The null label problem and the complexity of the reconstruction of $I_2(H)$

Niccolò Di Marco[1]

[1] Dipartimento di Matematica e Informatica,
Università di Firenze

Preliminary definitions

- A *graph* is a pair of sets $G = (V, E)$ where V is the set of vertices and $E \subseteq V \times V$ is the set of edges.

Preliminary definitions

- A *graph* is a pair of sets $G = (V, E)$ where V is the set of vertices and $E \subseteq V \times V$ is the set of edges.
- Generalization: a *hypergraph* is a pair $H = (V, E)$ where V is the set of vertices and $E \subset \mathcal{P}(V)$ is the set of hyperedges (briefly, edges);

Preliminary definitions

- A *graph* is a pair of sets $G = (V, E)$ where V is the set of vertices and $E \subseteq V \times V$ is the set of edges.
- Generalization: a *hypergraph* is a pair $H = (V, E)$ where V is the set of vertices and $E \subset \mathcal{P}(V)$ is the set of hyperedges (briefly, edges);
- The *degree* of a vertex v is the number of (hyper)edges in which v belongs;

Preliminary definitions

- A *graph* is a pair of sets $G = (V, E)$ where V is the set of vertices and $E \subseteq V \times V$ is the set of edges.
- Generalization: a *hypergraph* is a pair $H = (V, E)$ where V is the set of vertices and $E \subset \mathcal{P}(V)$ is the set of hyperedges (briefly, edges);
- The *degree* of a vertex v is the number of (hyper)edges in which v belongs;
- a hypergraph H is $k-$uniform if its edges have fixed cardinality k. In particular a graph is a $2-$uniform hypergraph. In the following we denote them $k-$hypergraphs;

Preliminary definitions

- A *graph* is a pair of sets $G = (V, E)$ where V is the set of vertices and $E \subseteq V \times V$ is the set of edges.
- Generalization: a *hypergraph* is a pair $H = (V, E)$ where V is the set of vertices and $E \subset \mathcal{P}(V)$ is the set of hyperedges (briefly, edges);
- The *degree* of a vertex v is the number of (hyper)edges in which v belongs;
- a hypergraph H is $k-$uniform if its edges have fixed cardinality k. In particular a graph is a $2-$uniform hypergraph. In the following we denote them $k-$hypergraphs;
- a hypergraph is *even* if every vertex has even degree .

Null label

Given a (hyper)graph, we can assign a label l with labels ± 1 to each (hyper)edge, resulting in positive and negative (hyper)edges.

Null label

Given a (hyper)graph, we can assign a label l with labels ± 1 to each (hyper)edge, resulting in positive and negative (hyper)edges.

The *positive* signed degree of a vertex v is the number of positive (hyper)edges in which it compares and it is denoted as $d_l^+(v)$. A similar definition holds for the *negative* signed degree. Thus, we define the signed degree of a vertex v as

$$d_l(v) = d_l^+(v) - d_l^-(v).$$

Null label

Given a (hyper)graph, we can assign a label l with labels ± 1 to each (hyper)edge, resulting in positive and negative (hyper)edges.

The *positive* signed degree of a vertex v is the number of positive (hyper)edges in which it compares and it is denoted as $d_l^+(v)$. A similar definition holds for the *negative* signed degree. Thus, we define the signed degree of a vertex v as

$$d_l(v) = d_l^+(v) - d_l^-(v).$$

Definition (Null hypergraph)

An assignment of ± 1 to the (hyper)edges of a (hyper)graph is a *null label* if $d(v) = 0$, for all vertices $v \in V$. A (hyper)graph with a null labelling is said to be a *null (hyper)graph*.

Null label

An obvious necessary condition for a (hyper)graph to have a null labelling is that each vertex must have even degree, i.e., it is an even (hyper)graph. We state the general problem below.

Null label

An obvious necessary condition for a (hyper)graph to have a null labelling is that each vertex must have even degree, i.e., it is an even (hyper)graph. We state the general problem below.

Hypergraph Null Labelling Problem: let H be a connected, even $k-$hypergraph. When can ± 1 be assigned to the hyperedges of H to produce a null-labelled $k-$hypergraph?

We study the problem in the case of $3-$hypergraphs.

Null label on graphs

The null label problem is completely solved for $2-$hypergraphs (i.e. graphs), thanks to the following Proposition.

Proposition

A graph G has a null labelling if and only if every connected component is an Eulerian graph with an even number of edges.

Null label on graphs

The null label problem is completely solved for 2−hypergraphs (i.e. graphs), thanks to the following Proposition.

Proposition

A graph G has a null labelling if and only if every connected component is an Eulerian graph with an even number of edges.

Moving to 3−hypegraphs the situation becomes more complex. One of the first results use the notion of *intersection graph*.

Definition

Let $H = (V, E)$ be a 3−hypergraph. The *intersection graph* of H, denoted as $I(H)$, is a graph in which the nodes are the hyperedges of H and two hyperedges are adjacent if their intersection is non-empty.

Previous results

The following result holds.

Theorem

Let H be a connected, even $3-$hypergraph, in which every vertex has degree two. Then H has a null labelling if and only if $I(H)$ is bipartite.

Previous results

The following result holds.

Theorem

Let H be a connected, even $3-$hypergraph, in which every vertex has degree two. Then H has a null labelling if and only if $I(H)$ is bipartite.

However, $I(H)$ is not useful in general.

