PRACE NAUKOWE AKADEMII EKONOMICZNEJ WE WROCŁAWIU
Nr 1064 2005
Pozyskiwanie wiedzy i zarządzanie wiedzą

Paweł B. Myszkowski
University of Information Technology and Management *Copernicus* Wrocław, Poland

A PARTIAL FITNESS FUNCTION IN GENETIC ALGORITHM APPLIED TO GRAPH COLORING PROBLEM

Introduction

The major component of any evolutionary algorithm is its evaluation (fitness) function, which serves as a major link between the algorithm and the problem being solved. The evaluation function is used to distinguish between better and worse individuals in the population [Michalewicz99].

In this paper we consider Graph Color Problem as constraint satisfaction problem [Eiben98] and we use genetic algorithm to solve it. In most articles evaluation of colored graph is given to whole the graph. We propose evaluation function based on evaluation of each graph vertex in context of other color vertices – we call it *partial fitness function (pff)*. Values of proposed function can be useful information in whole evolution process (e.g. direct or specialize operators, measure diversity). Also in this paper, we investigate some complexity aspects of computing *pff* values.

2. Graph Coloring Problem (GCP)

The *Graph Coloring Problem* (GCP) is one of the most studied NP-hard problems and can be defined as follows: given undirected graph $G=(V,E)$, where V is a set of $|V|=n$ vertices and $E \subseteq V \times V$ is a set of $|E|=m$ edges, and integer number of colors k, to find such function $\Psi: V \mapsto 1,2,..,k$ which, for every edge of graph $[u,v] \in E$, $u,v \in V$ should be satisfy a constraint $\Psi(v) \neq \Psi(u)$ [EDOM, 696]. In GCP we use k colors to proper color a given graph, but when graph can be properly colored with *k-1* colors, *k*-coloring is non

optimal. This way we can consider GCP as optimization problem – we try to find a minimal number of used colors (chromatic number χ_G) needed to proper coloring of given graph [Paquete02][EDOM].

Despite GCP seems to be theoretical problem, there are many practical applications, e.g. in scheduling [Marx03], job shop scheduling [Kubale04], timetabling [Myszkowski04], frequency assignment [Malkevitch03], routing problem and many others [Kubale02].

The GCP landscape S size equals $|S| = k^n$, and seems to be very large for graphs with 100 or more vertices [Dorne98]. GCP problem is considered as an NP-hard and we don't know any effective algorithms, which give a solution in polynomial time. Its NP-hard complexity practically means that it is very difficult to find in acceptable time for user a legal coloring even for a graph with few vertices [Kubale02]. There are many approximation methods applied in GCP: approximation algorithms [Kubale02][Culberson92], heuristics (e.g. DSATUR [Hamirez04], RLF [Vesel00]), local search algorithms [Chiarandinni02], or metaheuristics, such as tabu search [Paquete02], simulated annealing [Fotakis01], ant colony algorithms [Vesel00] or genetic algorithms [Juhos02][Myszkowski04].

A graph coloring task for genetic algorithm (GA [Goldberg89a] [Michalewicz96]) is defined as: to legal color a given undirected graph $G = (V, E)$ with k (if $k = \chi_G$ it is an optimal coloring) colors through reduction number of conflicts (two of vertices $u, v \in V$ are in conflict if both are assigned in the same color and exist an edge $[u, v] \in E$ [Chiarandini04]). So, each individual in GA represents a proposition of given graph coloring.

