A BRANCH AND BOUND METHOD FOR MIXED GRAPH COLORING AND SCHEDULING

George V. Andreev, Yuri N. Sotskov
Institute of Engineering Cybernetics
220012 Minsk / Belarus
email: { andreev, sotskov } @minsk.bas-net.by

Frank Werner
Otto-von-Guericke-University, PSF 4120
39016 Magdeburg / Germany
email: frank.werner@mathematik.uni-magdeburg.de

ABSTRACT

A mixed graph coloring is an assignment of positive integers (colors) to vertices of a mixed graph such that, if two vertices are joined by an edge, then their colors have to be different and if two vertices are joined by an arc, then the color of the start vertex has to be not greater than the color of the end vertex. We develop a branch and bound algorithm for determining an optimal mixed graph coloring (i.e. such a coloring of the mixed graph that the number of colors used is minimal), based on the conflict resolution strategy and adding some arcs in the mixed graph. We describe the main components of this algorithm: branching strategy, lower and upper bounds on the chromatic number, etc. Computational results for randomly generated mixed graphs of order $n \leq 150$ are given.

KEYWORDS: Vertex Coloring, Mixed Graph, Branch and Bound, Scheduling.

PROBLEM FORMULATION AND PREVIOUS RESULTS

One of the main restrictions for the application of contemporary scheduling theory is that it concentrates on the minimization of a function of the job completion times and does not take into account the cost of using a processing system. Recently in Krüger et al. (1998) and Sotskov (1996), the minimization of a function of both the number of machines used and the job completion times has been considered. The latter problem induces a mixed graph coloring for testing the feasibility of the assignment of jobs to machines and for calculating the value of the objective function. Other scheduling problems which require a mixed graph coloring are e.g. unit time scheduling problems subject to both precedence and disjunctive constraints or some types of lecture and exam scheduling (see e.g. Hansen et al. (1997), Sotskov and Tanaev (1976) and Sotskov et al. (1998)).

Let $G = (V, A, E)$ be a finite mixed pseudograph with the set of vertices $V = \{v_1, v_2, \ldots, v_n\}$, the set of arcs A and the set of edges E. It is possible that G may contain loops (both orientend and non-oriented), but we assume that no multiple arcs and multiple edges occur.

Definition 1 *Function ϕ is called a coloring of the mixed pseudograph $G = (V, A, E)$, if it defines for each vertex $v_i \in V$ a color (natural number) $\phi(v_i) \in N$ such that from inclusion $(v_i, v_j) \in A$ inequality $\phi(v_i) \leq \phi(v_j)$ follows and inclusion $[v_p, v_q] \in E$ implies $\phi(v_p) \neq \phi(v_q)$.*

If $A = \emptyset$, then function $\phi : V \to N$ represents the usual coloring of graph $(V, \emptyset, E)$ (note that for a pseudograph $(V, \emptyset, E)$ which contains a loop $[v_i, v_i] \in E$, there does not exist a coloring ϕ). Whereas vertex coloring problems of a graph $(V, \emptyset, E)$ were intensively investigated in the literature, such problems for a mixed graph have not found large attention so far. In Sotskov and Tanaev (1976) and Sotskov et al. (1998), the chromatic polynomial of a mixed graph was investigated and the following criteria for the existence of a coloring were found.

Theorem 1 *There exists a coloring ϕ for the mixed graph $G = (V, A, E)$ if and only if pseudograph $(V, \emptyset, E)$ does not have loops and the directed pseudograph $(V, A, \emptyset)$ does not contain any circuit with adjacent vertices in the pseudograph $(V, \emptyset, E)$.*

In the following we will call a pseudograph simply a graph. The determination of the chromatic number of a graph G (i.e. the determination of the smallest number of colors in a coloring ϕ) is discussed in Hansen et al. (1997) and Sotskov et al. (1998). In Hansen et al. (1997), the coloring of a mixed graph has been considered for which the following condition is satisfied.

