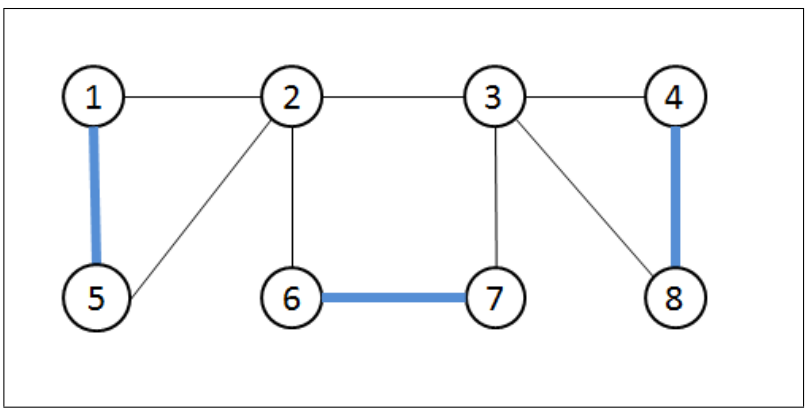

Figure 1.2. An example of induced matching

The graph $G' = (V', E')$ is a subgraph of $G = (V, E)$ if vertices and edges of G' form subsets of the vertices and edges of G. A subgraph G' is said to be induced if for

every pair of vertices u and v of G', (u, v) is an edge of G' if and only if (u, v) is an edge of G. In other words, G' is an induced subgraph of G if it has exactly the edges that appear in G over the same vertex set [5]. In Figure 1.3, G shows the original graph. Then, G' is a subgraph of G since it contains a subset of vertices and edges of G, but it is not an induced subgraph since edge $(4, 5)$ is not in G'. However, G'' is an induced subgraph of G. Then, an induced matching is a matching forming an induced subgraph [4].

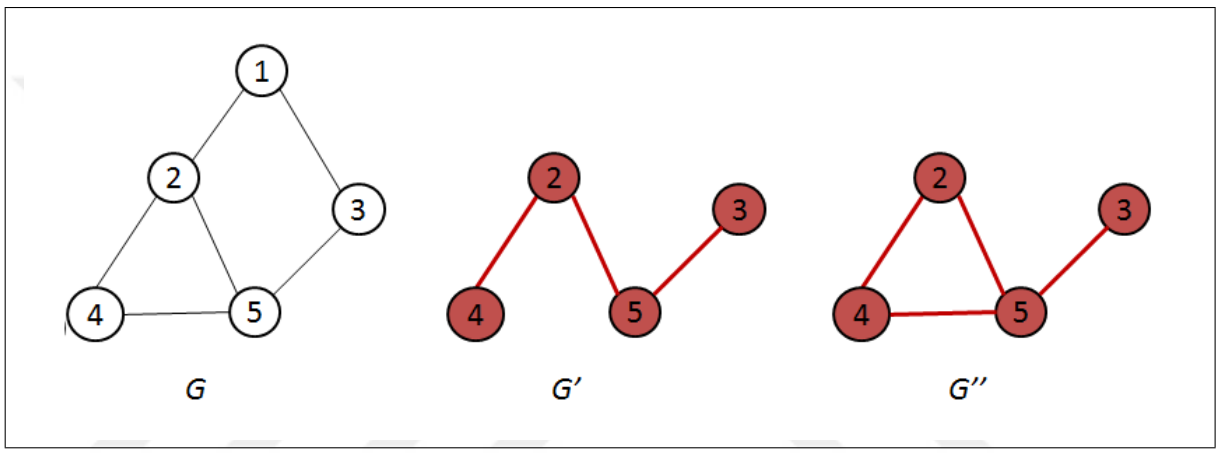

Figure 1.3. An example of subgraph (G') and induced subgraph (G'')

The size (or cardinality) of an induced matching is the number of edges in the induced matching. An induced matching is maximum if its size is the largest among all possible induced matchings. The Maximum Induced Matching problem (MIM) aims to find an induced matching with the largest cardinality [6]. In Figure 1.4, the edge set $\{(1, 2), (4, 7)\}$ is an induced matching whose size is 2. In fact, we cannot find another induced matching of size greater than or equal to 2. Therefore, it is a unique maximum induced matching.

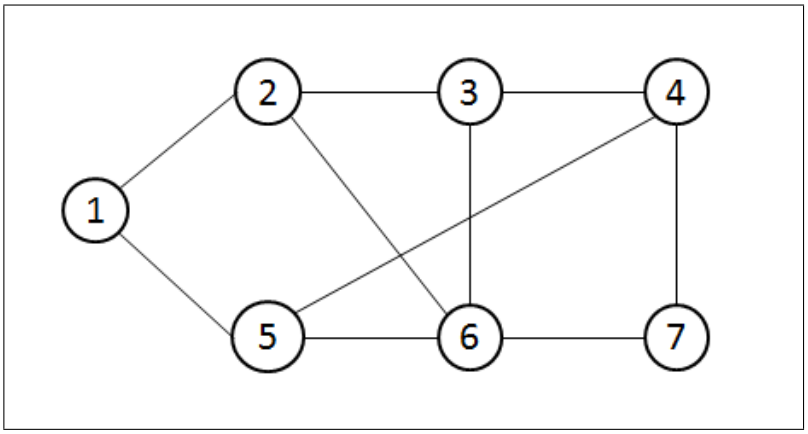

Figure 1.4. A graph with a unique maximum induced matching $\{(1,2),(4,7)\}$ of size 2

The line graph of G, denoted by $L(G)$, is defined as the intersection graph whose vertices correspond to the edges of G, and two vertices in $L(G)$ are adjacent if and only if their corresponding edges in G share a common vertex [7].

The distance between two vertices $u, v \in V$ is the number of edges on a shortest path from u to v in G [8]. Let k be a positive integer. The kth power of G, denoted as G^k, is the graph with the same vertex set as G such that two vertices are adjacent in G^k if and only if their distance in G is at most k. In particular, the square G^2 of a graph G has vertex set V and two vertices are joined in G^2 only if they are joined by an edge or a path of two edges in G [9].

A set of vertices is called independent if no two of them are joined by an edge. The size of an independent set is the number of vertices it contains. An independent set is maximum if its size is the largest among all possible independent sets [9]. Then, for any graph G, every induced matching in G corresponds to an independent set of vertices in the square of line graph of G, denoted by $L(G)^2$, and conversely since two edges induce a $2K_2$ (i.e. a disconnected subgraph consisting of four vertices and two non-incident edges) in G if and only if they are of distance at least 2 in $L(G)^2$. Also, finding a maximum induced matching in G is equivalent to finding a maximum independent set in $L(G)^2$ [4]. In Figure 1.5, the edge set $\{a, b\}$ is an induced matching in G and it corresponds to an independent set of vertices in $L(G)^2$.

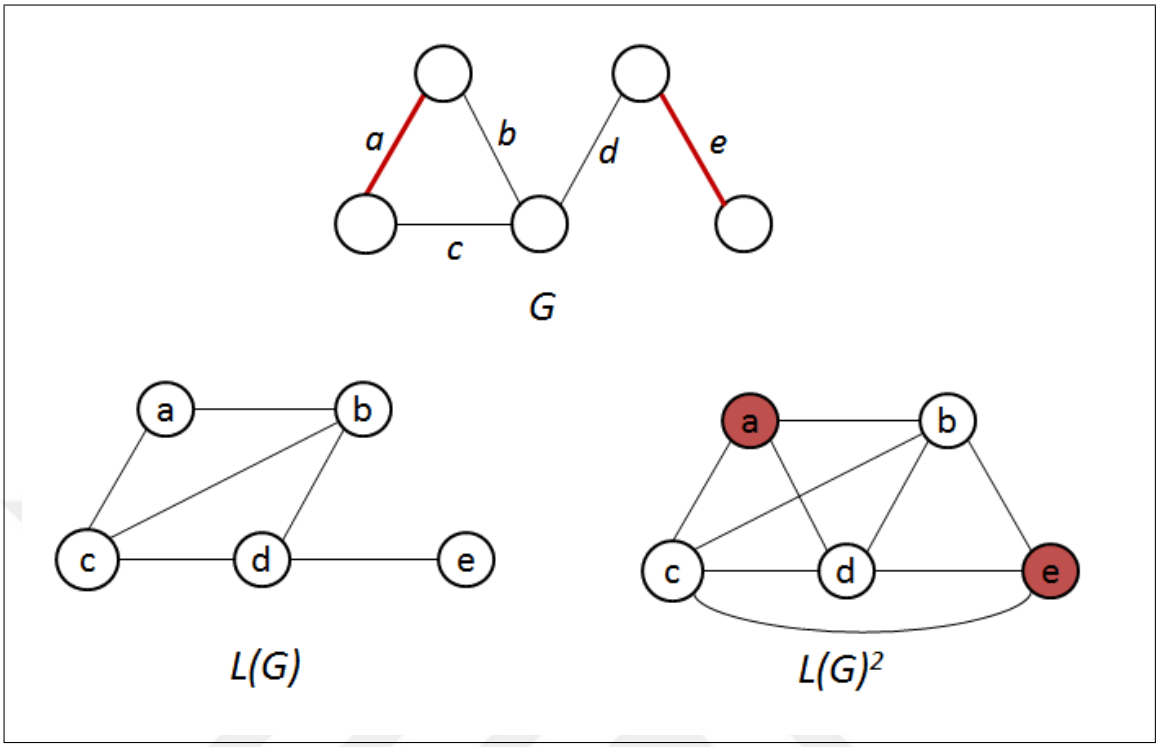

Figure 1.5. An example of induced matching in G which corresponds to an independent set in $L(G)^2$

A claw is a complete bipartite graph $K_{1,3}$. A graph is claw-free if it does not contain a claw as an induced subgraph. The line graph of any graph is claw-free, and there is a polynomial time algorithm for finding a maximum independent set of vertices in claw-free graphs [10]. However, squares of line graphs do not need to be claw-free as can be seen from Figure 1.6, in which the bold edges of G form a claw in $L(G)^2$. Therefore, MIM can be solved in polynomial time whenever the maximum independent set problem can be solved in polynomial time for $L(G)^2$. This property is heavily used to show that the problem is polynomial time solvable for some restricted graph classes [11].

