Graph width-parameters and algorithms

Jisu Jeong (KAIST)

joint work with Sigve Hortemo Sæther and Jan Arne Telle
(University of Bergen)

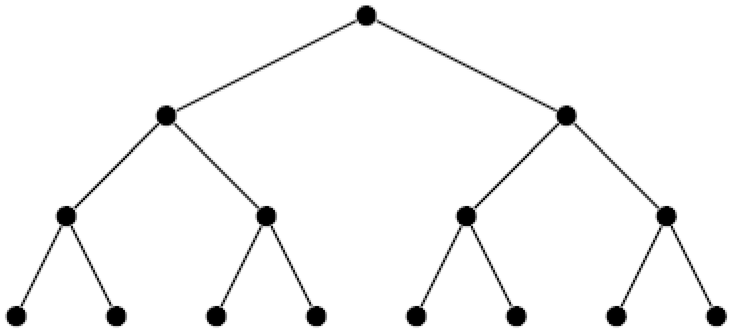
2015 KMS Annual Meeting
2015.10.24. YONSEI UNIVERSITY

Graph width-parameters

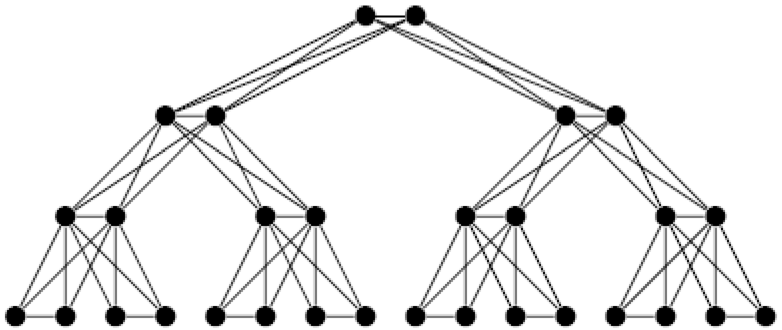
- tree-width (Halin 1976, Robertson and Seymour 1984)
- branch-width (Robertson and Seymour 1991)
- carving-width (Seymour and Thomas 1994)
- clique-width (Courcelle and Olariu 2000)
- rank-width (Oum and Seymour 2006)
- boolean-width (Bui-Xuan, Telle, Vatshelle 2011)
- maximum matching-width (Vatshelle 2012)

Tree-width

- tree-width (Halin 1976, Robertson and Seymour 1984)
  A measure of how "tree-like" the graph is.



tree

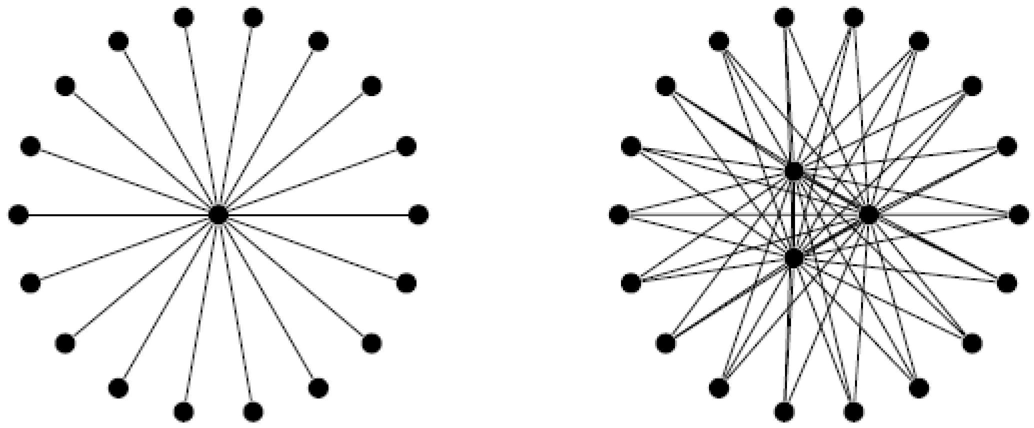


tree-like

Figures from http://fptschool.mimuw.edu.pl/slides/lec6.pdf

Tree-width

- tree-width (Halin 1976, Robertson and Seymour 1984)
  A measure of how "tree-like" the graph is.

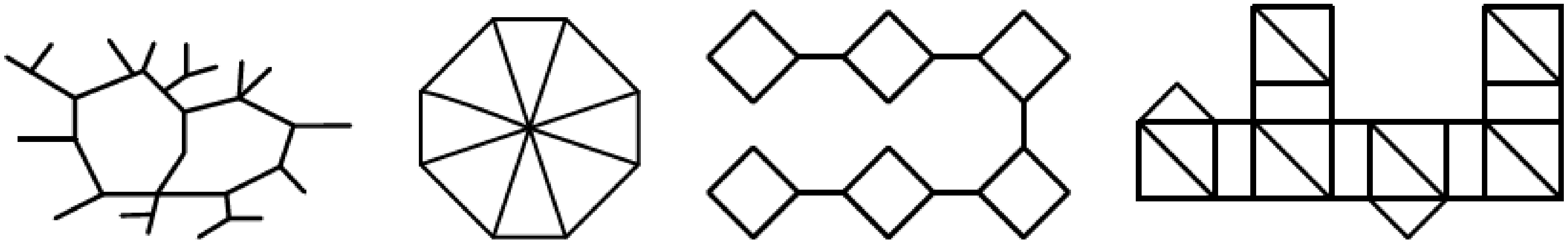


tree　　　　tree-like

Figures from http://fptschool.mimuw.edu.pl/slides/lec6.pdf

Tree-width

- tree-width (Halin 1976, Robertson and Seymour 1984)
  A measure of how “tree-like” the graph is.

