RS Global Journals

Scholarly Publisher
RS Global Sp. z O.O.
ISNI: 0000 0004 8495 2390

Dolna 17, Warsaw, Poland 00-773
Tel: +48 226 0 227 03
Email: editorial_office@rsglobal.pl

JOURNAL	World Science
p-ISSN	2413-1032
e-ISSN	2414-6404
PUBLISHER	RS Global Sp. z O.O., Poland

ARTICLE TITLE	LIST COLORING OF BLOCK GRAPHS AND COMPLETE BIPARTITE GRAPHS
AUTHOR(S)	Albert Khachik Sahakyan
ARTICLE INFO	Albert Khachik Sahakyan. (2021) List Coloring of Block Graphs and Complete Bipartite Graphs. World Science. 8(69). doi: 10.31435/rsglobal_ws/30082021/7661
DOI	https://doi.org/10.31435/rsglobal_ws/30082021/7661
RECEIVED	19 June 2021
ACCEPTED	21 August 2021
PUBLISHED	25 August 2021
LICENSE	This work is licensed under a **Creative Commons Attribution 4.0 International License.**

PHYSICS AND MATHEMATICS

LIST COLORING OF BLOCK GRAPHS AND COMPLETE BIPARTITE GRAPHS

Albert Khachik Sahakyan,
Chair of Discrete Mathematics and Theoretical Informatics, Faculty of Informatics and Applied Mathematics, Yerevan State University, Armenia

DOI: https://doi.org/10.31435/rsglobal_ws/30082021/7661

ARTICLE INFO

Received: 19 June 2021
Accepted: 21 August 2021
Published: 25 August 2021

KEYWORDS

block graph, complete bipartite graphs, list coloring, edge coloring, NP-complete, dynamic programming, bipartite matching.

ABSTRACT

List coloring is a vertex coloring of a graph where each vertex can be restricted to a list of allowed colors. For a given graph G and a set L(v) of colors for every vertex v, a list coloring is a function that maps every vertex v to a color in the list L(v) such that no two adjacent vertices receive the same color. It was first studied in the 1970s in independent papers by Vizing and by Erdős, Rubin, and Taylor. A block graph is a type of undirected graph in which every biconnected component (block) is a clique. A complete bipartite graph is a bipartite graph with partitions V 1, V 2 such that for every two vertices v_1∈V_1 and v_2∈V_2 there is an edge (v 1, v 2). If |V_1 |=n and |V_2 |=m it is denoted by K_(n,m). In this paper we provide a polynomial algorithm for finding a list coloring of block graphs and prove that the problem of finding a list coloring of K_(n,m) is NP-complete even if for each vertex v the length of the list is not greater than 3 (|L(v)|≤3).

Citation: Albert Khachik Sahakyan. (2021) List Coloring of Block Graphs and Complete Bipartite Graphs. *World Science*. 8(69). doi: 10.31435/rsglobal_ws/30082021/7661

Introduction. All graphs considered in this paper are undirected (unless explicitly said), finite, and have no loops or multiple edges. For an undirected graph G, let $V(G)$ and $E(G)$ denote the sets of vertices and edges of G, respectively. The degree of a vertex $v \in V(G)$ is denoted by $d_G(v)$.

Let I_k be the set $\{1, \dots, k\}$ of colors. For a set S let 2^S be the set of all the subsets of the set S. For a graph G and a function $L: V(G) \to 2^{I_k}$ a list coloring $\beta: V(G) \to I_k$ is a coloring of the graph vertices with integers from I_k such that for every vertex v, $\beta(v) \in L(v)$ and every two adjacent vertices have different colors. It was first introduced in [3] and [4]. For each vertex v the set $L(v)$ is called a list of colors.

A block graph or clique tree [1] is a type of undirected graph in which every biconnected component (block) is a clique (every two distinct vertices in the clique are adjacent). Fig. 1 illustrates an example of a block graph.