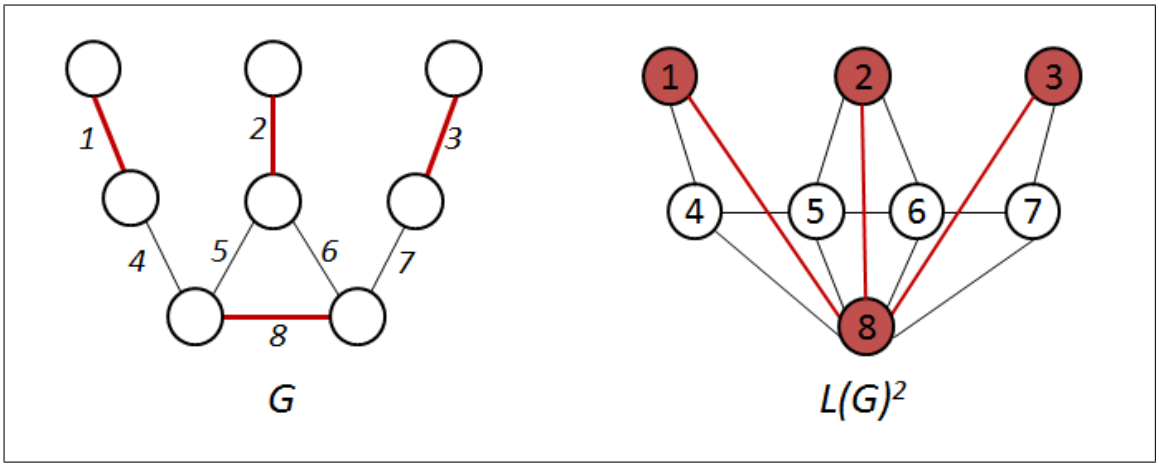

Figure 1.6. An example of a graph whose square of line graph is not claw-free

The distance between edges e and e' is the length of a shortest path from a vertex of e to a vertex of e'. Stockmeyer and Vazirani [8] defined maximum δ-separated matching in which the distance between any two edges in the matching is at least δ. For $\delta = 1$, the problem is equivalent to regular maximum matching problem, which can be solved in polynomial time using Edmond's blossom algorithm [12]. Stockmeyer and Vazirani [8] showed that, for any integer $\delta > 1$, the related decision problem is NP-complete. Also, they proved that the problem is NP-complete even for bipartite graphs of degree 4.

For $\delta = 2$, 2-separated matching is equivalent to induced matching as the distance between any two edges in an induced matching is at least 2. Stockmeyer and Vazirani [8] called maximum 2-separated matching as Risk-free Marriage Problem, finding the maximum number of married couples such that every person is not compatible with any married people other than the person (s)he is married to. The same problem is also called as strong matching [13] or maximum distance-2 matching (D2EMIS) [14] in the literature.

Induced matching problem is extensively studied due to its theoretical and practical importance. In practice, induced matchings are also heavily used in communication industry. For a bipartite graph $G = (X, Y, E)$, let edges represent communication capabilities between broadcaster vertices in X and receiver vertices in Y. For secure

communication channels, the aim is to select k edges e_i $(i = 1, ..., k)$ such that messages on channel i will be passed from broadcaster $X(e_i)$ to receiver $Y(e_i)$ by ensuring that it is impossible for a message broadcast on channel i to be leaked or intercepted. Induced matchings are also used in applications for VLSI and network flow problems. [15]

Another application area of induced matching is as follows: for general graphs that are not necessarily bipartite, let each vertex represent a spy and edges show that the corresponding spies who know each other. Then, an induced matching of size k is equivalent to selecting k pairs of spies to work together such that no active spy knows any other spies except his/her partner [15].

Balakrishnan *et al.* [14] studied the problem of determining the maximum capacity of the media access (MAC) layer in wireless ad-hoc networks. Due to spatial contention for the shared wireless medium, nodes in wireless networks that are close to each other in space may not be able to transmit data concurrently. A MAC protocol in each node enables these nodes to resolve channel contention and avoid collisions. The maximum number of possible concurrent transmissions at the media access layer that are possible in an ad hoc wireless network is an estimate of the maximum network capacity. This problem can be modeled as a maximum induced matching problem in the underlying wireless network. In their paper, Balakrishnan *et al.* called the problem as Maximum Distance-2 matching (D2EMIS) problem since the distance between any two edges in the matching is at least 2.

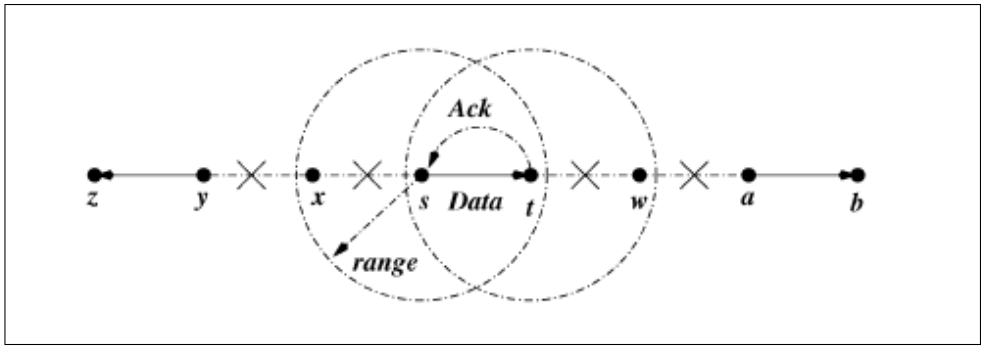

Figure 1.7. An example of wireless ad-hoc network

Figure 1.7 is an example of ad-hoc network topology. It shows the set of links that can communicate concurrently using virtual carrier sensing, where s is the sender and t is the receiver. First, node s initiates communication to a neighbor by broadcasting a "request-to-send" (RTS) message. Upon hearing this message, all of s's neighbors remain silent for a certain period of time. If t is willing to accept (i.e., it has not heard any other on-going transmissions), it responds with a "clear-to-send" (CTS) broadcast message that contains information (e.g., packet length) informing all the nodes of the duration of the data transmission will last. Upon hearing a successful CTS message, all of t's neighbors keep silent for some amount of time (i.e., do not send an RTS or respond with a CTS). By this way, node s can transmit a data frame to t, and in return, t sends a link-layer acknowledgment (ACK) that informs a successful transmission. During this period of time, s and t's neighbors marked with X's in the figure keep silent and cannot send any transmisson [14].

Balakrishnan *et al.* [14] modeled radio networks as geometric intersection graphs and introduced graph theoretical modeling of this problem. Assume we are given a graph $G = (V, E)$, where the vertices of the graph denote the nodes and an edge (u, v) represents the fact that u is in transmission range of v. The aim is to choose a subset of edges on which the transmission can occur without conflict. Then, if (s, t) and (s', t') carry simultaneous transmissions, none of the edges (s, s'), (s, t'), (s', t) or (t, t') can be active in the interference graph. With this constraint, the set of edges that can be chosen is equivalent to an induced matching in G.

Since wireless networks are represented by disk graphs, which are intersection graphs of a family of disks in the Euclidean space, Balakrishnan *et al.* [14] mainly focused on these graphs. They proposed an efficient way to compute an upper bound on the maximum wireless network capacity. They also gave a distributed algorithm for the problem for unit disk graphs where the transmission ranges of all nodes are assumed to be equal. They proved that the greedy solution gives a constant factor approximation for the D2EMIS problem in unit disk graphs. They provided numerical results of their greedy solution for different transmission ranges, number of wireless nodes and probability distributions for the nodes placement in a unit square and showed that the

maximum network throughput depends on the number of nodes and the distribution of the nodes in the unit square.

Later, Vassilaras and Christou [2] studied the same problem and extended the results obtained in [14]. To the best of our knowledge, this was the first study in the literature where the problem has been addressed from mathematical programming point of view. They developed an integer programming based exact algorithm to solve the D2EMIS problem in unit disk graphs. The details of their integer programming formulation is given in Section 2. They also proposed a greedy algorithm to approximate the capacity of the network in polynomial time. They compared the exact and approximate solutions for different network sizes and node distributions. Experimental results show that their greedy solution procedure provides a good approximation of the exact solution even for small network sizes.

A set S $\in V$ is called a k-packing if for all pairs of distinct vertices $u, v \in V$, the distance between u and v is at least k. A maximum k-packing is defined as a k-packing with the maximum cardinality of all possible k-packings in G. Then, any distance-k matching problem can be transformed into an equivalent k-packing problem by taking the line graph of G since there always exist a line graph $L(G)$. However, the reverse may not be possible because we can find a graph L with $\max_{v \in V(L)}\{deg(v)\} > 2$, for which there is no graph G such that the line graph of G is L [1].

The maximum k-packing problem is NP-hard for all k in general. Christou and Vassilaras [1] described a highly parallel distributed algorithm to obtain a near optimal solution to the maximum 2-packing problem, and therefore for the D2EMIS problem. The proposed algorithm is in the category of Greedy Randomized Adaptive Search Procedures (GRASP) for solving combinatorial optimization problems, which have been successfully applied to a large number of problems. To test the applicability of their algorithm, they also proposed an integer programming formulation for D2EMIS problem, which we further disscuss in Section 2. Their experiments show that for large graph instances, their algorithm can quickly find near-optimal solutions whereas commercial MIP solver Gurobi fails to solve the problem to optimality as computer

memory is insufficient for the solver.

Induced matchings have also stimulated a great deal of interest in the discrete mathematics community. Cameron [4] and Stockmeyer and Vazirani [8] showed that MIM is NP-hard for general graphs, even for bipartite graphs. However, the problem is shown to be polynomial time solvable for some graph classes, which we describe next.

A graph is called chordal if for each cycle C in G with four or more vertices, there is an edge of G joining nonconsecutive vertices of C. This edge is called a chord. A clique-neighborhood is defined as the set K of edges of a clique together with some edges each of which is incident to a member of K. Cameron [4] showed that for a chordal graph $G = (V, E)$, $L(G)^2$ itself is also chordal. As the maximum independent set problem can be solved in polynomial time in chordal graphs [16], maximum induced matching problem can be solved in polynomial time by solving maximum independent set problem in $L(G)^2$. In addition, if G is chordal, the cardinality of a maximum induced matching in G equals the cardinality of a minimum set of clique-neighborhoods in G which covers $E(G)$ [4].

A tree T is a graph in which any two vertices are connected by exactly one path. For a tree T, $L(T)^2$ does not have to be a tree as can be seen in Figure 1.8. However, since trees are a special case of chordal graphs, for every tree T, $L(T)^2$ is a chordal graph. Therefore, there is a polynomial-time algorithm for finding the maximum induced matching for trees [4]. However, $L(T)^2$ may have as many as $O(n^2)$ edges, for example when T is a tree of height one with n vertices. Thus, this algorithm does not run in linear time. Fricke and Laskar [17] showed that determining the size of a maximum induced matching in a tree can be done in linear time.

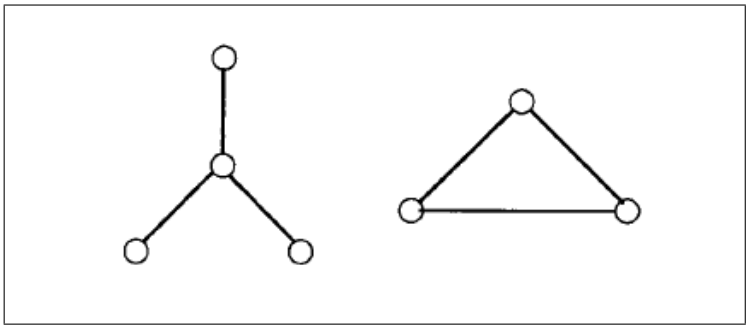

Figure 1.8. An example of a tree T where $L(T)^2$ is not a tree

It has been an open problem if there is a linear-time algorithm for MIM on chordal graphs or not since the construction of $L(G)^2$ requires $O(m^2)$ time, where m is the number of edges in G. Brandstädt and Hoàng [18] presented a linear time algorithm for MIM based on perfect elimination ordering and Lexicographic Breadth-First Search (LexBFS) on chordal graphs.

