Available at
www.ElsevierComputerScience.com
POWERED BY SCIENCE DIRECT®
Journal of Computer and System Sciences 67 (2003) 833–847
JOURNAL OF COMPUTER AND SYSTEM SCIENCES
http://www.elsevier.com/locate/jcss

Constrained minimum vertex cover in bipartite graphs: complexity and parameterized algorithms ☆

Jianer Chen[a,*,1] and Iyad A. Kanj[b,2]

[a] *Department of Computer Science, Texas A&M University, College Station, TX 77843-3112, USA*
[b] *School of CTI, DePaul University, 243 S. Wabash Avenue, Chicago, IL 60604, USA*

Received 21 September 2001; revised 25 August 2003

Abstract

Motivated by the research in reconfigurable memory array structures, this paper studies the complexity and algorithms for the *constrained minimum vertex cover* problem on bipartite graphs (MIN-CVCB) defined as follows: given a bipartite graph $G = (V, E)$ with vertex bipartition $V = U \cup L$ and two integers k_u and k_l, decide whether there is a minimum vertex cover in G with at most k_u vertices in U and at most k_l vertices in L. It is proved in this paper that the MIN-CVCB problem is NP-complete. This answers a question posed by Hasan and Liu. A parameterized algorithm is developed for the problem, in which classical results in matching theory and recently developed techniques in parameterized computation theory are nicely combined and extended. The algorithm runs in time $O(1.26^{k_u+k_l} + (k_u + k_l)|G|)$ and significantly improves previous algorithms for the problem.

Keywords: Vertex cover; Bipartite graph; Graph matching; Parameterized computation

1. Introduction

Parameterized computation and complexity [7] have recently drawn a lot of attention. In particular, by fully taking the advantage of small parameter values, the development of efficient parameterized algorithms has provided a new approach for *practically* solving problems that are theoretically intractable. For example, parameterized algorithms for the NP-hard problem

☆ A preliminary version of this paper was presented at the *27th International Workshop on Graph-Theoretical Concepts in Computer Science*, Boltenhagen, Germany, 2001, and appeared in [4].

* Corresponding author. Fax: +1-409-847-8578.
E-mail addresses: chen@cs.tamu.edu (J. Chen), ikanj@cs.depaul.edu (I.A. Kanj).

[1] Supported in part by the National Science Foundation under Grants CCR-0000206 and CCR-0311590.
[2] Supported in part by DePaul University Competitive Research Grant.

0022-0000/$ - see front matter © 2003 Published by Elsevier Inc.
doi:10.1016/j.jcss.2003.09.003

VERTEX COVER [1,5] have found applications in biochemistry [2], and parameterized algorithms in computational logic [13] have provided an effective method for solving practical instances of the ML TYPE-CHECKING problem, which is complete for the class EXPTIME [11].

In this paper, we study the parameterized problem *constrained minimum vertex cover in bipartite graphs* (shortly MIN-CVCB) defined as follows:

> Given a bipartite graph $G = (V, E)$ with the vertex bipartition $V = U \cup L$ and two integers k_u and k_l, determine whether there is a minimum vertex cover of G with at most k_u vertices in U and at most k_l vertices in L,

where the parameter values k_u and k_l are assumed to be much smaller than the graph size $|G|$.

This problem is motivated by the study of fault coverage in reconfigurable memory array structures. A typical reconfigurable memory array consists of a rectangular array plus a certain number of spare rows and spare columns [10]. A defective element in the array can be repaired by replacing the row or the column that contains the element with a spare row or a spare column. In general, the number of spare rows and spare columns is much smaller compared with the total number of rows and columns in the array, and the cost of the fault coverage is proportional to the number of replaced rows and columns. Therefore, it is desirable to minimize the total number of spare rows and spare columns used in the fault coverage. Moreover, the number of available spare rows and the number of available spare columns pre-specify a further constraint in the replacement.

This problem can be reduced to the MIN-CVCB problem as follows. For a given reconfigurable array A, we construct a bipartite graph $G_A = (U \cup L, E)$, where each vertex in U corresponds to a row in A and each vertex in L corresponds to a column in A. There is an edge between a vertex in U and a vertex in L if and only if the element at the intersection of the corresponding row and the corresponding column is defective. Now suppose that the reconfigurable array A has k_u spare rows and k_l spare columns. It is easy to see that constructing a fault coverage that minimizes the total number of replaced rows and columns, with the constraint that at most k_u spare rows and at most k_l spare columns can be used, is equivalent to constructing a minimum vertex cover of the graph G_A with at most k_u vertices in U and at most k_l vertices in L.

The MIN-CVCB problem and its variations have been extensively studied in the last two decades (see, e.g., [10,15,22] and their references). In particular, a more general version of MIN-CVCB, the CVCB problem in which the minimization of the vertex cover is not required, is proved to be NP-complete [12]. Hasan and Liu [10] asked about the complexity of the MIN-CVCB problem. In order to solve the MIN-CVCB problem, Hasan and Liu [10] introduced the concept of *critical sets* to develop a branch-and-bound algorithm solving the MIN-CVCB problem, based on the A^* algorithm [20]. No explicit analysis was given in [10] for the running time of the algorithm, but it is not hard to see that the worst-case running time of the algorithm is $\Omega(2^{k_1+k_2} + m\sqrt{n})$.

In the current paper, we continue to study the complexity of the MIN-CVCB problem. We first show that the MIN-CVCB problem is NP-complete by a reduction from the NP-hard problem CLIQUE. This answers the question posed by Hasan and Liu [10]. We then concentrate on the development of more efficient algorithms for the MIN-CVCB problem. Classical results in matching theory and recently developed techniques in parameterized computation theory are nicely combined and extended. In particular, we observe that the Gallai–Edmonds Structure Theorem

[14] applied to bipartite graphs reduces the MIN-CVCB problem to the MIN-CVCB problem on bipartite graphs with perfect matching. In fact, the Gallai–Edmonds Structure Theorem is a stronger (and a more systematical) version of the concept of the critical sets proposed by Hasan and Liu [10]. The Gallai–Edmonds Structure Theorem also provides a solid basis so that the recently developed technique in parameterized algorithms, *reduction to problem kernel*, can be nicely applied. We then use the Dulmage–Mendelsohn Decomposition [14] to decompose a bipartite graph with a perfect matching into elementary bipartite subgraphs. This decomposition makes the other technique, *bounded search tree*, in the study of parameterized computation become much more effective. Combining all these enables us to derive an algorithm for the MIN-CVCB problem whose running time is bounded by $O(1.26^{k_u+k_l}+(k_u+k_l)n)$. This is a significant improvement over Hasan and Liu's [10] algorithm of running time $\Omega(2^{k_u+k_l}+m\sqrt{n})$.

