Journal of King Saud University – Engineering Sciences (2015) **27**, 114–118

King Saud University

Journal of King Saud University – Engineering Sciences

www.ksu.edu.sa
www.sciencedirect.com

ORIGINAL ARTICLE

Solving the graph coloring problem via hybrid genetic algorithms

Sidi Mohamed Douiri *, **Souad Elbernoussi**

University Mohammed V, Faculty of Sciences-Agdal, Rabat, Morocco

Received 16 August 2012; accepted 1 April 2013
Available online 3 April 2014

KEYWORDS

Graph coloring problem;
Genetic algorithm;
DBG heuristic

Abstract Let $G = (V,E)$ an undirected graph, V corresponds to the set of vertices and E corresponds to the set of edges, we focus on the graph coloring problem (GCP), which consist to associate a color to each vertex so that two vertices connected do not possess the same color. In this paper we propose a new hybrid genetic algorithm based on a local search heuristic called DBG to give approximate values of $\chi(G)$ for GCP. The proposed algorithm is evaluated on the DIMACS benchmarks and numerical results show that the proposed approach achieves highly competitive results, compared with best existing algorithms.

1. Introduction

The graph coloring problem (GCP) is the combinatorial optimization problem the most studied in computer science and mathematics. It is related to several traditional applications in various fields such as telecommunications, bioinformatics and Internet. Among these applications we find, timetable problem (de Werra, 1985; Burke et al., 1994), crew scheduling (Gamache et al., 2007; Zufferey et al., 2008), supply chain and logistics optimization (Lim and Wang, 2005), air traffic flow management (Barnier and Brisset, 2002) and frequency assignment problem (Gamst, 1986; Park and Lee, 1996). Garey et al. in 1979 demonstrated that the k-coloring problem is NP-complete and that the determination of the chromatic number $\chi(G)$ is NP-hard (Garey and Johnson, 1979). Therefore several methods and heuristics have been proposed to solve this problem. The first used algorithms were constructive algorithms. Among the most employed include RFL (Leighton, 1979) and DSATUR (Brelaz, 1979), both approaches use an order constructed dynamically on the vertices. Subsequently a large number of local search algorithms have been intended to solve the coloring problem. Among these methods the tabu search is in the first place, several authors have introduced this technique in their works (Hertz and de Werra, 1987; Dorne and Hao, 1998; Gonzlez-Velarde and Laguna, 2002). There exist other variants of this local research: reactive partial tabu search given by Blochliger et al. in 2008 (Blochliger and Zufferey, 2008), adaptive memory method (AMM) proposed by Galinier et al. (2002), this latter is only an adaptation algorithm of Galinier and Hao (1999), Chiarandini et al. presented the application of iterated local search (Chiarandini and Stutzle, 2002). The variable neighborhood search method was described in Avanthay et al. (2003), Hamiez and Hao (2002), and finally the variable space search was cited by Hertz

* Corresponding author.
E-mail address: douirisidimohamed@hotmail.fr (S.M. Douiri).

Peer review under responsibility of King Saud University.

http://dx.doi.org/10.1016/j.jksues.2013.04.001
1018-3639 © 2014 Production and hosting by Elsevier B.V. on behalf of King Saud University.

et al. (2008). These local search methods for GCP include also the metaheuris-tic simulated annealing (Chams et al., 1987; Johnson et al., 1991). About the evolutionary algorithms, Davis in 1991 was the first to apply genetic algorithms (GA) for solving GCP (Davis, 1991). Fleurent et al. adapted their GA by replacing the mutation operator by the tabu search of Fleurent and Ferland (1996). Malaguti et al. combined memetic approach and integer linear programing based on the techniques of column generation method (Malaguti and Monaci, 2008). Lu et al. introduced a hybrid metaheuristic incorporating a tabu search with an evolutionary algorithm with effective control of the balance between diversification and intensification (Lu and Hao, 2010). Recently, Wu et al. present a method to extract large independent sets from the graph denoted by EXTRACOL (Wu and Hao, 2012).

This paper presents the resolution of the graph coloring problem by combining a genetic algorithm with a local heuristic (DBG) (Douiri and Elbernoussi, 2011). We test multiple instances of graphs imported from the Dimacs library, and we compare the computational results with the currently best coloring methods, showing that the proposed approach achieves competitive results.