For a given family F of non-empty sets, the intersection graph of F has a vertex set F and an edge between u and v only if they intersect. Some subclasses of intersection graphs include interval graphs, where F is a set of intervals on a line; chordal graphs, where F is a set of subtrees of a tree; circular-arc graphs, where F is a set of arcs of a circle; circle graphs,where F is a set of chords of a circle; polygon-circle graphs, where F is a set of convex polygons inscribed on a circle. Cameron [11] proved that for some subclasses, if G has a nice representation as an intersection graph, $L(G)^2$ is also an intersection graph and if the independent set problem is polynomial time solvable in $L(G)^2$, the induced matching problem is also polynomial time solvable in G. In particular, she showed that MIM is polynomial time solvable for polygon-circle graphs, asteroidal triple-free graphs and interval-filament graphs.

Golumbic and Lewenstein [15] improved the time complexity of the problem for interval graphs to linear time and proved that induced matching problem is polynomial-time solvable for trapezoid graphs, k-interval-dimension graphs and cocomparability graphs by extending the techniques used for interval graphs. They gave a linear time algorithm for finding a maximum induced matching in a tree, which is simpler than

the algorithm suggested in [17].

An asteroidal triple (AT) of a graph is a set of three vertices such that any two of them are joined by a path avoiding the closed neighborhood of the third. A graph is called asteroidal triple-free (AT-free) if it does not contain an AT. Chang [19] studied the maximum induced matching problem on classes of graphs related to AT-free graphs. He defined a wider class of graphs called the line-asteroidal triple-free (LAT-free) graphs. Using characterization of LAT-free graphs, he showed that maximum induced matching problem and its generalization, the maximum δ-separated matching problem on LAT-free graphs can be solved in polynomial time. This result can be extended to the classes of graphs with bounded asteroidal index. He also proposed a linear-time algorithm for finding a maximum induced matching in a bipartite permutation (bipartite AT-free) graphs using the greedy approach.

For a graph $G = (V, E)$ the open neighborhood $N(x)$ of a vertex x is the set of vertices adjacent to x, and the closed neighborhood of x is the set $N[x] = x \cup N(x)$. A set of vertices X is called open-open irredundant (oo-irredundant) if for every $x \in X$, we have $\mathrm{N}(x) - N(X - x) \neq \emptyset$. $OOIR(G)$ denotes the maximum cardinality of an oo-irredundant set of vertices in G. Golumbic and Laskar [13] stated that every induced matching is also an oo-irredundant set in G. They also gave a relationship between the size of an induced matching and the irredundancy number for general graphs. They showed that for circular arc graphs, the maximum number of induced matching equals $\lfloor OOIR(G) \rfloor$.

A strong edge k-coloring of G is a proper k-coloring of the edges such that no edge is adjacent to two edges of the same color, i.e., a partitioning of its edge set into k induced matchings. Therefore, MIM is used as a sub-task in finding strong edge coloring where each color class forms an induced matching [20]. Strong chromatic index of a graph is defined as the minimum cardinality of strong edge-coloring. The chromatic number of a graph is the minimum size of a partitioning of its vertices into independent sets. Since finding maximum induced matching in a graph G is equivalent to finding a maximum independent set in $L(G)^2$, it follows that the strong chromatic

index of G equals the chromatic number of $L(G)^2$ [11]. If G has m edges and admits a strong edge k-coloring, then a largest induced matching in G has size at least m/k [21]. Figure 1.9 shows an example of strong edge coloring of Peterson graph [22] with five colors.

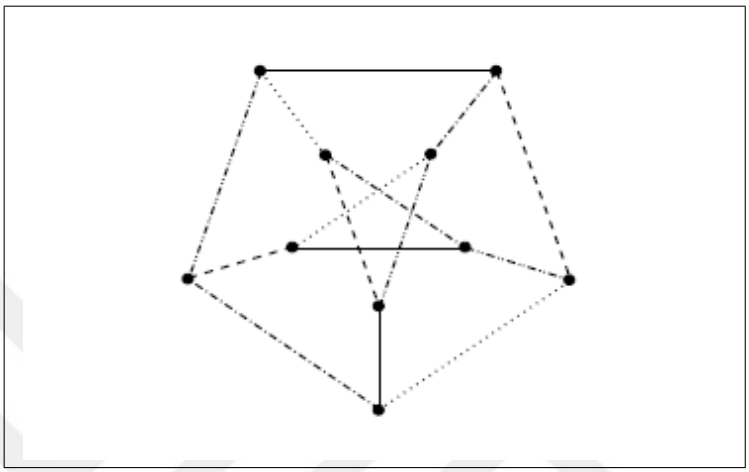

Figure 1.9. A strong edge coloring of Petersen graph with five colors

A graph is called weakly chordal if neither the graph itself nor the complement of the graph has an induced cycle on five or more vertices. Cameron *et al.* [6] proposed a polynomial time algorithm to find a maximum induced matching and minimum strong edge coloring in weakly chordal graphs. Their result also shows that the maximum induced matching problem can be solved in polynomial time for chordal bipartite graphs even though the problem is known to be NP-hard for bipartite graphs in general.

A graph is called planar if it can be drawn in such a way that no edges cross each other [7]. Kang, Mnich and Müller [21] presented an algorithm that, given as input a planar graph with m edges and maximum degree 3, finds an induced matching of size at least $m/9$ in linear-time. Their study support a conjecture of Faudree *et al.* [23] that every planar graph of maximum degree 3 is strongly edge 9-colorable.

In the literature, various approximation algorithms for MIM in general graphs have been proposed. As the optimization version of MIM is NP-hard, one way to cope with the NP-completeness of the problem is to relax the optimality condition and search for the existence of any polynomial time algorithm returning a solution

whose cardinality is close to the maximum number of induced matchings in the graph. A maximization problem P is said to be approximable with (performance) ratio ρ if there is a polynomial time algorithm which guarantees to find a solution whose size is at least $1/\rho$ times the size of an optimal solution [24].

The first study on the approximability of MIM is done by Zito [25]. He studied the complexity of MIM in regular graphs and trees. He proved that the maximum induced matching in a regular graph with degree d can be approximated with a performance of $d - 1/2$. He also showed that for every $k \geq 1$ there is a constant $c \geq 1$ such that approximating maximum induced matching within a factor c on $4k$-regular graphs is NP-hard, so maximum induced matching is APX-complete for $4k$-regular graphs.

Duckworth *et al.* [26] proved that there is some fixed constant c such that for $3s$-regular graphs with $s \geq 1$, the problem of approximating a maximum induced matching within a factor of c is NP-hard. They improved the approximation bound obtained by Zito [25] and gave an approximation algorithm which has asymptotic performance ratio $d-1$ for the problem in d-regular graphs with $d \geq 3$. Gotthilf and Lewenstein [27] presented another greedy algorithm which achieves $0.75d + 0.15$ approximation factor for the problem.

Furthermore, the parametrized complexity of MIM is studied for some restricted graph classes. In the field of parameterized complexity analysis, the complexity of a problem is analyzed in a two-dimensional framework: the input size n, and a parameter k. A parameterized problem is called fixed-parameter tractable if it can be solved in $f(k).n^{O(1)}$ time, where f is a function of the parameter k and does not depend on the input size n. W[1] is the basic complexity class for fixed-parameter intractability since there is good reason to believe that W[1]-hard problems are not fixed parameter tractable [28]. In terms of the parameterized complexity of the induced matching problem on general graphs, it is known that the problem is W[1]-hard. Hence, according to the parameterized complexity hypothesis, it is unlikely that MIM can be solved in time $O(f(k).n^c)$ for some constant c independent of k [29].

Kanj *et al.* [30] examined lower and upper bounds on the size of induced matchings for some graph classes like planar graphs, outerplanar graphs and graphs of bounded genus. They proved that for planar twinless graphs the maximum size of an induced matching is at least $n/40$ and this bound cannot be improved beyond $(n + 10)/27$. They improved the results of Moser and Sikdar [29] and showed that the induced matching problem on planar graphs has a kernel of size at most $40k$, which can be computed in linear time. They also showed that the decision version of the problem, that is whether a planar graph contains an induced matching of size at least k can be decided in $91^k + n$ time.

In this thesis, we first give a vertex-based binary integer programming formulation for MIM, in which the number of decision variables and constraints depend on the number of vertices in the graph ($O(|V|)$). Then, we extend the problem for graphs in which each vertex has a weight and the objective is to maximize the sum of weights of saturated vertices in an induced matching. The resulting problem is called Maximum Vertex-Weighted Induced Matching problem (MVWIM). Similarly, in Maximum Edge-Weighted Induced Matching problem (MEWIM), we assume that the weights are on the edges of the graph and the total edge weight in an induced matching is to be maximized. We adapt formulations found in the literature and our formulation for MIM to solve MVWIM and MEWIM.

In the generalized Maximum Weighted Induced Matching problem (MWIM), we consider graphs having both edge and vertex weights. The aim is to maximize the sum of weights of selected edges and saturated vertices in the matching induced by the selected edges. Our integer programming formulation for MWIM contains many decision variables and constraints. Instead of considering all decision variables and constraints simultaneously, we develop a Benders decomposition approach to find an optimal solution of the problem. By this way, we partition the problem into a master problem containing only a subset of the variables and a subproblem containing continuous variables. In each iteration, a master problem is solved and based on these values, the remaining values are determined by a subproblem. Using LP duality theory, new cuts are generated and added to the master problem to reach optimality.

The remaining of this thesis is organized as follows. In Section 2, we give formulations found in the literature and our formulation for MIM. Then, we adapt these formulations to solve MVWIM and MEWIM. Section 3 describes our decomposition approach used to solve MWIM. In Section 4, we explain some methods developed to increase the efficiency of our decomposition procedure. We present our computational results and some key observations about the methods we used in Section 5. Finally, we summarize our study in Section 6.

2. PROBLEM FORMULATIONS

2.1. Maximum Induced Matching Problem (MIM)

Vassilaras and Christou [2] introduced a binary integer programming formulation for MIM. Although they formulated the problem for unit disk graphs, their formulation can find a maximum induced matching for all graphs.

Let $G = (V, E)$ be an undirected graph with vertex set V and edge set E. In Vassilaras and Christou's [2] formulation, each edge $(i, j) \in E$ is represented by a binary decision variable y_{ij}, which takes value 1 if edge $(i, j) \in E$ is selected in an optimal MIM, and 0 otherwise. Let $N_{ij} \subseteq E$ be the set of edges that are adjacent to edge (i, j). Then, their formulation for MIM is as follows:

$$\textbf{(VC2011):} \quad max \quad \sum_{(i,j)\in E} y_{ij} \tag{2.1a}$$

$$s.t. \quad y_{ij} + \sum_{(k,l)\in N_{ij}} y_{kl} \leq 1 \quad \forall\, (i,j) \in E \tag{2.1b}$$

$$y_{ij} \in \{0, 1\} \quad \forall\, (i,j) \in E \tag{2.1c}$$

The objective function (2.1a) maximizes the number of selected edges. Constraints (2.1b) enforce the condition that if an edge $(i, j) \in E$ is selected ($y_{ij} = 1$), none of its adjacent edges can be selected (all $y_{kl} = 0$ for $(k, l) \in N_{ij}$), and if it is not selected ($y_{ij} = 0$), at most one of its adjacent edges can be selected. These constraints guarantee that the selection is an induced matching in G.

