List edge multicoloring in bounded cyclicity graphs

Dániel Marx
Budapest University of Technology and Economics
dmarx@cs.bme.hu

List edge multicoloring

Generalization of list edge coloring: multiple colors have to be assigned to each edge.

- **Given:** a graph $G(V, E)$, a list $L(e) \subseteq C$ for each edge e, and a demand function x: $E \longrightarrow \mathbb{N}$
- **Find:** an assignment $\Psi(e) \subseteq L(e)$ of $x(e)$ colors to every edge e, such that adjacent edges receive disjoint sets

List edge coloring is the special case $x(e) = 1$ for every edge e.

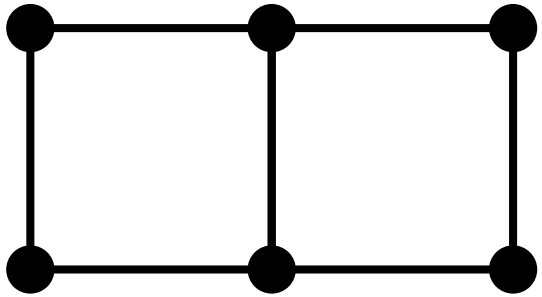
List edge multicoloring

Generalization of list edge coloring: multiple colors have to be assigned to each edge.

- **Given:** a graph $G(V, E)$, a list $L(e) \subseteq C$ for each edge e, and a demand function x: $E \longrightarrow \mathbb{N}$
- **Find:** an assignment $\Psi(e) \subseteq L(e)$ of $x(e)$ colors to every edge e, such that adjacent edges receive disjoint sets

List edge coloring is the special case $x(e) = 1$ for every edge e.

Example:



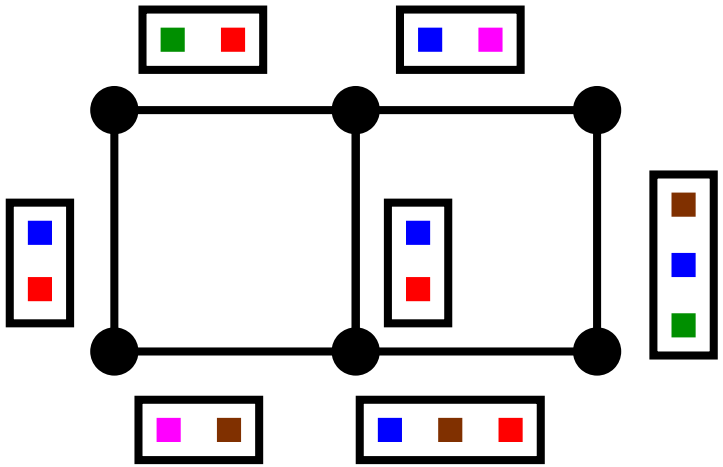
List edge multicoloring

Generalization of list edge coloring: multiple colors have to be assigned to each edge.

- **Given:** a graph $G(V, E)$, a list $L(e) \subseteq C$ for each edge e, and a demand function x: $E \longrightarrow \mathbb{N}$
- **Find:** an assignment $\Psi(e) \subseteq L(e)$ of $x(e)$ colors to every edge e, such that adjacent edges receive disjoint sets

List edge coloring is the special case $x(e) = 1$ for every edge e.

Example:



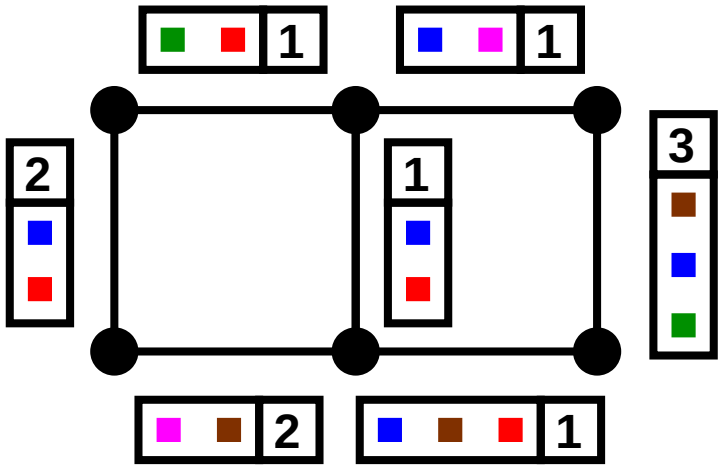
List edge multicoloring

Generalization of list edge coloring: multiple colors have to be assigned to each edge.

- **Given:** a graph $G(V, E)$, a list $L(e) \subseteq C$ for each edge e, and a demand function x: $E \longrightarrow \mathbb{N}$
- **Find:** an assignment $\Psi(e) \subseteq L(e)$ of $x(e)$ colors to every edge e, such that adjacent edges receive disjoint sets

List edge coloring is the special case $x(e) = 1$ for every edge e.

Example:



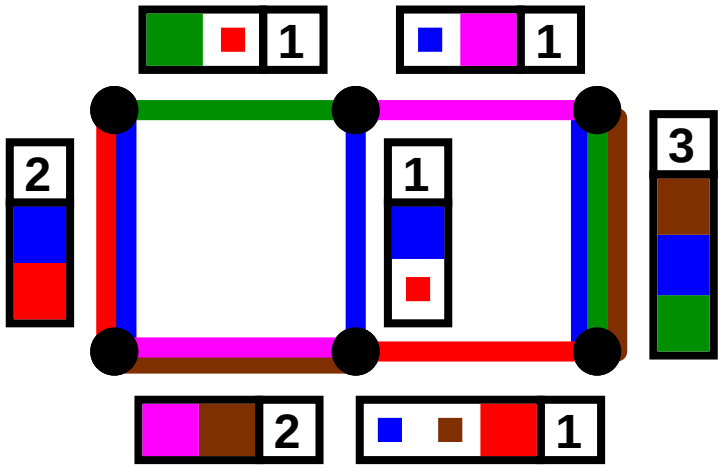
List edge multicoloring

Generalization of list edge coloring: multiple colors have to be assigned to each edge.