The paper is organized as follows: in Section 2, we present some definitions, Section 3 describes the proposed approach. In Section 4, we show computational results and comparisons. Conclusions are given in the last section.

2. Problem definition

- The graph coloring problem is to assign a color to each vertex so that two adjacent vertices do not have the same color. If the graph contains an edge (x,y), then x and y will have different colors.
- A valid k-coloring of vertices in a graph $G = (V,E)$ is an application $c: V \to \{1,\dots,k\}$ such as $c(x) \neq c(y), \forall (x,y) \in E$, the value $c(x)$ associated with vertex x is called color of x. If $(x,y) \in E$ and $c(x) = c(y)$ we say that x and y are in conflict. The vertices of the same color define a color class, such that there is no edge between two vertices of the same class. Since each color class induces an independent set of G, a coloring can also be seen as a partition of V as independent sets.
- The chromatic number of a graph G denoted $\chi(G)$ is the smallest number of colors used to color all vertices of G with a valid coloring.

3. Proposed approach

We solve a series of k-coloring problems each time when we find a valid coloring. This process takes place as follows: we start from an initial number of colors equal to k, we choose our individuals $p = < c(1), c(2), \dots, c(N) >$ corresponding to an assignment of k colors to all vertices of the graph G. If a valid k-coloring is found the number k is updated $(k \leftarrow 1 - k)$. We repeat this procedure as many times that k-coloring is achieved without conflict and the maximum iteration number is not reached.

The initialization of the number k of colors, is made using the algorithm below which partitions V in k independent sets (k color classes). For this we use the heuristic (DBG) (Douiri and Elbernoussi, 2011) to determine a maximal independent set approximation. This heuristic is based on the reduction of variables through a multiplier w of the surrogate constraint.

Algorithm 1

```
1. While $V \neq \emptyset$.
2. Apply (DBG) algorithm:
   a. Let $w = (1,\dots,1)^t$ and $V' = \emptyset$.
   b. Calculate the surrogate constraint $w^t A = \sum_{w_k \neq 0}(a_{k,.})$.
   c. Give $i = index(\min(w^t A):(w^t A)_i \neq 0)$, let $x_i = 1$ and $V' = V' \cup \{i\}$.
   d. For all $j \in$ Nodestar(i) if $a_{k,j} = 1$, then $w_k = 0$, if $\sum_{k=1}^{m} w_k = 0$ stop.
   Otherwise return to step b.
3. Make $V = V \setminus V'$.
4. If the vertices of $V$ are disjoint stop, otherwise go to 2.
5. End while.
```

3.1. Initial population

The individuals of initial population are generated with a slight modification of the above algorithm. The choice of vertex i is done this time in a random manner to give different configurations and thus diversify the initial population.

Algorithm 2

```
1. $nbrclr = 1$
2. While $V \neq \emptyset$.
3. If $nbrclr > k$, assign a random color chosen in $\{1,\dots,k\}$
   for the rest of vertices.Else
4. Apply (DBG) algorithm
   a. Let $w = (1,\dots,1)^t$ and $V' = \emptyset$.
   b. Calculate the surrogate constraint $w^t A = \sum_{w_k \neq 0}(a_k,.)$.
   c. Randomly select $i = index((w^t A):(w^t A)_i \neq 0)$, let $V' = V' \cup \{i\}$ and $c(i) = nbrclr$.
   d. For all $j \in$ Nodestar(i) if $a_{k,j} = 1$, then $w_k = 0$, if $\sum_{k=1}^{m} w_k = 0$ stop.Otherwise return to step b.
5. Make $V = V \setminus V'$.
6. $nbrclr = nbrclr + 1$
7. End If
8. End While
```

3.2. Objective function

We consider an objective function relating to the number of conflicts for each fixed k, we look during a k-coloring to reduce at each stage the number of conflicts in order to bring it to zero, which corresponds to a valid k-coloring. We note $M(i,j)$ the matrix of conflict and $c(i)$ the associated color of a vertex i.

$$M(i,j) = \begin{cases} 1 \ \mathit{if} \ c(i) = c(j) \ \mathit{and} \ \{i,j\} \in E \\ 0 \ \mathit{otherwise} \end{cases}$$

The fitness corresponds to the following conflicts sum: $f = \sum_{(i,j) \in E} M(i,j)$.

