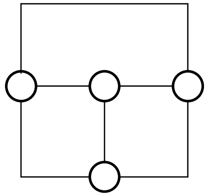
Orthogonal Drawings of Graphs and Their Relatives

Part 2 – Orthogonal drawings in the variable embedding setting

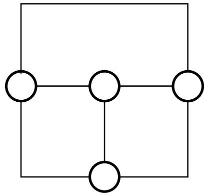
Walter Didimo
University of Perugia
walter.didimo@unipg.it

Summary

- The SQPR-tree data structure
- Bend-minimization of planar 3-graphs
  - Efficient algorithms
- Bend-minimization of planar 4-graphs
  - Exponential-time approaches



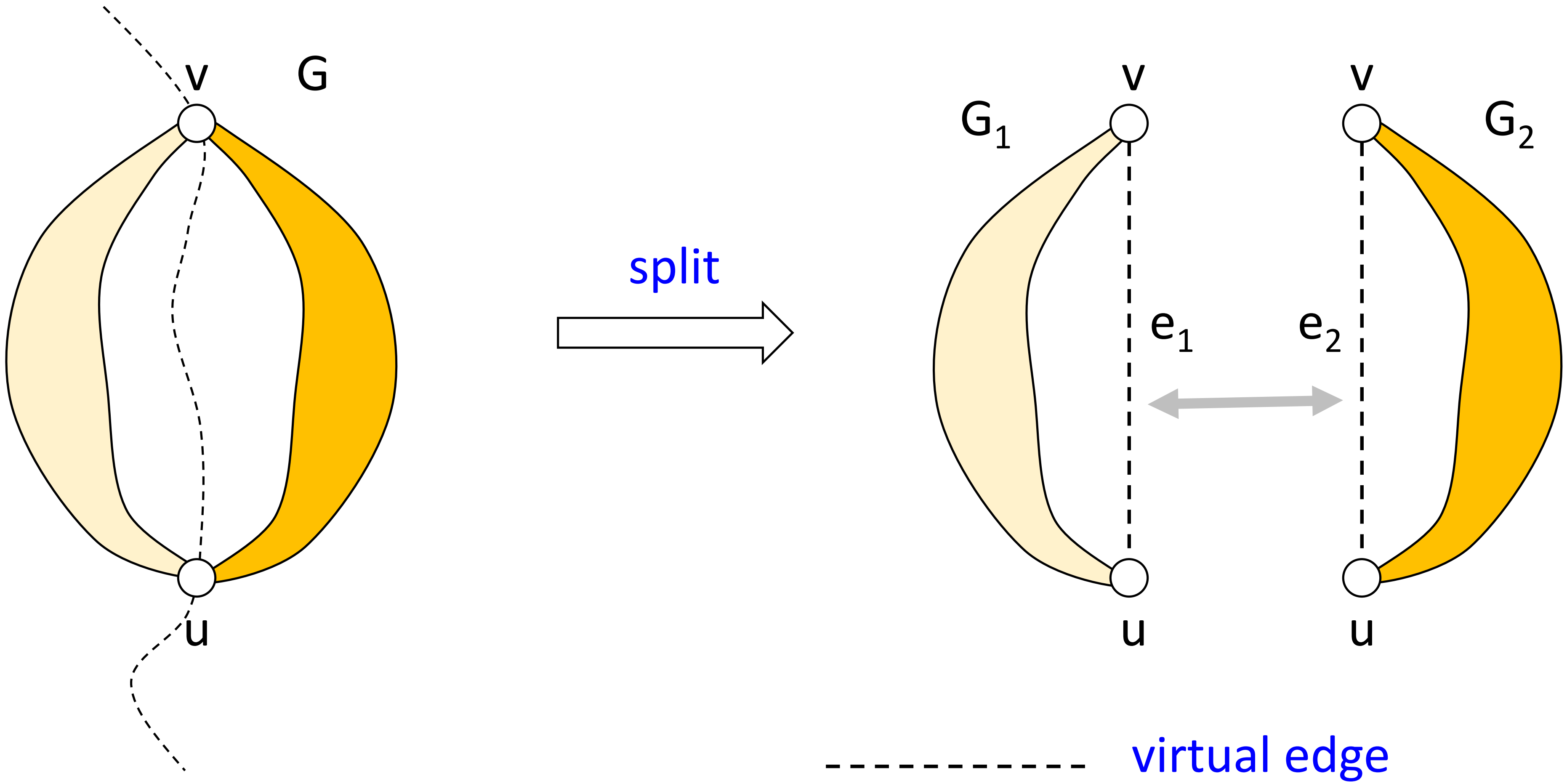
SPQR-trees



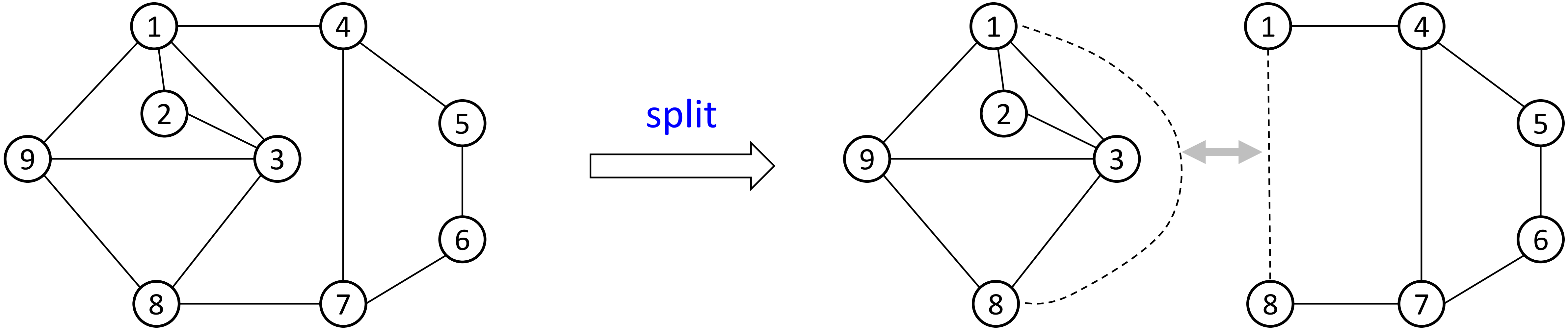
Triconnected components and SPQR-trees

- A biconnected graph can be decomposed into triconnected components
  - *J. E. Hopcroft, R. E. Tarjan*: Dividing a Graph into Triconnected Components. SIAM J. Comput. 2(3): 135-158 (1973)
- If G is a planar graph, the planar embeddings of G depend on the planar embeddings of its triconnected components
  - the SPQR-tree data structure provides an implicit representation of the triconnected components of G and of all planar embeddings of G [*G. Di Battista, R. Tamassia*: On-Line Planarity Testing. SIAM J. Comput. 25(5): 956-997 (1996)]

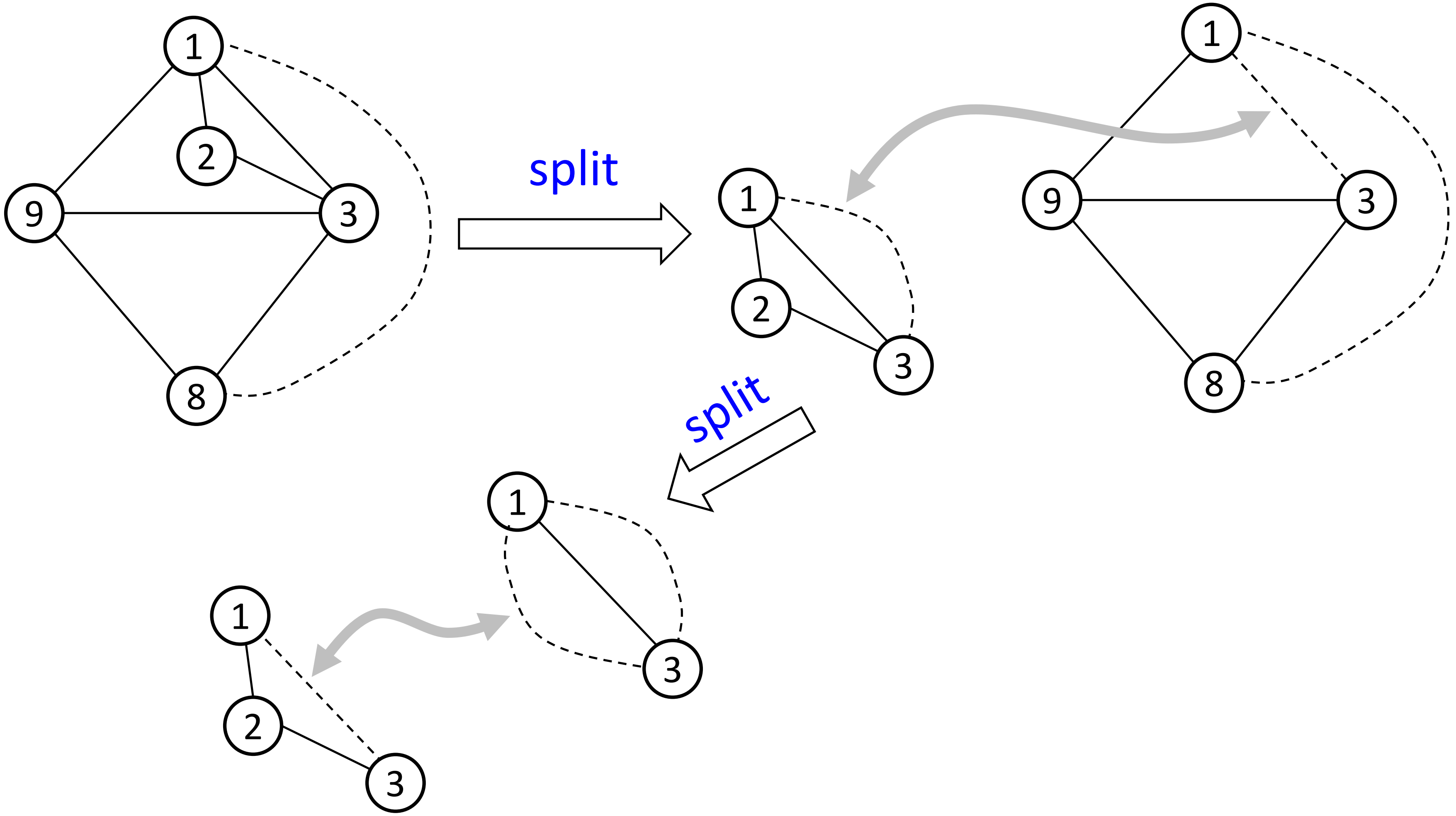
Separation pair and split operation



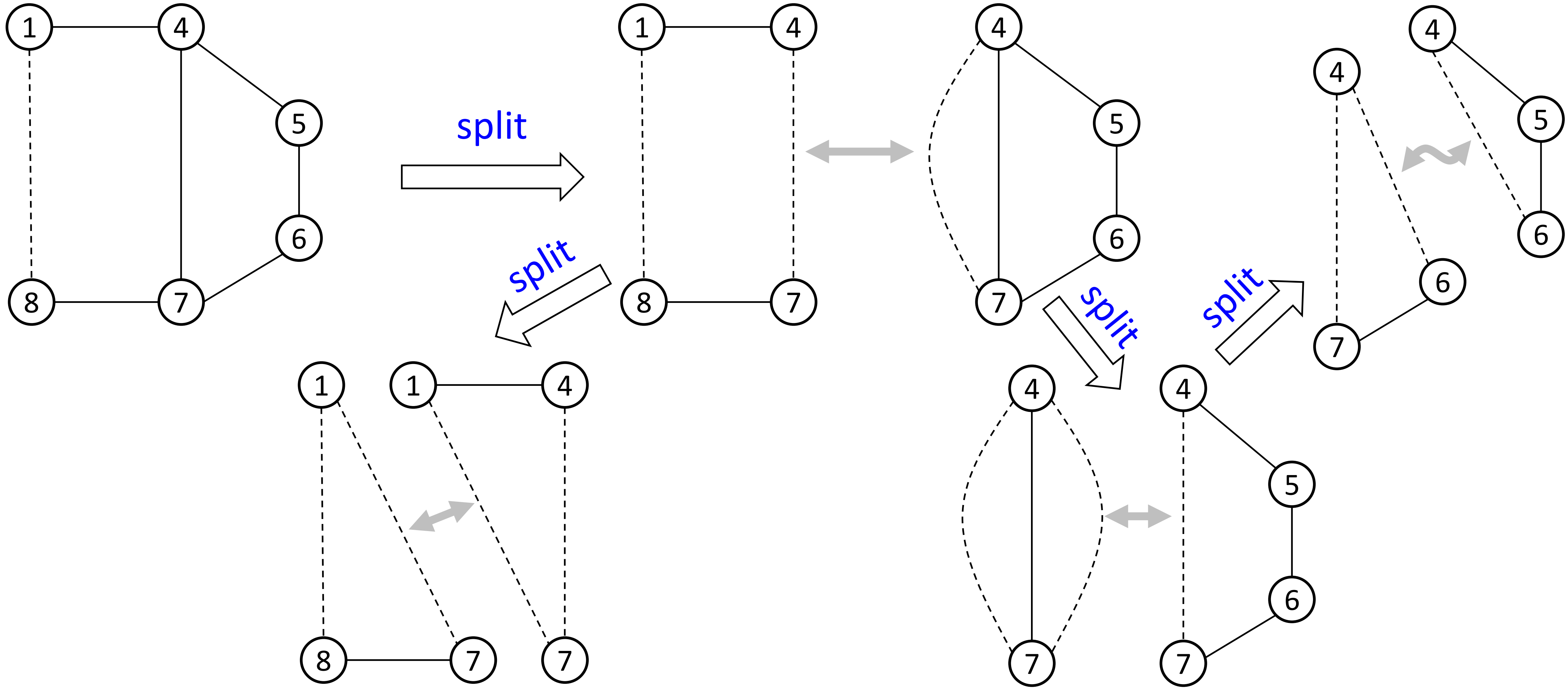
Recursive split operation



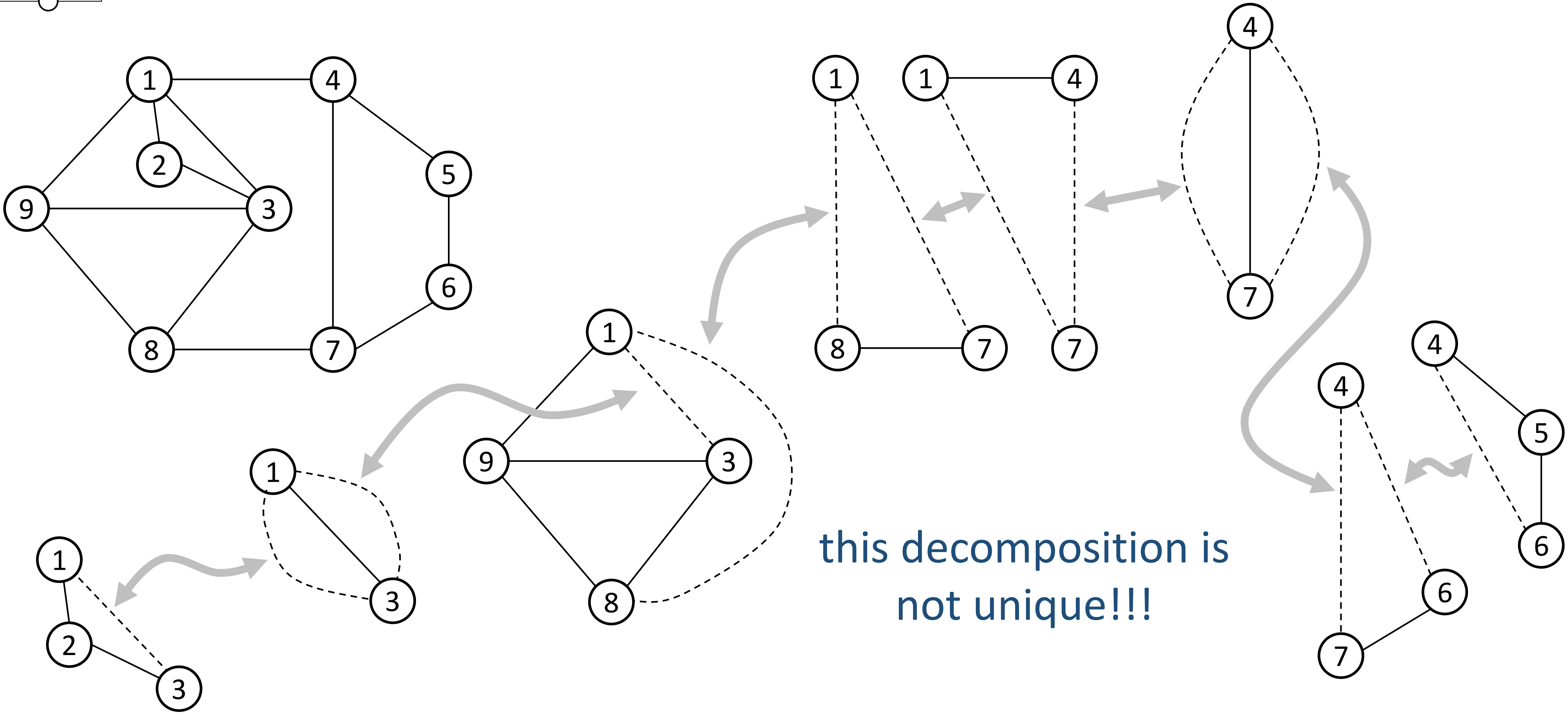
Recursive split operation



Recursive split operation

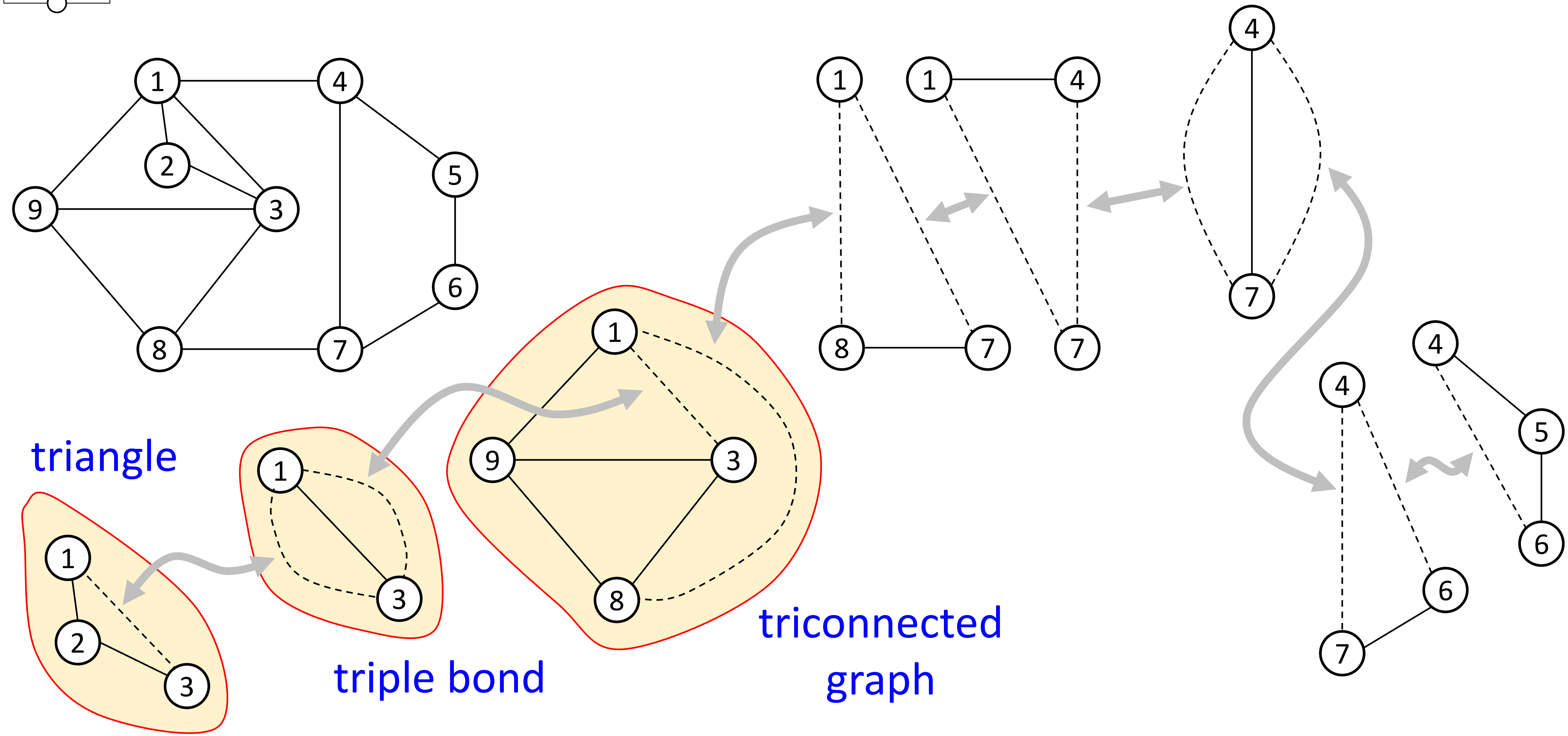


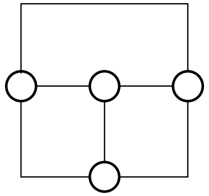
Recursive split operation - output



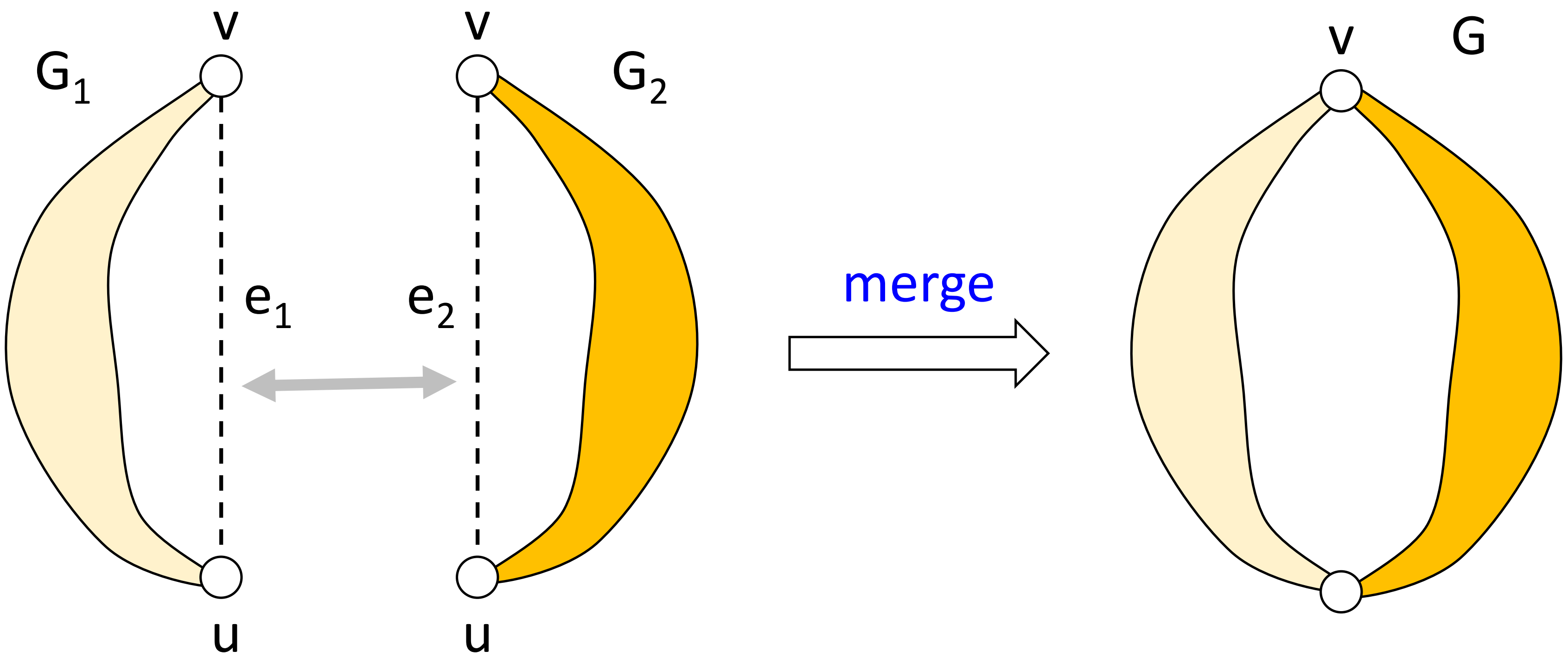
this decomposition is not unique!!!

Recursive split operation - output



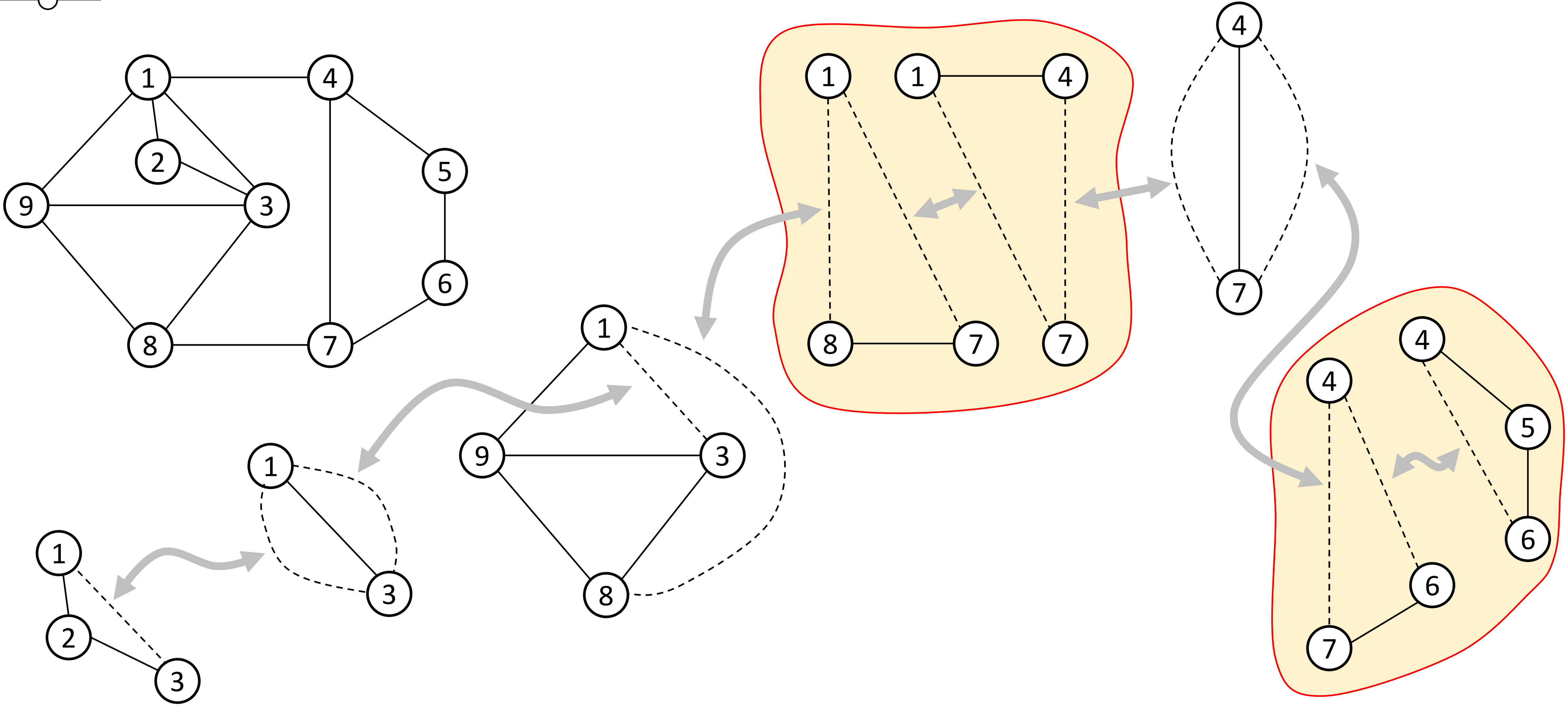


Merge operation

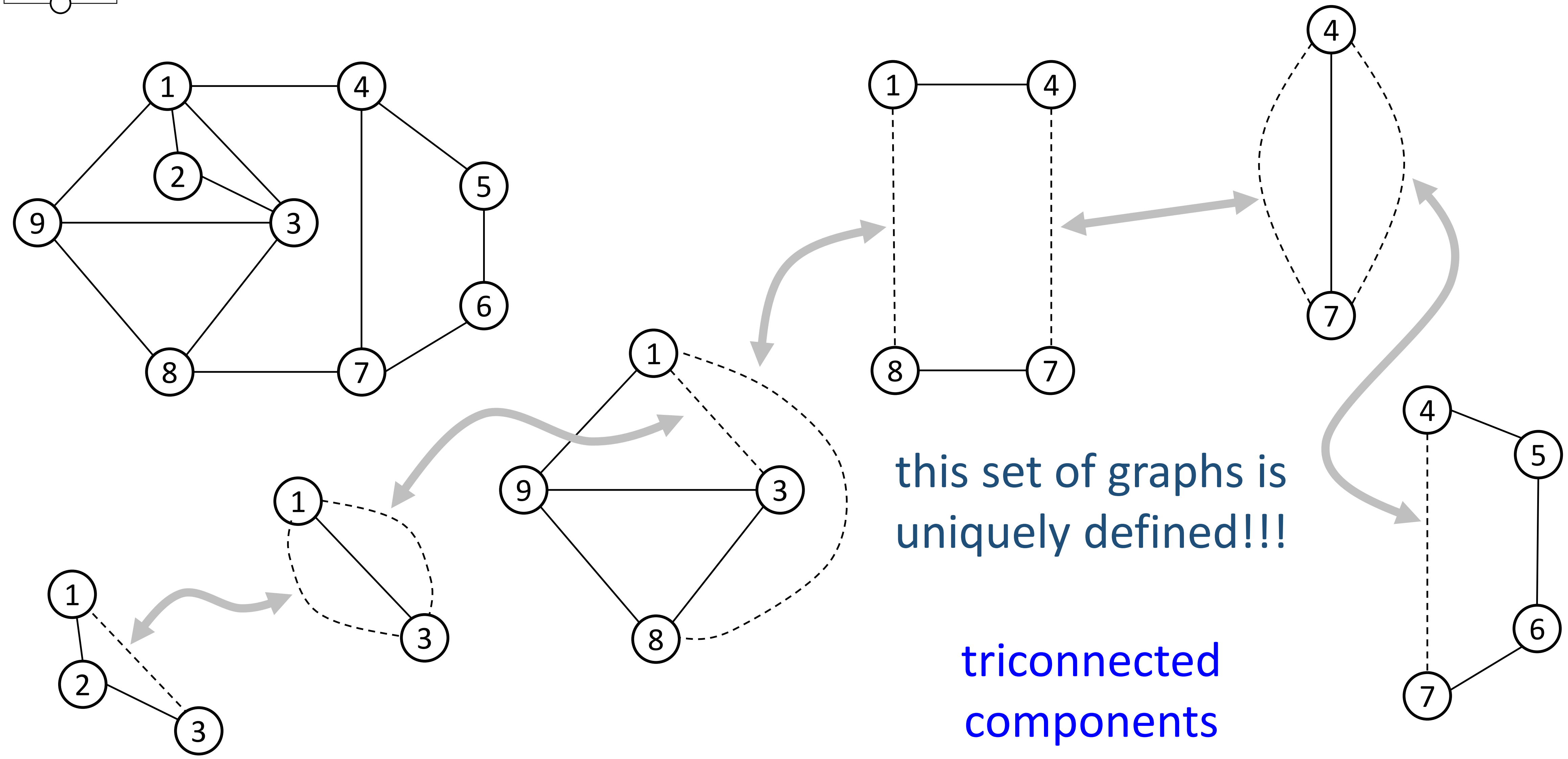


- If each G_i is a triple bond or (more in general) consists of a set of parallel edges only
- If each G_i is a triangle or (more in general) a simple cycle

Recursive merge operation



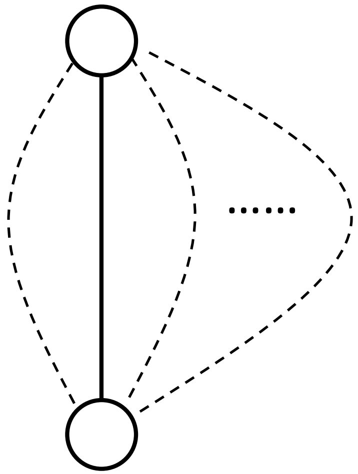
Recursive merge operation – final result



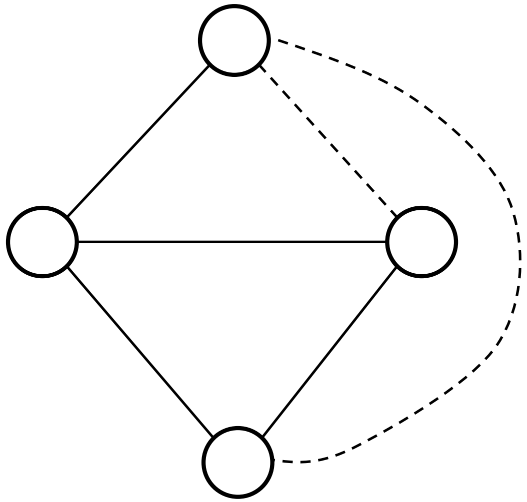
this set of graphs is uniquely defined!!!