- **Given:** a graph $G(V, E)$, a list $L(e) \subseteq C$ for each edge e, and a demand function x: $E \longrightarrow \mathbb{N}$
- **Find:** an assignment $\Psi(e) \subseteq L(e)$ of $x(e)$ colors to every edge e, such that adjacent edges receive disjoint sets

List edge coloring is the special case $x(e) = 1$ for every edge e.

Example:



Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

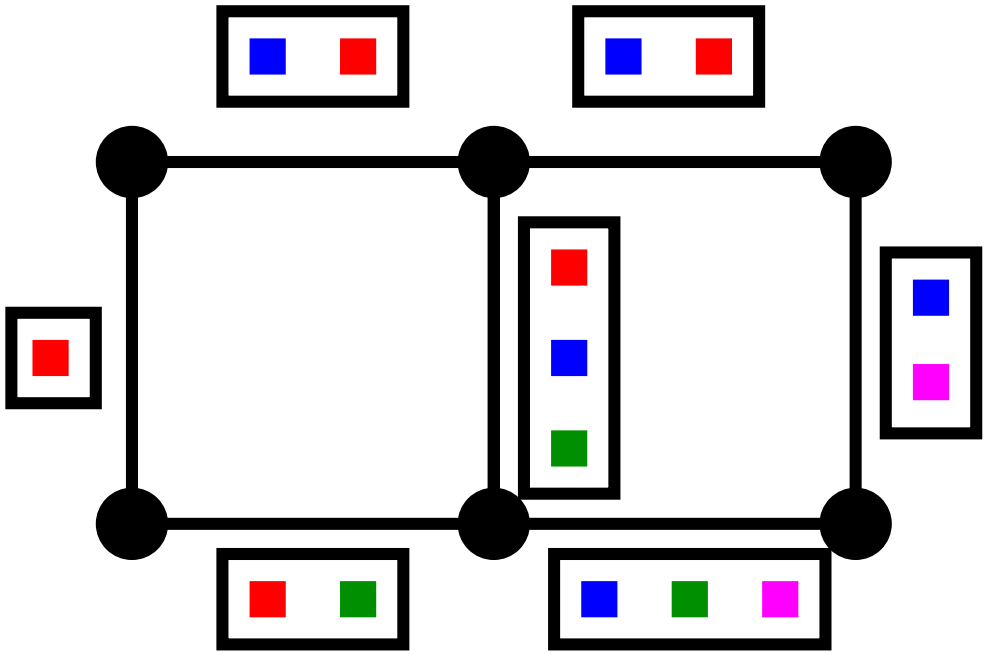
Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

Example: (every demand is 1)



$\sum_{c \in C} \nu_c(G) =$

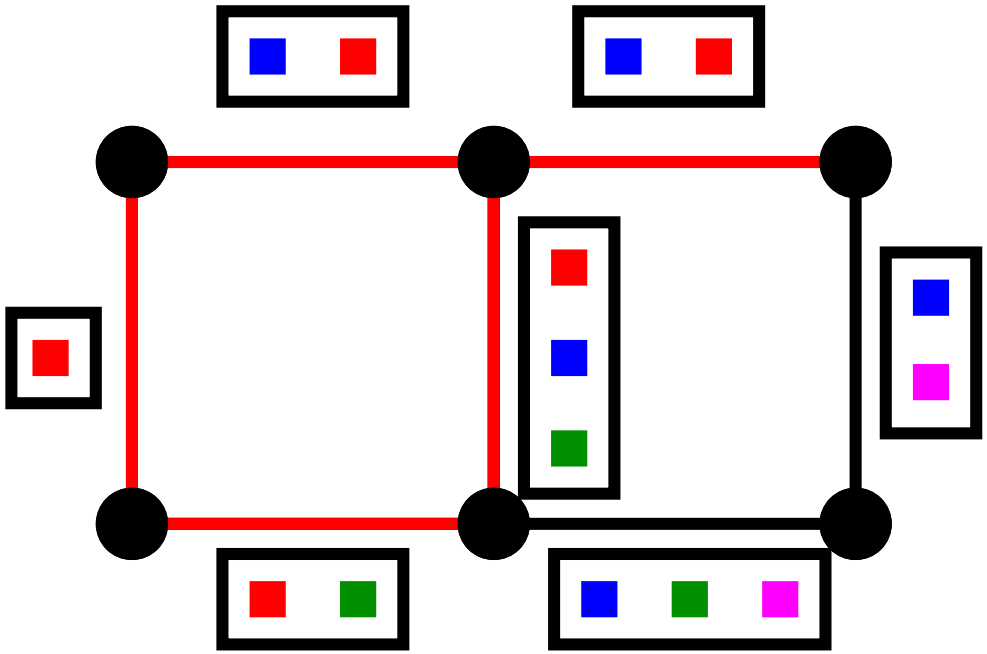
Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

Example: (every demand is 1)



