Polynomial Strategies for the 3-Coloring of a Graph.

Guillermo De Ita Luna*, Yuridiana Alemán* and Nahum Loya*

ABSTRACT

The coloring of a graph is a problem of interest in the area of computer science due to the many applications it offers. The graph coloring problem has many utilities in areas like scheduling problems, frequency allocation, planning, etc. In the coloring of a graph, we want to color the nodes properly with the smallest possible number of colors. We present some necessary conditions for the 3-coloring of an input graph. All of those conditions can be checked in polynomial time. We also propose an appropriate combinatorial pattern representing proper 3-coloring of a graph based in its basic cycles, and where such pattern is codified via satisfy assignments of a two conjunctive Boolean formula. The Boolean formula is formed according to the basic cycles appearing in the graph. This paper shows the methodology for 3-coloring of a graph using 2-CF (Conjunctive form), as well as by several examples illustrate the calculation of it.

RESUMEN

El coloreo de un grafo es un problema de interés en el área de las ciencias de la computación debido a las muchas aplicaciones que este ofrece. El problema de coloreo de un grafo tiene varias aplicaciones en áreas como en el problema de asignación de tareas, asignación de frecuencias, planeación, etc. En el coloreo de un grafo, se asigna un color apropiado a los nodos (de forma que dos nodos adyacentes no tengan igual color), usando el menor numero posible de colores. Se presentan algunas condiciones necesarias para el 3-coloreo de un grafo de entrada, todas esas condiciones se pueden comprobar en tiempo polinomial.También se propone un patrón combinatorio apropiado para la representación del 3-coloreo de un grafo basado en sus ciclos básicos, y donde dicho patrón es codificado a través de la satisfactibilidad de una formula booleana en dos forma conjuntiva. La formula booleana es formada de acuerdo a los ciclos básicos presentes en el grafo. En este artículo se presenta una metodología para el 3-coloreo de un grafo utilizando 2-CF (dos forma conjuntiva), así como su cálculo por medio de ejemplos.

Recibido: 22 de Diciembre del 2011
Aceptado: 14 de Febrero del 2012

INTRODUCTION

The coloring of a graph $G = (V, E)$, with a vertex set V and an edge set E, is a topic very important in the computers science area. This is an active field of researching with many interesting applications. Some of the possible applications for the coloring of a graph, are for example: scheduling problems, frequency allocation, planning, energizing networks [5] [6] [7] [9], etc. Essentially the problem of the coloring of a graph is to minimize the number of colors used to color the vertices with the restriction that two adjacent vertices cannot have the same color, as we shown in figure 1. The chromatic number of a graph G, denoted as $\chi(G)$, is the minimum number of colors for coloring G in a proper way (no any pair of adjacent vertices have the the same color).

Palabras clave:
3-coloreado; 2-Forma conjuntiva; coloreo de un grafo; números cromáticos de un grafo.
Keywords:
3-coloring; 2-Conjunctive Form; Graph Coloring,Chromatic Number of a Graph.

Determining the chromatic number of a graph G was one of the first 22 NP-complete problems initially presented by Karp. One of the first bound to color a 3-colorable graph was established by Wigderson [8], he showed how to color 3-colorable graphs with at most $3 \cdot \lceil \sqrt{n} \rceil$ colors, where n is the number of nodes on the graph. Blum and Karger [4] applied semidefinite programming (SDP) to improve this bound to $\widetilde{O}(n^{3/14})$, where the notation $\widetilde{O}$ is used to supress polylogarithmic factors. More recently, Arora *et al.* [1] using stronger SDPs have improved the bound to $O(n^{0.2111})$.

*Computer Sciences Department, Benemérita Universidad Autónoma de Puebla,Edif.104, 14 Sur con San Claudio, Sn. Manuel, C.P. 72570, Puebla, Pue., México deita@cs.buap.mx, annie_17656@hotmail.com, israel_loya@hotmail.com

Throughout these years of study, it has found that this problem can be partially modeled on a polynomial time when the degree of the graph n is $n \leq 2$, however when $n \geq 3$, then becomes a NP problem.