3.3. Selection operator

Every individual will be duplicated in a new population proportionally to its adaptation value, we carried as much draws with replacement that there are of elements in the population. The new probability of each individual to be reintroduced in the new population is:

$$1-\frac{f(x_i)}{\sum_{j=1}^{N} f(x_j)} \tag{3.1}$$

3.4. Crossing operator

We opt for a simple crossing in one point, we choose randomly a point inter-gene b in each parent P_1 and P_2, the crossing is in the following way:

$$C_1(i)=\begin{cases} P_1(i) \text{ } if \ i\in[1,b[\\ P_2(i) \text{ } if \ i\in[b,n] \end{cases}$$

$$C_2(i)=\begin{cases} P_2(i) \text{ } if \ i\in[1,b[\\ P_1(i) \text{ } if \ i\in[b,n] \end{cases}$$

The individuals C_1 and C_2 are not necessarily obtained following the above rule, but in some cases we make the needed changes to obtain individuals satisfying our problem's conditions. If after crossing the color l does not appear in an individual C we proceed as follows:

If the individual C obtained after crossing has no color l with $1\leqslant l\leqslant k$ then for $1\leqslant j\leqslant b$ and $b\leqslant i\leqslant n$ if $C(i)=C(j)$ then $C(j)=l$.

3.5. Mutation operator

Mutation can improve the found coloring and reach solutions in the search space that the crossing could not explore. This is done by replacing a solution p by another improved solution p'. After choosing a random invalid coloring p, we seek on this coloring a vertex i with a color l which represents the most conflicts between all vertices of p, then we change the color l by another different color $l'\in\{1,\ldots,k\}$ and we replace the solution p by p' if $f(p')\leqslant f(p)$.

4. Experimental results

To evaluate the proposed algorithms on the graph coloring problem, we present computational results on the 68 benchmark graphs from the well-known DIMACS graph coloring challenge (http://mat.gsia.cmu.edu/COLOR04/, ftp://dimacs.rutgers.edu/pub/challenge/graph/benchmarks/color/). According to Table 1 where $|V|$ denotes the vertices number, $|E|$ is the number of edges, k^* is the chromatic number or the best known bound of the chromatic number, k is the color number found for each graph using our algorithm, $T_{avg}(s)$ is the average execution time in seconds, each instance has been tested 15 times and the best values k as well as T_{avg} have been preserved, Column 4 shows the success rate (succ).

The used parameters in each method were chosen after several tests on different graphs.

For the genetic algorithm we worked with a population equal to 120, a crossover probability $p_c=0.7$, a mutation probability $p_m=0.15$, and iteration number varies between 250 and 2000.

From Table 1, except the graph dsjc 125.1 we note that the results obtained by our approach are identical to the best known solutions. Generally, except for the six graphs (namely dsjc125.9, dsjc125.9, fpsol2.i.1, fpsol2.i.2, fpsol2.i.3, zeroin.i.2), the execution time has not exceeded one minute for most instances.

In Table 2 we selected difficult graphs because they are the most challenging ones. We compare our approach with the best algorithms given in the literature (Chiarandini and Stutzle, 2002; Blochliger and Zufferey, 2008; Hertz et al., 2008; Malaguti and Monaci, 2008; Lu and Hao, 2010; Wu and Hao, 2012), columns 3 to 8 show their results. Columns 9 indicates the average computation time in seconds needed to find the k-colorings, last column shows the number of successful runs (succ), and for each graph, we executed 20 runs. The best results are shown in bold among the six approaches of the literature.

If we compare our results with those given in Chiarandini and Stutzle (2002) on thirteen tested instances, we observe that

Table 1 Experimental results.