$\sum_{c \in C} \nu_c(G) =$

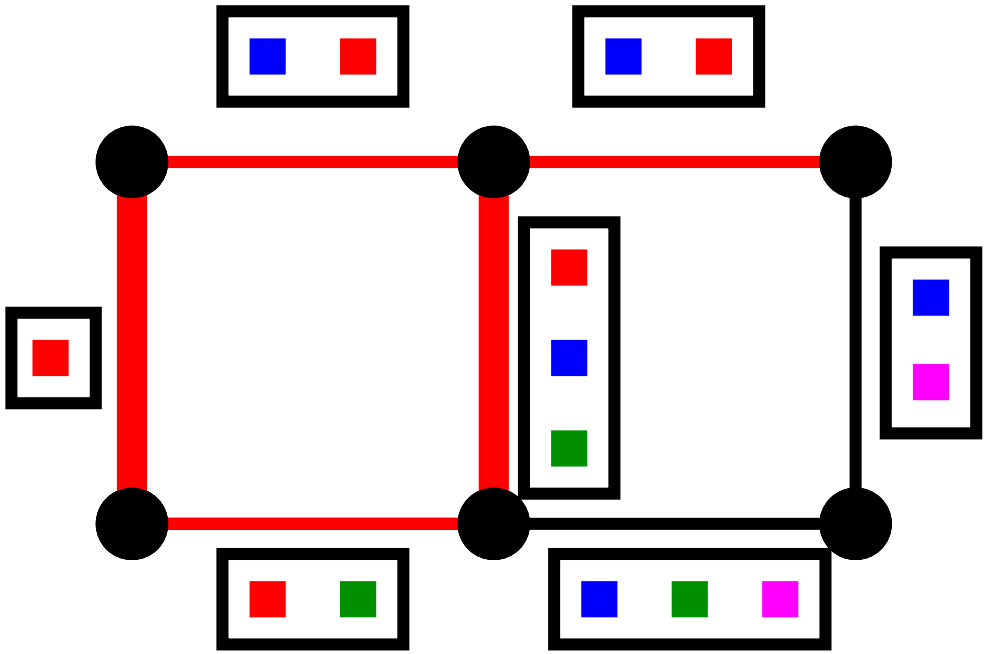
Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

Example: (every demand is 1)



$\sum_{c \in C} \nu_c(G) =$

2

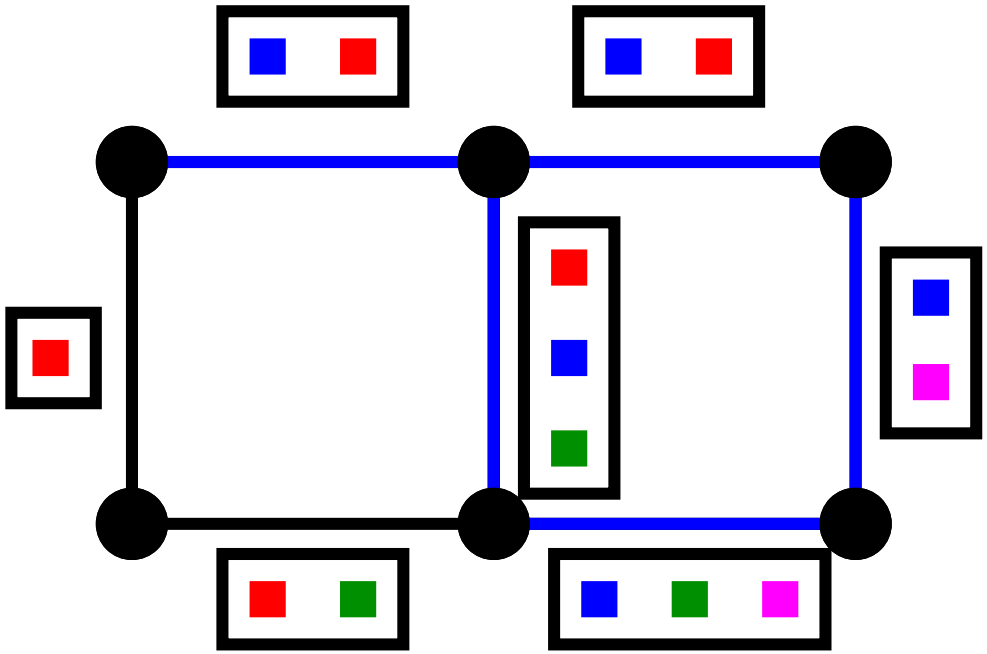
Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

Example: (every demand is 1)



$\sum_{c \in C} \nu_c(G) =$

2

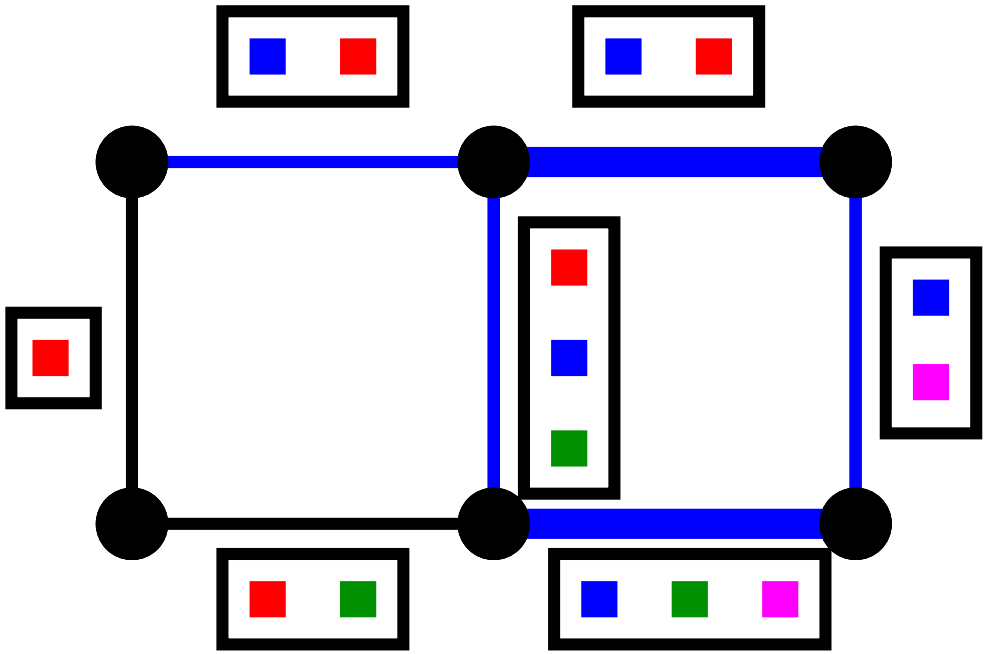
Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

Example: (every demand is 1)



