



A sufficient condition for bicolorable hypergraphs

David Défossez

► To cite this version:

David Défossez. A sufficient condition for bicolorable hypergraphs. 2005 European Conference on Combinatorics, Graph Theory and Applications (EuroComb '05), 2005, Berlin, Germany. pp.21-24, 10.46298/dmtcs.3427 . hal-01184384

HAL Id: hal-01184384

<https://inria.hal.science/hal-01184384>

Submitted on 14 Aug 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A sufficient condition for bicolorable hypergraphs

David Défossez¹

¹Laboratoire Leibniz-Imag, 46 avenue Félix Viallet, 38031 Grenoble Cédex, France. E-mail: david.defossez@imag.fr

In this note we prove Sterboul's conjecture, that provides a sufficient condition for the bicolability of hypergraphs.

Keywords: hypergraphs, coloring, Sterboul's conjecture

In [2], Fournier and Las Vergnas gave a sufficient condition for the bicolability of hypergraphs. Their theorem was a weaker form of a conjecture due to Sterboul, that we prove here. These facts are reproduced in [1] and [3].

A hypergraph is a pair $H = (V, \mathcal{E})$, where the elements of V are the *vertices*, and the elements of \mathcal{E} are subsets of V and are called the *edges*. A function $c : V \rightarrow \{1, 2\}$ is called a *bipartition* of V , and for $x \in V$, we call $c(x)$ the *color* of x . If c is such that for any $e \in \mathcal{E}$ with $|e| \geq 2$ both colors occur, then c is called a *bicoloration* of H . If only one color occurs, the edge is said to be *monochromatic*. If a hypergraph admits a bicoloration we say that it is *bicolorable*.

A sequence $(x_1, e_1, x_2, \dots, e_k, x_1)$, where the e_i 's are distinct edges, the x_i 's are distinct vertices, and $k \geq 3$, is said to be a *cycle* if $x_i \in e_{i-1} \cap e_i$ for $i = 2, \dots, k$ and $x_1 \in e_1 \cap e_k$. A cycle is said to be *odd* if it has an odd number of edges.

An odd cycle $(x_1, e_1, x_2, \dots, e_k, x_1)$ such that two non-consecutive edges are disjoint and $|e_i \cap e_{i+1}| = 1$ for $i = 1, 2, \dots, k-1$, is called a *Sterboul cycle*. If a hypergraph H has no Sterboul cycle, it is said to be a *Sterboul hypergraph*.

Then we can word Sterboul's conjecture as follows:

Theorem 1 *If H is a Sterboul hypergraph, then H is bicolorable.*

Proof: The proof works by induction on the number of edges.

When the hypergraph has no edge, the theorem clearly holds.

The general step assumes that we have a hypergraph $H = (V, \mathcal{E})$ and $e_0 \in \mathcal{E}$ such that $H \setminus e_0 = (V, \mathcal{E} \setminus e_0)$ has a bicoloration $c : V \rightarrow \{1, 2\}$.

We can assume that e_0 has size at least 2, and that c leaves e_0 monochromatic, or else we have nothing to do.

Now we use the following algorithm to transform the bipartition c into a bicoloration of H . The algorithm switches successively the colors of some vertices of H that are contained in a monochromatic edge in the current bipartition. It constructs an arborescence $G_0 = (V_0, E_0)$ and a mapping $g : V_0 \rightarrow \mathcal{E}$ that keep track of the running of the algorithm: the vertices of G_0 are those whose colors were switched, and g associates a vertex with the monochromatic edge that caused its color switch.

The vertices are chosen with a DFS (Depth-First Search) method, and to do so the algorithm uses a LIFO (Last In First Out) stack \mathcal{P} that contains the set of vertices whose colors have been switched, and so that might be in a monochromatic edge. $\text{top}(\mathcal{P})$ returns the last vertex entered in \mathcal{P} ; $\text{drop}(\mathcal{P})$ removes $\text{top}(\mathcal{P})$ out of \mathcal{P} ; and $\text{put}(x, \mathcal{P})$ enters a new vertex x in \mathcal{P} .

INPUT: A hypergraph $H = (V, \mathcal{E})$ and a bipartition c such that $e_0 \in \mathcal{E}$ is the only monochromatic edge.