Subsequently, several algorithms have been designed for modeling the $3-coloring$ graph problem and it has become an active field of researching. Among other goals, to reduce the performing time of algorithms for 3-coloring graphs is a relevant objective in this area.

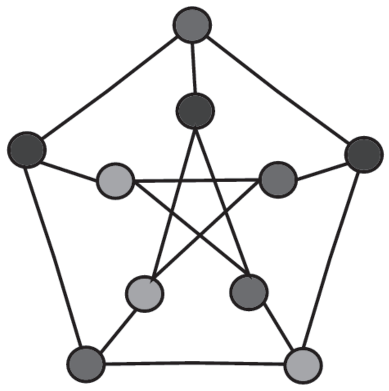

Figure 1 . Shows an example of an appropriate 3-coloring for the graph shown.

In the issue of graph coloring is very important the time complexity of the algorithm. Along the history, many researchers as: Lawler, Byskov, Wigderson, Blum and Kargen, among others, they have proposed different algorithms with varied time complexity in order to resolve the problem of $3-coloring$ graphs [5] [8] [4].

The analysis of the time complexity for the coloring problem is usually based on the size of the graph (number of vertices and number of edges). We believe that better upper bounds for this problem can be achieved if we consider the basic cycles of the input graph, since this allow us to find necessary conditions for the $3-coloring$ of a graph and with the advantages that such conditions can be checked in polinomial time.

It should be emphasized that not every graph is likely to be $3-colored$ since it depends on several characteristics that the graph must have. One of the main objetives of this article is to list such characteristics.

We consider the analysis of patterns formed by the basic cycles of a graph, and we try to discover which patterns give us necessary conditions for the $3-coloring$ of a graph. And when the graph has not those conditions, we present a novel reduction from the 3-coloring problem to the satisfiability problem of a two conjunctive form.

We associated a graph G with a two conjunctive form (2-CF) F_G in such a way that any model of F_G determines a proper $3-coloring$ of G. If G is not 3-colorable, then F_G is an unsatisfiable formula.

THEORETICAL FRAMEWORK

Let $G = (V, E)$ be an undirected simple graph (i.e. finite, loop-less and without multiple edges) with vertex set (or nodes set) V and set of edges E. $E(G)$ and $V(G)$ emphasize that these are the edges and vertex sets of a particular graph G. Two vertices v and w are called *adjacent* if there is an edge $\{v, w\} \in E$, joining them.

The *Neighborhood* of $x \in V$ is $N(x) = \{y \in V : \{x, y\} \in E\}$ and its *closed neighborhood* is $N(x) \cup \{x\}$ which is denoted by $N[x]$. Note that v is not in $N(v)$.

We denote the cardinality of a set A, by $|A|$. Given a graph $G = (V, E)$, the degree of a vertex $x \in V$, denoted by $\delta(x)$, is $|N(x)|$. The size of the neighborhood of x, $\delta(N(x))$, is $\delta(N(x)) = \sum_{y \in N(x)} \delta(y)$.
The maximum degree of G or just the degree of G is $\Delta(G) = max\{\delta(x) : x \in V\}$, while we denote with $\delta_{min}(G) = min\{\delta(x) : x \in V\}$ and with $\delta(G) = (2 \cdot |E|)/|V|$ the average degree of the graph.

Given a subset of vertices $S \subseteq V$ the subgraph of G denoted by $G|S$ has vertex set S and set of edges $E(G|S) = \{\{u, v\} \in E : u, v \in S\}$. To $G|S$ is called the *subgraph of G induced by S*. We write $G - S$ to denote the graph $G|(V - S)$. The subgraph induced by $N(v)$ is denoted as $H(v) = G|N(v)$ which has to $N(v)$ as the set of nodes and all edges upon them.

A path from a vertex v to a vertex w in a graph is a sequence of edges: $v_0v_1, v_1v_2, \ldots, v_{n-1}v_n$ such that $v = v_0$, $v_n = w$, v_k is adjacent to v_{k+1} and the length of the path is n. A simple path is a path such that $v_0, v_1, \ldots, v_{n-1}, v_n$ are all distinct. A cycle is just a nonempty path such that the first and last vertices are identical, and a simple cycle is a cycle in which no vertex is repeated, except the first and last vertices.