We point out that our results are relevant to the current study of "efficient" exponential time algorithms for NP-hard problems [6]. However, existing techniques developed to solve related NP-hard problems, such as the minimum vertex cover problem, do not seem to apply to the MIN-CVCB problem. For instance, the techniques developed for low-degree vertices to solve the minimum vertex cover problem (see for example [1,5,18]) cannot be used for the MIN-CVCB problem. An algorithm of running time $O(1.40^{k_u+k_l}+(k_u+k_l)n)$ developed by Fernau and Niedermeier [8] for the CVCB problem could be used to solve the MIN-CVCB problem, but our algorithm is significantly faster, and much simpler, using elegant and deep results in graph theory.

We close this section with a brief review on the necessary definitions. For a subset $S\subseteq V$ of vertices in a graph G, denote by $G(S)$ the subgraph of G induced by S. A vertex set C in a graph is a *vertex cover* if every edge in the graph has at least one endpoint in C. A vertex cover is *minimum*, if it contains the minimum number of vertices over all vertex covers of the graph. A graph G is *bipartite* if its vertex set can be partitioned into two sets U (the "upper part") and L (the "lower part") such that every edge in G has one endpoint in U and the other endpoint in L. A bipartite graph is written as $G=(U\cup L,E)$ to indicate the vertex bipartition. The vertex sets U and L are called the *U-part* and the *L-part* of the graph, and a vertex is a *U-vertex* (resp. an *L-vertex*) if it is in the U-part (resp. in the L-part) of the graph.

2. A linear kernel for the MIN-CVCB problem

Let $\langle x,k\rangle$ be an instance of a parameterized problem Q, where k is the parameter. By *kernelizing* the instance $\langle x,k\rangle$, we mean applying a polynomial time preprocessing algorithm on $\langle x,k\rangle$ to construct another instance $\langle x',k'\rangle$ for Q such that (1) $k'\leqslant k$; (2) the size $|x'|$ of x' is bounded by a function of k'; and (3) a solution for $\langle x,k\rangle$ can be derived in polynomial time from a solution for $\langle x',k'\rangle$. Kernelization has proved to be very effective in developing efficient parameterized algorithms [7].

In this section, we show that the classical Gallai–Edmonds Structure Theorem [14] can be applied to significantly improve the kernelization of the problem MIN-CVCB.

Recall that a *matching* M in a graph G is a set of edges in G such that no two edges in M share a common endpoint. A vertex v is *matched* if it is an endpoint of an edge in M and *unmatched* otherwise. The matching M is *perfect* if all vertices in G are matched. The matching M is

maximum if it has the maximum number of edges over all matchings in G. It is well known that for a bipartite graph, the number of edges in a maximum matching is equal to the number of vertices in a minimum vertex cover [14]. Given a matching M in a graph G, an *alternating path* (with respect to M) is a simple path $\{v_0, v_1, \ldots, v_r\}$ such that v_0 is unmatched and the edges $[v_{2k-1}, v_{2k}]$ are in M for all $1 \leqslant k \leqslant \lfloor r/2 \rfloor$. An alternating path is an *augmenting path* if it has odd length and both its endpoints are unmatched. It is well known that the matching M in G is maximum if and only if there is no augmenting path in G with respect to M [14]. A maximum matching of a graph with n vertices and m edges can be constructed in time $O(m\sqrt{n})$ [16]. For a bipartite graph, a minimum vertex cover can be constructed from a maximum matching in linear time [14].

Consider a bipartite graph $G = (V, E)$. Let D be the set of vertices in G such that each vertex in D is not matched in at least one maximum matching. Let A be the set of vertices in $V - D$ such that each vertex in A is adjacent to at least one vertex in D. Let $C = V - D - A$.

Proposition 2.1 (The Gallai–Edmonds Structure Theorem, Lovász and Plummer [14]). *Let G be a bipartite graph with the sets D, A, and C defined above. Then* (1) *the set D is an independent set in G in which no vertex is contained in any minimum vertex cover*; (2) *A is the intersection of all minimum vertex covers for G*; *and* (3) *the induced subgraph $G(C)$ has a perfect matching*.

We point out that the Gallai–Edmonds Structure Theorem was originally developed for general graphs. The Gallai–Edmonds Structure Theorem on bipartite graphs was also discovered by Dulmage and Mendelsohn (see [14, p. 99]).

The sets D, A, and C can be constructed via a maximum matching M in the bipartite graph G. We briefly describe the algorithm and prove its correctness as follows. Let W be the set of unmatched vertices under the matching M. Define $\overline{W}$ to be the set of vertices such that a vertex v is in $\overline{W}$ if and only if v is reachable from a vertex in W via an alternating path of even length. See Fig. 1 for an illustration.

Lemma 2.2. *The set $\overline{W}$ is exactly the set D in Proposition* 2.1.

Proof. Let $v \in \overline{W}$. By the definition of $\overline{W}$, there exists a vertex w_0 in W that is connected to v by an alternating path of even length. Let $P = (w_0, w_1, \ldots, w_{2k} = v)$ be this path, where $[w_{2i+1}, w_{2i+2}]$ are in M, for $i = 0, \ldots, k-1$. Since the number of edges in $P \cap M$ is equal to the number of edges in $P \cap (E - M)$, and since the vertex w_0 is unmatched, the matching

$$M' = M - \{[w_{2i+1}, w_{2i+2}] \mid i = 0, \ldots, k-1\} \cup \{[w_{2i}, w_{2i+1}] \mid i = 0, \ldots, k-1\}$$

is a maximum matching of G in which the vertex v is unmatched, i.e., $v \in D$. This proves $\overline{W} \subseteq D$.

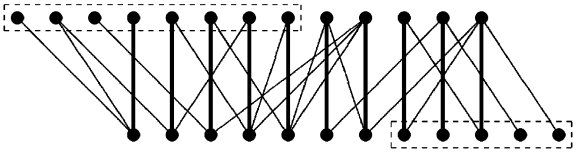

Fig. 1. The set $\overline{W}$ (marked by the dashed boxes).