3. A Partial Fitness Function (*pff*) and Coloring Representation

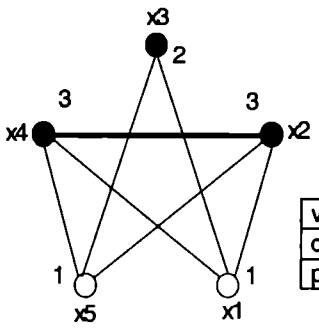
We decided to represent colored graph with k color as vector $I = < \Psi(v_0), \Psi(v_1), ..., \Psi(v_n) >$, where $\Psi(v_i)$ determine color of i vertex. Such representation is direct, intuitive and commonly used (e.g. [Croitoru02][Dorne98]). The evaluation function is based on penalty, which depends on number of pairs of neighbor vertices colored with the same color (conflict). Such value gives information how many conflicts is in already colored graph, however nothing is said about its localizations. In literature, number of conflicts corresponds only to colored graph; even if considerations are based on a single vertex, it is only a boolean value (if there is a conflict). Such evaluations make impossible not only the analysis of graph, but also the determination which one vertex is in „better" color which one in „worse".

The proposed partial fitness function (*pff*) each vertex V_i of graph evaluates in other vertices context, can be defined as:

$$pff(v_i) = \sum_{j=0}^{|V|} p(v_i, v_j), \quad where \; p(v_i, v_j) = \begin{cases} 1 \; if \; \Psi(v_i) = \Psi(v_j) \wedge [v_j, v_i] \in E \\ 0 \; else \end{cases}$$

If $pff(V_i)$ equals 0, it means that vertex V_i does not cause any conflicts with other vertices with the given color. A total sum of all *pff* vertices values gives a fitness of graph coloring.

On Picture 1 there is a graph with 5 vertices, colored with 3 colors and 1 conflict (between x2 and x4 vertex). Every vertex (except x2 and x4) *pff* value equals 0. Total evaluation value of given colored graph equals 2.



vertex	x1	x2	x3	x4	x5
color	1	3	2	3	1
pff value	0	1	0	1	0

A Graph evaluation = 2

Picture 1. Graph with 1 conflict and pff values of vertices

Thus, the proposed function *pff* extends a scalar evaluate function to a vector $E =< pff(v_0), pff(v_1), ..., pff(v_n) >$, assigns an evaluation value to every graph vertex. Such approach has many advantages, i.e. we can evaluate each vertex of colored graph. However there are also some disadvantages. A similar function is considered by Goldberg in [Godlberg89a,b][Goldberg90] where is drawn a conclusion that such function breaks a black box rule [Goldberg89a, pp.24-25] of genetic algorithm (do not analyze how solution is build). Also, there are many problems, which cannot be defined in this way – such function form limits applications to these tasks, which can be described by partial function [Goldberg 89b]. Even if the *pff* function limits applications to only GCP instances, in practice there are so many of them, that such limitation does not disqualify the approach.

4. Partial Fitness Function – Complexity Considerations

In GCP fitness function is one of most complexity components in genetic algorithm. Thus, we decide to analyze its method of implementation (FF1-FF4 function) and its relationship with graph coloring representation so as to minimize especially its computational complexity.

4.1. FF1 function (a simple one)

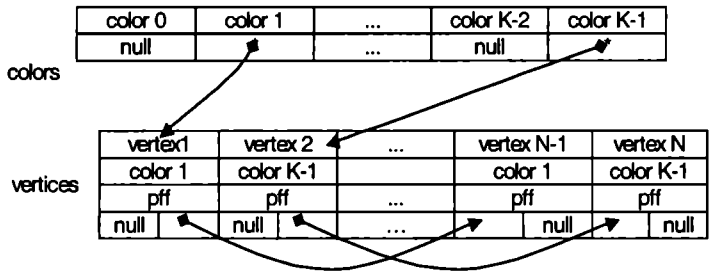
The simplest representation of graph coloring is a vector $I = < \Psi(v_0), \Psi(v_1), ..., \Psi(v_n) >$, where $\Psi(v_i)$ determine color of i vertex, but such representation needs evaluation function with a high computational complexity. We have to compare each vertex n in colored graph with the others and if there is a pair of vertices is in the same color, we must check whether there is a conflict. The memory complexity is relatively low and computational complexity equals:

$$O(n) = n(n-1) = n^2 - n$$

The highest cost of function is connected with comparison of each vertex with others in each step. This can be minimized by few methods – we consider it in function FF2 and FF3.