A k-cycle is a cycle of length k, that is, a k-cycle has k edges. A cycle of odd length is called an odd cycle, while a cycle of even length is called an even cycle. A graph G is acyclic if it has not cycles.

A *complete graph* of n nodes has $n \cdot (n-1)/2$ distinct edges, we denote K_n the complete graph of n nodes. A graph G is a *regular graph* if all vertices have the same degree, G is $k-regular$ if it is regular, of degree k.

A connected component of G is a maximal induced subgraph of G, that is, a connected subgraph which is not a proper subgraph of any other connected subgraph of G. Note that, in a connected component, for every pair of its vertices x, y, there is a path from x to y. If an acyclic graph is also connected, then it is called a free tree.

A coloring of a graph $G = (V, E)$ is an assignment of colors to its vertices. A coloring is *proper* if adjacent vertices always have different colors. A k-coloring of G is a mapping from V into the set $\{1, 2, \dots, k\}$ of k "colors". The chromatic number of G denoted by $\chi(G)$ is the minimum value k such that G has a proper k-coloring. If $\chi(G) = k$, G is then said to be k-chromatic. To determine the value $\chi(G)$ is polynomial computable when $\chi(G) \leq 2$, but when $\chi(G) \geq 3$, the problem becomes NP-complete, even for graphs G with degree $\Delta(G) \geq 3$.

Let $G = (V, E)$ be a graph, G is a *bipartite graph* if V can be partitioned into two subsets U_1 and U_2, called *partite sets*, such that every edge of G joins a vertex of U_1 and a vertex of U_2.

If $G = (V, E)$ is a k-chromatic graph, then it is possible to partition V into k independent sets $V_1, V_2, ..., V_k$, called *color classes*, but it is not possible to partition V into $k - 1$ independent sets.

The Basic Cycles of a Graph

Given an undirected connected graph $G = (V, E)$, applying a depth-first search for traversing G produces a tree graph T_G, where $V(T_G) = V(G)$. The edges in T_G are called *tree edges*, whereas the edges in $E(G)\backslash E(T_G)$ are called *back edges*.

Let $e \in E(G)\backslash E(T_G)$ be a given back edge. The union of the path in T_G between the endpoints of e with the edge e itself forms a simple cycle, such cycle is called a basic (or fundamental) cycle of G with respect to T_G. Each back edge holds the maximum path contained in the basic cycle which it is part of. We define the *end-nodes* of a cycle as the nodes which are part of the back edge of the cycle.

Let $C = \{C_1, C_2, ..., C_k\}$ be the set of fundamental cycles found during the depth-first search on G. Given any pair of basic cycles C_i and C_j from C, if C_i and C_j share any edges they are called *intersected* cycles; otherwise they are called *independent* cycles.

The or-exclusive operation between two *intersected* cycles: $C_i \oplus C_j$ form a new cycle, $\oplus$ denotes the or-exclusive operation between the set of edges of both cycles. In particular, if two cycles share one edge they are called *adjacent*. We say that a set of cycles is independent if no two cycles in the set are intersected.

It is known that any acyclic graph is 2-colorable and therefore, 3-colorable too. Furthermore, if a graph G has only cycles of even length then G is also 2-colorable since alternating two colors with respect to the levels of the tree T_G builds proper 2-colorings.

Cycles of odd length require at least 3 colors to make proper colorings. Therefore, the subgraph structures which generate conflict for the 3-colorings are the odd cycles in the graph. We show in the following subsection how to treat those structures for building 3-colorings.

POLYNOMIAL TIME PROCEDURES FOR COLORING GRAPHS

Let us assume an input connected graph $G = (V, E)$, with $n = |V|$ and $m = |E|$, as well as an order on the vertices for applying a depth-first search. For example, starting the search with a node $v \in V$ of minimum degree and visiting the node of lowest degree whenever there are multiple possible nodes to visit. Such depth first search will be denoted as $G' = dfs(G)$.