The above formulation can be equivalently transformed into a maximum 2-packing problem on $G^* = L(G)$. Here, a binary decision variable x_i takes value 1 if the corresponding vertex $i \in G^*$ is saturated, and 0 otherwise. Then the resulting formulation

is as follows [1]:

$$max \quad \sum_{i \in V(G^*)} x_i \tag{2.2a}$$

$$s.t. \quad x_i + \sum_{j \in N(i)} x_j \leq 1 \quad \forall\, i \in V(G^*) \tag{2.2b}$$

$$x_i \in \{0,1\} \quad \forall\, i \in V(G^*) \tag{2.2c}$$

Note that as the vertices in $G^* = L(G)$ corresponds to the edges in G, the number of binary variables and constraints in (2.2) is the same as (2.1), which depends on the number of edges in the original graph G. In order to solve the problem, Vassilaras and Christou [2] first generate the line graph $L(G)$ of G where the edges in G become vertices in $L(G)$ and the edges in $L(G)$ represent adjacent edges in G. Then, they add some valid inequalities to the model using the fact that a unit disk graph cannot contain an induced subgraph isomorphic to $K_{1,6}$, a tree with one internal vertex and 6 leaves. Since enumerating all these inequalities can take exponential time and space, they use a fixed threshold for the number of valid inequalities and solve the resulting integer programming formulation to optimality.

Christou and Vassilaras [1] propose a heuristic algorithm to quickly find near-optimal solutions on disk graphs. To test the performance of their algorithm, they give another integer programing formulation. In their formulation (2.3), each edge is also represented by a binary variable y_{ij}, but it has more constraints. Their second formulation is given below:

(CV2013): $$max \quad \sum_{(i,j) \in E(G)} y_{ij} \tag{2.3a}$$

$$s.t. \quad y_{ij} + y_{jk} \leq 1 \qquad \forall\, (i,j),(j,k) \in E(G) \tag{2.3b}$$

$$y_{ij} + y_{kl} \leq 1 \qquad \forall\, (i,j),(k,l) \in E(G) : d_e((i,j),(k,l)) = 2 \tag{2.3c}$$

$$y_{ij} \in \{0,1\} \quad \forall\, (i,j) \in E(G) \tag{2.3d}$$

where $d_e((i,j),(k,l))$ represents the distance between edges (i,j) and (k,l).

Here, the objective function (2.3a) again maximizes the number of selected edges. Constraints (2.3b) ensure that for any adjacent edge pairs, at most one of them can be selected. Similarly, constraints (2.3c) ensure that for any edge pair having distance of 2, at most one of them can be in an optimal induced matching.

The number of binary variables in these three formulations (2.1), (2.2) and (2.3) is proportional to the number of edges in the graph, which is $O(|V|^2)$ for dense graphs. Therefore, these formulations do not provide efficient solutions for graphs with a large number of edges.

Our key observation is that, instead of formulating MIM based on edges, one can focus on the vertices and decide on which vertices will be saturated in an optimal induced matching. Based on the saturated vertices, the selected edges in the induced matching can be easily obtained.

In our vertex-based formulation, a binary variable x_i takes value 1 if the corresponding vertex $i \in V$ is saturated by induced matching, and 0 otherwise. For a vertex $i \in V$, the neighborhood of i is denoted as $N(i)$. The objective is to maximize the number of saturated vertices (divided by 2 to obtain the same result as in the previous models). In order to have an induced matching, if a vertex $i \in V$ is saturated ($x_i = 1$), exactly one of its adjacent vertices must be saturated. Otherwise ($x_i = 0$), there is no restriction on the selection of its neighbors, so we can saturate at most $|N(i)|$ of them. These constraints are represented as the following binary integer programming formulation:

$$\textbf{(MIM):} \quad max \quad \sum_{i \in V(G)} x_i/2 \tag{2.4a}$$

$$s.t. \quad x_i \leq \sum_{j \in N(i)} x_j \quad \forall\, i \in V(G) \tag{2.4b}$$

$$\sum_{j \in N(i)} x_j \leq (|N(i)| - 1)(1 - x_i) + 1 \quad \forall\, i \in V(G) \tag{2.4c}$$

$$x_i \in \{0,1\} \quad \forall\, i \in V(G) \tag{2.4d}$$

Here, the number of binary variables and the number of constraints depend on the number of vertices in the graph ($O(|V|)$). Hence, they are not affected by the density of a graph and linearly increase as the number of vertices increases.

2.2. Maximum Vertex-Weighted Induced Matching Problem (MVWIM)

In this section, we extend previous formulations for MIM to solve maximum vertex-weighted induced matching problem (MVWIM). Recall that in MVWIM, we assume that each vertex $i \in V$ has a weight c_i and the objective is to maximize the total weights on vertices saturated by an induced matching. Then, MIM is a special case of MVWIM where $c_i = 1/2$ for all $i \in V$. So, our vertex-based model (2.4) can be reformulated to solve MVWIM by only changing the objective function (2.4a) with the following:

$$max \qquad \sum_{i \in V(G)} c_i x_i \tag{2.5}$$

In Vassilaras and Christou's formulations (2.1) and (2.3), there is no decision variable representing the saturated vertices. To reformulate their models to solve MVWIM, we can set $w_{ij} = c_i + c_j$ for all $(i,j) \in E$, where w_{ij}'s represent edge weights on the graph. With this transformation, an instance of MVWIM becomes an instance of MEWIM, and an optimal solution can be obtained using the models given in the following section.

2.3. Maximum Edge-Weighted Induced Matching Problem (MEWIM)

In maximum edge-weighted induced matching problem (MEWIM), we assume that each edge $(i,j) \in E$ has a weight w_{ij} and the total edge weights in an induced

matching is maximized. Therefore, MIM is a special case of MWIM where $w_{ij} = 1$ for all $(i, j) \in E$. Since Vassilaras and Christou [1, 2] use a decision variable to represent an edge, we can reformulate their models (2.1) and (2.3) by slightly changing the objective function. We need to change the objective functions (2.1a) and (2.3a) with (2.6). The rest of the constraints (2.1b) - (2.1c) and (2.3b) - (2.3d) remain the same.

$$max \quad \sum_{(i,j)\in E(G)} w_{ij} y_{ij} \tag{2.6}$$

In our formulation (2.4), there is no decision variable representing the selected edges. To reformulate our model to solve MEWIM instances, we need to define a new binary decision variable y_{ij} which takes value 1 if edge $(i, j) \in E$ is selected in the optimal solution and 0 otherwise. New constraints are added to the previous formulation to link new y_{ij} variables to original x_i variables. The resulting formulation is as follows:

$$\textbf{(MEWIM):} \quad max \quad \sum_{(i,j)\in E(G)} w_{ij} y_{ij} \tag{2.7a}$$

$$s.t. \quad x_i \leq \sum_{j\in N(i)} x_j \quad \forall\, i \in V(G) \tag{2.7b}$$

$$\sum_{j\in N(i)} x_j \leq (|N(i)| - 1)(1 - x_i) + 1 \quad \forall\, i \in V(G) \tag{2.7c}$$

$$y_{ij} \leq x_i \quad \forall\, (i, j) \in E(G) \tag{2.7d}$$

$$y_{ij} \leq x_j \quad \forall\, (i, j) \in E(G) \tag{2.7e}$$

$$y_{ij} \geq x_i + x_j - 1 \quad \forall\, (i, j) \in E(G) \tag{2.7f}$$

$$x_i \in \{0, 1\} \quad \forall\, i \in V(G) \tag{2.7g}$$

$$y_{ij} \in \{0, 1\} \quad \forall\, (i, j) \in E(G) \tag{2.7h}$$

The objective function (2.7a) maximizes the sum of total edge weights corresponding to the selected edges. Constraints (2.7b) and (2.7c), as in the previous formulation for MIM, guarantee that the selection is an induced matching in G. An edge $(i, j) \in E$

is selected in an optimal solution ($y_{ij} = 1$) only if both of its end vertices are saturated by the matching ($x_i = x_j = 1$). This is ensured by constraints (2.7d), (2.7e) and (2.7f). Note that this formulation is valid even when we have some negative w_{ij} values. Also note that if at least one of $x_i = 0$ or $x_j = 0$, edge (i, j) cannot be selected ($y_{ij} = 0$) because of constraints (2.7d) and (2.7e). Also, constraints (2.7f) forces that if both $x_i = x_j = 1$, edge (i, j) is selected ($y_{ij} = 1$) in the matching. Thus, y-variables always take on binary values even though they are defined as continuous and we can relax them as continuous in (2.7h).

3. BENDERS DECOMPOSITION FOR GENERALIZED MWIM

MEWIM formulations given in Section 3.2 can be extended to also handle vertex weighted version of the problem. In the generalized Maximum Weighted Induced Matching problem (MWIM), we consider graphs having both edge and vertex weights.

In addition to edge weights (w_{ij}), assume each vertex $i \in V$ has a weight c_i, and the sum of weights on selected edges and saturated vertices is to be maximized. Then, we can generalize Vassilaras and Christou's formulation (2.1) for MWIM as:

$$\textbf{(VC2011 MWIM):} \quad max \quad \sum_{(i,j)\in E(G)} (w_{ij} + c_i + c_j)\, y_{ij} \tag{3.1a}$$

$$s.t. \quad y_{ij} + \sum_{(k,l)\in N_{ij}} y_{kl} \leq 1 \quad \forall\, (i,j) \in E(G) \tag{3.1b}$$

$$y_{ij} \in \{0,1\} \quad \forall\, (i,j) \in E(G) \tag{3.1c}$$

Also, our generalized version of MWIM formulation is obtained by replacing the objective function in (2.7). The resulting formulation is as follows:

$$\textbf{(MWIM):} \quad max \quad \sum_{(i,j)\in E(G)} w_{ij} y_{ij} + \sum_{i\in V(G)} c_i x_i \tag{3.2a}$$

$$s.t. \quad x_i \leq \sum_{j\in N(i)} x_j \quad \forall\, i \in V(G) \tag{3.2b}$$

$$\sum_{j\in N(i)} x_j \leq (|N(i)| - 1)(1 - x_i) + 1 \quad \forall\, i \in V(G) \tag{3.2c}$$

$$y_{ij} \leq x_i \quad \forall\, (i,j) \in E(G) \tag{3.2d}$$

$$y_{ij} \leq x_j \quad \forall\, (i,j) \in E(G) \tag{3.2e}$$

$$y_{ij} \geq x_i + x_j - 1 \quad \forall\, (i,j) \in E(G) \tag{3.2f}$$

$$x_i \in \{0,1\} \quad \forall\, i \in V(G) \tag{3.2g}$$

$$y_{ij} \in \{0,1\} \quad \forall\, (i,j) \in E(G) \tag{3.2h}$$

Note that our model contains $|V| + |E|$ binary decision variables. Even if the y-variables can be relaxed as continuous, the number of constraints is $O(|V| + |E|)$. It may be computationally difficult for large graphs to solve the problem with all decision variables and constraints simultaneously. In this section, we focus on deriving a Benders decomposition algorithm for solving the generalized MWIM.