4.2. FF2 function (a table of colors)

The graph coloring representation was extended (its schema is presented on Picture 2), and every vertex has information about its color, partial fitness function value (*pff*), but also has a pointer to next (and previous) vertex in its color. Thus, the representation includes a table of colors, where each color k has information about the first vertex in selected color.



Picture 2. A modified representation of colored graph in FF2 function

The memory complexity of this representation is bigger than FF1, but computational complexity decrease to:

$$O(n) = k(\frac{n}{k})(\frac{n}{k} - 1) = n(\frac{n}{k} - 1) = \frac{n^2}{k} - n$$

The reduction of complexity is connected with the fact, that in evaluation of each vertex we consider only the ones in the same color (not all vertices as in function FF1).

As it was said, such representation is much more memory demanding, however is worth to decrease the computational complexity, especially when we color a graph with 1000 or more vertices.

4.3. FF3 function (a table of colors + sequential evaluation)

When a FF2 function evaluates the whole graph, each pair of vertices in the same color is considered twice. This can be reduced when vertex evaluation will be based on sequential vertex inspection – evaluation of each vertex with its following vertices (evaluation with its possible proceedings was done before). The FF3 function pseudo code is presented on Picture 3. The function *fitness* returns value of evaluation graph coloring and function *calculatePff* computes a value of *pff* function given vertex V_i.

The memory complexity does not change (in relation to FF2), but computational complexity decrease on half to:

$$O(n) = \frac{1}{2}(\frac{n^2}{k} - n).$$

Functions FF1, FF2 and FF3 are based on checking if in selected color, given pair of vertices causes conflict (if there is a edge between them). The FF4 function works completely different. To evaluate a given graph we consider its structure, especially its edges.

```
function fitness(graph G)
begin

for (every vertex Vi of graph G) pff(Vi):=0;

for (every vertex Vi of graph G) calculatePff(Vi);

sum    0;

for (each vertex Vi of graph G) sum    sum + pff(Vi);

fitness    sum;
end

procedure calculatePff (vertex Vi)
begin
       If Vi <> null then
              While Vi.next <> null
                     Vj := Vi.next;
                     If IsConfict(Vi,Vj) then
                                   Increase pff(Vi);
                                   Increase pff(Vj);
                     Vi:= Vj;
end
```

Picture 3. A FF3 pseudo code. A graph colorings evaluation based on partial fitness function

4.4. FF4 function (a broken constraints)

Considerations of FF1-FF3 function forms allow us to come to conclusions that very often we analyze pairs of vertices that are not connected with other by edge.

This situation can be reversed and we can consider graph edges as constraints, which must be satisfied. Then an evaluation of graph coloring will be connected with the check of each edge of given graph whether its vertices are in different colors.

Such evaluation method has different complexity and equals to number of edges $O(n)=m=|E|$ of colored graph. But there is a risk that number of edges (with reference to number of vertices) is so large that the method is ineffective. Thus we can draw a **condition of method FF4 efficiency** as:

$$\frac{1}{2}(\frac{n^2}{k}-n)>m$$

Given formula is based on comparison of computational complexity of functions FF3 and FF4. The FF4 function is more profitable when given formula is true otherwise a FF3 function is recommended.

The analysis of 63 benchmark DIMACS (DIMACS GCP instances: http://mat.gsia.cmu.edu/COLORING03/) graph give conclusion that in case of 27 graphs FF4 is more effective than FF3. To show the difference of cost values, see Picture 4. There are presented 3 selected graphs: one with the biggest number of vertices, one with the biggest number of edges and the last one with the biggest density. The graph density is interpreted as a number of vertices in relation to number of edges and defined as follows: *den*(G)=2m/(n(n-1)) [Borowiecki01].

Graph	Cost of FF3	Cost of FF4	Difference (better)
A graph with **the biggest number of vertices** (4-fullins_5.col), n=4246, m=77305, k=9, den=1%	952 889	77 305	~12x (FF4)
A graph with **the biggest number of edges** (dsjc1000.9), n=1000, m=449449, k=224, den=90%	1792	449 449	~259x (FF3)
A graph with **the biggest density** (dsjc500.9), n=500, m=224 876, k=126, den=180%,	742	224876	~303x (FF3)

Picture 4. Selected graphs, their features and costs of FF3 and FF4 functions.