The main class of graphs which is known to be colored in polynomial time is the class of bipartite graphs.

Lemma 1. A graph G has chromatic number 2 if and only if G is bipartite.

This is true because the partite set U_1 can be colored with the first color while the other partite set U_2 is colored with the second color. Furthermore, as the bipartite property can be recognized in polynomial time, then a 2-colorable graph can be recognized in polynomial time, based on the following property.

Lemma 2. A graph G is bipartite if and only if G contains no odd cycles.

$G' = dfs(G)$ builds the new depth-first graph G' and moreover, dfs allows us to detect if G has cycles or not and whether these cycles are even or odd in time $O(m + n)$ [2].

Given $G' = dfs(G)$, let T_G be the spanning tree of G'. Notice that $V(T_G) = V(G)$. If G is an acyclic graph, then $T_G = G'$. Let $\{G, R\}$ be the set of two basic colors. If G' is acyclic or it contains only even fundamental cycles, then G is bipartite and it is proper colorable with just the two basic colors, because the vertices in G' can be colored according to the levels of the spanning tree T_G, that is, all vertices in the same level on the tree have the same color and the nodes of two sequential levels from the tree are colored with two alternating colors.

Given a graph $G = (V, E)$, $S \subseteq V$ is an independent set in G if for whatever two vertices v_1, v_2 in S, $\{v_1, v_2\} \notin E$. Let $I(G)$ be the set of all independent sets of G. An independent set $S \in I(G)$ is *maximal*, abbreviated as MIS, if it is not a subset of any larger independent set and, it is *maximum* if it has the largest size among all independent sets in $I(G)$.

The most common method for coloring a graph G, is to build a Maximum (or sometimes a Maximal) Independent Set of G in order to determine a class color for the graph [3, 4, 5]. Many of the coloring algorithms utilize a combination of improved upper bounds on the number of MIS's of size at most k, and some changes in the way to fill the table of sub-solutions in a dynamic programming approach. We address here another interesting direction, we look for cycle-patterns which allow us to determine when the graph could be coloring with at most three colors.

Necessary Conditions for the 3-Coloring of a Graph

It is known that any simple odd cycle requests three colors to be colored. We enumerate now several cases describing necessary conditions for the 3-coloring of a graph.

1. If G' has no odd basic cycles then G is 2-colorable. This option considers the case where G is a bipartite graph. Since alternating two colors with respect to the levels of the tree T_G builds proper 2-colorings.
2. If G' is a planar triangle-free graph then G is 3-colorable.
3. If G' has odd cycles, but every one of them is independent from every other odd cycle, then G is 3-colorable.
   Because we can color G' by levels and using the third color for coloring just one end-node of each odd cycle.
4. If all odd cycles of a graph G can be arranged in groups with less than four intersected odd cycles and without any even cycle then G is 3-colorable. As in the previous case, G' is two colored by levels, and the third color is used for coloring one end-node of an odd cycle which appears as a common node of a pair of intersected odd cycles.
5. If all set of intersected odd cycles in G can be arranged as embedded plane cycles, then G is 3-colorable. We color any set of embedded plane cycles from the most internal to the most external cycle. When we start to color a new cycle, we use a different color to their neighboring nodes (at most there are two neighboring nodes), since we consider that only the nodes in the internal cycles have already been colored. In this case, we can consider the input graph G as an instance of a *Series-Parallel graph*.

Notice that all the previous cases describe cycle-patterns which allow efficient 3-colorings of graphs. Furthermore, all the procedures described in those cases have a polynomial time complexity on the size of the input graphs. Therefore, if the topological structure of the input graph G is any of the aforementioned basic cases, G is 3-colorable in polynomial time.

If a graph G is not 3-colorable, then it does not hold any of the previous cycle-patterns. For example, a graph G with two intersected basic odd cycles which are intersected with only one basic even cycle is given by the complete graph K_4 which requires 4 colors to be colored. In the same way, a simple graph formed by 3 odd cycles and 1 even cycle can not be 3-colored; all of them are intersected cycles and that graph requests a minimum of 4 colors to be colored.