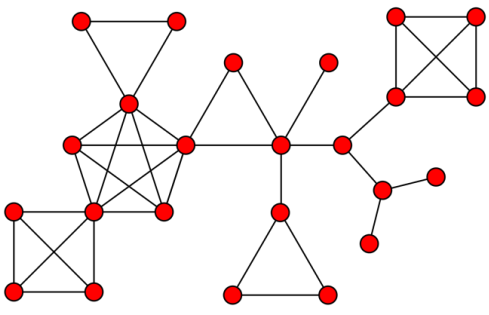

Fig 1. A block graph.

A cut vertex is any vertex whose removal increases the number of connected components [1] illustrated in Fig 2. Any connected graph decomposes into a tree of biconnected components called the block-cut tree of the graph [2]. In block graphs each block is clique.

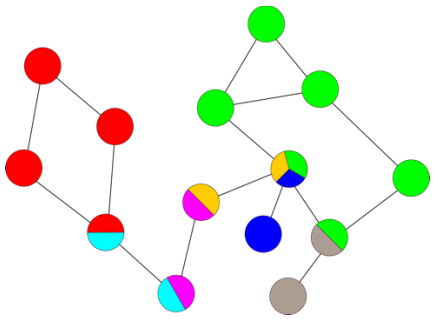

Fig 2. Each color corresponds to a biconnected component. Multi-colored vertices are cut vertices, and thus belong to multiple biconnected components.

A complete bipartite graph is a simple bipartite graph such that two vertices are adjacent if and only if they are in different partite sets. When the sets have sizes n and m, the (unlabeled) complete bipartite graph is denoted by $K_{n,m}$ [1].

For a directed graph $\vec{G}$ if there is an edge from a vertex u to a vertex v we will denote it as $u \to v$. The graph G is called the underlying graph of a directed graph $\vec{G}$ if $V(G) = V(\vec{G})$ and between any pair of vertices u and v, if the directed graph has an edge $u \to v$ or an edge $v \to u$, the underlying graph includes the edge (u, v).

For a tree T and a vertex r let T_r be the directed graph whose underlying graph is T and in T_r each edge is directed in such a way that for all vertices $v \in T_r$ there is a path in T_r from r to v. We will say that T_r is a rooted tree with the root r. Fig. 3 illustrates the rooted tree T_{v_1} with the root v_1.

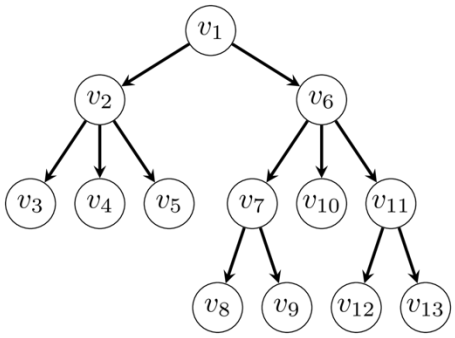

Fig. 3. A rooted tree T_{v_1} with the root v_1.

A vertex u is said to be the parent of the vertex v, denoted by $p(v)$ if $u \to v$ and in such a case, the vertex v is said to be a child of the vertex u. The children of a vertex $v \in V(T_r)$ are the set of all vertices $W \subseteq V(T_r)$ such that $v \to w$ for all $w \in V(T_r)$. A vertex having no children is said to be a leaf vertex. For a vertex v let $ST(v)$ be the subtree induced [1] by all the vertices w such that there is a path from v to w in T_r.

List coloring was first introduced in [3] and [4]. The list-coloring problem is NP-complete for general perfect graphs [5], and is also NP-complete for many subclasses of perfect graphs, including split graphs [6], interval graphs [7,8], and bipartite graphs [9]. In [6] a polynomial solution was provided for partial k-trees with the time complexity of $O(|V(G)|^{k+2})$. In [10] different subclasses of perfect graphs were considered where the vertex coloring problem has a polynomial solution but the list coloring problem is NP-Complete [11, 12]. In [13] the extended version of list coloring was considered where the colors are intervals of integers and it was shown that for bipartite and complete graphs the problem is NP-Complete.

