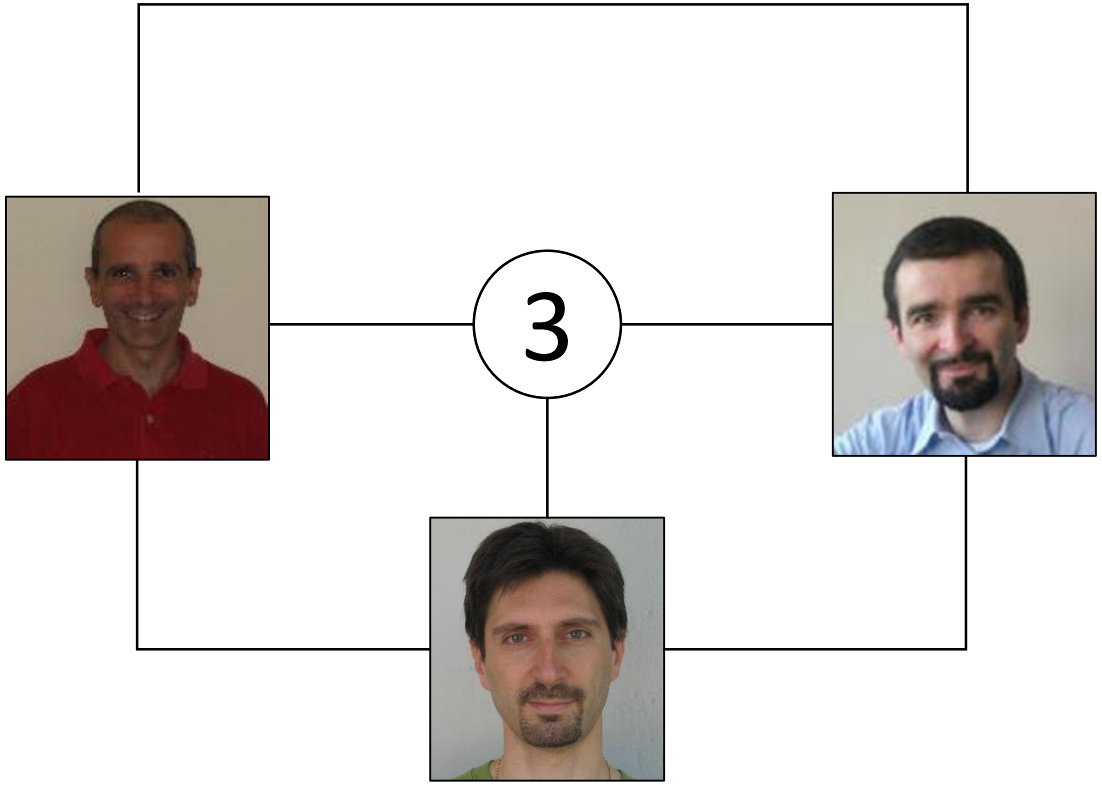
Bend-minimum Orthogonal Drawings in Quadratic Time