In general, we can color a graph by coloring group of intersected cycles, which include at least two odd cycles involved in such group since otherwise, the graph would fall into one of the previous cases.

Now, we propose a way to color from one intersected cycle group to another, assuming that there are not common cycles between two different groups of intersected set cycles. For each group, we design the procedure to be presented in the following section in order to determine the 3-colorability of any group of intersected cycles.

PROCEDURE FOR 3-COLORING A SET OF INTERSECTED CYCLES

Let $G = (V, E)$ be a simple graph. Let $G' = dfs(G)$ and T_G be as we have described them in previous section. Let $C = \{C_1, C_2, ..., C_k\}$ be the set of basic cycles found during the depth-first search on G.

During the application of the "depth-first search", two sets C_o and C_e are formed. C_o contains the basic cycles of odd length and C_e the basic cycles of even length of G.

We apply the following procedure in order to determine the possible 3-colorability of the input graph G.

1. We start colouring the nodes in T_G using two colors $\{G, R\}$ - the same colour for all nodes at the same level, and with different colours for different levels, that is, alternating colours with respect to the levels of the tree T_G.

2. If $C_o = \emptyset$ then G is 2-Colorable.

3. If all the odd basic cycles are independent, then G is 3-colorable.
   Two alternating colours on T_G and a third colour assigned to just one of the two nodes of each back edge of the cycles in C_o build the proper 3-Coloring of G.

4. Let $CE \subseteq C_e$ be the set of cycles of even length which are intersecting with any odd cycle in G', i.e.,

$$CE = \{C_i \in C_e : \exists C_i \in C_o, C_i \cap C_j \neq \};$$

5. We assume $k = |C_o|$, there are k odd basic cycles in G'. Let $De = \{be_1, \ldots, be_k\}$ be the set of k back edges formed from C_o, such that each $be_j \in De$ is the back edge of $C_j \in C_o$, $j = 1, \ldots, k$.

6. For each $be_j = \{x_j, y_j\} \in De$, $j = 1, 2, \ldots, k$ two binary clauses are built. I.e., $A_j = (x_j \vee y_j) \wedge (\sim x_j \vee \sim y_i)$

7. For each pair of different back edges in D_e:
   $be_l = \{x_l, y_l\}$ and $be_j = \{x_j, y_j\}, l \neq j$ where one of its endpoints (x_j or y_j) is adjacent to x_l o y_l, assuming that x_l y x_j are adjacent nodes to x_l or y_l, the following binary clause B_j is built as: $(\sim x_l \vee \sim x_j)$.

8. For each back edge $e_l = \{x_l, y_l\}$ of a cycle in CE, the following binary clause E_l is built as: $(\sim x_l \vee \sim y_l)$.

9. Let F_G be the 2-CF formed by the conjunction of the $A_i's$ and $B_j's$ and $E_l's$ clauses built in the previous steps.

10. Determine if F_G is satisfiable or not. If F_G is satisfiable then G is 3-colourable and any model of F_G represents a proper 3-Colouring of G. Otherwise, our heuristic can not determine a proper 3-Colouring of G.

11. The variables assigned **true** in a model of F_G correspond to the nodes to be colored with the third color "W". If such nodes are deleted from G, the remaining subgraph is bipartite and its nodes can be colored with the two basic colors: $\{G, R\}$.

Each type of clause in our procedure has a specific function into the 3-coloring. Table 1 describes the meaning of each clause.

Table 1 .

Relationship between the third color and the satisfiability of the clauses.

Clauses	Intended meaning
$A'_j s$	For each back edge of an odd cycle, just one of its two nodes must be colored with W
$B'_j s$	No pair of nodes with color W may be adjacent
$E'_j s$	Both end-nodes of a back edge of an even cycle are not colored with W

The assignments that satisfy F_G conform a subset of all possible ways of 3-coloring G. Any model of F_G represents a way to $3-coloring$ G, although it may happen that G is $3-colorable$, but F_G is unsatisfiable. We can conclude about of our proposal that any satisfiable assignment of the 2-CF F_G determines a proper $3-coloring$ for the input graph G.

