A Polynomial-Time Algorithm for Finding a Minimal Conflicting Set Containing a Given Row

Guillaume Blin Romeo Rizzi Stéphane Vialette

LIGM Université Paris-Est Marne-la-Vallée, France

DIMI Università degli Studi di Udine, Italy.

June 2011

Consecutive ones property

Definition

A $(0, 1)$-matrix has the consecutive ones property (C1P) for rows if there is a permutation of its columns that leaves the 1's consecutive in every row.

Example

$$M = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 \end{bmatrix} \qquad MP = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 \end{bmatrix}$$

Minimal Conflicting Sets

Definition

A Minimal Conflicting Set of Rows (MCSR) is a set of rows R of a matrix that does not have the C1P but such that any proper subset of R has the C1P.

The Conflicting Index (CI) of a given row is the number of MCSR involving this last.

Example - MCSR

$$M = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix}$$

Minimal Conflicting Sets

Definition

A Minimal Conflicting Set of Rows (MCSR) is a set of rows R of a matrix that does not have the C1P but such that any proper subset of R has the C1P.

The Conflicting Index (CI) of a given row is the number of MCSR involving this last.

Example - MCSR

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \qquad RP = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Minimal Conflicting Sets

Definition

A Minimal Conflicting Set of Rows (MCSR) is a set of rows R of a matrix that does not have the C1P but such that any proper subset of R has the C1P.

The Conflicting Index (CI) of a given row is the number of MCSR involving this last.

Example - MCSR

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \qquad RP = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

Minimal Conflicting Sets

Definition

A Minimal Conflicting Set of Rows (MCSR) is a set of rows R of a matrix that does not have the C1P but such that any proper subset of R has the C1P.

The Conflicting Index (CI) of a given row is the number of MCSR involving this last.

Example - MCSR

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \qquad RP = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \end{bmatrix}$$

Minimal Conflicting Sets

Definition

A Minimal Conflicting Set of Rows (MCSR) is a set of rows R of a matrix that does not have the C1P but such that any proper subset of R has the C1P.

The Conflicting Index (CI) of a given row is the number of MCSR involving this last.

Example - MCSR

$$R = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{bmatrix} \qquad RP = \begin{bmatrix} 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}$$

Background

Theorem (Chauve et al., 09)

Let M be a $m \times n$ $(0,1)$-matrix with at most Δ 1-entries per row. Deciding if a given row of M has a positive CI can be decided in $O(\Delta^2 m^{max(4,\Delta+1)}(n+m+e))$ time.

Main result

What about unbounded Δ ?

We prove it is still polynomial by combining characterization of matrices having the C1P with graph pruning techniques.

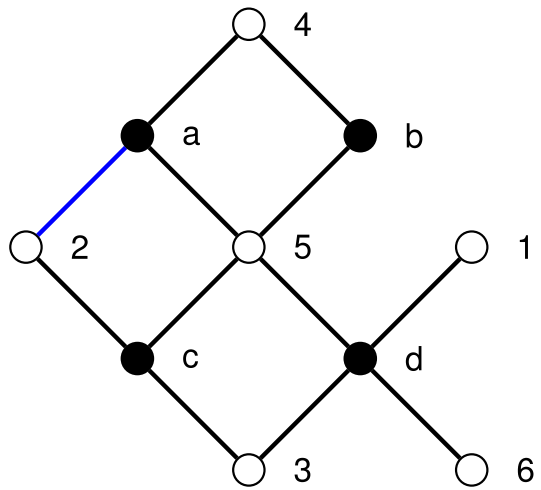
From $(0,1)$-matrices to colored bipartite graphs

Definition

Let M be a $(0,1)$-matrix. Its corresponding vertex-colored bipartite graph $G(M) = (V_M, E_M)$ is defined by associating a *black vertex* to each row of M, a *white vertex* to each column of M, and by adding an edge between the vertices that correspond to the i^{th} row and the j^{th} column of M if and only of $M[i,j] = 1$.

Example

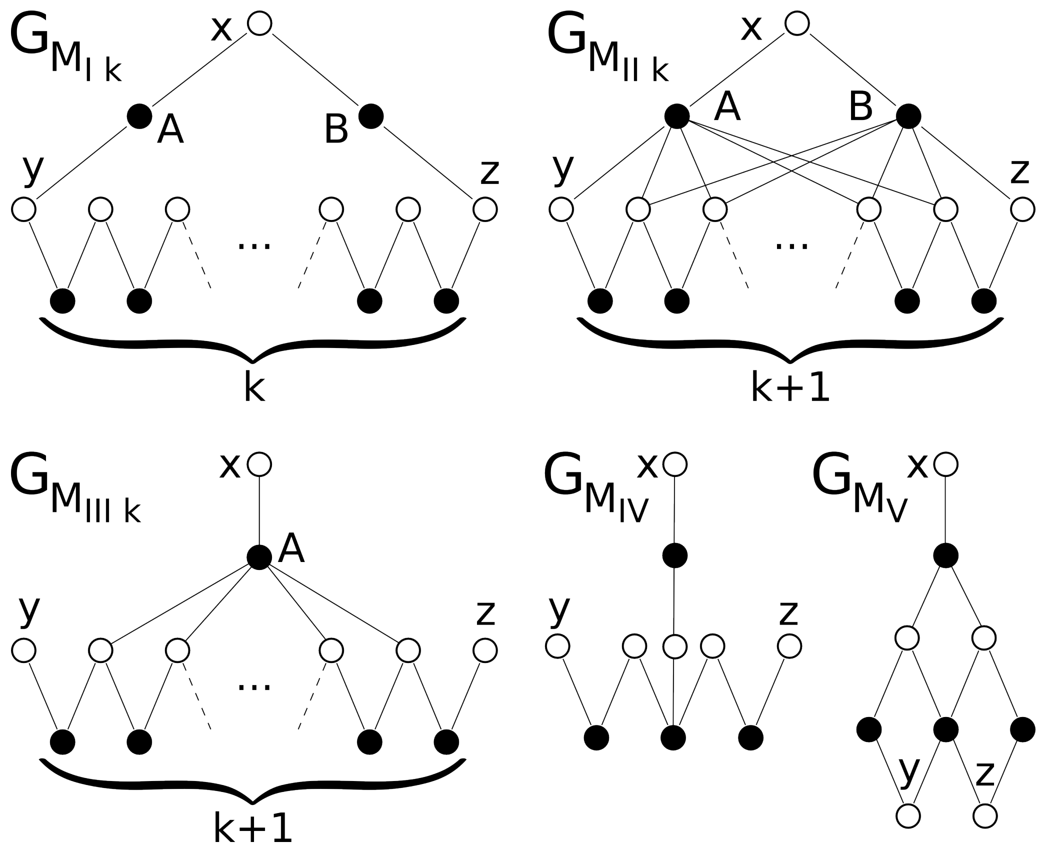
$$M = \begin{bmatrix} & 1 & 2 & 3 & 4 & 5 & 6 \\ a & 0 & 1 & 0 & 1 & 1 & 0 \\ b & 0 & 0 & 0 & 1 & 1 & 0 \\ c & 0 & 1 & 1 & 0 & 1 & 0 \\ d & 1 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