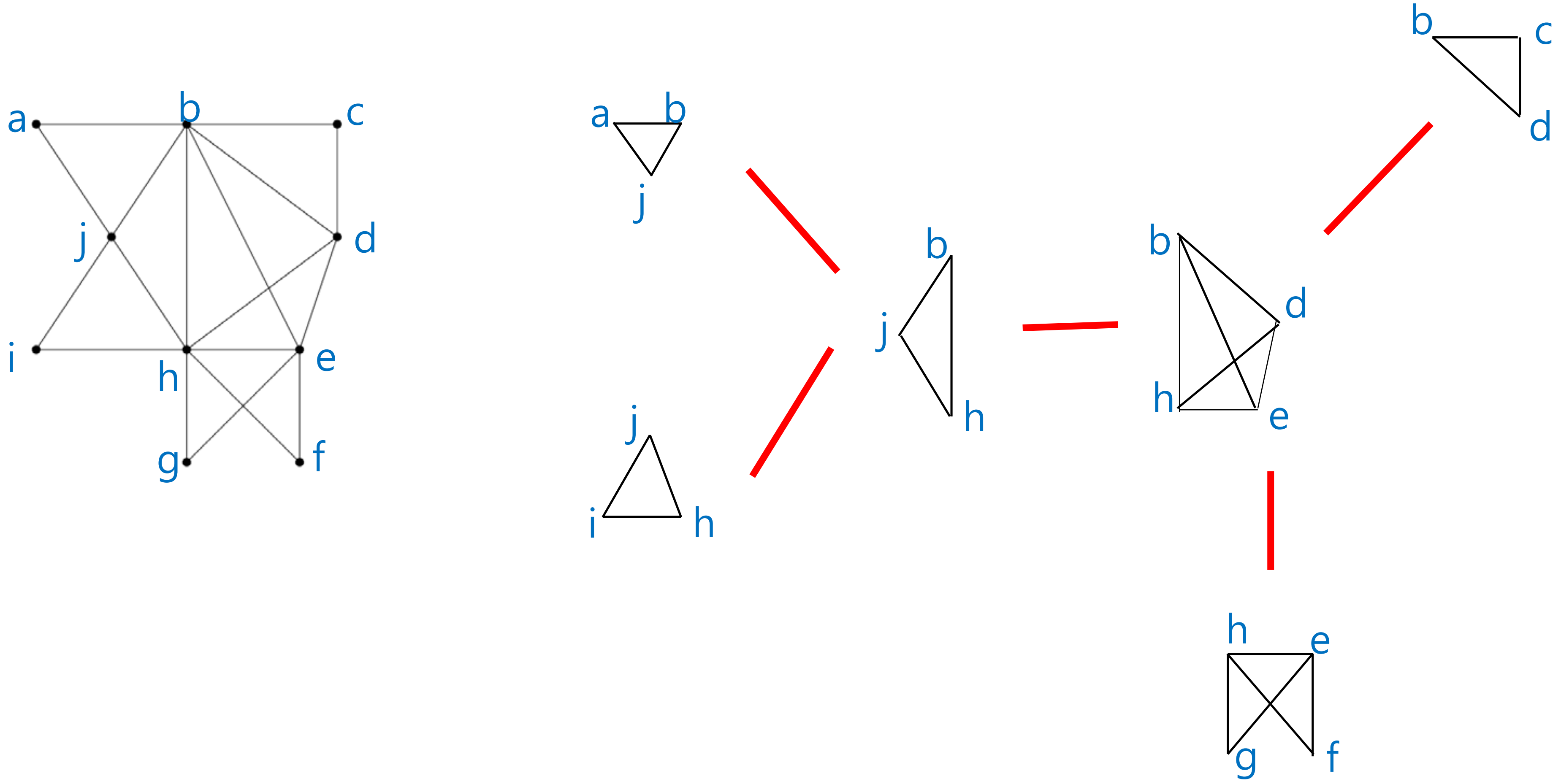


bad bad good good

Figures from http://fptschool.mimuw.edu.pl/slides/lec6.pdf

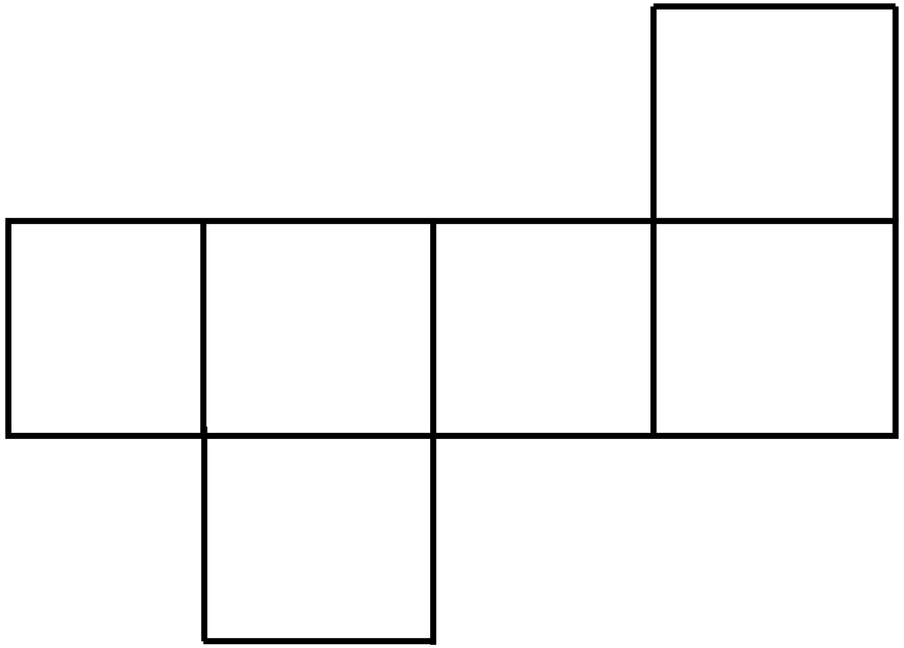
A *tree-decomposition* of a graph G is a pair $(T, \{X_t\}_{t \in V(T)})$ consisting of a tree T and a family $\{X_t\}_{t \in V(T)}$ of subsets X_t of $V(G)$, called *bags*, satisfying the following three conditions:

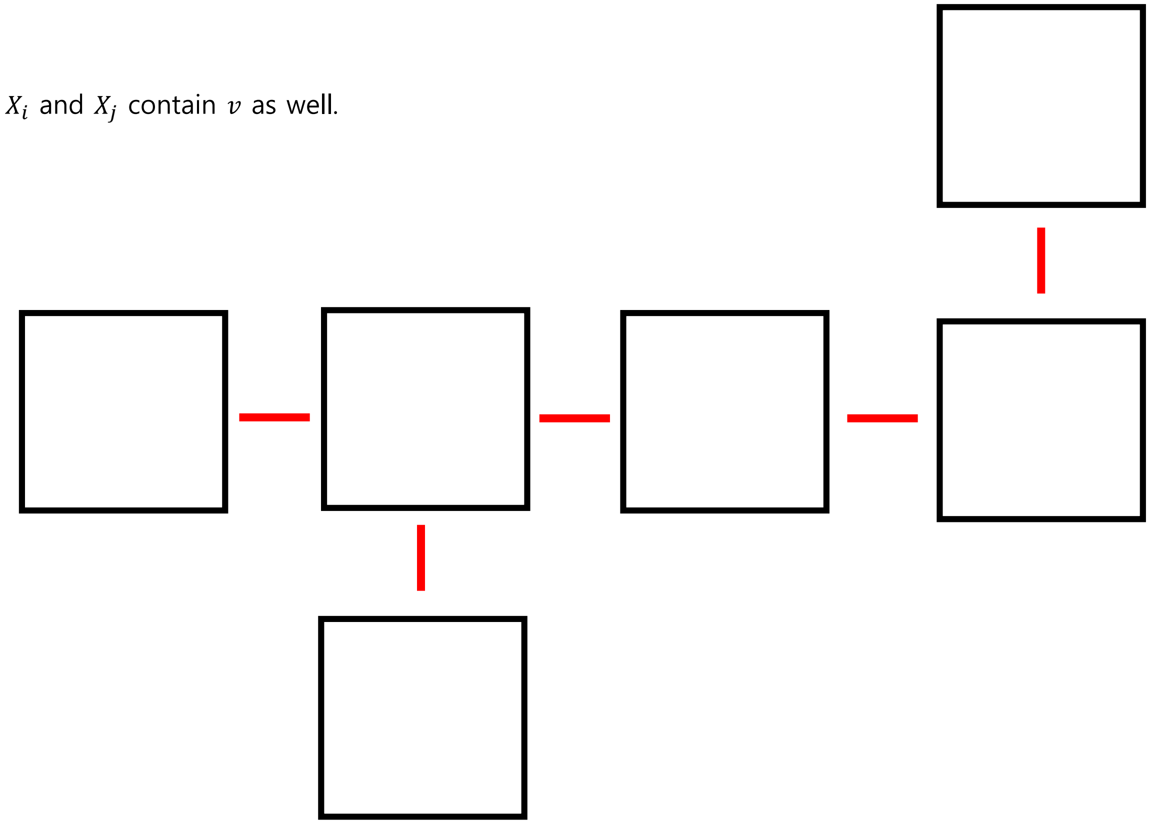
1. each vertex of G is in at least one bag,
2. for each edge uv of G, there exists a bag that contains both u and v,
3. if X_i and X_j both contain a vertex v, then all bags X_k in the path between X_i and X_j contain v as well.



A *tree-decomposition* of a graph G is a pair $(T, \{X_t\}_{t\in V(T)})$ consisting of a tree T and a family $\{X_t\}_{t\in V(T)}$ of subsets X_t of $V(G)$, called *bags*, satisfying the following three conditions:

1. each vertex of G is in at least one bag,
2. for each edge uv of G, there exists a bag that contains both u and v,
3. if X_i and X_j both contain a vertex v, then all bags X_k in the path between X_i and X_j contain v as well.