triconnected components

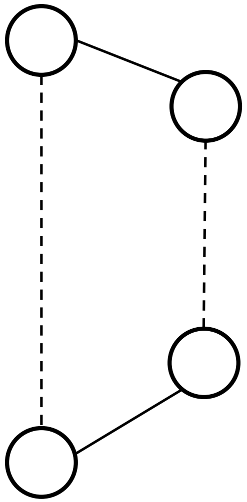
Triconnected components



parallel component

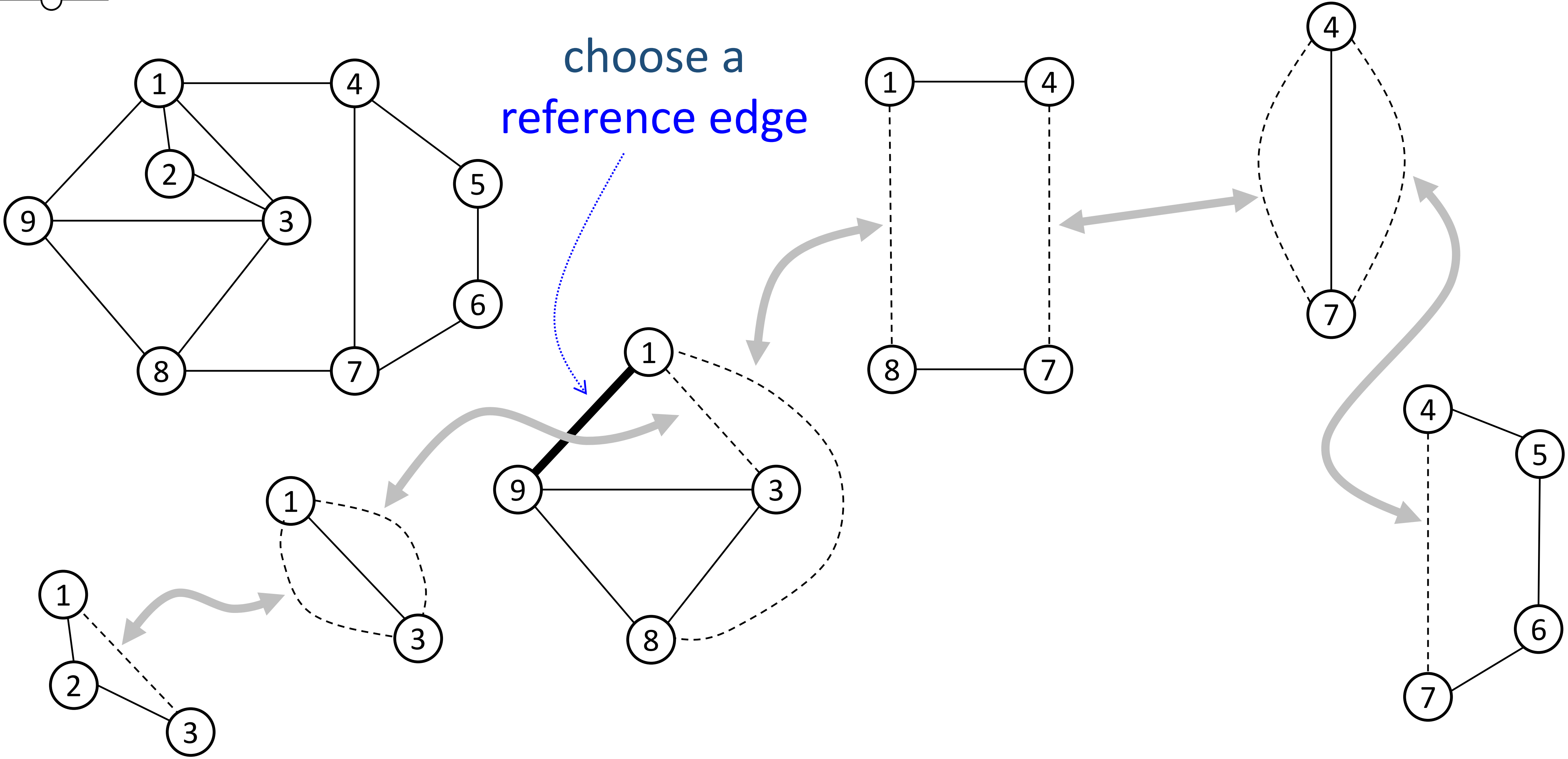


rigid component

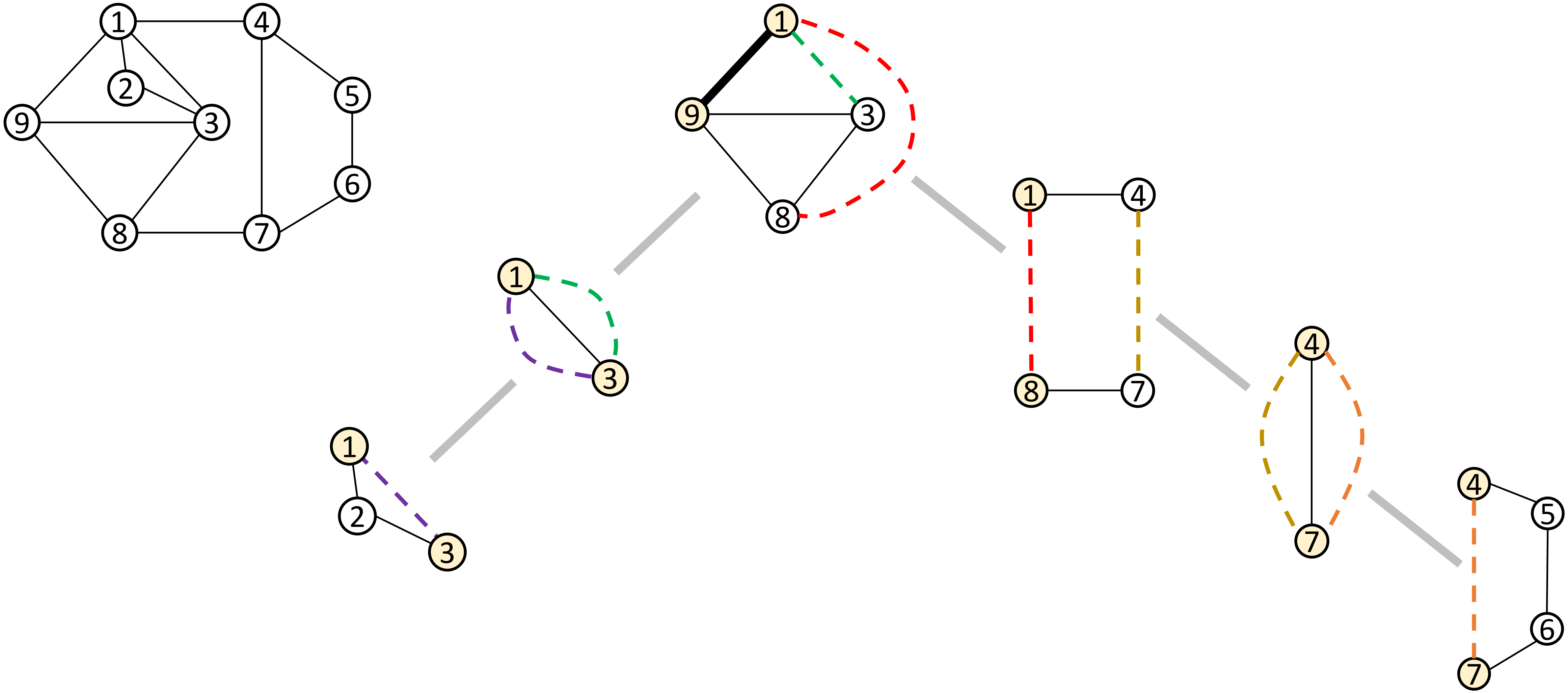


series component

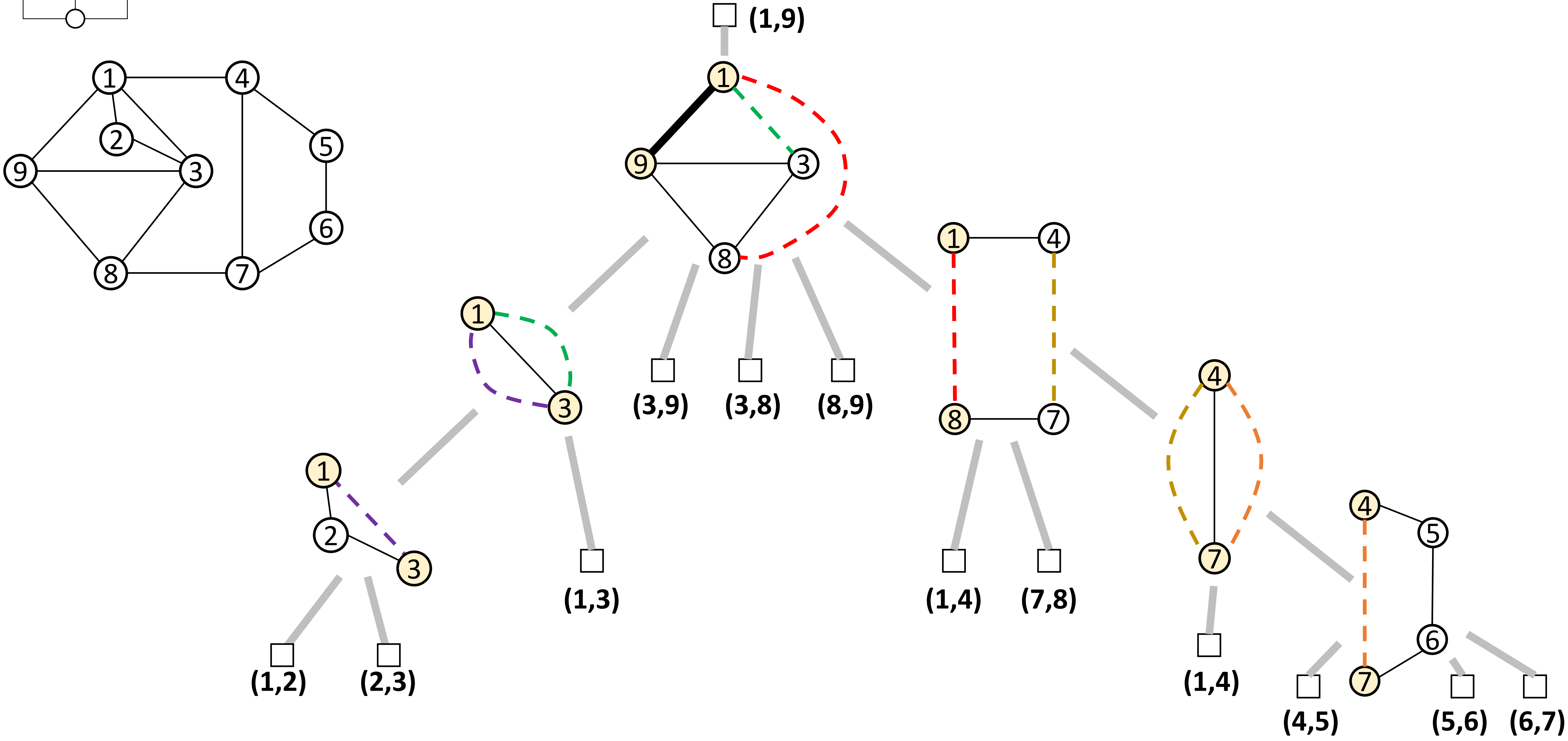
Towards SPQR-trees



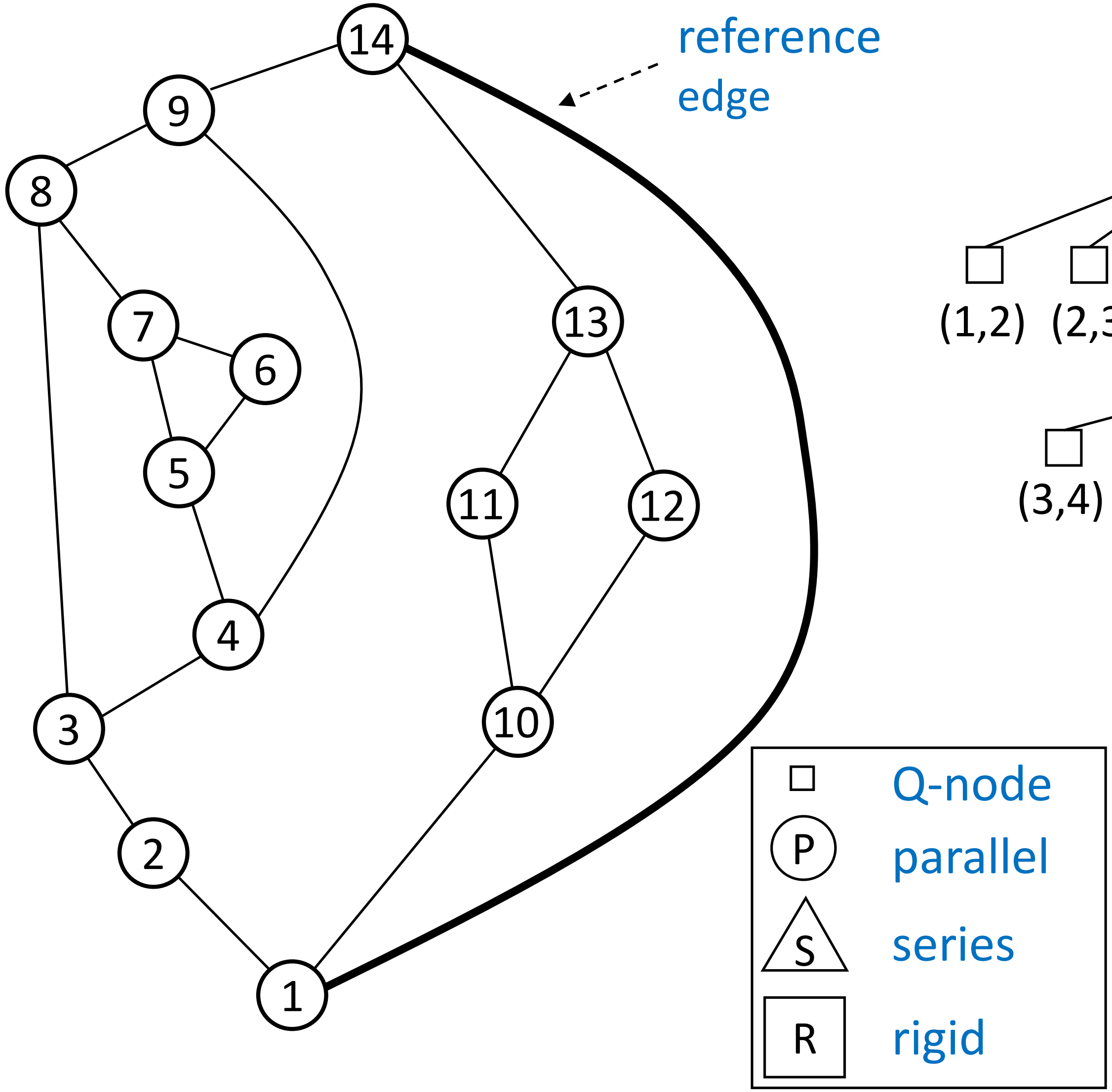
Towards SPQR-trees

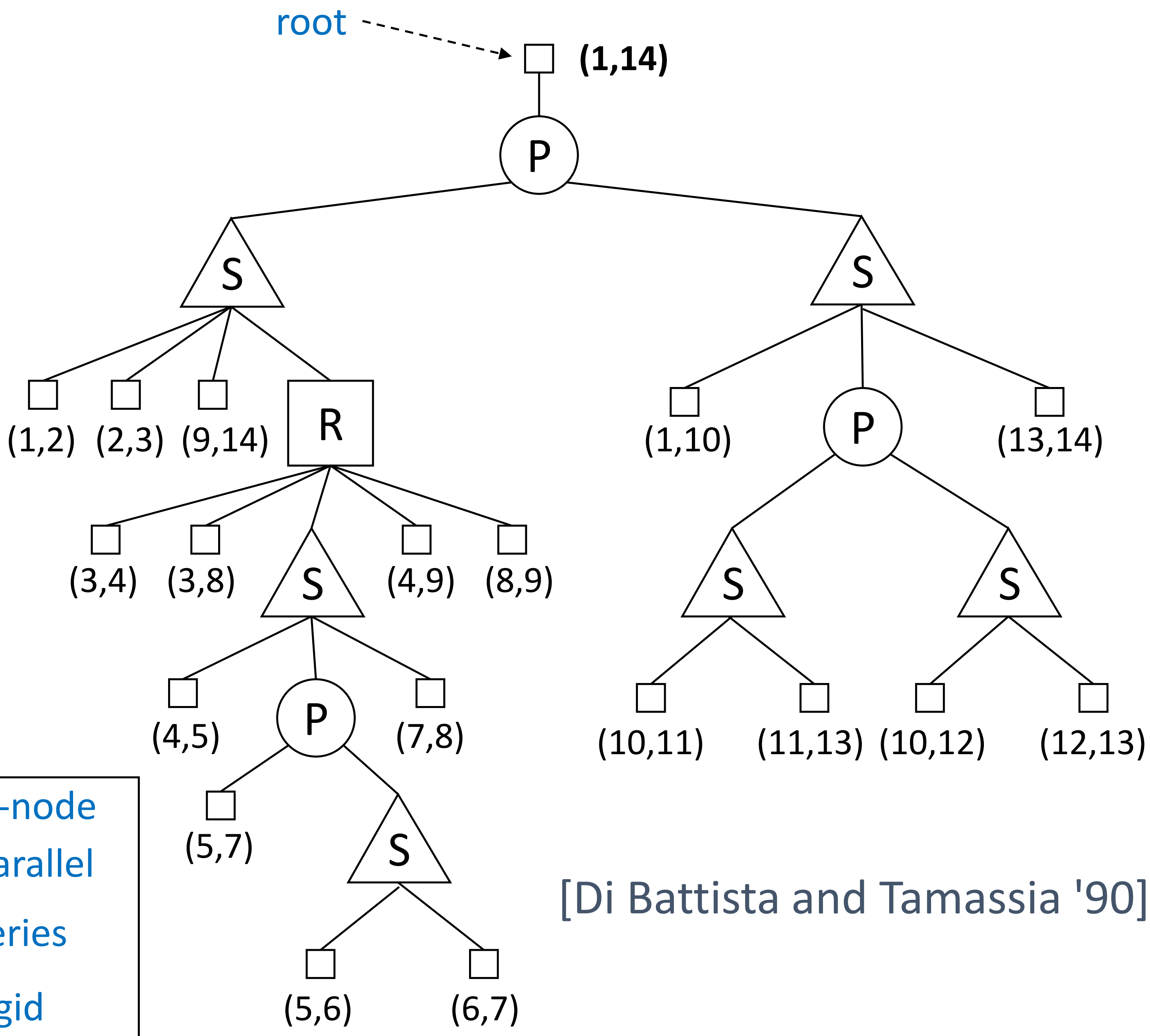


Towards SPQR-trees

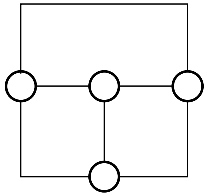


SPQR-trees

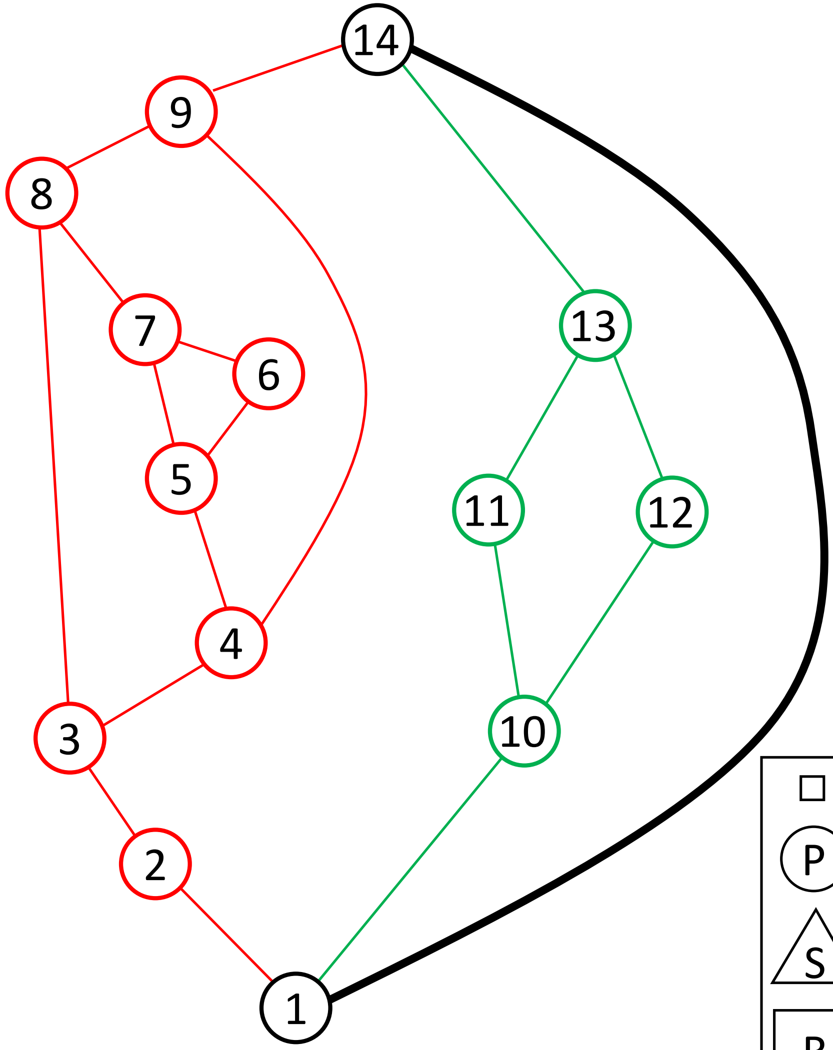


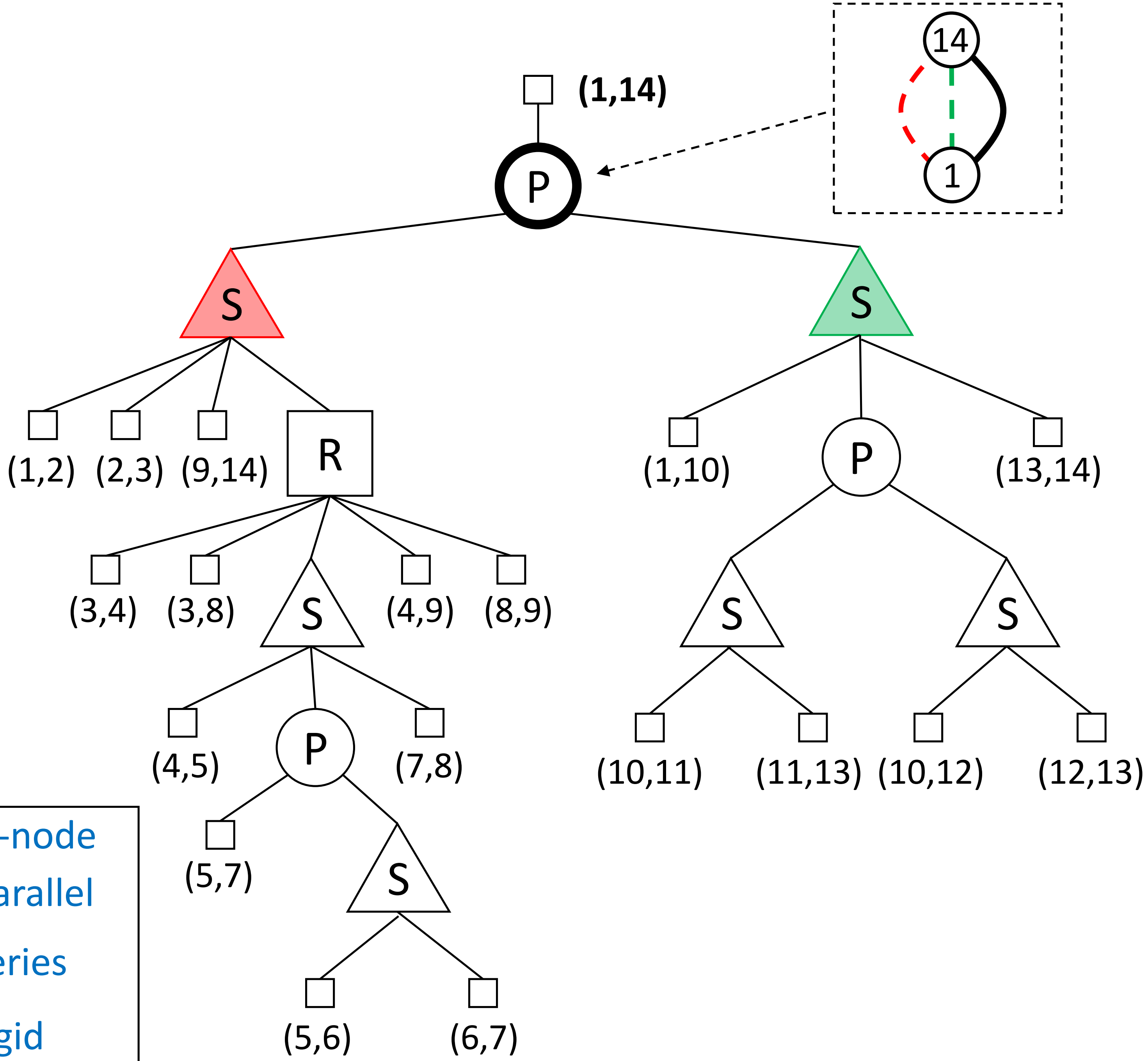


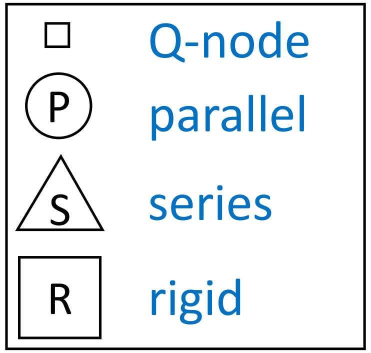
[Di Battista and Tamassia '90]

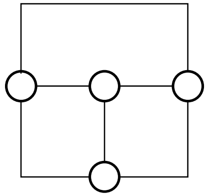


SPQR-trees

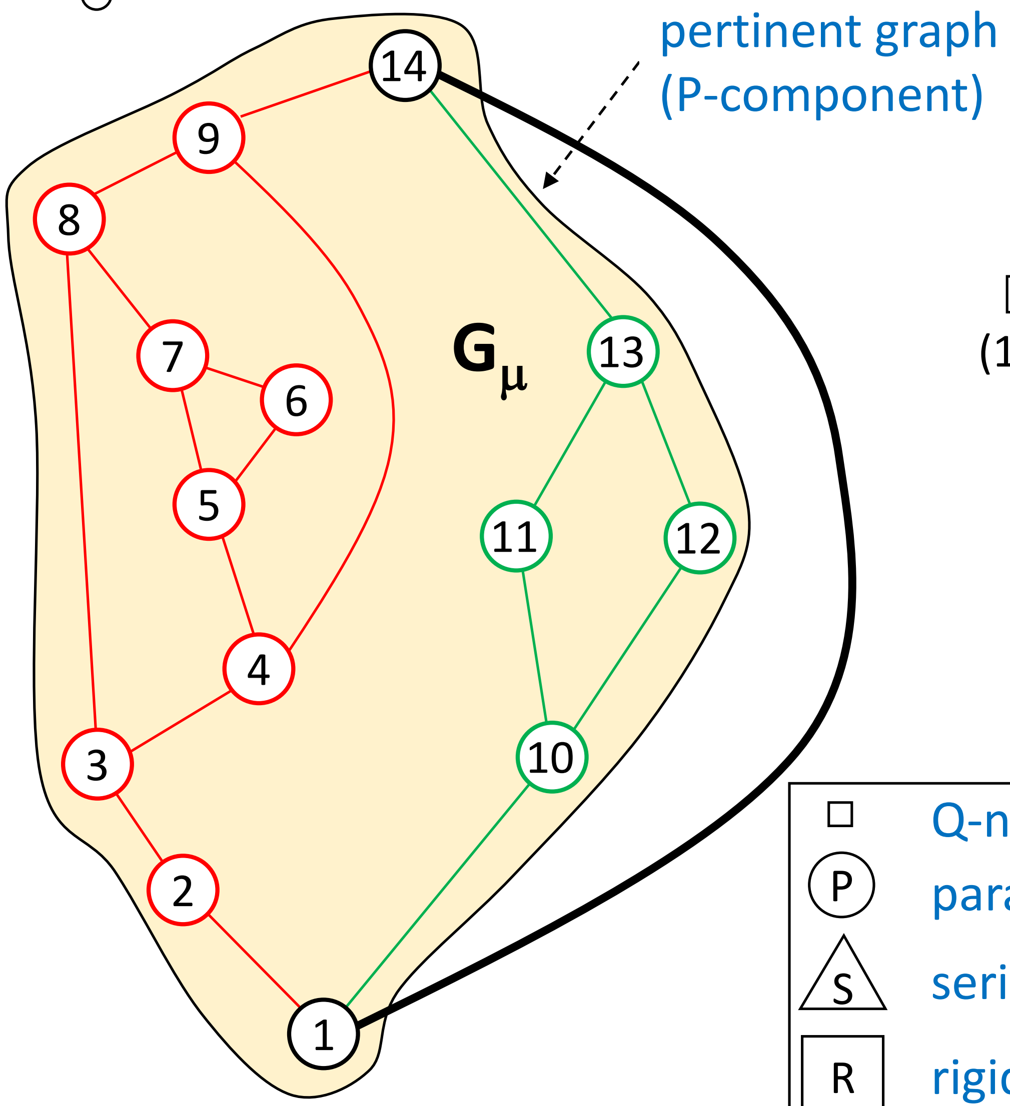

14
9
8
7
6
5
4
3
2
1
13
11
12
10


(1,14)
P
14
1
S
S
(1,2)
(2,3)
(9,14)
R
(1,10)
P
(13,14)
(3,4)
(3,8)
S
(4,9)
(8,9)
S
S
(4,5)
P
(7,8)
(10,11)
(11,13)
(10,12)
(12,13)
(5,7)
S
(5,6)
(6,7)

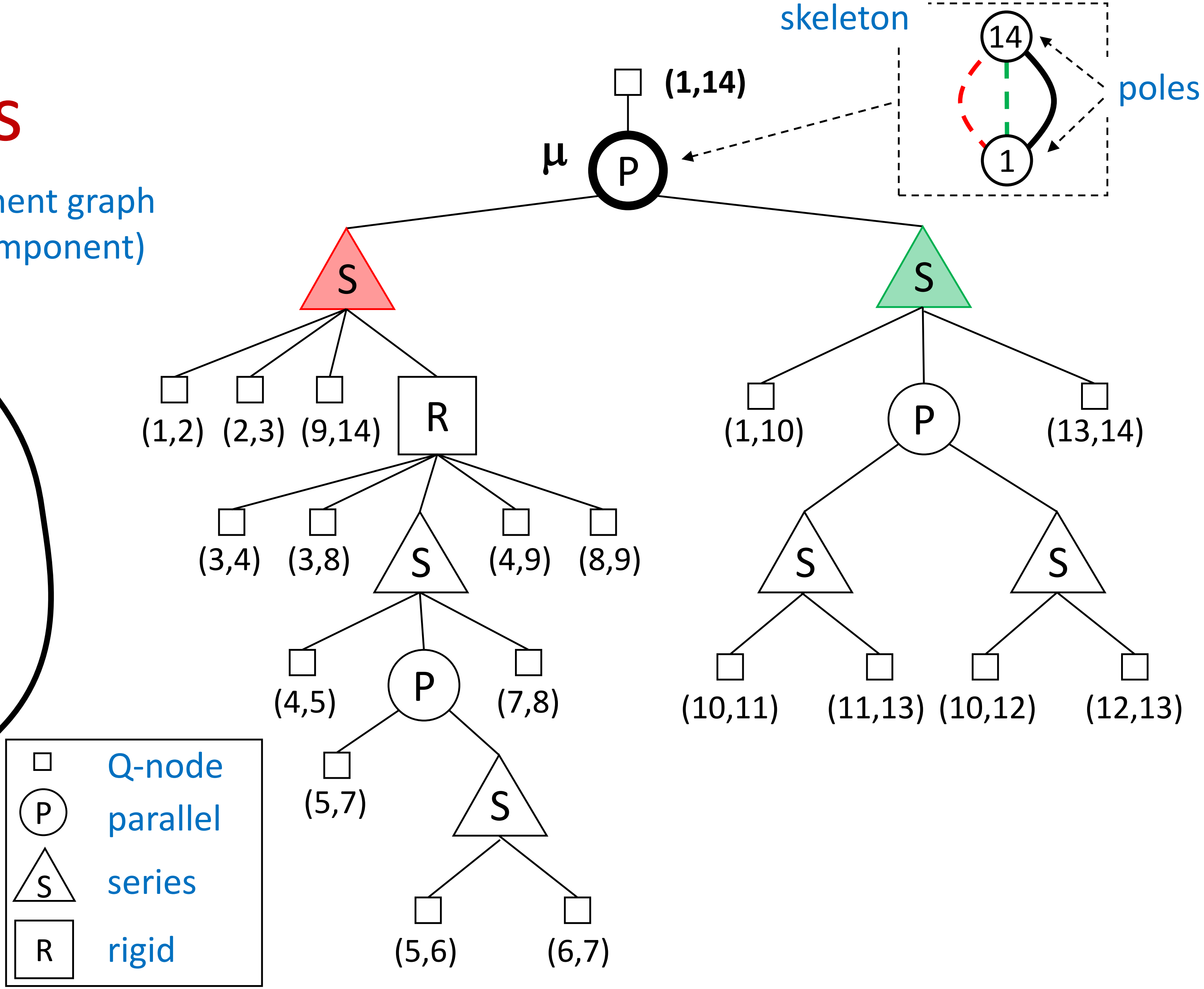

Q-node
P
parallel
S
series
R
rigid



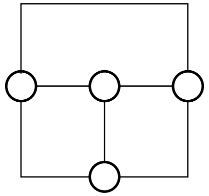
SPQR-trees



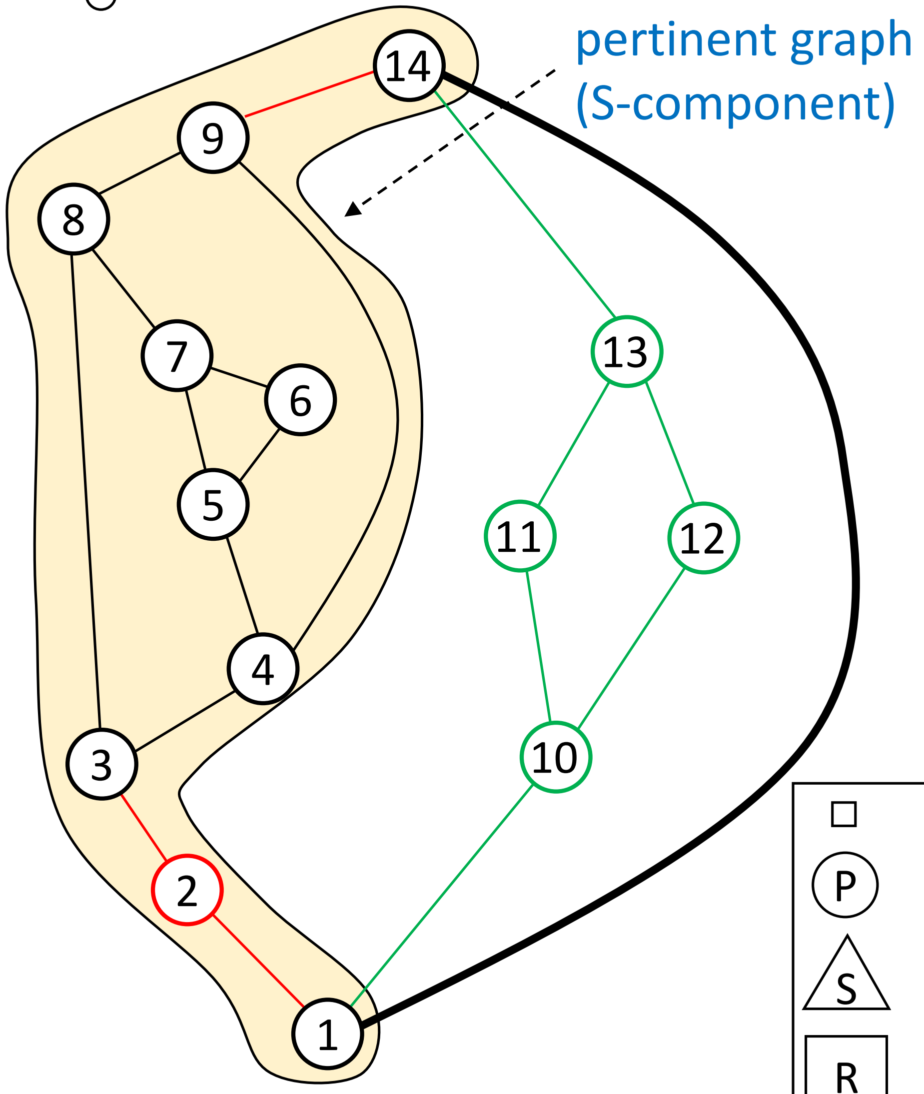
pertinent graph
(P-component)
G_μ
14
9
8
7
6
5
4
3
2
1
13
11
12
10



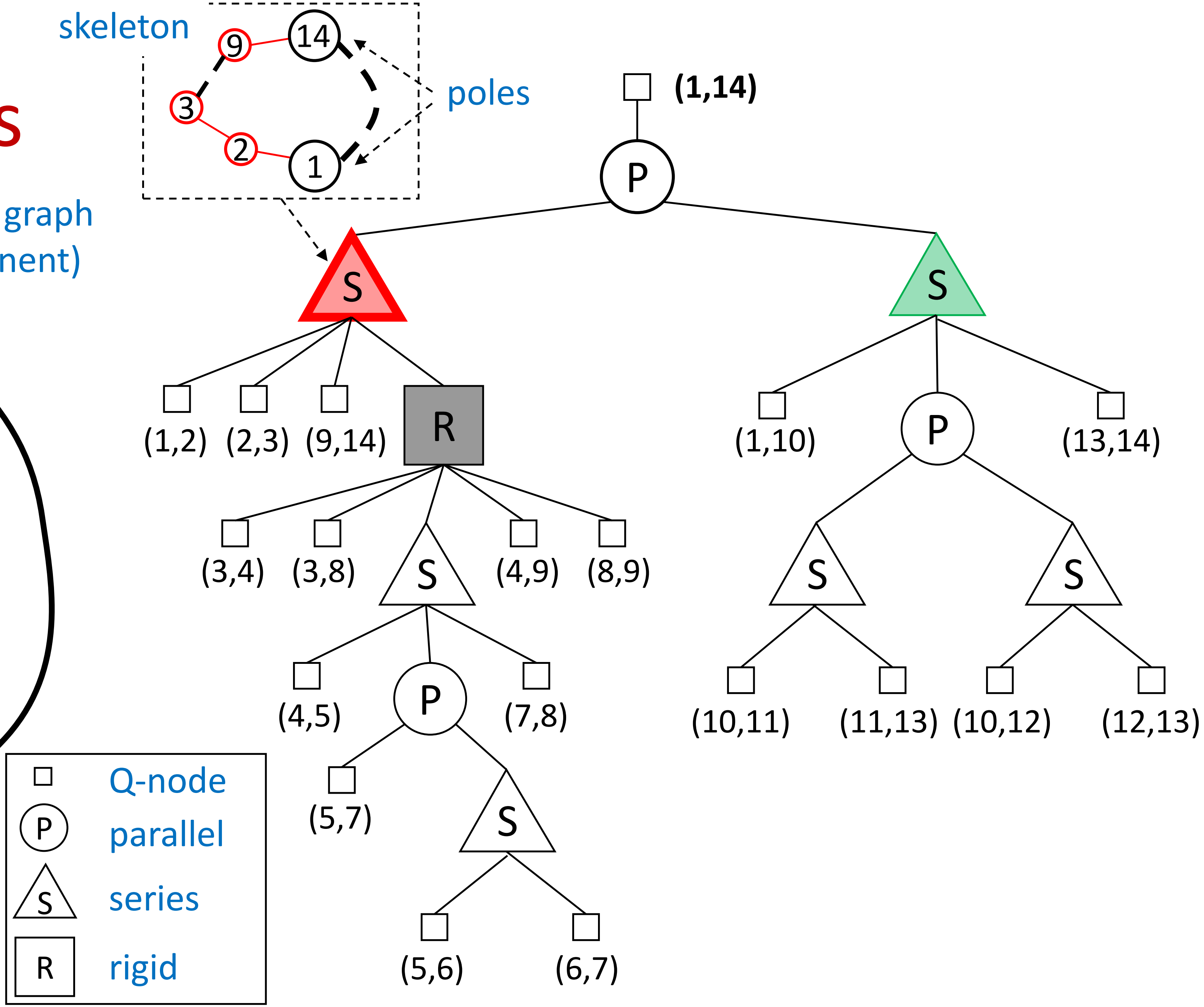
skeleton
poles
14
1
(1,14)
μ
P
S
S
(1,2)
(2,3)
(9,14)
R
(3,4)
(3,8)
S
(4,9)
(8,9)
(4,5)
P
(7,8)
(5,7)
S
(5,6)
(6,7)
(1,10)
P
(13,14)
S
S
(10,11)
(11,13)
(10,12)
(12,13)
Q-node
P
parallel
S
series
R
rigid