$\sum_{c \in C} \nu_c(G) =$

$2+2$

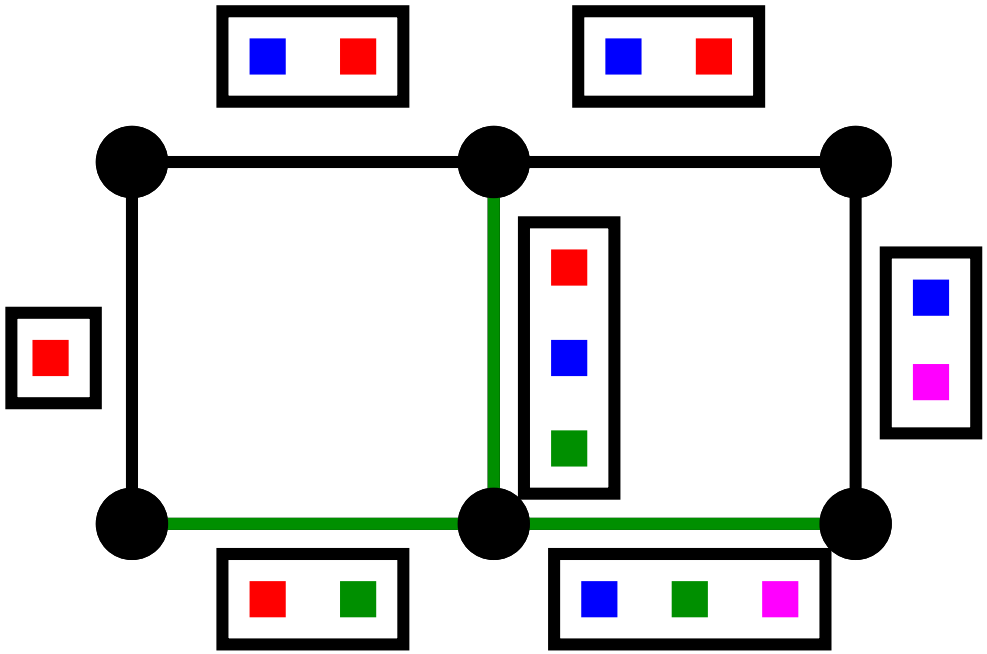
Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

Example: (every demand is 1)



$\sum_{c \in C} \nu_c(G) =$

$2+2$

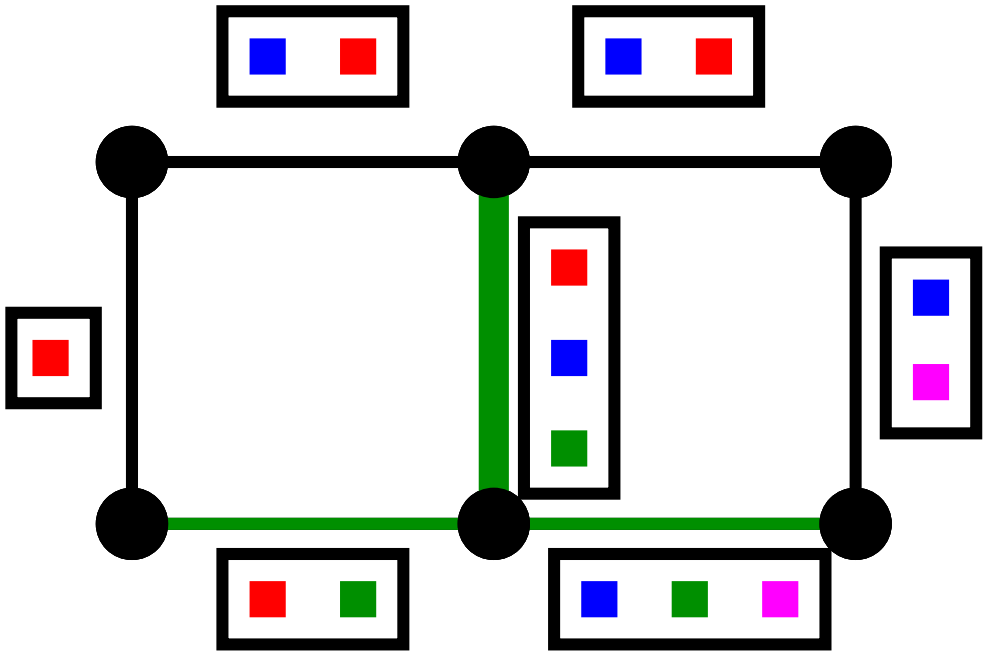
Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

Example: (every demand is 1)



$\sum_{c \in C} \nu_c(G) =$

$2+2+1$

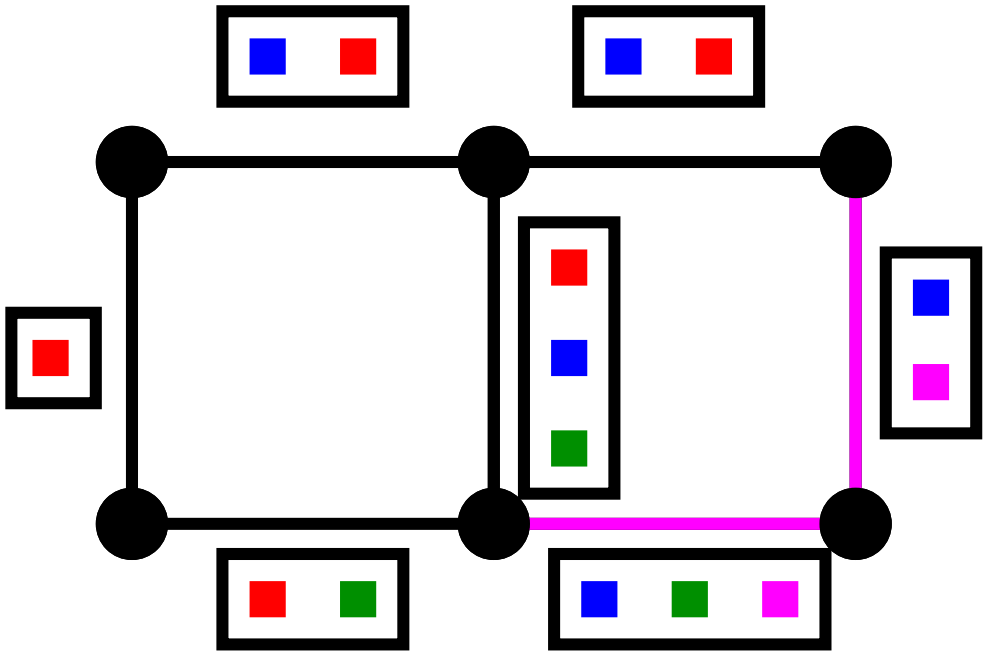
Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