Consider the following $3-$hypergraphs H_1 and H_2 on six vertices and whose hyperedges, arranged in matrix form, are:

$$H_1 = \begin{bmatrix} 1 & 2 & 3 \\ 1 & 4 & 5 \\ 2 & 4 & 6 \\ 3 & 5 & 6 \end{bmatrix} \qquad H_2 = \begin{bmatrix} 1 & 2 & 5 \\ 2 & 3 & 5 \\ 2 & 3 & 4 \\ 1 & 2 & 4 \end{bmatrix}$$

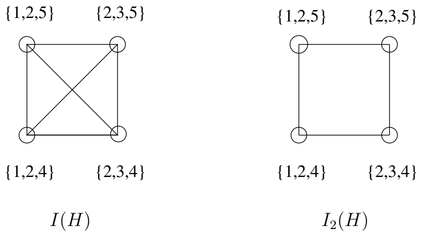
It is easy to check that the vector of labels $l = (1, -1, 1, -1)$ is a null label for H_2, while H_1 has no null labelling. However, H_1 and H_2 have the same intersection graph K_4, i.e. the complete graph on four vertices.

2−intersection graph

Relying on this fact, we decide to use the notion of *2-intersection graph*.

Definition

Let $H = (V, E)$ be a 3−hypergraph. The *2-intersection graph* $I_2(H)$ is a graph in which the nodes are the hyperedges of H and two hyperedges are adjacent if and only if they a share a common pair of vertices.



Hamiltonian cycle and null label

Idea: a connected, even graph G is Eulerian. An Euler tour in G corresponds to a Hamiltonian cycle in its line graph $L(G)$, and conversely. Furthermore, by alternate labelling this cycle we find a null labelling of G.

Thus, Hamiltonian cycles in $L(G)$ can be used to determine null labellings of G.

Hamiltonian cycle and null label

Idea: a connected, even graph G is Eulerian. An Euler tour in G corresponds to a Hamiltonian cycle in its line graph $L(G)$, and conversely. Furthermore, by alternate labelling this cycle we find a null labelling of G.

Thus, Hamiltonian cycles in $L(G)$ can be used to determine null labellings of G.

Since the concept of $I_2(H)$ and $L(G)$ are very similar, we extend the previous idea and prove that, starting from an Hamiltonian cycle in $I_2(H)$ and alternate labelling its nodes (i.e. hyperedges of H), we can determine a null label of H.

Hamiltonian cycle and null label

Consider the 3-hypergraph $H = (V, E)$ on six vertices and $E = \{e_1, \dots, e_8\}$, where $e_1 = \{1, 2, 3\}, e_2 = \{1, 2, 4\}, e_3 = \{1, 2, 5\}, e_4 = \{1, 2, 6\}, e_5 = \{1, 3, 4\}, e_6 = \{1, 3, 5\}, e_7 = \{2, 3, 5\}, e_8 = \{2, 5, 6\}$.

The related 2-intersection graph $I_2(H)$ in Fig. 1 has the Hamiltonian cycle $C_1 = (v_{e_1}, v_{e_3}, v_{e_2}, v_{e_4}, v_{e_8}, v_{e_7}, v_{e_6}, v_{e_5}, v_{e_1})$

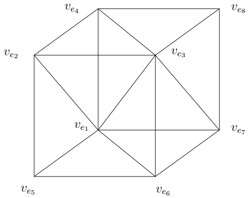


Figure: The 2-intersection graph of the 3-hypergraph considered in the Example.

Hamiltonian cycle and null label

It is easy to check that alternately labelling ± 1 the vertices of C_1, starting with $+1$, we obtain the null labelling $l_1 = (1, 1, -1, -1, -1, 1, -1, 1)$ on the eight hyperedges of H such that $l_1(i)$ is the label of e_i, with $1 \leq i \leq 8$.

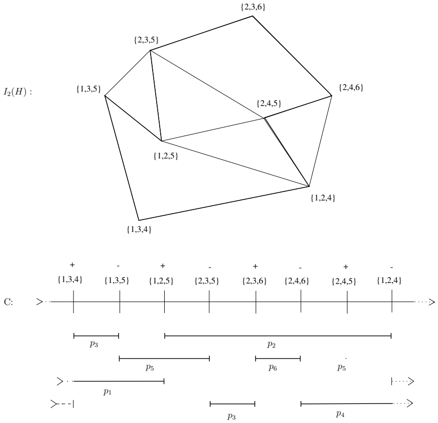
Hamiltonian cycle and null label

It is easy to check that alternately labelling ± 1 the vertices of C_1, starting with $+1$, we obtain the null labelling $l_1 = (1, 1, -1, -1, -1, 1, -1, 1)$ on the eight hyperedges of H such that $l_1(i)$ is the label of e_i, with $1 \leq i \leq 8$.

Unfortunately, not every Hamiltonian cycle provides a null labelling. A second Hamiltonian cycle $C_2 = (v_{e_1}, v_{e_2}, v_{e_3}, v_{e_4}, v_{e_8}, v_{e_7}, v_{e_6}, v_{e_5}, v_{e_1})$ exists such that the alternating labelling $l_2 = (1, -1, 1, -1, -1, 1, -1, 1)$ is not null on H, as $d(v_4) = -2$ and $d(v_5) = +2$.

Notation

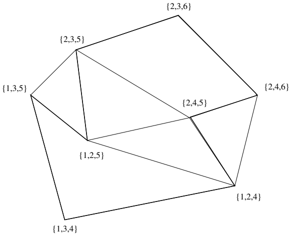
Suppose that v_{e_i} and v_{e_j} are two consecutive vertices of the alternate labelled Hamiltonian cycle C of $I_2(H)$, with $e_i = \{u, x, y\}$ and $e_j = \{v, x, y\}$. We see that $v \notin e_i$. There may be several consecutive vertices of C that contain v. Denote by $p_v = (v_{e_{j_1}}, \ldots, v_{e_{j_k}})$ the longest sub-path of C starting in v_{e_j} such that every vertex of p_v contains v.



