On Maximum Common Subgraph Problems in Series-Parallel Graphs

Nils Kriege **Florian Kurpicz** Petra Mutzel

Dept. of Computer Science, TU Dortmund

25th International Workshop on Combinatorial Algorithms

Duluth, MN, 15 – 17 October, 2014

Maximum Common Subgraph Problem

Definition (Maximum Common Induced Subgraph (MCIS))

Input: Graphs G, H.

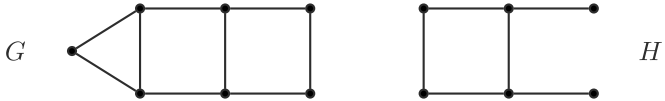
Output: Maximum number of vertices in a connected graph that is isomorphic to an induced subgraph of G and H.

Maximum Common Subgraph Problem

Definition (Maximum Common Induced Subgraph (MCIS))

Input: Graphs G, H.

Output: Maximum number of vertices in a connected graph that is isomorphic to an induced subgraph of G and H.

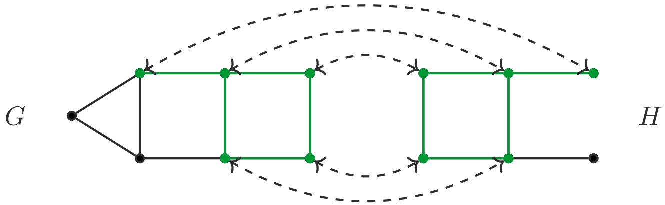


Maximum Common Subgraph Problem

Definition (Maximum Common Induced Subgraph (MCIS))

Input: Graphs G, H.

Output: Maximum number of vertices in a connected graph that is isomorphic to an induced subgraph of G and H.

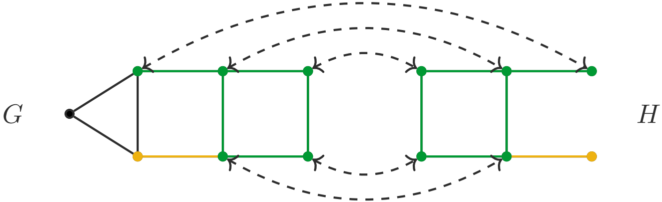


Maximum Common Subgraph Problem

Definition (Maximum Common Induced Subgraph (MCIS))

Input: Graphs G, H.

Output: Maximum number of vertices in a connected graph that is isomorphic to an induced subgraph of G and H.

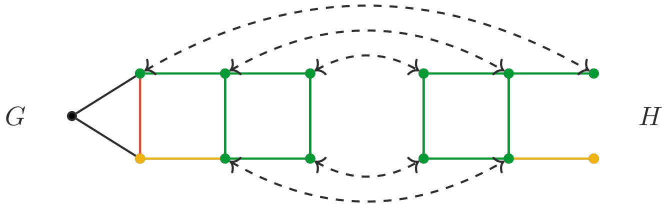


Maximum Common Subgraph Problem

Definition (Maximum Common Induced Subgraph (MCIS))

Input: Graphs G, H.

Output: Maximum number of vertices in a connected graph that is isomorphic to an induced subgraph of G and H.

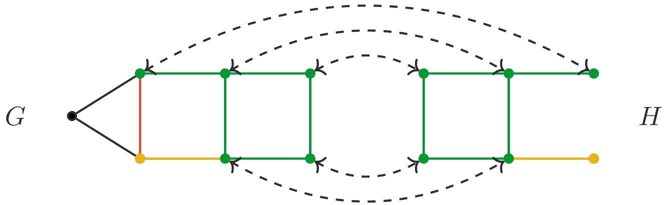


Maximum Common Subgraph Problem

Definition (Maximum Common Induced Subgraph (MCIS))

Input: Graphs G, H.

Output: Maximum number of vertices in a connected graph that is isomorphic to an induced subgraph of G and H.

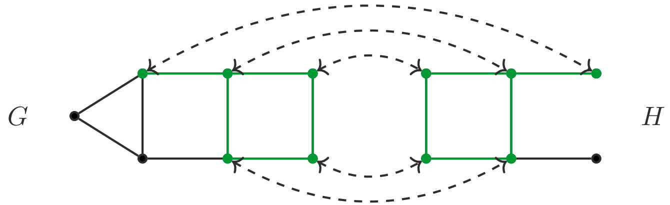


Maximum Common Subgraph Problem

Definition (Maximum Common Induced Subgraph (MCIS))

Input: Graphs G, H.

Output: Maximum number of vertices in a connected graph that is isomorphic to an induced subgraph of G and H.



The Considered Graph Class

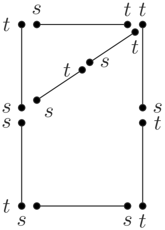
Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

The Considered Graph Class

Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

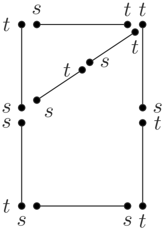


The Considered Graph Class

Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

S-Operation: Join an s- and a t-node in series.

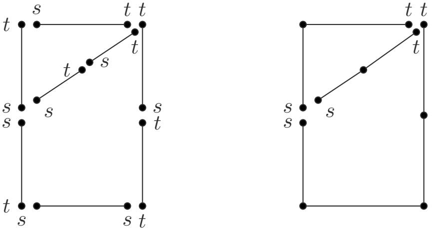


The Considered Graph Class

Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

S-Operation: Join an s- and a t-node in series.

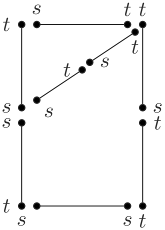


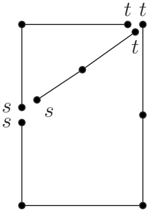
The Considered Graph Class

Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

S-Operation: Join an s- and a t-node in series.





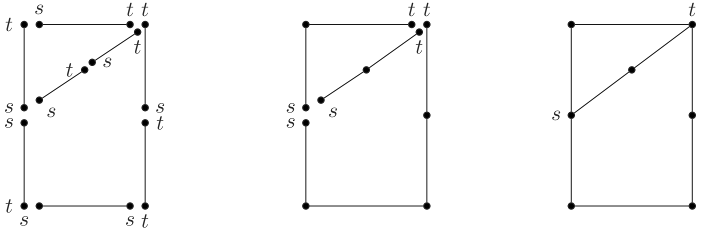
The Considered Graph Class

Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

S-Operation: Join an s- and a t-node in series.

P-Operation: Join two s- and t-nodes in parallel.



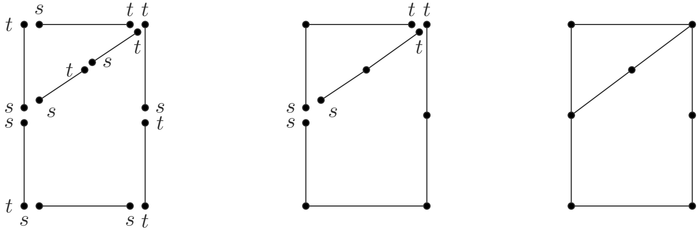
The Considered Graph Class

Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

S-Operation: Join an s- and a t-node in series.

P-Operation: Join two s- and t-nodes in parallel.