Figure 2 shows an example of a graph G, which meets all the requirements to apply the algorithm presented. The segmented edges represent back edges found during the search graph depth.

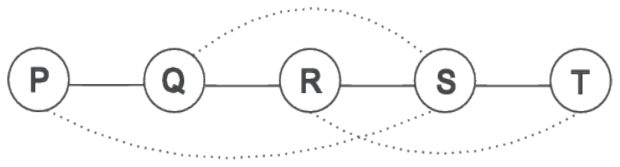

Figure 2 . Example of graph before apply 3-coloring of a graph.

From graph In figure 2, we obtain the following:

- Basic cycles of odd length.
  $C_o = \{cicle(r, s, t), cicle(q, r, s)\}$.

- Basic cycles of even length.
  $C_e = \{cicle(p, q, r, s)\}$.

- The basic cycles of odd length are not independent
  $D_e = \{\{q, s, \}, \{r, t\}\}$.

- The set of even length cycles that intersect with odd-length cycles, is:
  $C_e = \{cicle(p, q, r, s)\}$

From previous sets, consider the following clauses:

- $A_1 = (q \vee s) \wedge (\sim q \vee \sim s)$
- $A_2 = (r \vee t) \wedge (\sim r \vee \sim t)$
- $B_1 = (\sim r \vee \sim s)$
- $E_1 = (\sim p \vee \sim s)$

Therefore, F_G is defined as:

$$F_G = A_1 \wedge A_2 \wedge B_1 \wedge E_1$$

The values that satisfy F_G are $p = F$, $q = V$, $r = F$, $t = V$. Hence the nodes with the color R are: q and t. Once colored, these nodes are removed from the main graph. The remaining subgraph is bipartite and then, it can be colored with just two colors. For example, p and r are assigned the color G, and s is colored with the color B, as it is shown in figure 3.

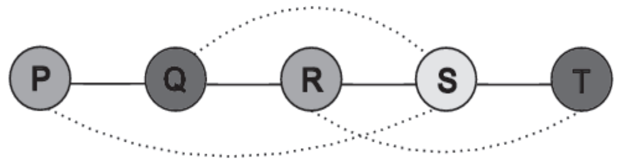

Figure 3 . Example of graph after apply 3-coloring of a graph.

Let $G = (V, E)$ be a simple connected graph with $n = |V|$ and $m = |E|$. A depth-first search over G is of order $O(m + n)$ and it builds an equivalent depth-first graph G'. The set of basic cycles $C = C_o \cup C_e$ is built during the depth-first search. As every cycle has no more than n nodes, then to determine the size of the cycle during the depth-first search involves $O(n * m)$ basic operations.

If the number of basic cycles in T_G is not big, e.g., it is upper bounded by a polynomial function on n or m then the Boolean formula F_G is formed with a time complexity of order $O(poly(n, m))$, $poly$ being a polynomial function.

Otherwise, it is possible to consider topological graphs where their number of basic cycles grow as an exponential function over n, e.g., the class of graphs K_n (the complete graphs with n nodes), and for this class of graphs, both the set of basic cycles C and the Boolean formula F_G are built in exponential time complexity on n.

Let $nc = |C|$ be the number of basic cycles of the input graph. The time required to determine if there are intersecting cycles in a graph is of order $O(nc^2)$ since it consists of the following loop:
for all $C \in C$, for all $C' \in C$, $C \neq C'$, test $C \cap C' \neq \emptyset$, $\cap$ being the intersection operation between the set of edges of both cycles. If $C \cap C' \neq \emptyset$ then both cycles C and C' are intersected cycles.

Then, following the algorithm, we must verify that the graph meets all the requirements for determining the $3-coloring$, which can be checked in polynomial time, and the construction of the 2-CF F_G is relative to $|F_G|$(number of clauses in the formula).