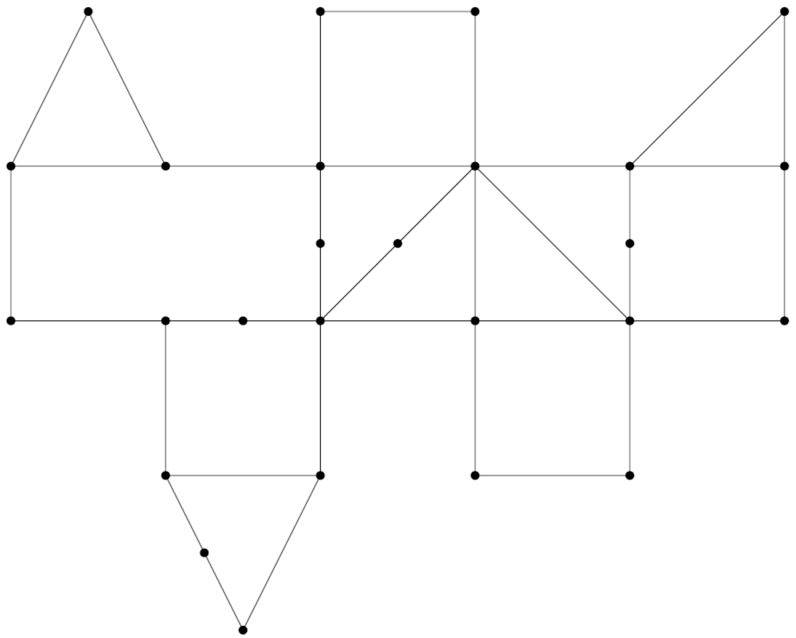


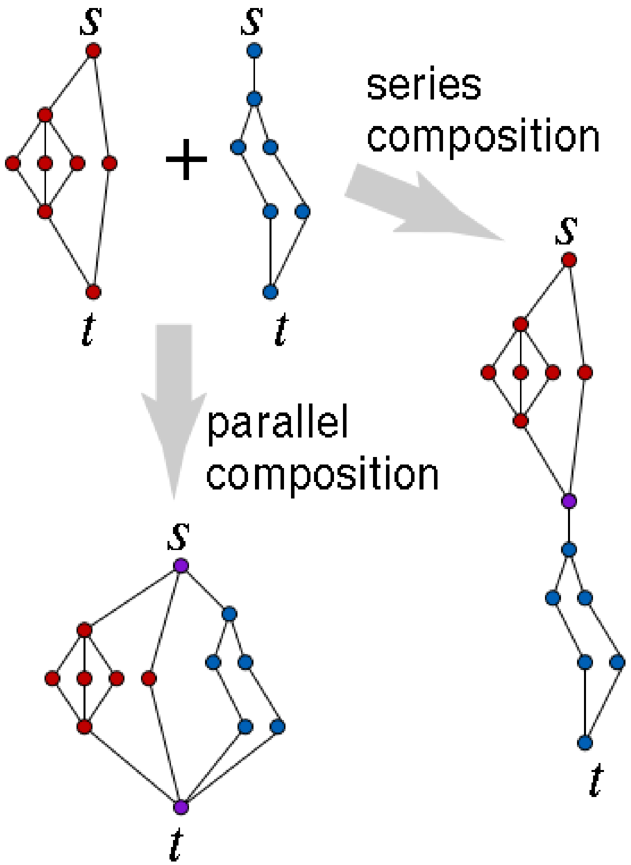
The *width* of a tree-decomposition $(T, \{X_t\}_{t\in V(T)})$ is $\max|X_t| - 1$.

The *tree-width* of a graph G, denoted by $tw(G)$, is the minimum width over all possible tree-decompositions of G.

Examples

- tree-width $\leq 1 \Leftrightarrow$ a forest $\Leftrightarrow$ no cycle
- tree-width $\leq 2 \Leftrightarrow$ a series-parallel graph $\Leftrightarrow$ no K_4 minor

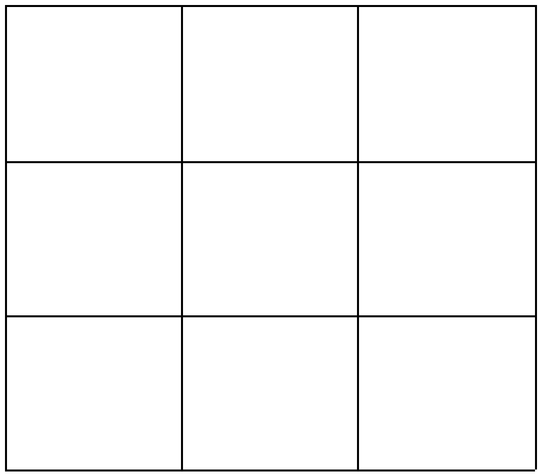




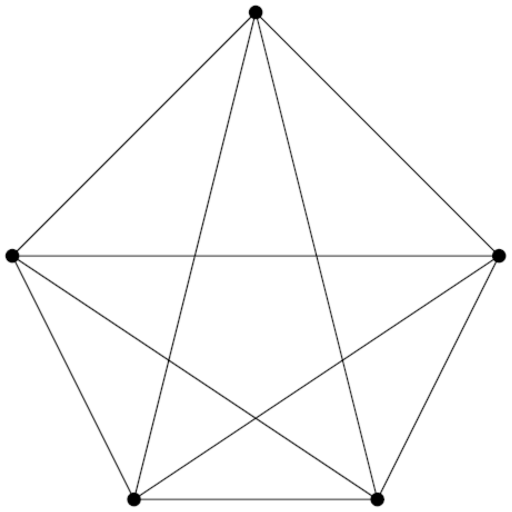
Figures from wikipedia

Examples

- tree-width $\leq 1 \Leftrightarrow$ a forest $\Leftrightarrow$ no cycle
- tree-width $\leq 2 \Leftrightarrow$ a series-parallel graph
  $\Leftrightarrow$ no K_4 minor
- The tree-width of a $k \times k$ grid is k.
- The tree-width of K_n is $n - 1$.