Graph	$\mid V\mid$	$\mid E\mid$	k^*	k_{Ours}	$T_{avg}(s)$	succ
anna	138	493	11	11	11.63	12/15
david	87	406	11	11	10.89	14/15
huck	74	301	11	11	11.10	15/15
2-Inser-3	37	72	4	4	2.23	15/15
3-Inser-3	56	110	4	4	3.37	15/15
jean	80	254	10	10	9.92	15/15
queen5.5	25	160	5	5	1.20	15/15
queen6.6	36	290	7	7	1.32	15/15
queen7.7	49	476	7	7	6.91	15/15
queen8.8	64	728	9	9	9.87	11/15
queen9.9	81	1056	10	10	13.96	13/15
miles250	128	387	8	8	4.39	11/15
miles500	128	1170	20	20	14.48	10/15
games 120	120	638	9	9	10.77	15/15
mug88-1	88	146	4	4	4.05	15/15
mug88-25	88	146	4	4	3.67	13/15
mug 100-1	100	166	4	4	8.34	11/15
mug 100-25	100	166	4	4	8.21	13/15
mycie13	11	20	4	4	0.01	15/15
mycie14	23	71	5	5	0.89	15/15
mycie15	47	236	6	6	5.41	15/15
mycie16	95	755	7	7	12.77	11/15
mycie17	191	2360	8	8	20.19	9/15
dsjc 125.1	125	736	5	6	13.12	14/15
dsjc1 125.5	125	3891	17	17	19.71	8/15
dsjc1 125.9	125	6961	44	44	35.87	7/15
dsjc1250.1	250	3218	8	8	26.07	10/15
dsjc1250.9	250	27897	72	72	75.81	6/15
fpso12.i.1	496	11654	65	65	82.03	2/15
fpso12.i.2	451	8691	30	30	69.79	6/15
fpso12.i.3	425	8688	30	30	67.14	4/15
zeroin.i.1	211	4100	49	49	28.24	5/15
zeroin.i.2	211	3541	30	30	83.39	9/15
zeroin.i.3	206	3540	30	30	27.06	6/15
mulsol.i.1	197	3925	49	49	25.75	2/15
mulsol.i.2	188	3885	31	31	22.07	7/15
mulsol.i.3	184	3916	31	31	23.49	5/15
mulsol.i.4	185	3946	31	31	25.22	4/15
mulsol.i.5	186	3973	31	31	28.33	7/15

Table 2 Experimental results.

Graph	|*V*|	Wu and Hao (2012)	Lu and Hao (2010)	Blochliger and Zufferey (2008)	Hertz et al. (2008)	Malaguti and Monaci (2008)	Chiarandini and Stutzle (2002)	k_{Ours}	$T_{avg}(s)$	succ
dsjc250.5	250	*	**28**	*	*	**28**	**28**	**28**	89	18/20
dsjc500.1	500	*	**12**	**12**	**12**	**12**	**12**	**12**	112	15/20
dsjc500.5	500	*	**48**	**48**	**48**	**48**	49	**48**	147	18/20
dsjc500.9	500	*	**126**	127	**126**	127	**126**	**126**	190	19/20
dsjc 1000.1	1000	**20**	**20**	**20**	**20**	**20**	*	**20**	4982	9/20
dsjc1000.5	1000	**83**	**83**	89	86	**83**	89	**83**	2164	13/20
dsjc1000.9	1000	**222**	223	226	224	224	*	224	8092	7/20
dsjr500.1c	500	*	**85**	**85**	**85**	**85**	*	**85**	581	16/20
dsjr500.5	500	*	**122**	125	125	**122**	124	124	728	13/20
r250.5	250	*	**65**	66	*	**65**	*	**65**	261	17/20
r250.1	250	*	*	*	*	**8**	*	**8**	308	20/20
r250.1c	250	*	*	**64**	*	**64**	*	**64**	266	18/20
r1000.1	1000	*	*	*	*	**20**	*	**20**	895	10/20
r1000.1c	1000	101	**98**	**98**	*	**98**	*	**98**	1329	6/20
r1000.5	1000	249	245	248	*	**234**	*	242	1507	3/20
le450_15*c*	450	*	**15**	**15**	**15**	**15**	**15**	**15**	93	20/20
le450_15*d*	450	*	15	15	15	15	15	15	228	20/20
le450_25*c*	450	*	**25**	**25**	**25**	**25**	26	**25**	740	20/20
le450_25*d*	450	*	26	25	25	25	26	25	382	18/20
flat300_20_0	300	*	*	*	*	**20**	*	**20**	241	20/20
flat300_26_0	300	*	**26**	*	*	**26**	**26**	**26**	275	16/20
flat300_28_0	300	*	29	**28**	**28**	31	31	**28**	319	18/20
flat1000_50_0	1000	**50**	**50**	**50**	**50**	**50**	*	**50**	802	9/20
flat1000_60_0	1000	**60**	**60**	**60**	**60**	**60**	*	**60**	1344	5/20
flat1000_76_0	1000	**82**	**82**	87	85	**82**	*	**82**	3795	7/20
C2000.5	2000	**146**	148	*	*	*	*	151	8421	8/20
C2000.9	2000	**409**	*	*	*	*	*	411	7368	5/20
C4000.5	4000	**260**	272	*	*	*	*	282	212881	2/20
latin_*sqr*_10	900	*	**99**	*	*	101	**99**	**99**	1267	11/20