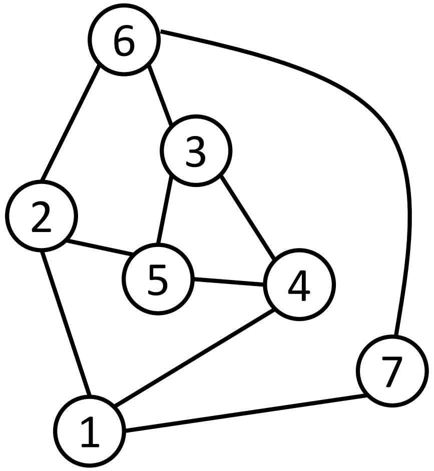
Giuseppe Liotta

Maurizio Patrignani

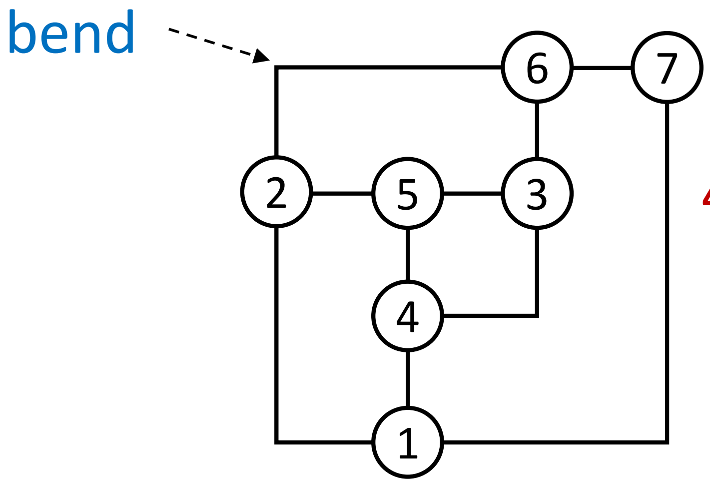
Walter Didimo

The problem

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

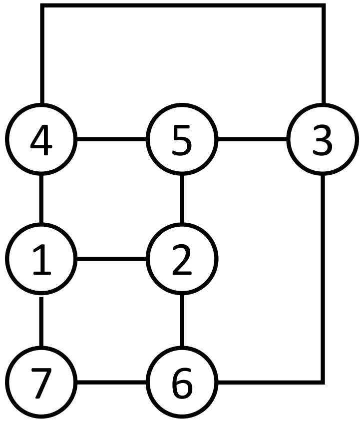


plane 3-graph



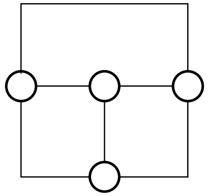
4 bends

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



3 bends

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



A bit of history

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

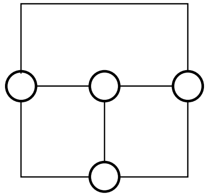
- plane 4-graphs
  - $O(n^2 \log n)$ [Tamassia (1987)]
  - $O(n^{7/4} \sqrt{\log n})$ [Garg and Tamassia (2001)]
  - $O(n^{1.5})$ [Cornelsen and Karrenbauer (2011)]

  based on min-cost flow

- plane 3-graphs

  $O(n)$ [Rahman and Nishizeki (2002)]

  not based on flow techniques



A bit of history

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

- planar 4-graphs: NP-hard [Garg and 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

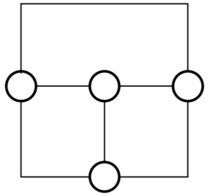
$O(n^{2.43} \log^k n)$
Chang and Yen

2018

$O(n^2)$
our result

?

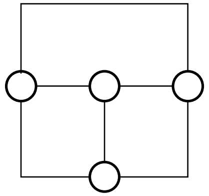
Can we do better?



Our 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



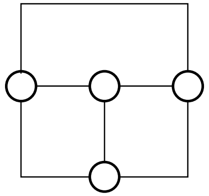
General strategy for biconnected graphs

input: G biconnected planar 3-graph with n vertices

output: bend-min orthogonal drawing Γ of G

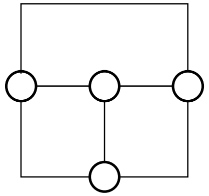
- for each edge *e* of G
  - $\Gamma_e \leftarrow$ min-bend orthogonal drawing of G with *e* on the external face
- return $\Gamma \leftarrow$ min-bends $\{\Gamma_e\}$