NP-completeness of the list coloring for complete bipartite graphs

Given a complete bipartite graph $K_{n,m}$ with vertices in the first part denoted by V_1 and the vertices in the second part denoted by V_2 ($V(K_{n,m}) = V_1 \cup V_2$). In $K_{n,m}$ for every vertex $u \in V_1$ and every vertex $v \in V_2$ there is an edge connecting them $(u, v) \in E(K_{n,m})$. We are going to prove that the list coloring of $K_{n,m}$ is NP-complete even if the lists have at most 3 colors $|L(v)| \leq 3$ for all $v \in V(G)$. To do that we are going to show that the problem of 3-satisfiability (3-SAT) [12] can be solved by finding a list coloring of some $K_{n,m}$ and L.

Problem 1: Given a complete bipartite graph $K_{n,m}$ and lists of colors $L: V(K_{n,m}) \to 2^{I_k}$ with $|L(v)| \leq 3$. Find a list coloring β for it.

3-SAT Problem: Given a collection $C = \{c_1, \dots, c_m\}$ of clauses on a set $X = \{x_1, \dots, x_n\}$ of Boolean variables such that $|c_i| = 3$ for $i = 1, \dots, m$. Is there a true assignment for X that satisfies all the clauses in C.

Theorem 1: The Problem 1 of finding a list coloring of complete bipartite graphs where lists can have at most 3 colors is NP-complete.

Proof. We will reduce the 3-SAT problem to the Problem 1. Assume we have an instance of 3-SAT with n variables X and m clauses C. We must construct a complete bipartite graph and provide lists of colors on it such that there is a list coloring on that graph if and only if there is a satisfying true assignment for X. Let x^σ be defined the following way: $x^\sigma = x$ when $\sigma = 1$ and $x^\sigma = \neg x$ when $\sigma = 0$. In that case each clause has this form $c_i = (x_{i1}^{\sigma_{i1}} \vee x_{i2}^{\sigma_{i2}} \vee x_{i3}^{\sigma_{i3}})$ where $x_{i1}, x_{i2}, x_{i3} \in X$ and $\sigma_{i1}, \sigma_{i2}, \sigma_{i3} \in \{0,1\}$.

We will construct the graph $K_{n,m}$ the following way: for each variable x_i for $i = 1, \dots, n$ we will construct a vertex u_i in V_1 ($|V_1| = n$) and for each clause c_i for $i = 1, \dots, m$ we will construct a vertex $v_i \in V_2$ ($|V_2| = m$). For every vertex u_i and a vertex v_j there is an edge $(u_i, v_j) \in K_{n,m}$. For every variable x_i we will introduce two colors. The color i will represent x_i and the color $n + i$ will represent $\neg x_i$. Hence $k = 2 * n$ and we will construct the lists the following way: for every vertex u_i the list $L(u_i) = \{i, n + i\}$. For every vertex v_i we will take its clause $c_i = (x_{i1}^{\sigma_{i1}} \vee x_{i2}^{\sigma_{i2}} \vee x_{i3}^{\sigma_{i3}})$ and the colors will be $L(v_i) = \{i1 + (1 - \sigma_{i1}) * n,\ i2 + (1 - \sigma_{i2}) * n, i2 + (1 - \sigma_{i2}) * n\}$ which means, if the variable x_j appears in the form of x_j in c_i then we have the color j in $L(v_i)$ and if the variable x_j appears in the form of $\neg x_j$ then we have the color $j + n$ in $L(v_i)$.

Now let us show that finding a list coloring on this $K_{n,m}, L$ is equivalent to finding a satisfying true assignment in C, X.

Let us first show that if $a_1, \dots, a_n \in \{false, true\}$ is a satisfying solution for C, X such that assigning a_j to x_j makes all $c_i = true$ then there is a list coloring $\beta: V(K_{n,m}) \to \{1, \dots, 2 * n\}$.