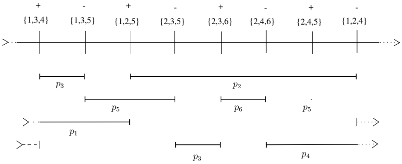
Notation

Let $l(p_v)$ denote the labels of the vertices of p_v, and let $\sigma(l(p_v))$ denote the sum of the elements of $l(p_v)$. Moreover, let $|p_v|$ denote the length $k-1$ of p_v.

$I_2(H)$:

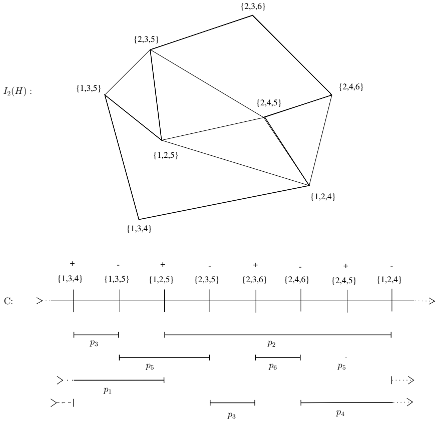


C:



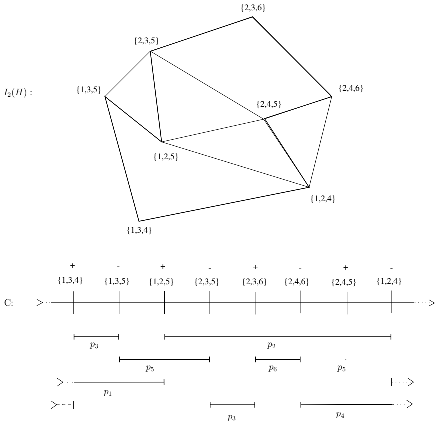
Notation

Moreover, we define the *distance* between two paths p_u and p_v as the distance along C between the last point of p_u and the first point of p_v. Finally, given a path p_u, $next(p_u)$ is the path beginning in the first vertex following the last vertex of p_u.



Notation

Note: in general, C may contain several different sub-paths of the form p_v, for each vertex v; we indicate them by $p_v^1, \ldots, p_v^n$.



Subpaths' properties

Property

Given an alternating labelling ± 1 on the vertices of a Hamiltonian cycle C of $I_2(H)$. For each sub-path $p_v = (v_{e_{j_1}}, \ldots, v_{e_{j_k}})$, the following holds:

- if p_v has odd length, then $\sigma(l(p_v)) = 0$, so that the labels of the hyperedges $e_{j_1}, \ldots, e_{j_k}$ containing v sum to zero in H. In this case the first and the last vertex of p_v have different labels;

Subpaths' properties

Property

Given an alternating labelling ± 1 on the vertices of a Hamiltonian cycle C of $I_2(H)$. For each sub-path $p_v = (v_{e_{j_1}}, \ldots, v_{e_{j_k}})$, the following holds:

- if p_v has odd length, then $\sigma(l(p_v)) = 0$, so that the labels of the hyperedges $e_{j_1}, \ldots, e_{j_k}$ containing v sum to zero in H. In this case the first and the last vertex of p_v have different labels;
- if p_v has even length, then $\sigma(l(p_v)) \neq 0$ and the sum of the labels of the hyperedges $e_{j_1}, \ldots, e_{j_k}$ containing v contribute $+1$ or -1 to the signed degree of v. In this case, the extremal vertices of p_v have the same label.

Subpaths' properties

Lemma

Let H be an even 3-hypergraph and $I_2(H)$ its 2-intersection graph. If $I_2(H)$ has a Hamiltonian cycle C, an alternating ± 1 labelling $l(C)$ defines a null label of H if and only if, for each $v \in V$:

i) each subpath p_v has odd length; OR

ii) the number of subpaths of v having even length is even and the sum of their labels is zero.

Algorithm

Let C be an Hamiltonian cycle not satisfying the conditions of Lemma 1. We define an algorithm to obtain a null label from C. It relies on the *Switch*() operator defined as follows:

Definition

Given two sub-paths $p_u = (v_{e_{i_1}}, \ldots, v_{e_{i_k}})$ and $p_v = (v_{e_{j_1}}, \ldots, v_{e_{j_{k'}}})$, where $p_v = next(p_u)$, and $e_{i_k} \neq e_{j_1}$, the operator $Switch(p_u, p_v)$ produces a new labelling $l'(C)$ by changing the signs of e_{i_k} and e_{j_1}: $l'(e_{i_k}) = -l(e_{i_k})$ and $l'(e_{j_1}) = -l(e_{j_1})$ and keeping the remaining labels of $l(C)$ unchanged.