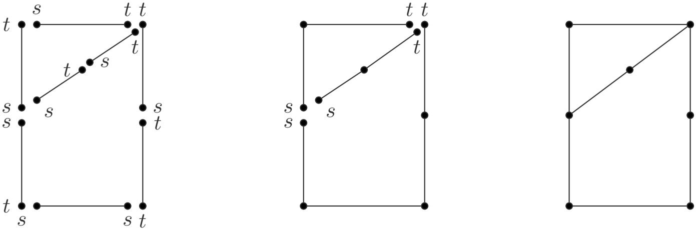
The Considered Graph Class

Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

S-Operation: Join an s- and a t-node in series.

P-Operation: Join two s- and t-nodes in parallel.



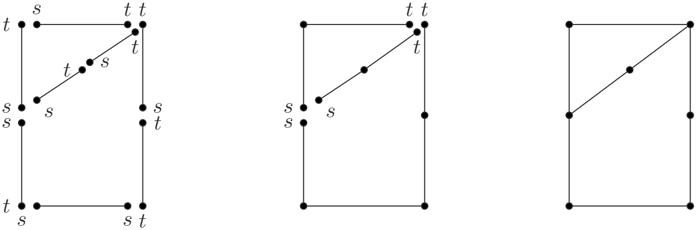
The Considered Graph Class

Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

S-Operation: Join an s- and a t-node in series.

P-Operation: Join two s- and t-nodes in parallel.



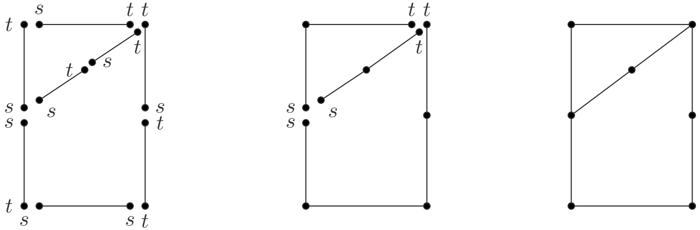
The Considered Graph Class

Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

S-Operation: Join an s- and a t-node in series.

P-Operation: Join two s- and t-nodes in parallel.



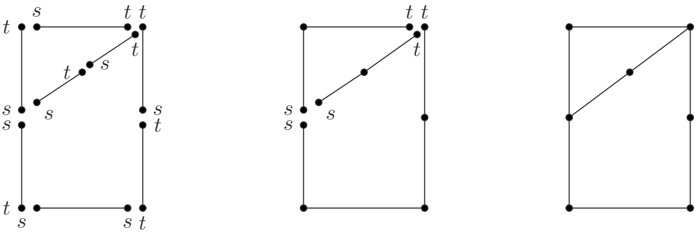
The Considered Graph Class

Definition (Series-Parallel Graphs (SPGs))

In a SPG each block can be constructed from a finite set of $K_2^{s,t}$'s with:

S-Operation: Join an s- and a t-node in series.

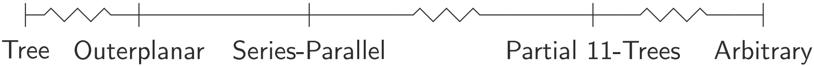
P-Operation: Join two s- and t-nodes in parallel.



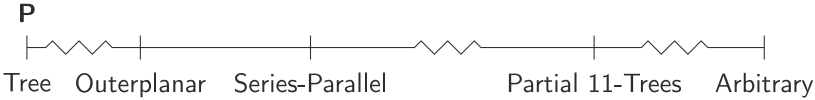
Lemma [Brandstadt et al., 1999]

G is series-parallel $\iff$ G is a partial 2-tree.

The Complexity of the Problem

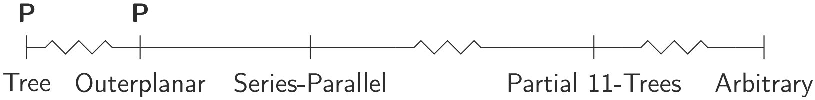


The Complexity of the Problem



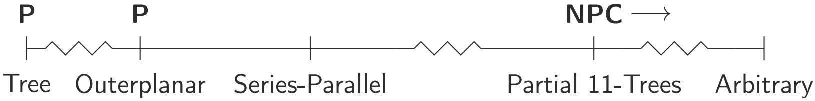
- Trees in **P** [Matula, 1978]

The Complexity of the Problem



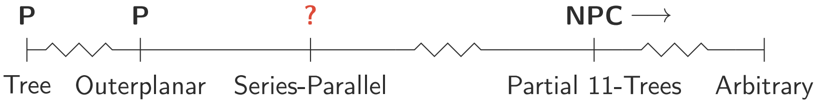
- Trees in **P** [Matula, 1978]
- Outerplanar with **bounded** degree in **P** [Akutsu & Tamura, 2013]

The Complexity of the Problem



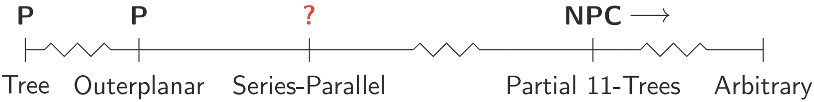
- Trees in **P** [Matula, 1978]
- Outerplanar with **bounded** degree in **P** [Akutsu & Tamura, 2013]
- Partial 11-trees with **bounded** degree in **NPC** [Akutsu & Tamura, 2013]

The Complexity of the Problem



- Trees in **P** [Matula, 1978]
- Outerplanar with **bounded** degree in **P** [Akutsu & Tamura, 2013]
- Partial 11-trees with **bounded** degree in **NPC** [Akutsu & Tamura, 2013]

The Complexity of the Problem

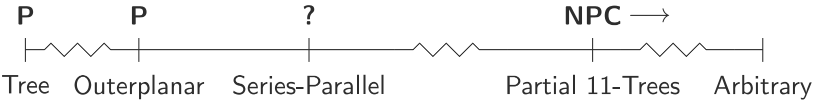


- Trees in **P** [Matula, 1978]
- Outerplanar with **bounded** degree in **P** [Akutsu & Tamura, 2013]
- Partial 11-trees with **bounded** degree in **NPC** [Akutsu & Tamura, 2013]

In this Talk

1. Proof of NP-hardness if degree is not bounded for all vertices.
2. Polynomial time algorithm for restricted feasible solutions.

The Complexity of the Problem

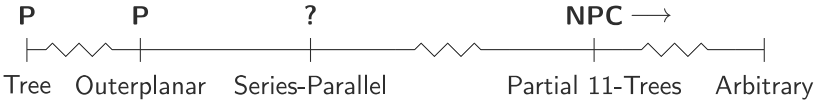


- Trees in **P** [Matula, 1978]
- Outerplanar with **bounded** degree in **P** [Akutsu & Tamura, 2013]
- Partial 11-trees with **bounded** degree in **NPC** [Akutsu & Tamura, 2013]

In this Talk

1. Proof of NP-hardness if degree is not bounded for all vertices.
2. Polynomial time algorithm for restricted feasible solutions.