SPQR-trees

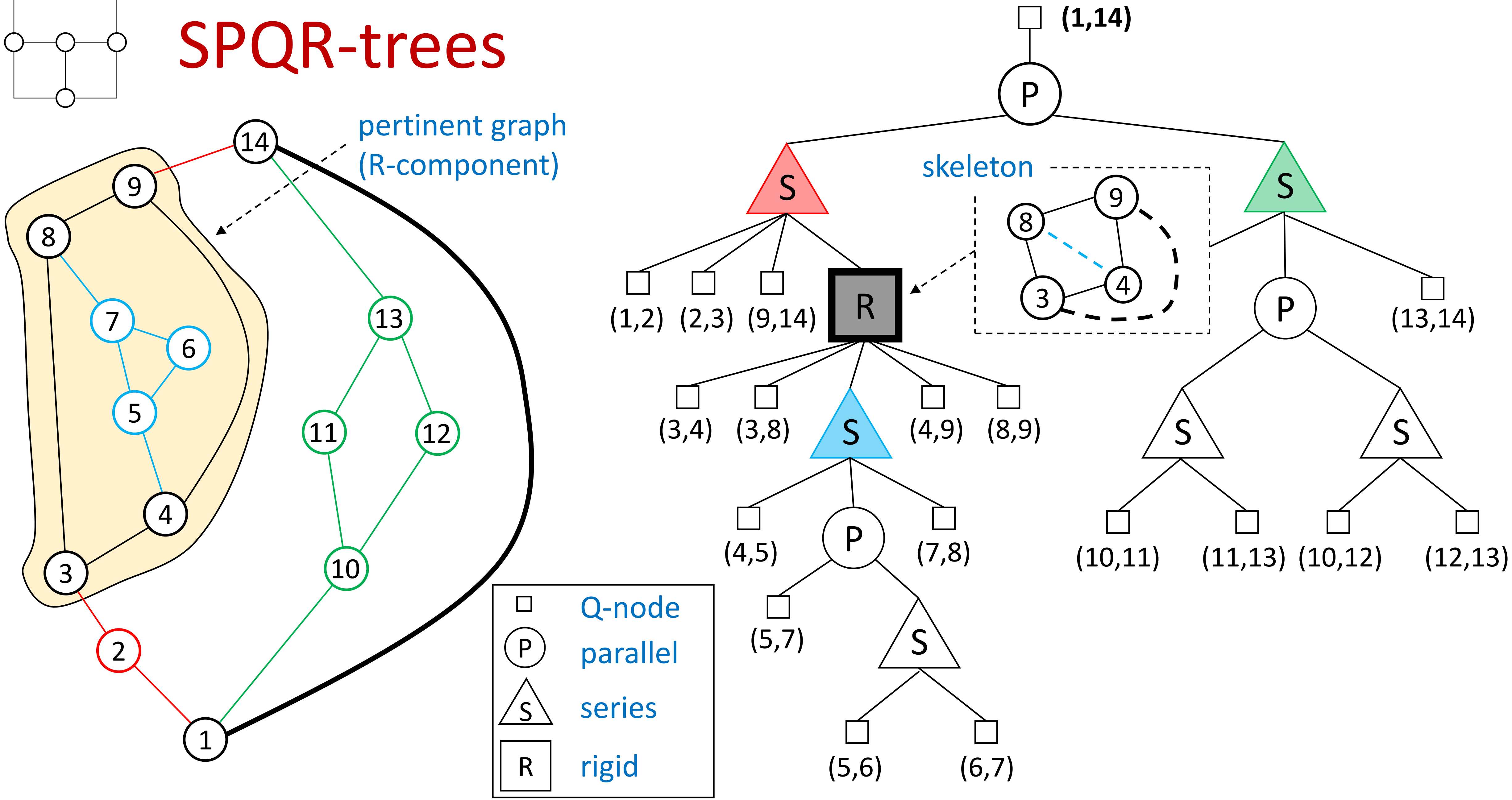


14
9
8
7
6
5
4
3
2
1
13
11
12
10
pertinent graph
(S-component)

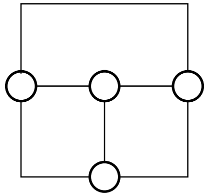


skeleton
9
14
3
2
1
poles
(1,14)
P
S
S
(1,2)
(2,3)
(9,14)
R
(1,10)
P
(13,14)
(3,4)
(3,8)
S
(4,9)
(8,9)
S
S
(4,5)
P
(7,8)
(10,11)
(11,13)
(10,12)
(12,13)
(5,7)
S
(5,6)
(6,7)
Q-node
P
parallel
S
series
R
rigid

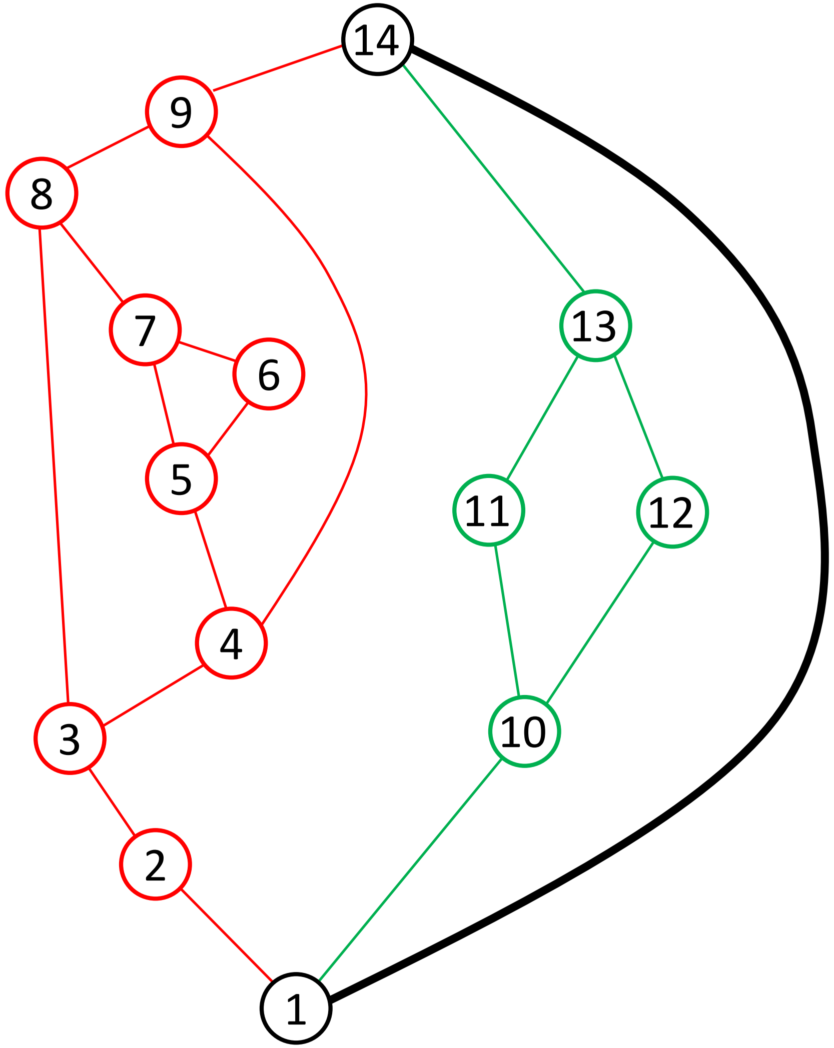
SPQR-trees

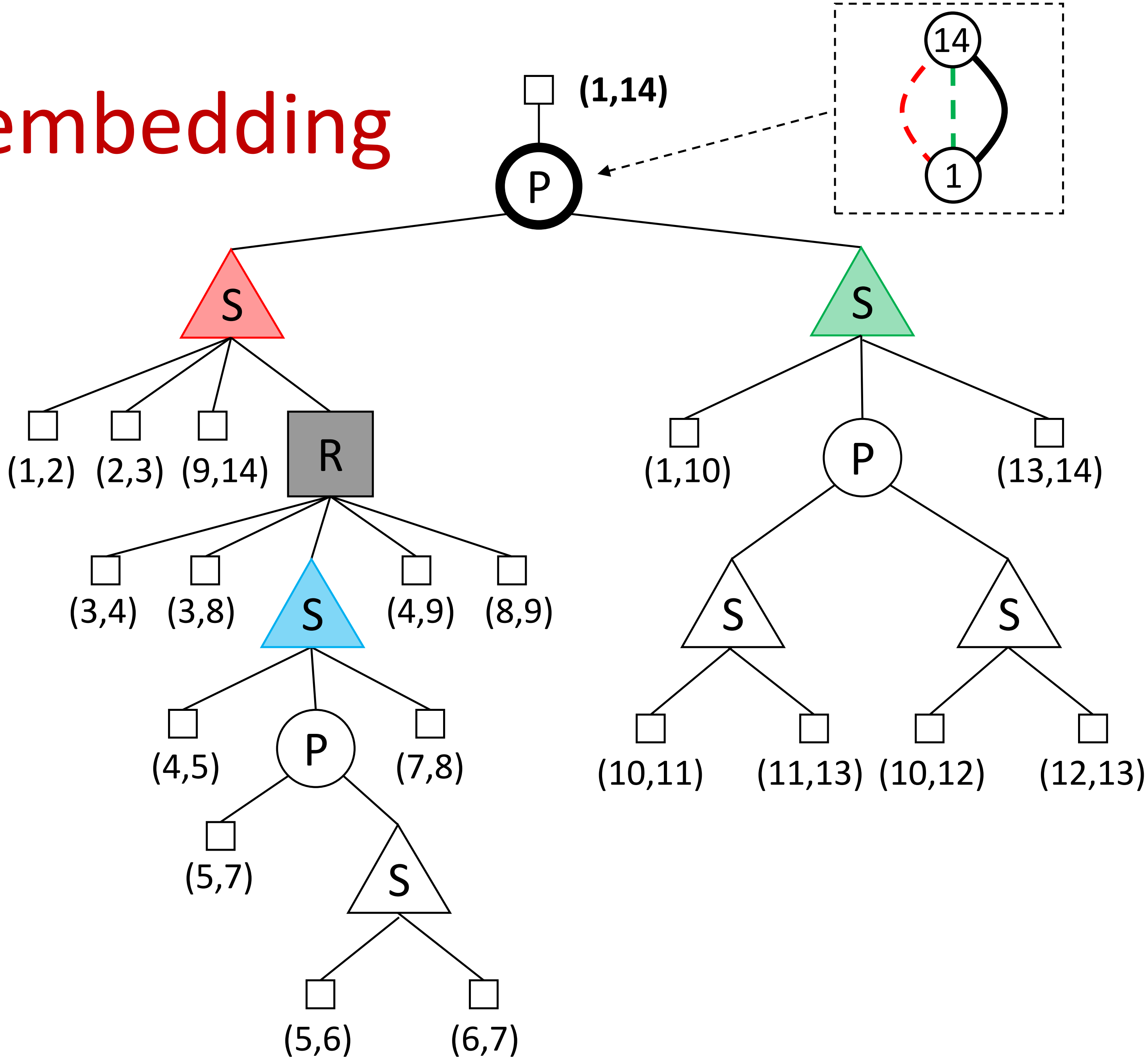


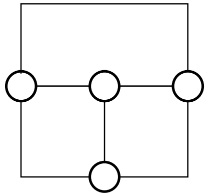
pertinent graph
(R-component)
14
9
8
7
6
5
4
3
2
1
13
11
12
10
(1,14)
P
S
skeleton
S
(1,2)
(2,3)
(9,14)
R
(13,14)
(3,4)
(3,8)
S
(4,9)
(8,9)
(4,5)
P
(7,8)
(5,7)
S
(5,6)
(6,7)
(10,11)
(11,13)
(10,12)
(12,13)
Q-node
P parallel
S series
R rigid



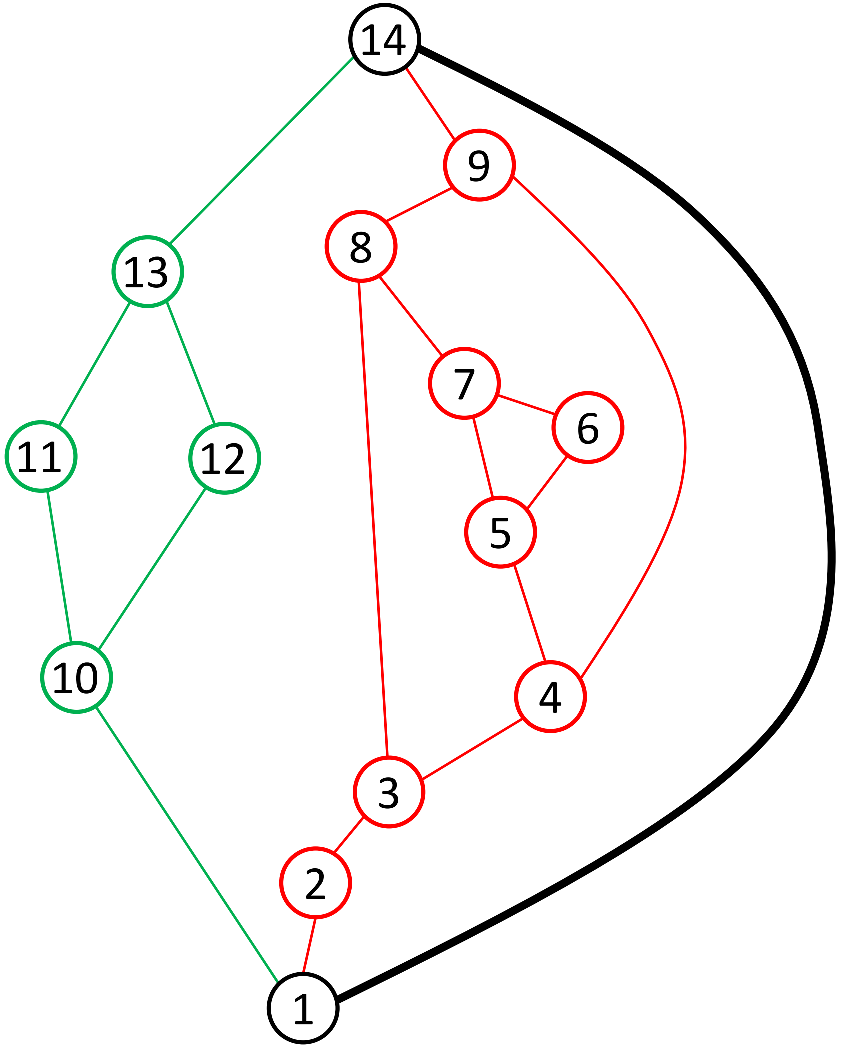
Changing the embedding

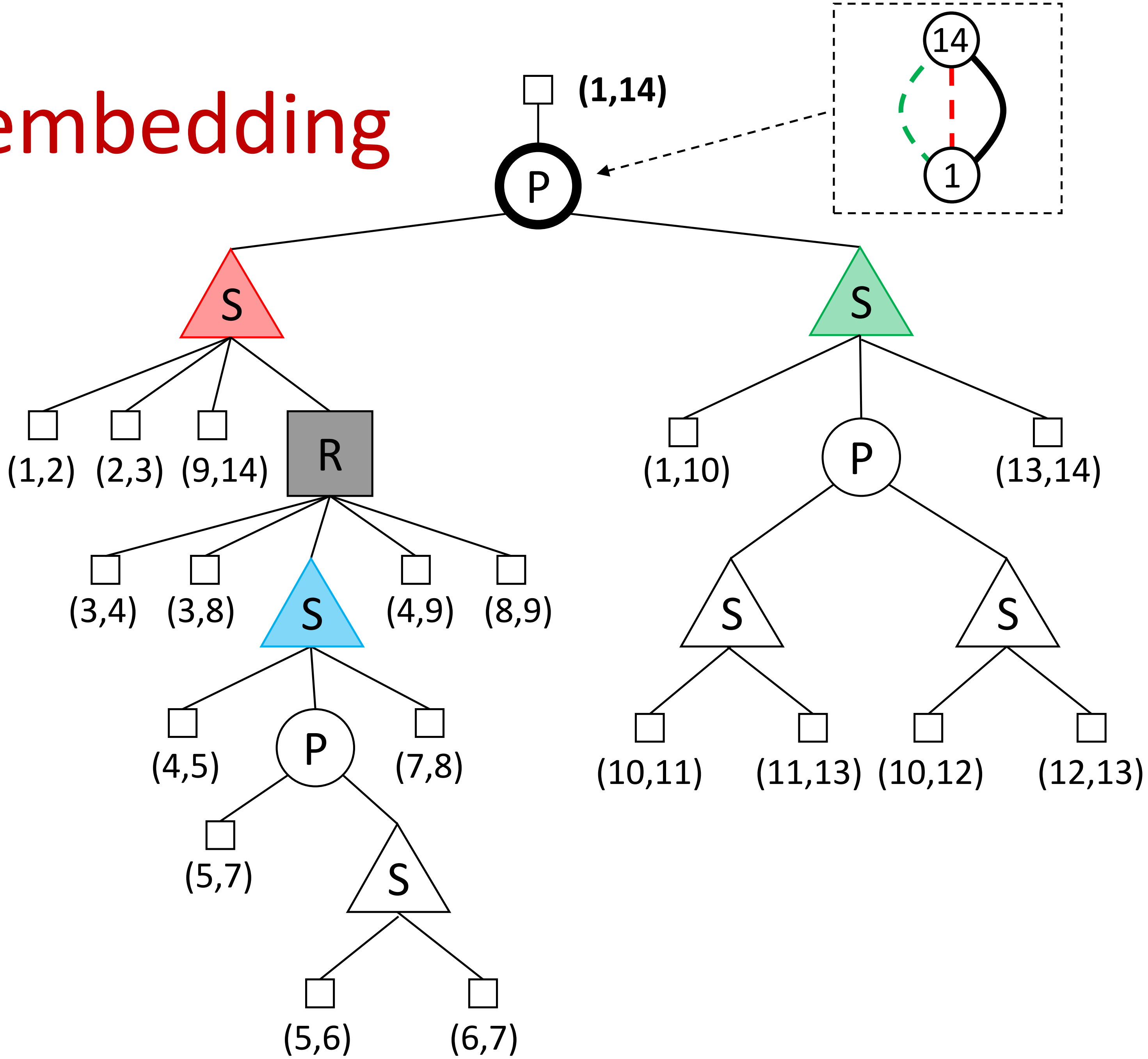




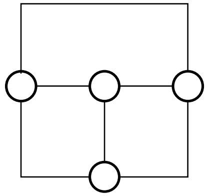


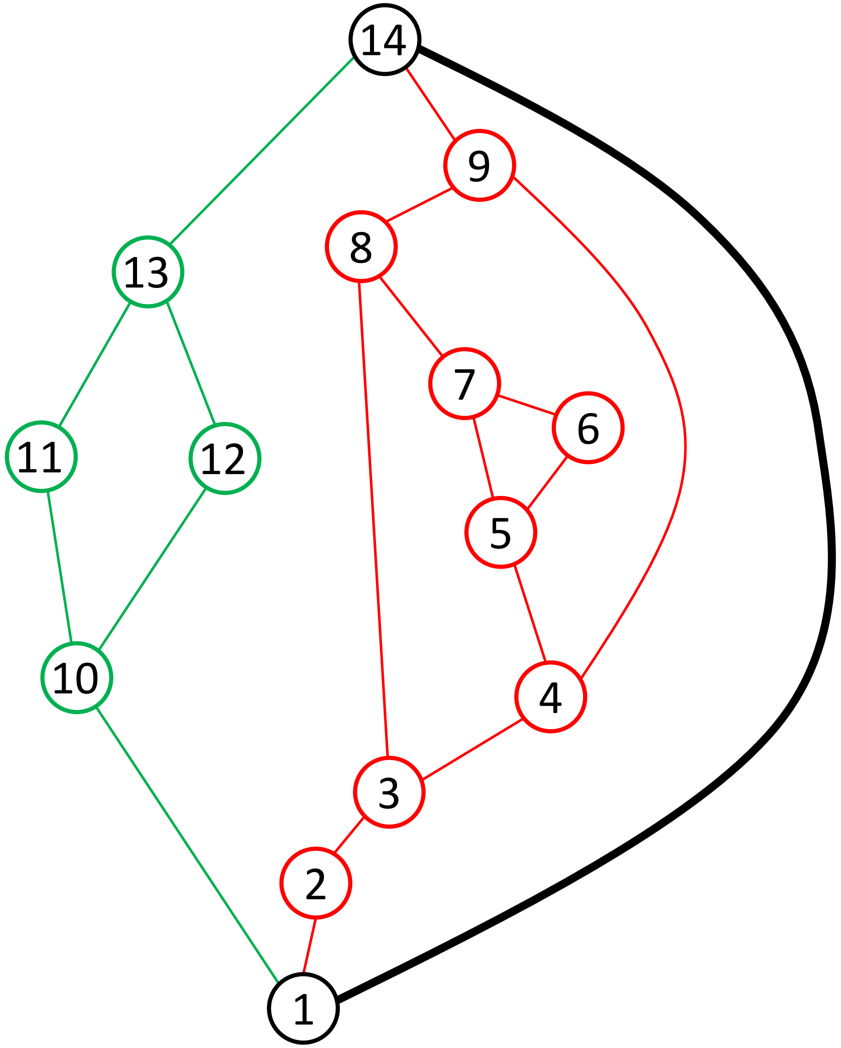
Changing the embedding

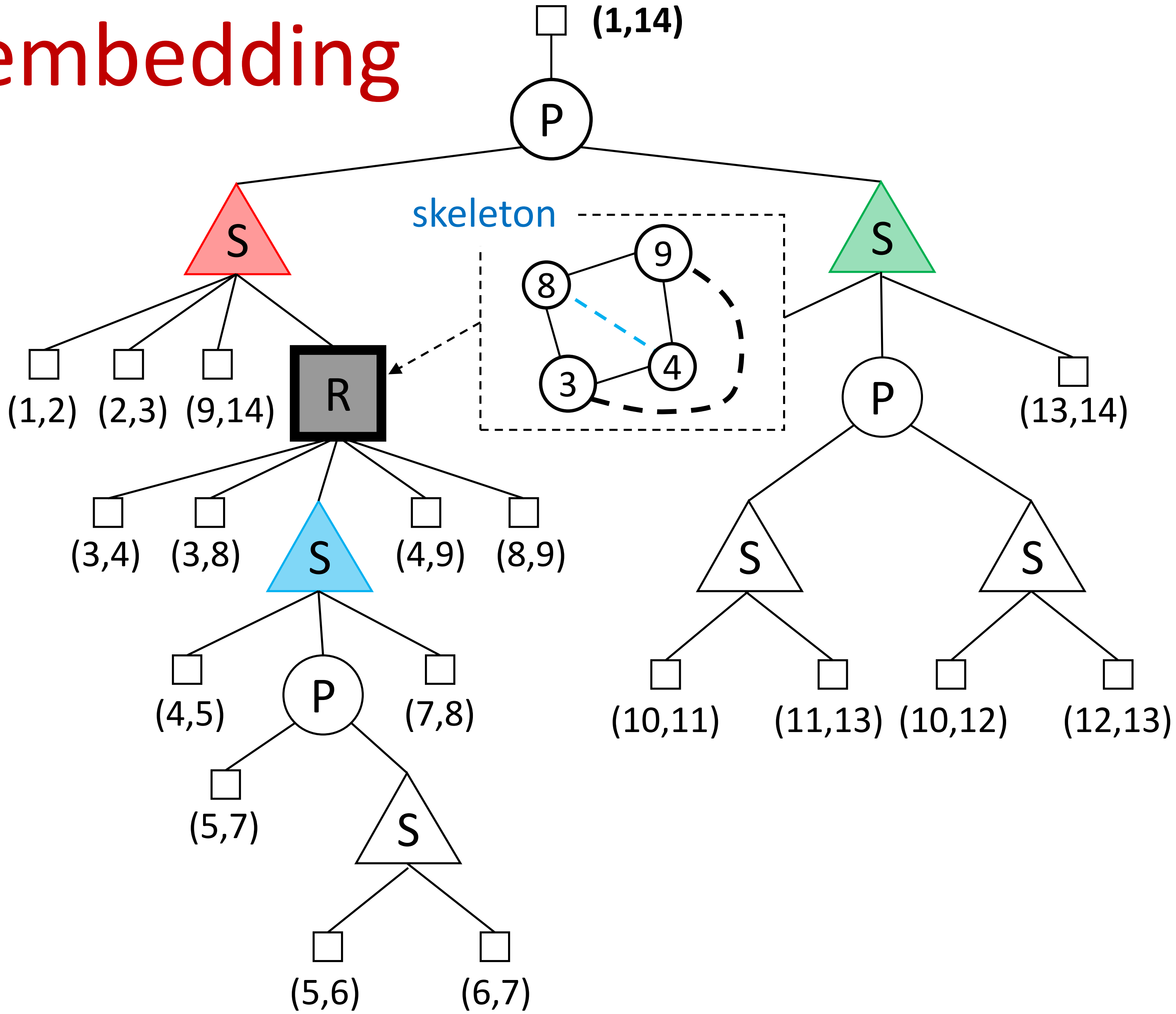




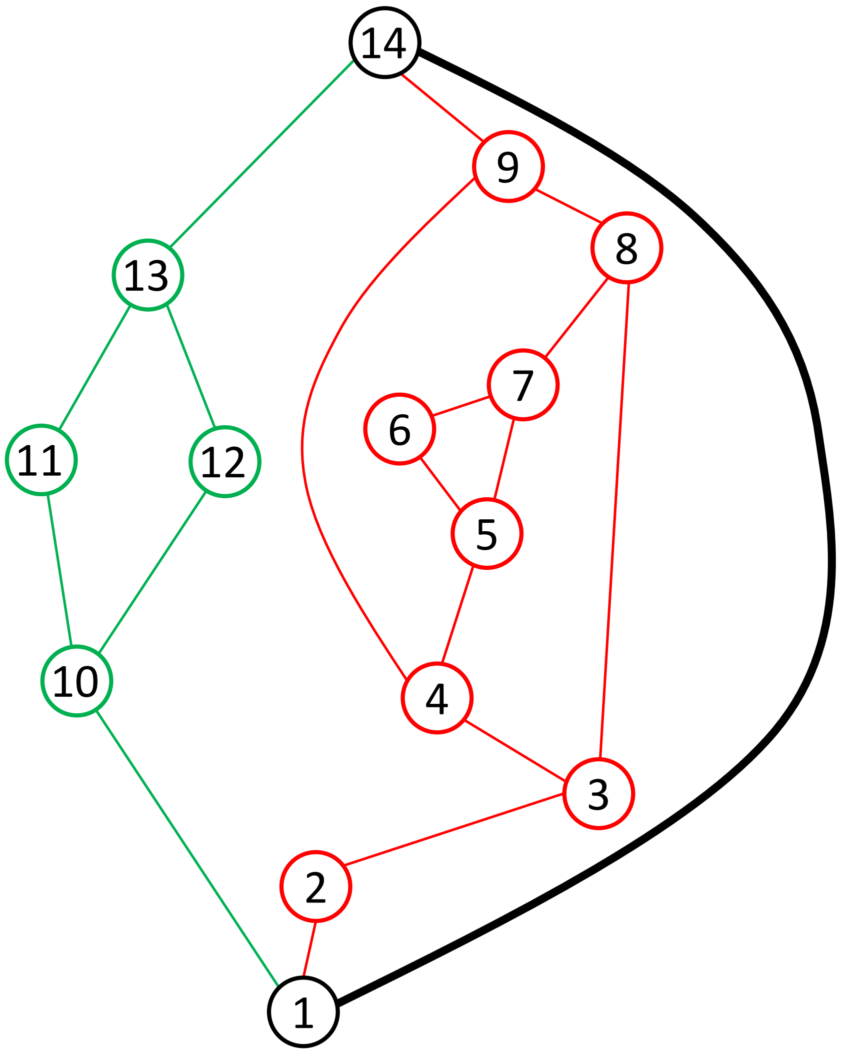
Changing the embedding

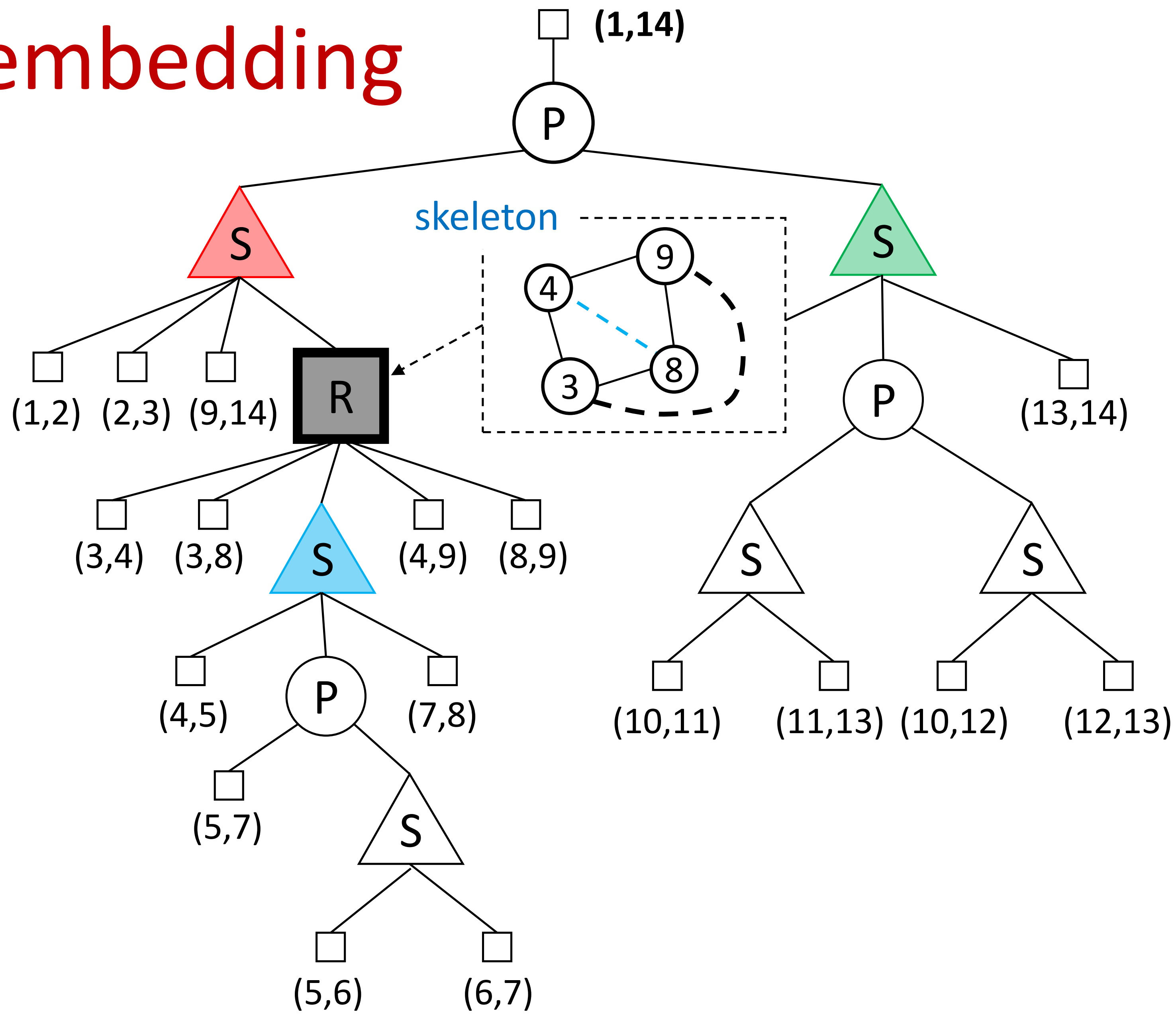


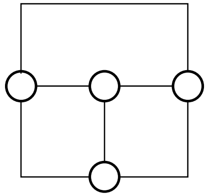




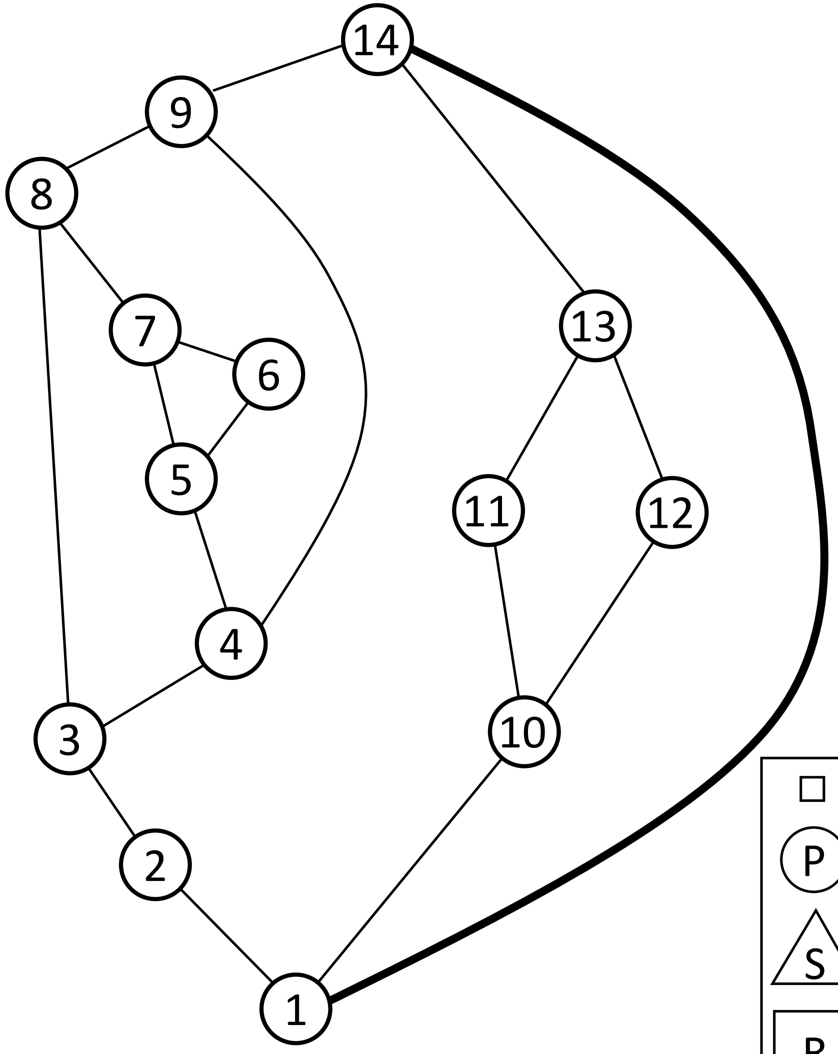
Changing the embedding

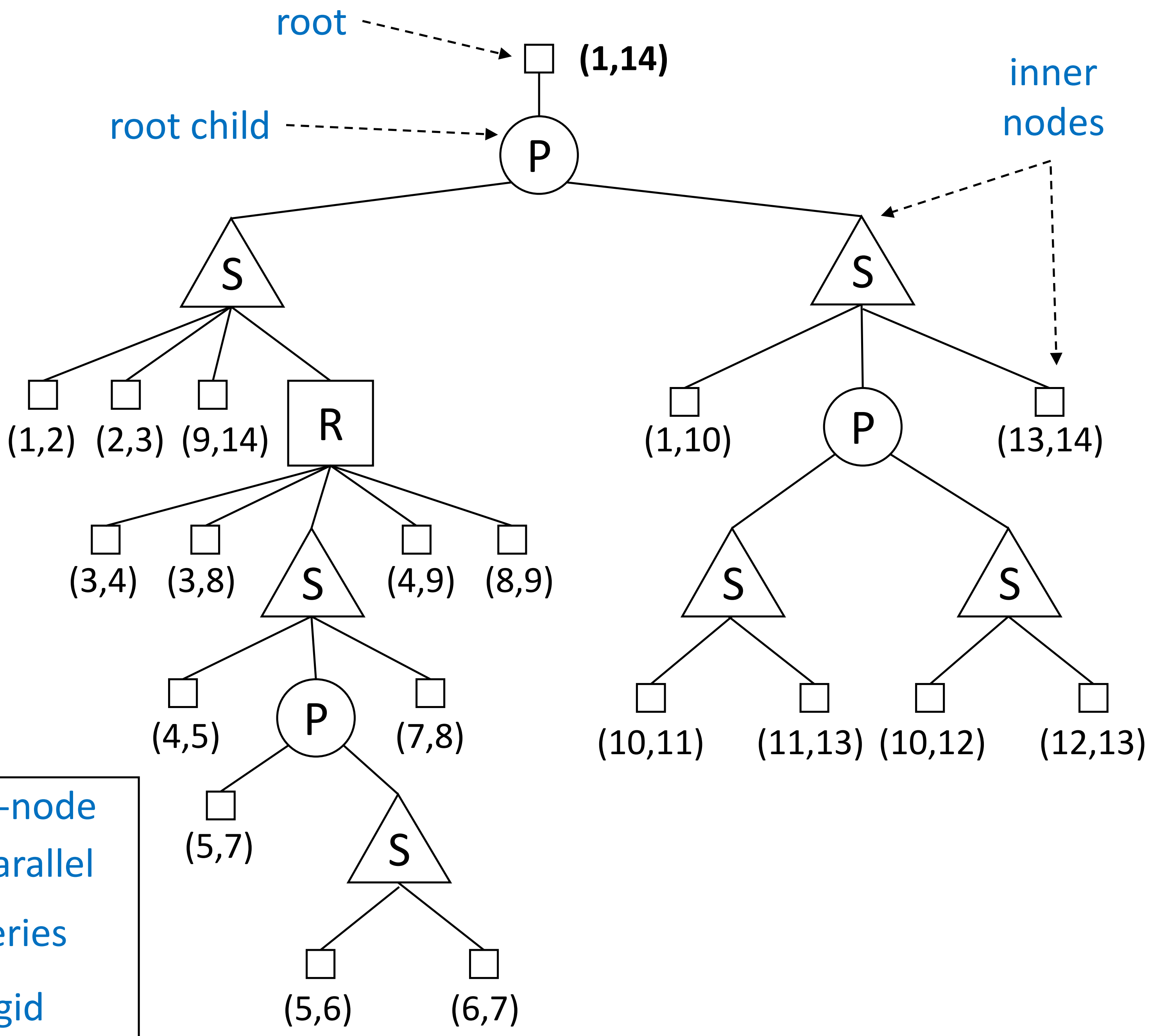


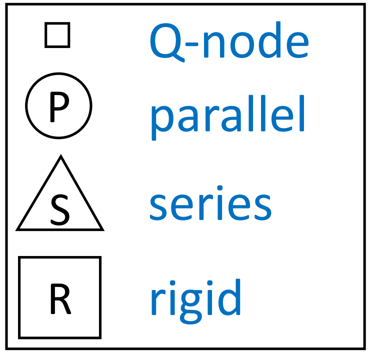


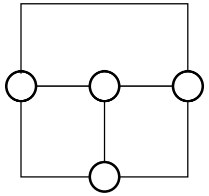


SPQR-trees


14
9
8
7
6
5
4
3
2
1
13
11
12
10


root
(1,14)
root child
P
inner
nodes
S
S
(1,2)
(2,3)
(9,14)
R
(1,10)
P
(13,14)
(3,4)
(3,8)
S
(4,9)
(8,9)
S
S
(4,5)
P
(7,8)
(10,11)
(11,13)
(10,12)
(12,13)
(5,7)
S
(5,6)
(6,7)

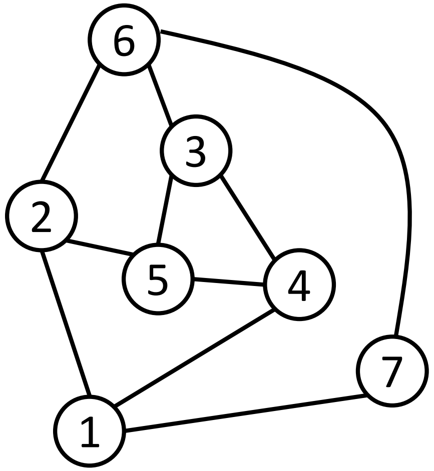

Q-node
P
parallel
S
series
R
rigid



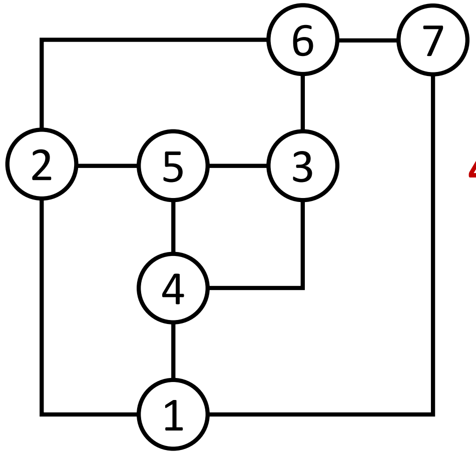
Bend-minimum orthogonal drawings of planar 3-graphs

The problem

Problem: planar **3-graph** ⟹ planar **bend-minimum** orthogonal drawing

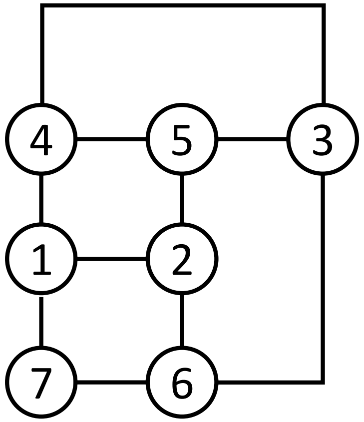


plane 3-graph



4 bends

bend-min orthogonal drawing
(**fixed embedding**)



3 bends

bend-min orthogonal drawing
(**variable embedding**)

History reminder

Bend-min orthogonal drawings: **fixed embedding**

- plane 4-graphs
  - $O(n^2 \log n)$ [*Tamassia* (1987)]
  - $O(n^{7/4} \sqrt{\log n})$ [*Garg, Tamassia* (2001)]
  - $O(n^{1.5})$ [*Cornelsen, Karrenbauer* (2011)]

based on
min-cost flow

- plane 3-graphs

  $O(n)$ [*Rahman, Nishizeki* (2002)]

not based on
flow techniques

History reminder

Bend-min orthogonal drawings: **variable embedding**

- planar 4-graphs: NP-hard [Garg, Tamassia (2001)]
- planar 3-graphs

1998

$O(n^5 \log n)$
Di Battista-Liotta-Vargiu

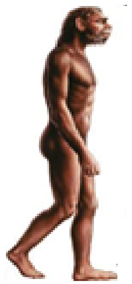
2011

$O(n^{4.5})$
consequence of Cornelsen-Karrenbauer

2017

$O(n^{2.43} \log^k n)$
Chang and Yen

2018



$O(n^2)$
next slides

?