Conversely, let $v \notin \overline{W}$. Since $W \subseteq \overline{W}$, $v \notin W$, and hence $[v, v']$ is in M for some vertex v'. Now remove v from G, v' cannot be connected to any vertex in W by an alternating path of odd length; otherwise, v would be connected to a vertex in W by a path of even length, and consequently, v would be in $\overline{W}$ which is not the case. Therefore, after removing v from the graph G, there is no augmenting path in $G - \{v\}$ with respect to the matching $M - [v, v']$. Thus, the matching $M - [v, v']$ is maximum, and every maximum matching in the graph $G - \{v\}$ contains $|M| - 1$ edges. This shows that the vertex v must be in every maximum matching in the graph G, and hence, $v \notin D$. Consequently, $D \subseteq \overline{W}$ and the set $\overline{W}$ is exactly the set D in Proposition 2.1. □

The sets A and C can be constructed in linear time from the set D in a straightforward way.

Theorem 2.3. *The time complexity of solving an instance* $\langle G; k_u, k_l \rangle$ *of the* MIN-CVCB *problem, where* G *is a bipartite graph of* n *vertices and* m *edges, is bounded by* $O(m\sqrt{n} + t(k_u + k_l))$, *where* $t(k_u + k_l)$ *is the time complexity for solving an instance* $\langle G', k'_u, k'_l \rangle$ *of* MIN-CVCB, $k'_u \leqslant k_u$, $k'_l \leqslant k_l$, *and* G' *has a perfect matching and contains at most* $2(k'_u + k'_l)$ *vertices.*

Proof. Suppose that the graph is $G = (U \cup L, E)$, and the instance $\langle G, k_u, k_l \rangle$ asks for a minimum vertex cover for G with at most k_u U-vertices and at most k_l L-vertices. By the Gallai–Edmonds Structure Theorem, we apply the algorithm described above to construct the sets D, A, and C. The algorithm has its time complexity dominated by the complexity of constructing a maximum matching in G, which is bounded by $O(m\sqrt{n})$. Moreover, the graph $G' = G(C) = (U' \cup L', E')$ is bipartite, where $U' = U \cap C$ and $L' = L \cap C$. Since the set A is contained in *every* minimum vertex cover for G and no vertex in the set D is in *any* minimum vertex cover, if the set A contains h_u U-vertices and h_l L-vertices, and if we let $k'_u = k_u - h_u$ and $k'_l = k_l - h_l$, then the graph G has a minimum vertex cover with at most k_u U-vertices and at most k_l L-vertices if and only if the graph G' has a minimum vertex cover with at most k'_u U-vertices and at most k'_l L-vertices. By the Gallai–Edmonds Structure Theorem, the graph G' has a perfect matching, which implies that a minimum vertex cover of G' contains exactly half of the vertices in G'. In consequence, the graph G' contains at most $2(k'_u + k'_l)$ vertices. □

We point out that Theorem 2.3 is a significant improvement over previous research when kernelization for the problem MIN-CVCB is concerned. Hasan and Liu [10] proposed the concept of *critical sets* in their study of the MIN-CVCB problem. The critical set of a bipartite graph G is the intersection of all minimum vertex covers for G. Equivalently, the critical set of the bipartite graph G is the set A in the Gallai–Edmonds Structure Theorem. However, Hasan and Liu were unable to characterize the structure of the remaining graph when the critical set is removed from the graph G. Due to the lack of this observation, they proposed to interlace the process of branch-and-bound searching and the process of constructing the critical set [10]. On the other hand, Theorem 2.3 indicates explicitly that the remaining graph G' after removing the critical set A and the independent set D has a perfect matching. This fact immediately limits the size of the graph G'. Moreover, since every minimum vertex cover for the graph G' contains exactly one endpoint from every edge in a perfect matching in G', a careful study can show that the construction of critical sets during the branch-and-bound process proposed by Hasan and Liu [10] is totally unnecessary.

This observation can significantly speed up the searching process. Finally, having a perfect matching in the graph G' will enable us to derive further powerful structure techniques in the searching process, which will be illustrated in detail in Section 4.

Fernau and Niedermeier [8] proposed a kernelization technique when they studied the more general problem CVCB. If we apply their kernelization to an instance $\langle G, k_u, k_l \rangle$ of the MIN-CVCB problem, we will get a reduced graph G' of at most $2k_uk_l$ vertices, while Theorem 2.3 enables us to reduce the graph size to at most $2(k_u + k_l)$ vertices. In the development of parameterized algorithms for minimum vertex cover for general graphs, Chen et al. [5] proposed to apply the classical Nemhauser–Trotter Theorem [17] that can reduce the problem size to $2k$, where k is the size of a minimum vertex cover. Similar to the Gallai–Edmonds Structure Theorem, the Nemhauser–Trotter Theorem decomposes the vertices of a graph into three parts D', A', and C', where D' is an independent set, A' is a subset of *some* minimum vertex cover, and the induced subgraph $G(C')$ has its minimum vertex cover of size at least half of C'. When applied to bipartite graphs, the Gallai–Edmonds Structure Theorem, which decomposes the vertices of G into the three sets D, A, and C, is much stronger than the Nemhauser–Trotter Theorem in all aspects: the set A is contained in *every* minimum vertex cover for G while the set A' is only contained in *some* minimum vertex cover for G. In particular, for the application of the MIN-CVCB problem, the set A' cannot be automatically included in the minimum vertex cover being searched, since a minimum vertex cover with at most k_u U-vertices and at most k_l L-vertices may not contain the entire set A'. Moreover, since the subgraph $G(C)$ has a perfect matching, it follows directly that a minimum vertex cover for $G(C)$ contains at least half of the vertices in C. Also, since the subgraph $G(C')$ does not necessarily have a perfect matching, $G(C')$ cannot take the advantage of the techniques used in the searching process, which will be described in Section 4.

3. The MIN-CVCB problem is NP-complete

Hasan and Liu [10] have asked for the complexity of the MIN-CVCB problem. In this section, we give a precise answer to this question by showing that the MIN-CVCB problem is NP-complete.

A *clique* of size q in a graph G is a set Q of q vertices in G such that every two vertices in Q are adjacent. An instance of the CLIQUE problem is a pair $\langle G, q \rangle$, where G is a graph and q is an integer, asking whether the graph G contains a clique of size q. CLIQUE is a well-known NP-complete problem [9].