$G(M) =$ 

C1P and forbidden structures

Theorem (Tucker, 72)

A $(0,1)$-matrix has the C1P if and only if it contains none of the matrices M_{I_k}, M_{II_k}, M_{III_k} $(k \geq 1)$, M_{IV}, and M_V depicted below:



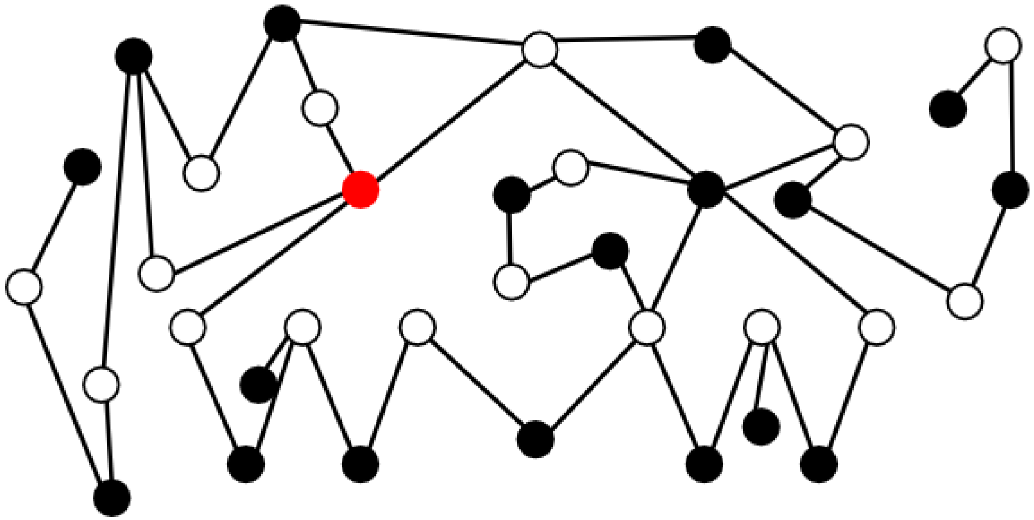
... that we will try to detect

Process of finding MCSR of r

Definition

Finding a set of black nodes R not having C1P and any $R' \subset R$ has C1P

$\Rightarrow \exists$ Tucker configuration (e.g. holes of size ≥ 6) using the set of rows R and
$\nexists$ a Tucker configuration using a proper subset of R

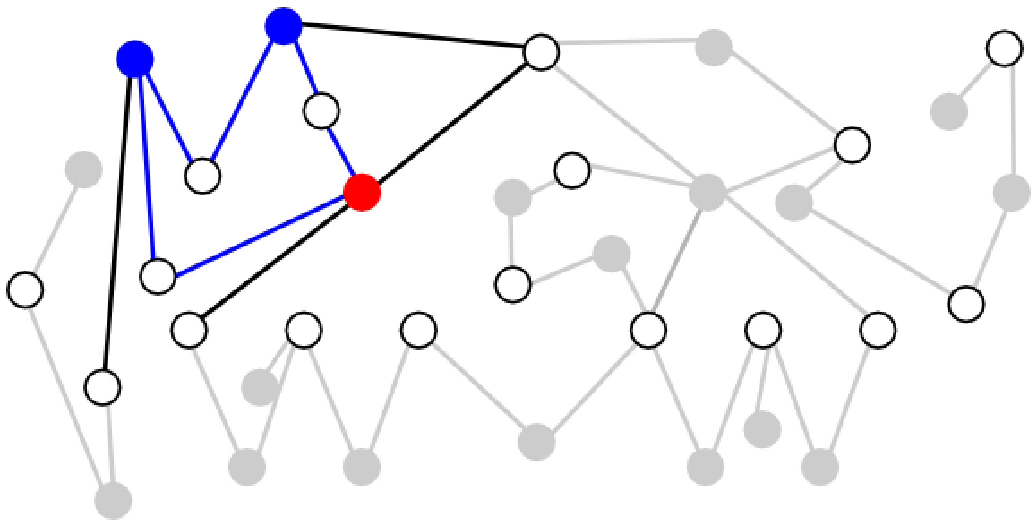


Process of finding MCSR of r

Definition

Finding a set of black nodes R not having C1P and any $R' \subset R$ has C1P

$\Rightarrow \exists$ Tucker configuration (e.g. holes of size ≥ 6) using the set of rows R and
$\nexists$ a Tucker configuration using a proper subset of R



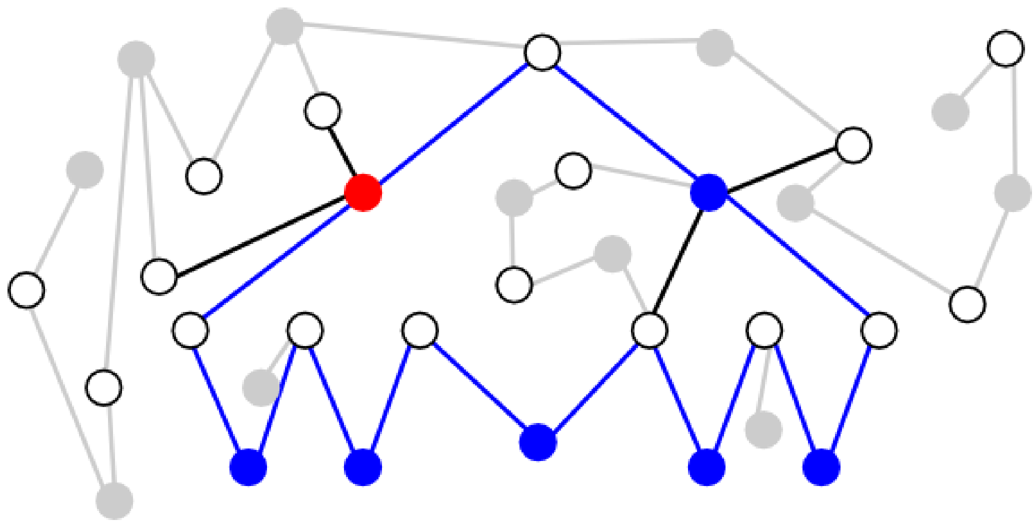
remind that we are pruning the rows but not the columns