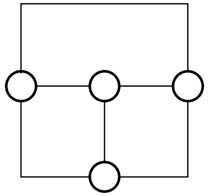
Can we do better?

Result

Theorem. Let G be an n-vertex (simple) planar 3-graph. There exists an $O(n^2)$-time algorithm that computes a bend-minimum orthogonal drawing of G, with at most two bends per edge.

P. S. the algorithm takes $O(n)$ time if we require that a prescribed edge of G is on the external face

W. Didimo, G. Liotta, M. Patrignani: Bend-Minimum Orthogonal Drawings in Quadratic Time. Graph Drawing 2018: 481-494



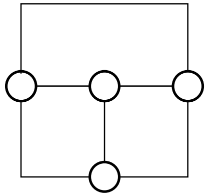
General strategy for biconnected graphs

input: G biconnected planar 3-graph with n vertices

output: bend-min orthogonal drawing Γ of G

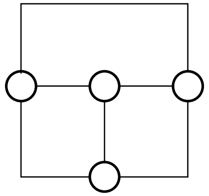
- for each edge *e* of G
  - $\Gamma_e \leftarrow$ bend-min orthogonal drawing of G with *e* on the external face
- return $\Gamma \leftarrow$ min-bends $\{\Gamma_e\}$

Γ_e is computed in O(n) time



Strategy for the linear-time algorithm

- Incremental construction of Γ_e
  1. bottom-up visit of the SPQR-tree + *orthogonal spirality*
     - similar to [*G. Di Battista, G. Liotta, F. Vargiu*: Spirality and optimal orthogonal drawings, SIAM J. Comput., 27 (1998)]
  2. new properties of bend-min orthogonal drawings of planar 3-graphs
  3. non-flow based computation of bend-min orthogonal drawings for the rigid components



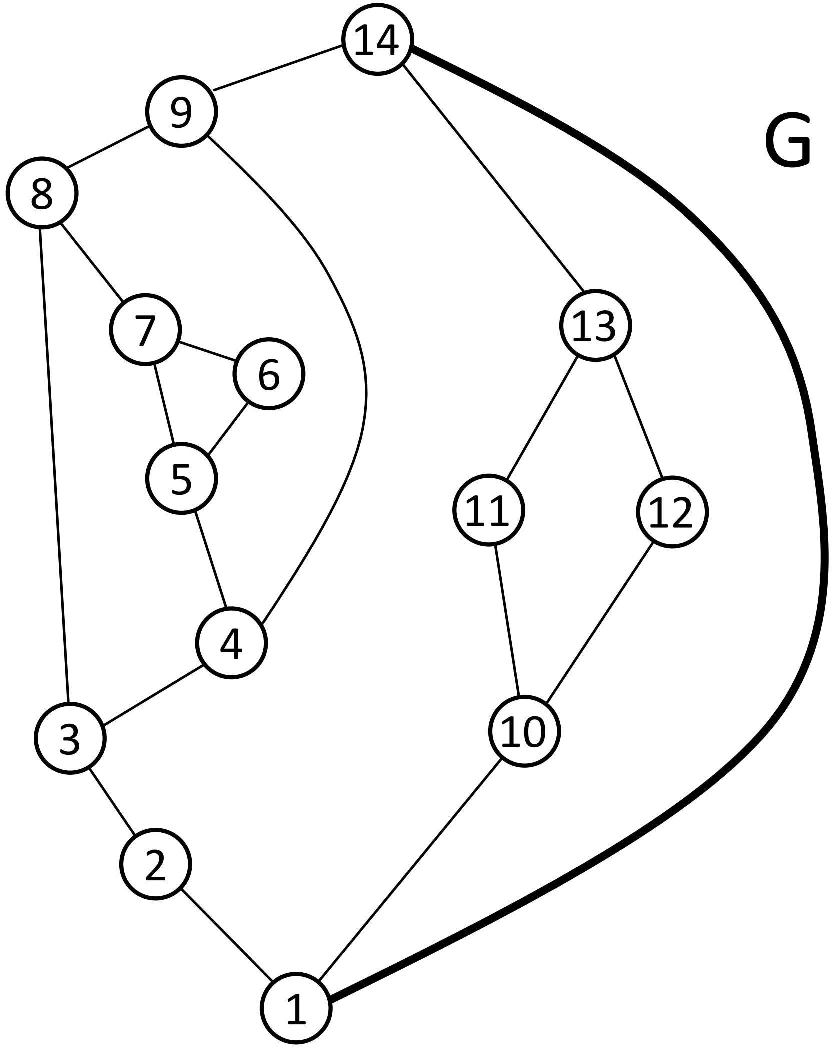
Orthogonal representations: reminder

orthogonal representation = equivalence class of orthogonal drawings with the same vertex angles and the same sequence of bends along the edges

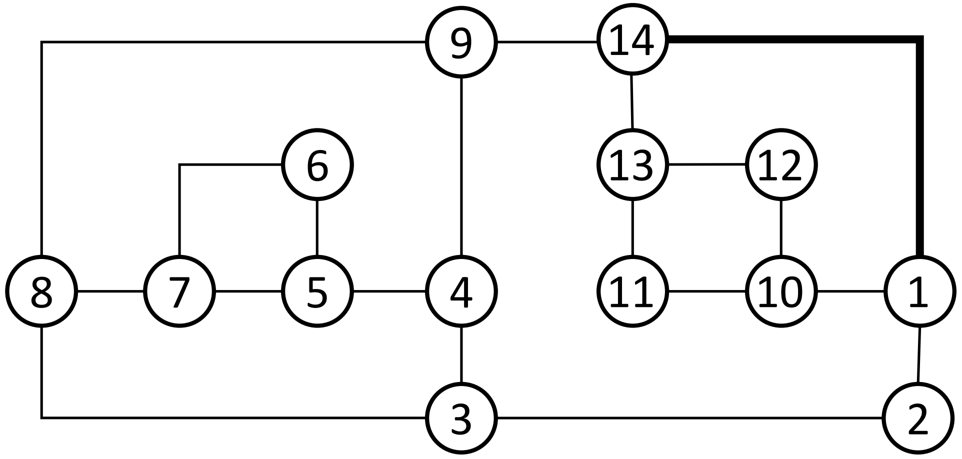
- a drawing of an orthogonal representation can be computed in linear time

orthogonal component = orthogonal representation H_μ of a component G_μ

Orthogonal components: example

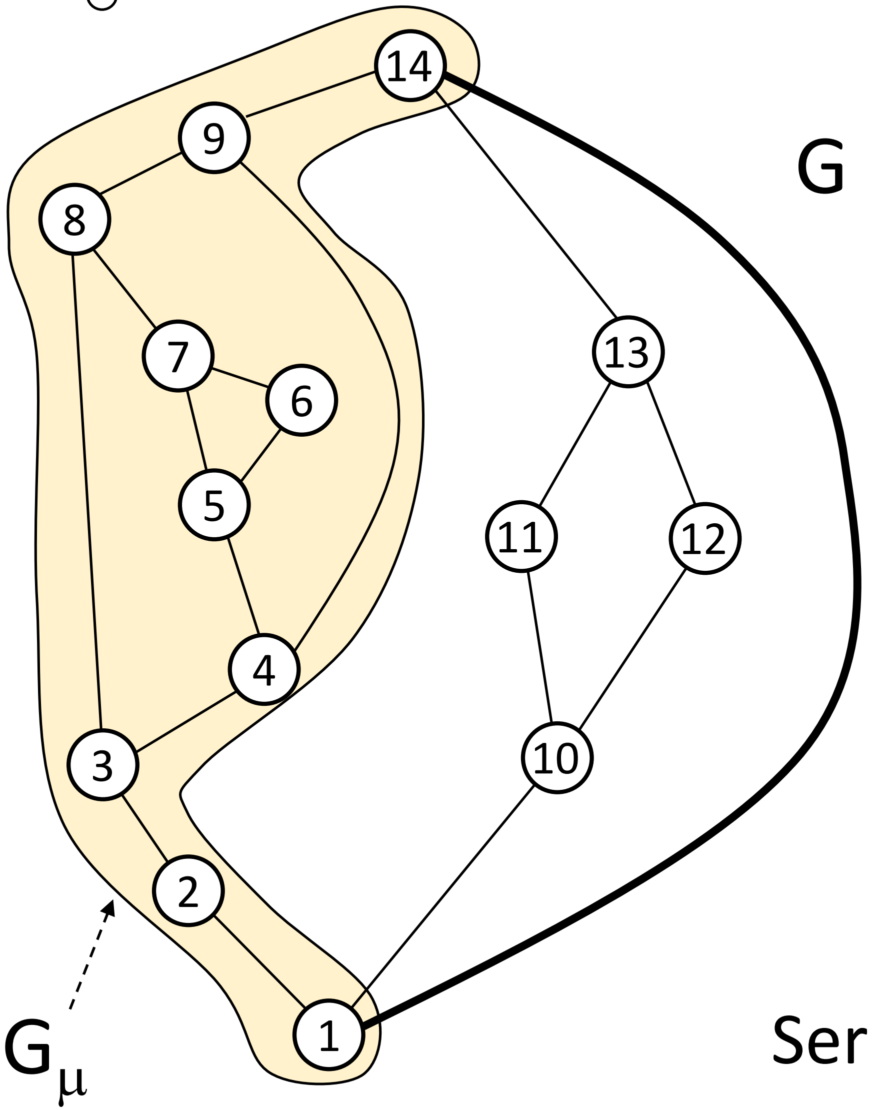


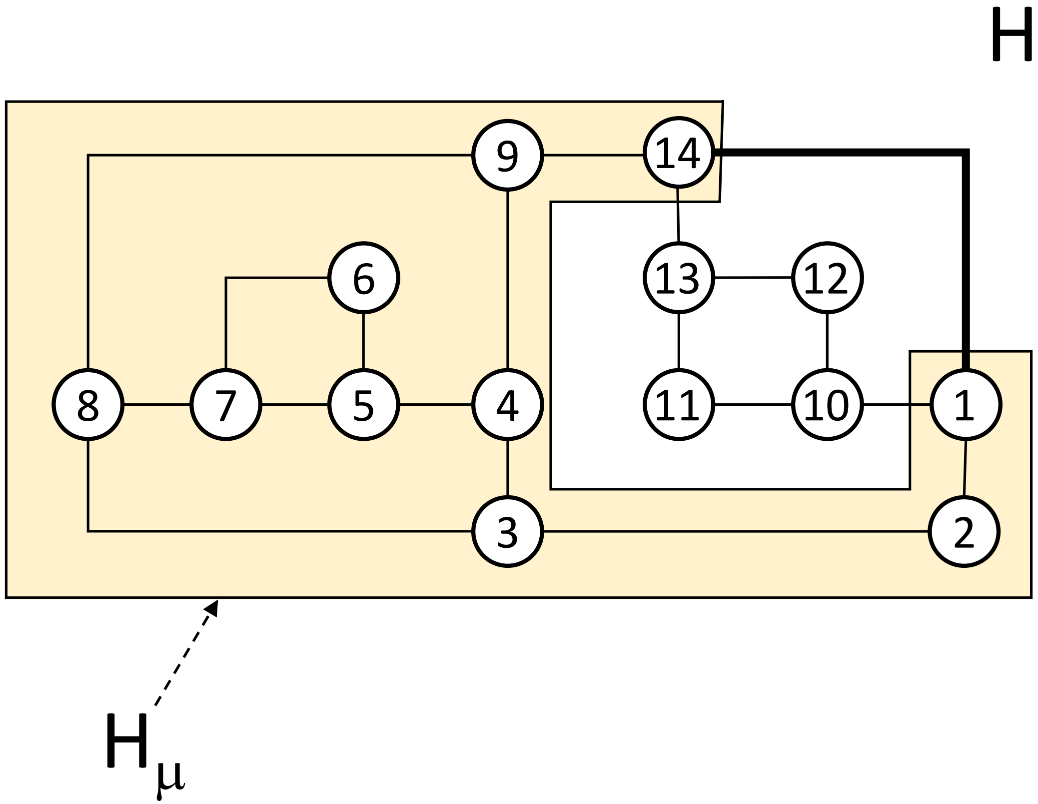
G



H

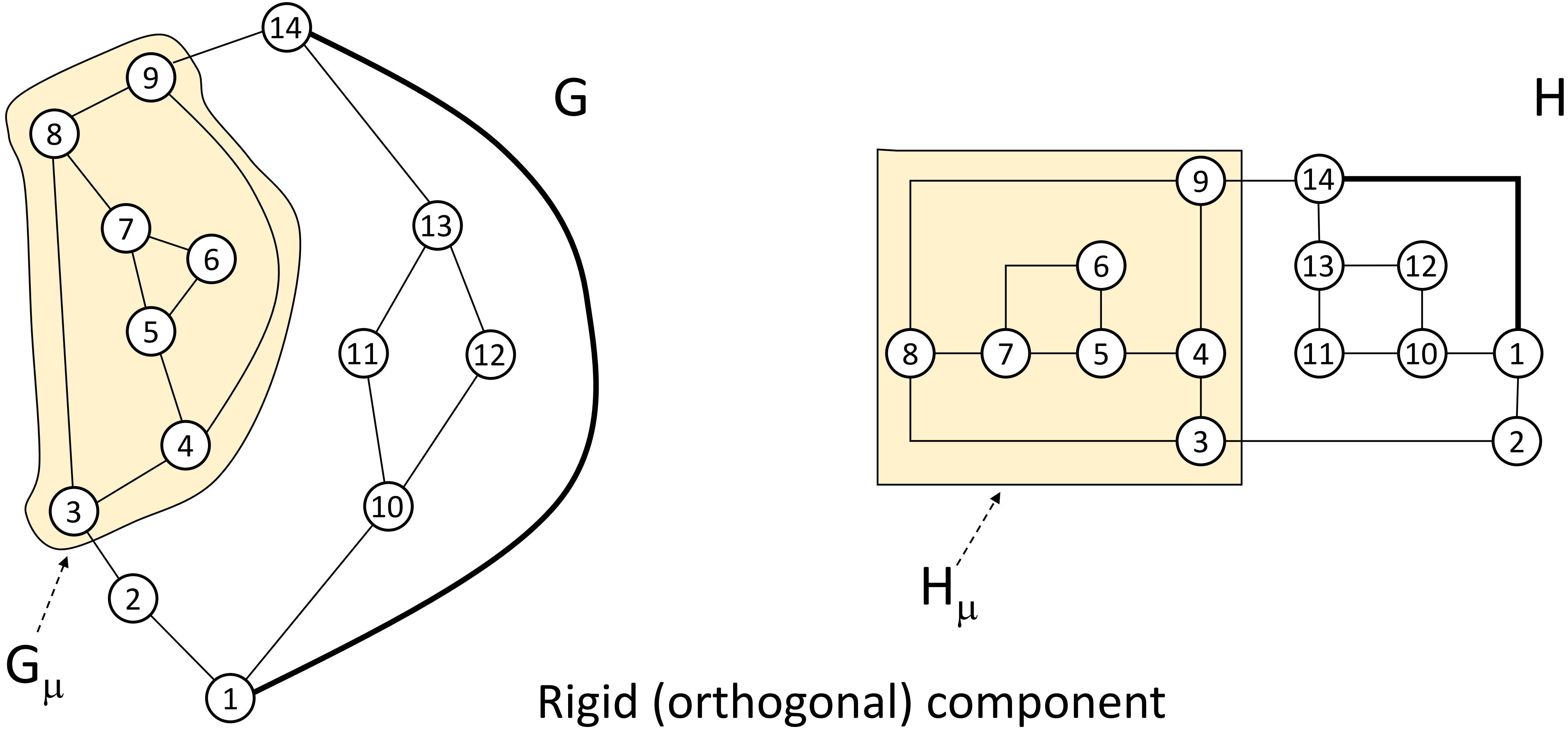
Orthogonal components: examples





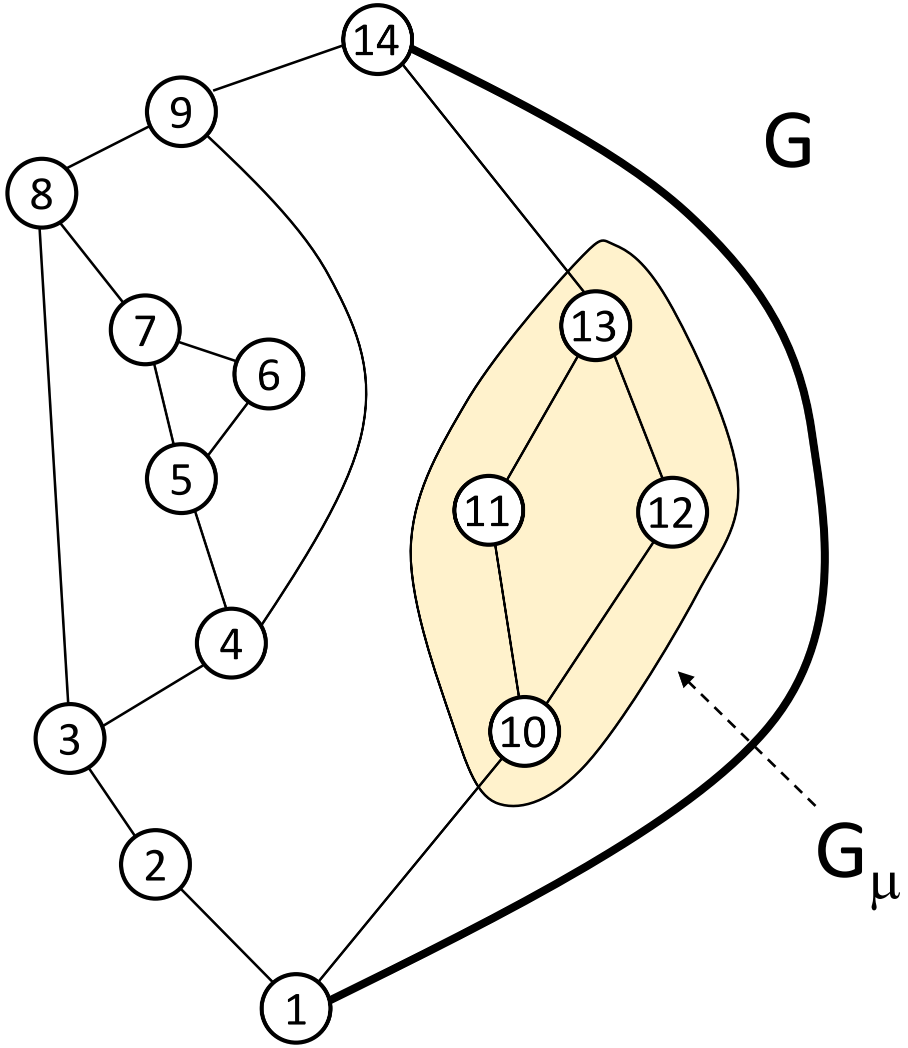
Series (orthogonal) component

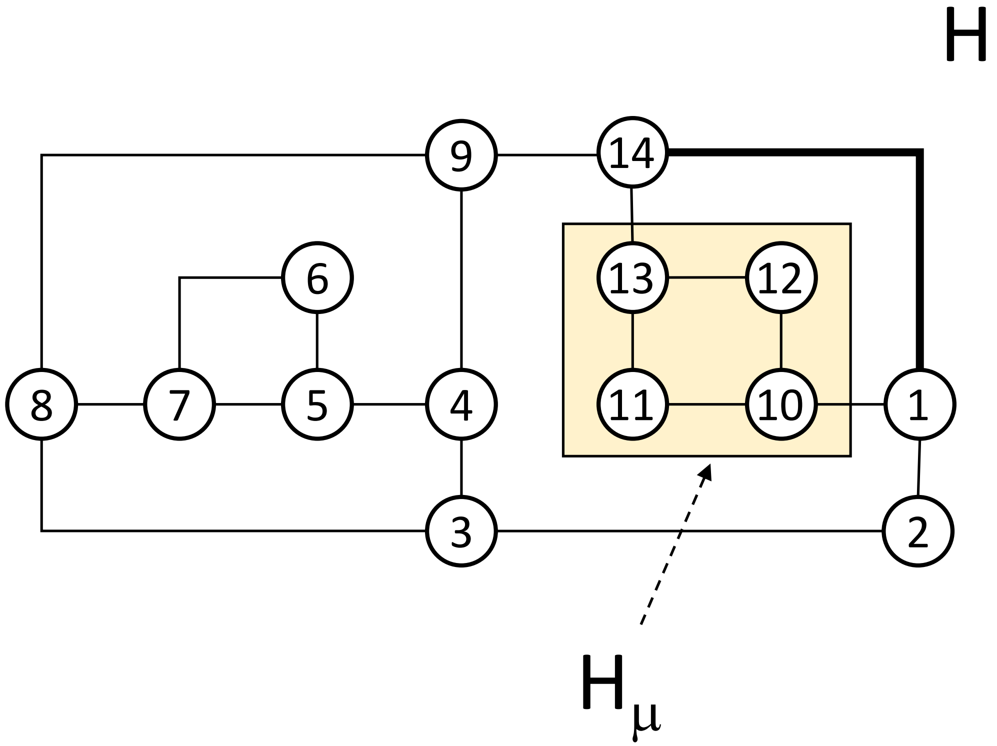
Orthogonal components: examples



Rigid (orthogonal) component

Orthogonal components: examples

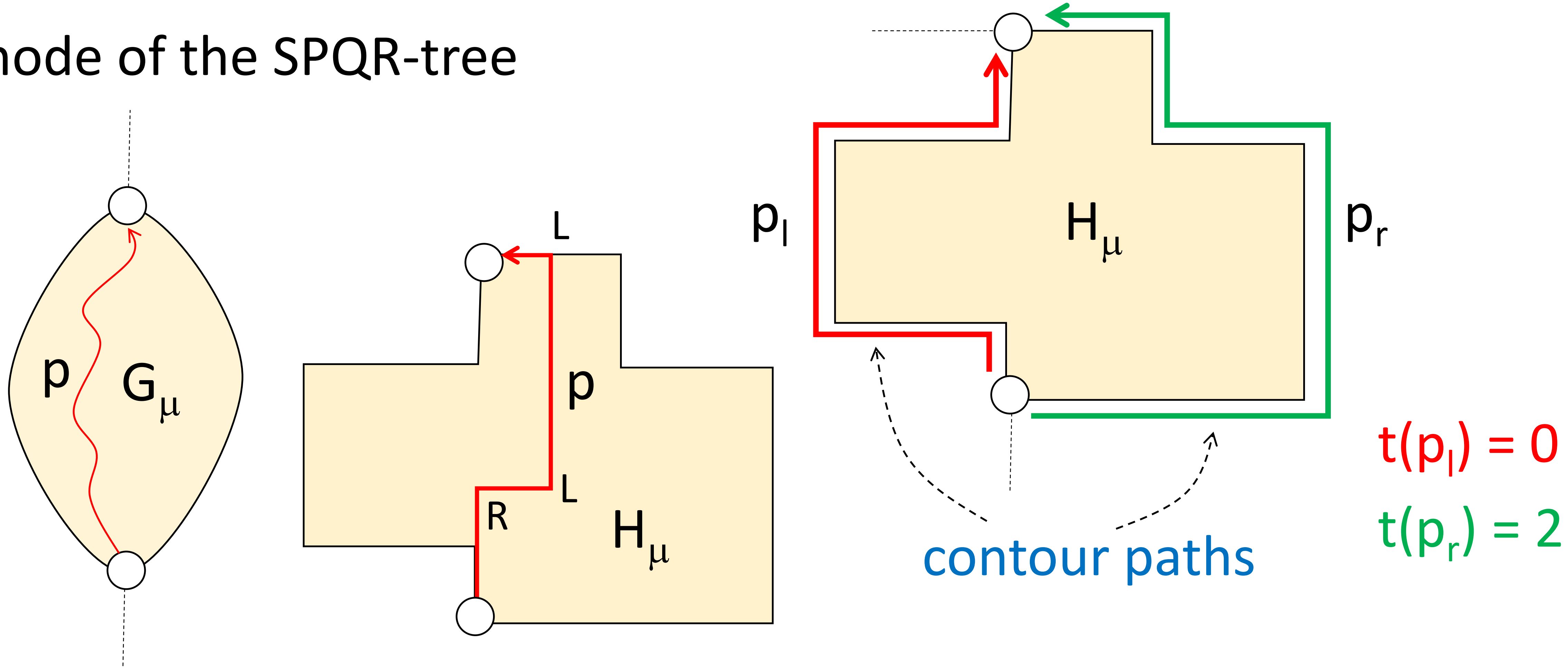




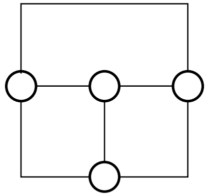
Parallel (orthogonal) component

Turn number and contour paths

μ = node of the SPQR-tree



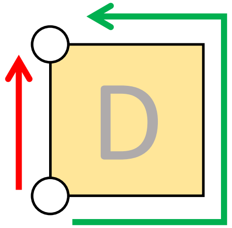
t(p) = turn number = |#left turns – # right turns| (along p)



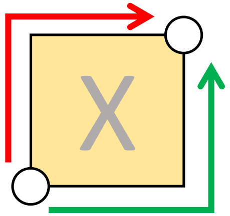
P- and R-components: Representative shapes

μ = P-node or R-node

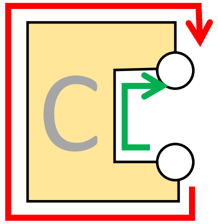
H_μ is D-shaped $\Leftrightarrow$ $t(p_l) = 0$ and $t(p_r) = 2$ or vice versa



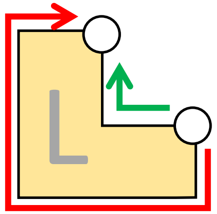
H_μ is X-shaped $\Leftrightarrow$ $t(p_l) = t(p_r) = 1$

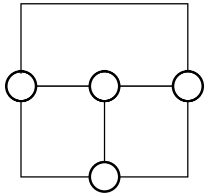


H_μ is C-shaped $\Leftrightarrow$ $t(p_l) = 4$ and $t(p_r) = 2$ or vice versa



H_μ is L-shaped $\Leftrightarrow$ $t(p_l) = 3$ and $t(p_r) = 1$ or vice versa

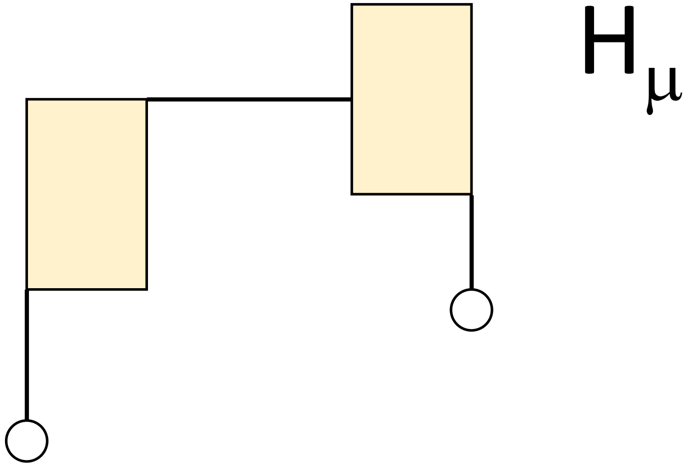


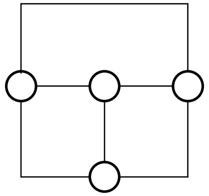


Inner S-components: spirality

μ = inner S-node

Lemma. All paths between the poles of an orthogonal component H_μ have the same turn number

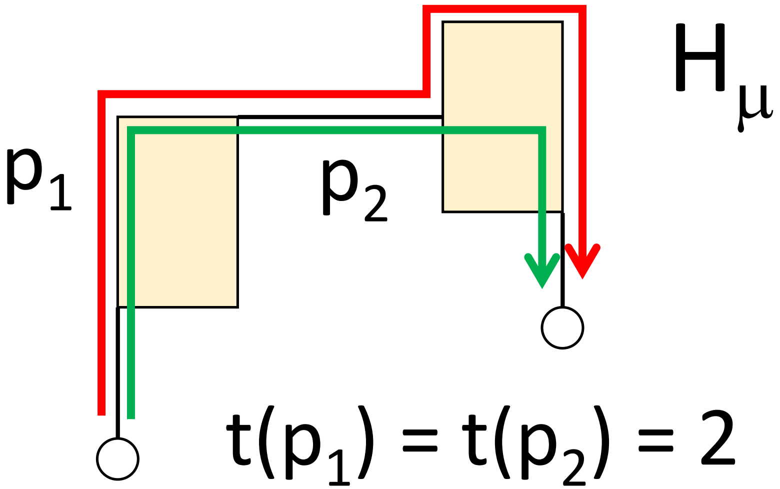




Inner S-components: spirality

μ = inner S-node

Lemma. All paths between the poles of an orthogonal component H_μ have the same turn number

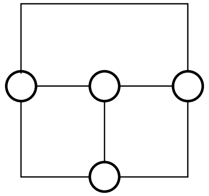


$t(p_1) = t(p_2) = 2$

$t(p) = k$

H_μ is k-spiral

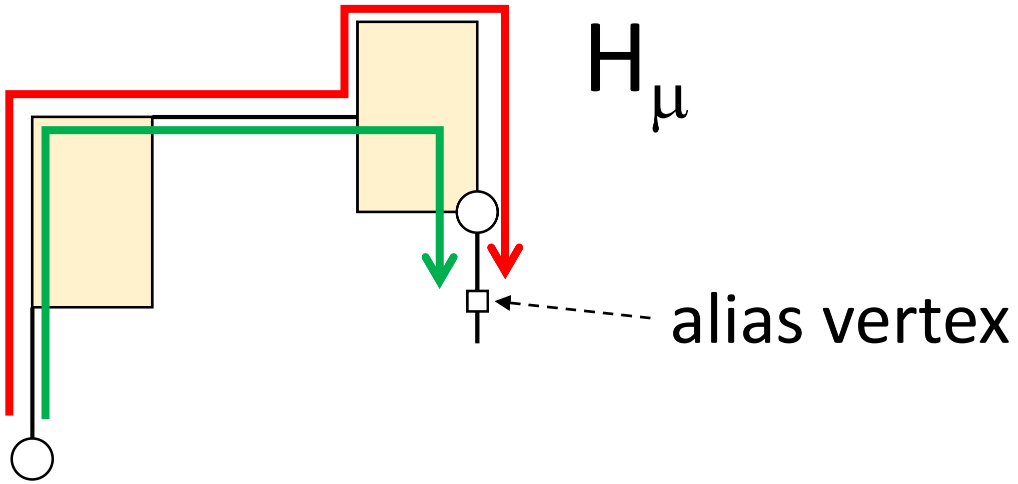
H_μ has spirality k

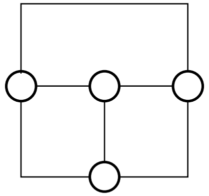


Root child S-components: spirality

μ = root child S-node

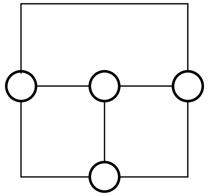
The definition of k-spiral and the lemma are extended by considering an external alias vertex in place of a pole with in-degree 2





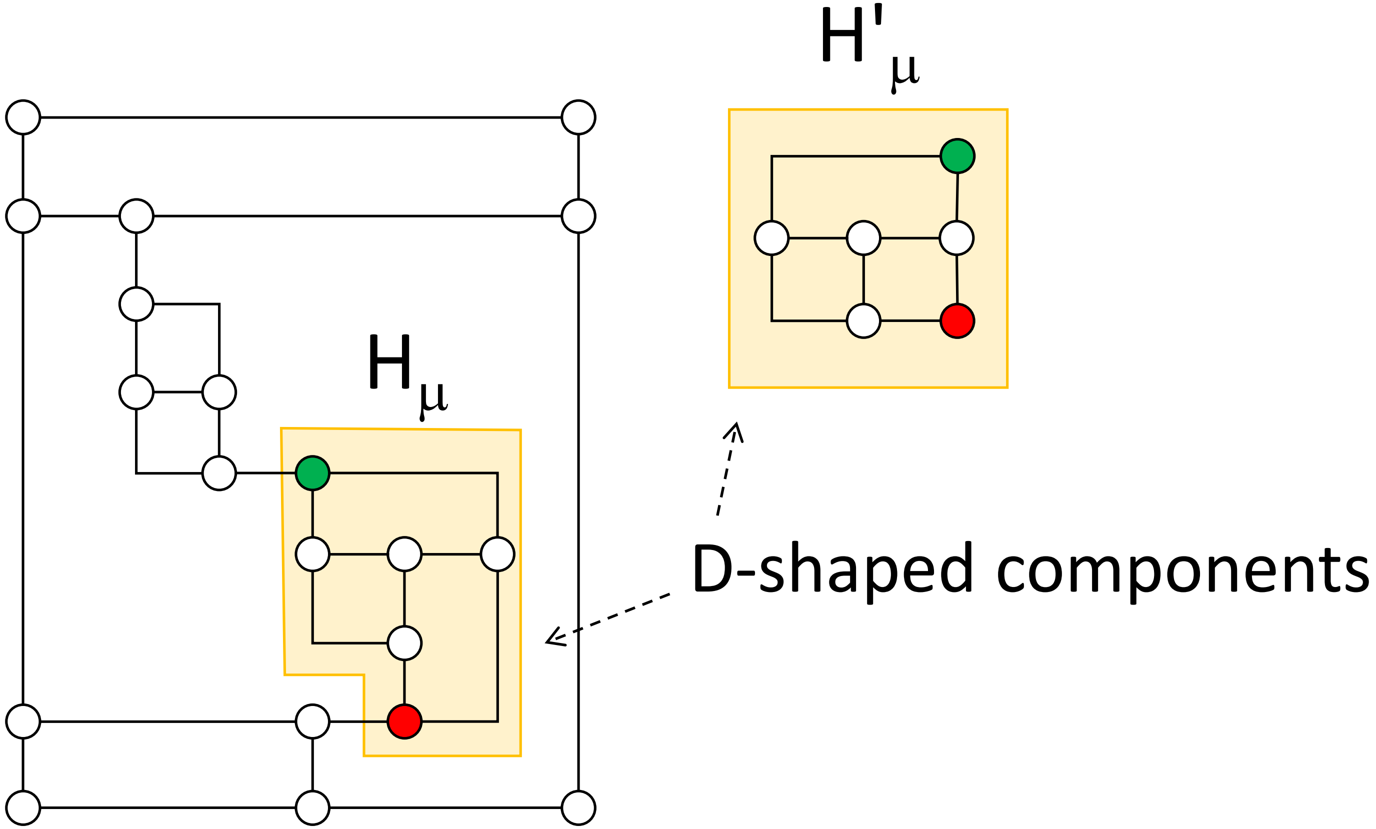
Equivalent orthogonal components

- H_μ and H'_μ = two distinct orthogonal representations of G_μ
- H_μ and H'_μ are equivalent if:
  - μ is a P- or an R-node and H_μ, H'_μ have the same representative shape
  - μ is an S-node and H_μ, H'_μ have the same spirality