Our decomposition approach first finds a feasible induced matching using a master problem that contains only x-variables and then checks whether this induced matching provides an optimal selection of edges using a subproblem. Let us first reformulate the problem in terms of only x-variables and an additional continuous variable t which predicts the maximum edge weight that can be obtained with selection of x-variables:

$$\textbf{(MP):} \quad max \quad \sum_{i \in V(G)} c_i x_i + t \tag{3.3a}$$

$$s.t. \quad x_i \leq \sum_{j \in N(i)} x_j \quad \forall\, i \in V(G) \tag{3.3b}$$

$$\sum_{j \in N(i)} x_j \leq (|N(i)| - 1)(1 - x_i) + 1 \quad \forall\, i \in V(G) \tag{3.3c}$$

$$t \leq UB \tag{3.3d}$$

$$x_i \in \{0,1\} \quad \forall\, i \in V(G) \tag{3.3e}$$

where UB is an upper bound on the weight of any maximum induced matching. This formulation is similar to binary integer programming formulation for MVWIM, with an additional constraint (3.3d). It contains significantly fewer decision variables and constraints than the original model (3.2), which is advantageous in terms of computational effort.

In the formulation of master problem (MP), constraints (3.3b) and (3.3c) provide a feasible induced matching. Since the decision variable t predicts the maximum total

edge weight in the matching, an initial UB can be calculated by summing up all edge weights. Tighter values for UB and some valid inequalities for t-variable are given in Section 4.1. Also, in each iteration, additional cuts are generated and added to the master problem using a subproblem to reach optimality.

After solving this master problem and finding a feasible vertex selection, denoted by $\hat{x}$, the corresponding edge selection represented by y-variables and the total edge weights on these edges can be obtained using the following subproblem ($SP(\hat{x})$):

$$\textbf{(SP}(\hat{\boldsymbol{x}})\textbf{)}: \quad max \quad \sum_{(i,j)\in E(G)} w_{ij}y_{ij} \tag{3.4a}$$

$$s.t \quad y_{ij} \leq \hat{x}_i \quad \forall\,(i,j) \in E(G) \tag{3.4b}$$

$$y_{ij} \leq \hat{x}_j \quad \forall\,(i,j) \in E(G) \tag{3.4c}$$

$$y_{ij} \geq \hat{x}_i + \hat{x}_j - 1 \quad \forall\,(i,j) \in E(G) \tag{3.4d}$$

$$y_{ij} \in \{0,1\} \quad \forall\,(i,j) \in E(G) \tag{3.4e}$$

In this formulation, y-variables can be relaxed as continuous as explained in Section 2.3. Then the subproblem becomes a linear programming problem. Furthermore, we do not need to solve it as an LP since the solution can be obtained trivially by inspection for any given $\hat{x}$ values. Note that $SP(\hat{x})$ is always feasible and the optimal solution is bounded. Hence, the dual of $SP(\hat{x})$ is always feasible and bounded. Let α_{ij}, β_{ij} and γ_{ij} be dual multipliers associated with the constraints (3.4b), (3.4c) and (3.4d) in $SP(\hat{x})$, respectively. Then the dual formulation of the subproblem for a given $\hat{x}$ is given below:

$$\textbf{(DSP}(\hat{\boldsymbol{x}})\textbf{)}: \quad min \quad \sum_{(i,j)\in E(G)} (\alpha_{ij}\hat{x}_i + \beta_{ij}\hat{x}_j + \gamma_{ij}(\hat{x}_i + \hat{x}_j - 1)) \tag{3.5a}$$

$$s.t \quad \alpha_{ij} + \beta_{ij} + \gamma_{ij} \geq w_{ij} \quad \forall\,(i,j) \in E(G) \tag{3.5b}$$

$$\alpha_{ij} \geq 0 \quad \forall\,(i,j) \in E(G) \tag{3.5c}$$

$$\beta_{ij} \geq 0 \quad \forall\,(i,j) \in E(G) \tag{3.5d}$$

$$\gamma_{ij} \leq 0 \quad \forall\, (i,j) \in E(G) \tag{3.5e}$$

It can be seen that the feasible region of $DSP(\hat{x})$ does not depend on the value of $\hat{x}$, it only affects the objective function. Also, as we mentioned before, $DSP(\hat{x})$ is always feasible and bounded.

Although $DSP(\hat{x})$ is a linear programming problem, its optimal solution can be obtained without solving it as an LP. Note that the formulation can be decomposed for each edge $(i,j) \in E$ such that the optimum objective value of $DSP(\hat{x})$ is the summation of the optimum objective values of decomposed problems. To find an optimal solution for $DSP(\hat{x})$, we can use the following procedure:

If both $\hat{x}_i = \hat{x}_j = 0$, we must set $\gamma_{ij} = 0$ to minimize the objective function. In addition, we need to satisfy the condition that $\alpha_{ij} + \beta_{ij} \geq w_{ij}$. If only one of $\hat{x}_i = 1$ or $\hat{x}_j = 1$, the corresponding dual variable must take a value of 0, and we need to satisfy the condition that $\beta_{ij} + \gamma_{ij} \geq w_{ij}$ or $\alpha_{ij} + \gamma_{ij} \geq w_{ij}$, respectively. If both $\hat{x}_i = \hat{x}_j = 1$, we need to satisfy the condition that $\alpha_{ij} + \beta_{ij} + \gamma_{ij} = w_{ij}$. Thus, we can obtain infinitely many (α, β, γ) solutions for $DSP(\hat{x})$. Let θ_{ij} represents the proportion of w_{ij} that α_{ij} will take. Then, an optimal solution for $DSP(\hat{x})$ can be found using Algorithm 1 for a given $\hat{x}$:

Algorithm 1 Solution of $DSP(\hat{x})$

Require: A graph $G = (V, E)$ and a binary vector $\hat{x}$ of size $|V|$

Ensure: A solution of $DSP(\hat{x})$

1: For each edge $(i, j) \in E$,
2: **if** $\hat{x}_i = 1$ and $\hat{x}_j = 1$, or $\hat{x}_i = 0$ and $\hat{x}_j = 0$ **then**
3: set $\alpha_{ij} = \theta_{ij} w_{ij}$, $\beta_{ij} = (1 - \theta_{ij}) w_{ij}$ and $\gamma_{ij} = 0$
4: **else if** $\hat{x}_i = 1$ and $\hat{x}_j = 0$ **then**
5: set $\alpha_{ij} = 0$, $\beta_{ij} = w_{ij}$ and $\gamma_{ij} = 0$
6: **else if** $\hat{x}_i = 0$ and $\hat{x}_j = 1$ **then**
7: set $\alpha_{ij} = w_{ij}$, $\beta_{ij} = 0$ and $\gamma_{ij} = 0$
8: **end if**

Our Benders decomposition strategy first solves master problem to optimality, yielding a feasible $(\hat{x}, \hat{t})$. Then, we need to find the optimum objective function value of $SP(\hat{x})$ and compare it with $\hat{t}$. Since by duality theorem, the optimum objective function values of $SP(\hat{x})$ and $DSP(\hat{x})$ are the same, we solve $DSP(\hat{x})$ using Algorithm 1 and calculate the total edge weight obtained by the current selection, denoted by t^*. If $\hat{t} = t^*$, then $\hat{x}$ corresponds to an optimal induced matching. On the other hand, if $\hat{t} > t^*$, we need to put an additional bound on t-variable in the master problem. Let $(\hat{\alpha}, \hat{\beta}, \hat{\gamma})$ be an optimal dual multipliers obtained by solving $DSP(\hat{x})$. Then, the following constraint is added to the master problem and re-solved in the next iteration to obtain a new candidate optimal solution:

$$t \leq \sum_{(i,j) \in E(G)} (\hat{\alpha}_{ij} x_i + \hat{\beta}_{ij} x_j + \hat{\gamma}_{ij} (x_i + x_j - 1)) \tag{3.6}$$

This type of constraints are called "Benders optimality cuts" as they are based on the optimality conditions of the subproblem. Since $DSP(\hat{x})$ can have infinitely many solutions, the selection of θ_{ij} parameters results in different cuts. Different θ_{ij} selection procedures have been applied to compare computational results in Section 5.

Remark. *For any optimality cut, the violation is obtained by*

$$t - \sum_{(i,j)\in E(G)} (\hat{\alpha}_{ij}\hat{x}_i + \hat{\beta}_{ij}\hat{x}_j + \hat{\gamma}_{ij}(\hat{x}_i + \hat{x}_j - 1)) = t - t^* \tag{3.7}$$

where t^* is the current optimum objective function value for $DSP(\hat{x})$. Thus, all optimality cuts generated in this way have the same violation regardless of the value of $(\hat{\alpha}, \hat{\beta}, \hat{\gamma})$.

In master problem, instead of adding a single decision variable t, we can create a decision variable t_i for each vertex $i \in V$. These variables are used to predict the maximum edge weight that can be obtained if we saturate vertex i. To incorporate this change in our master problem, we simply replace the objective function (3.3a) and constraint (3.3d) with the following objective function (3.8) and constraints (3.9), respectively:

$$max \qquad \sum_{i\in V(G)} (t_i \ / \ 2 \ + \ c_i x_i) \tag{3.8}$$

$$t_i \leq UB_i \tag{3.9}$$

where UB_i is an upper bound on the value of t_i. A valid numerical value for UB_i can be the maximum edge weight emanating from vertex i. Another valid inequality for t_i-variables is given is Section 4.1.

In this case, Benders decomposition procedure given above needs to be adjusted accordingly. Optimality cut (3.6) can also be decomposed such that each time we add a new cut for every t_i :

$$t_i \leq \sum_{j\in N(i)} (\hat{\alpha}_{ij}x_i + \hat{\beta}_{ij}x_j + \hat{\gamma}_{ij}(x_i + x_j - 1)) \tag{3.10}$$

The above solution procedure for MWIM using Benders decomposition is sum-

marized in Algorithm 2:

Algorithm 2 MWIM Benders Decomposition

Require: A graph $G = (V, E)$ with edge weights w_{ij} and vertex weights c_i

Ensure: A maximum weighted induced matching

1: Set $LB = -\infty$ and $UB = \infty$

2: Solve MP. Let $(\hat{x}, \hat{t})$ be a candidate optimal solution. Set $UB = \hat{t} + \sum_{i \in V(G)} c_i \hat{x}_i$ (or set $UB = \sum_{i \in V(G)} (\hat{t}_i \ / \ 2 \ + \ c_i \hat{x}_i)$)

3: Obtain sum of weights of selected edges by $DSP(\hat{x})$ using Algorithm 1, denoted by t^*. Set $LB = t^* + \sum_{i \in V(G)} c_i \hat{x}_i$.

4: **if** $UB = LB$ **then**

5: Current edge selection is a maximum weighted induced matching of G, STOP

6: **else**

7: Let $(\hat{\alpha}, \hat{\beta}, \hat{\gamma})$ denote an optimal solution of $DSP(\hat{x})$.