Process of finding MCSR of r

Definition

Finding a set of black nodes R not having C1P and any $R' \subset R$ has C1P

$\Rightarrow \exists$ Tucker configuration (e.g. holes of size ≥ 6) using the set of rows R and
$\nexists$ a Tucker configuration using a proper subset of R

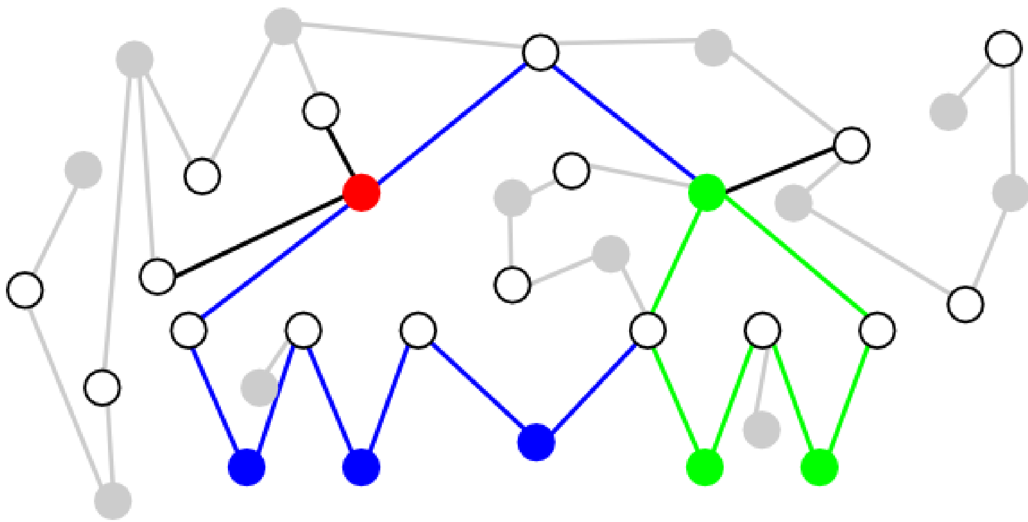


Process of finding MCSR of r

Definition

Finding a set of black nodes R not having C1P and any $R' \subset R$ has C1P

$\Rightarrow \exists$ Tucker configuration (e.g. holes of size ≥ 6) using the set of rows R and
$\nexists$ a Tucker configuration using a proper subset of R

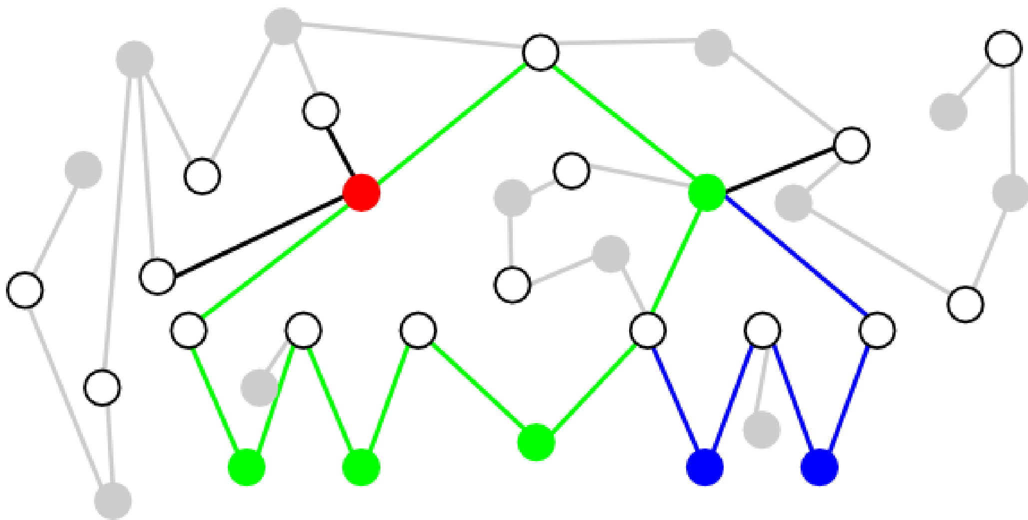


Process of finding MCSR of r

Definition

Finding a set of black nodes R not having C1P and any $R' \subset R$ has C1P

$\Rightarrow \exists$ Tucker configuration (e.g. holes of size ≥ 6) using the set of rows R and
$\nexists$ a Tucker configuration using a proper subset of R

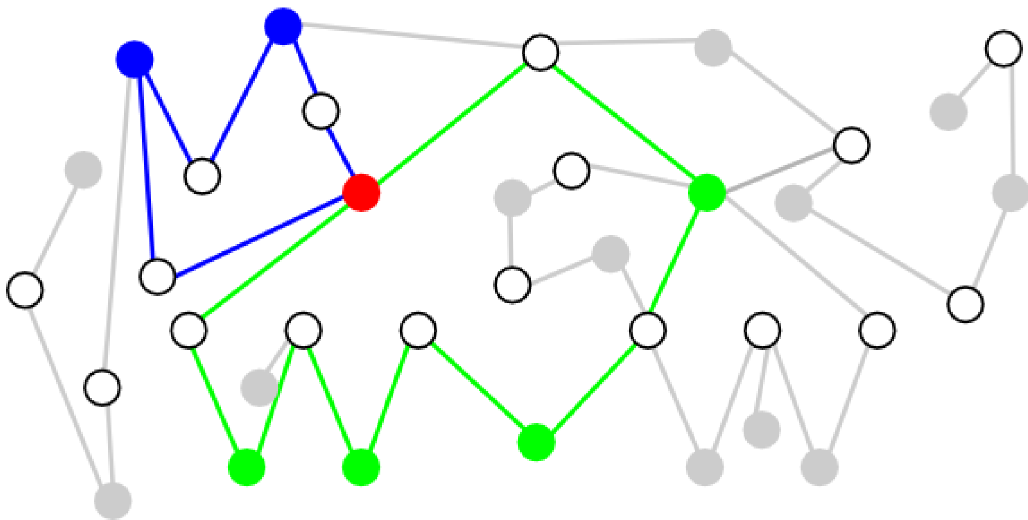


Process of finding MCSR of r

Definition

Finding a set of black nodes R not having C1P and any $R' \subset R$ has C1P

$\Rightarrow \exists$ Tucker configuration (e.g. holes of size ≥ 6) using the set of rows R and
$\nexists$ a Tucker configuration using a proper subset of R



both are minimal and finding at least one is enough to prove that the $CI(r) > 0$

General idea

Theorem

Let M be $m \times n$ $(0,1)$-matrix. Deciding if a given row of M has a positive CI can be decided in $O(m^6 n^5 (m+n)^2 \log(m+n))$ time.