Thus an intuition gives us a hint, that there is some relationship between graph density and FF4 efficiency. Data presented on seem to confirm it.

We determine a m variable form a density formula and we provide it in FF4 efficiency formula. We receive the formula:

$$\frac{\frac{n}{k}-1}{n-1}>den(G)$$

The structure of colored graph cannot be change, so variable n, m and graph density $den(G)$ is constant. We have only one variable k (number of used colors), which value can be changed. It gives a conclusion: the k value bigger the less is usage method FF4 efficiency.

Briefly analysis of benchmark DIMACS graphs gives information, that FF4 is profitable for one graph classes (such MIZ: ash,will or CAR: x_insertions_x, x_fullins_x). In case of other classes (with small exceptions) it is more profitable to apply a FF3 function.

The application of FF4 function (if the FF4 efficiency formula has true value) gives a desirable reduction of computational complexity, but this situation occurs only in less than half of analyzed graphs. Thus, it is recommended a FF3 and FF4 coexistence and depends on condition of method FF4 efficiency, we run only one. Additionally, FF3 function has a feature (FF4 does not), which allows us to evaluate a part of graph (e.g. vertices in selected color). This feature is very useful to direct or/and specialize genetic operators.

5. An Applications of Partial Fitness Function

The partial fitness function has disadvantage because we have to store its value for each vertex, but we benefit many useful *pff* applications to e.g. diversity population measure or direct/specialize genetic operators.

5.1. A Measure of Population Diversity

A population diversity in genetic algorithm is a crucial factor its efficiency. A fast lost of the diversity in population leads to premature convergence [Galinier99]. So it is important to keep population diversity on high level. There are two types of diversity measures: *genotypic diversity measure* (GDM) and *phenotypic diversity measure* (PDM). The first one involves genetic material held in population, the second one concerns the fitness of individuals [Herrera98].

The graph coloring is represented as vector $I = < \Psi(v_0), \Psi(v_1), ..., \Psi(v_n) >$ and simply can measure distance between two colorings I_K and I_L by *Hamming distance*:

$$D_H(I_K, I_L) = \sum_{i=0}^{|V|} h(\Psi_K(v_i), \Psi_L(v_i))$$

$$where\ h(\Psi_K(v_i), \Psi_L(v_i)) = \begin{cases} 1\ if & \Psi_K(v_i) \neq \Psi_L(v_i) \\ 0\ else & \end{cases}$$

The D_H distance has many disadvantages (e.g. does not take into consideration a space symmetry [Hamirez01]), there are also other GDM applied, based on [Herrera96]: histograms, dispersion statistical, Euclidean distance, Entropy distance or others [Hamirez01][Galinier99].

A construction of GDM is rather a simply task, more problematic is PDM. In [Herrera98] it is proposed to control relation between individuals with the best and average fitness in population (or average and the worst) to keep information about current population diversity. However this measure gives only

statistical information for given population but with aid of *pff* values we can extend it so as to have more precise measure between two evaluations of graph colorings E_K and E_L:

$$D_{PFF}(E_K, E_L) = \sum_{i=0}^{n} \left| pff_K(v_i) - pff_L(v_i) \right|$$

where $pff_K(V_i)$ means a *pff* value of i vertex in K coloring. A D_{PFF} measure gives information how two colorings are different on phenotypic level.

The PDM seems to be useless but in [Herrera98] it is said, that principal feature of this type of measures is that they allow premature convergence to be predicted rather than being detected. This features gives information about low level of population diversity and we can run a preventive strategy.