OUTPUT: A bicoloration of H or an Error message only if H is not a Sterboul hypergraph.

```

let  $x_0 \in e_0$ 
 $G_0 := (\{x_0\}, \emptyset)$ 
 $g(x_0) := e_0$ 
switch  $c(x_0)$ 
put( $x_0, \mathcal{P}$ )
While  $\mathcal{P} \neq \emptyset$  do
  let  $v = \text{top}(\mathcal{P})$ 
  If there exists  $e \in \mathcal{E}$ ,  $|e| \geq 2$ , monochromatic such that  $v \in e$  then
    If  $e \setminus V_0 = \emptyset$  then
      return Error
    else
      let  $w \in e \setminus V_0$ 
       $V_0 := V_0 \cup w$  ;  $E_0 := E_0 \cup (vw)$ 
       $g(w) := e$ 
      switch  $c(w)$ 
      put( $w, \mathcal{P}$ )
    end If
  else
    drop( $\mathcal{P}$ )
  end If
end While

```

First we remark that G_0 is indeed an arborescence since the end point of each new arc of G_0 is a new vertex. Then for a given $x \in V_0$ there is a unique path in G_0 from x_0 to x . Moreover when x is at the top of \mathcal{P} , then \mathcal{P} contains exactly the vertices of that path (because \mathcal{P} is a LIFO stack).

We can also remark that if the algorithm does not return **Error**, then at each iteration either a new vertex is put into \mathcal{P} , or a vertex is dropped out of \mathcal{P} . Since a vertex appears at most once in G_0 and thus can be put at most once in \mathcal{P} , we have at most $2|V|$ iterations, and the algorithm ends.

We note $\mathcal{P}^{(i)}, G_0^{(i)} = (V_0^{(i)}, E_0^{(i)})$, $c^{(i)}, g^{(i)}$ the values of $\mathcal{P}, G_0 = (V_0, E_0)$, c, g (respectively) at the beginning of the i -th iteration. We also note $c^{(0)}$ the original bipartition (which is different from $c^{(1)}$ because of the switch of $c(x_0)$).

To prove the validity of the algorithm, we have to prove that:

- **ERROR** cannot be returned if H is a Sterboul hypergraph.
- The output of the algorithm if no **ERROR** occurs is a bicolouration.

Before proving those points, we claim the following:

Claim 1 *Suppose that H is a Sterboul hypergraph. Consider the beginning of the i -th iteration. Let $\mathcal{P}^{(i)} = (x_k \dots x_0)$, and $e_j = g^{(i)}(x_j)$ for $j = 0, \dots, k$. Then we have:*

- (a) *For each $j = 0, \dots, k$, x_j is the only vertex of its color in e_j .*
- (b) *For each $j = 0, \dots, k-1$ we have $e_j \cap e_{j+1} = \{x_j\}$.*
- (c) *Two non-consecutive edges are disjoint.*

Proof: The proof works by induction on i .

For $i = 1$ the claim clearly holds since $\mathcal{P}^{(1)} = (x_0)$.

We now consider $i \geq 1$ and we suppose the claim holds at iteration i . We are going to prove that it also holds at iteration $i+1$. Let $\mathcal{P}^{(i)} = (x_k \dots x_0)$, and $e_j = g^{(i)}(x_j)$ for $j = 0, \dots, k$.

If during the i -th iteration the algorithm dropped x_k out of \mathcal{P} (that is $\mathcal{P}^{(i+1)} = (x_{k-1} \dots x_0)$), the claim clearly holds at iteration $i+1$. Thus we assume that the algorithm found $e_{k+1} \in \mathcal{E}$ with $x_k \in e_{k+1}$ that is monochromatic for $c^{(i)}$, and $x_{k+1} \in e_{k+1} \setminus V_0^{(i)}$ (because we assume that there is a $(i+1)$ -th iteration or else the claim is true) so that $\mathcal{P}^{(i+1)} = (x_{k+1} x_k \dots x_0)$.

Since (a) holds at iteration i , we know that if $w \in e_k \setminus x_k$ then $c^{(i)}(w) \neq c^{(i)}(x_k)$. As $x_k \in e_{k+1}$ and e_{k+1} is monochromatic for $c^{(i)}$, then $e_k \cap e_{k+1} = \{x_k\}$ and (b) holds at iteration $i+1$.