Well it is polynomial ....

To be compared to the $O(\Delta^2 m^{max(4,\Delta+1)}(n+m+e))$ time for bounded case

Proof

We provide a sequence of polynomial-time algorithms for finding a minimal Tucker configuration of a given type in $\{M_{I_k}, M_{III_k}, M_{II_k}, M_{IV}, M_V\}$ (in this particular order) responsible for an MCSR involving a given row (if it exists).

Graph pruning and exhaustive search

Our algorithm is by combining shortest paths and two graph pruning techniques (`clean` and `anticlean`) together with some exhaustive search procedures (`guess`), *i.e.*,

- guessing (`guess`):
  exhaustive brute-force search.
- cleaning (`clean`):
  clean the neighbordhood of a vertex.
- anticleaning (`anticlean`):
  clean the non-neighbordhood of a vertex.

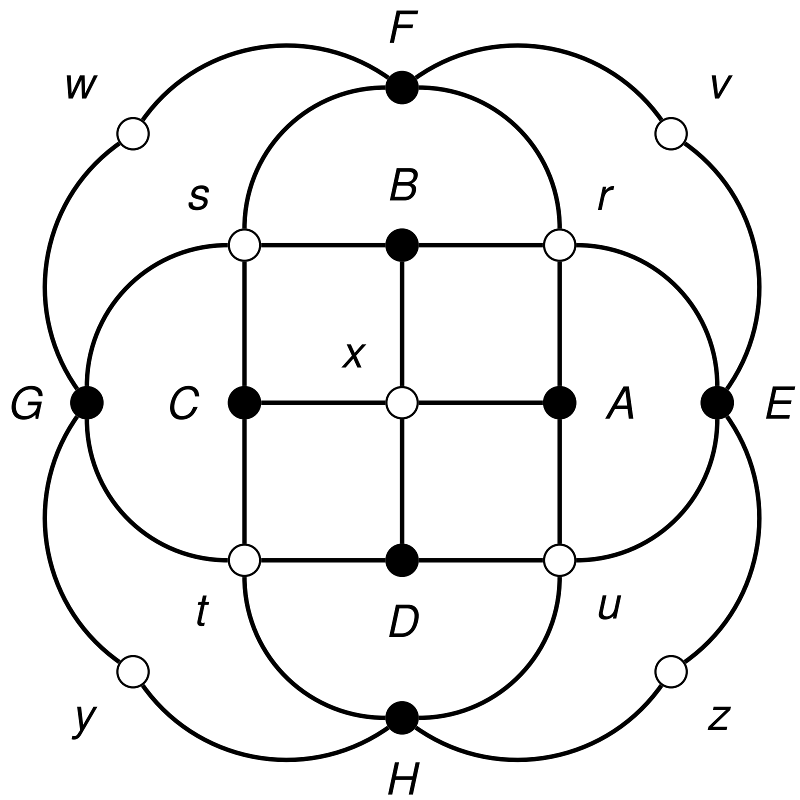
Note that guessed nodes are not affected by (anti)cleaning operations

Cleaning vertices

Definition (`clean`)

For any node x of $G(M)$, $\texttt{clean}(x)$ results in the graph where any neighbor of x has been deleted,

Example

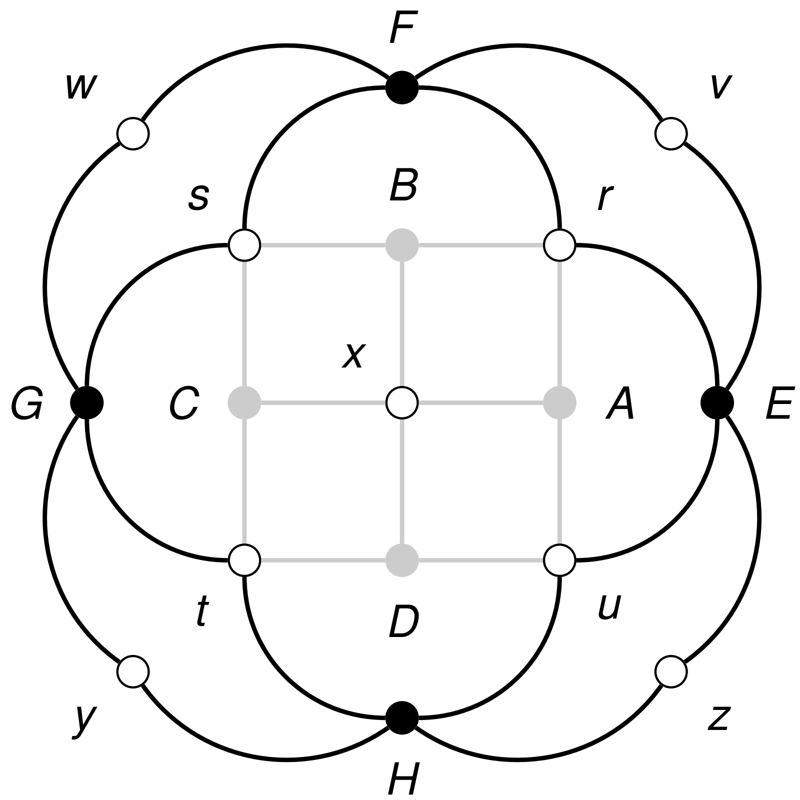


Cleaning vertices

Definition (`clean`)

For any node x of $G(M)$, $\mathtt{clean}(x)$ results in the graph where any neighbor of x has been deleted,