Example: (every demand is 1)



$\sum_{c \in C} \nu_c(G) =$

$2+2+1$

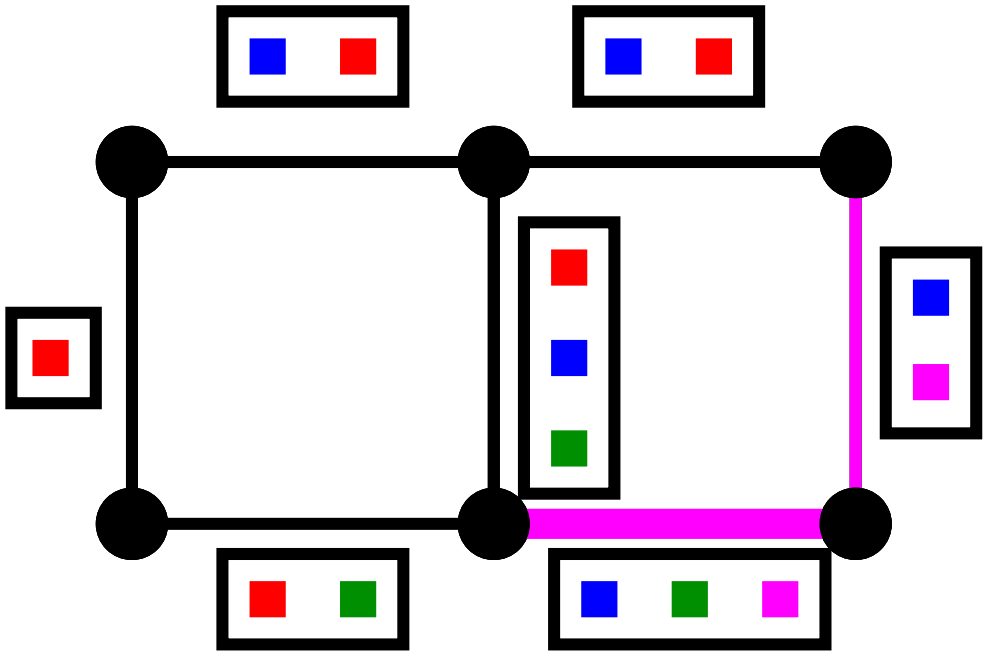
Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

Example: (every demand is 1)



$\sum_{c \in C} \nu_c(G) =$

$2+2+1+1$

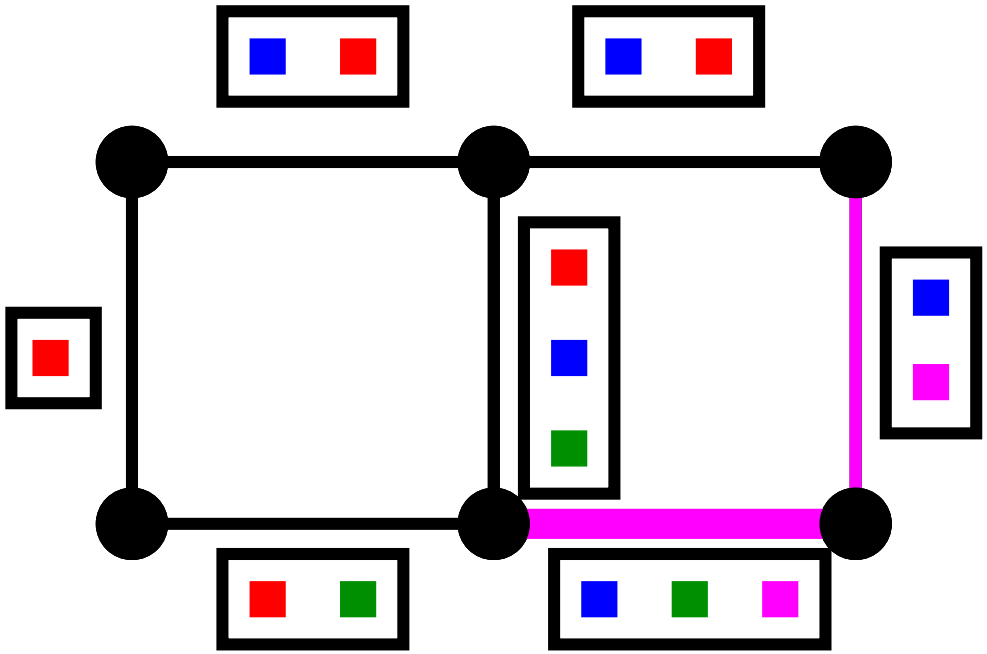
Hall's condition

$\nu_c(G)$: maximum number of independent edges that can receive color c

Necessary condition for the existence of the coloring:

$$\sum_{c \in C} \nu_c(G) \geq \sum_{e \in E(G)} x(e)$$

Example: (every demand is 1)



$\sum_{c \in C} \nu_c(G) =$

$2+2+1+1 < 7,$

Necessary condition is violated, there is no coloring.

Hall's condition (cont.)

Hall's condition: For every subgraph $H \subseteq G$

$$\sum_{c \in C} \nu_c(H) \geq \sum_{e \in E(H)} x(e)$$

Hall's condition is necessary for the existence of the coloring.