4×4 grid



K_5

Algorithm using tree-decomposition

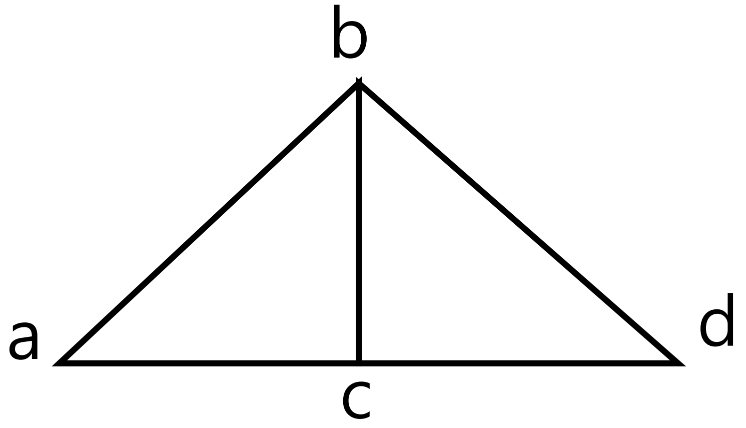
Exercise

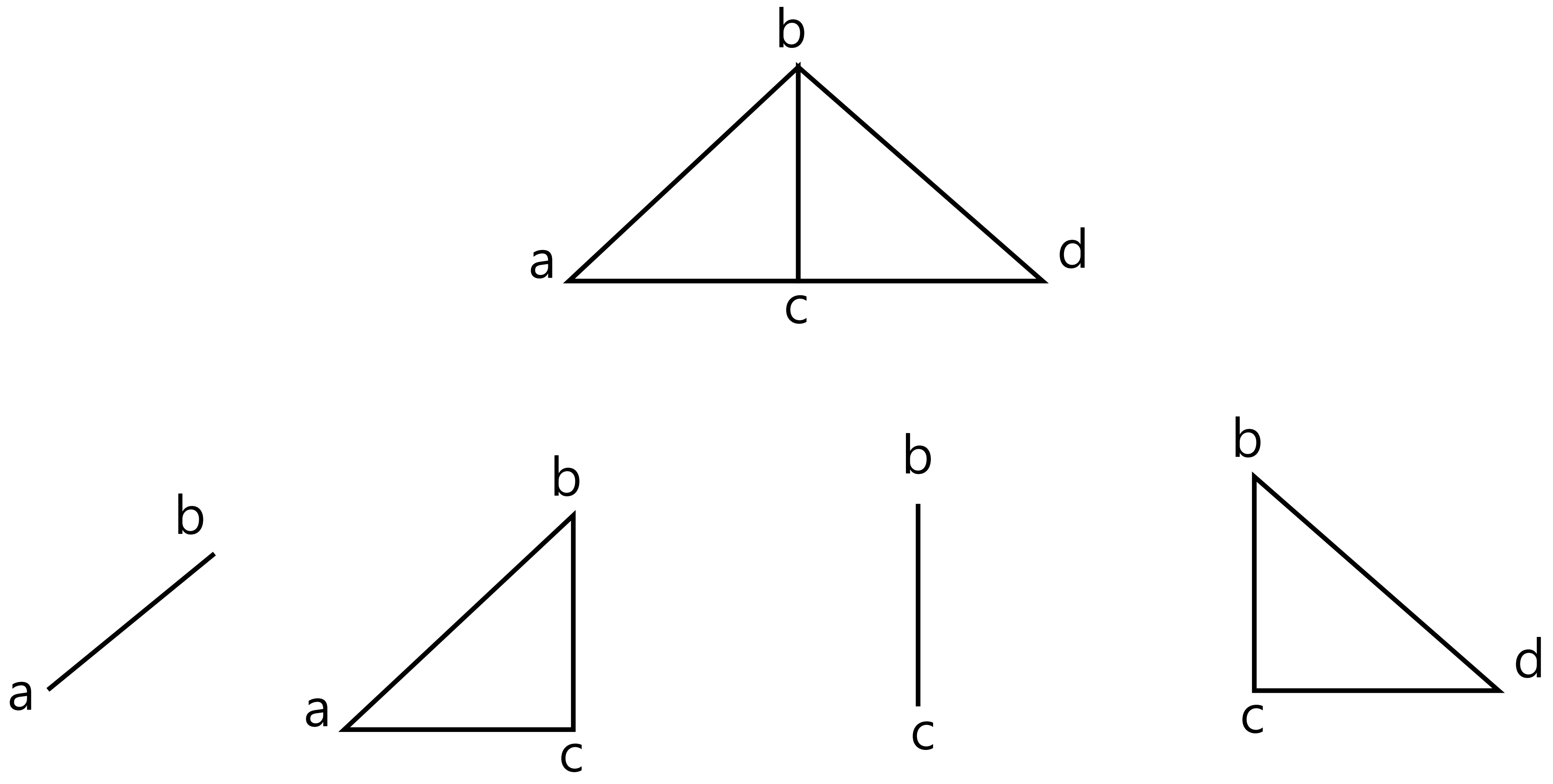
Given a tree-decomposition of width t of a graph G, 3-COLORABILITY can be solved in time $O(t3^t n)$.