Let $G = (U \cup L, E)$ be a bipartite graph with a perfect matching. The graph G is *elementary* if every edge in G is contained in a perfect matching in G. It is known that an elementary bipartite graph has exactly two minimum vertex covers, namely U and L (see [14, Theorem 4.1.1]). An example of an elementary bipartite graph is a simple cycle of even length $[v_1, v_2, \ldots, v_{2h}]$, in which there are only two minimum vertex covers $\{v_1, v_3, \ldots, v_{2h-1}\}$ and $\{v_2, v_4, \ldots, v_{2h}\}$.

Theorem 3.1. *The* MIN-CVCB *problem is* NP*-complete.*

Proof. It is quite easy to see that the MIN-CVCB problem is in NP: given an instance $\langle G; k_u, k_l \rangle$, where $G = (U \cup L, E)$ is a bipartite graph, simply guess no more than k_u U-vertices and

no more than k_l L-vertices in the graph G, then check if they make a minimum vertex cover (note that the size of a minimum vertex cover for a bipartite graph can be calculated in polynomial time).

Now we show that the NP-complete problem CLIQUE is polynomial time reducible to the MIN-CVCB problem. Let $\langle G, q \rangle$ be an instance of the CLIQUE problem, where $G = (V, E)$ is a simple graph with n vertices and m edges, and q is a positive integer. We construct an instance $\langle B, k_u, k_l \rangle$ for the MIN-CVCB problem, where $B = (U \cup L, E_B)$ is a bipartite graph, and k_u, k_l are non-negative integers, as follows. Each vertex w in G is associated with a *vertex-block* B_w in B consisting of a vertex $u_w \in U$, a vertex $v_w \in L$, and an edge $[u_w, v_w] \in E_B$. Each edge e in G is associated with an *edge-block* B_e in B that is a cycle of $2n+2$ vertices with the vertex bipartition (U_e, L_e), where $U_e \subseteq U$ has $n+1$ vertices and $L_e \subseteq L$ has $n+1$ vertices (note that for two edges e and e' in G, the two corresponding edge-blocks B_e and $B_{e'}$ in B share no common vertex). Finally, if a vertex w in G is an endpoint of an edge e in G, we add a new edge from the U-vertex u_w in the vertex-block B_w to any L-vertex in the edge-block B_e. This kind of edges will be called "inter-block edges". Note that there is no inter-block edge that is incident either to an L-vertex in a vertex-block or to a U-vertex in an edge-block. This completes the construction of the graph B. It is easy to see that the graph B is bipartite, and that B has a perfect matching of $n + m(n+1)$ edges, which consists of the edge from each vertex-block and $n+1$ alternate edges from each edge-block. Therefore, a minimum vertex cover in B contains exactly $n + m(n+1)$ vertices. Now we let $k_u = q + q(q-1)(n+1)/2$ and $k_l = n + m(n+1) - k_u$. The instance $\langle B, k_u, k_l \rangle$ of the MIN-CVCB problem can certainly be constructed from the instance $\langle G, q \rangle$ of the CLIQUE problem in polynomial time.

Every minimum vertex cover for the bipartite graph B contains at least one vertex in each vertex-block (since the two vertices in a vertex-block are connected by an edge) and at least $n+1$ vertices in each edge-block (since an edge-block has $2n+2$ vertices and has a perfect matching). Now since the graph B has exactly n vertex-blocks and m edge-blocks, and a minimum vertex cover for B has exactly $n + m(n+1)$ vertices, we conclude that every minimum vertex cover contains exactly one vertex in each vertex-block and exactly $n+1$ vertices in each edge-block. Thus, the intersection of a minimum vertex cover and a block (either a vertex- or an edge-block) must be a minimum vertex cover for the block.

The vertex- and edge-blocks in B are elementary bipartite graphs. Thus, each block has exactly two minimum vertex covers, namely the U-part and the L-part of the block. In consequence, each minimum vertex cover of the graph B consists of either the U- or the L-part (but not both) from each block.

Now, we are ready to prove that the original graph G has a clique of size q if and only if the bipartite graph B has a minimum vertex cover with at most k_u U-vertices and at most k_l L-vertices.

Let Q be a clique of q vertices in the graph G. We construct a minimum vertex cover with k_u U-vertices and k_l L-vertices for the bipartite graph B. Let C_Q be the set in B that consists of the U-parts of the q vertex-blocks and the $q(q-1)/2$ edge-blocks corresponding to the q vertices and $q(q-1)/2$ edges in the clique Q, and the L-parts of the rest of the vertex-blocks and edge-blocks. The set C_Q has totally $n + m(n+1)$ vertices, which is the size of a minimum vertex cover for B, with $k_u = q + q(q-1)(n+1)/2$ U-vertices in B and $k_l = n + m(n+1) - k_u$ L-vertices in B. Therefore, it suffices to show that the set C_Q is a vertex cover for B.

Each edge connecting two vertices in the same block is covered by C_Q since either the U- or the L-part of the block is entirely contained in the set C_Q. Now consider an inter-block edge e' that connects the U-vertex of a vertex-block B_w to an L-vertex in an edge-block B_e, where w is an endpoint of the edge e in the graph G. If the edge e is not in the clique Q, then the L-part of B_e is in the set C_Q, while if the edge e is in the clique Q, then the U-vertex of B_w is in the set C_Q. Therefore, all edges in B are covered by the vertex set C_Q and C_Q makes a minimum vertex cover for B with k_u U-vertices and k_l L-vertices. Thus, a clique of size q in the graph G implies a minimum vertex cover of k_u U-vertices and k_l L-vertices for the bipartite graph B.

Now suppose that the bipartite graph B has a minimum vertex cover C of k_u U-vertices and k_l L-vertices. Since $k_u = q + q(q-1)(n+1)/2$ and the intersection of C with each block is either exactly the U-part or exactly the L-part of the block, we conclude that the vertex cover C contains the U-parts of exactly q vertex-blocks and $q(q-1)/2$ edge-blocks. Let S_C be the set of these blocks in B, and let V_C be the vertex set of the q vertices in the graph G corresponding to the q vertex-blocks in S_C, and E_C be the edge set of $q(q-1)/2$ edges in G corresponding to the $q(q-1)/2$ edge-blocks in S_C. Consider an edge-block B_e in S_C, where $e = [w_1, w_2]$ is an edge in the graph G. Then there must be an inter-block edge from the U-vertex of the vertex-block B_{w_1} to an L-vertex in the edge-block B_e, and an inter-block edge from the U-vertex in the vertex-block B_{w_2} to an L-vertex in the edge-block B_e. Since no L-vertex in the edge-block B_e is in the vertex cover C, the U-parts of the vertex-blocks B_{w_1} and B_{w_2} must be in C. In consequence, the vertices w_1 and w_2 are in the set V_C. Now each of the $q(q-1)/2$ edges in the set E_C has its both endpoints in the set V_C, and the set V_C has only q vertices. We conclude that the vertex set V_C and the edge set E_C must make a clique of size q in the graph G. Thus, a minimum vertex cover with k_u U-vertices and k_l L-vertices in the bipartite graph B implies a clique of q vertices in the original graph G.