Suppose $j_0 = \max\{0 \leq j \leq k-1 \mid e_j \cap e_{k+1} \neq \emptyset\}$ exists, and let $y \in e_{j_0} \cap e_{k+1}$. If $k - j_0$ is odd, then $(y, e_{j_0}, x_{j_0}, \dots, e_{k+1}, y)$ is a Sterboul cycle (because (c) holds at iteration i and (b) holds at iteration $i+1$), so $k - j_0$ is even. But then since (a) holds at iteration i , we have $c^{(i)}(y) \neq c^{(i)}(x_{j_0})$ ($y \neq x_{j_0}$ by definition of j_0), $c^{(i)}(x_{j_0}) \neq c^{(i)}(x_{j_0+1})$, ..., $c^{(i)}(x_{k-1}) \neq c^{(i)}(x_k)$ and then $c^{(i)}(y) \neq c^{(i)}(x_k)$, which is impossible because e_{k+1} is monochromatic for $c^{(i)}$. Hence j_0 does not exist, and (c) holds at iteration $i+1$.

Thus $x_{k+1} \notin e_j$ for all $j = 0, \dots, k$. Since the only color switch done during the i -th iteration concerns x_{k+1} , then (a) holds at iteration $i+1$.

This achieves to prove the claim. □

Now we are able to prove the validity of the algorithm.

- Suppose that H is a Sterboul hypergraph. Consider an iteration i , and let $\mathcal{P}^{(i)} = (x_k \dots x_0)$. If $k = 1$ then from (b) of the claim we have $x_1 \notin e_0$. If $k \geq 2$ then from (c) of the claim we also have $x_k \notin e_0$. This proves that we always have $e_0 \cap V_0 = \{x_0\}$.

If **ERROR** is returned, it means that at a given iteration i_0 , the algorithm found an edge e monochromatic for $c^{(i_0)}$ such that $e \setminus V_0^{(i_0)} = \emptyset$. Then e was also monochromatic for $c^{(0)}$, but e_0 was the only such edge,

so we have a contradiction because we have just seen that we must have $e_0 \cap V_0^{(i_0)} = \{x_0\}$. Thus if H is a Sterboul hypergraph, `ERROR` cannot be returned.

- Finally, if the bipartition obtained by the algorithm is not a bicolouration then we have some $e \in \mathcal{E}$ that is monochromatic. Consider i_0 the last iteration during which the algorithm dropped a vertex of e out of \mathcal{P} (i_0 exists or else e was monochromatic with $c^{(0)}$, but e_0 was the only such edge and e_0 is not monochromatic at the end of the algorithm), and let y_0 be that vertex. Then e was not monochromatic with $c^{(i_0)}$ or else the algorithm would have considered e during the iteration i_0 instead of dropping y_0 . But since no color switch concerning a vertex of e occurs afterwards (by choice of i_0), we have a contradiction. Thus the final bipartition is a bicolouration.

So the algorithm is correct and the theorem is proved. \square

We can slightly modify the algorithm so that it gives a Sterboul cycle instead of just returning `ERROR` when the hypergraph is not Sterboul (in order to have a certificate that the hypergraph is not Sterboul).

To do so we just have to check at each iteration that the properties of Claim 1 still hold. If not it means that the monochromatic edge considered intersects the path induced by the stack, and a Sterboul cycle can be easily found.

Our algorithm finds a bicolouration for Sterboul hypergraphs in polynomial time. However it cannot be used to recognize bicolourable hypergraphs (since a Sterboul hypergraph may be bicolourable) and neither to recognize Sterboul hypergraphs (since it may happen that it gives a bicolouration for a hypergraph that is not Sterboul).

The problem of recognizing bicolourable hypergraphs is well-known to be NP-complete [4]. But we leave the following question open: what is the complexity of recognizing Sterboul hypergraphs?

References

- [1] P. Duchet, *Hypergraphs*, in R. Graham, M. Grötschel, L. Lovász, editors, *Handbook of Combinatorics* (1995), chapter 7, 381-432.
- [2] J.-C. Fournier, M. Las Vergnas, *Une classe d'hypergraphes bichromatiques II*, *Discrete Mathematics* 7 (1974) 99-106.
- [3] J.-C. Fournier, M. Las Vergnas, *A class of bichromatic hypergraphs*, *Annals of Discrete Mathematics* 21 (1984) 21-27.
- [4] L. Lovász, *Coverings and colorings of hypergraphs*, *Proc. 4th Southeastern Conference on Combinatorics, Graph Theory, and Computing*, Utilitas Mathematica Publishing, Winnipeg (1973) 3-12.