8: Generate optimality cut(s) using (3.6) (or (3.10)) with current optimal dual solution $(\hat{\alpha}, \hat{\beta}, \hat{\gamma})$, add them to the master problem. Go to step 2.

9: **end if**

4. ALGORITHMIC IMPROVEMENTS

4.1. Valid Inequalities

In the master problem formulation (3.3), before adding any optimality cuts, the only bound for t-variable is the sum of all edge weights, denoted by UB. Since this is a maximization problem, in the first iteration, t-variable will be at its upper bound. By tightening its upper bound and adding valid inequalities on t-variable, we can have smaller upper bound for the objective function value at the first iterations of Algorithm 2, which can result in faster convergence.

To obtain a tighter upper bound for t-variable, we will use the following observation: for a graph $G = (V, E)$, a matching can contain at most $\lfloor |V|/2 \rfloor$ edges. This is also valid for any induced matching since an induced matching is also a matching in G. Based on this observation, we can find an upper bound on the value of t-variable. Since it is used to predict the maximum edge weight of an induced matching in G, a numerical bound, denoted by UB^*, can be found by sorting all w_{ij} values and summing up the greatest $\lfloor |V|/2 \rfloor$ of them. Therefore an initial bound on t-variable can be written as:

$$t \leq UB^* \tag{4.1}$$

Another approach to find a valid inequality for t-variable is as follows: for any vertex $i \in V$, if $x_i = 1$, i.e. if it is saturated by an induced matching, it can at most increase the objective function value by the maximum edge weight emanating from it and if $x_i = 0$, i.e. if it is not saturated by an induced matching, there is no increase in the objective function value. Thus, the following inequality is valid for t:

$$t \leq \sum_{i \in V(G)} \max_{j \in N(i)} \{w_{ij}\} \; x_i \; / \; 2 \tag{4.2}$$

Similarly, in the decomposed version of master problem, the only bound for t_i-variable is UB_i, which is the maximum edge weight emanating from vertex i. In the formulation, t_i-variables represent the maximum edge weight that can be obtained by saturating vertex i. With a similar argument, we can say that the following inequality is valid for t_i:

$$t_i \leq \max_{j \in N(i)} \{w_{ij}\}\ x_i \tag{4.3}$$

In Section 5, we add these upper bounds and valid inequalities into master problem formulation and compare their effects on the performance of our decomposition approach.

4.2. Formulation Tightening

In our formulation (2.4) for MIM, constraints (2.4c) are based on the observation that if a vertex $i \in V$ is not saturated ($x_i = 0$), we can saturate at most $|N(i)|$ of its neighbors. However, it may be infeasible to saturate all of its neighbors in an induced matching. To obtain a tighter upper bound for these constraints, we will use the following method:

For any vertex $i \in V$, let $S = N(i)$ and $S' = N(S) - \{i\}$. Assume we assign weights to edges such that all edges $\{(i,j) : i \in S, j \in S\}$ will have weight 2, $\{(i,j) : i \in S, j \in S'\}$ will have weight 1 and other edges will have weight 0. Figure 4.1 shows assignment of edge weights for vertex i. Then, finding the maximum number of vertices in S saturated by a matching is equivalent to finding the maximum weighted matching in G, which can be found by weighted version of Edmond's blossom algorithm [12]. If for every vertex $i \in V$, we find maximum weighted matching in such a way and denote the sum of edge weights on these edges by N^*, we can replace $|N(i)|$ in constraints

(2.4c) with N^* and rewrite them as:

$$\sum_{j \in N(i)} x_j \leq (N^* - 1)(1 - x_i) + 1 \quad \forall\, i \in V(G) \tag{4.4}$$

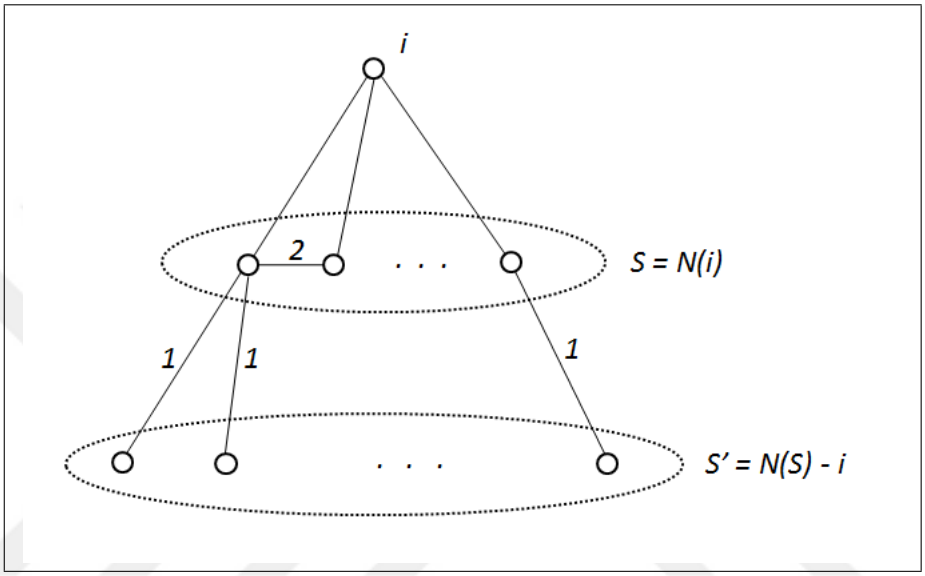

Figure 4.1. An example of construction of maximum weighted induced matching problem for vertex i

In section 5, we try to replace constraints (2.4c) with (4.4) to see the effect on the computational time of MIM formulation (2.4).

4.3. Single Branch-and-Bound Tree

In our Benders decomposition approach, we solve the master problem to optimality at each iteration and check whether it is an optimal solution or not using a subproblem. If not, we add an optimality cut and re-solve it. Although new cuts may change the structure of the branch-and-bound tree, we may need to revisit candidate solutions that are discarded earlier. Hence, this process can be very expensive from a computational point of view. Instead, we can interrupt the branch-and-bound solution process of master problem each time the solver finds an integer solution $\hat{x}$ (and $\hat{t}$) and check whether optimality cut (3.6) (or (3.10)) that is violated by the current integer

solution can be generated. If we can generate such an optimality cut, we reject the current solution, add the newly generated cuts to the problem and resume the solution process. Otherwise, we accept the current solution as the new incumbent and again resume the solution process. In our computational tests, this approach consistently outperformed solving master problem to optimality at each iteration and re-optimizing it. With this approach, we can solve the problem using a single branch-and-bound tree that is tightened as necessary as opposed to repeatedly generating a branch-and-bound tree in each iteration. Therefore, we avoid considerable rework by never visiting a node and overlooking a truly superior solution. A similar approach was also used in [31, 32]

5. COMPUTATIONAL RESULTS

To test the efficiency of formulations and improvements mentioned in previous sections, we conducted a series of experiments. We executed all integer programming formulations using CPLEX 12.6.1 running on a Windows 7 PC with a 2.3 GHz Intel Core i5 CPU and 4 GB RAM. We also used CPLEX's callback functions to solve our model in a single branch-and-bound tree as described in Section 4.3. We used LEMON Graph Library 1.2.4 [33] to efficiently implement graph related data structures and algorithms. Our base test data set contains randomly generated graph instances having expected edge density (measured as $D = \frac{2|E|}{|V|X(|V|-1)}$) of 0.2, 0.5 and 0.8. We increased the number of vertices $|V|$ up to a level that all methods are unable to solve the problem. To generate weighted instances, we first generated random graphs as in the unweighted case, and then assigned an integer weight uniformly distributed between 1 and 10 to each vertex and/or edge. We generated five problem instances for each problem size, determined by the expected edge density and the number of vertices.

For each problem size, we report the following statistics calculated over five random instances:

- "Solved:" the number of problem instances solved to optimality within the allowed time limit of 1800 seconds.
- "Gap:" the average final percentage optimality gap for all instances (calculated as $(UB-LB)/LB$ where UB denotes the upper bound and LB denotes the lower bound).
- "Time:" the average amount of time in seconds spent by each algorithm on all instances.

In Table 5.1, we compare the performances of Vassilaras and Christou's formulations VC2011, CV2013 with our vertex-based formulation MIM given in Section 2.4. In the table, "-" shows that the corresponding method is unable to solve instances in the given time limit and "*" is used if the corresponding method is unable to solve

instances due to insufficient memory. First of all, we observe that for any D and $|V|$ values, Vassilaras and Christou's first formulation (denoted by VC2011) [2] performs significantly better compared to their second formulation (denoted by CV2013) [1]. It solves more instances in the given time limit and does not cause out of memory status. Also, for low-density, VC2011 has smaller total running time than our formulation. On the other hand, for moderate and high densities, our formulation performs faster and solves more instances in the given time limit.

In Table 5.2, we show the effect of formulation tightening approach described in Section 4.2 for MIM formulation. Here, time column in the modified formulation also includes the required time to calculate the maximum weighted matchings in the graph. It can be seen that the modification suggested in Section 4.2 does not provide a significant difference in total running time and optimality gap for MIM. It follows that the use of maximum weighted matching for formulation tightening is not justified by our experiments.

Recall that in our Benders decomposition approach proposed in Section 3, $DSP(\hat{x})$ can have infinitely many solutions. By changing θ_{ij} values, we can obtain different $(\hat{\alpha}, \hat{\beta}, \hat{\gamma})$, resulting in different cuts in (3.6) and (3.10). To test the effect of θ_{ij} selection, we solved all instances by setting all θ_{ij} values to 0.2, 0.5 and 0.8. Also, we tried to assign random values between 0 and 1 to θ_{ij} in each iteration. The result of this experiment is shown in Table 5.3. Note that regardless of θ_{ij} values, all possible optimality cuts have the same violation, therefore there is no significant difference between these options. Computational results also show that θ_{ij} selection has no clear effect on total running time and optimality gap. Therefore, we set $\theta_{ij} = 0.5$ in the remaining parts for simplicity.