Condition 1: *If $(v_i, v_j) \in A$, then $[v_i, v_j] \in E$.*

In Hansen et al. (1997), an algorithm of complexity $O(n^2)$ was given for determining a vertex coloring with minimal number of colors for a mixed tree. In Hansen et al. (1997), Klimova and Sotskov (1983) and Sotskov et al. (1998), algorithms for coloring the vertices of a mixed graph with a smallest number of colors were given and evaluated as to their efficiency.

CONFLICT EDGES

The smallest number of colors in a feasible coloring of the mixed graph $G = (V, A, E)$ is called *chromatic number* and denoted by $\gamma(G)$. A coloring $\phi : V \to \{1, 2, \ldots, \gamma(G)\}$ is called *optimal*. If $E = \emptyset$, then function $\phi(v_i) = 1, v_i \in V$, is obviously an optimal coloring of digraph $G = (V, A, \emptyset)$ and consequently, we obtain $\gamma(V, A, \emptyset) = 1$. If the following condition 2 is satisfied, the corresponding graph can be optimally colored in polynomial time.

Condition 2: *If $[v_i, v_j] \in E$, then $(v_i, v_j) \in A$ or $(v_j, v_i) \in A$.*

Indeed, if condition 2 is satisfied, then we assign weight 0 to each arc $(v_i, v_j) \in A$ if $[v_i, v_j] \notin E$, and weight 1 if $[v_i, v_j] \in E$. By the critical path method we determine the earliest times $r_i(G)$ for all vertices $v_i \in V$ in $O(|A|)$ time with respect to the mixed graph G. It is easy to see that function $\phi(v_i) = r_i(G^A) + 1, v_i \in V$, defines an optimal coloring of the mixed graph G provided that condition 2 is satisfied. Thus, the value $\gamma(G) - 1$ is equal to the length $l(G)$ of a critical (i.e. maximal) path in G: $\gamma(G) = l(G) + 1 = \max\{r_i(G) | v_i \in V\} + 1$.

Assume now that condition 2 is not satisfied for the mixed graph G. We denote by $G^A = (V, A, E \setminus E^A)$ the mixed graph obtained from G by deleting a subset $E^A \subseteq E$ of all edges $[v_i, v_j]$ for which the vertices v_i and v_j are not connected in digraph $(V, A, \emptyset)$. We will denote the edges of set E^A as *free* edges. Then we get the following lower bound for value $\gamma(G)$:

$$\gamma(G) \geq l(G^A) + 1 \tag{1}$$

since function $\phi(v_i) = r_i(G^A) + 1, v_i \in V$, is a coloring of the mixed graph G^A (in the following, we will denote this coloring as *early coloring*) which is the subgraph of mixed graph G. According to Definition 1, one can easily see that inequality (1) turns into equality if for each edge $[v_i, v_j] \in E^A$ the equality

$$r_i(G^A) = r_j(G^A) \tag{2}$$

does not hold. An edge $[v_i, v_j] \in E^A$ for which equality (2) holds will be called *conflict edge for the early coloring*. In addition to an early coloring we can determine 'in an opposite way' by the critical path method a late coloring: $\phi(v_i) = p_i(G^A) + 1, v_i \in V$, where $p_i(G^A)$ is the latest time for vertex $v_i \in V$ with respect to the mixed graph G^A. Analogously, edge $[v_i, v_j] \in E^A$ is a conflict edge for a late coloring if

$$p_i(G^A) = p_j(G^A) \tag{3}$$

The edge $[v_i, v_j] \in E^A$ is called a *strong conflict edge*, if it is a conflict edge and both vertices v_i and v_j belong to a critical path of the mixed graph G^A. Obviously, for a strong conflict edge equalities

$$r_i(G^A) = r_j(G^A) = p_i(G^A) = p_j(G^A) \tag{4}$$

must hold. Then one can easily prove the following claim.