Γ_e is computed in linear time



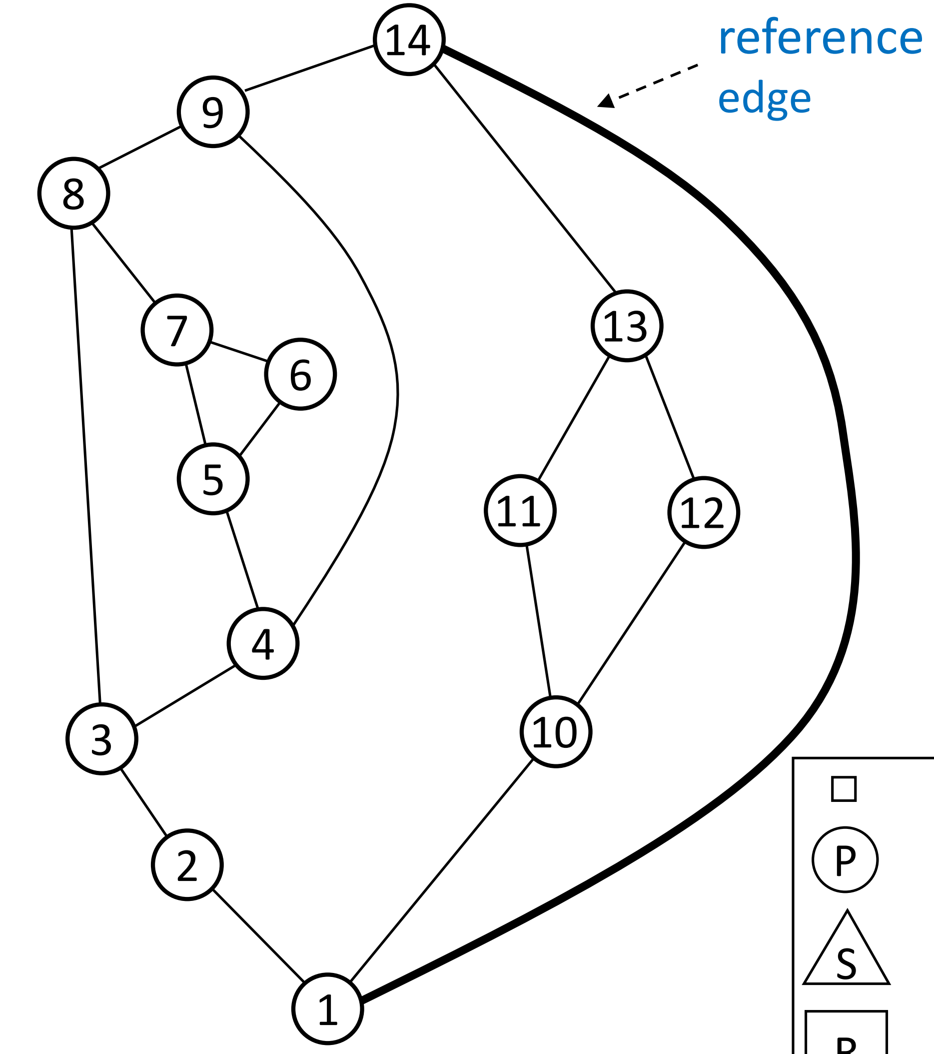
Strategy for the linear-time algorithm

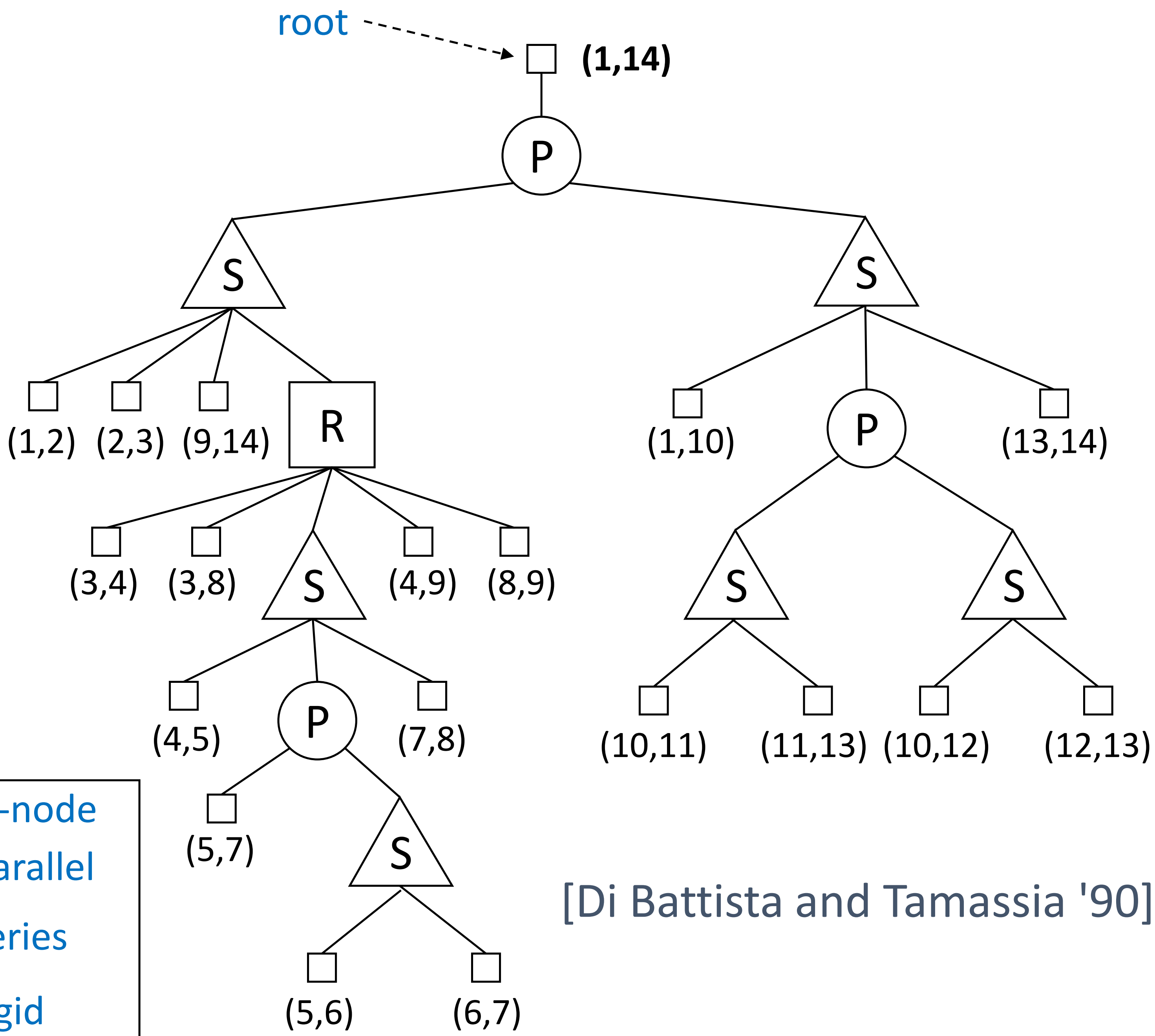
- Incremental construction of Γ_e
  1. bottom-up visit of the SPQR-tree + *orthogonal spirality* (similar to Di Battista, Liotta, Vargiu 1998)
  2. new properties of bend-min orthogonal drawings of planar 3-graphs
  3. non-flow based computation of bend-min orthogonal drawings for the rigid components