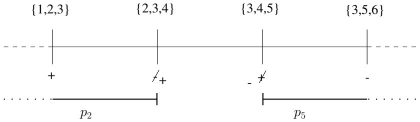
Algorithm

Let C be an Hamiltonian cycle not satisfying the conditions of Lemma 1. We define an algorithm to obtain a null label from C. It relies on the *Switch*() operator defined as follows:

Definition

Given two sub-paths $p_u = (v_{e_{i_1}}, \ldots, v_{e_{i_k}})$ and $p_v = (v_{e_{j_1}}, \ldots, v_{e_{j_{k'}}})$, where $p_v = \mathit{next}(p_u)$, and $e_{i_k} \neq e_{j_1}$, the operator $\mathit{Switch}(p_u, p_v)$ produces a new labelling $l'(C)$ by changing the signs of e_{i_k} and e_{j_1}: $l'(e_{i_k}) = -l(e_{i_k})$ and $l'(e_{j_1}) = -l(e_{j_1})$ and keeping the remaining labels of $l(C)$ unchanged.

This is an example of $\mathit{Switch}(p_2, p_5)$ between the two consecutive paths p_2 and p_5, i.e., such that $p_5 = \mathit{next}(p_2)$.



Algorithm

We will start with an alternating labelling $l(C)$ and gradually change it using $Switch()$.

Property

Let H be a even 3-hypergraph, C a Hamiltonian cycle of $I_2(H)$ and l a ± 1 labelling of C. Consider a sub-path p_u of C whose last element v_{e_i} with label $+1$, and the sub-path $p_v = next(p_u)$ whose first element v_{e_j} with label -1. The operator $Switch(p_u, p_v)$ modifies l into l' so that $d_{l'}(u) = d_l(u) - 2$, $d_{l'}(v) = d_l(v) + 2$ and all the remaining signed degrees are left unchanged.

Algorithm

We will start with an alternating labelling $l(C)$ and gradually change it using $\mathit{Switch}()$.

Property

Let H be a even 3-hypergraph, C a Hamiltonian cycle of $I_2(H)$ and l a ± 1 labelling of C. Consider a sub-path p_u of C whose last element v_{e_i} with label $+1$, and the sub-path $p_v = \mathit{next}(p_u)$ whose first element v_{e_j} with label -1. The operator $\mathit{Switch}(p_u, p_v)$ modifies l into l' so that $d_{l'}(u) = d_l(u) - 2$, $d_{l'}(v) = d_l(v) + 2$ and all the remaining signed degrees are left unchanged.

Proof.

Without loss of generality, assume that $e_i = \{u, x, y\}$ and $e_j = \{v, x, y\}$. It is immediate that the change of the opposite labels of e_i and e_j keeps the signed degrees of x and y, while it subtracts 2 from u and adds 2 to v. A symmetric result holds. □

Algorithm

The algorithm $Balance()$ modifies a labelling $l(C)$ of a Hamiltonian cycle C of $I_2(H)$ in order to change the signed degree of two input vertices u and v of H, if possible, otherwise it gives failure.

Algorithm

The algorithm $Balance()$ modifies a labelling $l(C)$ of a Hamiltonian cycle C of $I_2(H)$ in order to change the signed degree of two input vertices u and v of H, if possible, otherwise it gives failure.

$Balance()$ uses consecutive iterations of $Switch()$ and it can be summarized in the following steps:

1. choose a subpath p_u^i (suppose $d_l(u) = 2$ and $d_l(v) = -2$);
2. if $|p_u^i|$ is even, suppose $p_t = next(p_u)$. Apply $Switch(p_u, p_t)$. Now $d_l'(u) = 0$ and and $d_l'(t) = d_l(t) + 2$. Set $t = u$ and repeat step 2)
3. if $|p_u^i|$ is odd, we know that there exists another subpath p_u^j such that $\sigma(p_u^j) = 1$ (otherwise is not possible that $d_l(u) = 2$). Choose p_u^j and apply step 2).

The procedure stops when we (eventually) obtain $d(u) = d(v) = 0$

Correctness of *Balance*

First, we proved that $Balance()$ computes a null labelling starting from the alternating labelling $l(C)$ in the easiest case of having only two signed degrees u and v different from zero, in particular $+2$ and -2, respectively.

Lemma

Let H be a 3-hypergraph, C a Hamiltonian cycle of $I_2(H)$, and l an alternating labelling of C. If u and v are the only nodes of H with signed degree different from zero, in particular $d_l(u) = +2$ and $d_l(v) = -2$, then $Balance(u, v, l(C))$ returns a null labelling $l'(C)$ of H.

Correctness of *Balance*

First, we proved that $Balance()$ computes a null labelling starting from the alternating labelling $l(C)$ in the easiest case of having only two signed degrees u and v different from zero, in particular $+2$ and -2, respectively.

Lemma

Let H be a 3-hypergraph, C a Hamiltonian cycle of $I_2(H)$, and l an alternating labelling of C. If u and v are the only nodes of H with signed degree different from zero, in particular $d_l(u) = +2$ and $d_l(v) = -2$, then $Balance(u, v, l(C))$ returns a null labelling $l'(C)$ of H.

The key of the proof relies on the fact that $Switch()$ **always** changes $+$ with a $-$.

Correctness of *Balance*

The previous Lemma can be generalized thanks to two final results.

Lemma

Let $H = (V, E)$ be a 3-hypergraph, C a Hamiltonian cycle of $I_2(H)$ and l an alternating labelling of C. If v_1 and v_2 are the only nodes of H with signed degree different from zero with respect to l, say $d_l(u) = +2k$ and $d_l(v) = -2k$, where $k \geq 1$, then H admits a null labelling.

Correctness of *Balance*

The previous Lemma can be generalized thanks to two final results.

Lemma

Let $H = (V, E)$ be a 3-hypergraph, C a Hamiltonian cycle of $I_2(H)$ and l an alternating labelling of C. If v_1 and v_2 are the only nodes of H with signed degree different from zero with respect to l, say $d_l(u) = +2k$ and $d_l(v) = -2k$, where $k \geq 1$, then H admits a null labelling.