For the vertices in V_1 let $\beta(u_j) = j$ if $a_j = false$ and $\beta(u_j) = n + j$ if $a_j = true$. Now for each c_i one of the $a_{i1}^{\sigma_{i1}}, a_{i2}^{\sigma_{i2}}, a_{i3}^{\sigma_{i3}}$ is $true$. Suppose $a_{i1}^{\sigma_{i1}} = true$. If $\sigma_{i1} = 1$ then $a_{i1} = true$ and we will take $\beta(v_i) = i1$ and if $\sigma_{i1} = 0$ then $a_{i1} = false$ and we will take $\beta(v_i) = i1 + n$. Note that from this coloring the only conflict that can happen for the vertex v_i is the vertex u_{i1} because we either color it with $i1$ or $i1 + n$ and in V_1 only the vertex u_{i1} can have these colors. But we color the vertex v_i with $i1$ when $a_{i1} = true$ in which case the color of the vertex u_{i1} is $i1 + n$, and we color the vertex v_i with $i1 + n$ when $a_{i1} = false$ in which case the color of the vertex u_{i1} is $i1$. In other words, we color the vertices of V_2 with the colors for which they become true in the assignment and we color the vertices of V_1 with the opposite colors. This means that any satisfying assignment in C, X is also producing a list coloring in $K_{n,m}, L$.

Now suppose we have a list coloring β in $K_{n,m}, L$, let us show that there is a satisfying solution for C, X. For each vertex in u_i we either have $\beta(u_i) = i$ or $\beta(u_i) = n + i$. Let us construct the assignment $a_1, \dots, a_n$ the following way: $a_i = true$ if $\beta(u_i) = n + i$ and $a_i = false$ if $\beta(u_i) = i$. We now want to show that taking $x_j = a_j$ is a satisfying solution for every c_i clause. $c_i = (x_{i1}^{\sigma_{i1}} \vee x_{i2}^{\sigma_{i2}} \vee x_{i3}^{\sigma_{i3}})$ and we need to show that one of the $a_{i1}^{\sigma_{i1}}, a_{i2}^{\sigma_{i2}}, a_{i3}^{\sigma_{i3}}$ is $true$. Without loose of generality assume the color of the vertex v_i is $i1 + (1 - \sigma_{i1}) * n$. If $\sigma_{i1} = 1$ it means $\beta(v_i) = i1$ which means $\beta(u_{i1}) = i1 + n$ (because the colors should be different), which means $a_{i1} = true$ resulting $a_{i1}^{\sigma_{i1}} = true$ and hence $c_i = true$. If $\sigma_{i1} = 0$ it means $\beta(v_i) = i1 + n$ which means $\beta(u_{i1}) = i1$, which

means $a_{i1} = false$ resulting $a_{i1}^{\sigma_{i1}} = true$ and hence $c_i = true$. This means that any list coloring in $K_{n,m}, L$ is also producing a satisfying assignment in C, X proving that the two problems are equivalent, which means the Problem 1 is NP-complete.

A polynomial algorithm for the list coloring of block graphs.

For a given block graph G let $N = |V(G)|$ and there are restrictions L on the vertices such that for each vertex v the restriction $L(v)$ is a set of colors that is allowed to use for the vertex v and $L(v) \subseteq I_k$ (for a given k). We need to find a vertex coloring β such that for each vertex the restriction is met $\beta(v) \in L(v)$.

Problem: Given an arbitrary block graph G with $N = |V(G)|$ vertices and given arbitrary restrictions L for every vertex v with $L(v) \subseteq I_k$. Determine whether it is possible to have a vertex coloring $\beta: V(G) \to I_k$ such that $\beta(v) \in L(v)$ for every vertex v.

For the block graph G let us construct its respective block-cut tree and denote it as T. Each vertex in T is either a cut vertex or a block of the graph G. Fig 4 illustrate a block graph and its respective block-cut tree. Vertices 2, 8, 13, 14, 15 in the graph G are respectively the vertices c_1, c_2, c_3, c_4, c_5 in the block-cut tree T. In the block-cut tree T if the vertex is a cut vertex in the graph G we will call it a cut vertex in T and if the vertex is a block of the graph G we will call it a block vertex in the tree T. We will draw cut vertices with circles and block vertices with squares. If the vertex v is a block vertex in the tree T then let $B(v)$ be the block of the graph G associated with that block vertex. Since G is a block graph $B(v)$ will be a clique. For a cut vertex v in the tree T let $C(v)$ be the cut vertex in the graph G.