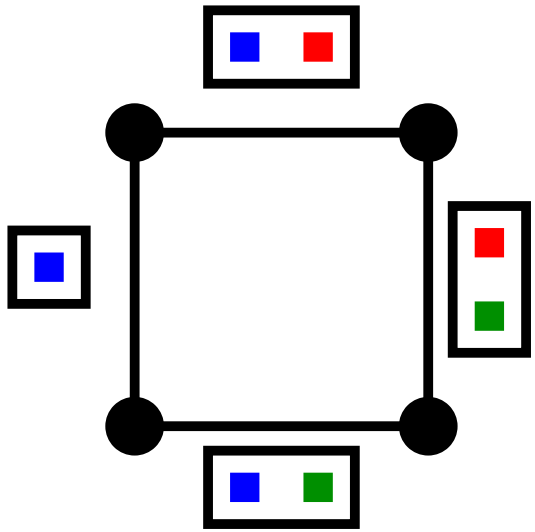
Hall's condition (cont.)

Hall's condition: For every subgraph $H \subseteq G$

$$\sum_{c \in C} \nu_c(H) \geq \sum_{e \in E(H)} x(e)$$

Hall's condition is necessary for the existence of the coloring.

However, it is not sufficient in general:



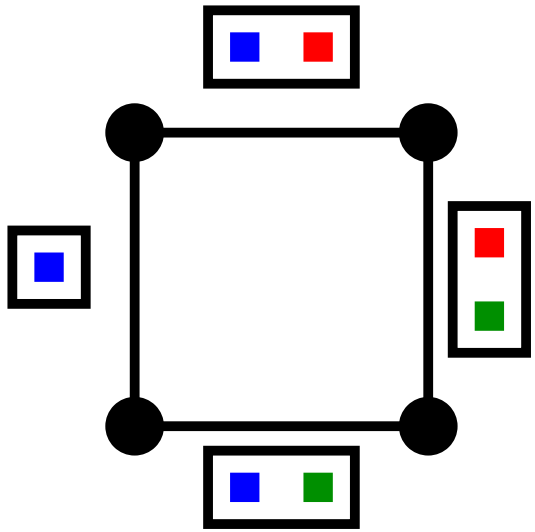
Hall's condition (cont.)

Hall's condition: For every subgraph $H \subseteq G$

$$\sum_{c \in C} \nu_c(H) \geq \sum_{e \in E(H)} x(e)$$

Hall's condition is necessary for the existence of the coloring.

However, it is not sufficient in general:



$\mathbf{2} + \mathbf{1} + \mathbf{1} \geq \mathbf{4}$,
Hall's condition is satisfied,
but there is no coloring.

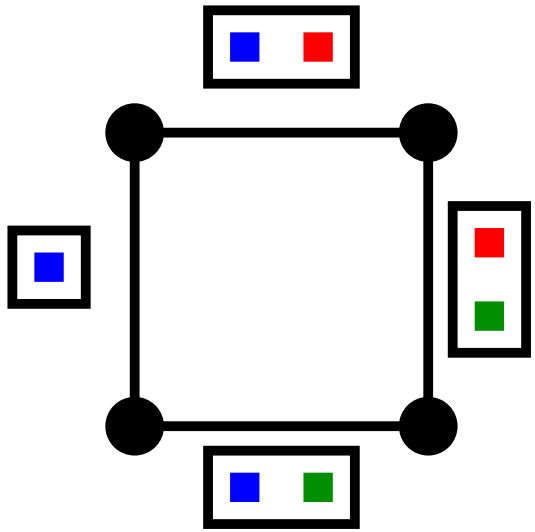
Hall's condition (cont.)

Hall's condition: For every subgraph $H \subseteq G$

$$\sum_{c \in C} \nu_c(H) \geq \sum_{e \in E(H)} x(e)$$

Hall's condition is necessary for the existence of the coloring.

However, it is not sufficient in general:



$2 + 1 + 1 \geq 4$,
Hall's condition is satisfied,
but there is no coloring.

Theorem (Marcotte and Seymour, 1990) If G is a tree, then Hall's condition is sufficient and necessary for list edge multicoloring.

Algorithmic complexity

List edge coloring is **NP**-complete in **complete bipartite graphs**.
(Partial Latin square extension is a special case)

Algorithmic complexity

List edge coloring is **NP**-complete in **complete bipartite graphs**.
(Partial Latin square extension is a special case)

Theorem (Marcotte and Seymour, 1990) If G is a **tree**, then Hall's condition is sufficient and necessary for list edge multicoloring.

Proof is based on the total unimodularity of a network matrix $\Rightarrow$ polynomial time algorithm by reduction to network flow

Algorithmic complexity

List edge coloring is **NP**-complete in **complete bipartite graphs**.
(Partial Latin square extension is a special case)

Theorem (Marcotte and Seymour, 1990) If G is a **tree**, then Hall's condition is sufficient and necessary for list edge multicoloring.

Proof is based on the total unimodularity of a network matrix $\Rightarrow$ polynomial time algorithm by reduction to network flow

G is a **path**: simpler algorithm by Goldwasser and Klostermeyer, 2002

Algorithmic complexity

List edge coloring is **NP**-complete in **complete bipartite graphs**.
(Partial Latin square extension is a special case)

Theorem (Marcotte and Seymour, 1990) If G is a **tree**, then Hall's condition is sufficient and necessary for list edge multicoloring.

Proof is based on the total unimodularity of a network matrix $\Rightarrow$ polynomial time algorithm by reduction to network flow

G is a **path**: simpler algorithm by Goldwasser and Klostermeyer, 2002

? What about **cycles**?
What about **"almost trees"** (graphs having at most k cycles)?