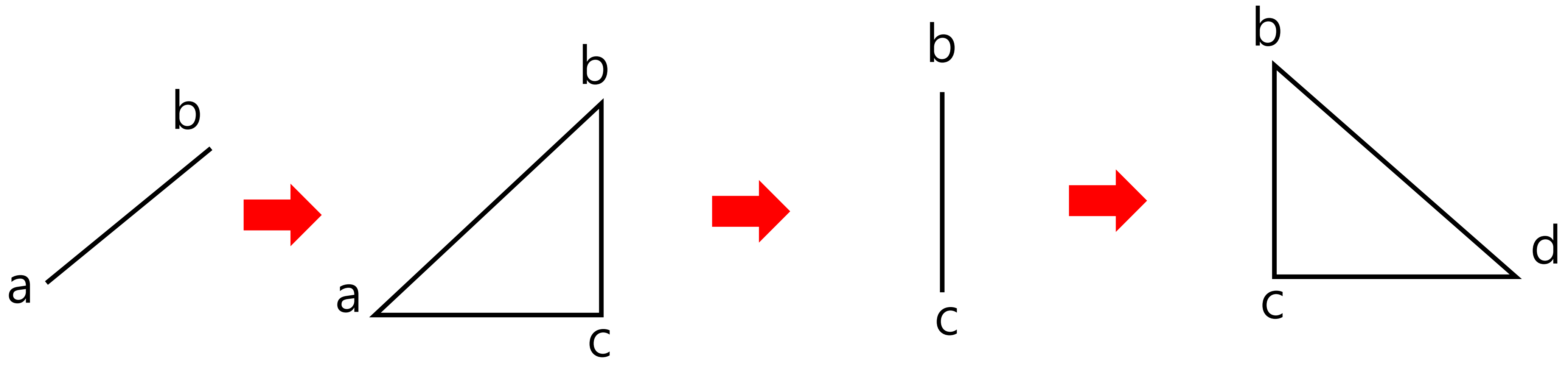
Algorithm using tree-decomposition

Exercise

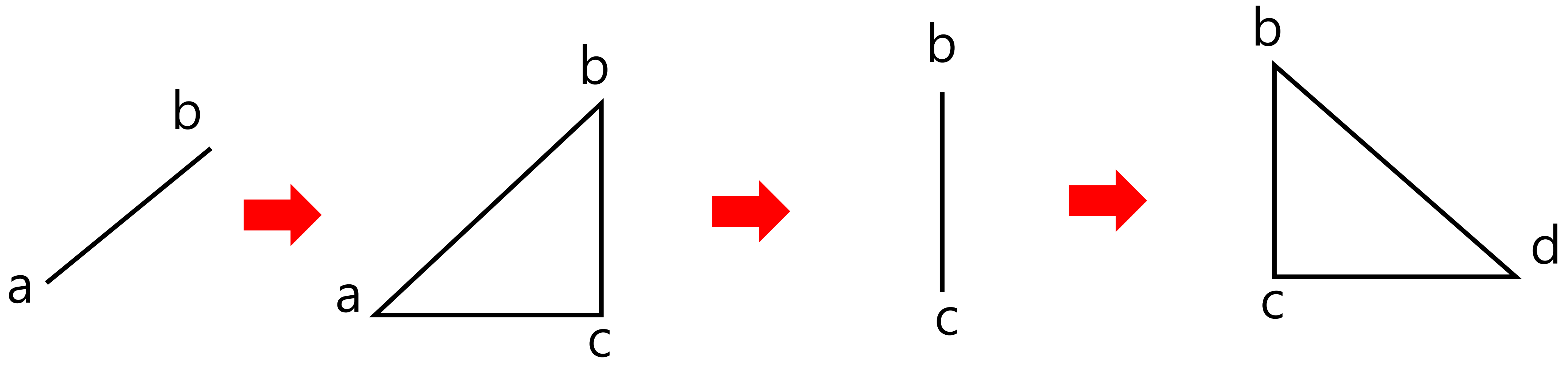
Given a tree-decomposition of width t of a graph G,
3-COLORABILITY can be solved in time $O(t3^t n)$.