Theorem

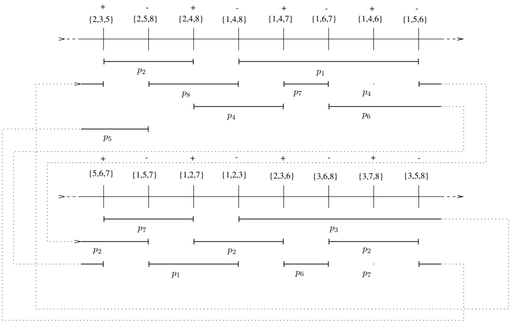
Let H be a 3-hypergraph. If the 2-intersection graph $I_2(H)$ is Hamiltonian, then H admits a null labelling.

The proof of these last results is simply based on a multiple iteration of *Balance*.

Example

Consider the following 3−hypergraph $H = (V, E)$ with $V = \{1, \ldots, 8\}$ and E = {{2,3,5}, {2,5,8}, {2,4,8}, {1,4,8}, {1,4,7}, {1,6,7}, {1,4,6}, {1,5,6}, {5,6,7}, {1,5,7}, {1,2,7}, {1,2,3}, {2,3,6}, {3,6,8}, {3,7,8}, {3,5,8}}

This figure shows a Hamiltonian cycle C of $I_2(H)$, and one of its alternating labellings $l(C)$.

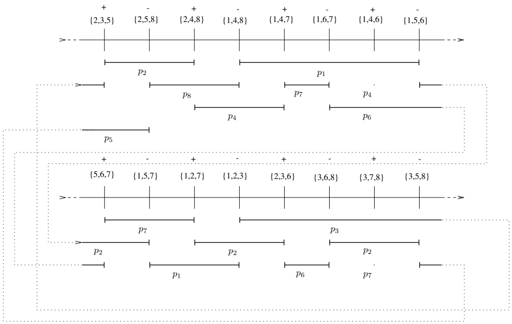


Example

The chosen labelling is not a null labelling of H. The vector of the signed degrees of the vertices of H is

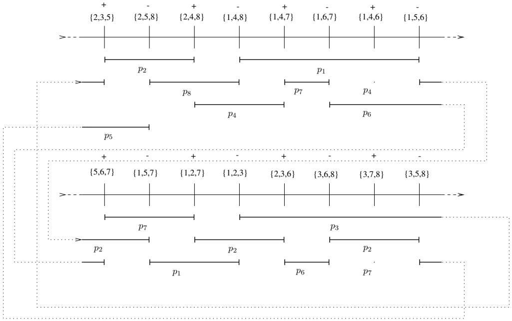
$$d = (-2, 2, 0, 2, -2, 0, 2, -2).$$

Let us perform a sequence of runs of $Balance()$ to compute a null labelling of H starting from $l(C)$.



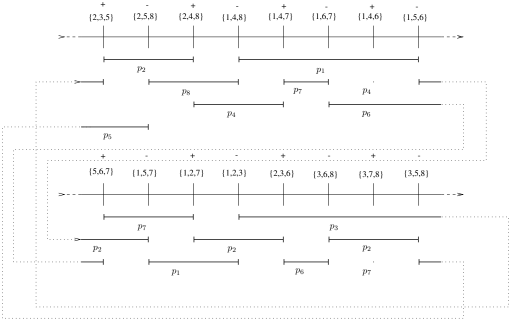
Example

Let us start, as an example, the run $Balance(2, v, l(C))$ in the p_2 sub-path having $\{2, 3, 5\}$ as first element. It calls $Switch(p_2, p_1)$, with $p_1 = next(p_2)$ and $|p_1|$ even. Since $d_l(1) = -2$, we perform the choice $v = 1$, and the switchings of $\{2, 4, 8\}$ and $\{1, 4, 8\}$ leading to the labelling $l^1(C)$ such that $d_{l^1}(1) = d_{l^1}(2) = 0$, leaving the remaining labels unchanged.



Example

Choose the vertex 7 such that $d_{l^1}(7) = +2$ and run $Balance(7, v, l^1(C))$ with the starting p_7 sub-path whose first element is $\{5,6,7\}$. The sub-path $p_3 = next(p_7)$ has odd length so the labels of $\{1,2,7\}$ and $\{1,2,3\}$ are switched and we obtain $d(7) = 0$ and $d(3) = +2$. Now $p_8 = next(p_3)$ and the labels of $\{2,3,5\}$ and $\{2,5,8\}$ are switched obtaining $d(3) = d(8) = 0$. Since $|p_8|$ is even, the run $Balance(7, v, l^1(C))$ ends setting $v = 8$. A new labelling $l^2(C)$ is returned as output.

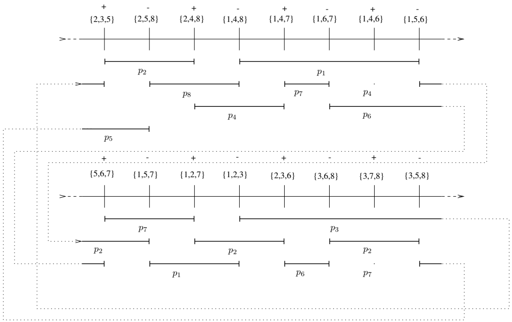


Example

Only the vertices 4 and 5 are left. A last run of $Balance(4, 5, l^2(C))$ is performed. Taking the p_4 subpath containing only $\{1, 4, 6\}$, we have $p_5 = next(p_4)$ with $|p_5|$ even. Therefore, switching the sign of $\{1, 4, 6\}$ and $\{1, 5, 6\}$ we obtain a new labelling l^3 such that $d_{l^3}(4) = d_{l^3}(5) = 0$ and $Balance(4, 5, l^2(C))$ ends. Therefore, the labelling

$$l^3 = (-1, 1, -1, 1, 1, -1, -1, 1, 1, -1, -1, 1, 1, -1, 1, -1)$$

is a null labelling of H.