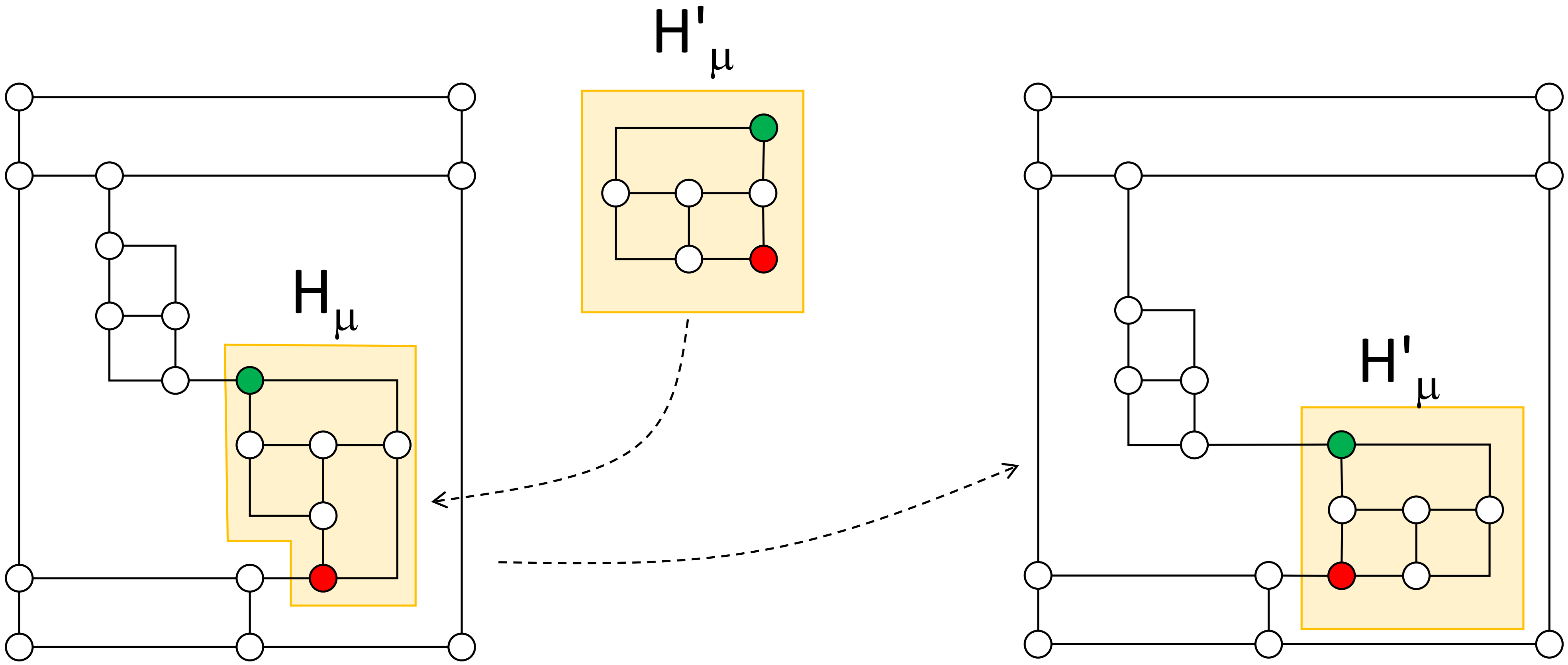
Equivalent orthogonal components

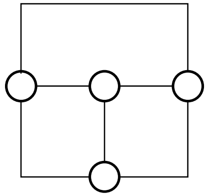
Theorem (substitution). Equivalent orthogonal components are interchangeable



Equivalent orthogonal components

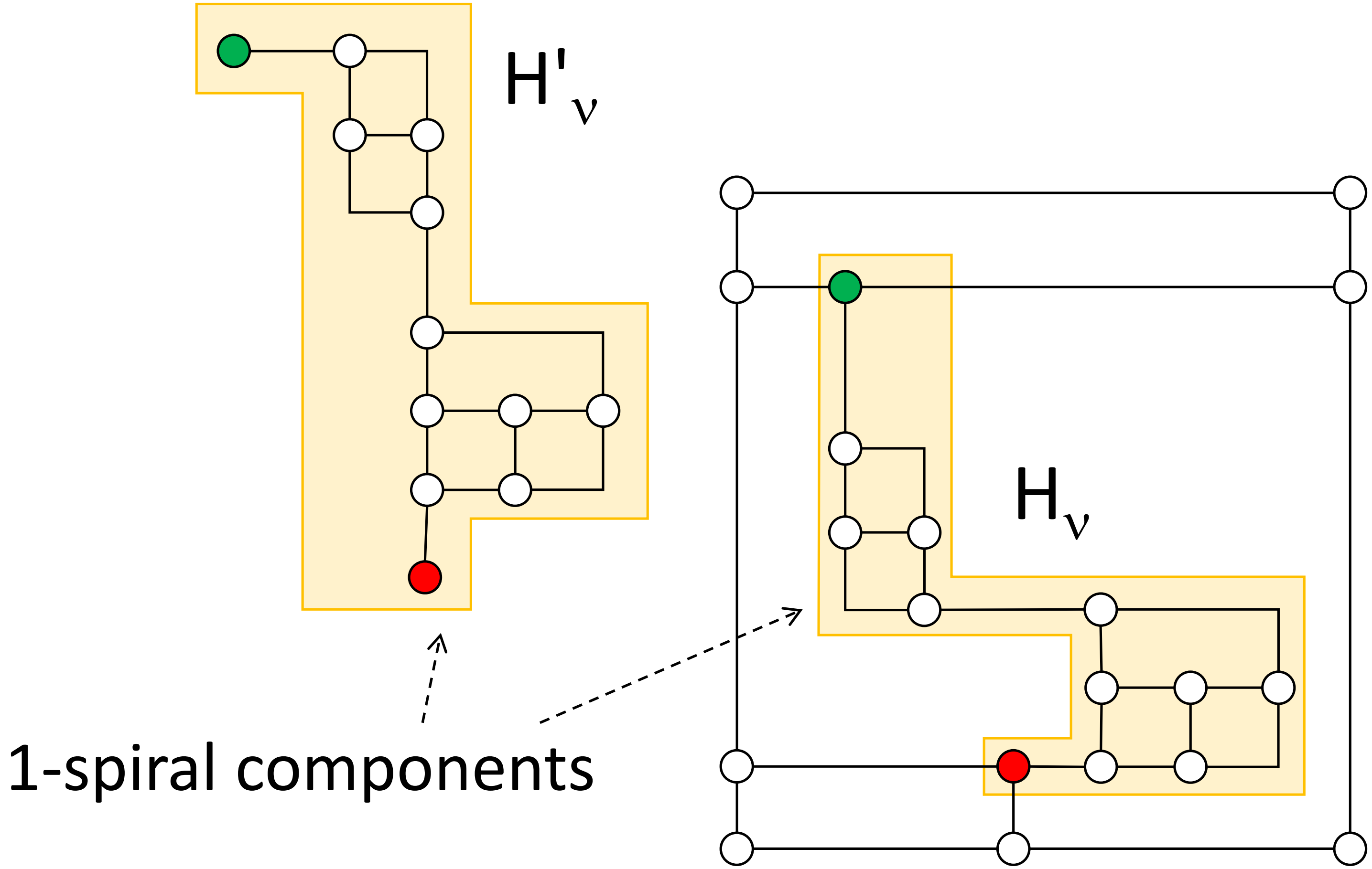
Theorem (substitution). Equivalent orthogonal components are interchangeable

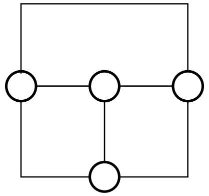




Equivalent orthogonal components

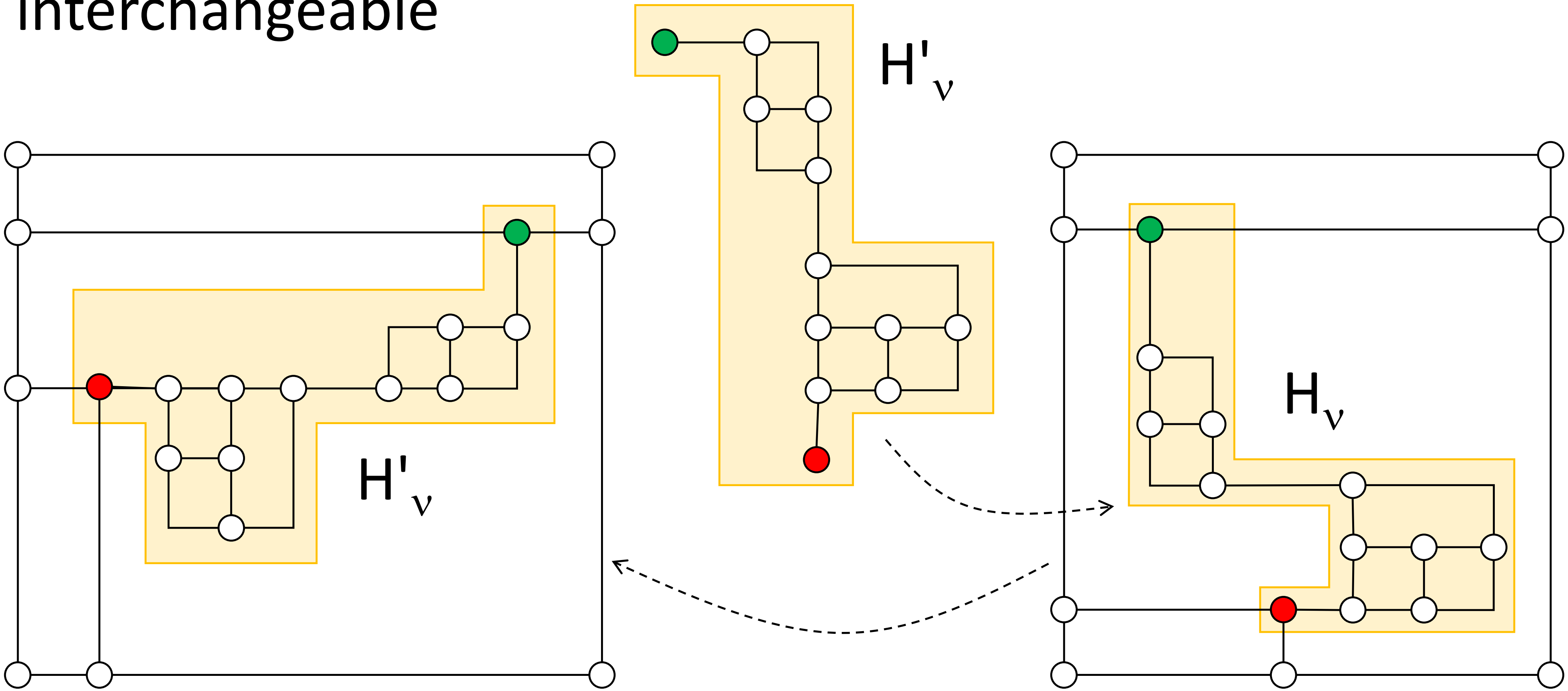
Theorem (substitution). Equivalent orthogonal components are interchangeable





Equivalent orthogonal components

Theorem (substitution). Equivalent orthogonal components are interchangeable



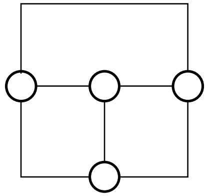
Key lemma

Key-Lemma. Every biconnected planar 3-graph with a given edge *e* admits a bend-min orthogonal representation with *e* on the external face such that:

O1. every edge has at most two bends

O2. every inner P- or R-component is D- or X-shaped; if the root child is a P- or an R-component, it is either D-, C-, or L-shaped

O3. every S-component has spirality at most 4



Key lemma

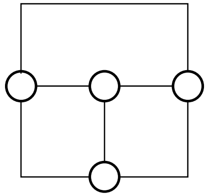
Key-Lemma. Every biconnected planar 3-graph with a given edge *e* admits a bend-min orthogonal representation with *e* on the external face such that:

O1. every edge has at most two bends

O2. every inner P- or R-component is D- or X-shaped; if the root child is a P- or an R-component, it is either D-, C-, or L-shaped

O3. every S-component has spirality at most 4

Proof ingredients: partially based on a characterization of no-bend orthogonal representations [Rahman, Nishizeki, Naznin 2003]



Key lemma: Consequence

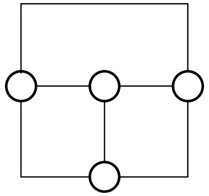
Key-Lemma. Every biconnected planar 3-graph with a given edge *e* admits a bend-min orthogonal representation with *e* on the external face such that:

O1. every edge has at most two bends

O2. every inner P- or R-component is D- or X-shaped; if the root child is a P- or an R-component, it is either D-, C-, or L-shaped

O3. every S-component has spirality at most 4

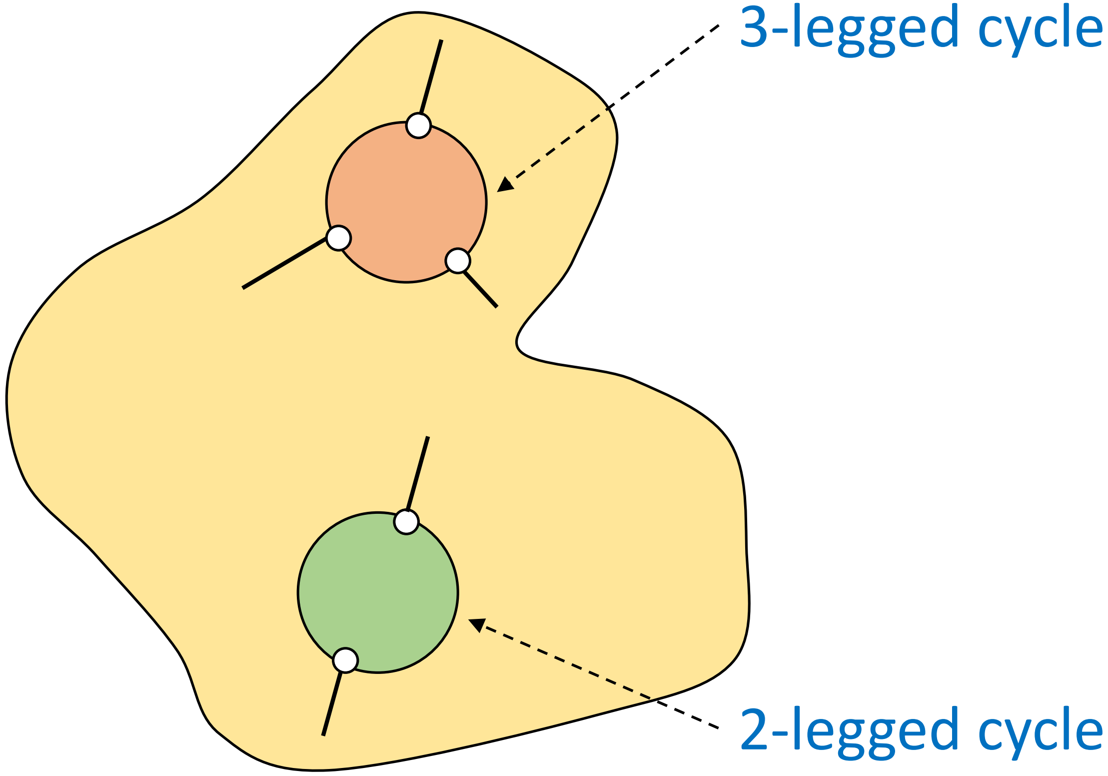
Consequence: we can restrict our algorithm to search for a bend-min representation that satisfies O1, O2, and O3.



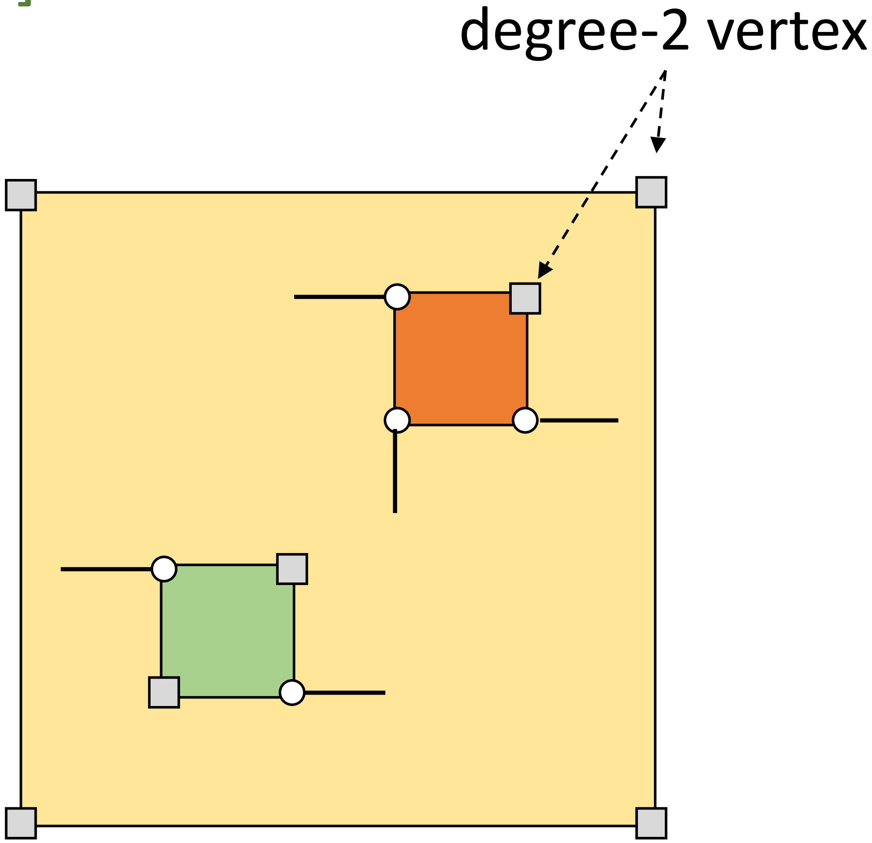
\begin{Characterization of no-bend drawings}

Characterization of no-bend drawings

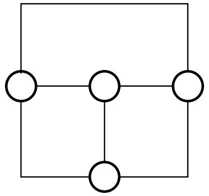
[Rahman, Nishizeki, Naznin, JGAA 2003] = [RNN'03]



biconnected plane 3-graph



no-bend orthogonal drawing of G



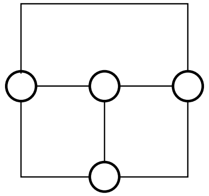
Characterization of no-bend drawings

Theorem [RNN'03]. Let G be a biconnected plane 3-graph. G admits a **no-bend** orthogonal drawing ⇔

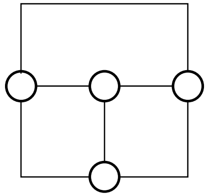
(i) the external cycle of G has at least 4 degree-2 vertices

(ii) each k-legged cycle of G has at least (4-k) degree-2 vertices

Definition: we call **bad** a 2-legged or a 3-legged cycle that does not satisfy (ii)



\end{Characterization of no-bend drawings}



Key-Lemma: O1

Key-Lemma. Let G be a biconnected planar 3-graph with a given edge *e*; G admits a bend-min orthogonal representation with *e* on the external face and having these properties:

O1. at most two bends per edge

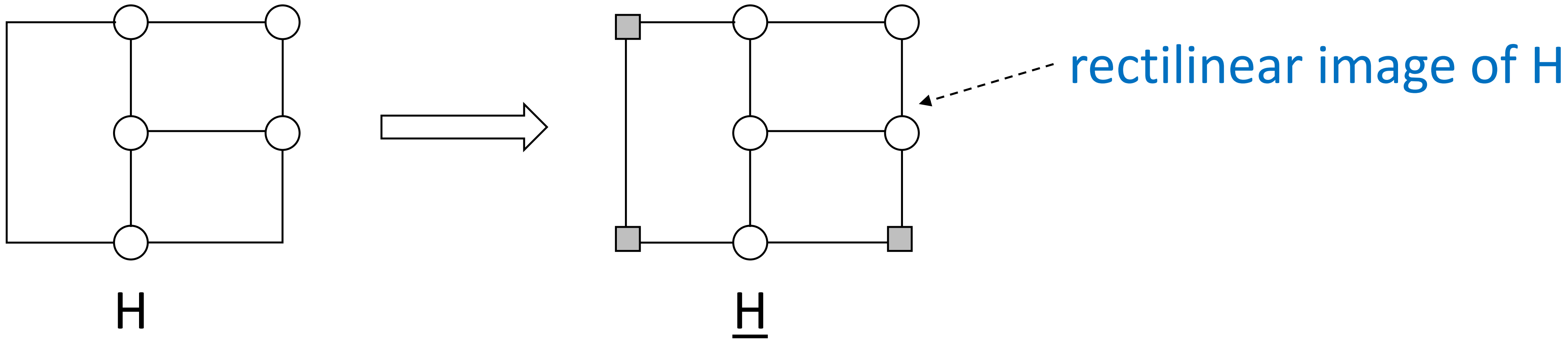
O2. every inner P- or R-component is D- or X-shaped; if the root child is a P- or an R-component, it is either D-, C-, or L-shaped

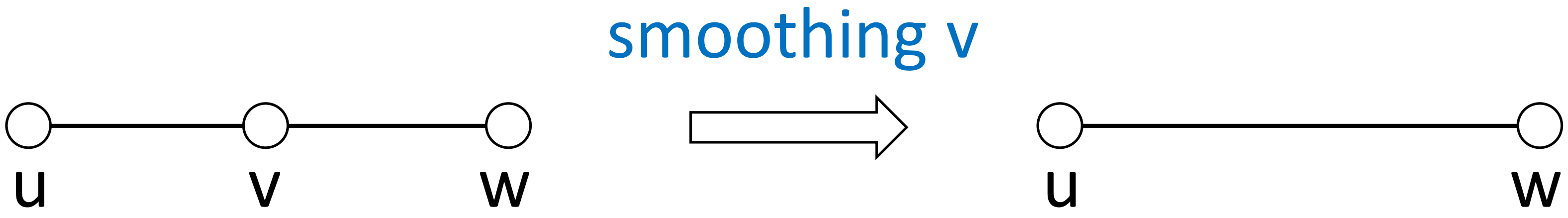
O3. every S-component has spirality at most 4

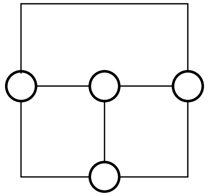
Key-Lemma: O1

Proof of O1 (at most two bends per edge)

Notation



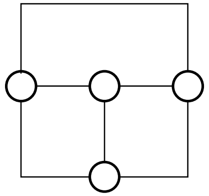




Key-Lemma: O1

Proof of O1 (at most two bends per edge)

- H = bend-min representation of G with *e* on the external face
- g = edge of H with (at least) three bends



Key-Lemma: O1

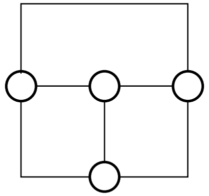
Proof of O1 (at most two bends per edge)

- H = bend-min representation of G with *e* on the external face
- g = edge of H with (at least) three bends
- v1, v2, v3 = the three bend-vertices of $\underline{H}$ corresponding to the bends of g

Key-Lemma: O1

Proof of O1 (at most two bends per edge)

- H = bend-min representation of G with *e* on the external face
- g = edge of H with (at least) three bends
- v1, v2, v3 = the three bend-vertices of $\underline{H}$ corresponding to the bends of g
- $\underline{H}$ has no-bend $\Rightarrow$ $\underline{G}$ satisfies (i) and (ii) of Th. [RNN'03]



Key-Lemma: O1

Proof of O1 (at most two bends per edge)

- H = bend-min representation of G with *e* on the external face
- g = edge of H with (at least) three bends
- v1, v2, v3 = the three bend-vertices of $\underline{H}$ corresponding to the bends of g
- $\underline{H}$ has no-bend $\Rightarrow$ $\underline{G}$ satisfies (i) and (ii) of Th. [RNN'03]

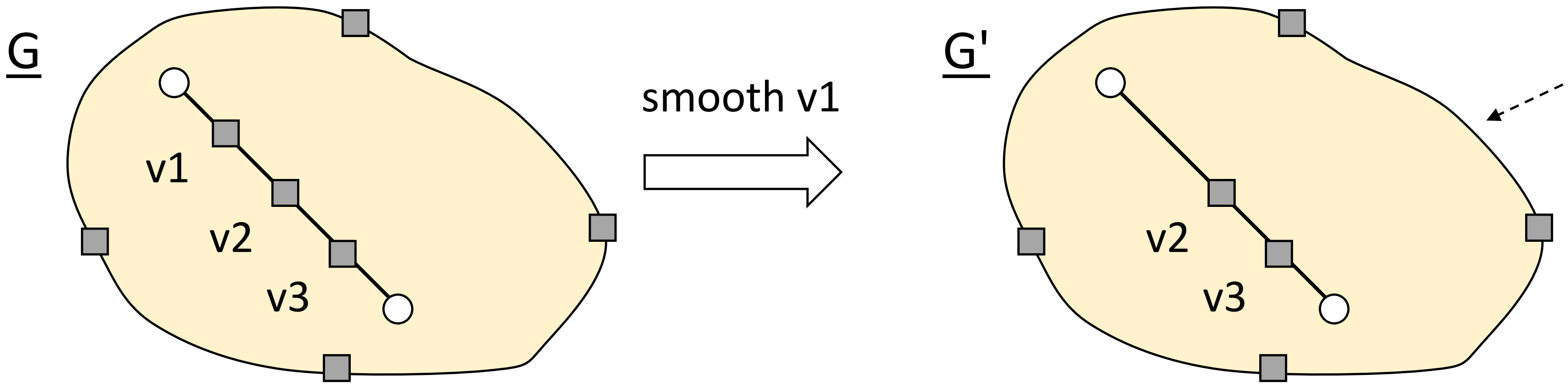
Case 1: g is an internal edge

Key-Lemma: O1

Proof of O1 (at most two bends per edge)

- H = bend-min representation of G with *e* on the external face
- g = edge of H with (at least) three bends
- v1, v2, v3 = the three bend-vertices of $\underline{H}$ corresponding to the bends of g
- $\underline{H}$ has no-bend $\Rightarrow$ $\underline{G}$ satisfies (i) and (ii) of Th. [RNN'03]

Case 1: g is an internal edge



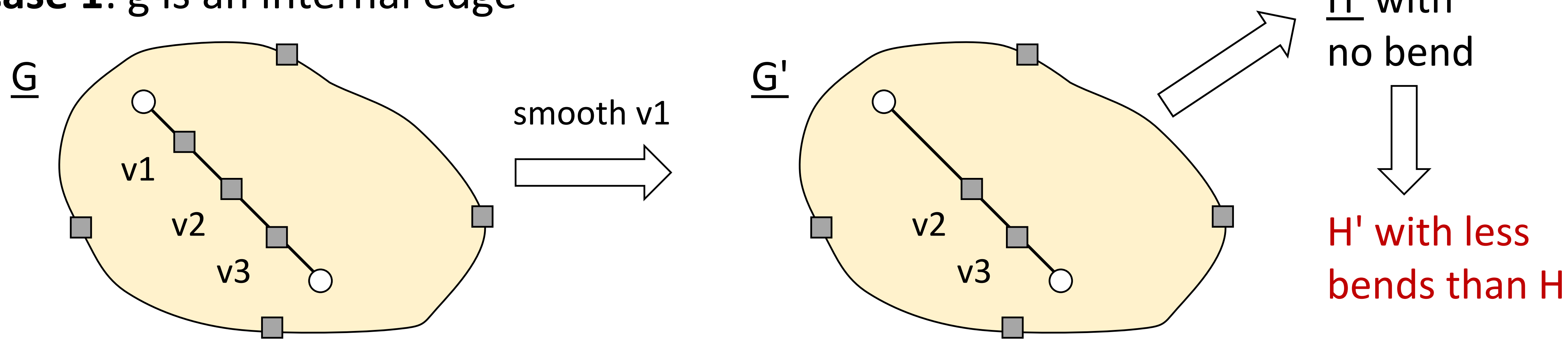
still satisfies (i) and (ii) of Th. [RNN'03]

Key-Lemma: O1

Proof of O1 (at most two bends per edge)

- H = bend-min representation of G with *e* on the external face
- g = edge of H with (at least) three bends
- v1, v2, v3 = the three bend-vertices of $\underline{H}$ corresponding to the bends of g
- $\underline{H}$ has no-bend $\Rightarrow$ $\underline{G}$ satisfies (i) and (ii) of Th. [RNN'03]

Case 1: g is an internal edge



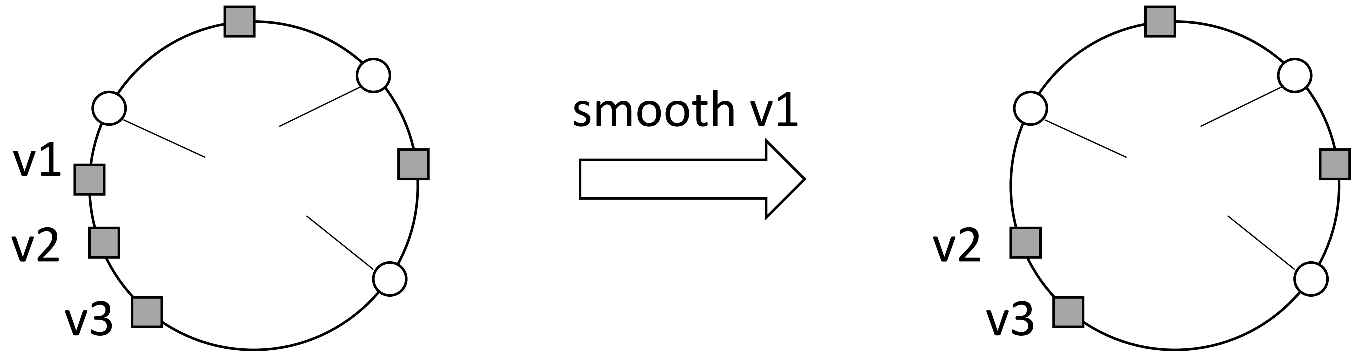
Key-Lemma: O1

Proof of O1 (at most two bends per edge)

- H = bend-min representation of G with *e* on the external face
- g = edge of H with (at least) three bends
- v1, v2, v3 = the three bend-vertices of $\underline{H}$ corresponding to the bends of g
- $\underline{H}$ has no-bend $\Rightarrow$ $\underline{G}$ satisfies (i) and (ii) of Th. [RNN'03]

Case 2: g is an external edge (call $C_0(G)$ the external boundary of G)

- **Case 2.1.** $C_0(\underline{G})$ has more than 4 degree-2 vertices



contradiction
as before

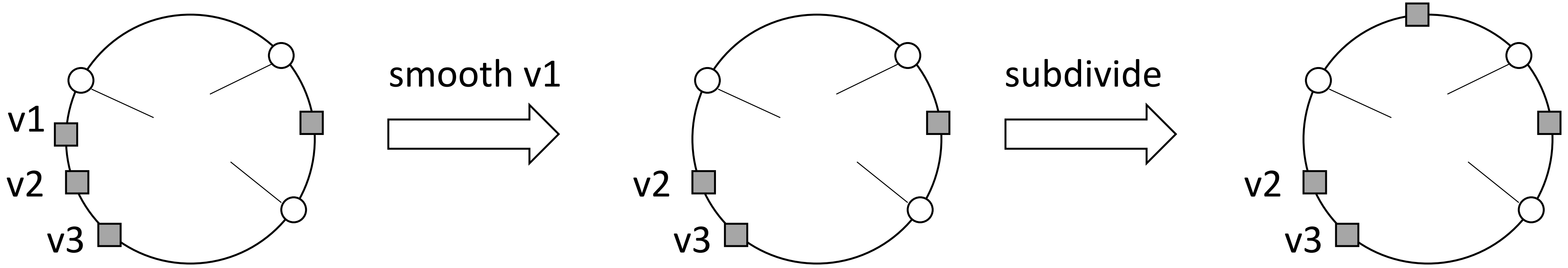
Key-Lemma: O1

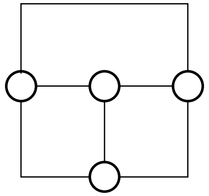
Proof of O1 (at most two bends per edge)

- H = bend-min representation of G with *e* on the external face
- g = edge of H with (at least) three bends
- v1, v2, v3 = the three bend-vertices of $\underline{H}$ corresponding to the bends of g
- $\underline{H}$ has no-bend $\Rightarrow$ $\underline{G}$ satisfies (i) and (ii) of Th. [RNN'03]

Case 2: g is an external edge (call $C_0(G)$ the external boundary of G)

- **Case 2.2.** $C_0(\underline{G})$ has exactly 4 degree-2 vertices





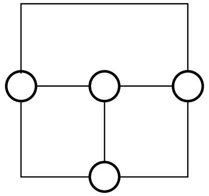
Key-Lemma: O2

Key-Lemma. Let G be a biconnected planar 3-graph with a given edge *e*; G admits a bend-min orthogonal representation with *e* on the external face and having these properties:

O1. at most two bends per edge

O2. every inner P- or R-component is D- or X-shaped; if the root child is a P- or an R-component, it is either D-, C-, or L-shaped

O3. every S-component has spirality at most 4



Key-Lemma: O2

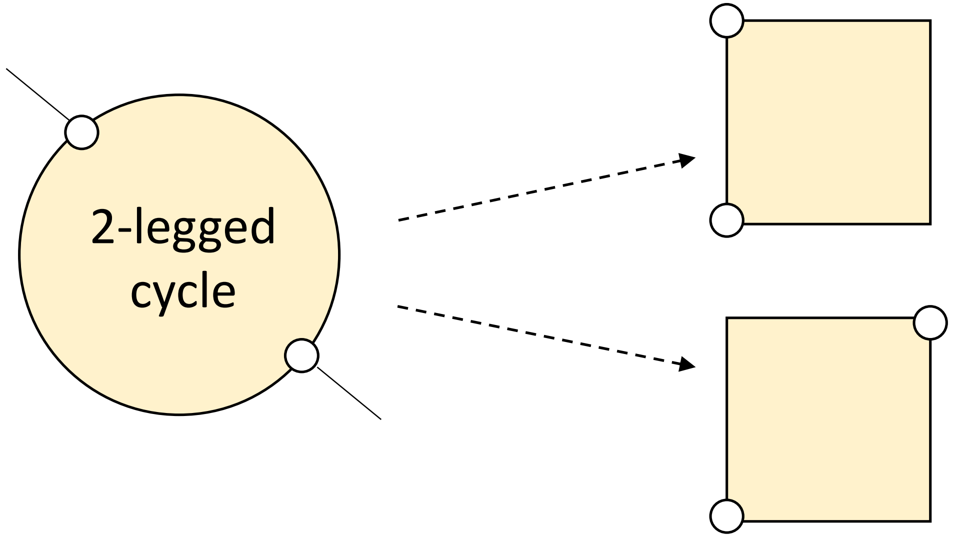
Proof of O2 (inner P- or R-components are D- or X-shaped)

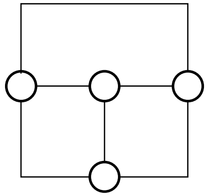
- H = bend-min representation of G with *e* on the external face and property O1
- $\underline{H}$ has no-bend $\Rightarrow$ $\underline{G}$ satisfies (i) and (ii) of Th. [RNN'03]

Key-Lemma: O2

Proof of O2 (inner P- or R-components are D- or X-shaped)

- H = bend-min representation of G with *e* on the external face and property O1
- H has no-bend ⇒ G satisfies (i) and (ii) of Th. [RNN'03]
- [RNN'03] gives an algorithm that computes a no-bend representation H' of G such that every 2-legged (and 3-legged) cycle is either D-shaped or X-shaped

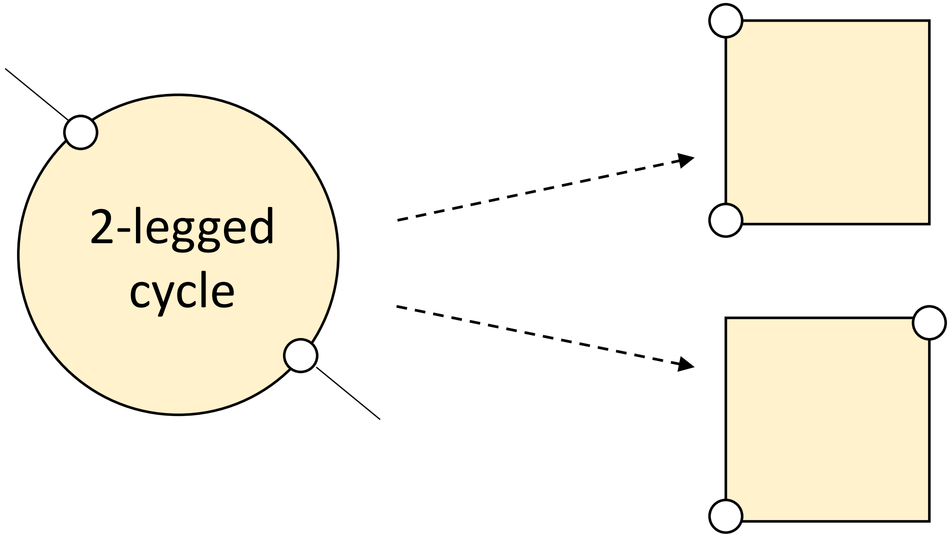




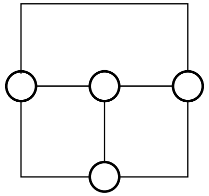
Key-Lemma: O2

Proof of O2 (inner P- or R-components are D- or X-shaped)

- H = bend-min representation of G with *e* on the external face and property O1
- H has no-bend ⇒ G satisfies (i) and (ii) of Th. [RNN'03]
- [RNN'03] gives an algorithm that computes a no-bend representation H' of G such that every 2-legged (and 3-legged) cycle is either D-shaped or X-shaped