Furthermore, to determine the satisfiability of F_G can be solved in polynomial time in the order $O(poly(nc))$. Therefore, considering the worst case, the entire algorithm can be considered with a time complexity of order $O(poly(nc))$. Thus, our procedure has a polynomial time complexity on the the number of basic cycles nc of the graph.

Example exception

In figure 4, we show a graph which satisfies the previous conditions in order to apply the 3-coloring procedure.

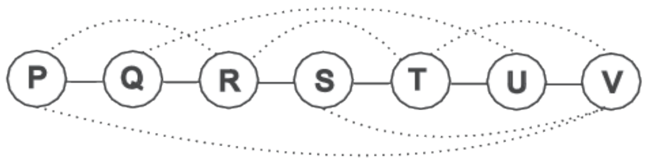

Figure 4 . A graph where its 2-CF is not satisfiable.

Applying the exposed method, we obtain the following 2-CF:

$(p \vee r) \wedge (\sim p \vee \sim r) \wedge (p \vee v) \wedge (\sim p \vee \sim v) \wedge (r \vee t) \wedge (\sim r \vee \sim t) \wedge (t \vee v) \vee (\sim t \vee \sim v) \wedge (q \vee u) \wedge (\sim q \vee \sim u) \wedge (\sim q \vee \sim p) \wedge (\sim q \vee \sim r) \wedge (\sim u \vee \sim t) \wedge (\sim u \vee \sim v) \wedge (\sim r \vee \sim s) \wedge (\sim s \vee \sim t) \wedge (\sim q \vee \sim u)$.
This 2-CF is not satisfiable. Althouh, the graph is 3-colorable. Figure 5 shows one of its possible 3-colorings.

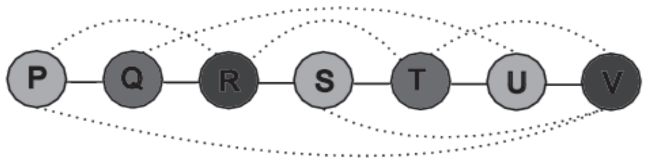

Figure 5 . A 3-coloring of the graph

CONCLUSION

We determine some basic patterns based on the basic cycles of a graph in order to determine the 3-colorability of the graph. We have designed an algorithm for the 3-coloring of a graph based on the same set of basic cycles of the input graph.

In our proposal, it is possible that for certain types of graphs, we can not find a model for the 2-CF associated to the graph although the input graph could be 3-coloreable. Since the 2-SAT problem is polynomially time solvable, then to search for reductions between the 3-coloring and the 2-SAT problem, would be very helpful in order to reduce the complexity time of the 3-coloring problem.

REFERENCES

[1] S. Arora, E. Chlamtac, M. Charikar. (2006). New Approximation Guarantee for Chromatic Number, *Proceedings STOC 2006* , May.

[2] Baase S., Gelder A. V. (2000). *Computer algorithms: Introduction to Design & Analysis* , Addison Wesley.

[3] Beigel R., Eppstein. (2005). D., 3-colouring in time $O(1.3289^n)$, *Journal of Algorithms* 54 (2), pp. 168--204.

[4] Blum A., Karger, D. (1997). An $O(n^{3/14})$-colouring algorithm for 3-colourable graphs, Inf. Proc. Lett. 61(1), pp.49-53.

[5] Byskov J.M. (2005). Exact Algorithms for graph colouring and exact satisifiability, *Phd thesis, University of Aarbus, Denmark.*

[6] Golumbic, M.C. (2004). *Algorithmic Graph Theory and Perfect Graphs, 2nd edn. North Holland.*

[7] Johnson D. (1985). The NP-Completeness Column: An Ongoing Guide, *Jour. of Algorithms* 6,434-451.

[8] Wigderson A. (1983). Improving the performance guarantee of approximate graph coloring, *Jour. of the ACM* ,30 (4):729-735.

[9] De Ita Guillermo, Bautista Cesar, Altamirano Luis. (2011). Solving 3-Colouring via 2-SAT *Lecture Notes in Computer Sciences* , Springer-Verlag, Vol. 6718, pp. 50-59.