Example

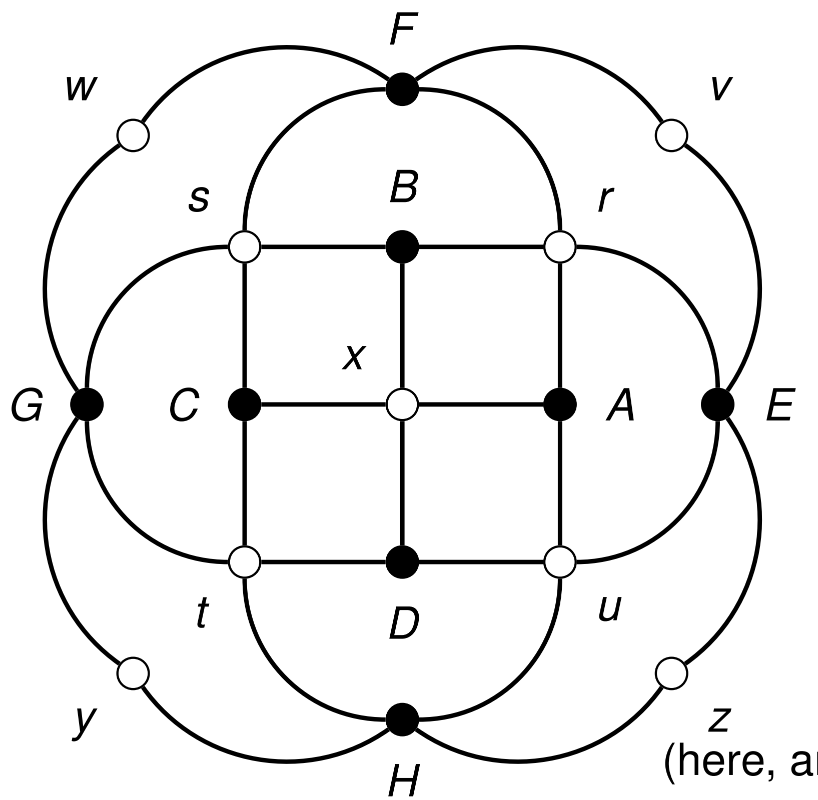


Anticleaning vertices

Definition (anticlean)

For any node x of $G(M)$, $\texttt{anticlean}(x)$ results in the graph where any vertex with a different color and not in the neighborhood of x has been deleted.

Example



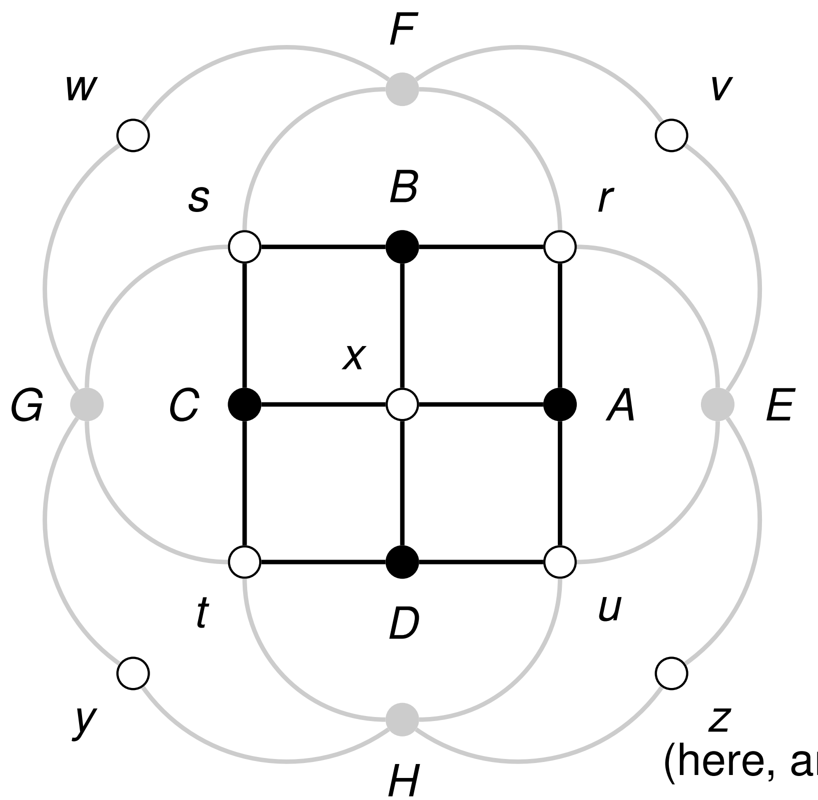
(here, any black node $\notin N(x)$)

Anticleaning vertices

Definition (`anticlean`)

For any node x of $G(M)$, `anticlean`(x) results in the graph where any vertex with a different color and not in the neighborhood of x has been deleted.

Example

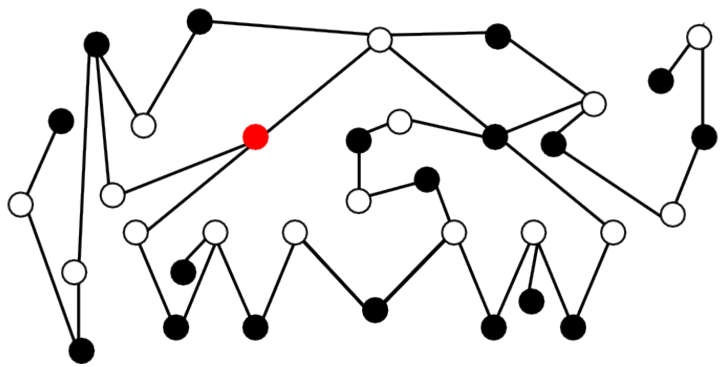


(here, any black node $\notin N(x)$)

Identifying M_{I_k} MCSR of r

Theorem

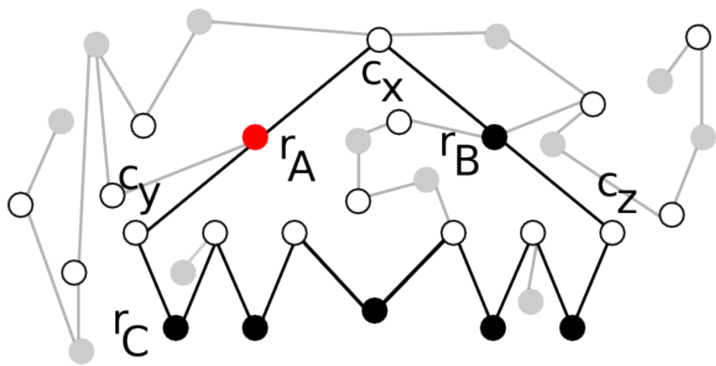
Let M be $m \times n$ $(0,1)$-matrix. Finding (if it exists) a minimal M_{I_k} structures responsible for an MCSR of r is a $O(m^4 n^4)$ time procedure.