Since $DSP(\hat{x})$ can have infinitely many solutions, it is also possible to add multiple cuts to the master problem in each iteration. To test the effect of this option, in each iteration, we generated 10 different cuts by assigning random values between 0 and 1 to θ_{ij} variables. In Table 5.4, "multiple cut" column shows the result of this operation. Here, we can see that adding multiple cuts to the master problem has significantly

Table 5.1. Comparison of formulations for solving MIM

		MIM			VC2011			CV2013		
D	$\|V\|$	Solved	Gap	Time	Solved	Gap	Time	Solved	Gap	Time
0.2	25	5	0%	0.1	5	0%	0.0	5	0%	0.1
	50	5	0%	1.2	5	0%	0.4	5	0%	2.0
	75	5	0%	36.8	5	0%	10.6	5	0%	75.0
	100	1	8%	1795.0	5	0%	904.8	*	*	*
	125	0	50%	1800.0	0	27%	1800.0	*	*	*
0.5	25	5	0%	0.1	5	0%	0.1	5	0%	0.3
	50	5	0%	0.9	5	0%	3.9	5	0%	25.6
	75	5	0%	8.0	5	0%	155.2	0	72%	1800.0
	100	5	0%	54.1	0	128%	1800.0	-	-	-
	125	5	0%	217.0	-	-	-	-	-	-
	150	5	0%	1163.3	-	-	-	-	-	-
	175	0	49%	1800.0	-	-	-	-	-	-
0.8	25	5	0%	0.1	5	0%	0.1	5	0%	0.3
	50	5	0%	0.3	5	0%	2.1	5	0%	12.7
	75	5	0%	1.2	5	0%	211.6	*	*	*
	100	5	0%	3.5	0	696%	1800.0	*	*	*
	125	5	0%	13.5	-	-	-	*	*	*
	150	5	0%	24.0	-	-	-	*	*	*
	175	5	0%	43.4	-	-	-	*	*	*
	200	5	0%	88.4	-	-	-	*	*	*
	225	5	0%	162.1	-	-	-	*	*	*
	250	5	0%	337.5	-	-	-	*	*	*

Table 5.2. Effect of formulation tightening method described in Section 4.2 for MIM formulation

		MIM			MIM-modified		
D	$\|V\|$	Solved	Gap	Time	Solved	Gap	Time
0.2	25	5	0%	0.1	5	0%	0.1
	50	5	0%	1.2	5	0%	1.2
	75	5	0%	36.8	5	0%	38.4
	100	1	8%	1795.0	1	8%	1788.9
	125	0	50%	1800.0	0	52%	1800.0
0.5	25	5	0%	0.1	5	0%	0.2
	50	5	0%	0.9	5	0%	1.2
	75	5	0%	8.0	5	0%	8.1
	100	5	0%	54.1	5	0%	52.3
	125	5	0%	217.0	5	0%	240.7
	150	5	0%	1163.3	5	0%	1220.3
	175	0	49%	1800.0	0	48%	1800.0
0.8	25	5	0%	0.1	5	0%	0.2
	50	5	0%	0.3	5	0%	0.6
	75	5	0%	1.2	5	0%	1.6
	100	5	0%	3.5	5	0%	4.4
	125	5	0%	13.5	5	0%	15.7
	150	5	0%	24.0	5	0%	25.5
	175	5	0%	43.4	5	0%	50.8
	200	5	0%	88.4	5	0%	103.5
	225	5	0%	162.1	5	0%	182.1
	250	5	0%	337.5	5	0%	356.9

Table 5.3. Effect of different θ_{ij} selection in the solution of $DSP(\hat{x})$

		θ_{ij}= 0.2			θ_{ij}= 0.5			θ_{ij}= 0.8			θ_{ij}= rand()		
D	$\|V\|$	Solved	Gap	Time	Solved	Gap	Time	Solved	Gap	Time	Solved	Gap	Time
0.2	25	5	0%	0.1	5	0%	0.1	5	0%	0.2	5	0%	0.1
	50	5	0%	1.4	5	0%	1.2	5	0%	1.1	5	0%	1.3
	75	5	0%	34.3	5	0%	36.8	5	0%	36.0	5	0%	37.4
	100	0	7%	1800.0	1	8%	1795.0	0	10%	1800.0	1	6%	1753.5
	125	0	54%	1800.0	0	50%	1800.0	0	50%	1800.0	0	50%	1800.0
0.5	25	5	0%	0.1	5	0%	0.1	5	0%	0.1	5	0%	0.1
	50	5	0%	1.0	5	0%	0.9	5	0%	0.8	5	0%	0.9
	75	5	0%	7.4	5	0%	8.0	5	0%	7.9	5	0%	8.4
	100	5	0%	56.6	5	0%	54.1	5	0%	55.7	5	0%	52.9
	125	5	0%	215.3	5	0%	217.0	5	0%	217.8	5	0%	220.5
	150	5	0%	1179.1	5	0%	1163.3	5	0%	1170.2	5	0%	1161.4
	175	0	44%	1800.0	0	49%	1800.0	0	51%	1800.0	0	48%	1800.0
0.8	25	5	0%	0.1	5	0%	0.1	5	0%	0.1	5	0%	0.1
	50	5	0%	0.2	5	0%	0.3	5	0%	0.3	5	0%	0.3
	75	5	0%	1.4	5	0%	1.2	5	0%	1.1	5	0%	1.4
	100	5	0%	3.4	5	0%	3.5	5	0%	3.7	5	0%	3.5
	125	5	0%	13.6	5	0%	13.5	5	0%	13.8	5	0%	13.7
	150	5	0%	23.8	5	0%	24.0	5	0%	23.8	5	0%	24.3
	175	5	0%	42.7	5	0%	43.4	5	0%	42.5	5	0%	43.2
	200	5	0%	85.3	5	0%	88.4	5	0%	89.7	5	0%	87.0
	225	5	0%	165.9	5	0%	162.1	5	0%	163.7	5	0%	161.6
	250	5	0%	336.4	5	0%	337.5	5	0%	337.4	5	0%	340.4

increased the computational time as it becomes harder to solve master problem with large number of constraints. Therefore, adding multiple cuts to the master problem in a single iteration is not justified by our experiments.

Our second experiment compares the performances of MWIM formulations given in Section 3 and shows the effect of proposed improvements suggested in Section 4.1. The results of this experiment are summarized in Tables 5.5, 5.6, 5.7 and 5.8. In Table 5.5, we observe that Vassilaras and Christou's formulation [2] modified to solve MWIM, denoted as VC2011 MWIM, can solve more instances to optimality within the given time limit with lower total running times and gaps compared to our full model. On the other hand, our full model for the generalized MWIM has higher total running times and gaps for all densities.

In Table 5.6, we measure the effect of the suggested improvements on the decomposition approach where we have single t-variable in the master problem. Here, we report the following results: "Single t:" Algorithm 2 with single t-variable in the master problem, "Single t + (4.1):" Algorithm 2 where we use (4.1) as an initial bound on t-variable in the master problem, "Single t + (4.2):" Algorithm 2 augmented with

Table 5.4. Effect of generating multiple cuts in each iteration

		Single cut			Multiple cuts		
D	$\|V\|$	Solved	Gap	Time	Solved	Gap	Time
0.2	25	5	0%	0.1	5	0%	0.2
	50	5	0%	1.2	5	0%	2.0
	75	5	0%	36.8	5	0%	59.3
	100	1	8%	1795.0	0	16%	1800.0
	125	0	50%	1800.0	0	90%	1800.0
0.5	25	5	0%	0.1	5	0%	0.1
	50	5	0%	0.9	5	0%	1.3
	75	5	0%	8.0	5	0%	12.5
	100	5	0%	54.1	5	0%	84.4
	125	5	0%	217.0	5	0%	287.6
	150	5	0%	1163.3	5	0%	1537.9
	175	0	49%	1800.0	0	78%	1800.0
0.8	25	5	0%	0.1	5	0%	0.1
	50	5	0%	0.3	5	0%	0.5
	75	5	0%	1.2	5	0%	1.9
	100	5	0%	3.5	5	0%	4.7
	125	5	0%	13.5	5	0%	19.6
	150	5	0%	24.0	5	0%	35.0
	175	5	0%	43.4	5	0%	70.3
	200	5	0%	88.4	5	0%	118.3
	225	5	0%	162.1	5	0%	202.9
	250	5	0%	337.5	5	0%	462.6

valid inequality (4.2) in the master problem. For the model with single t-variable in the master problem, it can be observed that inequality (4.1) has a slight improvement in total running time, but it is unable to solve instances that could not be solved by pure model. It can only decrease the optimality gap for these instances. On the other hand, inequality (4.2) has a significant effect on the performance in terms of computational time and we can solve more instances in the enforced time limit.

Similarly, in Table 5.7, we show the effect of valid inequality (4.3) on the decomposition approach where we have multiple t_i-variables in the master problem. We report the following results: "Multiple t:" Algorithm 2 with multiple t_i-variables in the master problem, "Multiple t + (4.3):" Algorithm 2 improved with valid inequality (4.3) in the master problem. Here, we note that valid inequality (4.3) has a significant improvement in terms of computational effort. It can be seen that the use of valid inequality (4.3) is justified by our experiments.

Finally, in Table 5.8, we summarize the best implementations for MWIM. If we compare the running times of full model, in which we have all decision variables and constraints, and our Benders decomposition approach, it is observed that our decomposition approach outperforms full model for all instances. Also, comparing two decomposition approaches, we note that the use of multiple t_i-variables in the master problem along with valid inequalities (4.3) has a significant effect on solution times and optimality gaps for all instances. By comparing its performance against Vassilaras and Christou's formulation [2] adapted for MWIM, we observe that VC2011 MWIM is faster than our decomposition approach for low density, but, our algorithm performs better for medium and large densities.

Table 5.5. Comparison of formulations for solving MWIM

		VC2011 MWIM			Full model		
D	$\|V\|$	Solved	Gap	Time	Solved	Gap	Time
0.2	25	5	0%	0.1	5	0%	0.5
	50	5	0%	0.4	5	0%	7.2
	75	5	0%	10.8	1	6%	1745.0
	100	5	0%	235.1	-	-	-
	125	0	56%	1800.0	-	-	-
0.5	25	5	0%	0.2	5	0%	0.5
	50	5	0%	6.2	5	0%	13.2
	75	5	0%	113.4	5	0%	149.8
	100	5	0%	1456.3	5	0%	1524.5
	125	0	152%	1800.0	0	860%	1800.0
	150	-	-	-	-	-	-
0.8	25	5	0%	0.2	5	0%	0.6
	50	5	0%	4.6	5	0%	12.2
	75	5	0%	55.2	5	0%	84.5
	100	5	0%	244.1	5	0%	342.3
	125	5	0%	896.7	5	0%	1211.2
	150	0	88%	1800.0	0	6360%	1800.0
	175	-	-	-	-	-	-
	200	-	-	-	-	-	-
	225	-	-	-	-	-	-
	250	-	-	-	-	-	-

Table 5.6. Comparison of formulations for solving MWIM

		Single t			Single t + (4.1)			Single t + (4.2)		
D	$\|V\|$	Solved	Gap	Time	Solved	Gap	Time	Solved	Gap	Time
0.2	25	5	0%	0.4	5	0%	0.5	5	0%	0.3
	50	3	12%	1453.7	3	7%	1247.8	5	0%	9.8
	75	0	410%	1800.0	0	200%	1800.0	3	7%	1759.6
	100	-	-	-	-	-	-	0	50%	1800.0
	125	-	-	-	-	-	-	-	-	-
0.5	25	5	0%	1.3	5	0%	1.1	5	0%	0.3
	50	5	0%	436.7	5	0%	324.2	5	0%	5.1
	75	0	707%	1800.0	0	346%	1800.0	5	0%	48.3
	100	-	-	-	-	-	-	5	0%	160.9
	125	-	-	-	-	-	-	5	0%	770.9
	150	-	-	-	-	-	-	0	33%	1800.0
0.8	25	5	0%	2.0	5	0%	1.6	5	0%	0.4
	50	5	0%	235.4	5	0%	197.3	5	0%	5.3
	75	5	0%	852.8	5	0%	824.6	5	0%	13.1
	100	0	1554%	1800.0	0	690%	1800.0	5	0%	27.4
	125	-	-	-	-	-	-	5	0%	93.7
	150	-	-	-	-	-	-	5	0%	273.1
	175	-	-	-	-	-	-	5	0%	450.1
	200	-	-	-	-	-	-	5	0%	1234.7
	225	-	-	-	-	-	-	0	86%	1800.0
	250	-	-	-	-	-	-	-	-	-