This completes the proof that the CLIQUE problem is polynomial time reducible to the MIN-CVCB problem. In consequence, the MIN-CVCB problem is NP-complete. □

4. Efficient branching by the Dulmage–Mendelsohn Decomposition

According to Theorem 2.3, we can concentrate on an instance $\langle G; k_u, k_l \rangle$ for the MIN-CVCB problem, where G is a bipartite graph that has a perfect matching and has at most $2(k_u + k_l)$ vertices.

Recall that a bipartite graph $G = (U \cup L, E)$ is *elementary* if every edge of G is contained in a perfect matching in G, and that the elementary bipartite graph G has exactly two minimum vertex covers, namely U and L. Note that this property of an elementary bipartite graph makes our searching process very efficient: on an elementary bipartite graph, we should either include the entire U-part or include the entire L-part of the graph in the minimum vertex cover being searched. There is no other possibility.

The following classical result [14] provides a nice structure for a bipartite graph with perfect matching so that the above suggested searching process can be applied effectively.

Proposition 4.1 (The Dulmage–Mendelsohn Decomposition Theorem, Lovász and Plummer [14]). *A bipartite graph $G = (U \cup L, E)$ with perfect matching can be decomposed and indexed into*

elementary subgraphs $B_i = (U_i \cup L_i, E_i)$, $i = 1, 2, \ldots, r$, *such that every edge in* G *from a subgraph* B_i *to a subgraph* B_j *with* $i < j$ *must have one endpoint in the* U*-part of* B_i *and the other endpoint in the* L*-part of* B_j.

The elementary subgraphs B_i will be called (elementary) *blocks*. The block B_i is a *d-block* if $|U_i| = |L_i| = d$. Edges connecting vertices in two different blocks will be called "inter-block edges".

We first consider the complexity of the construction of the Dulmage–Mendelsohn Decomposition.

Lemma 4.2. *Let* $G = (U \cup L, E)$ *be a bipartite graph with perfect matching. Then the Dulmage–Mendelsohn Decomposition of* G *can be constructed in time* $O(|E|^2)$.

Proof. It is known that an edge e in G is an inter-block edge if and only if e is not contained in any perfect matching in G ([14], Section 4.3). Consider the algorithm in Fig. 2.

Note that an edge $e = [u, v]$ is in a perfect matching in the graph G if and only if the graph $G - \{u, v\}$ has a perfect matching (where $G - \{u, v\}$ is the graph G with the two vertices u and v removed). Therefore, step 1 of the algorithm DM-Decomposition correctly determines the inter-block edges. By the Dulmage–Mendelsohn Decomposition Theorem, steps 2–3 correctly construct the elementary blocks for the graph G. Finally, if we construct a directed graph D whose vertices are the elementary blocks in G such that there is an edge from B to B' if and only if there is an inter-block edge connecting a U-vertex in B and an L-vertex in B', then by the Dulmage–Mendelsohn Decomposition Theorem, the graph D is a directed acyclic graph. Thus, a topological sorting will re-order the vertices in D such that each edge is from a vertex of lower index to a vertex of higher index. Equivalently, this indexing makes each inter-block edge in the graph G connect a U-vertex of a lower indexed block to an L-vertex of a higher indexed block.

Now we consider the complexity of the algorithm DM-Decomposition. Before the execution of step 1, we first construct a perfect matching M in the graph G. This takes time $O(|E|\sqrt{|U| + |L|}) = O(|E|^2)$ [3]. For each edge $e = [u, v]$ in G, if e is in M then obviously e is not an inter-block edge, while if e is not in M, then after removing the vertices u and v and their

Algorithm. DM-Decomposition.

Input: a bipartite graph $G = (U \cup L, E)$ with perfect matching

Output: the blocks $B_1, \ldots, B_r$ given by the Dulmage-Mendelsohn Decomposition

1. **for** each edge $e = [u, v]$ in G **do**
   if the graph $G^- = G - \{u, v\}$ has no perfect matching
   then mark e as an "inter-block edge";
2. remove all inter-block edges;
3. each connected component in the remaining graph makes an elementary block;
4. topologically sort the blocks into a sequence $B_1, \ldots, B_r$ so that every inter-block edge connects a U-vertex of a lower indexed block to an L-vertex of a higher indexed block.

Fig. 2. The Dulmage–Mendelsohn Decomposition.

incident edges, the matching M becomes a matching M^- of $|M|-2$ edges in the remaining graph $G^- = G - \{u, v\}$. Thus, the remaining graph G^- has a perfect matching if and only if there is an augmenting path in G^- with respect to the matching M^-. Since testing the existence of an augmenting path can be done in linear time using a variation of the Breadth First Search process [3], we conclude that with the perfect matching M, we can decide whether an edge in G is an inter-block edge in time $O(|E|)$. Therefore, step 1 of the algorithm DM-Decomposition takes time $O(|E|^2)$. Finally, the topological sorting in step 4 takes time $O(|E|)$. In summary, the time complexity of the algorithm DM-Decomposition is bounded by $O(|E|^2)$. □

Lemma 4.3. *Let G be a bipartite graph with perfect matching, and let $B_1, \ldots, B_r$ be the blocks of G given by the Dulmage–Mendelsohn Decomposition. Then any minimum vertex cover for G is the union of minimum vertex covers of the blocks $B_1, \ldots, B_r$.*

Proof. Let C be a minimum vertex cover for G, and let $d_i = |U_i| = |L_i|$ for each $B_i = (U_i, L_i)$. Since C is a minimum vertex cover for G, C contains exactly $|U| = |L| = \sum_{i=1}^{r} d_i$ vertices. Moreover, the induced cover by C on each block B_i (i.e., $C \cap B_i$) is a vertex cover for B_i. Since each block B_i is elementary, B_i has exactly two minimum vertex covers, namely U_i and L_i, each of size d_i. Consequently, the induced cover by C on B_i has size $|C \cap B_i|$ that is at least d_i. It follows from the equality $|C| = \sum_{i=1}^{r} d_i$ that $|C \cap B_i| = d_i$ for $i = 1, \ldots, r$, and hence the induced cover by C on each block B_i is a minimum vertex cover for B_i. □