A new problem

List edge multicoloring with **demand on the vertices**

- **Given:** a graph $G(V, E)$, a list $L(e) \subseteq C$ for each edge e and a demand function y: $V \to \mathbb{N}$
- **Find:** an assignment $\Psi(e) \subseteq L(e)$ of colors to every edge e, such that adjacent edges receive disjoint sets and there are $y(v)$ **colors in total on the edges incident to** v

Incidence matrix

Incidence matrix **B**:

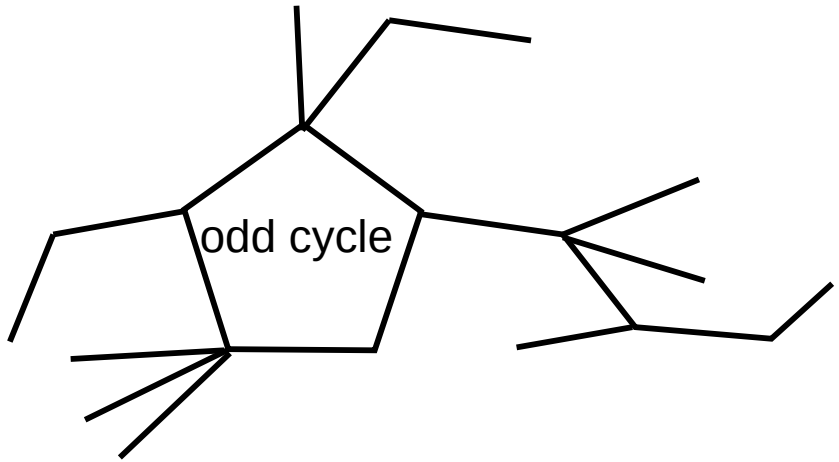
edges

vertices

$$\begin{pmatrix} & 0 & \\ \cdots & 1 & \cdots \\ & 0 & \\ \cdots & 1 & \cdots \\ & 0 & \end{pmatrix}$$

Definition: a graph has **full edge rank** if the columns of its incidence matrix are linearly independent.

A connected graph has full edge rank if and only if it is a tree, or it has only one cycle, and this cycle is odd.



Connections between the two problems

Problem 1

List edge multicoloring with demand on the edges $(G, \mathbf{x})$

$\mathbf{y} = \mathbf{B}\mathbf{x}$

$\Rightarrow$

Problem 2

List edge multicoloring with demand on the vertices $(G, \mathbf{y})$

Connections between the two problems

Problem 1		**Problem 2**
List edge multicoloring with demand on the edges $(G, \mathbf{x})$	$\mathbf{y} = \mathbf{Bx}$ $\Rightarrow$	List edge multicoloring with demand on the vertices $(G, \mathbf{y})$

Every solution for **Problem 1** is a also a solution for **Problem 2**:
$y(v) = \sum_{e \ni v} x(e)$ is the number of colors assigned to the edges incident to vertex v

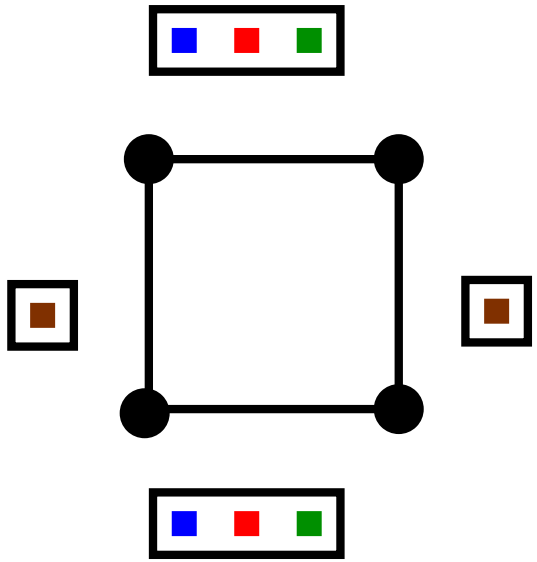
Connections between the two problems

Problem 1		**Problem 2**
List edge multicoloring with demand on the edges $(G, \mathbf{x})$	$\mathbf{y} = \mathbf{Bx}$ $\Rightarrow$	List edge multicoloring with demand on the vertices $(G, \mathbf{y})$

Every solution for **Problem 1** is a also a solution for **Problem 2**:
$y(v) = \sum_{e \ni v} x(e)$ is the number of colors assigned to the edges incident to vertex v

The other direction is not true:

- **Problem 1** (with $x(e) = 2$) has no solution,
- **Problem 2** (with $y(v) = 4$) has a solution.



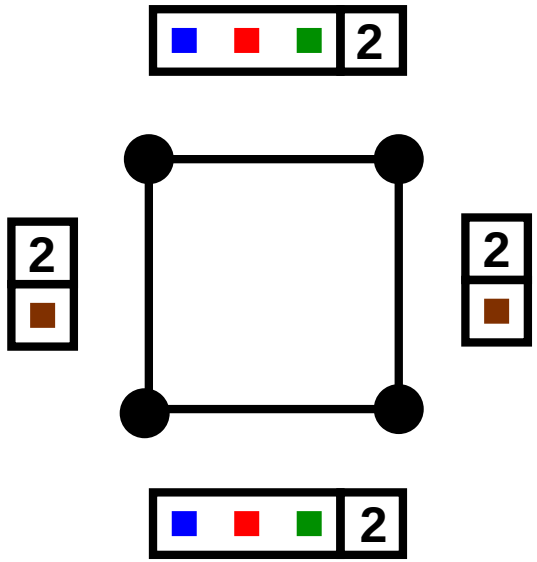
Connections between the two problems

Problem 1		**Problem 2**
List edge multicoloring with demand on the edges $(G, \mathbf{x})$	$\mathbf{y} = \mathbf{Bx}$ $\Rightarrow$	List edge multicoloring with demand on the vertices $(G, \mathbf{y})$

Every solution for **Problem 1** is a also a solution for **Problem 2**:
$y(v) = \sum_{e \ni v} x(e)$ is the number of colors assigned to the edges incident to vertex v

The other direction is not true:

- **Problem 1** (with $x(e) = 2$) has no solution,
- **Problem 2** (with $y(v) = 4$) has a solution.