The Complexity of the Problem



- Trees in **P** [Matula, 1978]
- Outerplanar with **bounded** degree in **P** [Akutsu & Tamura, 2013]
- Partial 11-trees with **bounded** degree in **NPC** [Akutsu & Tamura, 2013]

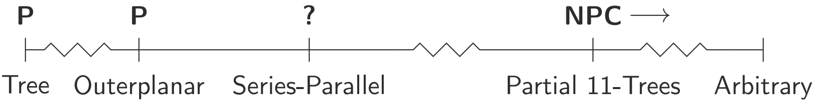
In this Talk

1. Proof of NP-hardness if degree is not bounded for all vertices.
2. Polynomial time algorithm for restricted feasible solutions.

Theorem [Gupta & Nishimura 1994]

SI in partial k-trees with degree $\leq k+2$ for all but k vertices is in **NPC**.

The Complexity of the Problem



- Trees in **P** [Matula, 1978]
- Outerplanar with **bounded** degree in **P** [Akutsu & Tamura, 2013]
- Partial 11-trees with **bounded** degree in **NPC** [Akutsu & Tamura, 2013]

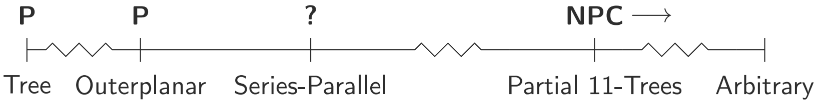
In this Talk

1. Proof of NP-hardness if degree is not bounded for all vertices.
2. Polynomial time algorithm for restricted feasible solutions.

Theorem [Gupta & Nishimura 1994]

SI in SPGs with degree ≤ 4 for all but 2 vertices is in **NPC**.

The Complexity of the Problem



- Trees in **P** [Matula, 1978]
- Outerplanar with **bounded** degree in **P** [Akutsu & Tamura, 2013]
- Partial 11-trees with **bounded** degree in **NPC** [Akutsu & Tamura, 2013]

In this Talk

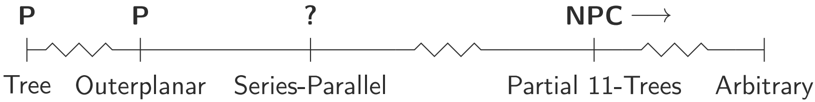
1. Proof of NP-hardness if degree is not bounded for all vertices.
2. Polynomial time algorithm for restricted feasible solutions.

Theorem [Gupta & Nishimura 1994]

SI in SPGs with degree ≤ 4 for all but 2 vertices is in **NPC**.

SI$^{\leq 4,2}$: **NP**-complete ✓ and SI $\leq_p$ MCIS

The Complexity of the Problem



- Trees in **P** [Matula, 1978]
- Outerplanar with **bounded** degree in **P** [Akutsu & Tamura, 2013]
- Partial 11-trees with **bounded** degree in **NPC** [Akutsu & Tamura, 2013]

In this Talk

1. Proof of NP-hardness if degree is not bounded for all vertices.
2. Polynomial time algorithm for restricted feasible solutions.

Theorem [Gupta & Nishimura 1994]

SI in SPGs with degree ≤ 4 for all but 2 vertices is in **NPC**.

$\text{SI}^{\leq 4,2}$: **NP**-complete ✓ and SI $\leq_p$ MCIS

$\text{MCIS}^{\leq 3,1}$: **NP**-hard → now

Reduction from the . . .

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?

Reduction from the ...

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \dots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \dots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \dots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \dots, n$?

Reduction from the . . .

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?

X	1	3	2	1	4	3

Y	3	1	4	2	3	1

Reduction from the ...

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?

X	1	3	2	1	4	3

Y	3	1	4	2	3	1

Reduction from the . . .

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?

X					
1	3	2	1	4	3

Y					
3	1	4	2	3	1

$\vec{b}$					
3	6	2	4	6	7

Reduction from the . . .

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?

X	1	3	2	1	4	3

Y	3	1	4	2	3	1

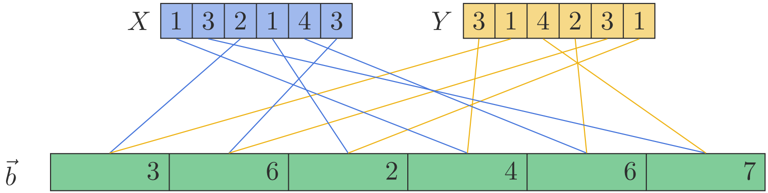
$\vec{b}$	3	6	2	4	6	7

Reduction from the ...

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?

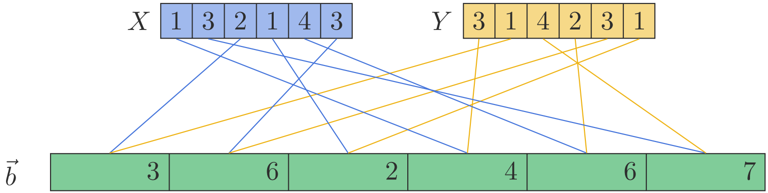


Reduction from the ...

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?

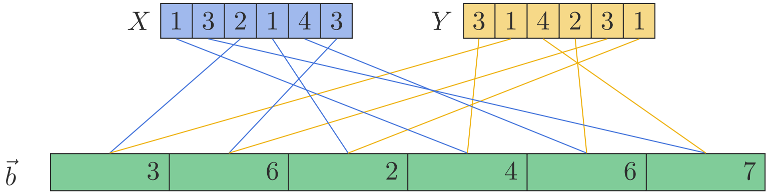


Reduction from the ...

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?

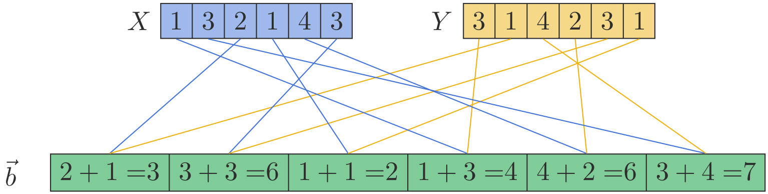


Reduction from the ...

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \dots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \dots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \dots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \dots, n$?

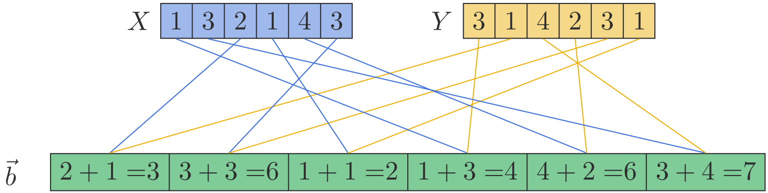


Reduction from the ...

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?

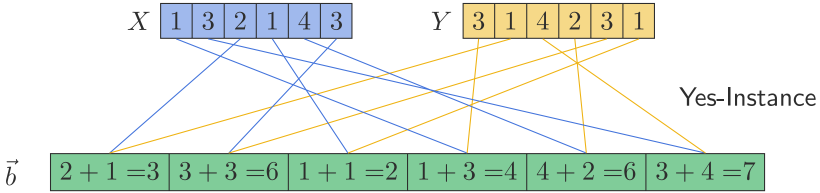


Reduction from the ...

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?

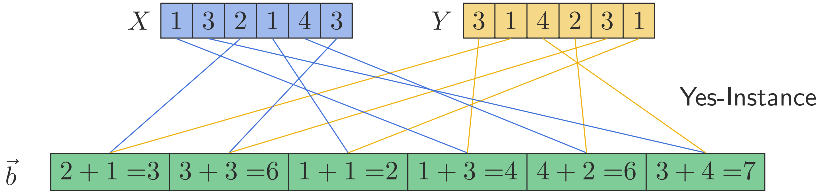


Reduction from the ...