Now, we are ready to describe the main body of our algorithm for solving the MIN-CVCB problem. Let $\langle G; k_u, k_l \rangle$ be an instance of the MIN-CVCB problem, where $G = (U \cup L, E)$ is a bipartite graph with perfect matching, and G has at most $2(k_u + k_l)$ vertices. We first apply the Dulmage–Mendelsohn Decomposition Theorem to the graph G to decompose it into elementary blocks $B_1, \ldots, B_r$ such that every inter-block edge connects a U-vertex in a block of lower index to an L-vertex in a block of higher index. By Lemma 4.3, a minimum vertex cover of the graph G is the union of minimum vertex covers of the elementary blocks $B_1, \ldots, B_r$. Since for each elementary block $B_i = (U_i \cup L_i, E)$, the only two minimum vertex covers of B_i are U_i and L_i, the constrained minimum vertex cover K of the graph G with at most k_u U-vertices and at most k_l L-vertices must be the union of properly selected U-parts and L-parts from the elementary blocks.

We construct a directed acyclic graph (DAG) D of vertices $\{B_1, \ldots, B_r\}$ such that there is an edge from B_i to B_j if and only if there is an inter-block edge in G from a U-vertex in B_i to an L-vertex in B_j. We assign each vertex B_i in D a *weight* $d_i = |U_i| = |L_i|$.

The execution of our algorithm is depicted by a search tree whose leaves correspond to the potential constrained minimum vertex covers K of the graph G with at most k_u U-vertices and at most k_l L-vertices. Each internal node of the search tree corresponds to a branch in the searching process. Let $F(k_u + k_l)$ be the number of leaves in the search tree for finding a minimum vertex cover of at most k_u U-vertices and at most k_l L-vertices in the bipartite graph G. Our algorithm is based on the DAG D constructed above. We list the possible situations in which we branch in our search process.

Case 1: A vertex B_i in D has weight at least 3.

Let $B_i = (U_i \cup L_i, E_i)$, where $d = |U_i| = |L_i| \geq 3$. Since the constrained minimum vertex cover K of the graph G either contains the entire U_i-part and is disjoint from the L_i-part, or contains the entire L_i-part and is disjoint from the U_i-part of the block B_i, we branch in this case by either including the entire U_i-part in K (and removing the L_i-part from the graph) or including the entire L_i-part in K (and removing the U_i-part from the graph). In each case, we add at least three vertices in the constrained minimum vertex cover K and remove the vertex B_i from the DAG D. Thus, this branch satisfies the recurrence relation

$$F(k_u + k_l) \leqslant 2F(k_u + k_l - 3). \tag{1}$$

We observe that the *chain implication* can speed up the searching process significantly, as follows. Let $[B'_1, B'_2, \ldots, B'_h]$ be a path in the DAG D. If we include the L-part of the block B'_1 in the minimum vertex cover K, then the U-part of B'_1 must be excluded from K. Since there is an edge in G from the U-part of B'_1 to the L-part of the block B'_2, we must also include the L-part of the block B'_2 in the minimum vertex cover K, which, in consequence, will imply that the L-part of the block B'_3 must be in the minimum vertex cover K, and so on. In particular, the L-part of the block B'_1 in the minimum vertex cover K implies that the L-parts of all blocks $B'_2, \ldots, B'_h$ on the path must be in the minimum vertex cover K. Similarly, the U-part of the block B'_h in the minimum vertex cover K implies that the U-parts of all blocks $B'_1, \ldots, B'_{h-1}$ must be in the minimum vertex cover K. This observation enables us to handle many cases very efficiently.

Case 2: There is a path in D in which a weight-2 vertex is an interior vertex.

Let B be a weight-2 vertex that is an interior vertex on a path in D: $[\ldots, B_i, B, B_j, \ldots]$. Then including the L-part of B in the minimum vertex cover K forces at least three vertices in K (the L-parts of B and B_j), and excluding the L-part of B also forces at least three vertices in K (the U-parts of B and B_i). Thus, in this case, the branch satisfies recurrence relation (1).

Case 3: There are three edges in D from three different weight-2 vertices to the same weight-1 vertex, or from the same weight-1 vertex to three different weight-2 vertices.

Suppose there are three edges $[B'_1, B'_0]$, $[B'_2, B'_0]$, and $[B'_3, B'_0]$ in the DAG D, where B'_1, B'_2, and B'_3 are weight-2 vertices and B'_0 is a weight-1 vertex in the DAG D. Then including the U-part of B'_0 forces at least 7 vertices (the U-parts of the blocks B'_0, B'_1, B'_2, and B'_3) in the minimum vertex cover K, and excluding the U-part of B'_0 forces at least one vertex (the L-part of B'_0) in K. Therefore, the branch satisfies the recurrence relation

$$F(k_u + k_l) \leqslant F(k_u + k_l - 7) + F(k_u + k_l - 1) \tag{2}$$

Similarly, if there are three edges from the same weight-1 vertex B'_0 to three different weight-2 vertices, then the branching on the L-part of B'_0 satisfies recurrence relation (2).

Case 4: All other cases not included in Cases 1–3.

Because Case 1 is excluded, the DAG D contains only weight-1 vertices and weight-2 vertices. Since Case 2 is excluded, each weight-2 vertex in the DAG D either has no incoming edges or has no outgoing edges. Therefore, we can re-order the vertices in D into a topologically sorted

sequence of three segments

$$B'_1, \ldots, B'_x, B'_{x+1}, \ldots, B'_y, B'_{y+1}, \ldots, B'_z, \tag{3}$$

where $B'_1, \ldots, B'_x$ are weight-2 vertices in D with no incoming edges, $B'_{x+1}, \ldots, B'_y$ are weight-1 vertices in D, and $B'_{y+1}, \ldots, B'_z$ are weight-2 vertices in D with no outgoing edges, such that every edge in D goes from a vertex of lower index to a vertex of higher index. Note that for any index i, $1 \leqslant i \leqslant z$, the set of vertices consisting of all U-parts of $B'_1, \ldots, B'_i$ and L-parts of $B'_{i+1}, \ldots, B'_z$ makes a minimum vertex cover for the bipartite graph G. Now to construct a constrained minimum vertex cover K of the bipartite graph G with at most k_u U-vertices and at most k_l L-vertices, we have the following situations.