Identifying M_{I_k} MCSR of r

Theorem

Let M be $m \times n$ $(0,1)$-matrix. Finding (if it exists) a minimal M_{I_k} structures responsible for an MCSR of r is a $O(m^4 n^4)$ time procedure.



- Brute-force seek for $G(M_{I_1})$ or $G(M_{I_2})$ s.t. no $G(M_{III_1})$ involving r exists (only smaller Tucker configuration that can occur)
- If none exists, $\texttt{guess}(r_A, r_B, r_C, c_x, c_y)$ s.t. $r = r_A$ and $(r_C, c_y, r_A, c_x, r_B)$ is a path in $G(M)$
- Otherwise call Check-$M_{I_k}(c_x, c_y, r_A, r_B, r_C)$

Identifying M_{I_k} MCSR of r

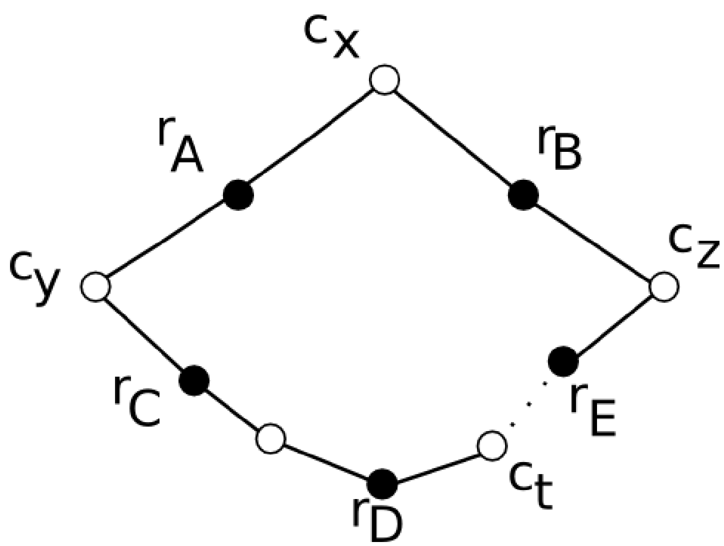
Check-$M_{I_k}(c_x, c_y, r_A, r_B, r_C)$

if $N(r_A) \cap N(r_B) \cap N(r_C) \neq \emptyset$ **then**
    return "NO"
end if
`clean`(c) for all $c \in N(r_A) \setminus N(r_B)$
`clean`(c) for all $c \in N(r_A) \setminus N(r_C)$
`clean`(r_A, c_x, c_y)
delete vertex r_A
if there exists a $r_B r_C$-path in the pruned graph **then**
    let P be a shortest $r_B r_C$-path in the pruned graph
    return return $\{r_A\} \cup \{r_i : r_i \in V(P) \cap \mathcal{R}\}$
else
    return "NO"
end if

Identifying M_{I_k} MCSR of r : safe pruning

1: **if** $N(r_A) \cap N(r_B) \cap N(r_C) \neq \emptyset$ **then**
2: **return** "NO"
3: **end if**

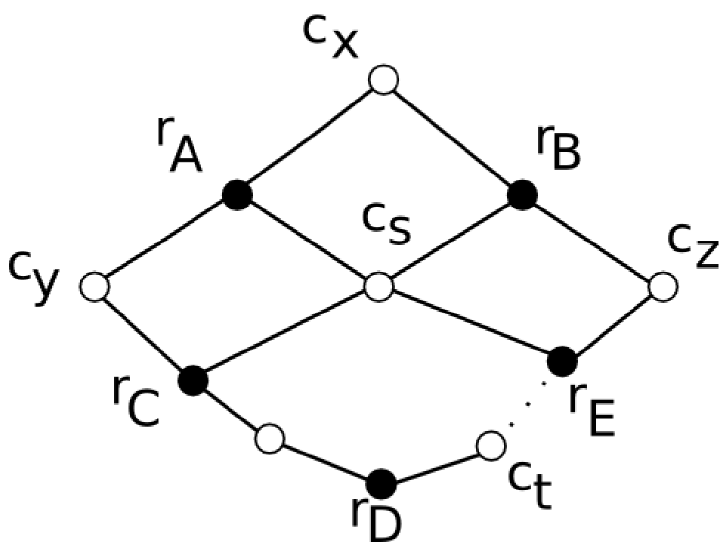
Remark that the minimal M_{I_k} configuration is



Identifying M_{I_k} MCSR of r : safe pruning

if $N(r_A) \cap N(r_B) \cap N(r_C) \neq \emptyset$ **then**
    return "NO"
end if

Remark that the minimal M_{I_k} configuration is

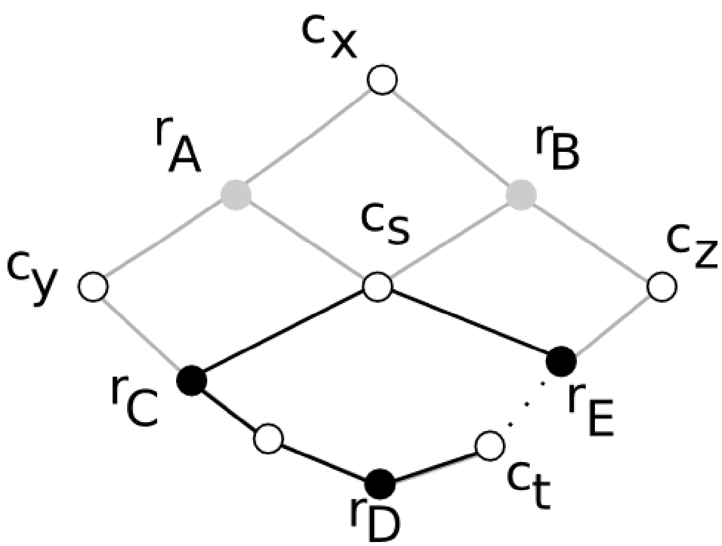


Suppose $N(r_A) \cap N(r_B) \cap N(r_C) = c_s$ and $c_s \notin N(r_D)$

Identifying M_{I_k} MCSR of r : safe pruning

1: **if** $N(r_A) \cap N(r_B) \cap N(r_C) \neq \emptyset$ **then**
2: **return** "NO"
3: **end if**