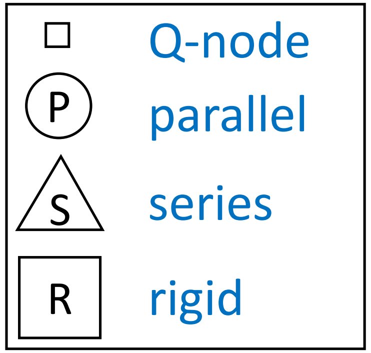


\begin{SPQR-trees}

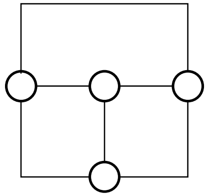
SPQR-trees



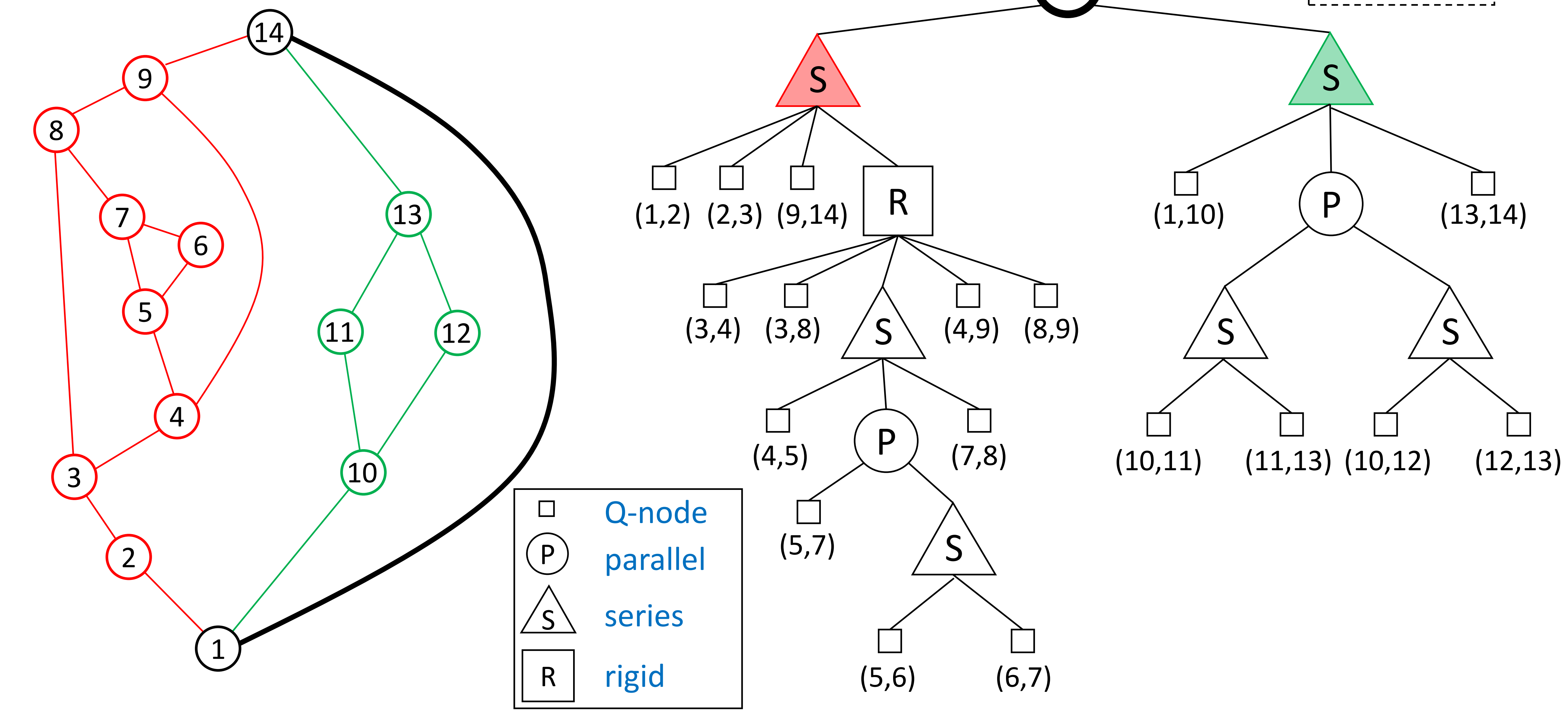




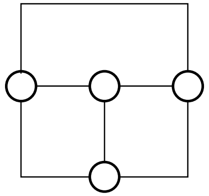
[Di Battista and Tamassia '90]