If $k_u + k_l$ is strictly larger than the size of a minimum vertex cover of the graph G, then we can pick an index i such that the U-parts of the blocks $B'_1, \ldots, B'_i$ consist of either $k_u - 1$ or k_u vertices (such an index always exists since all blocks are 1-blocks and 2-blocks). Now the U-parts of the blocks $B'_1, \ldots, B'_i$ together with the L-parts of the blocks $B'_{i+1}, \ldots, B'_z$ make a minimum vertex cover with at most k_u U-vertices and at most $k - (k_u - 1) \leqslant (k_u + k_l - 1) - (k_u - 1) = k_l$ L-vertices, where k is the size of a minimum vertex cover of G.

Thus, we can assume that $k_u + k_l$ is equal to the size of a minimum vertex cover of G.

If $2x \leqslant k_u \leqslant 2x + (y - x)$, then let $i = k_u - x$. The minimum vertex cover K consisting of the U-parts of the blocks $B'_1, \ldots, B'_i$ and the L-parts of the blocks $B'_{i+1}, \ldots, B'_z$ contains $(i - x) + 2x = k_u$ U-vertices and $k - k_u = k_l$ L-vertices in the bipartite graph G. Thus, K is a desired minimum vertex cover satisfying the constraint.

If $k_u < 2x$ and k_u is even, then the U-parts of the first $k_u/2$ 2-blocks plus the L-parts of the rest of the blocks make a desired minimum vertex cover.

Now consider the case $k_u < 2x$ and k_u is odd. If the graph G has no 1-blocks, then obviously there is no minimum vertex cover with k_u U-vertices and k_l L-vertices. Thus, we assume that the graph G has at least one 1-block.

It is easy to see that for case $k_u = 1$, there is a minimum vertex cover of exactly k_u U-vertices if and only if there is a weight-1 vertex in D that has no incoming edges; and that for case $k_u = 3$, there is a minimum vertex cover of exactly k_u U-vertices if and only if there is a weight-1 vertex in the DAG D that has at most one incoming edge from a weight-2 vertex. Thus, we can further assume that k_u is at least 5.

Pick the weight-1 vertex B of the minimum index i. Since Case 3 is excluded, there are at most two weight-2 vertices B' and B'' in D such that $[B', B]$ and $[B'', B]$ are edges in D. Note that since Case 2 is excluded, no inter-block edges can be incident on the L-parts of the blocks B' and B''. Now re-order sequence (3) by deleting B, B', and B'' from the sequence then reinserting $[B', B'', B]$ in front of the sequence. We get another topologically sorted sequence of the blocks of G (since there are no incoming edges to the blocks B' and B'' and $[B', B]$ and $[B'', B]$ are the only incoming edges to the vertex B). Now the U-parts of the blocks B', B'', and B plus the next $(k_u - 5)/2$ 2-blocks in the new sequence and the L-parts of the rest of the blocks give a minimum vertex cover for G of k_u U-vertices and k_l L-vertices. In case there are fewer than two incoming edges to the 1-block B, the desired minimum vertex cover K can also be constructed similarly.

Finally, we consider the case $k_u > 2x + (y - x)$. In this case, by symmetry, we reverse the sequence (3), exchange the U-part and the L-part of the bipartite graph G, and exchange the numbers k_u and k_l. This will reduce this case to the previous one.

In conclusion, when all Cases 1–3 are excluded, the MIN-CVCB problem can be solved in linear time.

5. Putting all together

We summarize all the discussions given in the previous sections and present the complete algorithm for the problem MIN-CVCB in Fig. 3. We explain the steps of the algorithm as follows.

Steps 2 and 3 make immediate decisions on high-degree vertices. If a U-vertex u of degree larger than k_l is not in the minimum vertex cover K, then all neighbors of u should be in K, which would exceed the bound k_l. Thus, every U-vertex of degree larger than k_l should be automatically included in K. Similar justification applies to L-vertices of degree larger than k_u. Of course, if k_u or k_l becomes negative in these steps, then we should stop and claim the nonexistence of the desired minimum vertex cover. Steps 2 and 3 obviously take linear time. After these steps, the degree of the vertices in the graph is bounded by $k' = \max\{k_u, k_l\}$. Since now each vertex can cover at most k' edges, the number of edges in the resulting graph must be bounded by $k'(k_u + k_l) \leqslant (k_u + k_l)^2$, otherwise the graph cannot have a minimum vertex cover of no more than $k_u + k_l$ vertices.

Theorem 2.3 allows us to apply the Gallai–Edmonds Structure Theorem in step 4 to further reduce the bipartite graph G so that G has a perfect matching (the integers k_u and k_l are also properly reduced). The running time of this step is bounded by $(k_u + k_l)^3$. The number of vertices in the graph G now is bounded by $2(k_u + k_l)$.

Step 5 applies the Dulmage–Mendelsohn Decomposition to decompose the graph G into elementary bipartite subgraphs. By Lemma 4.2, this takes time $O((k_u + k_l)^4)$.

Algorithm. CVCB-Solver

Input: a bipartite graph $G = (U \cup L, E)$ and two integers k_u and k_l

Output: a minimum vertex cover K of G with at most k_u U-vertices and at most k_l L-vertices, or report no such a vertex cover exists

1. $K = \emptyset$;
2. **for** each U-vertex u of degree larger than k_l
   include u in K and remove u from G; $k_u = k_u - 1$;
3. **for** each L-vertex v of degree larger than k_u
   include v in K and remove v from G; $k_l = k_l - 1$;
4. apply the Gallai-Edmonds Structure Theorem to reduce the instance so that G is a bipartite graph with perfect matching and with at most $2(k_u + k_l)$ vertices (with the integers k_u and k_l and the minimum vertex cover K also properly updated);
5. apply the Dulmage-Mendelsohn Decomposition to decompose the graph G into elementary blocks $B_1, B_2, \ldots, B_r$, sorted topologically;
6. as long as one of the Cases 1-3 in section 4 is applicable, branch accordingly;
7. solve the instance in Case 4 in section 4.

Fig. 3. The main algorithm.

As we have discussed in Section 4, if any of Cases 1–3 is applicable, we can branch in our searching process with one of the recurrence relations (1) and (2). Eventually, we reach Case 4 in Section 4, which, as described, can be solved in linear time.