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \ldots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \ldots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \ldots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \ldots, n$?



Definition (NP-complete in the strong sense) [Garey & Johnson 1979]

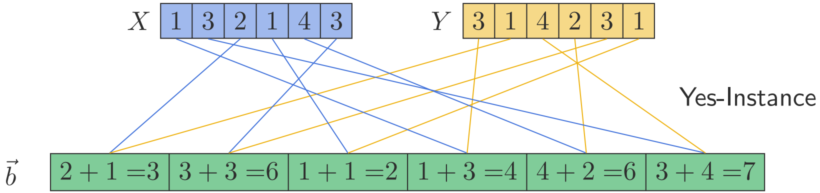
- **NPC** in the strong sense $\iff$ **NPC** with unary encoded input.

Reduction from the . . .

Definition (Numerical Matching with Target Sums (NMwTS))

Input: Two multisets of integers X and Y with $|X| = |Y| = n$ and a vector $\vec{b} = \langle b_1, \dots, b_n \rangle$ with $b_i \in \mathbb{N}_0$ for all $i = 1, \dots, n$.

Question: Can $X \cup Y$ be partitioned into disjoint sets $A_1, \dots, A_n$ each containing one element from X and Y, such that $\sum_{a \in A_i} v(a) = b_i$ for all $i = 1, \dots, n$?



Definition (NP-complete in the strong sense) [Garey & Johnson 1979]

- **NPC** in the strong sense $\iff$ **NPC** with unary encoded input.
- NMwTS is **NPC** in the strong sense.

Construction of the Reduction

Outline for each NMwTS-Instance $(X, Y, s, \vec{b})$

- Represent X, Y and $\vec{b}$ as graphs G and H.
- Size of MCIS indicates type of instance.

Construction of the Reduction

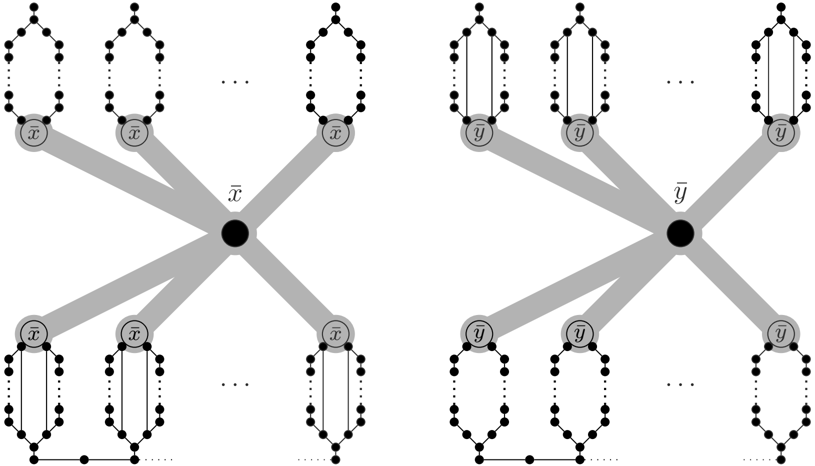
Outline for each NMwTS-Instance $(X, Y, s, \vec{b})$

- Represent X, Y and $\vec{b}$ as graphs G and H.
- Size of MCIS indicates type of instance.

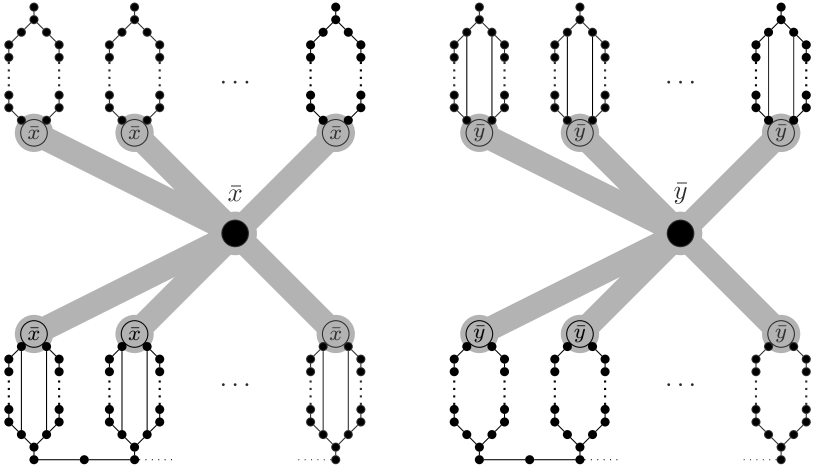
Parts of the Graphs

1. Base-Gadgets $\rightarrow$ Structure of the MCIS.
2. Encoding of the sizes of X, Y and $\vec{b}$ $\rightarrow$ Size of the MCIS.

The Base-Gadgets of G and H

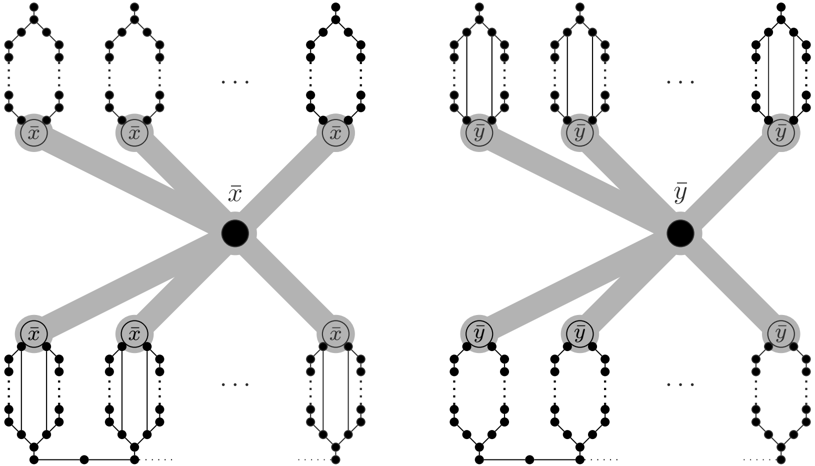


The Base-Gadgets of G and H



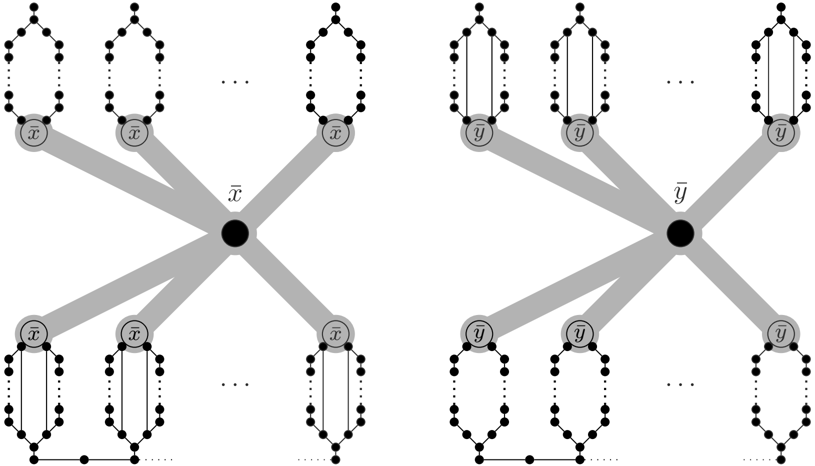
- $\bar{x}, \bar{y}$ with unbounded degree.