Theorem 2 *For $\gamma(G) = l(G^A) + 1$, it is necessary that in the mixed graph G there do not exist stong conflict edges, and it is sufficient that for an early or a late coloring there do not exist conflict edges.*

It follows from Theorem 2 that, if a strong conflict edge exists, then $\gamma(G) > \gamma(G^A)$. On the other hand, if there is no conflict edge for an early (resp. late) coloring, then the early (resp. late) coloring of the mixed graph G^A is also a coloring of the mixed graph G, i.e. equality $\gamma(G) = \gamma(G^A)$ holds.

In order to obtain an upper bound for $\gamma(G)$, we add in the mixed graph G for each free edge $[v_i, v_j]$ one of the arcs (v_i, v_j) or (v_j, v_i). If the resulting mixed graph $G_k^E =$

$(V, A \cup A^E_k, E)$ does not contain circuits, then we will denote this mixed graph as *extension* of the mixed graph G. If G^E_k is an extension of the mixed graph G, then inequality

$$\gamma(G) \leq l(G^E_k) + 1 \tag{5}$$

holds and therefore equality $\gamma(G) = \min\{\gamma(G^E_1), \gamma(G^E_2), \ldots, \gamma(G^E_\lambda)\}$ holds, where $\{G^E_1, G^E_2, \ldots, G^E_\lambda\}$, $\lambda \leq 2^{|E|}$, is the set of all extensions of the mixed graph G.

THE BRANCH AND BOUND ALGORITHM

The suggested branch and bound algorithm denoted as Algorithm B&B determines an optimal coloring of an arbitrary mixed graph G by using the lower and upper bounds (1) and (5) for the chromatic number $\gamma(G)$ and a branching procedure (the set of colorings to be considered successively is partitioned into two subsets) with the aim of solving conflicts (2), (3) or (4). As a result of the application of Algorithm B&B we obtain a search tree L, where each vertex is a mixed graph $(V, A \cup A', E), 0 \leq |A'| \leq |E^A|$. The root of the search tree is the mixed graph (V, A, E), and in each vertex $(V, A \cup A', E)$ which is not a sink, there are two immediate successors: vertex $(V, A \cup A' \cup \{(v_i, v_j)\}, E)$ and vertex $V, A \cup A' \cup \{(v_j, v_i)\}, E)$, which are determined by the selection of a conflict edge $[v_i, v_j] \in E^A$.

We apply a heuristic greedy algorithm (see Algorithm A&AE from Sotskov et al. (1998) with the complexity $O(n^2)$) of coloring the vertices of set V successively in the order $(v^{(1)}, v^{(2)}, \ldots, v^{(n)})$ by the smallest possible color: first vertex $v^{(1)} \in V$ is colored with color 1, then vertex $v^{(2)}$ is colored either with color 1 or color 2 depending on whether edge $[v^{(1)}, v^{(2)}]$ belongs to set E or not and so on, i.e. vertex $v^{(k)}, 2 < k \leq n$, is colored with the smallest possible color in dependence on the already colored vertices $v^{(1)}, v^{(2)}, \ldots, v^{(k-1)}$. To guarantee that such a procedure leads to a coloring ϕ of the mixed graph G, it is sufficient that the linear order $v^{(1)}, v^{(2)}, \ldots, v^{(n)}$ does not contradict to the partial order given on set V by the set of arcs $A \cup A'$, i.e. from inclusion $(v^{(i)}, v^{(j)}) \in A \cup A'$ inequality $i < j$ follows. We assume that the numbering of the vertices $v_1, v_2, \ldots, v_n$ does not contradict to the partial order given on the set of arcs A. Therefore, for the mixed graph $(V, A \cup \{(v_i, v_j)\}, E)$, the initial order $v_1, v_2, \ldots, v_n$ does not contradict to the partial order given on the set of arcs $A \cup \{(v_i, v_j)\}$. Thus, in a preprocessing step of Algorithm B&B, a list of immediate predecessors and immediate successors is built for each vertex $v_i \in V$. For each edge $[v_p, v_q] \in E$, for which a path from vertex v_p to vertex v_q exists, we add arc (v_p, v_q) which allows to exclude obvious non-conflict edges from the considerations. When choosing a conflict edge for branching next we prefer a strong conflict edge, then an edge which is a conflict edge both in the early and the late coloring (notice that these arcs are not necessarily strong conflict edges), then an edge which is only in the early coloring a conflict edge, and finally an edge which is only in the late coloring a conflict edge. It is easy to see that, if edge $[v_i, v_j]$ is a conflict edge in the circuit-free mixed graph $G' = (V, A \cup A', E)$, i.e. if one of the equalities (2) or (3) or equalities (4) is satisfied, then none of the digraphs $(V, A \cup A' \cup \{(v_i, v_j)\}, \emptyset)$ and $(V, A \cup A' \cup \{(v_j, v_i)\}, \emptyset)$ contains a circuit. Thus, in