Now following the standard techniques [5,19], we let $F(k_u + k_l) = x^{k_u+k_l}$, apply the expression in the recurrence relations (1) and (2), and solve the corresponding polynomials. We get $x \leqslant 1.2599$. This gives $F(k_u + k_l) \leqslant 1.2599^{k_u+k_l}$. Thus, the running time of the branching stage in our algorithm is bounded by $O(1.2599^{k_u+k_l}(k_u + k_l)^2)$, where the factor $(k_u + k_l)^2$ is the kernel size (i.e., the size of the graph when we start branching at step 6). Combining all steps together, we derive that the running time of our algorithm is bounded by

$$O((k_u + k_l)n + (k_u + k_l)^4 + 1.2599^{k_u+k_l}(k_u + k_l)^2) = O((k_u + k_l)n + 1.2599^{k_u+k_l}(k_u + k_l)^2).$$

Using the techniques described in [19], we can further drop the factor $(k_u + k_l)^2$ from the term $1.2599^{k_u+k_l}(k_u + k_l)^2$ in the above bound (see [19] for a more detailed description). This concludes the following theorem.

Theorem 5.1. *The* MIN-CVCB *problem can be solved in time* $O(1.26^{k_u+k_l} + (k_u + k_l)n)$.

6. Conclusions

Parameterized computation theory has been proved to be very useful in designing practical algorithms for industrial applications, in particular for many NP-hard optimization problems. In this paper, we have studied the complexity and developed a simple and significantly improved algorithm for the constrained minimum vertex cover problem on bipartite graphs, which has direct applications in the area of VLSI manufacturing. Our algorithm is conceptually simple, easy to implement, and nicely combines classical graph structural results with the recently developed techniques in the parameterized computation theory.

We point out that our investigation emphasizes on the practicality of algorithms, which includes simplicity (both conceptual and algorithmic) and effectiveness. The running time of our algorithm could be further improved slightly if we examine in more depth the combinatorial structures in a case-by-case exhaustive manner, which will deprive our algorithm from simplicity and practicality. Dynamic programming techniques as described in [21] could also slightly improve the base of the exponent, but at the cost of requiring exponential space.

References

[1] R. Balasubramanian, M.R. Fellows, V. Raman, An improved fixed parameter algorithm for vertex cover, Inform. Process. Lett. 65 (1998) 163–168.

[2] J. Cheetham, F. Dehne, A. Rau-Chaplin, U. Stege, P.J. Taillon, Solving large FPT problems on coarse grained parallel machines, J. Comput. System Sci., 2004, in press.

[3] J. Chen, Introduction to Tractability and Approximability of Optimization Problems, Lecture Notes, Department of Computer Science, Texas A&M University, 2002, available at: http://faculty/cs.tamu.edu/chen/notes/opt.ps.

[4] J. Chen, I.A. Kanj, On constrained minimum vertex covers of bipartite graphs: improved algorithms, Proceedings of the 27th International Workshop on Graph-Theoretical Concepts in Computer Science (WG'01), Lecture Notes in Computer Science, Vol. 2204, Springer, Heidelberg, 2001, pp. 55–65.
[5] J. Chen, I.A. Kanj, W. Jia, Vertex cover: further observations and further improvement, J. Algorithms 41 (2001) 280–301.
[6] DIMACS Workshop on Faster Exact Solutions for NP-Hard Problems, Princeton, February 23–24, 2000.
[7] R. Downey, M. Fellows, Parameterized Complexity, Springer, New York, 1999.
[8] H. Fernau, R. Niedermeier, An efficient exact algorithm for constraint bipartite vertex cover, J. Algorithms 38 (2001) 374–410.
[9] M. Garey, D. Johnson, Computers and Intractability: A Guide to the Theory of NP-completeness, Freeman, San Francisco, 1979.
[10] N. Hasan, C.L. Liu, Minimum fault coverage in reconfigurable arrays, Proceedings of the 18th International Symposium on Fault-Tolerant Computing (FTCS'88), IEEE Computer Society Press, Los Alamitos, CA, 1988, pp. 348–353.
[11] F. Henglein, H.G. Mairson, The complexity of type inference for higher order typed lambda calculi, J. Funct. Programming 4 (1994) 435–477.
[12] S.-Y. Kuo, W. Fuchs, Efficient spare allocation for reconfigurable arrays, IEEE Des. Test 4 (1987) 24–31.
[13] O. Lichtenstein, A. Pneuli, Checking that finite-state concurrents programs satisfy their linear specification, Proceedings of the 12th ACM Symposium on Principles of Programming Languages (POPL), ACM Press, New York, NY, 1985, pp. 97–107.
[14] L. Lovász, M.D. Plummer, Matching Theory, in: Annals of Discrete Mathematics, Vol. 29, North-Holland, Amsterdam, 1986.
[15] C.P. Low, H.W. Leong, A new class of efficient algorithms for reconfiguration of memory arrays, IEEE Trans. Comput. 45 (1996) 614–618.
[16] S. Micali, V. Vazirani, An $O(\sqrt{|V|} \cdot |E|)$ algorithm for finding maximum matching in general graphs, Proceedings of the 21st IEEE Symposium on the Foundation of Computer Science (FOCS'80), IEEE Computer Society Press, Los Alamitos, CA, 1980, pp. 17–27.
[17] G.L. Nemhauser, L.E. Trotter, Vertex packing: structural properties and algorithms, Math. Programming 8 (1975) 232–248.
[18] R. Niedermeier, P. Rossmanith, Upper bounds for vertex cover further improved, Proceedings of the 16th Symposium on Theoretical Aspects of Computer Science (STACS'99), Lecture Notes in Computer Science, Vol. 1563, Springer, Heidelberg, 1999, pp. 561–570.
[19] R. Niedermeier, P. Rossmanith, A general method to speed up fixed-parameter-tractable algorithms, Inform. Process. Lett. 73 (2000) 125–129.
[20] N.J. Nilsson, Principles of Artificial Intelligence, Tioga Publishing Co., Palo Alto, CA, 1980.
[21] J.M. Robson, Algorithms for maximum independent sets, J. Algorithms 7 (1986) 425–440.
[22] M.D. Smith, P. Mazumder, Generation of minimal vertex cover for row/column allocation in self-repairable arrays, IEEE Trans. Comput. 45 (1996) 109–115.