Table 5.7. Comparison of formulations for solving MWIM

		Multiple t			Multiple t + (4.3)		
D	$\|V\|$	Solved	Gap	Time	Solved	Gap	Time
0.2	25	5	0%	0.3	5	0%	0.3
	50	5	0%	26.2	5	0%	2.9
	75	0	146%	1800.0	5	0%	456.5
	100	-	-	-	0	28%	1800.0
	125	-	-	-	-	-	-
0.5	25	5	0%	0.7	5	0%	0.4
	50	5	0%	79.4	5	0%	3.2
	75	0	527%	1800.0	5	0%	39.3
	100	-	-	-	5	0%	73.0
	125	-	-	-	5	0%	562.5
	150	-	-	-	0	23%	1800.0
0.8	25	5	0%	1.5	5	0%	0.5
	50	5	0%	113.2	5	0%	3.0
	75	0	212%	1800.0	5	0%	7.7
	100	-	-	-	5	0%	13.9
	125	-	-	-	5	0%	41.6
	150	-	-	-	5	0%	75.9
	175	-	-	-	5	0%	101.5
	200	-	-	-	5	0%	240.2
	225	-	-	-	5	0%	482.7
	250	-	-	-	5	0%	621.2

Table 5.8. Comparison of formulations for solving MWIM

		VC2011 MWIM			Full model			Single t + (4.2)			Multiple t + (4.3)		
D	$\|V\|$	Solved	Gap	Time	Solved	Gap	Time	Solved	Gap	Time	Solved	Gap	Time
0.2	25	5	0%	0.1	5	0%	0.5	5	0%	0.3	5	0%	0.3
	50	5	0%	0.4	5	0%	7.2	5	0%	9.8	5	0%	2.9
	75	5	0%	10.8	1	6%	1745.0	3	7%	1759.6	5	0%	456.5
	100	5	0%	235.1	-	-	-	0	50%	1800.0	0	28%	1800.0
	125	0	56%	1800.0	-	-	-	-	-	-	-	-	-
0.5	25	5	0%	0.2	5	0%	0.5	5	0%	0.3	5	0%	0.4
	50	5	0%	6.2	5	0%	13.2	5	0%	5.1	5	0%	3.2
	75	5	0%	113.4	5	0%	149.8	5	0%	48.3	5	0%	39.3
	100	5	0%	1456.3	5	0%	1524.5	5	0%	160.9	5	0%	73.0
	125	0	152%	1800.0	0	860%	1800.0	5	0%	770.9	5	0%	562.5
	150	-	-	-	-	-	-	0	33%	1800.0	0	23%	1800.0
0.8	25	5	0%	0.2	5	0%	0.6	5	0%	0.4	5	0%	0.5
	50	5	0%	4.6	5	0%	12.2	5	0%	5.3	5	0%	3.0
	75	5	0%	55.2	5	0%	84.5	5	0%	13.1	5	0%	7.7
	100	5	0%	244.1	5	0%	342.3	5	0%	27.4	5	0%	13.9
	125	5	0%	1096.7	5	0%	1211.2	5	0%	93.7	5	0%	41.6
	150	0	88%	1800.0	0	6360%	1800.0	5	0%	273.1	5	0%	75.9
	175	-	-	-	-	-	-	5	0%	450.1	5	0%	101.5
	200	-	-	-	-	-	-	5	0%	1234.7	5	0%	240.2
	225	-	-	-	-	-	-	0	86%	1800.0	5	0%	482.7
	250	-	-	-	-	-	-	-	-	-	5	0%	621.2

6. CONCLUSION AND FURTHER RESEARCH

In the scope of this thesis, we studied Maximum Induced Matching Problem (MIM), where the aim is to find an induced matching with the largest cardinality. In practice, induced matchings are heavily used in communication industry to obtain secure communication channels. They are also used to determine the maximum capacity of MAC layer in wireless ad-hoc networks.

The problem is known to be NP-hard for general graphs, even for bipartite graphs. In the literature, the problem is studied for some restricted graph classes. It is addressed from mathematical programming point of view by Vassilaras and Christou [1, 2].

We give a vertex-based integer programming formulation for MIM, having less number of binary variables and constraints. Then, we described vertex-weighted and edge-weighted versions of the problem, and call them as Maximum Vertex-Weighted Induced Matching problem (MVWIM) and Maximum Edge-Weighted Induced Matching problem (MEWIM), respectively. We reformulated the models in the literature and our model to solve weighted instances.

In the generalized version of Maximum Weighted Induced Matching problem (MWIM), we considered graphs with both edge and wertex weights. As the formulation for MWIM contains many decision variables and constraints, we applied Benders decomposition to our proposed formulation. Our decomposition algorithm seeks a feasible induced matching using a master problem and tries to reach optimality with the help of cuts generated by the subproblem. We added upper bounds on variables and valid inequalities in the master problem to improve the efficiency of our algorithm.

We tested the efficacy of our approach on randomly generated graph instances. Computational results show that our decomposition approach performs better than solving the underlying integer programming formulation. Also, it outperforms the formulations found in the literature for medium and high densities.

As a future research, one can focus on solving MIM, MVWIM, MEWIM and MWIM problems in some specific graph classes and consider developing additional valid inequalities using their structural properties.

REFERENCES

1. Christou, I. T. and S. Vassilaras, "A parallel hybrid greedy branch and bound scheme for the maximum distance-2 matching problem", *Computers and Operations Research*, Vol. 40, pp. 2387–2397, 2013.

2. Vassilaras, S. and I. T. Christou, "On the optimal MAC layer capacity of delay tolerant mobile Ad Hoc networks with a finite number of nodes", *22nd annual IEEE international symposium on personal, indoor and mobile radiocommunications (PIMRC'11)*, Toronto, Canada, Sep 2011.

3. Gross, J. L. and Y. J., *Handbook of graph theory*, CRC Press, 2003.

4. Cameron, K., "Induced matchings", *Discrete Applied Mathematics*, Vol. 24, pp. 97–102, 1989.

5. Golumbic, M. C. and I. B.-A. Hartman, *Graph Theory, Combinatorics and Algorithms*, Springer, New York, NY, USA, 2004.

6. Cameron, K., R. Sritharan and Y. Tang, "Finding a maximum induced matching in weakly chordal graphs", *Discrete Mathematics*, Vol. 266, pp. 133–142, 2003.

7. Gross, J. L. and Y. J., *Graph Theory and Its Applications*, CRC Press, 2006.

8. Stockmeyer, L. and V. Vazirani, "NP-completeness of some generalizations of the maximum matching problem", *Information Processing Letters*, Vol. 15, p. 14–19, 1982.

9. Bondy, J. A. and M. U. S. R., *Graph Theory*, Springer, 2008.

10. Alekseev, V. E., "A polynomial algorithm for finding the largest independent sets in claw-free graphs", *Diskretn. Anal. Issled. Oper.*, Vol. 6, No. 4, p. 3–19, 1999.

11. Cameron, K., "Induced matchings in intersection graphs", *Discrete Applied Mathematics*, Vol. 278, pp. 1–9, 2004.

12. Edmonds, J., "Paths, trees, and flowers", *Canadian Journal Mathematics*, Vol. 17, p. 449–467, 1965.

13. Golumbic, M. and R. Laskar, "Irredundancy in circular arc graphs", *Discrete Applied Mathematics*, Vol. 44, pp. 79–89, 1993.

14. Balakrishnan, H., C. L. Barrett, V. S. A. Kumar, M. V. Marathe and S. Thite, "The distance-2 matching problem and its relationship to the MAC-layer capacity of ad hoc wireless networks", *IEEE Journal on Selected Areas in Communications*, Vol. 22, No. 6, 2004.

15. Golumbic, M. C. and M. Lewenstein, "New results on induced matchings", *Discrete Applied Mathematics*, Vol. 101, pp. 157–165, 2000.

16. Gavril, F., "Algorithms for minimum coloring, maximum clique, minimum covering by cliques, and maximum independent set of a chordal graph", *SIAM Journal on Computing*, 1972.

17. Fricke, G. and R. Laskar, "Strong matchings on trees", *Congr. Numer.*, Vol. 89, pp. 239–243, 1992.

18. Brandstädt, A. and C. T. Hoàng, "Maximum Induced Matchings for Chordal Graphs in Linear Time", *Algorithmica*, Vol. 52, p. 440–447, 2008.

19. Chang, J.-M., "Induced matchings in asteroidal triple-free graphs", *Discrete Applied Mathematics*, Vol. 132, p. 67 – 78, 2004.

20. Duckworthy, W., N. Wormald and M. Zito, "Maximum induced matchings of random cubic graphs", *Journal of Computational and Applied Mathematics*, Vol. 142, p. 39–50, 2002.

21. Kang, R. J., M. Mnich and M. T., "Induced matchings in subcubic planar graphs", *Discrete Mathematics*, Vol. 26, No. 3, p. 1383–1411, 2012.

22. Petersen, J., "Sur le théorème de Tait", *L'Intermédiaire des Mathématiciens*, Vol. 5, pp. 225–227, 1898.

23. Faudree, R. J., R. H. Schelp, G. A. and Z. Tuza, "The strong chromatic index of graphs", *Ars Combinatoria*, p. 205–211, 1990.

24. Vazirani, V. V., *Approximation Algorithms*, Springer, 2003.

25. Zito, M., "Induced matchings in regular graphs and trees", *Proc. of WG 1999*, pp. 89–100,, 1999.

26. Duckworthy, W., N. Wormald and M. Zito, "On the approximability of the maximum induced matching problem", *Journal of Discrete Algorithms*, Vol. 3, p. 79–91, 2005.

27. Gotthilf, Z. and L. M., "Tighter approximations for maximum induced matchings in regular graphs", *WAOA*, p. 270–281, 2005.

28. Downey, R. G. and F. M.R., *Parametrized Complexity*, Springer, 1999.

29. Moser, H. and S. Sikdar, "The parameterized complexity of the induced matching problem", *Discrete Applied Mathematics*, Vol. 157, No. 4, p. 715–727, 2009.

30. Kanj, I., M. J. Pelsmajer, M. Schaefer and G. Xia, "On the induced matching problem", *Journal of Computer and System Sciences*, Vol. 77, p. 1058–1070, 2011.

31. Bodur, M., T. Ekim and Z. C. Taşkın, "Decomposition Algorithms for Solving the Minimum Weight Maximal Matching Problem", *Networks*, Vol. 62, No. 4, pp. 273–287, 2013.

32. Taşkın, Z. C. and M. Çevik, "Combinatorial Benders Cuts for Decomposing IMRT Fluence Maps Using Rectangular Apertures", *Computers and Operations Research*, Vol. 40, No. 9, pp. 2178– 2186, 2013.

33. Dezsoa, B., A. Juttnerb and P. Kovacs, "LEMON – an open source C++ graph template library", *Electronic Notes in Theoretical Computer Science*, Vol. 264, pp. 23–45, 2011.