The Base-Gadgets of G and H



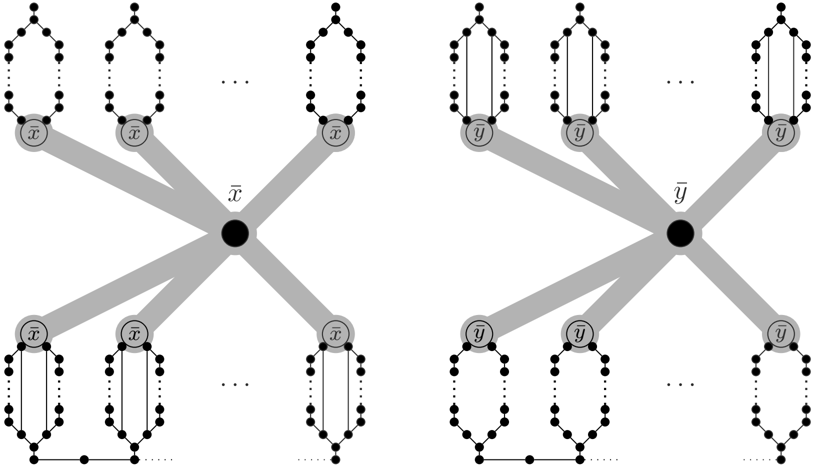
- $\bar{x}, \bar{y}$ with unbounded degree.
- Cycles: $2\sum_{z \in X \cup Y} v(z) + 2$ vertices.

The Base-Gadgets of G and H



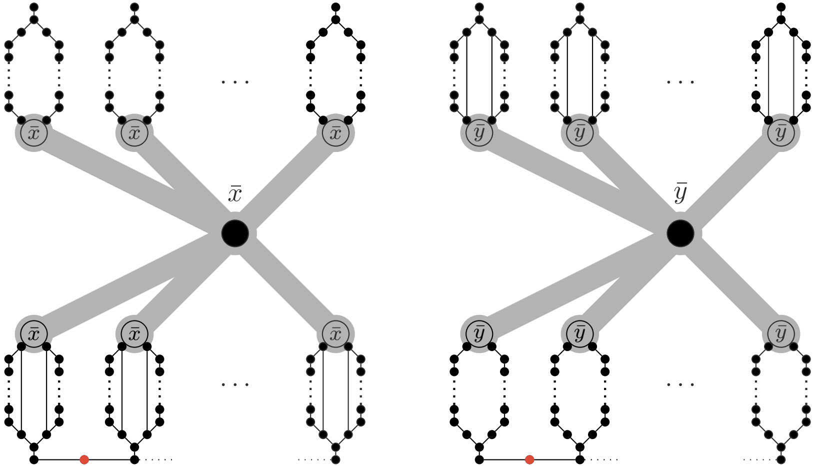
- $\bar{x}, \bar{y}$ with unbounded degree.
- Cycles: $2\sum_{z \in X \cup Y} v(z) + 2$ vertices.
- Anchor vertices and paths.

The Base-Gadgets of G and H



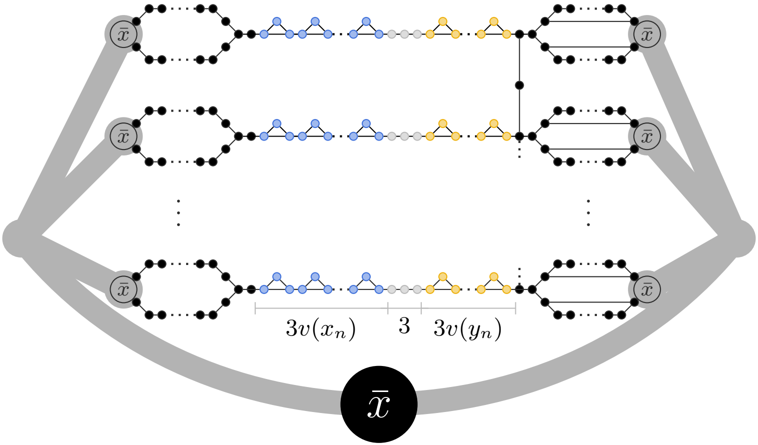
- $\bar{x}, \bar{y}$ with unbounded degree.
- Cycles: $2\sum_{z \in X \cup Y} v(z) + 2$ vertices.
- Anchor vertices and paths.
- n chordsless cycles and n cycles containing chords.

The Base-Gadgets of G and H



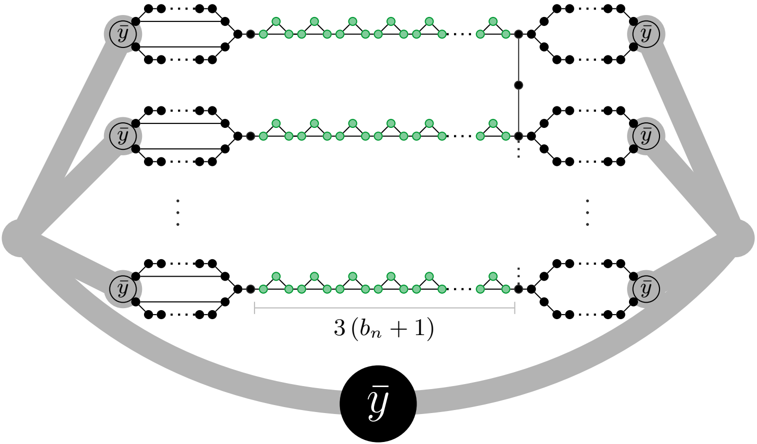
- $\bar{x}, \bar{y}$ with unbounded degree.
- Cycles: $2\sum_{z \in X \cup Y} v(z) + 2$ vertices.
- Anchor vertices and paths.
- n chordsless cycles and n cycles containing chords.

Encode the Sizes of X and Y



- Values $v(x_i)$ and $v(y_i)$
- Separating paths

Encode the Sizes of $\vec{b}$



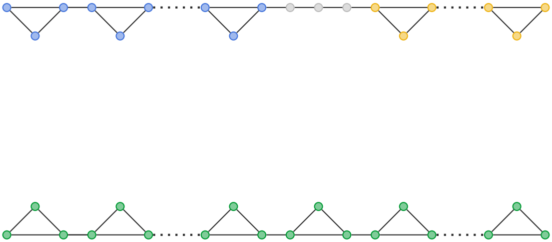
- Entries of $\vec{b}$
- $+1$ to compensate the separating paths

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1 = |V(H)| - 2n + 1$$

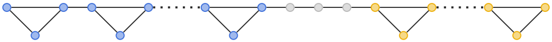


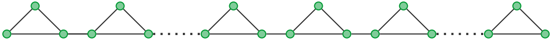
The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1$$
$$= |V(H)| - 2n + 1$$