b
a
d
c
b
a
b
a
c
b
c
b
d
c

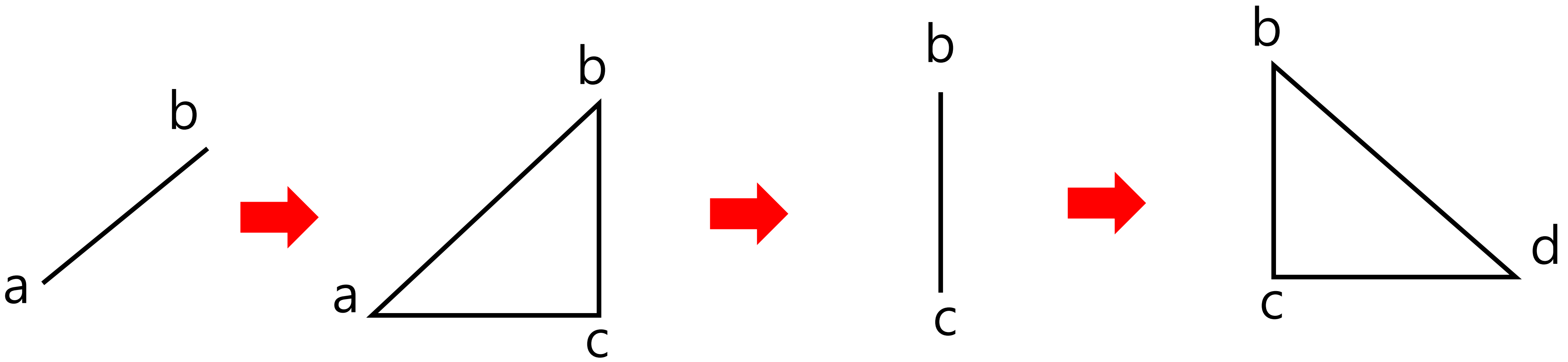


a	b
1	2
1	3
2	1
2	3
3	1
3	2

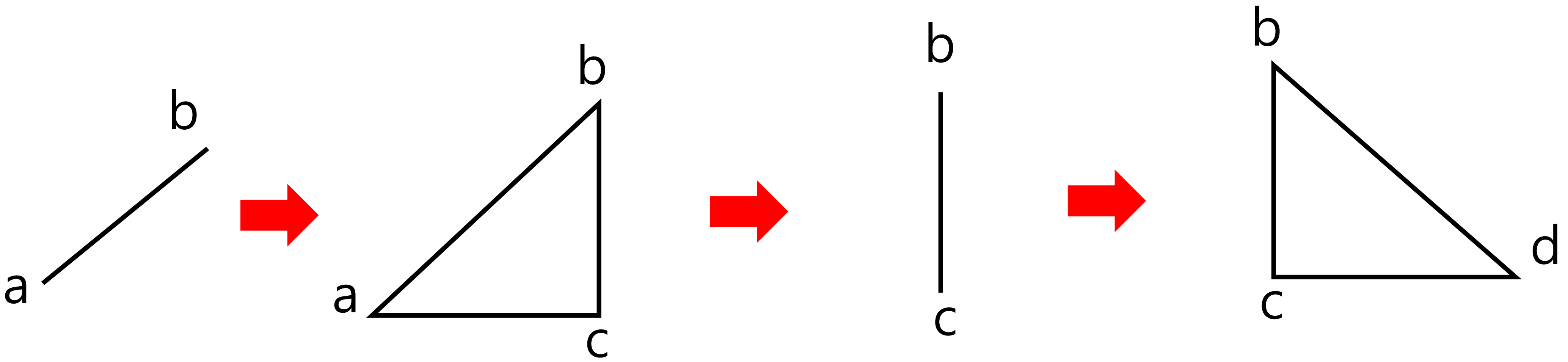


a	b
1	2
1	3
2	1
2	3
3	1
3	2

a	b	c
1	2	3
1	3	2
2	1	3
2	3	1
3	1	2
3	2	1

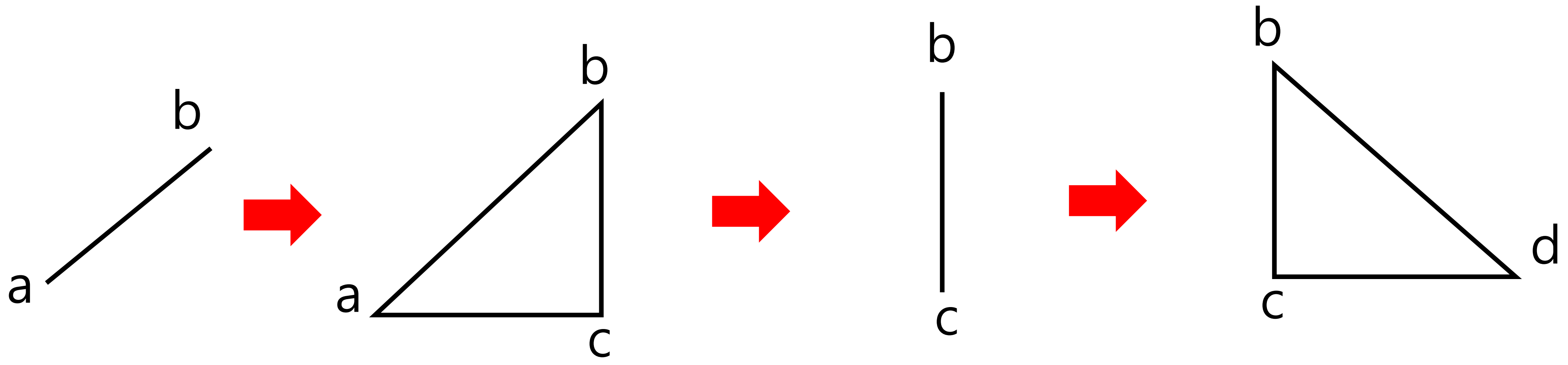


a	b	c
1	2	3
1	3	2
2	1	3
2	3	1
3	1	2
3	2	1



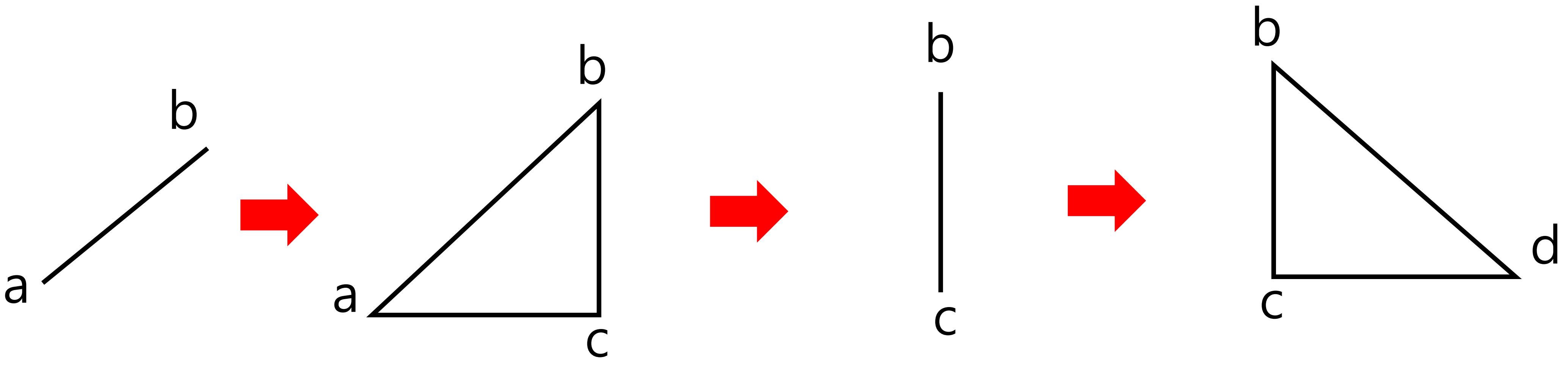
a	b	c
1	2	3
1	3	2
2	1	3
2	3	1
3	1	2
3	2	1

b	c
2	3
3	2
1	3
3	1
1	2
2	1



b	c
2	3
3	2
1	3
3	1
1	2
2	1

b	c	d
2	3	1
3	2	1
1	3	2
3	1	2
1	2	3
2	1	3

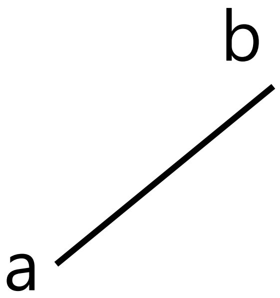


a	b
1	2
1	3
2	1
2	3
3	1
3	2

a	b	c
1	2	3
1	3	2
2	1	3
2	3	1
3	1	2
3	2	1

b	c
2	3
3	2
1	3
3	1
1	2
2	1

b	c	d
2	3	1
3	2	1
1	3	2
3	1	2
1	2	3
2	1	3



The number of columns of a table is at most $t+1$.

a	b
1	2
1	3
2	1
2	3
3	1
3	2

The number of rows of a table is at most 3^{t+1}.

2	1	3

1	3

1	3	2

The number of tables is at most $O(n)$.