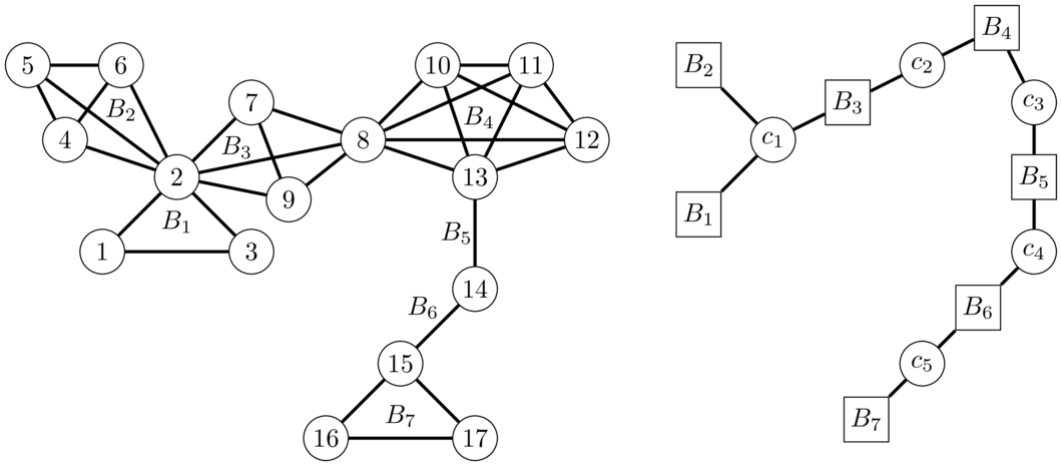

Fig 4. Block graph on the left and its respective block cut tree on the right.

All the leaf vertices in T ($d_T(v) = 1$) are block vertices since a cut vertex is always connected to at least 2 different block vertices in the tree (because removing a cut vertex of the graph G will make the graph disconnected).

If the block graph does not contain a cut vertex, then it is a complete graph. Let us first see how we can solve the list coloring problem in the case of complete graphs. For every vertex from $v_1, \dots, v_N$ we need to select a color from $1, \dots, k$ that meets the restriction $\beta(v_i) \in L(v_i)$. We will construct a bipartite graph the following way: the first partition V_1 will be $\{v_1, \dots, v_N\}$ and the second partition V_2 will be $\{u_1, \dots, u_k\}$. In this bipartite graph we will construct an edge (v_i, u_j) if the color $j \in L(v_i)$. In that case finding a list coloring is equivalent to finding a matching of size N in this bipartite graph because we need to assign a color to each vertex in a way that all the colors are different and the restrictions are met. Fig 5 illustrates that bipartite graph.

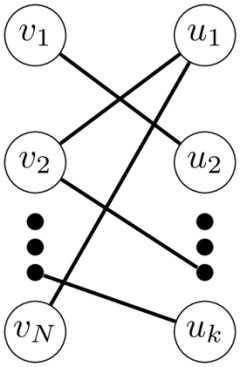

Fig 5. The bipartite graph constructed from the complete graph with provided list of colors.

Moving forward we will assume that there is at least one cut vertex in the graph G. Let r be an arbitrary cut vertex in T which will also be a cut vertex in the graph G. We are interested in the rooted tree T_r . In that case T_r would look like the tree shown in Fig. 6. Since r is a cut vertex the children of the vertex r are block vertices.