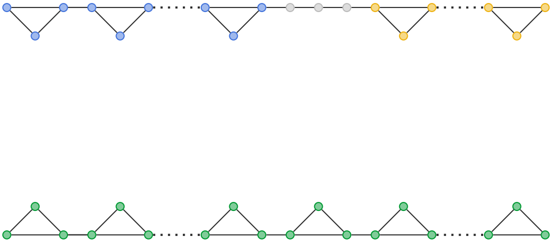
- $|V(G)| = |V(H)| \rightarrow$ Base-Gadgets & Encoding.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1 = |V(H)| - 2n + 1$$



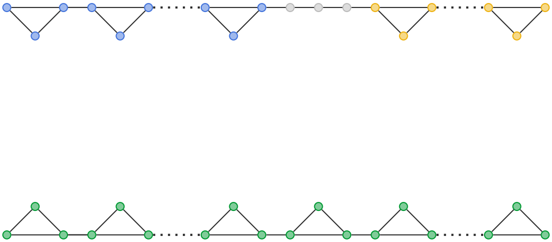
- $|V(G)| = |V(H)| \rightarrow$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1$$
$$= |V(H)| - 2n + 1$$



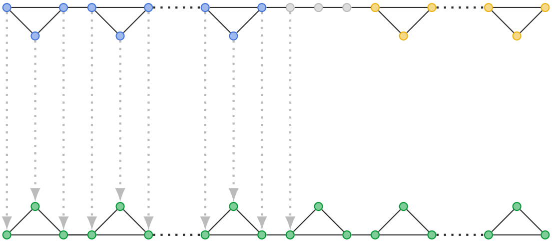
- $|V(G)| = |V(H)| \rightarrow$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.
- n missing vertices in the separating paths.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1 = |V(H)| - 2n + 1$$



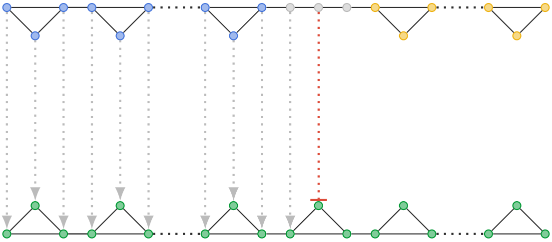
- $|V(G)| = |V(H)| \rightarrow$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.
- n missing vertices in the separating paths.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1$$
$$= |V(H)| - 2n + 1$$



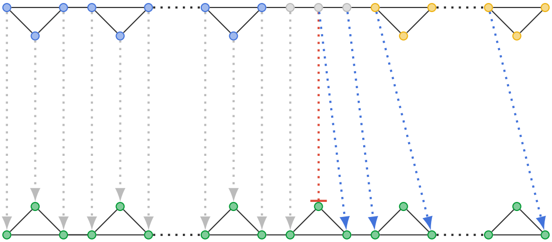
- $|V(G)| = |V(H)| \rightarrow$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.
- n missing vertices in the separating paths.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1 = |V(H)| - 2n + 1$$



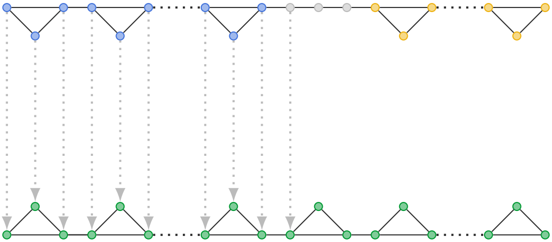
- $|V(G)| = |V(H)| \to$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.
- n missing vertices in the separating paths.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1$$
$$= |V(H)| - 2n + 1$$



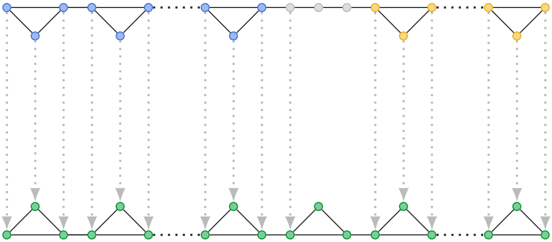
- $|V(G)| = |V(H)| \rightarrow$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.
- n missing vertices in the separating paths.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1 = |V(H)| - 2n + 1$$



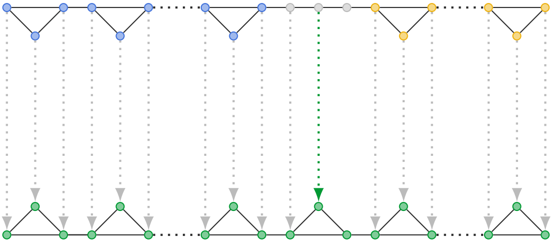
- $|V(G)| = |V(H)| \rightarrow$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.
- n missing vertices in the separating paths.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1 = |V(H)| - 2n + 1$$



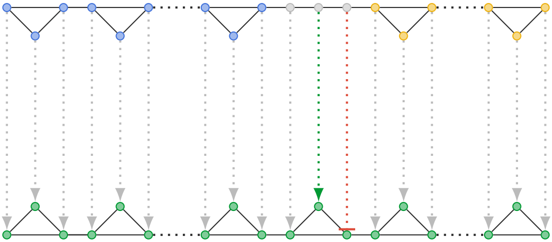
- $|V(G)| = |V(H)| \rightarrow$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.
- n missing vertices in the separating paths.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1 = |V(H)| - 2n + 1$$



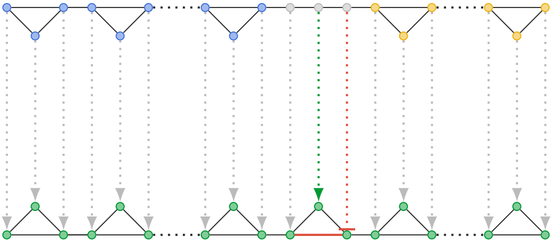
- $|V(G)| = |V(H)| \to$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.
- n missing vertices in the separating paths.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance } \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1$$
$$= |V(H)| - 2n + 1$$



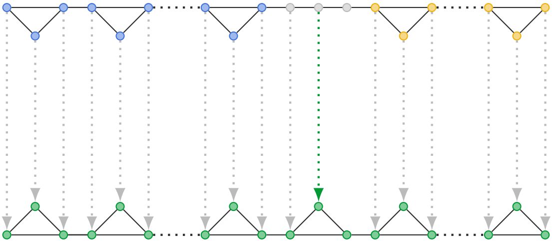
- $|V(G)| = |V(H)| \rightarrow$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.
- n missing vertices in the separating paths.

The Reduction (Sketch)

Theorem

MCIS in SPGs with degree ≤ 3 for all but 1 vertex is **NP**-hard.

$$(X, Y, s, \vec{b}) \text{ is a Yes-Instance} \iff \text{MCIS}(G, H) = |V(G)| - 2n + 1$$
$$= |V(H)| - 2n + 1$$



- $|V(G)| = |V(H)| \rightarrow$ Base-Gadgets & Encoding.
- $n - 1$ missing vertices in the base-gadget.
- n missing vertices in the separating paths.

A Polynomial Time Algorithm

Definition (k-Maximum Common Subgraph (k-MCS))

Input: k-connected partial k-trees G, H.

Output: Maximum common k-connected subgraph of G and H.

A Polynomial Time Algorithm

Definition (k-Maximum Common Subgraph (k-MCS))

Input: k-connected partial k-trees G, H.

Output: Maximum common k-connected subgraph of G and H.