we have improved five results for graphs (namely dsjc500.5, dsjc1000.5, *le*450_25*c*, *le*450_25*d*, *flat*300_28_0), for the rest we have obtained the same values.

Concerning results obtained by Blochliger and Zufferey (2008) on twenty tested instances, our approach has improved seven instances (namely dsjr500.5, dsjc500.9, dsjc1000.5, dsjc1000.9, r250.5, r1000.5, *flat*1000_76_0), and twelve results remained identical.

When comparing with the results reported in Hertz et al. (2008) on sixteen tested graphs, our algorithm is better on three graphs (namely dsjc1000.5, dsjr500.5, *flat*1000_76_0), we obtain the same values for the rest of instances.

Versus to the results presented in Malaguti and Monaci (2008) on twenty six tested instances, we have improved three results for the graphs (namely dsjc500.9, *flat*300_28_0, *latin_sqr*_10) and worse on two graphs (namely dsjr500.5, r1000.5).

Our algorithm provides better results than (Lu and Hao, 2010) on three graphs (namely r1000.5, *le*450_25*d*, *flat*300_28_0) and worse on three graphs (namely dsjr500.5, C2000.5, C4000.5).

Compared to the EXTRACOL method given by Wu and Hao (2012) tested on eleven graphs, we provide best results for two graphs (namely r1000.1c, r1000.5), five results are similar (namely dsjc1000.1, dsjc1000.5, *flat*1000_50_0, *flat*1000_60_0, *flat*1000_76_0) and worse on four graphs (namely dsjc1000.9, C2000.5, C2000.9, C4000.5).

For reaching the smallest *k*-coloring, majority of the instances have an execution time, not exceeding one hour (between 1min29s and 36min06s), but for some hard graphs, the CPU time was very large (namely dsjc1000.1, dsjc1000.9, *flat*1000_76_0, C2000.5, C2000.9, C4000.5) it ranges between 1h3min15s and 59h7min21s.

For large random graphs (namely dsjc1000.1, dsjc1000.5, r1000.1c, r1000.1, *flat*1000_50_0, *flat*1000_60_0, *flat*1000_76_0), our method can find the previous best known result with computation time not exceeding 2 hours. About the graph C4000.5 it needs an execution time of approximately 60 h, which shows the difficulty degree for its coloring.

5. Conclusion

In this paper we have proposed a hybrid genetic algorithm which combines a genetic algorithm with a local search heuristic (DBG) to solve the graph coloring problem. The computational experiments, carried out on a set of 68 benchmark graphs were imported from the DIMACS library among them twelve largest graphs with 900, 1000, 2000 and 4000 vertices, show that the results obtained by our approach is very competitive compared to the current best known results reported in 6 state-of-art published in the literature.

References

Avanthay, C., Hertz, A., Zufferey, N., 2003. A variable neighbourhood search for graph coloring. Eur. J. Oper. Res. 151 (2), 379–388.

Barnier, N., Brisset, P., 2002. Graph coloring for air traffic flow management. In CPAIOR'02 : Fourth International Workshop on

Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimisation Problems, Le Croisic, France, pp. 133–147.

Blochliger, I., Zufferey, N., 2008. A graph coloring heuristic using partial solutions and a reactive tabu scheme. Comput. Oper. Res. 35 (3), 960–975.

Brelaz, D., 1979. New methods to color the vertices of a graph. Commun. ACM 22 (4), 251–256.

Burke, E.K., Elliman, D.G., Weare, R.F., 1994. A university timetabling system based on graph colouring and constraint manipulation. J. Res. Comput. Educ. 27 (1), 1–18.

Chams, M., Hertz, A., de Werra, D., 1987. Some experiments with simulated annealing for coloring graphs. Eur. J. Oper. Res. 132 (2), 260–266.