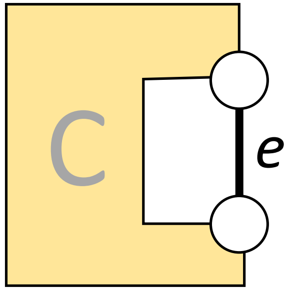
… each inner P- and R-component is a 2-legged cycle in G



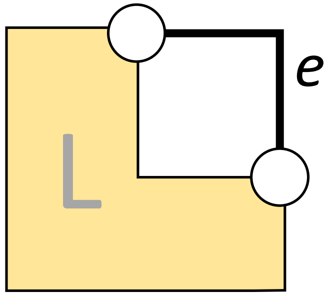
Key-Lemma: O2

Proof of O2 (root child P- or R-components are D-, C-, or L-shaped)

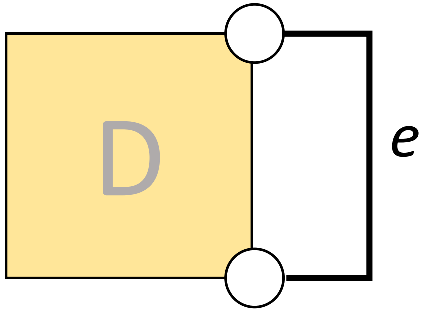
- H = bend-min representation of G with *e* on the external face and property O1



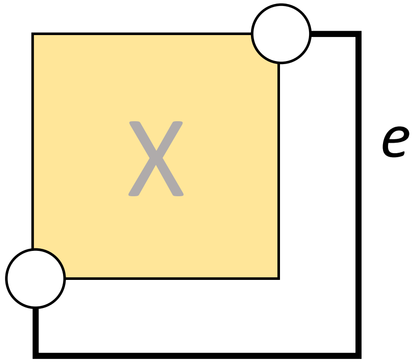
e has 0 bends



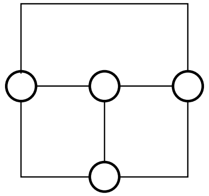
e has 1 bend



e has 2 bends



e has 3 bends



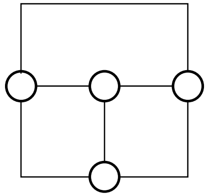
Key-Lemma: O3

Key-Lemma. Let G be a biconnected planar 3-graph with a given edge *e*; G admits a bend-min orthogonal representation with *e* on the external face and having these properties:

O1. at most two bends per edge

O2. every inner P- or R-component is D- or X-shaped; if the root child is a P- or an R-component, it is either D-, C-, or L-shaped

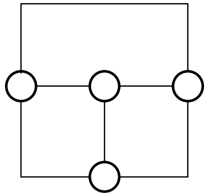
O3. every S-component has spirality at most 4



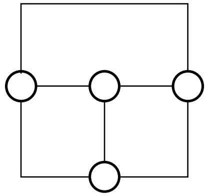
Key-Lemma: O3

Proof of O3 (S-components have spirality at most 4)

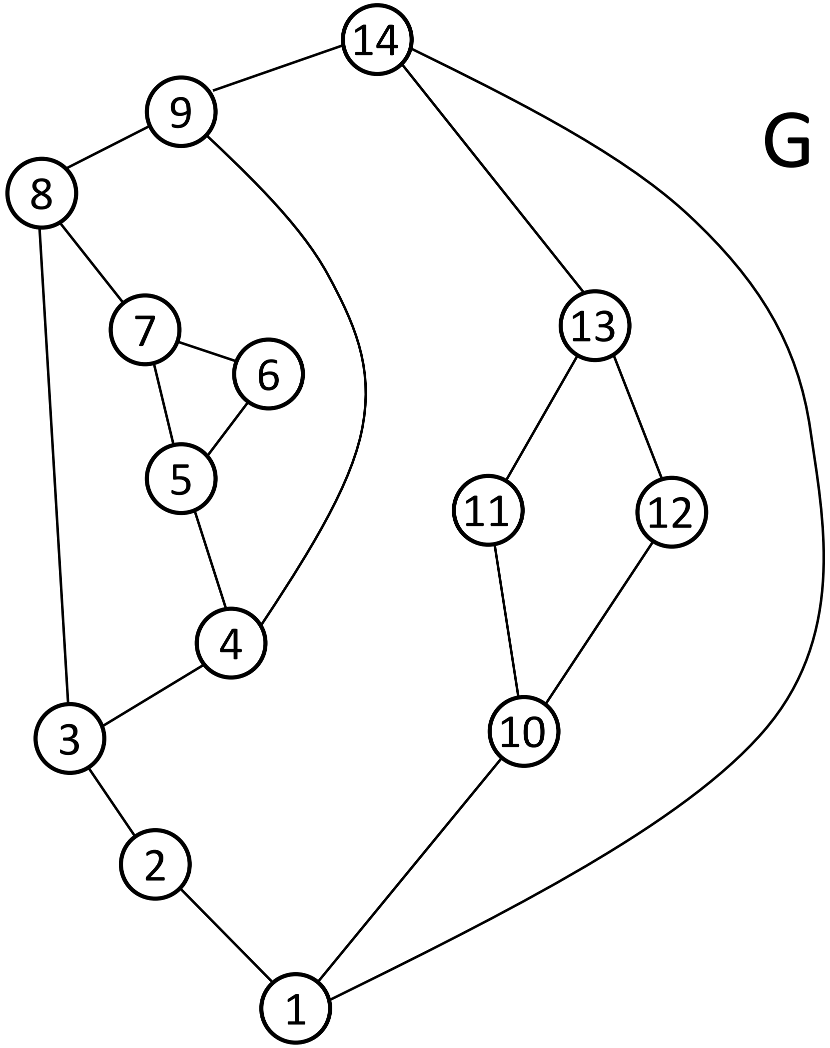
- H = bend-min representation of G with *e* on the external face and properties O1 and O2;
- $\underline{H}$ was computed with the [RNN'03] alg, which we call NoBend-Alg
- we prove that every S-component in $\underline{H}$ (and thus in H) has spirality at most 4



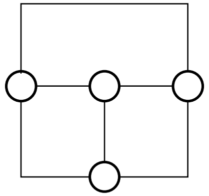
\begin{NoBend-Alg}



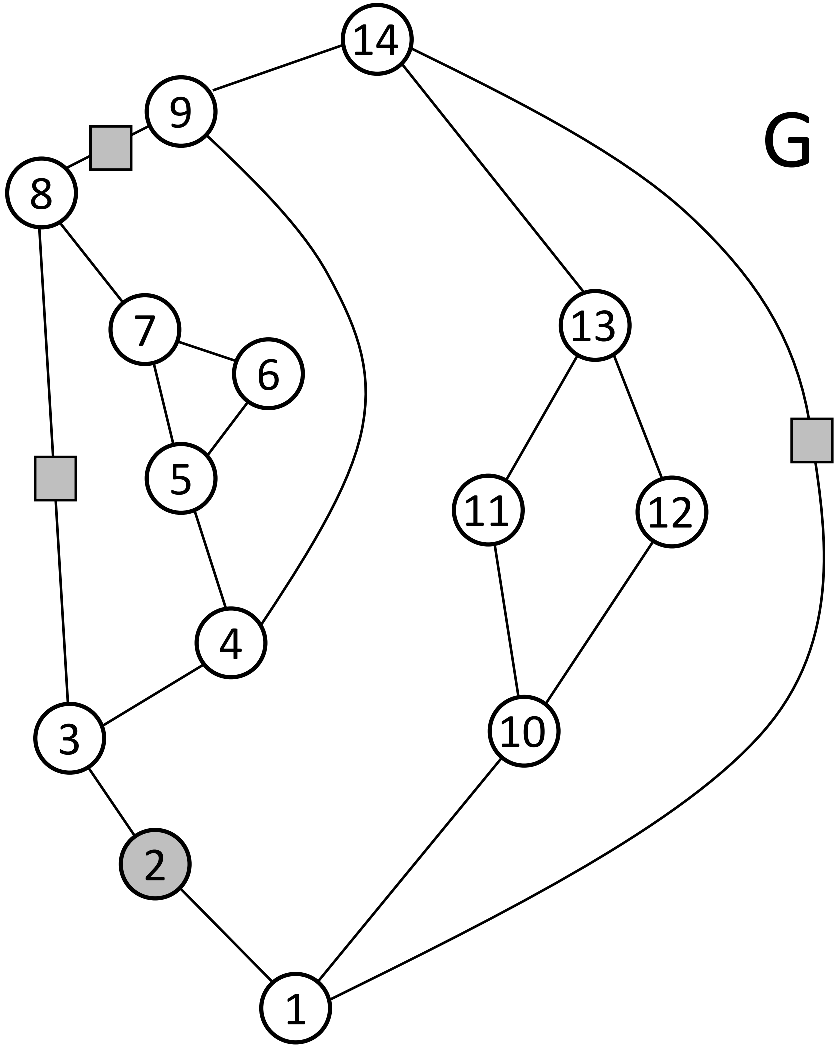
Step 1: choose 4 external corners



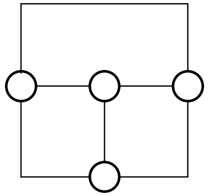
four vertices of degree 2 are used as corners (in our case, these vertices may be obtained by subdividing edges)



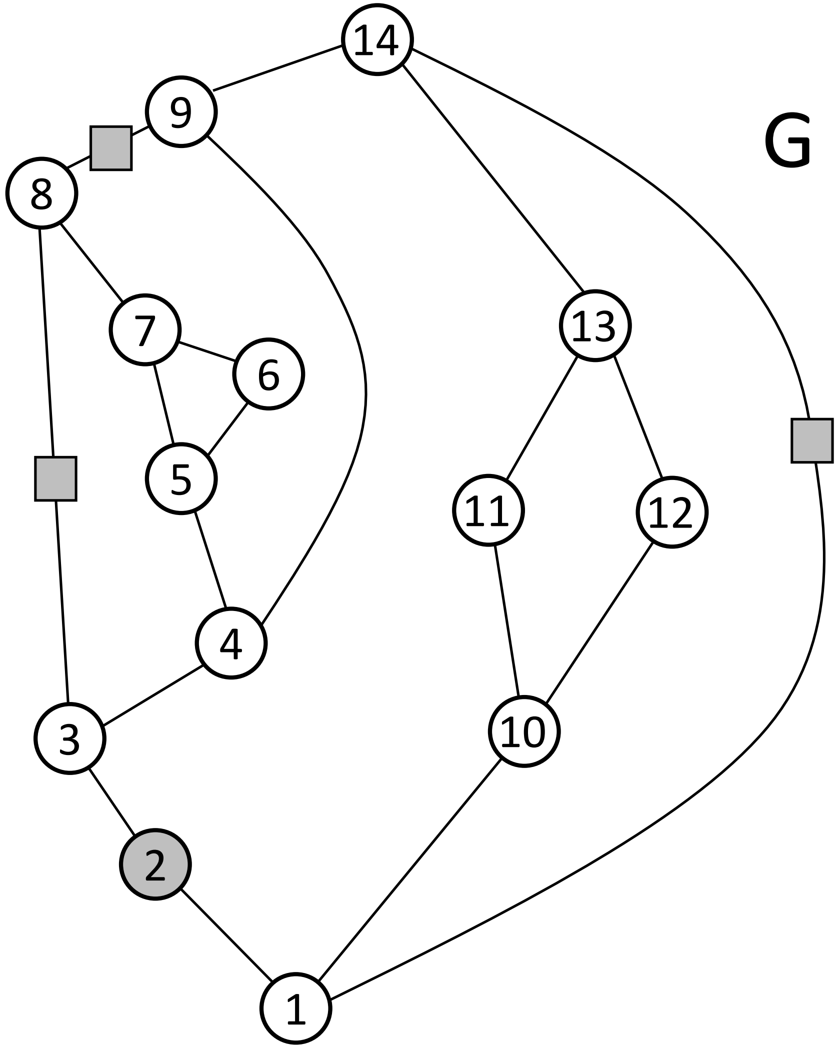
Step 1: choose 4 external corners



four vertices of degree 2 are used as corners (in our case, these vertices may be obtained by subdividing edges)

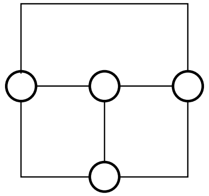


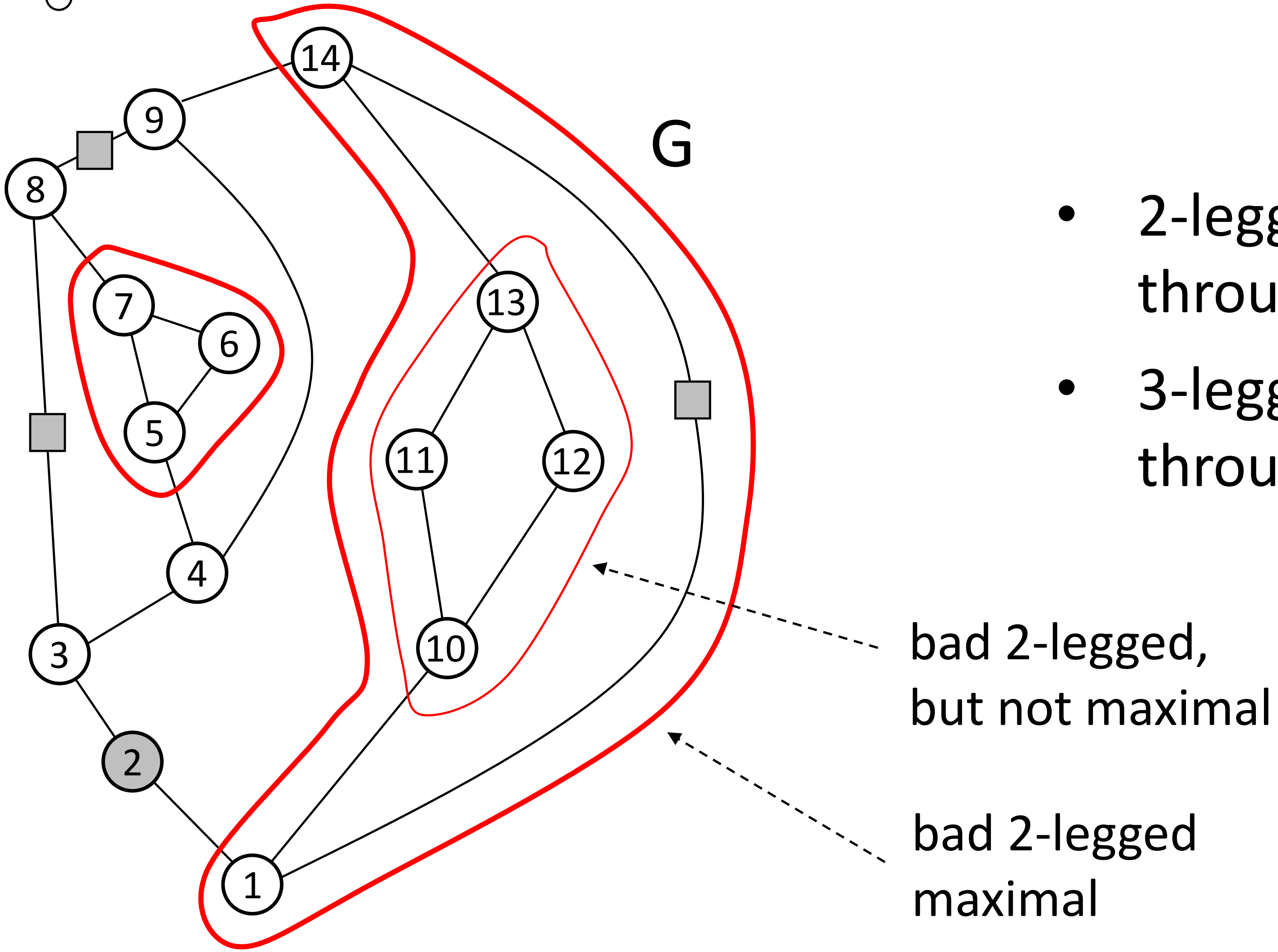
Step 2: find maximal bad cycles w.r.t. the corners



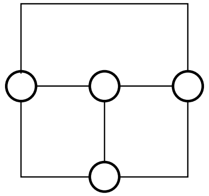
- 2-legged cycles not passing through (at least) 2 corners
- 3-legged cycles not passing through (at least) 1 corner

Step 2: find maximal bad cycles w.r.t. the corners

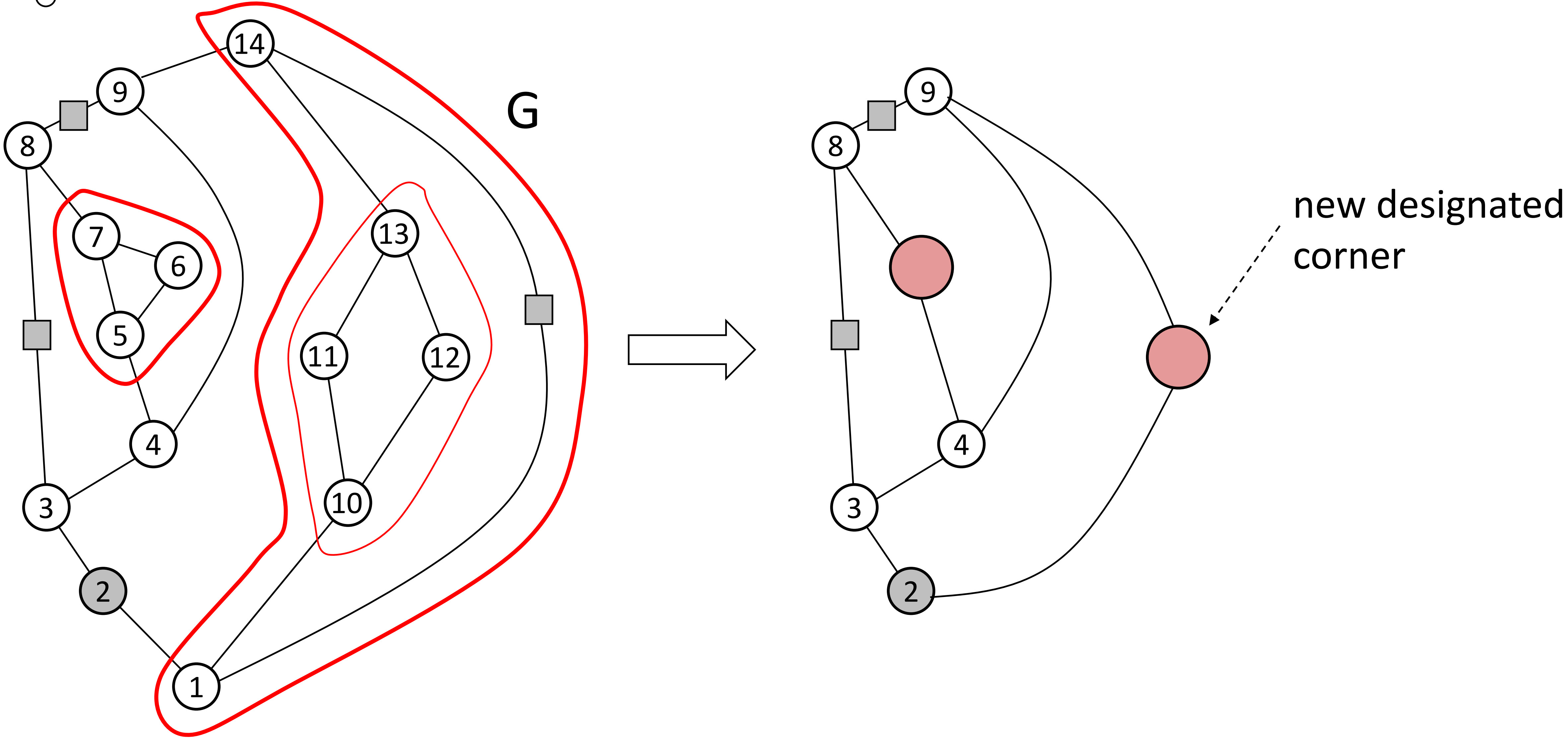


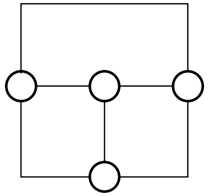


- 2-legged cycles not passing through (at least) 2 corners
- 3-legged cycles not passing through (at least) 1 corner

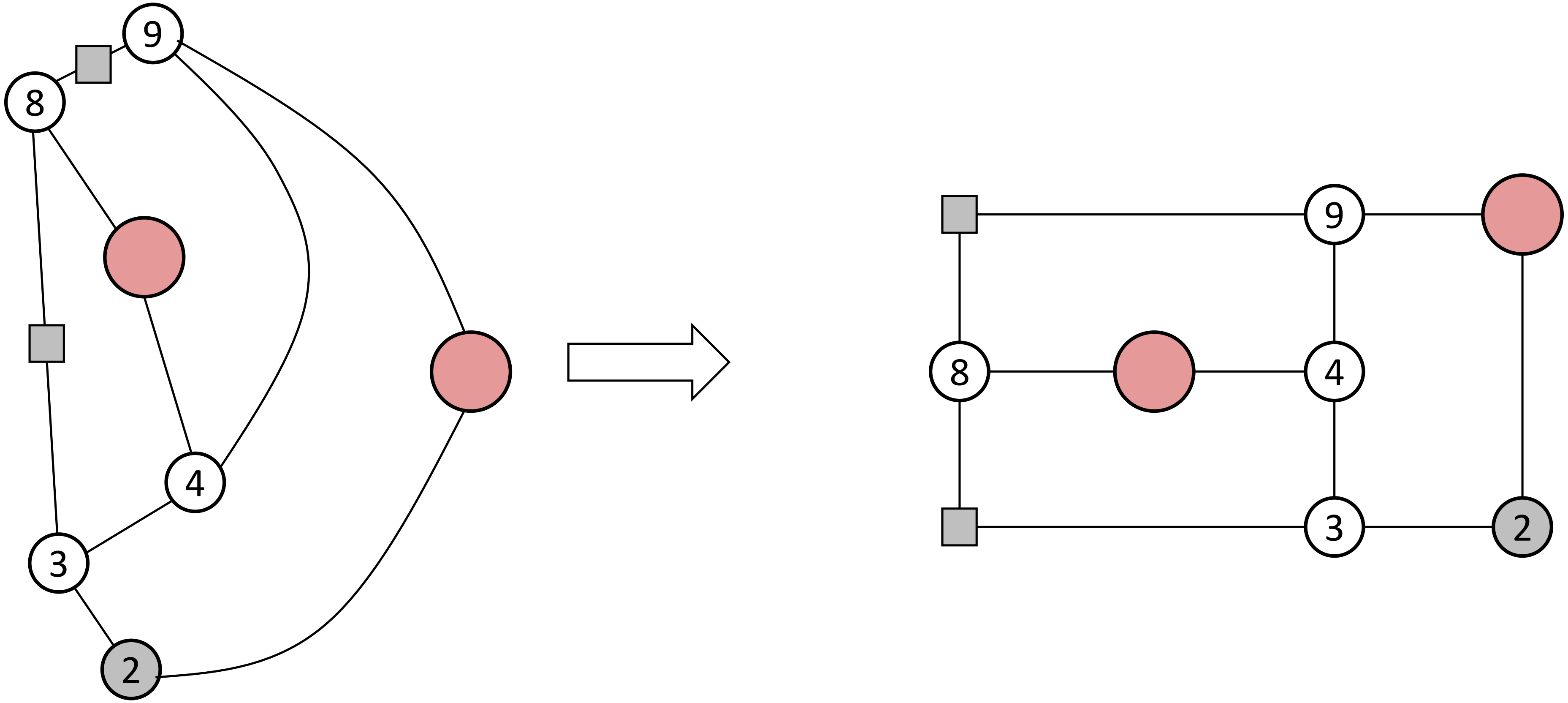


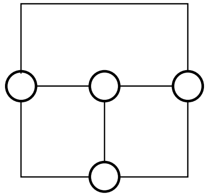
Step 3: collapse maximal bad cycles



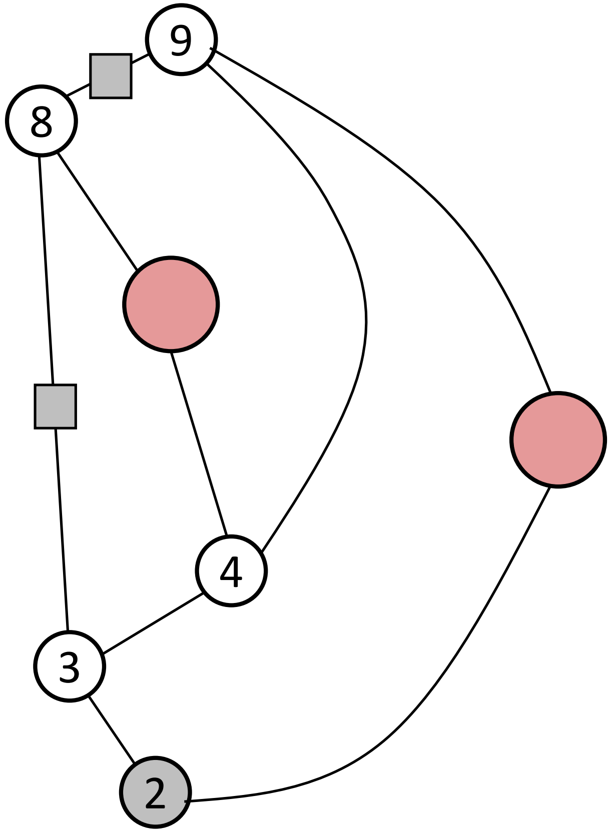


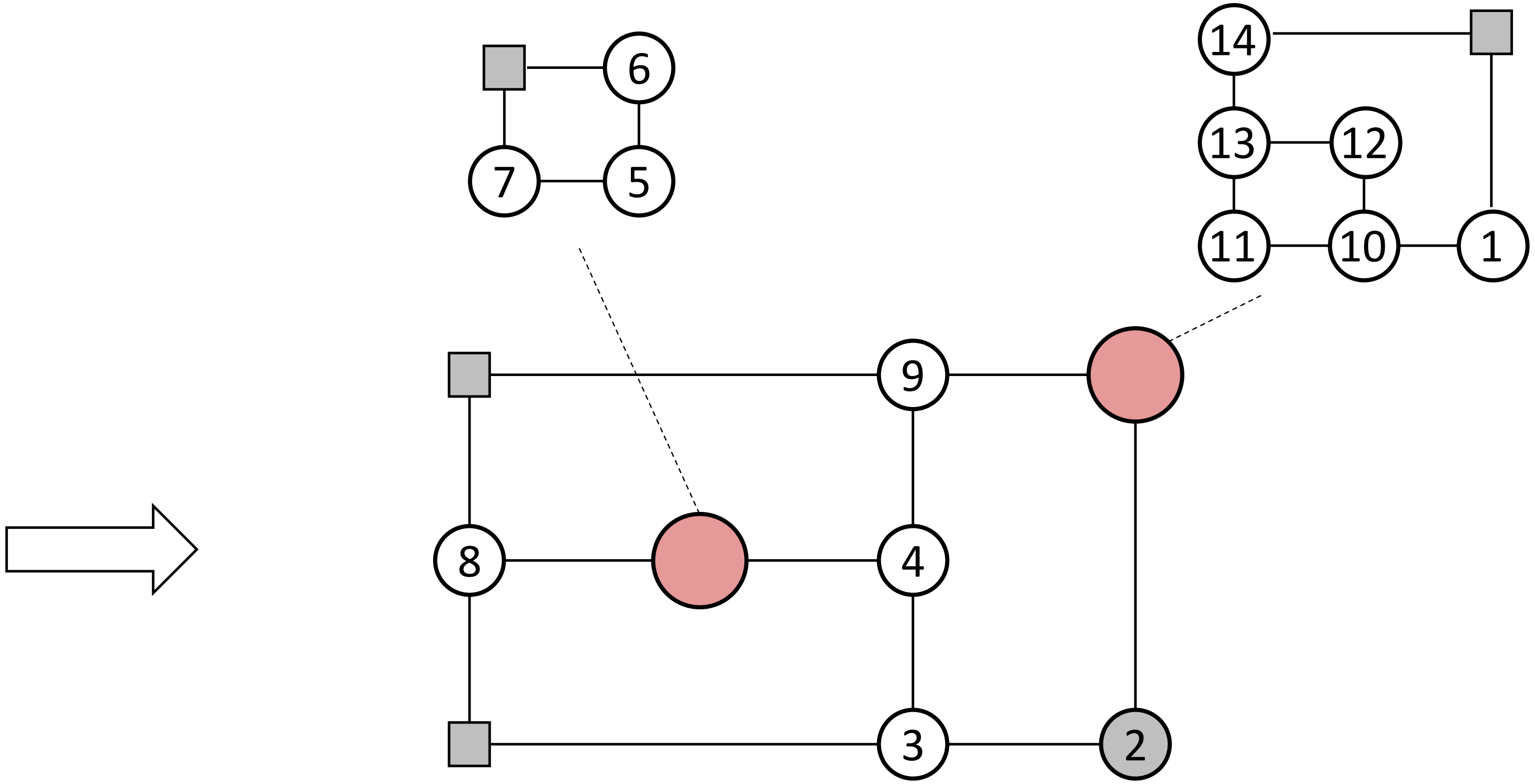
Step 4: compute a rectangular representation

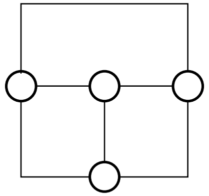




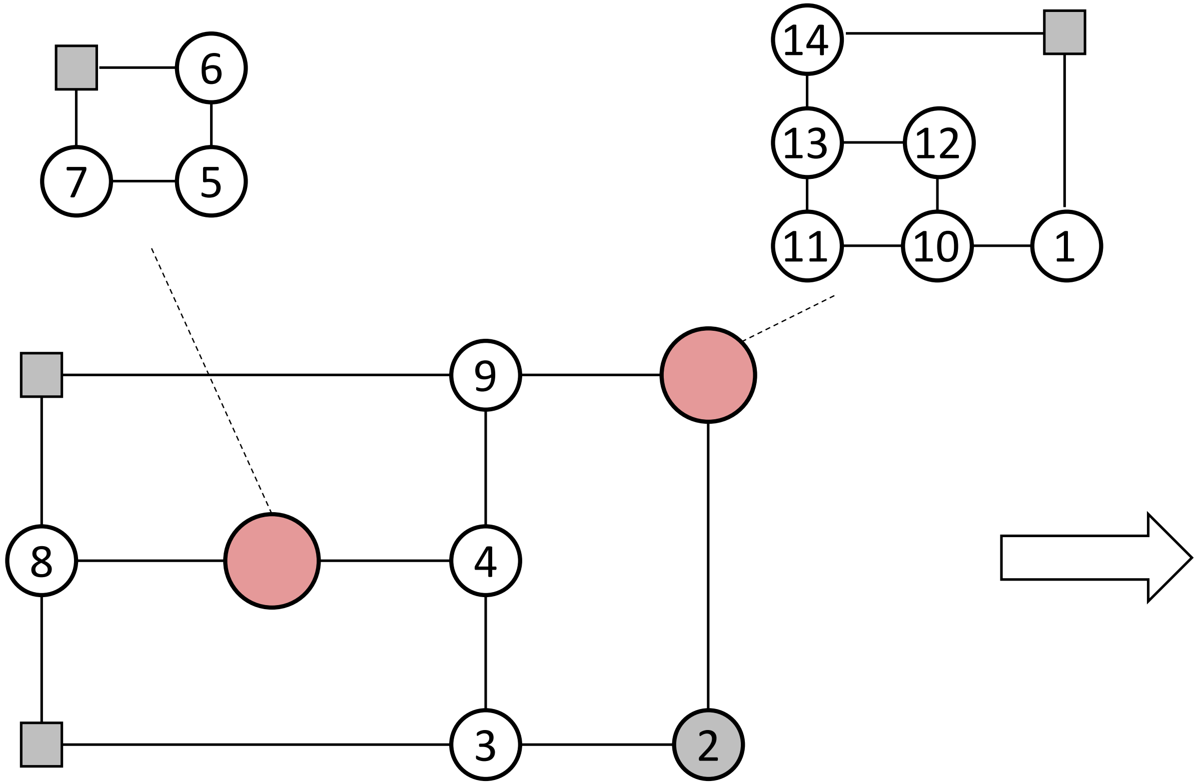
Step 5: recourse into the collapsed nodes

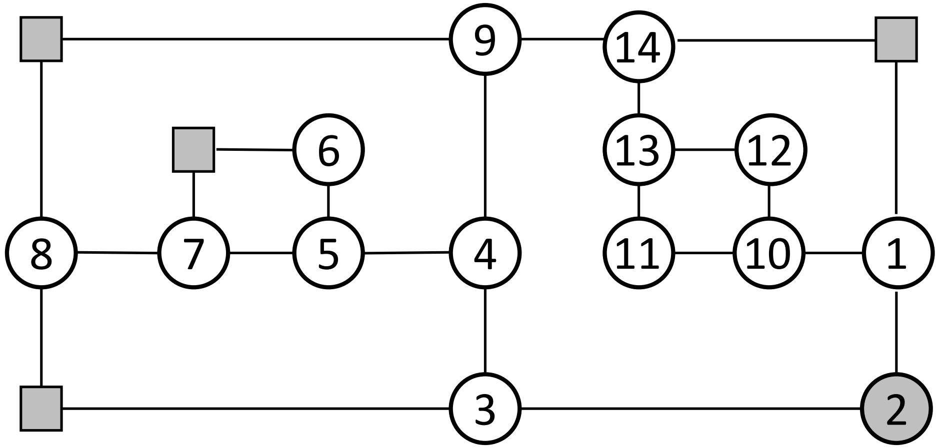


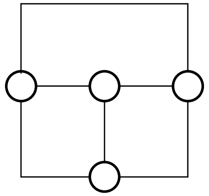




Step 6: … and plug the components





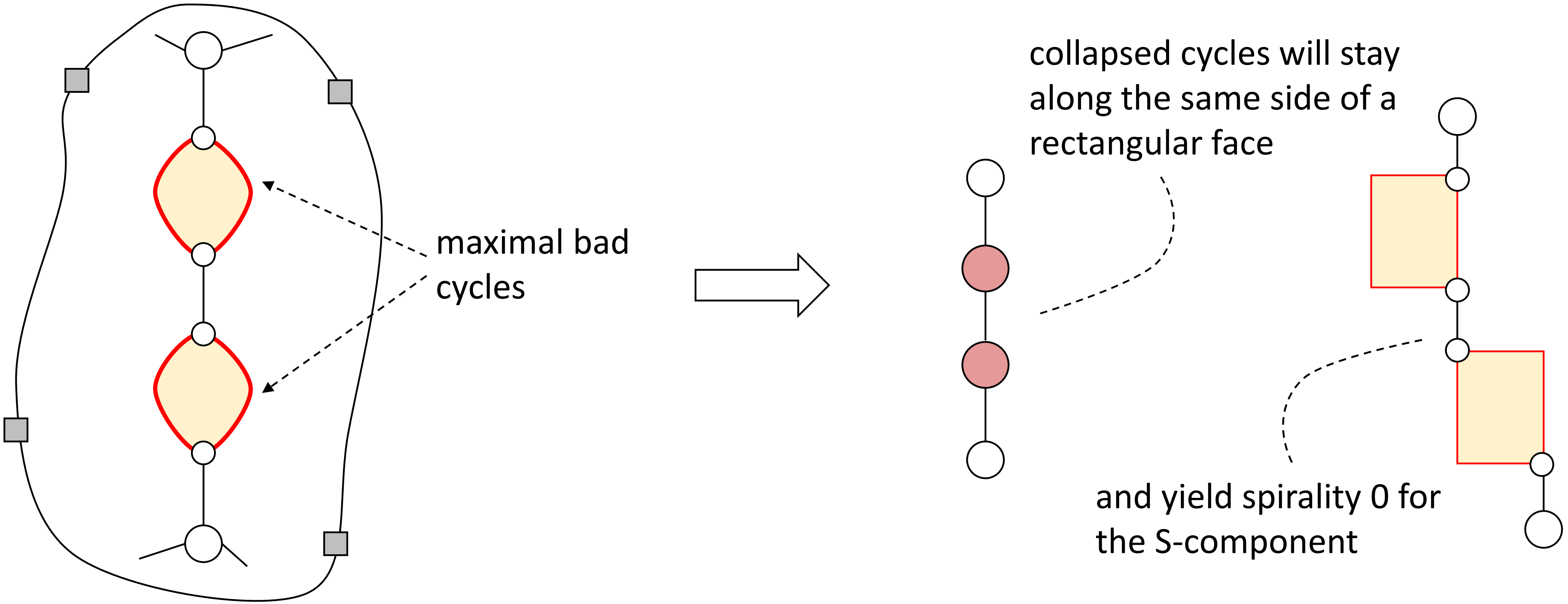


\end{NoBend-Alg}

Key-Lemma: O3

Proof of O3 (<u>inner</u> S-components have spirality at most 4)