5.2. A Directed Mutation Operator

A mutation operator works on single graph coloring and returns this coloring with some changes. The standard mutation operator is based on selection of vertex and random change its color. The mutation is applied with a constant probability P_m for each vertex. Why P_m probability has the same value for vertex with and without conflicts? It is more profitable to distinguish value of probability P_m [Myszkowski03a] and increase its value in proportion to number of conflicts in given vertex. In this simple way we can direct a mutation P_{dM} of vertex e:

$$P_{dM}(e) = P_m + FitStr \cdot \frac{pff(e)}{fitness} \cdot P_m,$$

where $FitStr \in <0;5,0>$ is „strength" of direction, *fitness* is evaluation of all vertices in given graph coloring (sum of all *pff* values in graph).

5.3. A Highly Specialized Genetic Operators

The *pff* gives another possibility – we can build highly specialized genetic operators. Such highly specialized operator IBIS (*Iteration Build Solution*) is presented in [Myszkowski03b,04], based on *pff* values and fuzzy logic.

Also, crossover operators can take to advantage of *pff* function, because „designing crossover requires first identification of some 'good properties' of the problem which must be transmitted from parents to offspring and the development of an appropriate recombination mechanism" [Galinier99, pp.380]. So, the *pff* values give us such useful information about given graph coloring, about its 'good' and also about 'bad' properties.

References

[Borowiecki01] Borowiecki P., Kubale M., *A Survey of Hard-to-Color graphs for off-line and on-line model of vertex coloring*, Proceedings of the 10th International Conference on System-Modelling-Control, Zakopane, 2001.

[Chiarandinni02] Chiarandini M., Stuetzle T., *An application of Iterated Local Search to Graph Coloring Problem*, [in:] D.S. Johnson, A. Mehrotra, and M. Trick, editors, Proceedings of the Computational Symposium on Graph Coloring and its Generalizations, p. 112-125, Ithaca, New York, USA, September 2002.

[Culberson92] Culberson J.C., *Iterated Greedy Graph Coloring and the Difficulty Landscape*, Technical Report 92-07, University of Alberta, Canada, 1992.

[Dorne98] Dorne R., Hao J-K., *A New Genetic Local Search Algorithm for Graph Coloring*, Proceedings of the 5th International Conference on Parallel Problem Solving from Nature, p. 745-754, September 27-30, 1998.

[Eiben98] Eiben A.E., van der Hauw J.K., van Hemert J.I., *Graph Coloring with Adaptive Evolutionary Algorithms*, Journal of Heuristics, 4(1), pp. 25-46, 1998.

[EDOM] *Encyclopenic Dictionary of Mathematic*, ed. Kijosi Ito, The MIT Press, Cambridge, Massachusetts and London, England; second edition, 1993.

[Fotakis01] Fotakis D.A., Likothanassis S.D., Stefanakos S.K., *An evolutionary annealing approach to graph coloring*, E.J.W. Boers and al. (Eds.) *EvoWorkshop* 2001, LNCS 2037, pp. 120-129, Springer-Verlag 2001.

[Galinier99] Galinier P., Hao J-K., *Hybrid Evolutionary Algorithms for Graph Coloring*, Journal of Combinational Optimization, 3(4), pp. 379-397, 1999.

[Goldberg89a] Goldberg D.E., *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley Publ. (1989), (*polish translation Algorytmy genetyczne i ich zastosowania*, Wydawnictwo Naukowo-Techniczne WNT, Warszawa 1995).

[Goldberg89b] Goldberg D.E., Korb B, Deb K., *Messy Genetic Algorithms: Motivation, Analysis, and First Results*, Complex Systems 3 (1989), pp. 493-530, 1989.

[Goldberg90] Goldberg D.E., Deb K., Korb B., *Messy Genetic Algorithms Revisited: Studies in Mixed Size and Scale*, Complex Systems 4 (1990), pp. 415-444, 1990.

[Hamirez04] Hamirez J-P., Hao J-K., *An analysis of solution properties of the graph coloring problem*, Applied Optimization, Metaheuristics: Computer decision-making, pp. 325-345, ISBN:1-4020-7653-3, 2004.