Chiarandini, M., Stutzle, T., 2002. An application of iterated local search to graph coloring. In: Johnson, D.S., Mehrotra, A., Trick, M. (Eds.), Proceedings of the Computational Symposium on Graph Coloring and its Generalizations, Ithaca, New York, USA, pp. 112–125.

Davis, L., 1991. Order-based genetic algorithms and the graph coloring problem. In: Handbook of Genetic Algorithms. Van Nostrand Reinhold, New York, pp. 72–90.

de Werra, D., 1985. An introduction to timetabling. Eur. J. Oper. Res. 19 (2), 151–162.

Dorne, R., Hao, J.K., 1998. In: Voss, S., martello, S., osman, I.H., roucairol, C. (Eds.), tabu search for graph coloring, t-colorings and set t-colorings metaheuristics, Adv. Trends Local Search Paradigms Optim., Vol. 117. Kluwer, pp. 77–92.

Douiri, S.M., Elbernoussi, S., 2011. New heuristic for the sum coloring problem, applied mathematical sciences. Appl. Math. Sci. 5 (63), 3121–3129.

Fleurent, C., Ferland, J., 1996. Genetic and hybrid algorithms for graph coloring. Ann. Oper. Res. 63, 437–464.

Galinier, P., Hertz, A., Zufferey, N., 2002. Adaptive Memory Algorithms for Graph Coloring, Proceedings of the Computational Symposium on Graph Coloring and its Generalizations. In: Johnson, D.S., Mehrotra, A., Trick, M. (Eds.). COLOR, Ithaca, USA, pp. 75–82.

Galinier, P., Hao, J.K., 1999. Hybrid evolutionary algorithms for graph coloring. J. Comb. Optim. 3, 379–397.

Gamache, M., Hertz, A., Ouellet, J.O., 2007. A graph coloring model for a feasibility problem in monthly crew scheduling with preferential bidding. Comput. Oper. Res. 34 (8), 2384–2395.

Gamst, A., 1986. Some lower bounds for a class of frequency assignment problems. IEEE Trans. Vehicular Technol. 35, 8–14.

Garey, M.R., Johnson, D.S., 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman and Company, New York, Nantes, France.

Gonzlez-Velarde, J., Laguna, M., 2002. Tabu search with simple ejection chains for coloring graphs. Ann. Oper. Res. 117 (1-4), 165–174.

Hamiez, J.P., Hao, J.K., 2002. Scatter search for graph coloring. Selected papers from AE-01. In: Lecture Notes in Computer Science, vol. 2310. Springer-Verlag, pp. 168–179.

Hertz, A., de Werra, D., 1987. Using tabu search techniques for graph coloring. Computing 39 (4), 345–351.

Hertz, A., Plumettaz, M., Zufferey, N., 2008. Variable space search for graph coloring. Discrete Appl. Math. 156 (13), 2551–2560.

Johnson, D.S., Aragon, C.R., McGeoch, L.A., Schevon, et C., 1991. Optimization by simulated annealing: an experimental evaluation, part ii, graph coloring and number partitioning. Oper. Res. 39(3), 378–406.

Leighton, F.T., 1979. A graph coloring algorithm for large scheduling problems. J. Res. Natl. Bur. Stand. 84 (6), 489–506.

Lim, A., Wang, F., 2005. Robust graph coloring for uncertain supply chain management. In: Proceedings of the 38th Annual Hawaii International Conference on System Sciences (HICSS'05)-Track 3-Volume 03, vol. 11(8), IEEE Computer Society, pp. 263–277.

Lu, Z., Hao, J.K., 2010. A memetic algorithm for graph coloring. Eur. J. Oper. Res. 203 (1), 241–250.

Malaguti, E., Monaci, M., Toth, et P., 2008. A metaheuristic approach for the vertex coloring problem. INFORMS J. Comput. 20(2), 302–316.

Park, T., Lee, C.Y., 1996. Application of the graph coloring algorithm to the frequency assignment problem. J. Oper. Res. Soc. Jpn 39 (2), 258–265.

Wu, Q., Hao, J.K., 2012. An effective heuristic algorithm for sum coloring of graphs. Comput. Oper. Res. 39 (Issue 7), 1593–16001.

Zufferey, N., Amstutz, P., Giaccari, P., 2008. Graph colouring approaches for a satellite range scheduling problem. J. Scheduling 11 (8), 263–277.