Known Variants of k-MCS in P

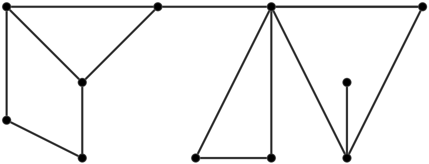
1-MCS $\Leftrightarrow$ Maximum Common Subtree Problem ✓ [Matula, 1978]

2-MCIS in SPGs ✓ [Kriege & Mutzel, 2014]

Graph Decomposition

Definition (BC-Tree)

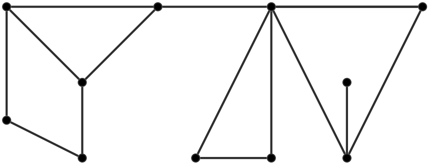
- Tree consisting of a node for each cut vertex and block.
- Distinguish between bridges and non-bridge blocks.
- Edges between nodes if cut vertex is contained in block.



Graph Decomposition

Definition (BC-Tree)

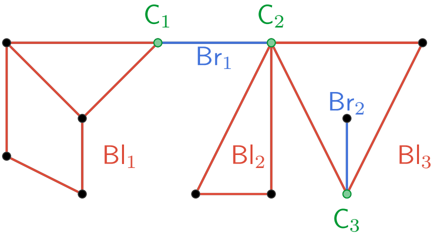
- Tree consisting of a node for each cut vertex and block.
- Distinguish between bridges and non-bridge blocks.
- Edges between nodes if cut vertex is contained in block.



Graph Decomposition

Definition (BC-Tree)

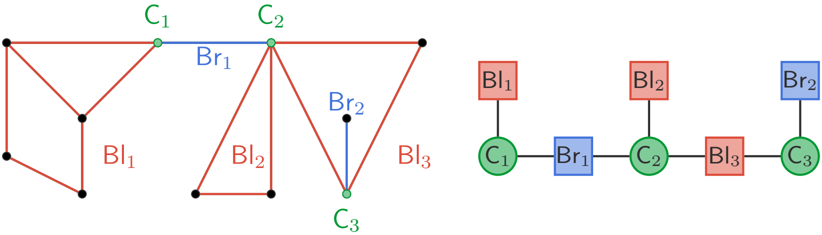
- Tree consisting of a node for each cut vertex and block.
- Distinguish between bridges and non-bridge blocks.
- Edges between nodes if cut vertex is contained in block.



Graph Decomposition

Definition (BC-Tree)

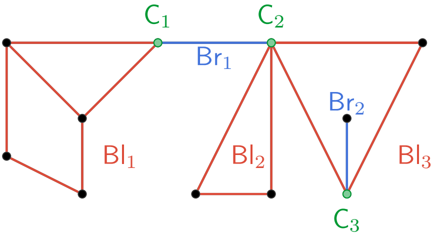
- Tree consisting of a node for each cut vertex and block.
- Distinguish between bridges and non-bridge blocks.
- Edges between nodes if cut vertex is contained in block.

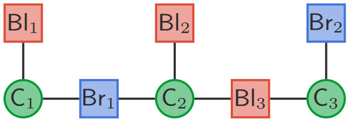


Graph Decomposition

Definition (BC-Tree)

- Tree consisting of a node for each cut vertex and block.
- Distinguish between bridges and non-bridge blocks.
- Edges between nodes if cut vertex is contained in block.





- BC-tree is bipartite with respect to cut vertices and blocks.
- Non-bride blocks represent 2-connected SPGs.

Fewer Restrictions

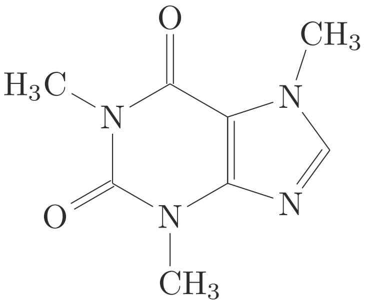
Definition (Block-and-Bridge Preserving MCS (BBP-MCS))

- Variant of the general common subgraph problem [Schietgat et al., 2007]
- Non-Bridge blocks are only mapped onto non-bridge blocks.
- Bridges are only mapped onto bridges.

Fewer Restrictions

Definition (Block-and-Bridge Preserving MCS (BBP-MCS))

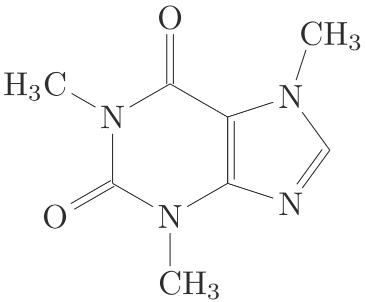
- Variant of the general common subgraph problem [Schietgat et al., 2007]
- Non-Bridge blocks are only mapped onto non-bridge blocks.
- Bridges are only mapped onto bridges.



Fewer Restrictions

Definition (Block-and-Bridge Preserving MCS (BBP-MCS))

- Variant of the general common subgraph problem [Schietgat et al., 2007]
- Non-Bridge blocks are only mapped onto non-bridge blocks.
- Bridges are only mapped onto bridges.



- Only compare blocks of the same type.
- There is an algorithm for 2-MCIS.

Idea of the Algorithm

Edmonds-Matula Algorithm for 1-MCS (Sketch)

1. Decompose trees into rooted subtrees.
2. Compute MCS of subtrees (*Maximum Weighted Bipartite Matching*).
3. Combine partial solutions.

Idea of the Algorithm

Edmonds-Matula Algorithm for 1-MCS (Sketch)

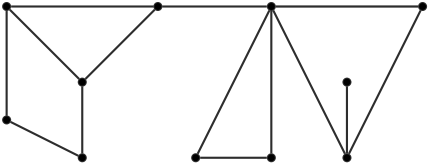
1. Decompose trees into rooted subtrees.
2. Compute MCS of subtrees (*Maximum Weighted Bipartite Matching*).
3. Combine partial solutions.

Algorithm for BBP-MCIS (Sketch)

1. Decompose BC-trees into rooted BC-subtrees.
2. Compute MCS of subgraphs represented by subtrees:
   - *Maximum Weighted Bipartite Matching*
   - Extended 2-MCIS algorithm → work with cut vertices
3. Combine partial solutions.

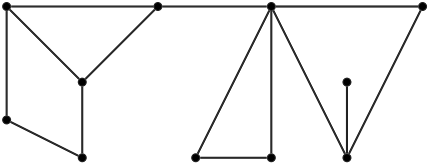
Rooting a BC-Tree

- The root is a vertex in the input graph.
- A subgraph can now be induced by a BC-subtree.



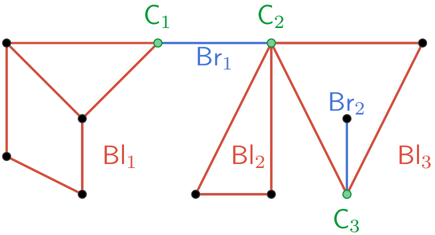
Rooting a BC-Tree

- The root is a vertex in the input graph.
- A subgraph can now be induced by a BC-subtree.