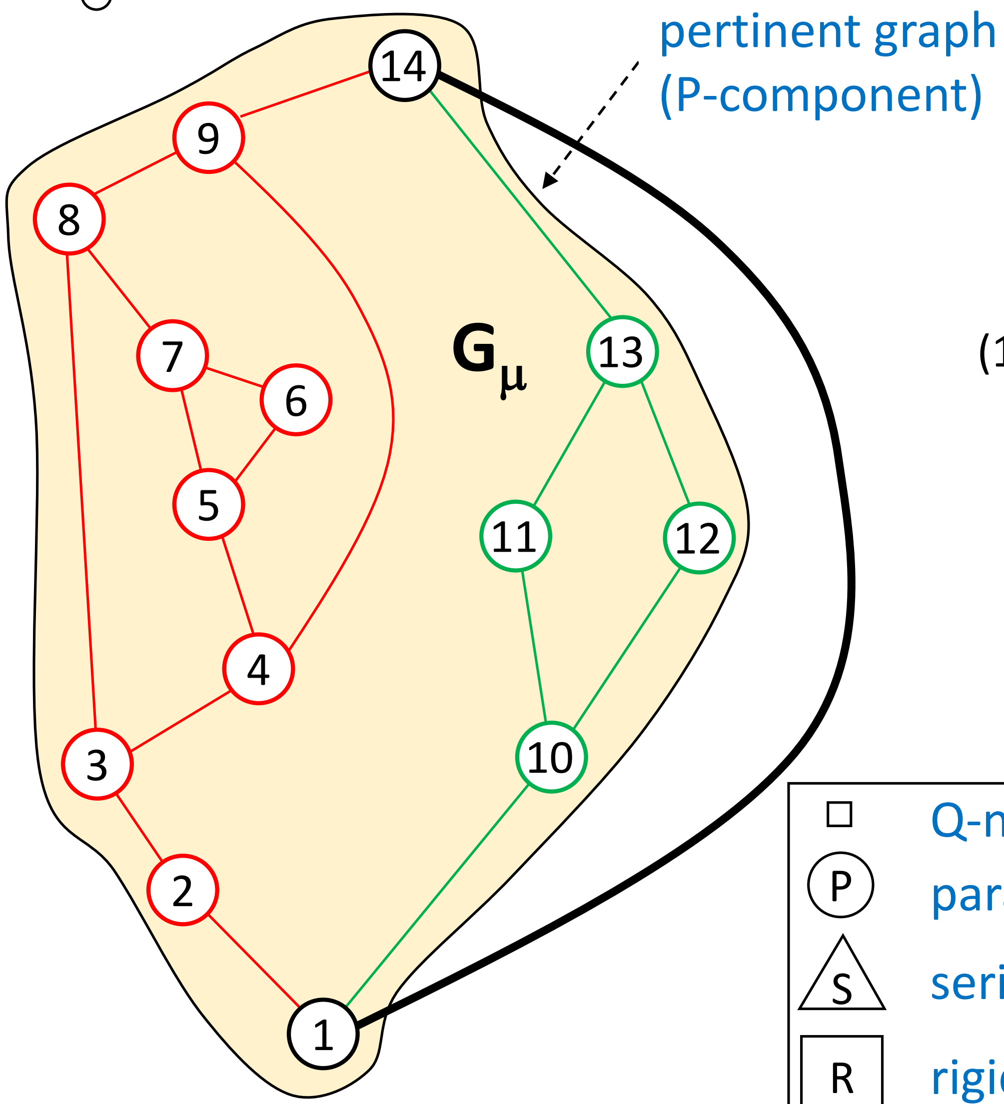
SPQR-trees

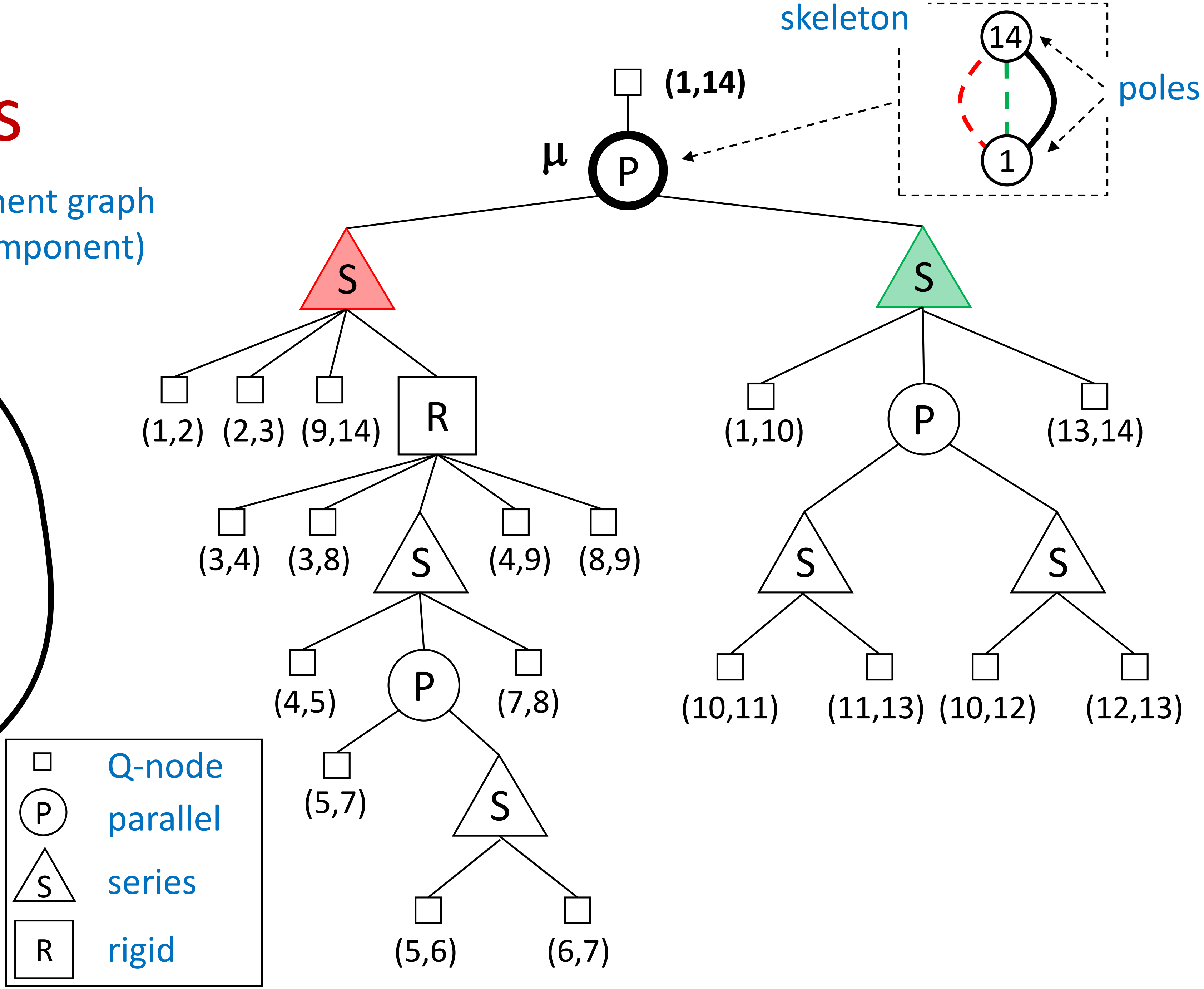


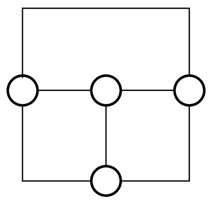
14
9
8
7
6
5
4
3
2
1
13
11
12
10
(1,14)
P
14
1
S
S
(1,2)
(2,3)
(9,14)
R
(1,10)
P
(13,14)
(3,4)
(3,8)
S
(4,9)
(8,9)
S
S
(4,5)
P
(7,8)
(10,11)
(11,13)
(10,12)
(12,13)
(5,7)
S
(5,6)
(6,7)
Q-node
P
parallel
S
series
R
rigid