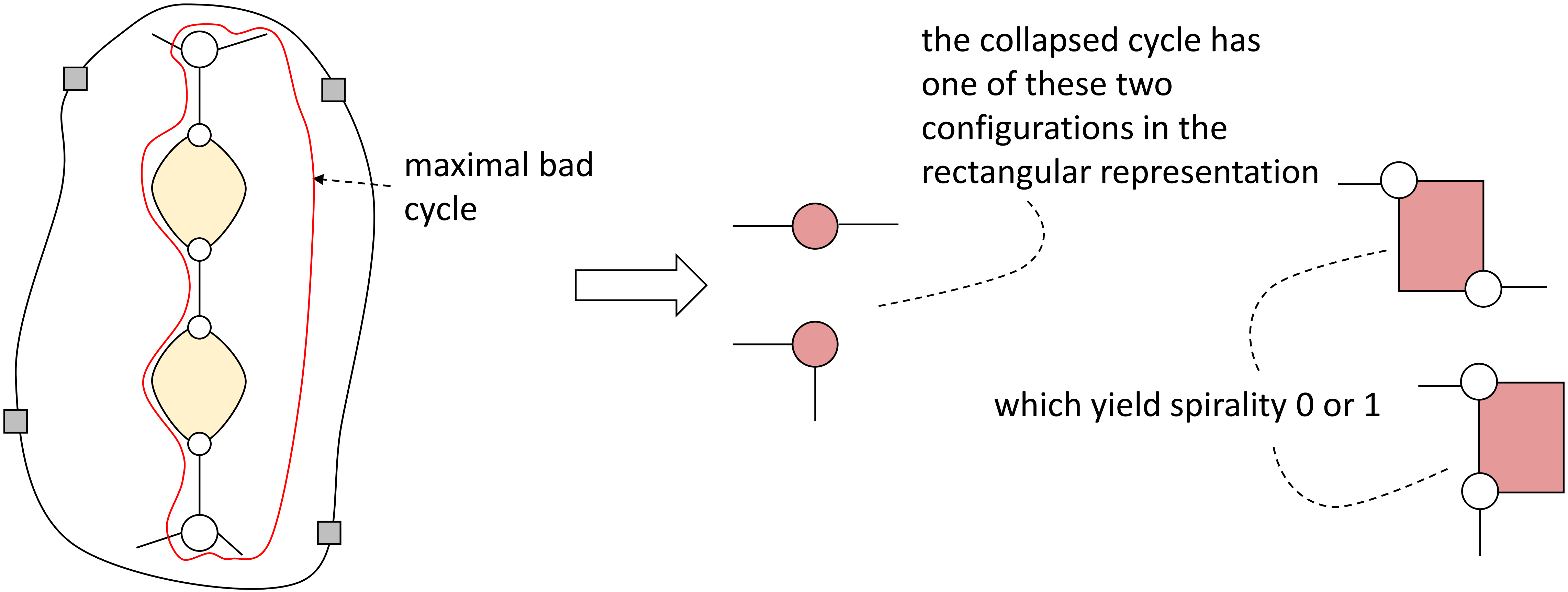
Case 1. the S-component is not inside a maximal bad cycle and all its edges are internal

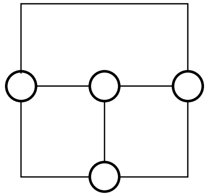


Key-Lemma: O3

Proof of O3 (<u>inner</u> S-components have spirality at most 4)

Case 2. the S-component is inside a maximal bad cycle that traverses the component

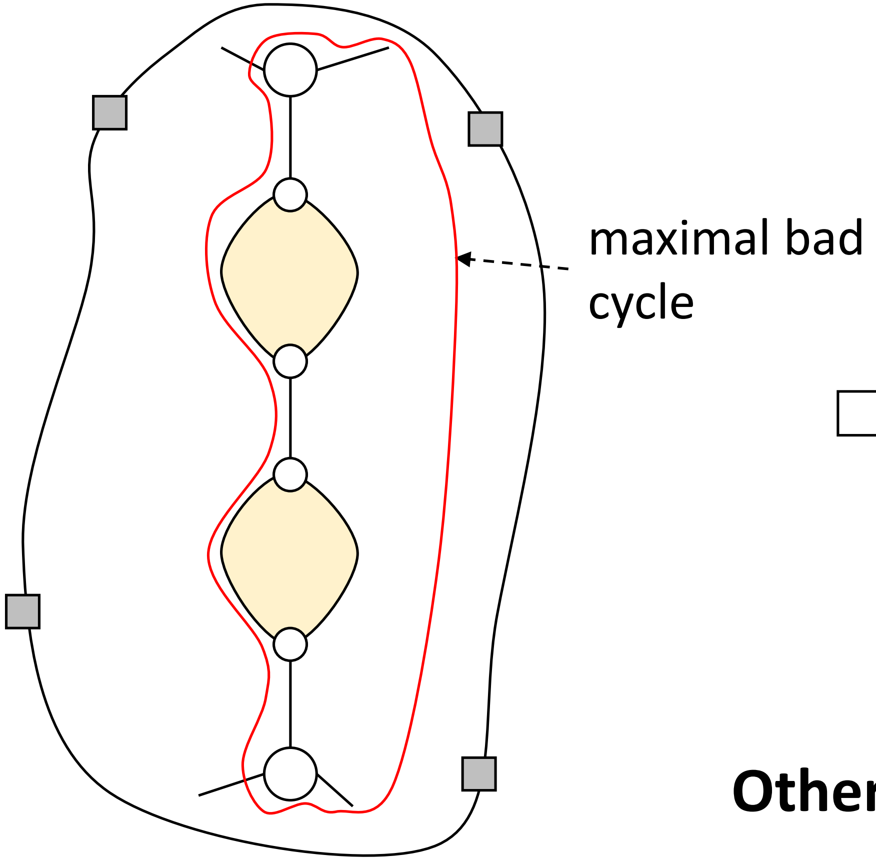


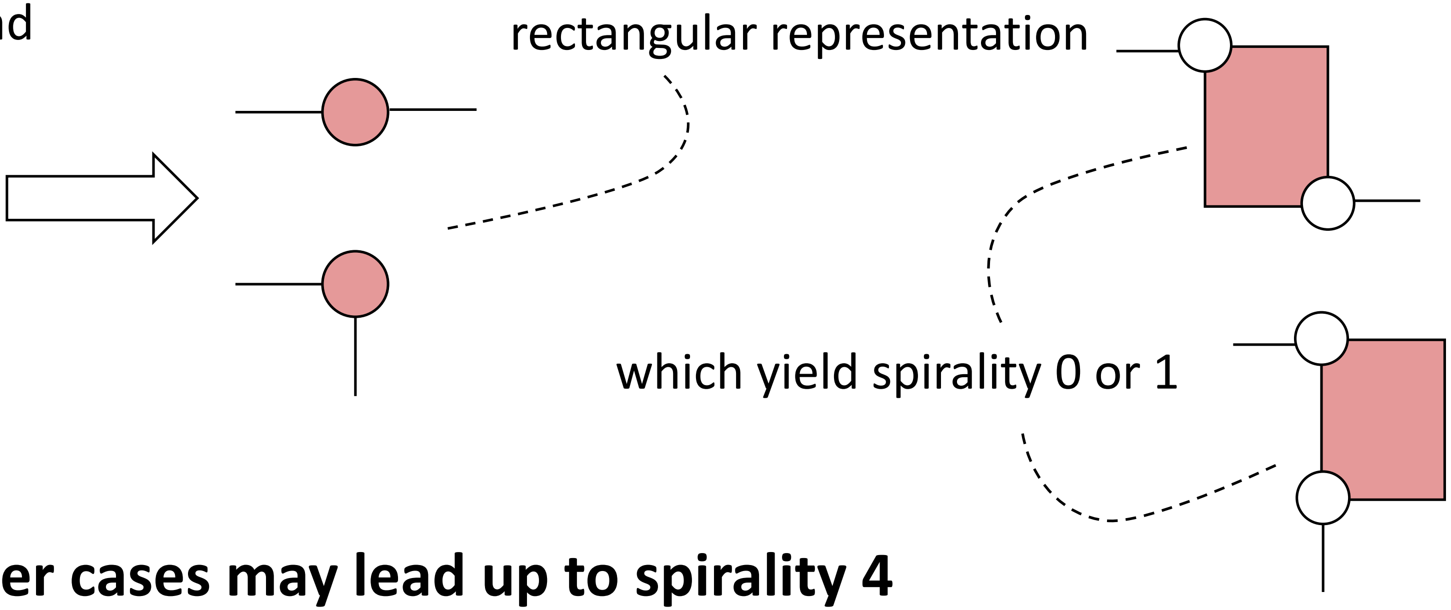


Key-Lemma: O3

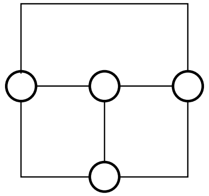
Proof of O3 (inner S-components have spirality at most 4)

Case 2. the S-component is inside a maximal bad cycle that traverses the component



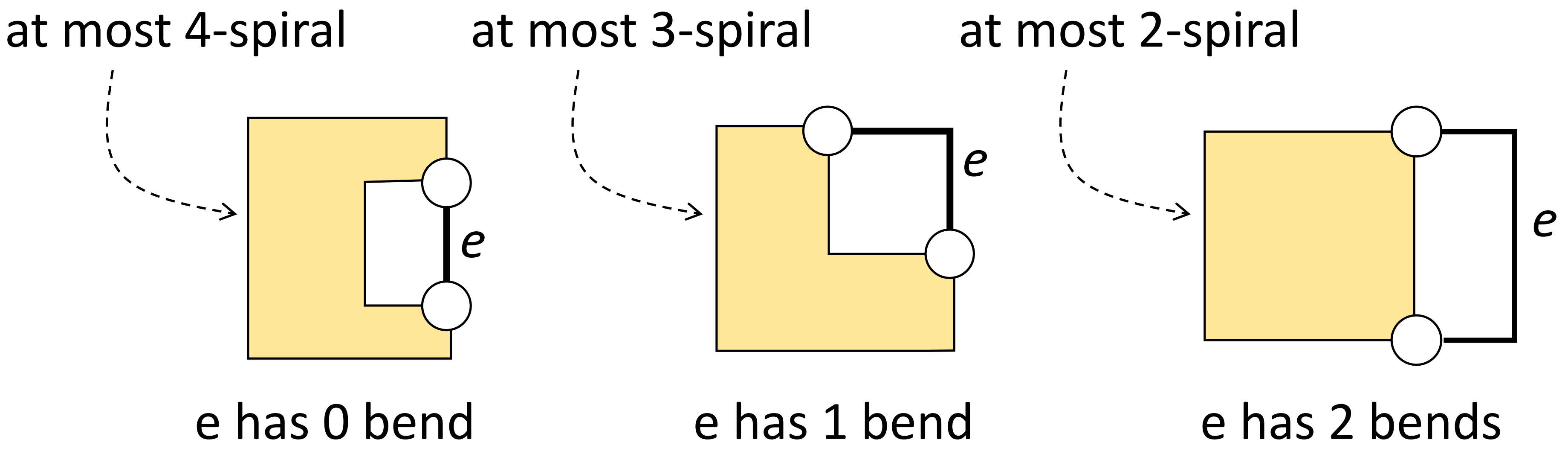


Other cases may lead up to spirality 4

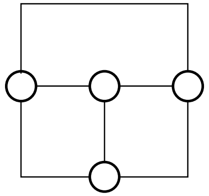


Key-Lemma: O3

Proof of O3 (a root child S-component has spirality at most 4)



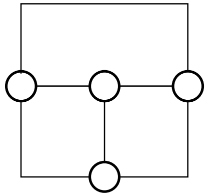
Higher values of spirality may only increase the number of bends



Algorithm

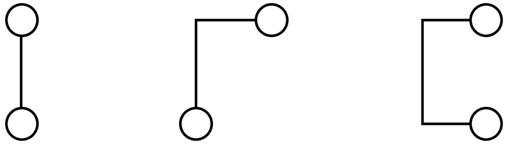
- **input**: biconnected planar 3-graph G with a reference edge *e*
- **output**: bend-min representation H of G with *e* on the external face

1. construct the SPQR-tree T of G with respect to *e*
2. visit the nodes μ of T **bottom-up**:
   - **μ inner node** ⇒ store in μ a candidate set of bend-min representations of G_{μ}- one for each distinct representative shape, thanks to the substitution theorem
   - **μ the root child** ⇒ construct H by suitably merging *e* with the candidate representations stored at the children of μ; consider {0, 1, 2} bends for *e*, thanks to O1 of the key-lemma

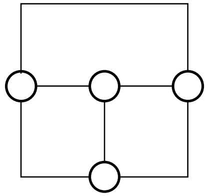


Candidate sets for the tree nodes

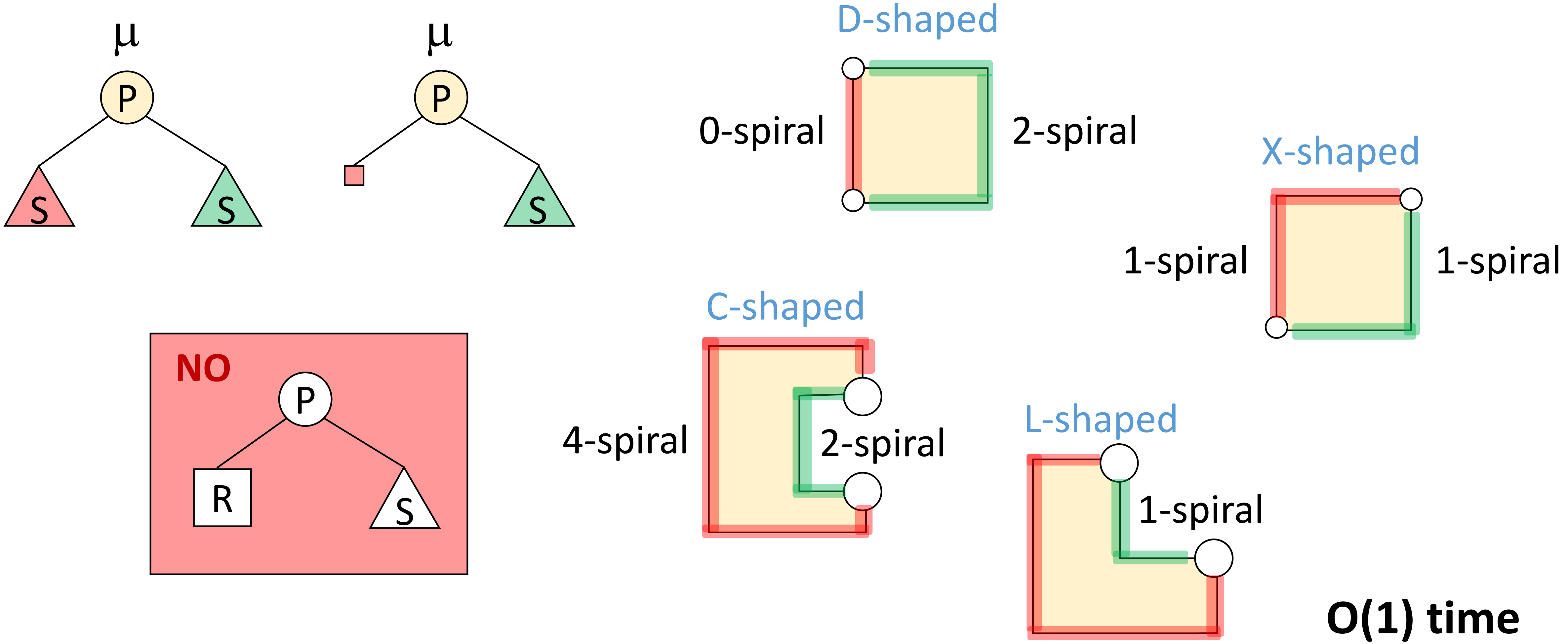
- **Q-node:** a representation for each number of bends in {0, 1, 2}
  - thanks to O1 of the key-lemma



- **P/R-node:** the cheapest D- and X-shaped representations for the inner nodes and the cheapest D-, C-, and L-shaped representations for the root child
  - thanks to O2 of the key-lemma
- **S-node:** the cheapest representation for each value of spirality in {0, 1, 2, 3, 4}
  - thanks to O3 of the key-lemma



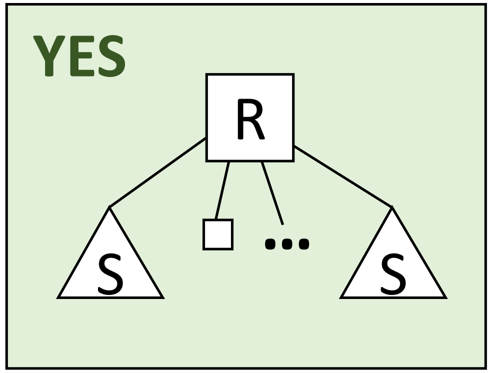
Candidate set of a P-node

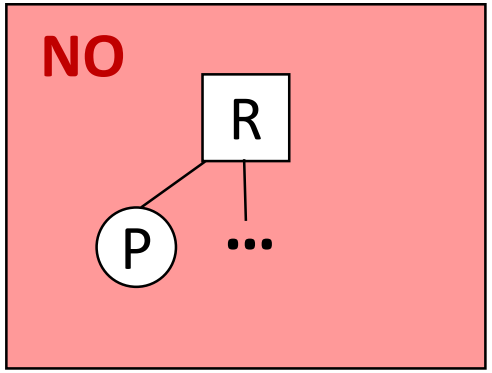


O(1) time

Candidate set of an R-node

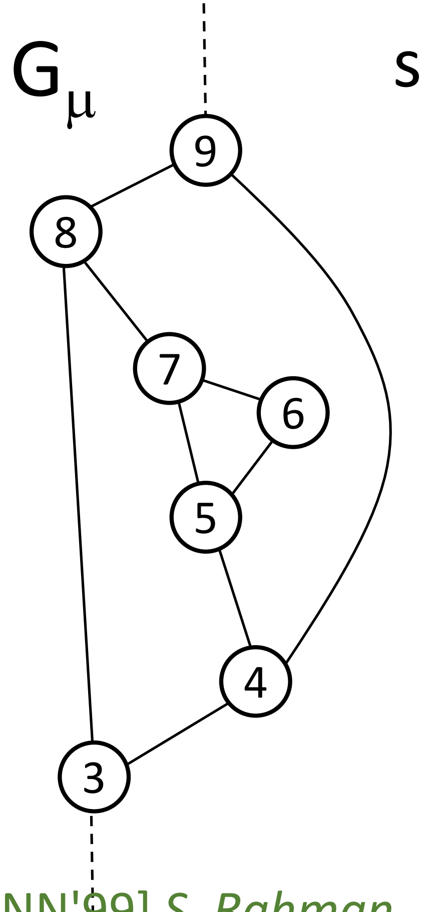
Each child of an R-node is either a Q- or an S-node



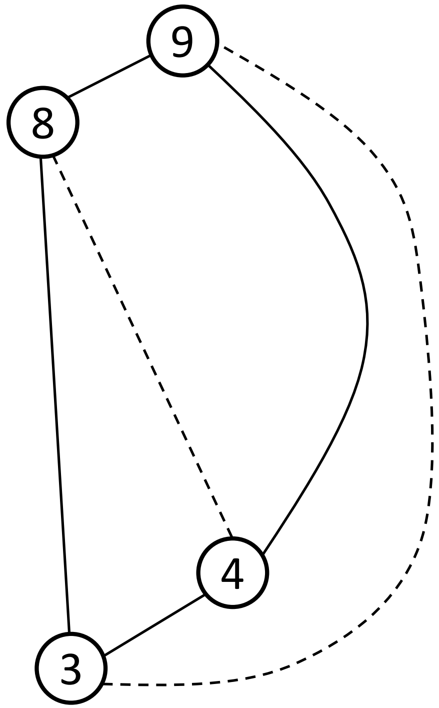


Candidate set of an R-node (sketch)

G_μ



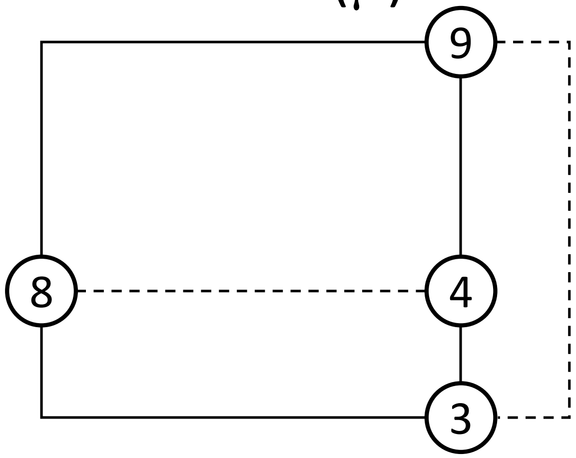
skel(μ)



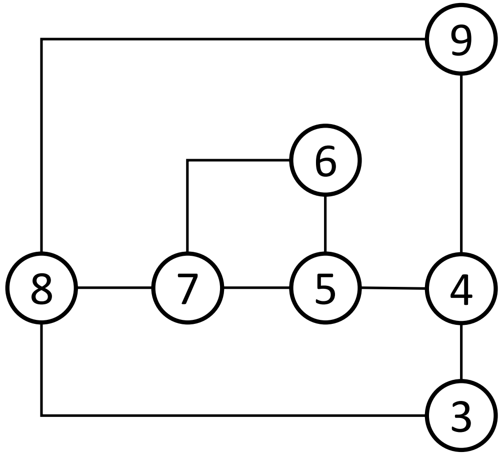
O(n)-time variant of [RNN'99]

bend-min 3-connected cubic (with constraints)

constrained bend-min repr. of skel(μ)

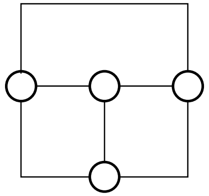


bend-min D-shaped

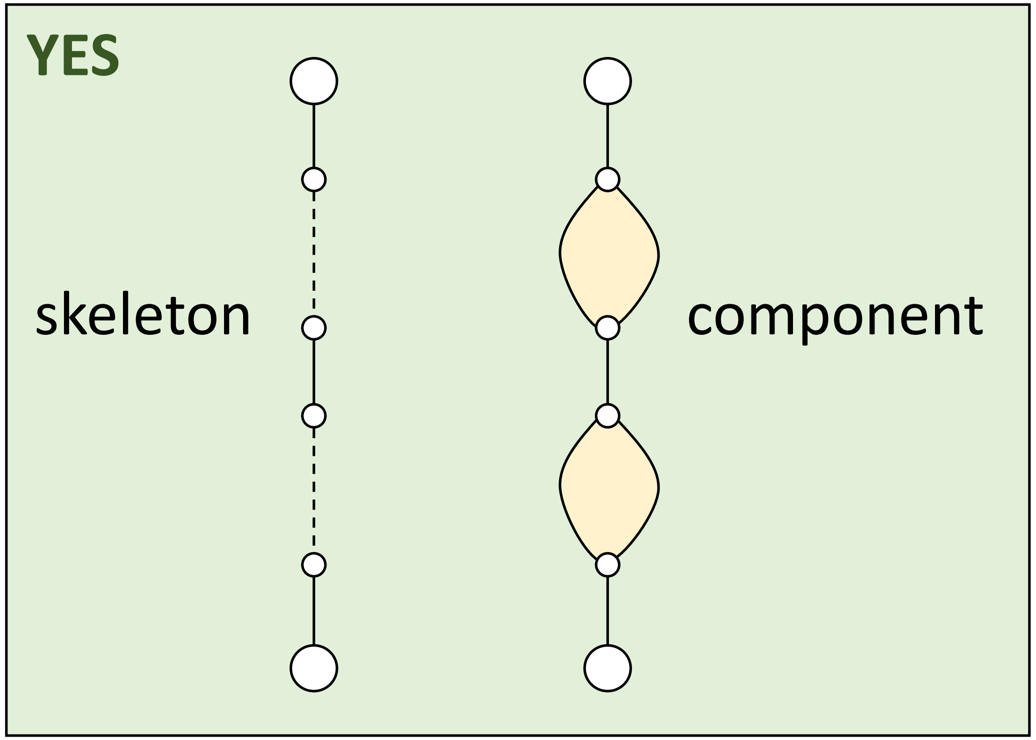


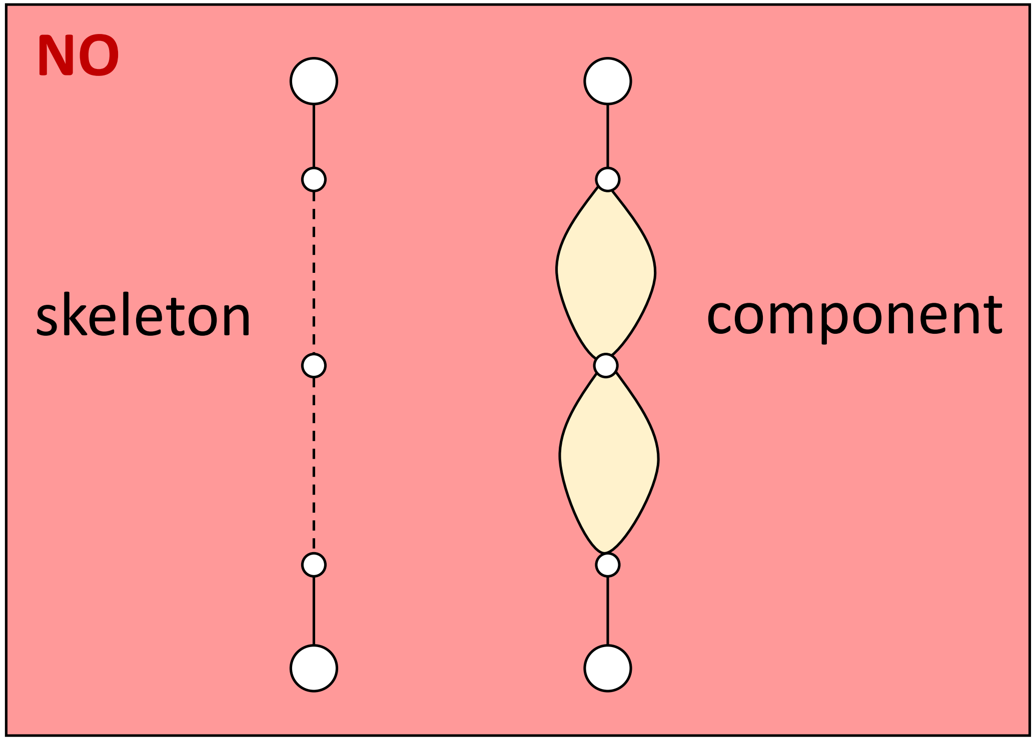
O(n_μ) time

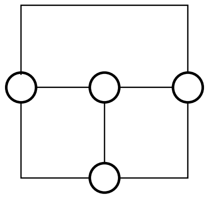
[RNN'99] *S. Rahman, S.-I. Nakano, T. Nishizeki*: A Linear Algorithm for Bend-Optimal Orthogonal Drawings of Triconnected Cubic Plane Graphs. J. Graph Algorithms Appl. 3(4): 31-62 (1999)



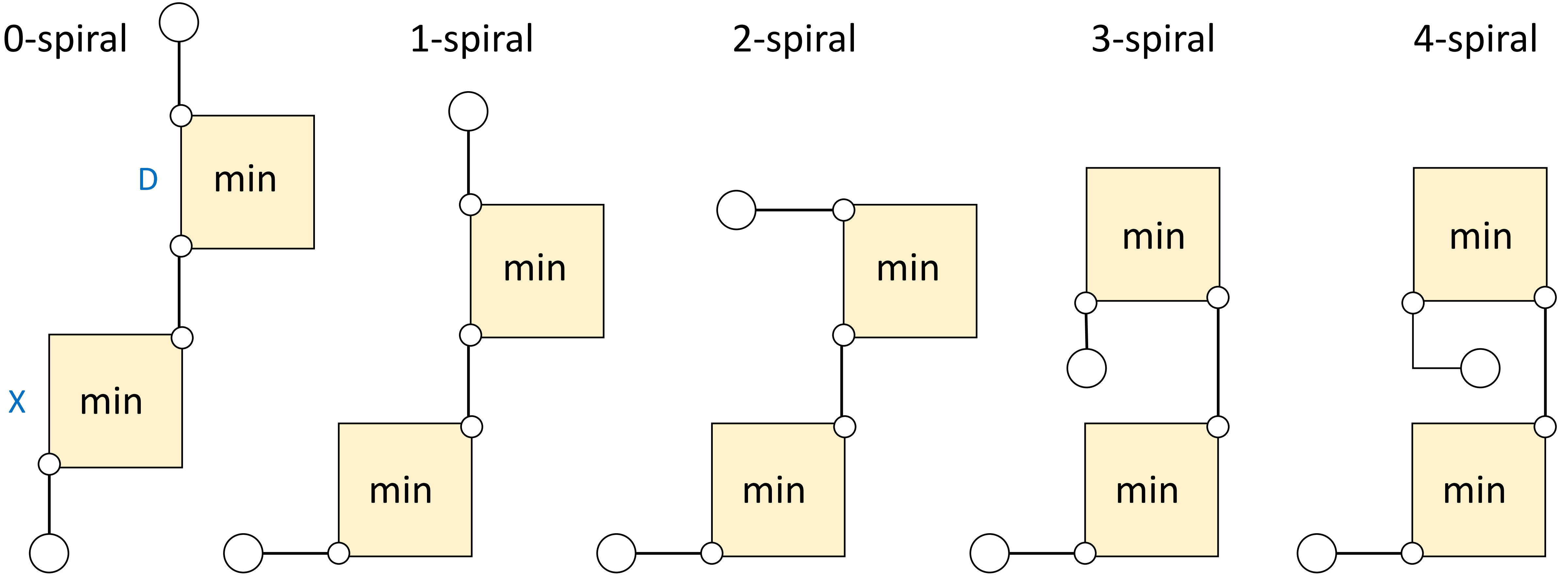
Candidate set of an S-node





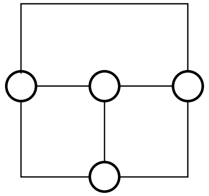


Candidate set of an S-node



#(extra bends) = max{0, spirality – (#D-shaped + #Q-nodes – 1)}

$O(n_\mu)$ time

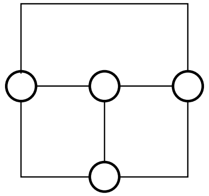


Question

- Is there a subquadratic-time algorithm to compute a bend-minimum orthogonal drawing of a planar 3-graph?

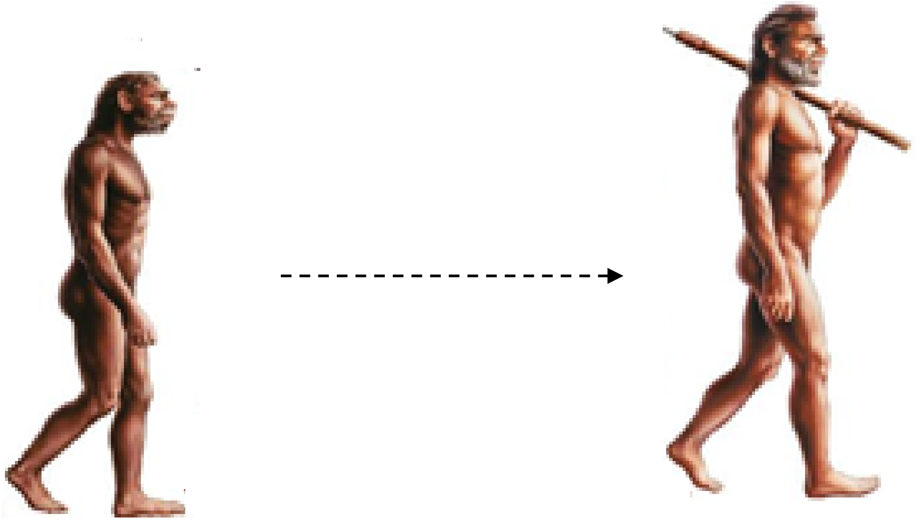
$O(n^2)$

Can we do better?



Question

- Is there a subquadratic-time algorithm to compute a bend-minimum orthogonal drawing of a planar 3-graph?

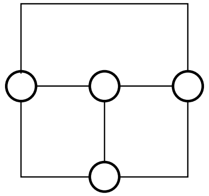


$O(n^2)$

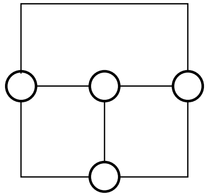
O(n)-time algorithm

Ingredients:

- new data structure for the rigid components
- labeling procedure for the candidate sets
- reusability principle for the SPQR-tree nodes



Bend-minimum orthogonal drawings of planar 4-graphs

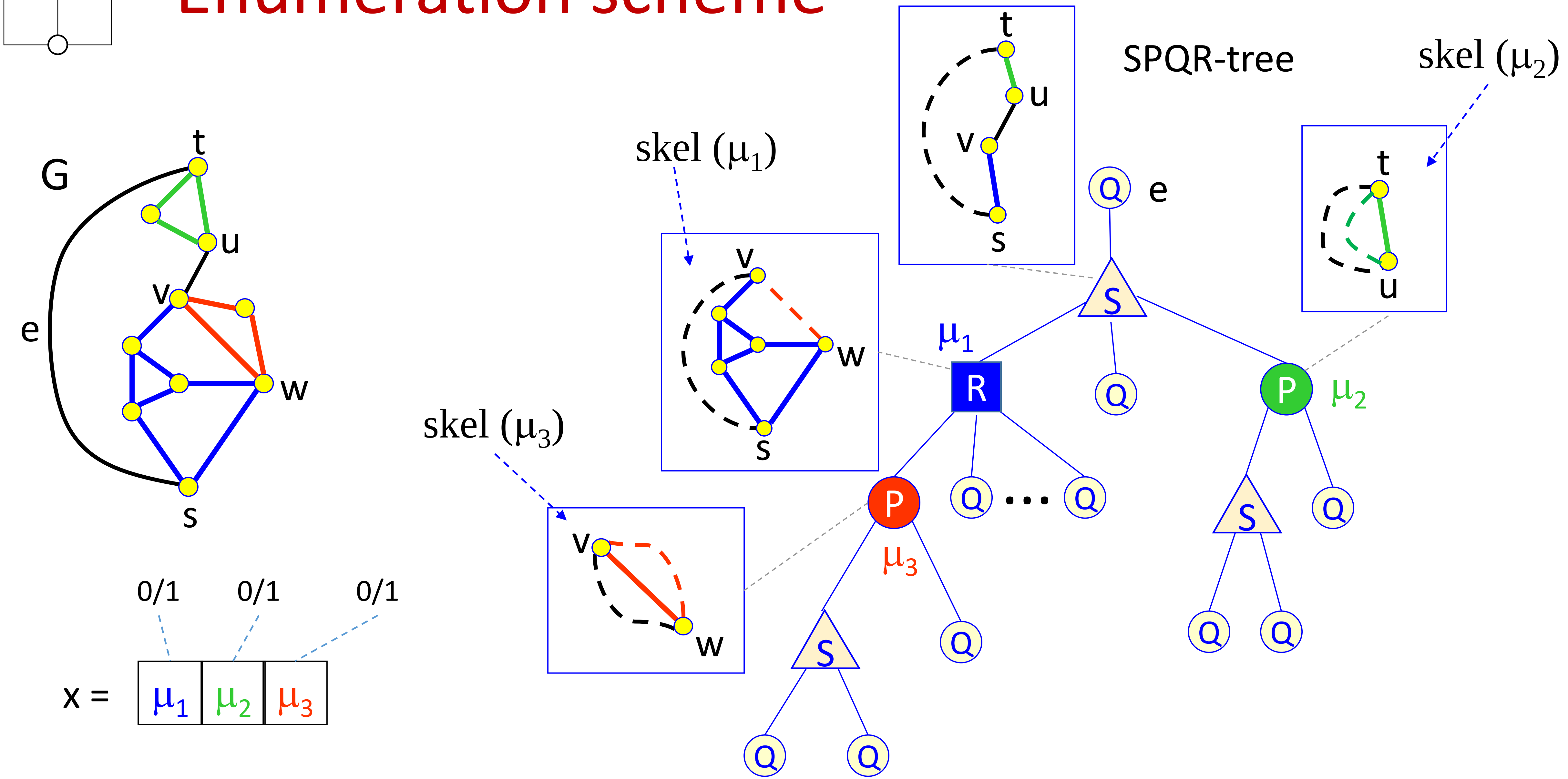


Bend-min of planar 4-graphs

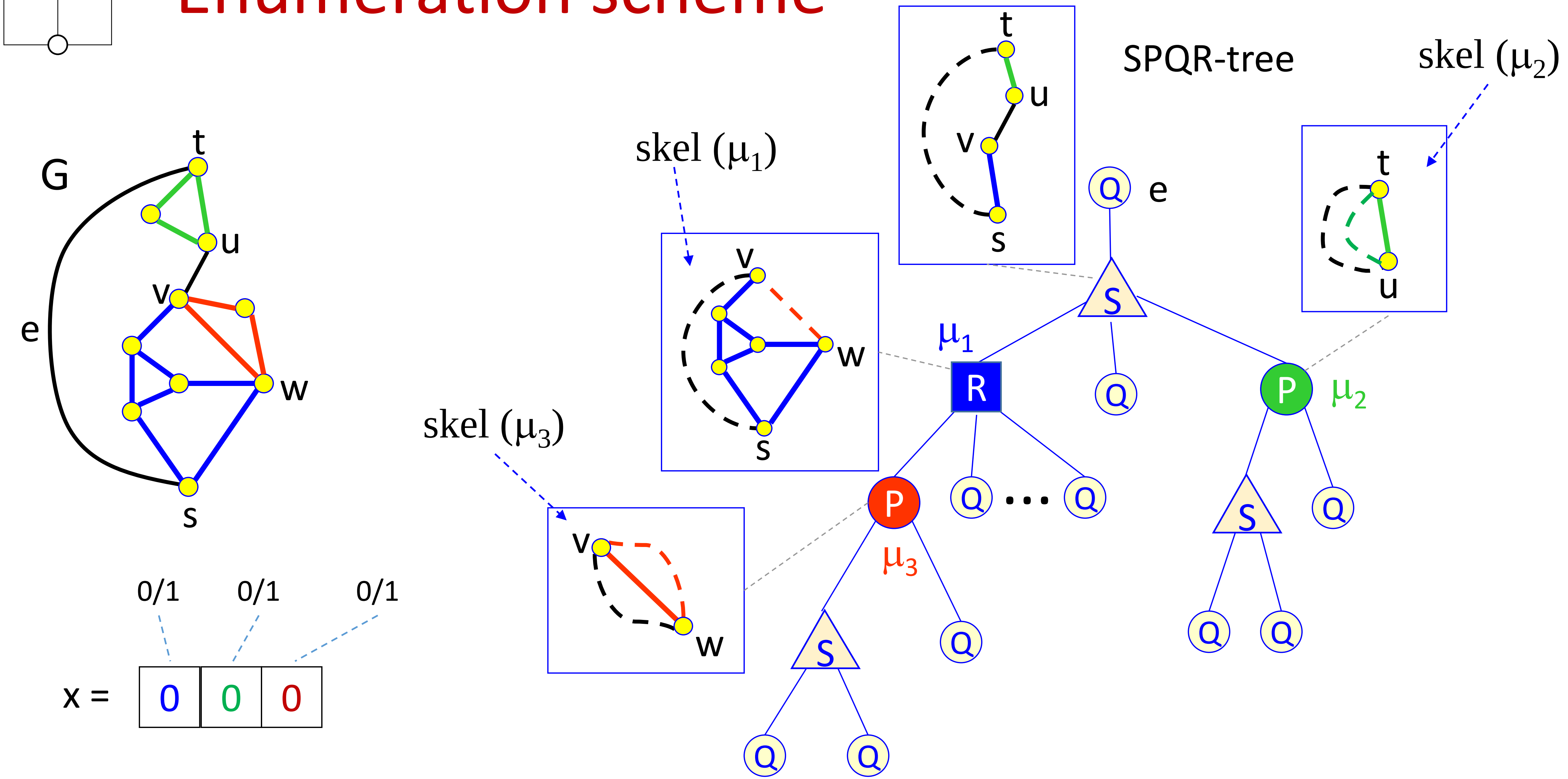
- Branch-and-bound algorithm for a biconnected graph G
  - *P. Bertolazzi, G. Di Battista, W. Didimo: Computing Orthogonal Drawings with the Minimum Number of Bends. IEEE Trans. Computers 49(8): 826-840 (2000)*