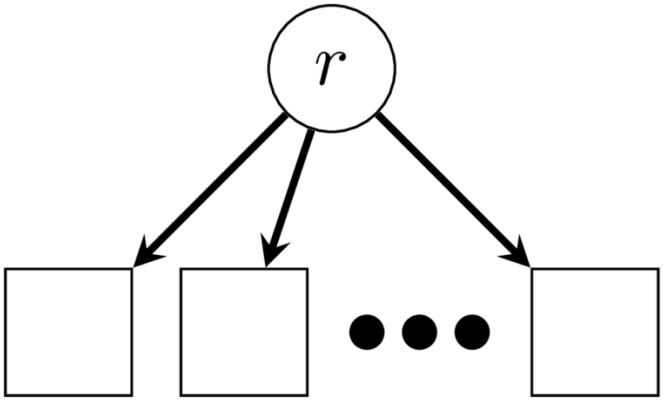

Fig 6. The rooted block-cut tree T_r

For each vertex v in the tree T_r let $U(v)$ be the subtree $ST(v)$ if the vertex v is a cut vertex and the subtree induced by the subset $V(ST(v)) \cup \{p(v)\}$ of the vertices in T_r if the vertex v is a block vertex. Fig. 7 illustrates the subtree $U(v_3)$ and the subtree $U(v_4)$ in T_{v_1}.

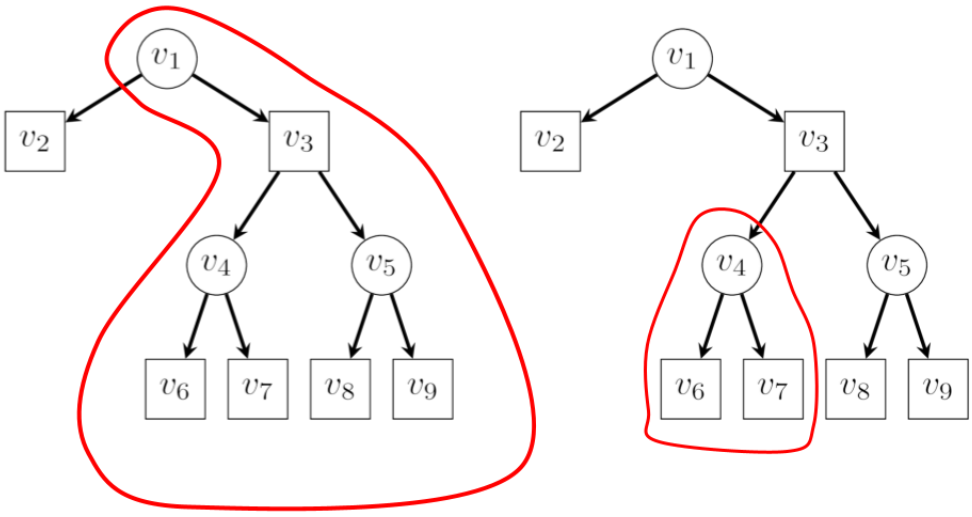

Fig 7. The subtree $U(v_3)$ on the left and the subtree $U(v_4)$ on the right in the tree T_{v_1}.

For a subtree $U(v)$ let $root(U(v))$ be the vertex v if it is a cut vertex, and the vertex $p(v)$ if it is a block vertex. For a block vertex v the vertex $p(v)$ is a cut vertex and $C(p(v))$ is included in the block $B(v)$, $C(p(v)) \in B(v)$. For a non-root cut vertex v in T_r the vertex $p(v)$ is a block vertex and $C(v) \in B(p(v))$. For every v the vertex $root\big(U(v)\big)$ is a cut vertex that is connected to some child block vertices.

For every vertex $v \in T_r$ let $SG(v)$ be the subgraph of the block graph G induced by all the vertices that are included in any of the block vertices of $U(v)$ in T_r, i.e. if u is a block vertex in T_r and $u \in V(U(v))$ then $V(B(u)) \subseteq V(SG(v))$. Let $dp[v][c] = 1$ if it is possible to have a list coloring in the subgraph $SG(v)$ in a way that the vertex $C(root(U(v))$ has the color c. Since the vertex $root(U(v))$ is a cut vertex we want to color it with the color c and have a vertex coloring in $SG(v)$. If it is impossible to have such list coloring then $dp[v][c] = 0$. Here c goes from 1 to k.

In order to calculate the values of $dp[v][c]$ we will need to calculate these values for the children $u_1, \dots, u_m$ of the vertex v in T_r. Suppose we already calculated the values for the child vertices, how can we use those values to calculate $dp[v][c]$? We will consider two cases: the vertex v is a cut vertex and the vertex v is a block vertex.