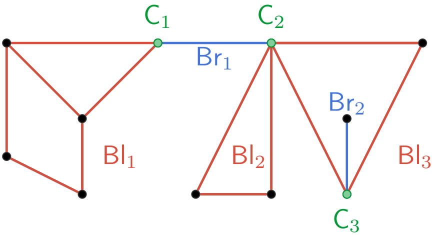
Rooting a BC-Tree

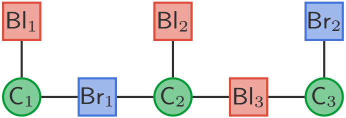
- The root is a vertex in the input graph.
- A subgraph can now be induced by a BC-subtree.



Rooting a BC-Tree

- The root is a vertex in the input graph.
- A subgraph can now be induced by a BC-subtree.





Analysis

Theorem

BBP-MCIS in SPGs can be solved in $\mathcal{O}\left(n^6\right)$, where $n = |V(G)| + |V(H)|$.

Analysis

Theorem

BBP-MCIS in SPGs can be solved in $\mathcal{O}\left(n^6\right)$, where $n = |V(G)| + |V(H)|$.

Proof (Sketch)

- (Extended) 2-MCIS can be solved in $\mathcal{O}\left(n^6\right)$.
- Comparison of all blocks at cut vertices in $\mathcal{O}\left(n^5\right)$ $\rightarrow$ at most $\mathcal{O}\left(n^2\right)$ pairs of cut vertices and MWBM is solvable in $\mathcal{O}\left(n^3\right)$.
- Total time $\rightarrow \mathcal{O}\left(n^6\right)$.

Analysis

Polynomial Time Algorithm

Theorem

BBP-MCIS in SPGs can be solved in $\mathcal{O}\left(n^6\right)$, where $n = |V(G)| + |V(H)|$.

Proof (Sketch)

- (Extended) 2-MCIS can be solved in $\mathcal{O}\left(n^6\right)$.
- Comparison of all blocks at cut vertices in $\mathcal{O}\left(n^5\right) \to$ at most $\mathcal{O}\left(n^2\right)$ pairs of cut vertices and MWBM is solvable in $\mathcal{O}\left(n^3\right)$.
- Total time $\to \mathcal{O}\left(n^6\right)$.

Theorem

BBP-MCIS in outerplanar graphs can be solved in $\mathcal{O}\left(n^5\right)$.

Analysis

Theorem

BBP-MCIS in SPGs can be solved in $\mathcal{O}(n^6)$, where $n = |V(G)| + |V(H)|$.

Proof (Sketch)

- (Extended) 2-MCIS can be solved in $\mathcal{O}(n^6)$.
- Comparison of all blocks at cut vertices in $\mathcal{O}(n^5) \to$ at most $\mathcal{O}(n^2)$ pairs of cut vertices and MWBM is solvable in $\mathcal{O}(n^3)$.
- Total time $\to \mathcal{O}(n^6)$.

Theorem

BBP-MCIS in outerplanar graphs can be solved in $\mathcal{O}(n^5)$.

Proof (Sketch)

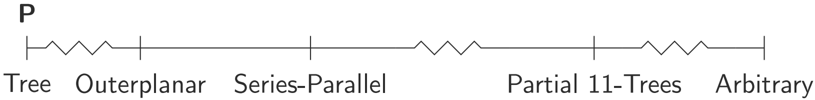
- (Extended) 2-MCIS in outerplanar graphs can be solved in $\mathcal{O}(n^3)$.
- Comparison of all blocks stays expensive.

Conclusion & Future Work

- **NP**-hardness of MCIS in SPGs with one vertex of unbounded degree.
- BBP-MCIS in SPGs can be solved in polynomial time.

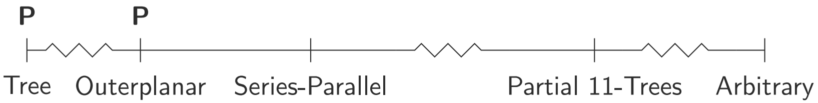
Conclusion & Future Work

- **NP**-hardness of MCIS in SPGs with one vertex of unbounded degree.
- BBP-MCIS in SPGs can be solved in polynomial time.



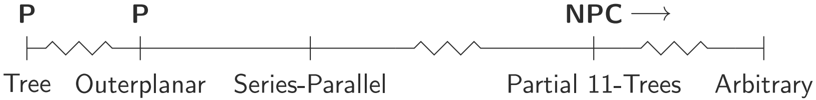
Conclusion & Future Work

- **NP**-hardness of MCIS in SPGs with one vertex of unbounded degree.
- BBP-MCIS in SPGs can be solved in polynomial time.



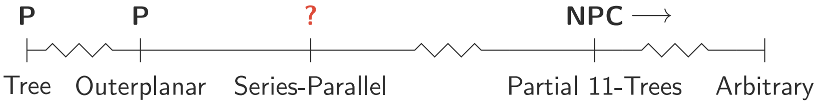
Conclusion & Future Work

- **NP**-hardness of MCIS in SPGs with one vertex of unbounded degree.
- BBP-MCIS in SPGs can be solved in polynomial time.



Conclusion & Future Work

- **NP**-hardness of MCIS in SPGs with one vertex of unbounded degree.
- BBP-MCIS in SPGs can be solved in polynomial time.

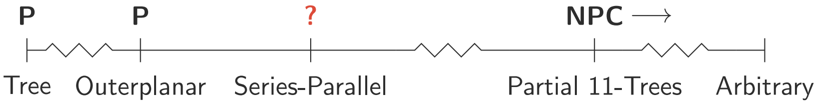


Open Questions

- Complexity of MCS in SPGs with bounded degree?

Conclusion & Future Work

- **NP**-hardness of MCIS in SPGs with one vertex of unbounded degree.
- BBP-MCIS in SPGs can be solved in polynomial time.

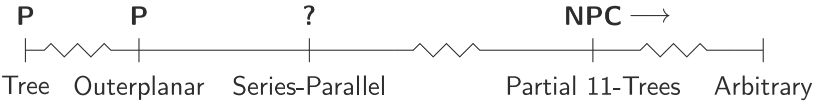


Open Questions

- Complexity of MCS in SPGs with bounded degree?
- *Edge-Disjoint-Path* Problem in SPGs is in **NPC**. [Nishizeki et al., 2001]

Conclusion & Future Work

- **NP**-hardness of MCIS in SPGs with one vertex of unbounded degree.
- BBP-MCIS in SPGs can be solved in polynomial time.

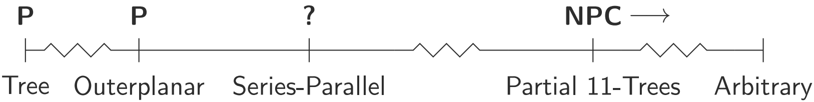


Open Questions

- Complexity of MCS in SPGs with bounded degree?
- *Edge-Disjoint-Path* Problem in SPGs is in **NPC**. [Nishizeki et al., 2001]
- More general: Complexity of MCS in partial k-trees.

Conclusion & Future Work

- **NP**-hardness of MCIS in SPGs with one vertex of unbounded degree.
- BBP-MCIS in SPGs can be solved in polynomial time.



Open Questions

- Complexity of MCS in SPGs with bounded degree?
- *Edge-Disjoint-Path* Problem in SPGs is in **NPC**. [Nishizeki et al., 2001]
- More general: Complexity of MCS in partial k-trees.

Thank You!