[Herrera98] Herrera F., Lozano M., *Fuzzy Genetic Algorithms: Issues and Models*, Technical Report DECSAI-98116, Dept. of Computer Science and A.I., University of Granada, June 1998, 1998.

[Juhos03] Juhos I., Toth A., Tezuka M., Tann P., van Hemert J.I., *A New Permutation Model for Solving the Graph k-Coloring Problem*, Proceedings Kalmfr Workshop on Logic and Computer Science, pp. 189-199, 2003.

[Kubale02] Kubale M. (red), *Optymalizacja dyskretna: modele i metody kolorowania grafów*, Wydawnictwo Naukowo-Techniczne, Warszawa 2002. (*polish*).

[Kubale04] Kubale M., Nadolski A., *Chromatic Scheduling in a Cyclic Open Shop*, European Journal of Operational Research (acceoted), 2004.

[Malkevitch03] Malkevitch J., *Cell Phone Tidbit*, http://www.york.cuny.edu/~malk/tidbits/tidbit-cellphone.html, 2003.

[Marx03] Marx D., *Graph Coloring Problems and Their Applications in Scheduling*, John von Neumann PhD stundents Conference, Budapest University of Technology and Economics, 2003.

[Michalewicz96] Michalewicz Z., *Genetic algorithms + data structures = evolution programs*, Springer-Verlag Berlin Heidelberg (1996), (polish translation: *Algorytmy + struktury danych = programy ewolucyjne*, Wydawnictwo Naukowo-Techniczne WNT, Warszawa 1996).

[Michalewicz99] Michalewicz Z., *The Significance of the Evaluation Function in Evolutionary Algorithms*, [in:] Davis D., deJong K., Vose M.D., Whithey L.D. (eds)., *Evolutionary Algorithms*, The IMA Volumes in Mathematics and its applications, vol. 11, Springer, 1999. 1999.

[Myszkowski03a] Myszkowski P.B., Norberciak M., *Evolutionary Algorithms for Timetable Problems*, Annales UMCS Informatica AI 1 (2003), pp. 115-125, 2003.

[Myszkowski03b] Myszkowski P.B., *A Hybrid Genetic Algorithm for Timetable Problem*, Proceedings of 9th International Conference on Soft Computings MENDEL 2003 (Brno, Czech Republic), pp. 102-106, 2003.

[Myszkowski04] Myszkowski P.B., Kwaśnicka H., *IBIS: A New Evolutionary Algorithm for the Timetable Problem*, Proccedings Intelligent Information Processing and Web Minning, IIS:IIPWM'04, [in:] Advances in Soft Computing, pp. 454-458, Springer 2004.

[Paquette02] Paquete L., Stuetzle T., *An Experimental Investigation of Iterated Local Search for Coloring Graphs*, Applications of Evolutionary Computing, Proceedings of EvoWorkshops 2002, volume 2279 of Lecture Notes in Computer Science, p. 122-131. Springer-Verlag, 2002. Best Paper Award EvoCOP, 2002.

[Vesel00] Vesel A., Zerovnik J., *How Good Can Ants Color Graphs*?, Journal of Computing and Information Technology – CIT (8), pp. 131-136, 2000.

CZĘŚCIOWA FUNKCJA OCENY W ALGORYTMIE EWOLUCYJNYM ZASTOSOWANYM W ROZWIĄZYWANIU PROBLEMU KOLOROWANIA GRAFU

Streszczenie

Artykuł dotyczy zastosowania algorytmów ewolucyjnych w problemie kolorowania grafu. Zaproponowano sposób oceny pokolorowania grafu biorący pod uwagę lokalizacje konfliktu (sąsiednie wierzchołki pokolorowane tym samym kolorem). Pokazano możliwości zastosowania takiej funkcji oceny przy ukierunkowaniu, wyspecjalizowaniu operatorów genetycznych i pomiarze różnorodności fenotypowej w populacji. Także przeanalizowano sposoby implementacji funkcji oceny, ze szczególnym uwzględnieniem optymalizacji jej złożoności obliczeniowej.