Algorithm B&B the use of a procedure for checking whether a circuit occurs in the generated mixed graph is not necessary.

A substantial reduction of the search tree can be obtained due to introducing an additional arc $(v^{(j)}, v^{(i)})$ (or an arc $(v^{(i)}, v^{(j)})$ into the current graph $G' = (V, A \cup A', E)$, if inequality

$$r_i(G') + l(G') - p_j(G') + 1 \leq l(G^E_{k^*}) \qquad (6)$$

(resp. inequality

$$r_j(G') + l(G') - p_i(G') + 1 \geq l(G^E_{k^*})). \qquad (7)$$

is satisfied. Here $G^E_{k^*}$ denotes the extension of the mixed graph G which defines the best coloring of the mixed graph G being built when the mixed graph G' is considered. Indeed, let e.g. inequality (6) be satisfied, then the mixed graph $(V, A \cup A' \cup \{v^{(i)}, v^{(j)}\}, E)$ contains a path with a weight not smaller than $l(G^E_{k^*})$. Therefore for a coloring of a mixed graph generated from G in the search tree at least $l(G^E_{k^*}) + 1$ colors is required. However, there is already a coloring of mixed graph G with exactly $l(G^E_{k^*}) + 1$ colors.

If for the mixed graph G' both inequalities (6) and (7) are satisfied, then it is not necessary to construct a subtree of the search tree from the root G' since this construction cannot improve the record coloring already constructed. In other words, if (6) and (7) are both satisfied, one can stop with branching in the search tree L from vertex G' when applying Algorithm B&B, i.e. such vertex G' is a terminal vertex in the final search tree L.

We can also stop with branching vertex G' if one of the following cases is satisfied: There is no conflict edge in an early coloring, there is no conflict edge in a late coloring, or inequality

$$l(G') \geq l(G^E_{k^*}) \qquad (8)$$

is satisfied.

COMPUTATIONAL RESULTS

Algorithm B&B has been implemented in FORTRAN and run on a PC 486 (120 MHz) for (pseudo)randomly generated mixed graphs. Analogously to Hansen et al. (1997) and Sotskov et al. (1998), we generated a mixed graph G with $|A|$ arcs (and $|E|$ edges) by choosing randomly generated numbers τ_l (resp. numbers $\tau_{(l)}$) from the set $\{1, 2, \ldots, k\}$, where $k = \binom{n}{2}, \tau_1 = (v_1, v_2), \tau_2 = (v_1, v_3), \ldots, \tau_k = (v_{n-1}, v_n)$ and for each arc $\tau_j = (v_r, v_s)$ inequality $r < s$ is satisfied. As a result, we obtain a mixed graph G of order n with the set of arcs $A = \{\tau_1, \tau_2, \ldots, \tau_{|A|}\}$ and the set of edges $E = \{\tau_{(1)}, \tau_{(2)}, \ldots, \tau_{|E|}\}$. Taking into account the limitation of internal memory of the computer, two versions of Algorithm B&B have been implemented: a basic version, where the whole search tree L is treated by the internal memory, and an auxiliary version, where only the path from root G up to the currently considered vertex G' is in the internal memory (the whole search tree which is necessary for the reconstruction of all arcs and edges in the search tree is in the external memory on a hard disk). In the basic version, the external memory on the hard disk is not used. Therefore, if during