Remark that the minimal M_{I_k} configuration is

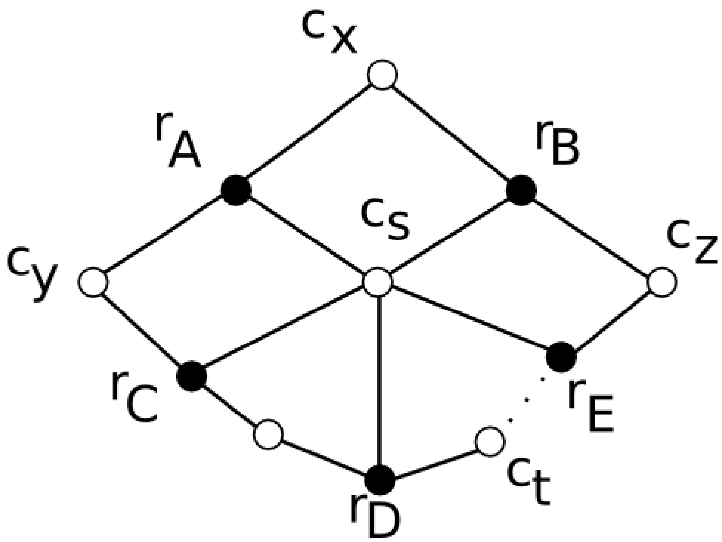


Then there exists a smaller M_{I_k} configuration (impossible if we proceed k increasingly)

Identifying M_{I_k} MCSR of r : safe pruning

if $N(r_A) \cap N(r_B) \cap N(r_C) \neq \emptyset$ **then**
    return "NO"
end if

Remark that the minimal M_{I_k} configuration is

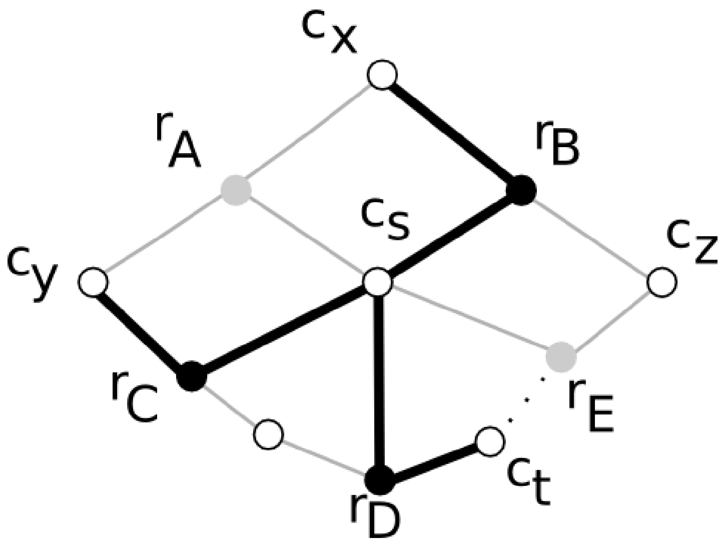


Thus, $N(r_A) \cap N(r_B) \cap N(r_C) = c_s$ is a common neighbor of any black node

Identifying M_{I_k} MCSR of r : safe pruning

if $N(r_A) \cap N(r_B) \cap N(r_C) \neq \emptyset$ **then**
    return ”NO”
end if

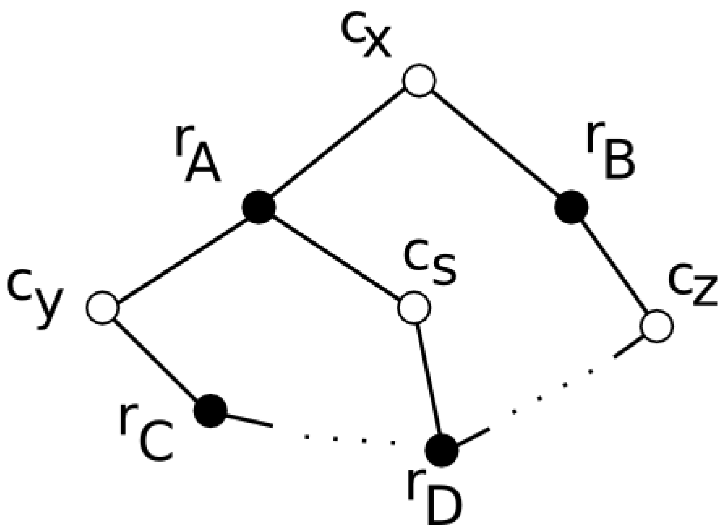
Remark that the minimal M_{I_k} configuration is



Then there exists a smaller M_{III_1} configuration

Identifying M_{I_k} MCSR of r : safe pruning

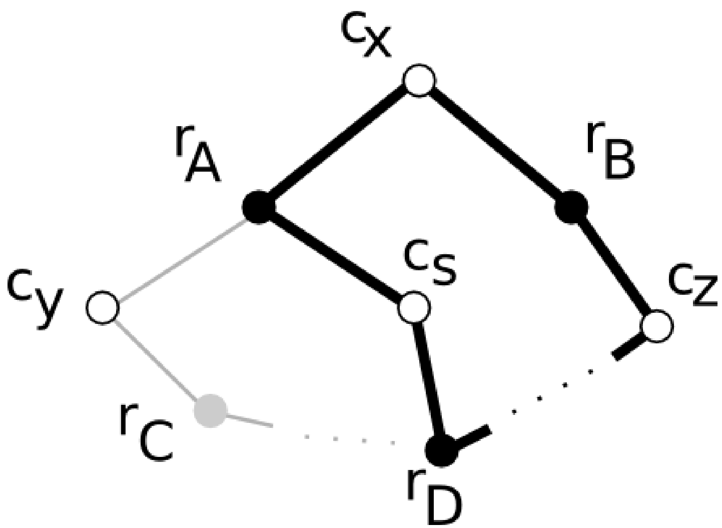
1: $\texttt{clean}(c)$ for all $c \in N(r_A) \setminus N(r_B)$



Suppose that $\texttt{clean}(c_s)$ is not a safe operation (we will ”break” a solution). Then it follows that $c_s \in N(r_D)$ for some black vertex of the solution