The importance of the $2-$intersection graph in the null labelling problem lead to the study of the following problem.

Problem: given a graph G, it is possible to decide in polynomial time the existence of a 3-hypergraph H such that $G = I_2(H)$?

The importance of the $2-$intersection graph in the null labelling problem lead to the study of the following problem.

Problem: given a graph G, it is possible to decide in polynomial time the existence of a 3-hypergraph H such that $G = I_2(H)$?

In general, we say that a graph G has the $2-$*intersection property* if it is the $2-$intersection graph of some $3-$hypergraph H.

Properties of $I_2(H)$

Property

The edges of a 3-hypergraph H sharing the same couple of vertices originate a clique in the 2-intersection graph $I_2(H)$.

Properties of $I_2(H)$

Property

The edges of a 3-hypergraph H sharing the same couple of vertices originate a clique in the 2-intersection graph $I_2(H)$.

However, this Property does not characterize the cliques of $I_2(H)$ since there may appear triangles (K_3 cliques) whose edges do not share a common label. More precisely, the configurations $v_1 = \{x, y, z\}$, $v_2 = \{x, y, t\}$, and $v_3 = \{x, z, t\}$ and $v_1 = \{x, y, z\}$, $v_2 = \{x, y, t\}$, and $v_3 = \{x, y, k\}$ are both triangle in $I_2(H)$.

Properties of $I_2(H)$

Property

The edges of a 3-hypergraph H sharing the same couple of vertices originate a clique in the 2-intersection graph $I_2(H)$.

However, this Property does not characterize the cliques of $I_2(H)$ since there may appear triangles (K_3 cliques) whose edges do not share a common label. More precisely, the configurations $v_1 = \{x, y, z\}$, $v_2 = \{x, y, t\}$, and $v_3 = \{x, z, t\}$ and $v_1 = \{x, y, z\}$, $v_2 = \{x, y, t\}$, and $v_3 = \{x, y, k\}$ are both triangle in $I_2(H)$.

In order to distinguish them, we indicate the first and the second ones with T and K triangles, respectively.

A similar situation holds for K_4 cliques, since they can arise from T or K triangles. Maintaining the introduced notation, we indicate square configuration with S, and we differentiate it from the K square, i.e., the 4-clique whose edges have a common label.

A similar situation holds for K_4 cliques, since they can arise from T or K triangles. Maintaining the introduced notation, we indicate square configuration with S, and we differentiate it from the K square, i.e., the 4-clique whose edges have a common label.

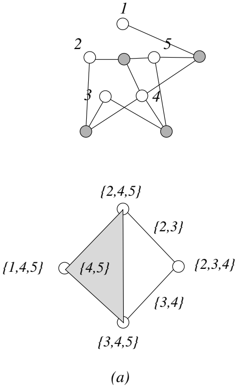


Figure: (a) example of adjacent T and K triangles. (b) example of S square.

Property

The biggest clique that admit a label without a common couple is a S square.

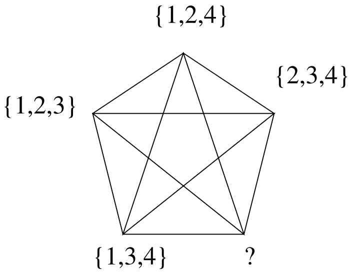


Figure: Labelling a K_5 clique without a common couple lead to an unsatisfactory labeling.

Lemma

If G is has the $2-$intersection property, then every node of G must belong to at most 3 maximal cliques or 5 maximal cliques two of which being non-adjacent T triangles.

Lemma

If G is has the $2-$intersection property, then every node of G must belong to at most 3 maximal cliques or 5 maximal cliques two of which being non-adjacent T triangles.

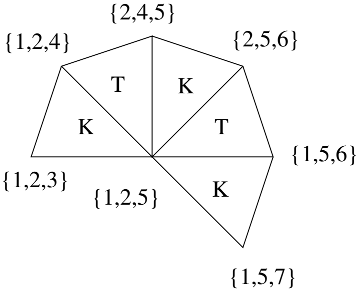


Figure: Reconstruction of the configuration in which a point belongs to 5 triangles.

Complexity of the problem

We proved that is NP-complete to determine whether an arbitrary graph G is the 2-intersection graph of a 3-hypergraph. We reduced the problem to the following variant of the 3-SAT.

Complexity of the problem

We proved that is NP-complete to determine whether an arbitrary graph G is the 2-intersection graph of a 3-hypergraph. We reduced the problem to the following variant of the 3-SAT.

MAX-3-SAT: Consider a set U of variables involved into C clauses over U such that each clause $c \in C$ has $|c| = 3$ and the literals related to a variable appear three times at most. Is there a satisfying truth assignment for C?

Complexity of the problem

We proved that is NP-complete to determine whether an arbitrary graph G is the 2-intersection graph of a 3-hypergraph. We reduced the problem to the following variant of the 3-SAT.

MAX-3-SAT: Consider a set U of variables involved into C clauses over U such that each clause $c \in C$ has $|c| = 3$ and the literals related to a variable appear three times at most. Is there a satisfying truth assignment for C?