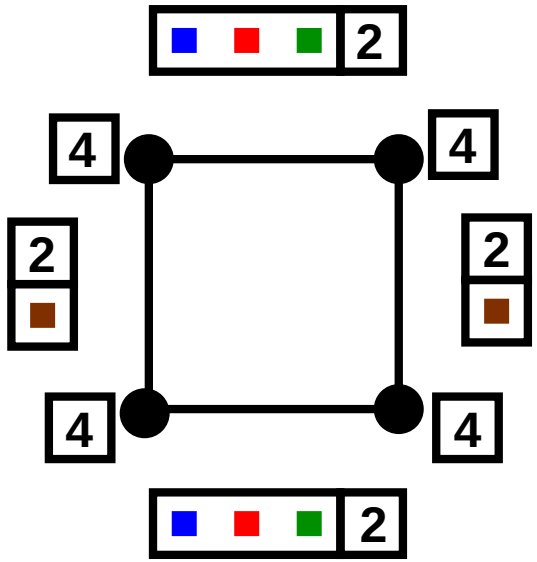
Connections between the two problems

Problem 1		**Problem 2**
List edge multicoloring with demand on the edges $(G, \mathbf{x})$	$\mathbf{y} = \mathbf{Bx}$ $\Rightarrow$	List edge multicoloring with demand on the vertices $(G, \mathbf{y})$

Every solution for **Problem 1** is a also a solution for **Problem 2**:
$y(v) = \sum_{e \ni v} x(e)$ is the number of colors assigned to the edges incident to vertex v

The other direction is not true:

- **Problem 1** (with $x(e) = 2$) has no solution,
- **Problem 2** (with $y(v) = 4$) has a solution.



Connections between the two problems

Problem 1
List edge multicoloring
with demand on the edges
$(G, \mathbf{x})$

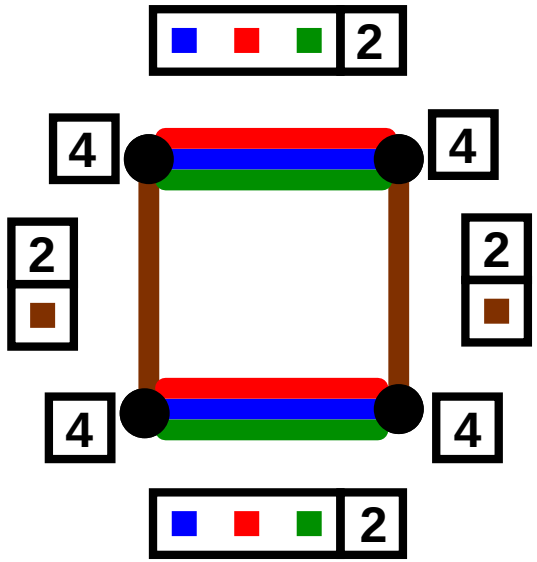
$\mathbf{y} = \mathbf{B}\mathbf{x}$
$\Rightarrow$

Problem 2
List edge multicoloring
with demand on the vertices
$(G, \mathbf{y})$

Every solution for **Problem 1** is a also a solution for **Problem 2**:
$y(v) = \sum_{e \ni v} x(e)$ is the number of colors assigned to the edges incident to vertex v

The other direction is not true:

- **Problem 1** (with $x(e) = 2$) has no solution,
- **Problem 2** (with $y(v) = 4$) has a solution.



Connections between the two problems (cont.)

Problem 1		**Problem 2**
List edge multicoloring with demand on the edges $(G, \mathbf{x})$	$\mathbf{y} = \mathbf{Bx}$ $\Rightarrow$	List edge multicoloring with demand on the vertices $(G, \mathbf{y})$

However, if G has full edge rank, then every solution for **Problem 2** is also a solution for **Problem 1**. There is only one vector $\mathbf{x}$ that satisfies $\mathbf{Bx} = \mathbf{y} \Rightarrow$ if the demand of the vertices are satisfied, then the demand of the edges are also satisfied.

$\Downarrow$

Problem 1 and **Problem 2** are equivalent in graphs with full edge rank.

Connections between the two problems (cont.)

Problem 1		**Problem 2**
List edge multicoloring with demand on the edges $(G, \mathbf{x})$	$\mathbf{y} = \mathbf{Bx}$ $\Rightarrow$	List edge multicoloring with demand on the vertices $(G, \mathbf{y})$

However, if G has full edge rank, then every solution for **Problem 2** is also a solution for **Problem 1**. There is only one vector $\mathbf{x}$ that satisfies $\mathbf{Bx} = \mathbf{y} \Rightarrow$ if the demand of the vertices are satisfied, then the demand of the edges are also satisfied.

$\Downarrow$

Problem 1 and **Problem 2** are equivalent in graphs with full edge rank.

Problem 2 can be solved in polynomial time for **any** graph: reduction to perfect matching.

$\Downarrow$

List edge multicoloring can be solved in polynomial time for graphs with full edge rank (incuding odd cycles).

Bounded cyclicity graphs

Randomized polynomial time algorithm for list edge multicoloring in connected graphs with $|V| + k$ edges (for every fixed k):

- if there is no solution, algorithm says **NO** with probabilty 1
- if there is solution, algorithm says **YES** with probability $\geq \frac{1}{2}$

Bounded cyclicity graphs

Randomized polynomial time algorithm for list edge multicoloring in connected graphs with $|V| + k$ edges (for every fixed k):

- if there is no solution, algorithm says **NO** with probabilty 1
- if there is solution, algorithm says **YES** with probability $\geq \frac{1}{2}$

Method:
Reduction to the **exact matching** problem: given a graph with some of its edges colored red, find a perfect matching with **exactly** m red edges (known to be solvable in randomized polynomial time, Mulmuley, Vazirani, Vazirani, 1987).

More generally:
Given a partition $E_1, E_2, \ldots, E_k$ of the edges, find a perfect matching with **exactly** ℓ_i edges from E_i.

Conclusions

- **Previous result**: list edge multicoloring can be solved in polynomial time for **trees**
- **New result**: more general polynomial algorithm for **full edge rank** graphs
- **New result**: randomized algorithm for **bounded cyclicity graphs**
- **Question:** deterministic algorithm for **even cycles**?