Identifying M_{I_k} MCSR of r : safe pruning

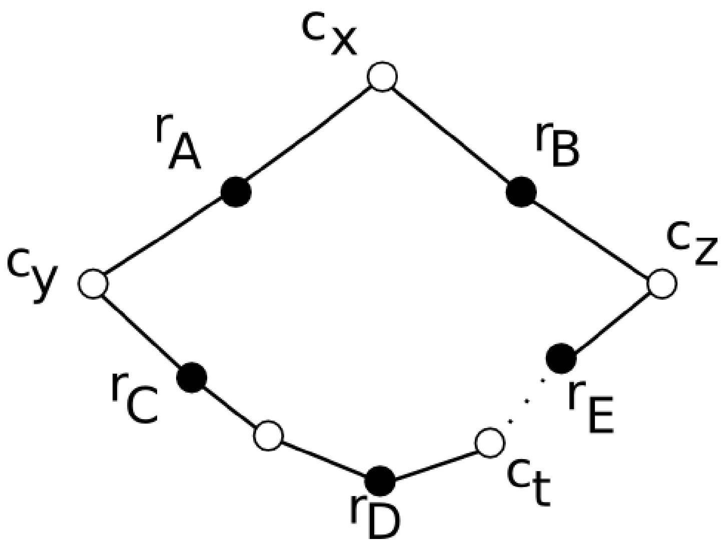
1: $\texttt{clean}(c)$ for all $c \in N(r_A) \setminus N(r_B)$



Then there exists a smaller $M_{I_{k'}}$ configuration

Identifying M_{I_k} MCSR of r : safe pruning

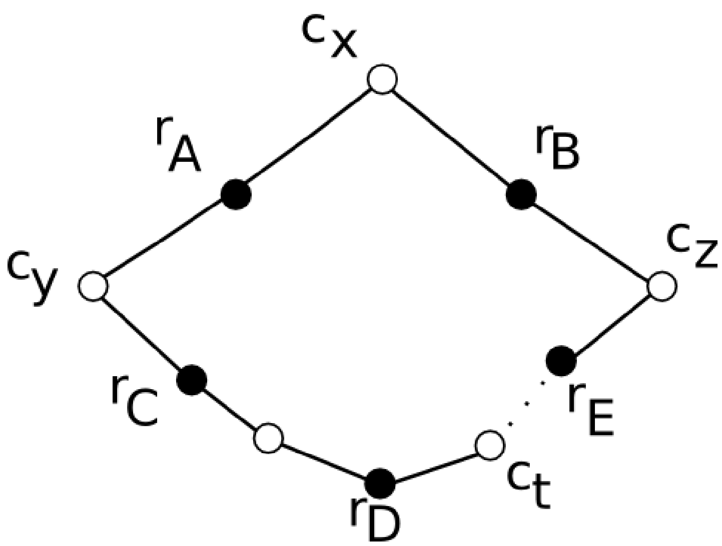
1: `clean`(c) for all $c \in N(r_A) \setminus N(r_C)$
2: `clean`(r_A, c_x, c_y)



Similar proof for $c \in N(r_A) \setminus N(r_C)$.
Moreover, since T is a chordless cycle, no black vertices of the solution other than the guessed ones can see c_x or c_y

Identifying M_{I_k} MCSR of r : safe pruning

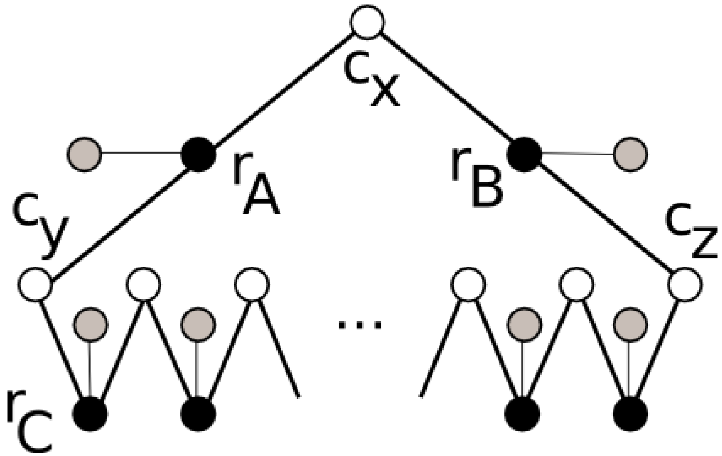
```
delete vertex r_A
if there exists a r_B r_C-path in the pruned graph then
    let P be a shortest r_B r_C-path in the pruned graph
    return  return {r_A} ∪ {r_i : r_i ∈ V(P) ∩ R}
else
    return  "NO"
end if
```



Finding the shortest path ensures the minimality of our configuration

Identifying M_{I_k} MCSR of r : safe pruning

delete vertex r_A
if there exists a $r_B r_C$-path in the pruned graph **then**
    let P be a shortest $r_B r_C$-path in the pruned graph
    return return $\{r_A\} \cup \{r_i : r_i \in V(P) \cap \mathcal{R}\}$
else
    return "NO"
end if



One can prove that considering back all the white vertices leads to a M_{I_k} MCSR

Identifying other MCSR of r

In a similar way, we designed 4 other algorithms to detect MCSR of a given type leading to

Tucker configuration	Running time
M_{I_k}	$O(m^3 n^4)$
M_{II_k}	$O(m^6 n^5 (m+n)^2 \log(m+n))$
M_{III_k}	$O(m^5 n^5 (m+n)^2 \log(m+n))$
M_{IV}	$O(m^2 n^6)$
M_V	$O(m^3 n^5)$
Total	$O(m^6 n^5 (m+n)^2 \log(m+n))$

Matrices with unbounded Δ

Theorem

Let M be $m \times n$ $(0, 1)$-matrix. Deciding if a given row of M has a positive CI can be decided in $O(m^6 n^5 (m+n)^2 \log(m+n))$ time.

Going further...

Our graph pruning techniques can be used for solving related combinatorial problems.

Working also for Minimal Conflicting Set of Columns

Implying a polynomial-time algorithm for the *Circular Ones Property* (Circ1P) studied by Dom et al. 2009. (considering the matrix as being wrapped around a vertical cylinder).

A Polynomial-Time Algorithm for Finding a Minimal Conflicting Set Containing a Given Row

Guillaume Blin Romeo Rizzi Stéphane Vialette

LIGM Université Paris-Est Marne-la-Vallée, France

DIMI Università degli Studi di Udine, Italy.

June 2011