Table 1: Mixed graphs of order 70

edge density											
0.1				0.2				0.1			
arc de.	$\|L\|$	γ	CPU	arc de.	$\|L\|$	γ	CPU	arc de.	$\|L\|$	γ	CPU
0.1	496	7	16.66	0.1	12736	10	1085.50	0.1	–	–	–
0.2	71	10	2.27	0.2	2119	13	153.10	0.2	9706	17	1280.84
0.3	19	12	0.54	0.3	92	15	5.43	0.3	2298	20	158.94
0.4	17	14	0.53	0.4	37	19	2.17	0.4	149	24	10.04
0.5	5	15	0.20	0.5	13	20	0.57	0.5	58	25	2.71
0.6	4	15	0.16	0.6	7	20	0.34	0.6	34	26	1.47
0.7	5	17	0.19	0.7	6	23	0.27	0.7	4	28	0.26
0.8	2	17	0.09	0.8	2	23	0.14	0.8	3	31	0.20
0.9	1	18	0.06	0.9	2	24	0.10	0.9	2	30	0.14

the use of the basic version the internal memory is not sufficient, then the auxiliary version is applied which uses the hard disk for storing the search tree. Unfortunately, the auxiliary version uses a rule for selecting the mixed graph for branching not so effective than the basic version does.

For each vertex $G' = (V, A \cup A', E)$, both versions of the program calculate a heuristic coloring for the mixed graph G' in $O(n^2)$ time, and an early and late coloring for the mixed graph $(V, A \cup A', E \setminus E^{A \cup A'})$. In the basic version, the number of remaining conflict edges for early and late colorings is calculated. In both versions, the program realized a depth-first search with choosing the sink with the smallest lower bound (1), i.e. beginning with the root vertex $G = (V, A, E)$ of the search tree, it goes down in the search tree and constructs a new best coloring or inequality (8) will be reached. Then the basic version of the program selects the sink vertex with smallest bound (1). If several such vertices in the current search tree exist, then among them the vertex with the smallest number of conflict edges for an early or late coloring is chosen. The depth-first search is continued from the chosen mixed graph. In the auxiliary version of the program, the last sink vertex with the smallest bound (1) is chosen. Since it is in this case sufficient to know the path from the root vertex G to the considered mixed graph, this variant saves internal memory (at the same time, this variant considers substantially more vertices of the search tree in comparison with the basic variant).

The results of the experiments are given in Tables 1 – 3, where each entry represents the results of a series of 10 instances with the same density of arcs and the same density of edges. Column 1 gives the arc density, column 2 gives the average cardinality $|L|$ of the vertex set in the search tree, column 3 gives the average value of the chromatic number $\gamma(G)$, and column 4 gives the average CPU time. If column 4 does not contain some upper index in parantheses, then the average CPU time (in seconds) for the basic version of the program is given, otherwise it gives the average CPU time of the basic

Table 2: Mixed graphs of order 100

edge density							
0.1				0.2			
arc de.	$\|L\|$	γ	CPU	arc de.	$\|L\|$	γ	CPU
0.1	12476	10	2628.43$^{(1)}$	0.1	–	–	–
0.2	1235	13	202.98	0.2	11831	18	3369.61
0.3	95	16	15.28	0.3	3021	23	584.97
0.4	13	19	1.22	0.4	1076	27	231.80
0.5	7	21	1.14	0.5	40	30	5.25
0.6	10	22	0.88	0.6	13	32	1.48
0.7	4	24	0.47	0.7	8	35	1.08
0.8	2	25	0.31	0.8	4	34	0.53
0.9	1	26	0.19	0.9	1	35	0.28

and the auxiliary versions of the program, where the upper index in parantheses gives the number of instances, where the (less efficient) auxiliary version of the program has been used.

Both versions of the program require as input data the list of vertices V, the list of arcs A and the list of edges E of the mixed graph G, where the arcs and edges have to be in lexicographical order of their pairs of vertices (the required time for this ordering never exceeded 1 second for all instances and is not considered in column 4). The notations in Tables 2 and 3 are identical to those in Table 1. The symbol '*' stands for series of mixed graphs, where the input data required more than 64 KB which cannot be handled by the used computer version. The symbol '–' stands for series of mixed graphs, where the solution tree L contained more than 30,000 vertices for the basic program version.