- Ingredients:
  - enumeration scheme for the planar embeddings of G
  - effective lower bounds on the number of bends
  - simple upper bounds on the number of bends

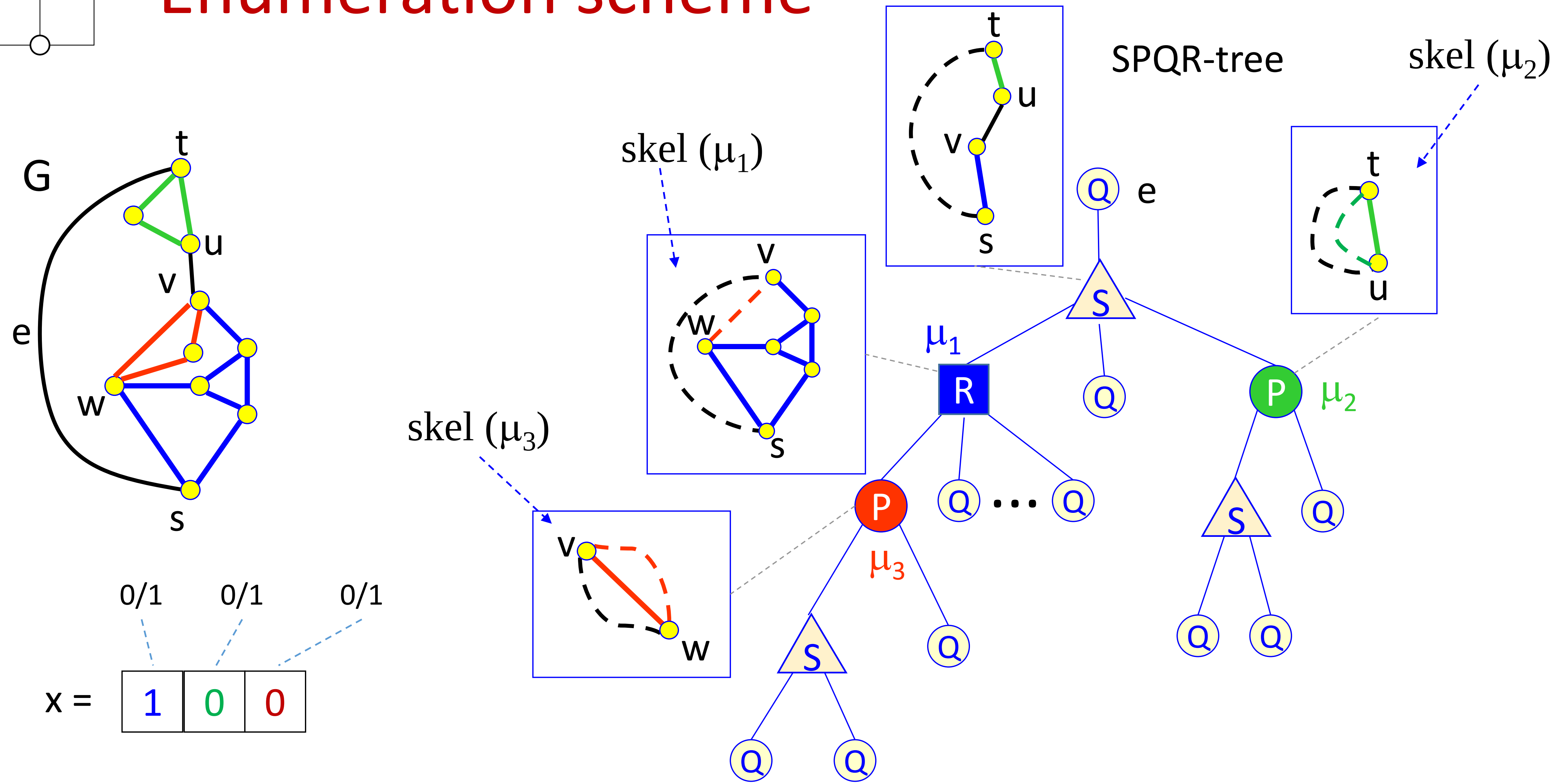
Enumeration scheme



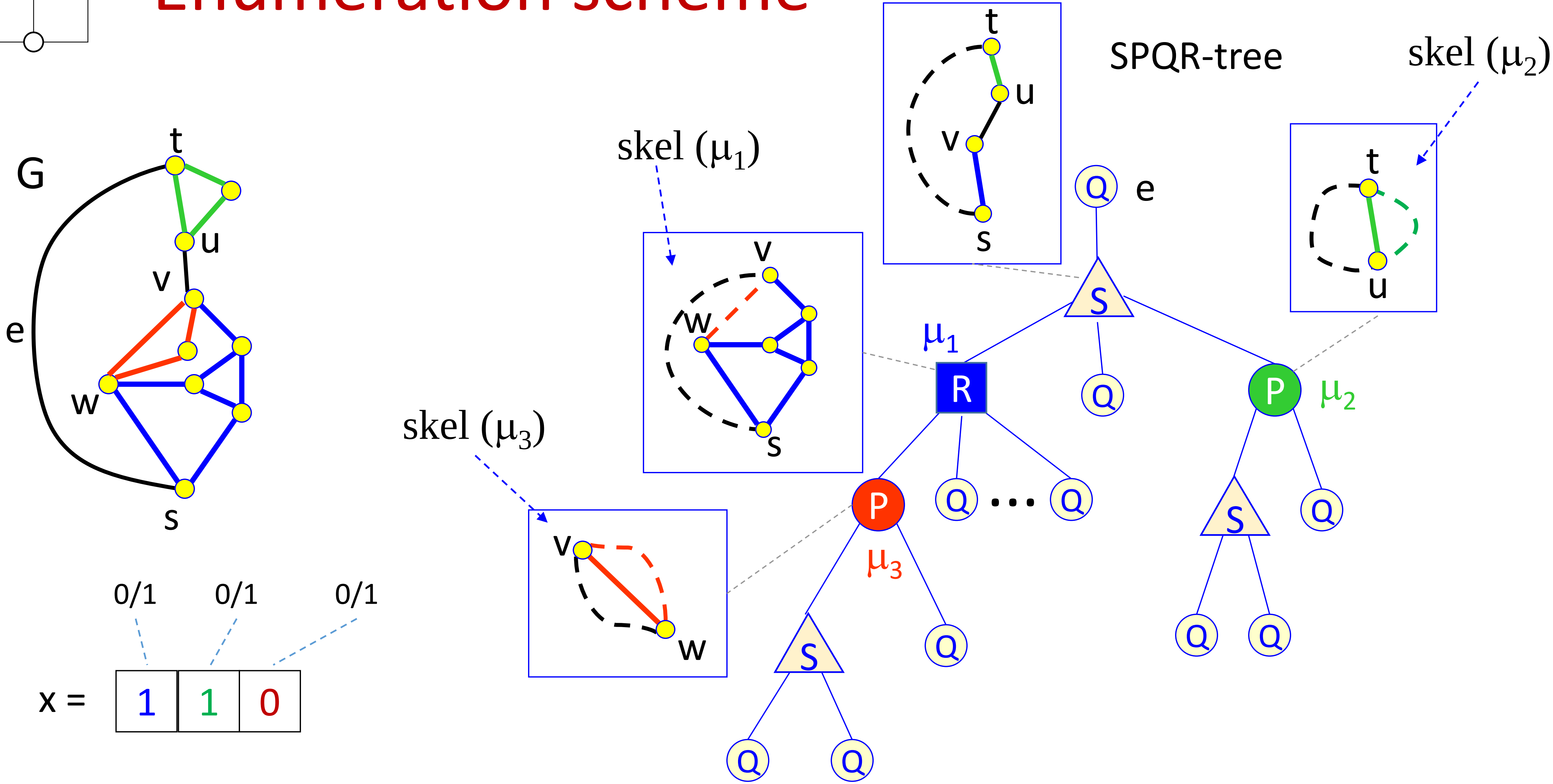
Enumeration scheme



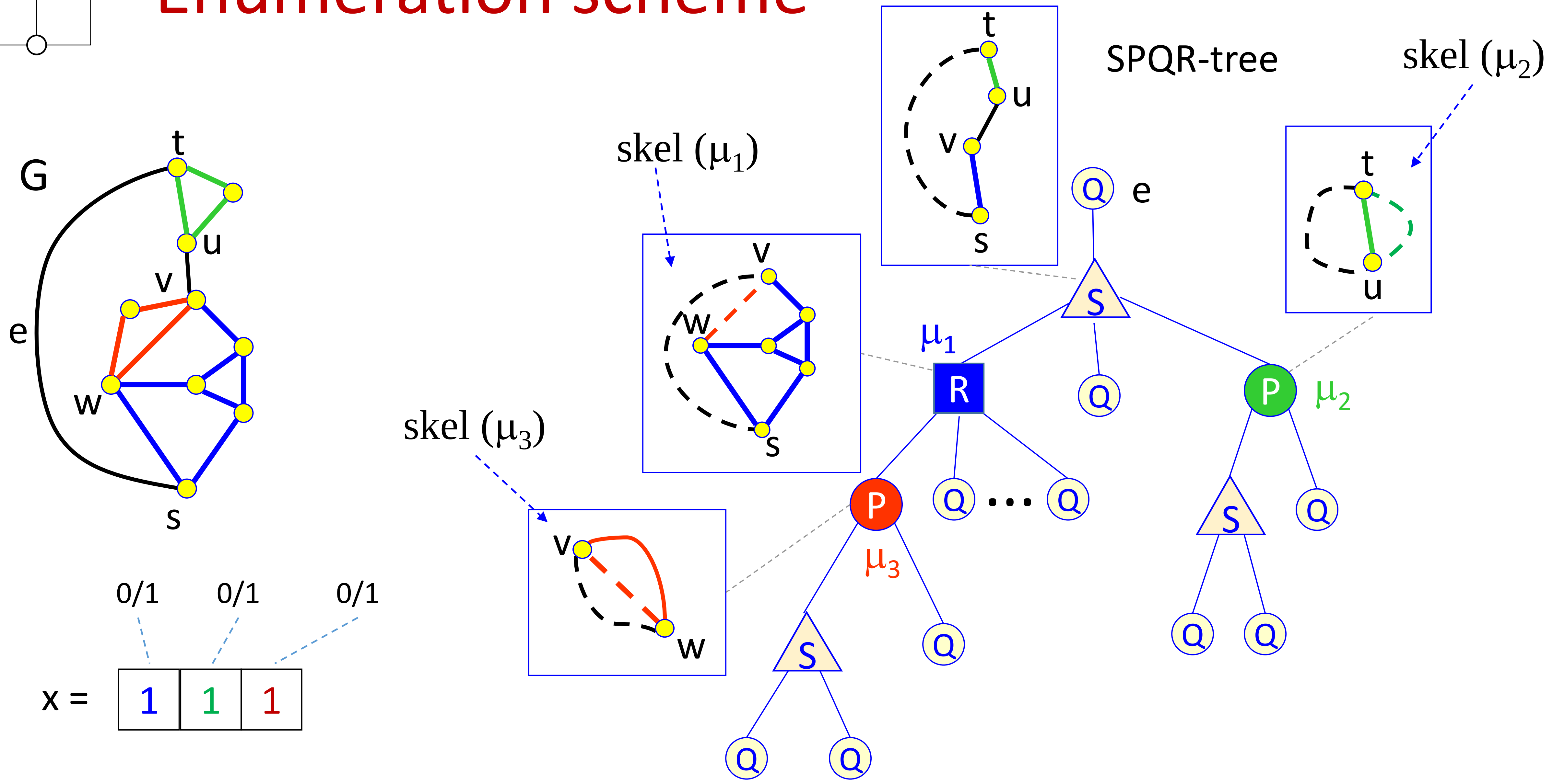
Enumeration scheme



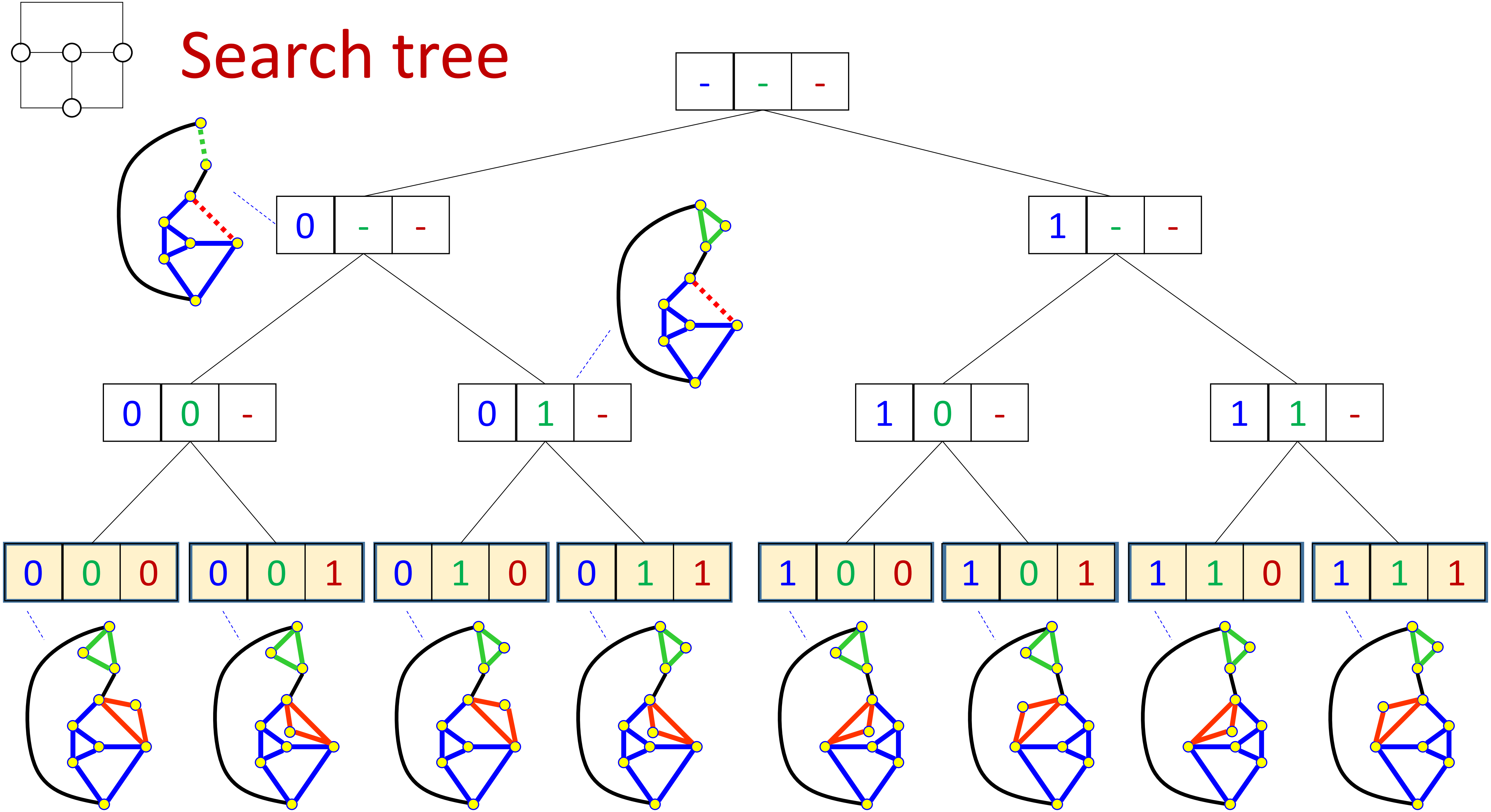
Enumeration scheme



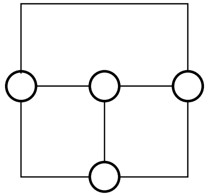
Enumeration scheme

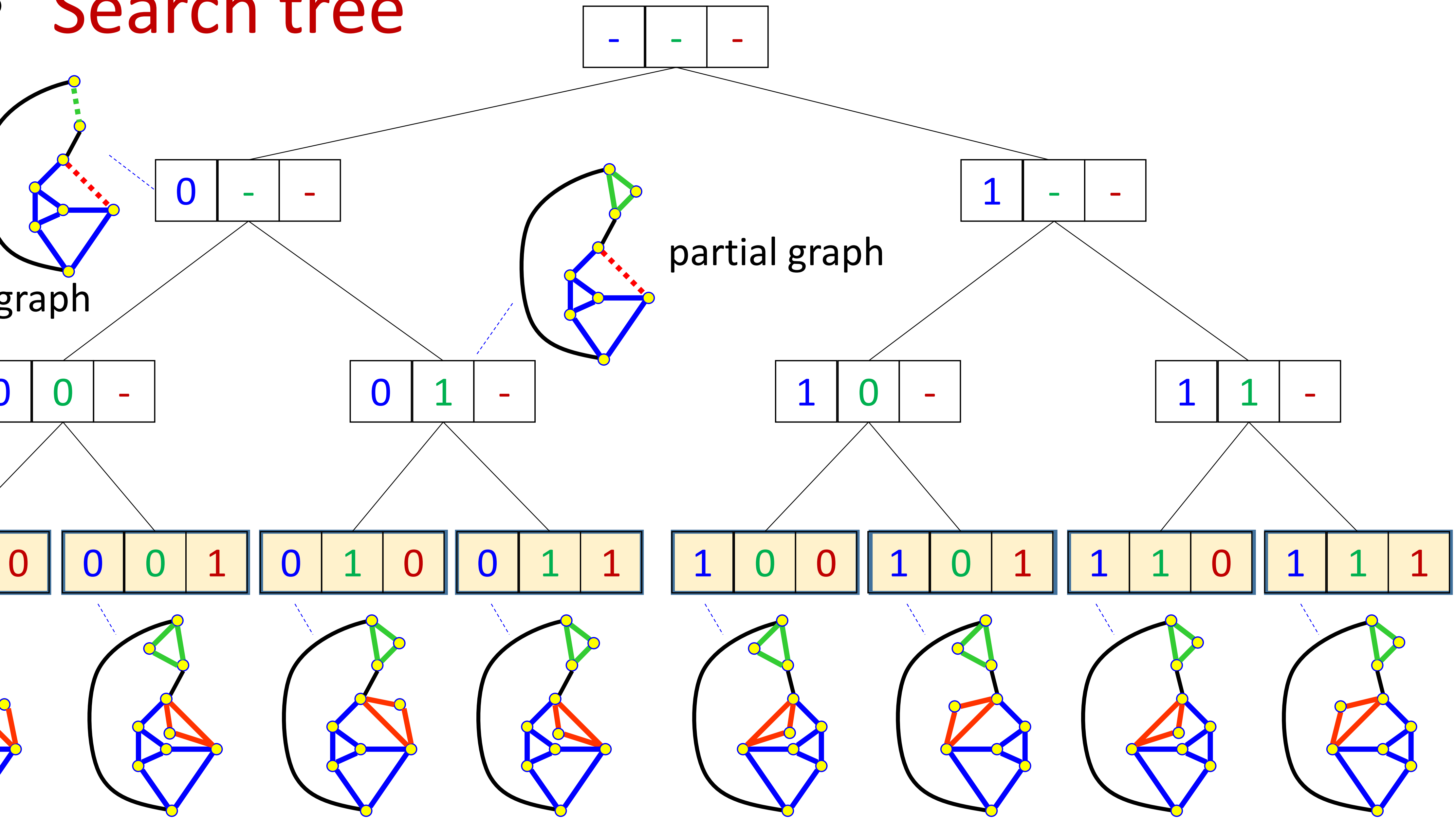


Search tree

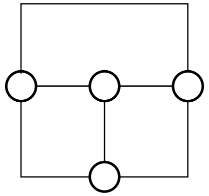


Search tree





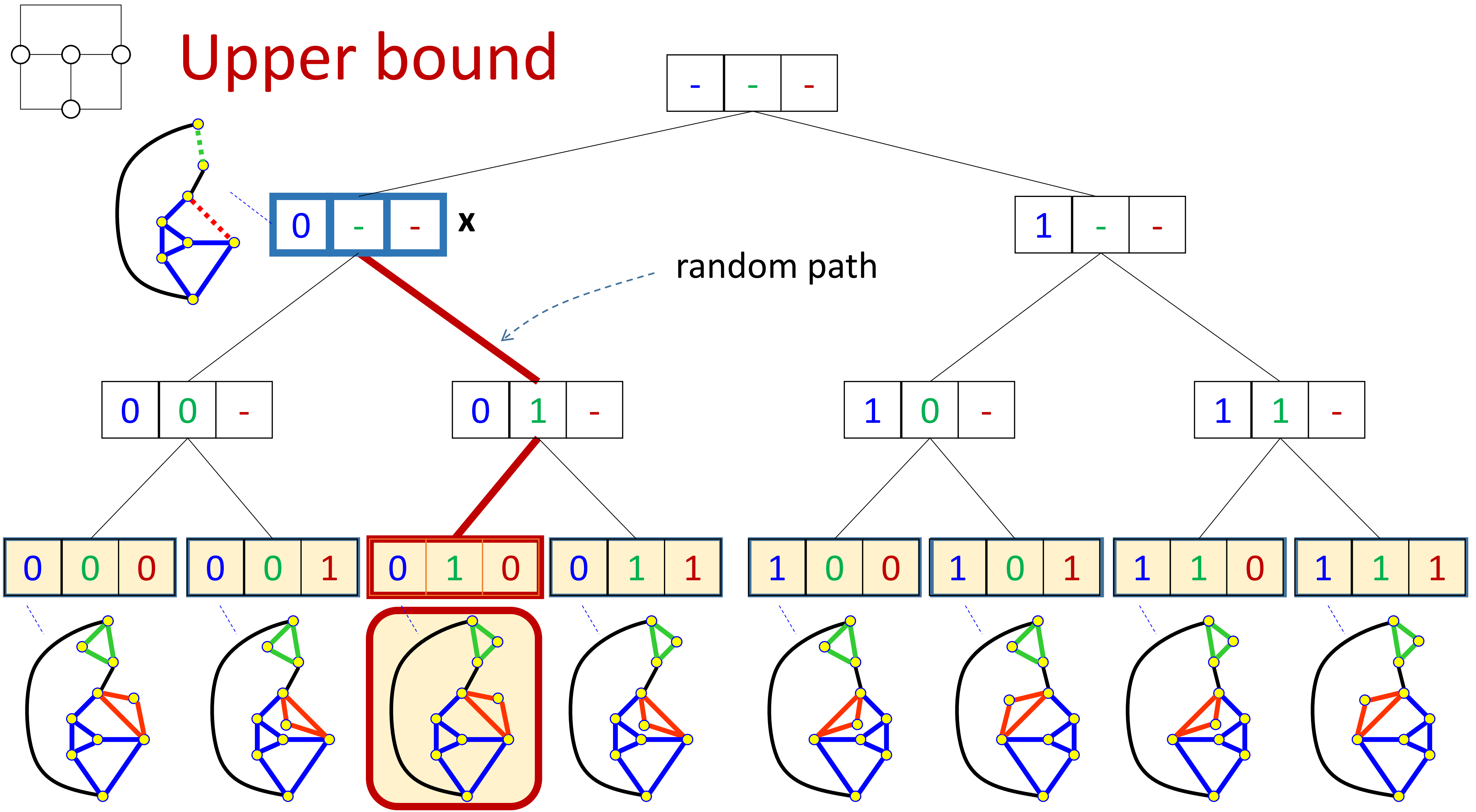
- - -
0 - -
1 - -
partial graph
partial graph
0 0 -
0 1 -
1 0 -
1 1 -
partial graph
0 0 0
0 0 1
0 1 0
0 1 1
1 0 0
1 0 1
1 1 0
1 1 1



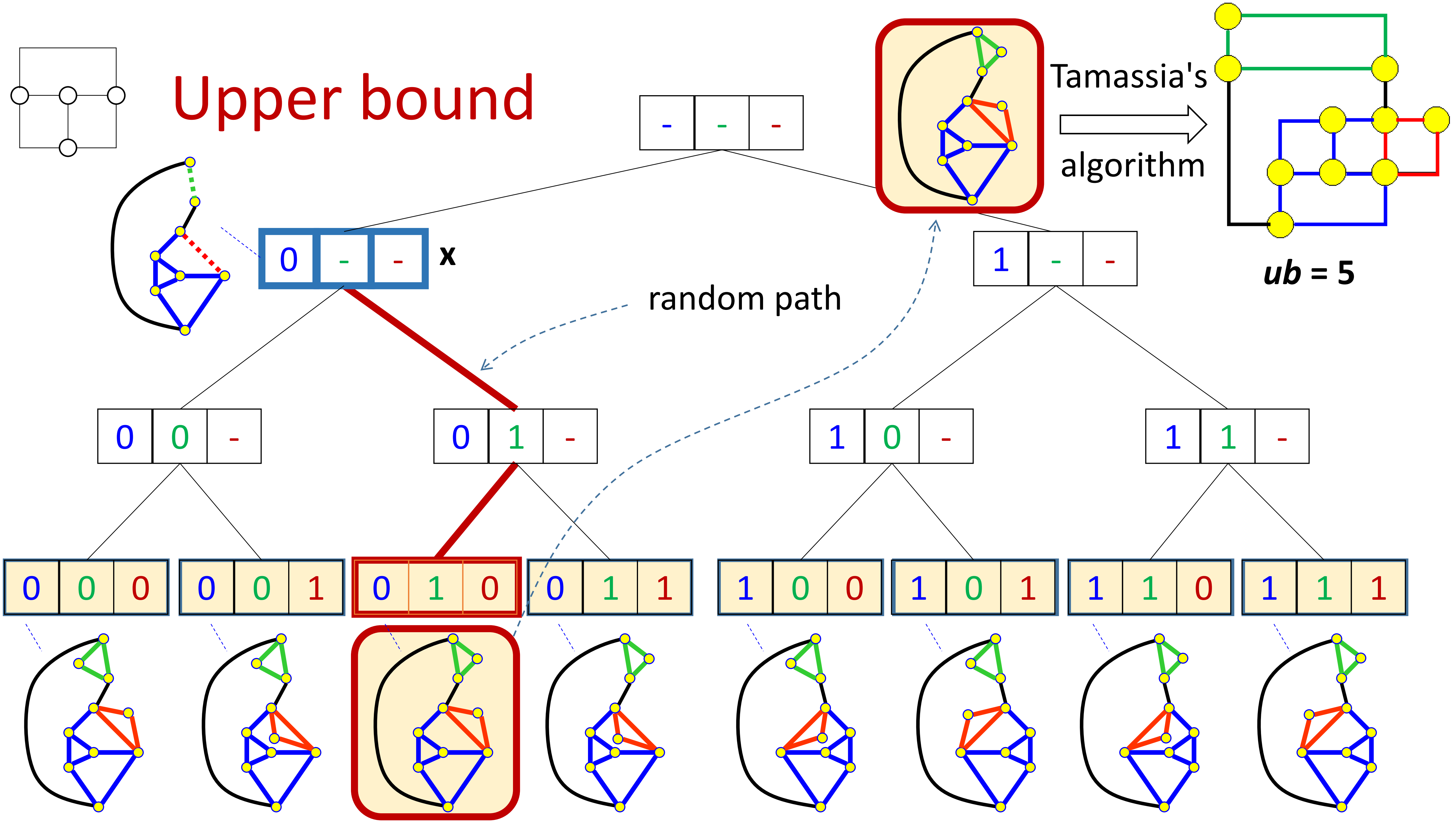
Branch-and-Bound algorithm

- mb $\leftarrow +\infty$ // minimum number of bends known so far
- visit the search tree from the root (use a BFS or DFS)
- when a node x is visited:
  - compute an upper bound *ub* on the number of bends of an orthogonal representation with embedding in the subtree rooted at x
    - **If** (ub < mb) **then** mb $\leftarrow$ ub
  - compute a lower bound *lb* on the number of bends of an orthogonal representation with embedding in the subtree rooted at x
    - **If** (lb > mb) **then** cut x and its subtree
- **return** mb

Upper bound



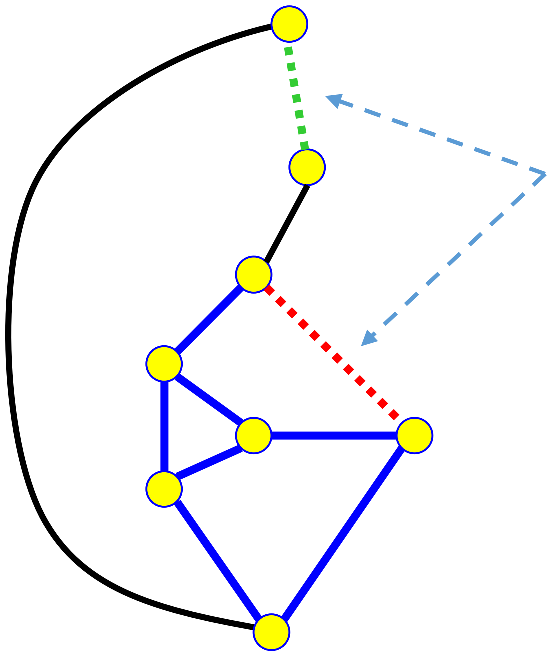
Upper bound



- - -
0 - -
x
1 - -
random path
Tamassia's
algorithm
ub = 5
0 0 -
0 1 -
1 0 -
1 1 -
0 0 0
0 0 1
0 1 0
0 1 1
1 0 0
1 0 1
1 1 0
1 1 1

Lower bound: Notation

partial graph



EV = set of virtual edges

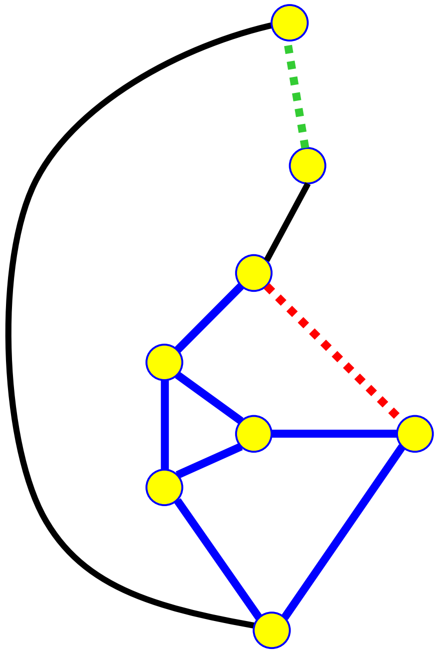
ER = set of real edges

$\mathbf{b_{ER}(H)}$= # of bends of H along the real edges

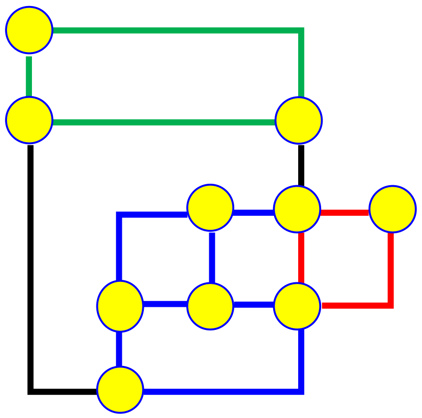
Lower bound: Preliminary lemma

- H' = representation of G' with minimum bends on ER
- H = bend-min representation of G that preserves the embedding of G'

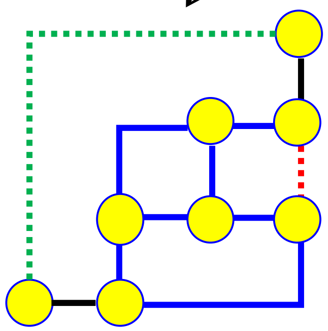
partial graph G'



$$\mathbf{b_{ER}(H') \leq b_{ER}(H)}$$



b_{ER} (H) = 3



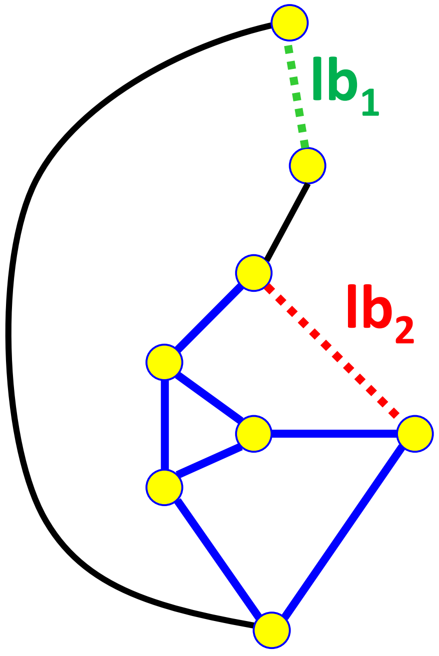
b_{ER} (H') = 2

b_{ER}(H') can be computed by imposing cost 0 for the bends on the virtual edges in Tamassia's flow network

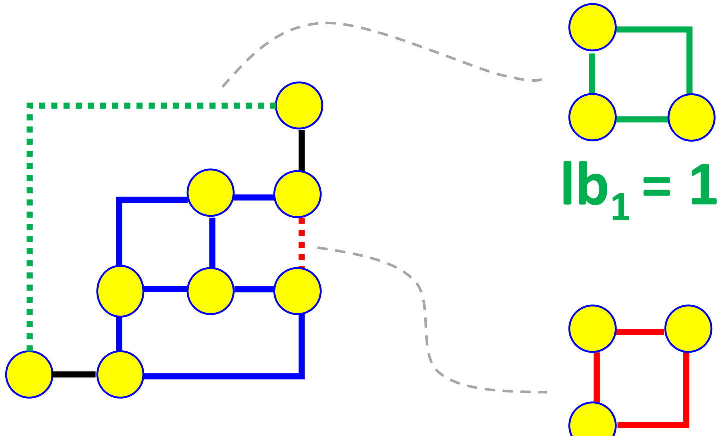
Lower bound: Recursive approach

- **lb_i**= lower bounds on the # of bends in the pertinent graph of a component G_i

partial graph G'



$$\boldsymbol{lb = b_{ER}(H') + \Sigma_i\, lb_i}$$



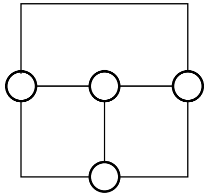
b_{ER} (H') = 2

$lb_1 = 1$

$lb_2 = 1$

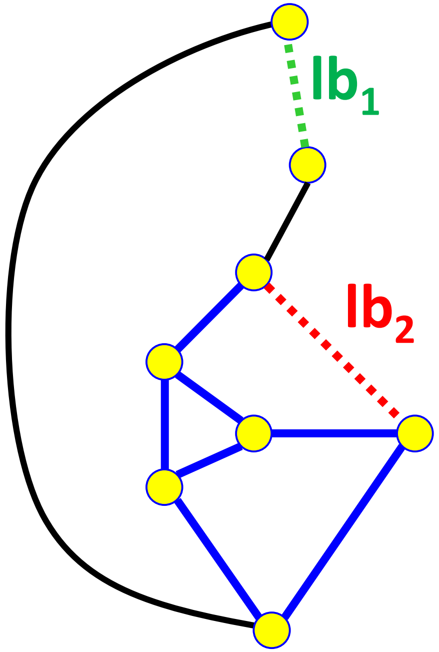
$lb = 4$

the set of **lb_i** can be computed through a bottom-up visit of the SPQR-tree in a pre-processing step



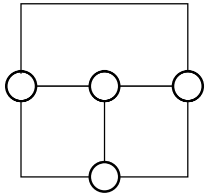
Lower bound: Further improvement

partial graph G'



- If some lb_i is zero, replace the corresponding virtual edge with a simple path π between the poles of G_i and regard the edges of π as real edges

$$\boldsymbol{lb = b_{ER}(H') + \Sigma_i\, lb_i}$$



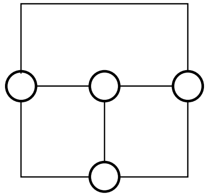
Some experimental data

density/vertices	10	20	30	40	50	60	70	80	90	100
1.1	6	10	10	25	25	10	10	4	13.33	0
1.2	37.5	32.38	27	26.33	41.3	38.67	32.1	17.32	33.28	31.76
1.4	20.82	22.31	19.99	19.92	22.35	28.99	24.88	16.59	20.36	14.2
1.6	19.75	15.05	20.76	12.16	13.14	12.4	15.92	11.87	14.61	12.65
1.8	13.04	11.05	10.46	10.08	8.15	9.94	4.07	4.77	4.21	

% avg. improvement on the number of bends w.r.p. to a bend-minimum orthogonal drawing in the fixed embedding setting

Additional reading

- *P. Mutzel, R. Weiskircher*: Bend Minimization in Planar Orthogonal Drawings Using Integer Programming. SIAM Journal on Optimization 17(3): 665-687 (2006)



Bend-min of planar 4-graphs: Open problem

- **Problem**: Let G be a biconnected 4-planar graph with a given combinatorial embedding. is there an $o(n^{2.5})$-time algorithm that computes a bend-minimum orthogonal drawing of G overall possible choices of the external faces? (the combinatorial embedding is preserved)