If the vertex v is a cut vertex, then the vertices $u_1, \dots, u_m$ are block vertices and for subtrees $U(u_i)$ we have $root\big(U(u_i)\big) = v$. Which means they all have the common vertex v. In this case $root\big(U(v)\big) = v$ too, so we essentially want to know for which colors c if it is possible to have a list coloring of the subgraph $SG(v)$. In this case $dp[v][c] = 1$ if and only if $dp[u_i][c] = 1$ for all $i = 1, \dots, m$ since $root\big(U(u_i)\big) = v$ and hence the color of the $root(U(u_i))$ should be the same color c. Fig 8 illustrates this case.

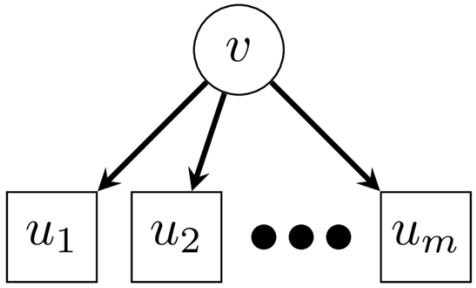

Fig 8. The subtree $U(v)$ when the vertex v is a cut vertex.

If the vertex v is a block vertex it means that the child vertices $u_1, \dots, u_m$ are cut vertices and the vertex $root\big(U(v)\big) = p(v)$ is also a cut vertex. In other words, the vertices $C\big(p(v)\big), C(u_1), \dots, C(u_m) \in B(v)$. Fig 9 illustrates this case.

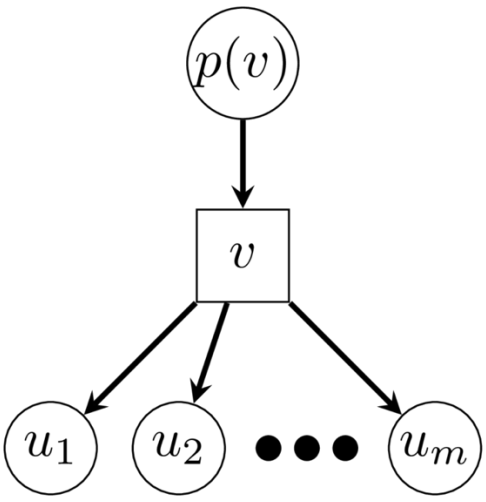

Fig 9. The subtree $U(v)$ when the vertex v is a block vertex.

Consider the block $B(v)$, since it is a clique, we should make sure that all the vertices in this block have different colors. Let $t = |V(B(v))| - 1$ and let the vertices of $B(v)$ be $\{g_0, \dots, g_t\}$ such that $g_0 = C(p(v)), g_1 = C(u_1), \dots, g_m = C(u_m)$. The vertices $g_0, g_1, \dots, g_m$ are cut vertices and the vertices $g_{m+1}, \dots, g_t$ are not cut vertices in the block graph G. In order to find a list coloring of $SG(v)$ we need to find a list coloring for the block $B(v)$ and also for all the subgraphs $SG(u_1), \dots, SG(u_m)$. This means we need to find distinct colors $c_0, \dots, c_t$ such that $c_i \in L(g_i)$ for all $0 \le i \le t$ and $dp[u_j][c_j] = 1$ for all the child vertices ($1 \le j \le m$). If we are calculating the value $dp[v][c]$ it means that $c_0 = c$. We will construct a bipartite graph the following way: the left partition will be V_1 and will have the vertices $\{g_0, \dots, g_t\}$ and the right partition will be V_2 with the vertices $u_1, \dots, u_k$. To calculate the answer for the color $c \in L(g_0)$ we will construct the edges of the bipartite graph the following way: for the vertex g_0 we will only add the edge (g_0, u_c), for the vertices $g_1, \dots, g_m$ we will add an edge (g_a, u_b) if $dp[g_a][b] = 1$, for the vertices $g_{m+1}, \dots, g_t$ we will add an edge (g_a, u_b) if $b \in L(g_a)$. Finding a maximal matching of size $t + 1$ means assigning different colors to the vertices $g_0, \dots, g_t$ such that the restrictions $L(g_j)$ are satisfied, and it is possible to color the subgraphs $SG(u_i)$ for all the child cut vertices. If we find such matching, we assign $dp[v][c] = 1$ otherwise we assign $dp[v][c] = 0$. Fig 10 illustrates the bipartite graph for this case.