Given an instance C of MAX-3-SAT, we construct a graph G_C so that there is a solution of the MAX-3-SAT instance if and only if G_C is a 2-intersection graph.

First results

Property

If two triangles intersecting in one vertex have the 2-intersection property, and there are no edges joining the triangles, then they are not both T-triangles.

We will call two triangles sharing one vertex, with no edges between them, a *ribbon* configuration.

First results

Property

If two triangles intersecting in one vertex have the 2-intersection property, and there are no edges joining the triangles, then they are not both T-triangles.

We will call two triangles sharing one vertex, with no edges between them, a *ribbon* configuration.

Figure: Two possible labels of a ribbon configuration. In each of them at most one triangle is a T-triangle.

First results

Property

Let T_1 and T_2 be two triangular cliques. Suppose there are just two edges joining a vertex of T_1 to a vertex of T_2, and that these edges have no common endpoints. Then the obtained configuration has the 2-intersection property. Furthermore, T_1 and T_2 cannot both be T-triangles. It is possible for T_1 and T_2 to be both K-triangles.

First results

Property

Let T_1 and T_2 be two triangular cliques. Suppose there are just two edges joining a vertex of T_1 to a vertex of T_2, and that these edges have no common endpoints. Then the obtained configuration has the 2-intersection property. Furthermore, T_1 and T_2 cannot both be T-triangles. It is possible for T_1 and T_2 to be both K-triangles.

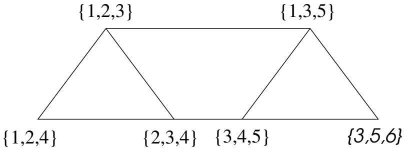
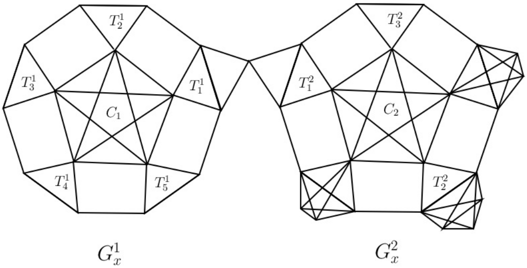


Figure: The configuration obtained by two triangles with joined by two edges. One possible labelling is shown.

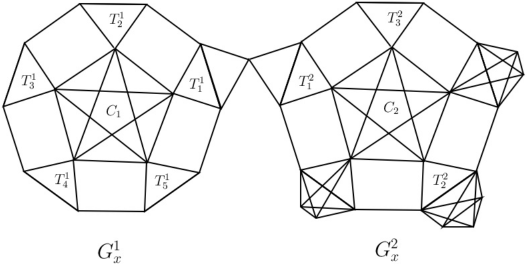
Variables representation

Define a *variable gadget*, denoted G_x, to represent a variable x in U. The gadget is a 2-intersection graph obtained by the union of different configurations as defined in the following figure.



Variables representation

Define a *variable gadget*, denoted G_x, to represent a variable x in U. The gadget is a 2-intersection graph obtained by the union of different configurations as defined in the following figure.



Property

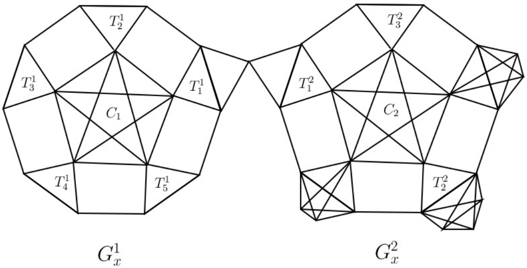
The variable gadget G_x is a 2-intersection graph.

Variables representation

Lemma

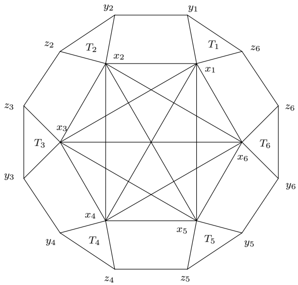
A labelling of the vertices of G_x allows only the following configurations for triangles T_2^1, T_5^1 and T_3^2:

$$i)\ \text{if } T_3^2 = T \text{ then } T_2^1 = T_5^1 = K;$$
$$ii)\ \text{if } T_5^1 = T \text{ (resp. } T_2^1 = T\text{) then } T_3^2 = K.$$



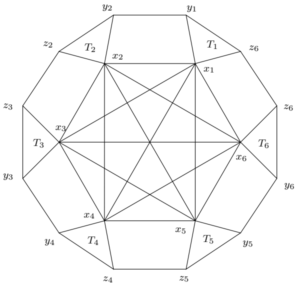
Clauses representation

The *clause gadget* G_c in Figure represents a single clause $c \in C$. It consists of a central clique K_6 whose vertices also belong to six different K_3 cliques, called *boundary triangles*.



Clauses representation

The *clause gadget* G_c in Figure represents a single clause $c \in C$. It consists of a central clique K_6 whose vertices also belong to six different K_3 cliques, called *boundary triangles*.



Lemma

A clause gadget G_c is a 2-intersection graph if and only if its boundary triangles do not contain either exactly three or exactly one T-triangles.

Note that by previous results G_c cannot have more than 3 $T-triangles$ as boundary triangles.

NP-Completeness

Let us consider the instance $C = \{c_1, \dots, c_n\}$ of MAX-3-SAT involving the variables in the set $U = \{x_1, \dots x_m\}$. Based on the gadgets already defined, we construct a graph G_C whose labels determine its 2-intersection property and express the desired valuations of C.