Algorithm using tree-decomposition

Easy exercise

Given a tree-decomposition of width t of a graph G,
3-COLORABILITY can be solved in time $O(t3^t n)$.

Difficult exercise

Given a tree-decomposition of width t of a graph G,
Minimum Dominating Set Problem can be solved in time $O(4^t n)$.

Algorithm using tree-decomposition

Difficult exercise

Given a tree-decomposition of width t of a graph G,
Minimum Dominating Set Problem can be solved in time $O(4^t n)$.

- In dominating set D
- Dominated by D
- Not in D, but do not have to be dominated by D (will be dominated later)

- TRUE / FALSE → the size of D
(if D is not a dominating set, then ∞)

Algorithm using tree-decomposition

Theorem (van Rooij, Bodlaender, Rossmanith 2009)

Minimum Dominating Set Problem can be solved in time $O(3^t n)$ when a graph and its tree-decomposition of width t is given.

Theorem (Lokshtanov, Marx, Saurabh 2011)

Minimum Dominating Set Problem can**not** be solved in time $O((3-\varepsilon)^t n)$ where t is the tree-width of the given graph.

New width-parameter

Maximum matching width (mmw)

Theorem (Vatshelle 2012)

For every graph G,

$$mmw(G) \leq tw(G) + 1 \leq 3\, mmw(G).$$

A graph G has bounded tree-width if and only if G has bounded mm-width.

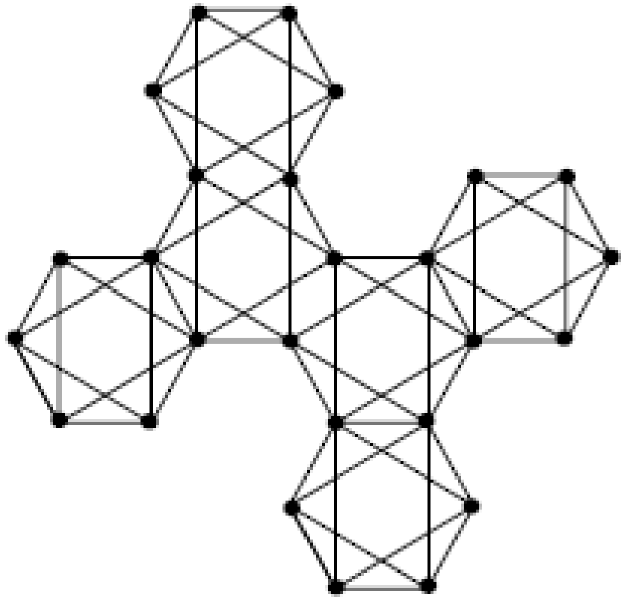
Algorithm using mmw

Theorem (J., Sæther, Telle 2015)

Minimum Dominating Set Problem can be solved in time $O(8^m n)$ when a graph and its mm-decomposition of mm-width m is given.

Using tree-width: $O(3^t n)$

Using mm-width: $O(8^m n)$



Our algorithm is faster when $8^m < 3^t$, that is,

$$1.893\, mmw(G) < tw(G).$$

Note that for every graph G,

$$mmw(G) \leq tw(G) + 1 \leq 3\, mmw(G).$$

What if **only a graph** is an input?

Theorem (Oum, Seymour 2006)

Given a graph G, a branch-decomposition over $V(G)$ of mm-width at most $3mmw(G)+1$ can be found in time $O^*\left(2^{3mmw(G)}\right)$.

Runtime : $O^*(8^m) = O^*(8^{3mmw(G)})$

Theorem (Amir 2010)

Given a graph G, a tree-decomposition over $V(G)$ of width at most $3.67tw(G)$ can be found in time $O^*\left(2^{3.7tw(G)}\right)$.

Runtime : $O^*(3^t) = O^*(3^{3.67tw(G)})$

What if **only a graph** is an input?

Theorem (Oum, Seymour 2006)

Given a graph G, a branch-decomposition over $V(G)$ of mm-width at most $3mmw(G)+1$ can be

Runtime : $O^*(8^m) = O^*(8^{3mmw(G)})$

Theorem (Amir 2010)

Given a graph G, a tree-decompositi
most $3.67tw(G)$ can be found in time $O^*\left(2^{3.7tw(G)}\right)$.

Runtime : $O^*(3^t) = O^*(3^{3.67tw(G)})$

Our algorithm is faster if an input graph G satisfies $1.55\, mmw(G) < tw(G)$.

Open questions

Theorem (J., Sæther, Telle 2015)

Minimum Dominating Set Problem can be solved in time $O(8^m n)$ when a graph and its mm-decomposition of mm-width m is given.

- Improve $O(8^m n)$ or show that it is tight
- Other problems
- Other width-parameters

Theorem (van Rooij, Bodlaender, Rossmanith 2009)

Minimum Dominating Set Problem can be solved in time $O(3^t n)$ when a graph and its tree-decomposition of width t is given.

Theorem (J., Sæther, Telle 2015)

Minimum Dominating Set Problem can be solved in time $O(8^m n)$ when a graph and its mm-decomposition of mm-width m is given.

Our algorithm is faster when $8^m < 3^t$, that is,

$$1.893\, mmw(G) < tw(G).$$

Thank you

New characterization

For any $k \geq 2$, a graph G on vertices $v_1, v_2, \ldots, v_n$ has tree-width (mm-width, branch-width) at most k *if and only if* there are subtrees $T_1, T_2, \ldots, T_n$ of a tree T where all internal vertices have degree 3

such that 1) if $v_i v_j \in E(G)$, then T_i and T_j have at least one vertex (vertex, edge) of T in common,

2) for each vertex (edge, edge) of T, there are at most $k-1$ (at most k, at most k) subtrees containing it.

Paul, Telle, Edge-maximal graphs of branchwidth k: the k-branches. 2009