CONCLUSION

From Tables 1 – 3 it can be seen that Algorithm B&B is considerably more effective than Algorithm A&AE&E presented in Sotskov et al. (1998). The new algorithm is clearly superior in the case of large values $\gamma(G)$. Algorithm B&B is also superior to the branch and bound algorithm given in Hansen et al. (1997) independently of the fact that the latter algorithm is only applicable to mixed graphs for which condition 1 is satisfied. We notice that the rather bad results of Algorithm B&B for mixed graphs G, in which the arc density is smaller than the edge density, comes from the fact that in bounds (1) and (5) for the chromatic number the existence of free edges in the set E^A in the mixed graph G is not taken into consideration.

It is obvious that for mixed graphs G with a larger cardinality of the set of free edges than the cardinality of arcs, it is necessary to use in addition to bounds (1) and (5)

Table 3: Mixed graphs of orders 120 and 150

$n = 120$ edge density								$n = 150$ edge density			
0.1				0.2				0.1			
arc de.	$\|L\|$	γ	CPU	arc de.	$\|L\|$	γ	CPU	arc de.	$\|L\|$	γ	CPU
0.1	28330	13	5009.9(8)	0.1	–	–	–	0.1	– –	–	–
0.2	700	16	249.27	0.2	5711	22	2661.51	0.2	10745	20	9659.2(3)
0.3	290	20	100.59	0.3	1961	28	683.56	0.3	1532	25	1139.49
0.4	41	23	7.27	0.4	1562	31	439.60	0.4	314	28	205.23(1)
0.5	25	25	4.77	0.5	220	35	42.63(1)	0.5	50	32	26.73
0.6	11	27	1.45	0.6	25	38	6.50	0.6	9	33	3.43
0.7	4	29	0.95	0.7	8	39	1.99	0.7	*	*	*
0.8	2	27	0.57	0.8	6	43	1.23	0.8	*	*	*
0.9	1	31	0.38	0.9	2	43	0.56	0.9	*	*	*

a corresponding bound for the chromatic number of subgraph $(V, \emptyset, E)$. Additional investigations are necessary in this connection in order to select from the known bounds for the chromatic number of graph (V, A, E) that bound which corresponds in a largest degree to a vertex coloring of the mixed graph (V, A, E).

Acknowledgement: This work has been supported by INTAS (project 96-0820) and ISTC (project B-104-98).

REFERENCES

Hansen, P., Kuplinsky, J. and D. de Werra (1997). *Math. Meth. Oper. Res.*, 45, 145 - 160.

Klimova, L.I. and Y.N. Sotskov (1983). Collect. Sci. Work, ITK AN BSSR: Algorithms and Programms for Solving optimization problems. Minsk, 90 – 96.

Krüger, K., Y.N. Sotskov and F. Werner (1998). Heuristics for generalized shop scheduling problems based on decomposition. *Int. J. Oper. Res.*, 36, N. 11, 3013 - 3033.

Levin, G.M. (1982). On an estimation of the chromatic characteristics of a compound graph (in Russian). *Izv. Akad. Nauk BSSR*, Ser. Fiz.-Mat. Nauk, No.1, 17 – 20.

Sotskov, Y.N. (1996). Mixed multigraph approach to scheduling jobs on machines of different types. *Optimization*, 42, 245 – 280.

Sotskov, Y.N. and V.S. Tanaev (1976). Chromatic polynomial of a mixed graph (in Russian). *Izv. Akad. Nauk BSSR. Ser. Fiz.-Mat. Nauk*, N. 6, 20 – 23.

Sotskov, Y.N., Tanaev, V.S. and F. Werner (1998). Scheduling problems and mixed graph colorings. Otto-von-Guericke-Universität Magdeburg, FMA, *Preprint 38/98*, 1998.