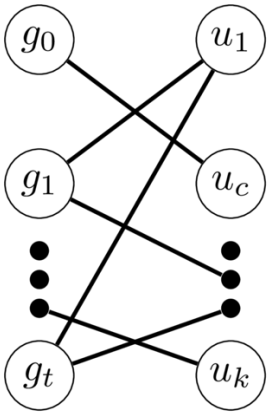

Fig 10. The bipartite graph for a block with vertices $g_0, \dots, g_t$.

We can calculate the values of $dp[v][i]$ in the tree T_r from the bottom to the top. If there is a color i such that $dp[r][i] = 1$ it means we were able to find a list coloring for $SG(r)$ which is the entire graph G. Storing the results of maximal matchings will allow us to later construct the list coloring from top to bottom.

Let us now calculate the complexity of the algorithm. For every block of the graph G we would need to find a maximal matching. If $N = |V(G)|$ and we want to color with the colors from $1, \dots, k$ then for a block of size t we would need to calculate a matching for a bipartite graph that has t vertices on the left partition and k vertices on the right partition which can be done in $O(t * t * k)$ for every color i. Since the sum of the number of vertices from all the blocks is $O(N)$ then it would take about $O(N^2 * k^2)$ operations. If $k = O(N)$ then the algorithm will run in $O(N^4)$. Note that we can always assume that $k = |\bigcup_{v \in V(G)} L(v)| \le \sum_{v \in V(G)} |L(v)|$ since we can remove the redundant colors and reindex the colors. This means that k is less than the size of the input and hence the algorithm has a polynomial time complexity.

REFERENCES

1. West D.B. Introduction to Graph Theory. Prentice-Hall, New Jersey, 1996.
2. Harary F. Graph Theory. *Addison-Wesley Publishing Company*, Boston, 1969.
3. P. Erdős, A.L. Rubin, H. Taylor, Choosability in graphs, Proc. West Coast Conf. on Com- binatorics, Graph Theory and Computing, Congr. Numer. 26, 1979, pp. 125-157.
4. Vizing, V. G. (1976), "Vertex colorings with given colors", Metody Diskret. Analiz. (In Russian), 29: 3–10
5. Hougardy, S. (2006). Classes of perfect graphs. Discret. Math., 306, 2529-2571.

6. Jansen, K., & Scheffler, P. (1997). Generalized coloring for tree-like graphs. *Discrete Applied Mathematics*, *75*(2), 135–155. https://doi.org/10.1016/s0166-218x(96)00085-6
7. Biró, M., Hujter, M. & Tuza, Z. (1992). Precoloring extension. I. Interval graphs. Discrete Mathematics, 100, 267-279.
8. Marx, D. (2006). Precoloring extension on unit interval graphs. Discrete Applied Mathematics, 154, 995–1002.
9. Jansen, K. (1997). The Optimum Cost Chromatic Partition Problem. In G. C. Bongiovanni, D. P. Bovet & G. D. Battista (eds.), CIAC (p./pp. 25-36),: Springer. ISBN: 3-540-62592-5
10. Bonomo, F., Durán, G. & Marenco, J. (2009). Exploring the complexity boundary between coloring and list-coloring. Ann. Oper. Res., 169, 3-16.
11. Karp, R. (1972). Reducibility among combinatorial problems. In R. Miller & J. Thatcher (ed.), Complexity of Computer Computations (pp. 85-103). Plenum Press.
12. Cook, S. A. (1971). The complexity of theorem proving procedures. Proceedings of the Third Annual ACM Symposium (p./pp. 151--158), New York.
13. Kubale M. Interval vertex-coloring of a graph with forbidden colors. Discret. Math., 74, (1989) 125-136.