NP-Completeness

Let us consider the instance $C = \{c_1, \ldots, c_n\}$ of MAX-3-SAT involving the variables in the set $U = \{x_1, \ldots x_m\}$. Based on the gadgets already defined, we construct a graph G_C whose labels determine its 2-intersection property and express the desired valuations of C.

We can assume that each variable must appear at most three times: one in a form and two in the opposite form. For each variable $x_i \in U$, we define a variable gadget G_{x_i}, and associate the triangle T_3^2 with the single occurrences of a literal x_i. The at-most-two remaining occurrences of the opposite literal are associated with the triangles T_2^1 and T_5^1.

NP-Completeness

Let us consider the instance $C = \{c_1, \ldots, c_n\}$ of MAX-3-SAT involving the variables in the set $U = \{x_1, \ldots x_m\}$. Based on the gadgets already defined, we construct a graph G_C whose labels determine its 2-intersection property and express the desired valuations of C.

We can assume that each variable must appear at most three times: one in a form and two in the opposite form. For each variable $x_i \in U$, we define a variable gadget G_{x_i}, and associate the triangle T_3^2 with the single occurrences of a literal x_i. The at-most-two remaining occurrences of the opposite literal are associated with the triangles T_2^1 and T_5^1.

For each clause $c_j \in C$, we construct a clause gadget G_{c_j} and label its boundary triangles $T_1 \ldots T_6$. Finally, we connect variable gadgets and clause gadgets together as follows: for each clause c_j, with $1 \leq j \leq n$, we use a ribbon to the triangle T_{2i-1} of the clause gadget G_c, to the corresponding triangle associated with the i^{th} literal of c in the G_x gadget of its variable.

NP-Completeness

Let us consider the instance $C = \{c_1, \ldots, c_n\}$ of MAX-3-SAT involving the variables in the set $U = \{x_1, \ldots x_m\}$. Based on the gadgets already defined, we construct a graph G_C whose labels determine its 2-intersection property and express the desired valuations of C.

We can assume that each variable must appear at most three times: one in a form and two in the opposite form. For each variable $x_i \in U$, we define a variable gadget G_{x_i}, and associate the triangle T_3^2 with the single occurrences of a literal x_i. The at-most-two remaining occurrences of the opposite literal are associated with the triangles T_2^1 and T_5^1.

For each clause $c_j \in C$, we construct a clause gadget G_{c_j} and label its boundary triangles $T_1 \ldots T_6$. Finally, we connect variable gadgets and clause gadgets together as follows: for each clause c_j, with $1 \leq j \leq n$, we use a ribbon to the triangle T_{2i-1} of the clause gadget G_c, to the corresponding triangle associated with the i^{th} literal of c in the G_x gadget of its variable.

Theorem

Given an instance C of MAX-3-SAT, the graph G_C has the 2-intersection property if and only if the instance C has a solution.

Sketch of the proof

Let us assume that there exists a valuation for the MAX-3-SAT instance C. Given a variable $x \in U$, for each literal with value *true* associated with x, we assign the triangles associated with it to be T-triangles, and we assign the triangles associated with its negation to be K-triangles. For each clause $c_j \in C$, in its clause gadget G_{c_j}, we assign the triangles associated with the literals having valuation *true* to be K-triangles, but T-triangles for the literals having valuation *false*. The previous Lemmas assure that G_C has the 2−intersection property.

Sketch of the proof

Let us assume that there exists a valuation for the MAX-3-SAT instance C. Given a variable $x \in U$, for each literal with value *true* associated with x, we assign the triangles associated with it to be T-triangles, and we assign the triangles associated with its negation to be K-triangles. For each clause $c_j \in C$, in its clause gadget G_{c_j}, we assign the triangles associated with the literals having valuation *true* to be K-triangles, but T-triangles for the literals having valuation *false*. The previous Lemmas assure that G_C has the $2-$intersection property.

Suppose G_C has the 2-intersection property. For each clause gadget G_{c_j}, with $c_j \in C$, there exists at least one triangle among T_1, T_3 and T_5, say T' of type K. Previous properties assures that T' having type K leads to a T-triangle in the variable gadget G_x to which a literal, say l, is associated. We assign such a literal value *true*. The opposite literal $\bar{l}$ is then associated with *false*. The valuation defined is a solution of the MAX-3-SAT instance C.

Example

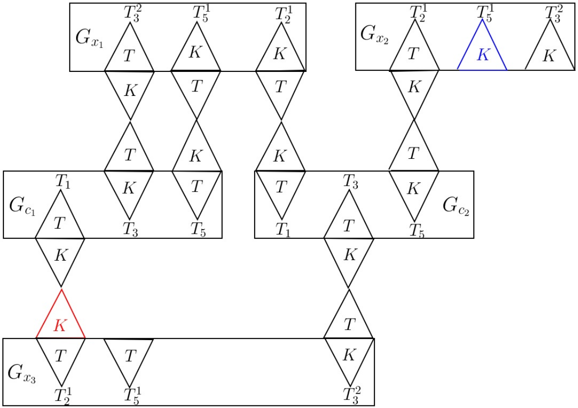


Figure: Example of the construction of the gadget for $C = \{c_1, c_2\}$, $c_1 = (x_3, x_1, \overline{x}_1)$, $c_2 = (\overline{x}_1, \overline{x_3}, x_2)$. One valuations obtained by the labelling of the corresponding G_C graph is $x_1 = true$, $x_2 = true$ and $x_3 = false$.

Thanks for your attention!