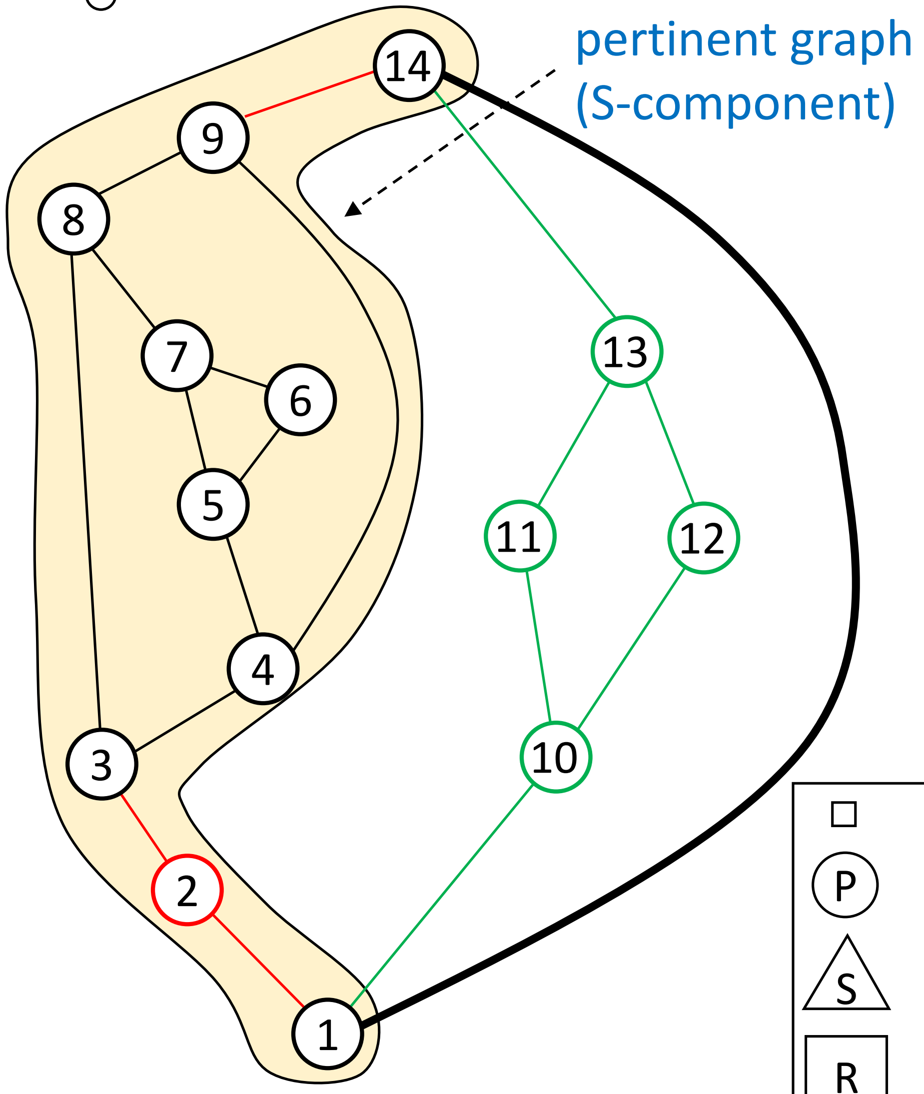
SPQR-trees

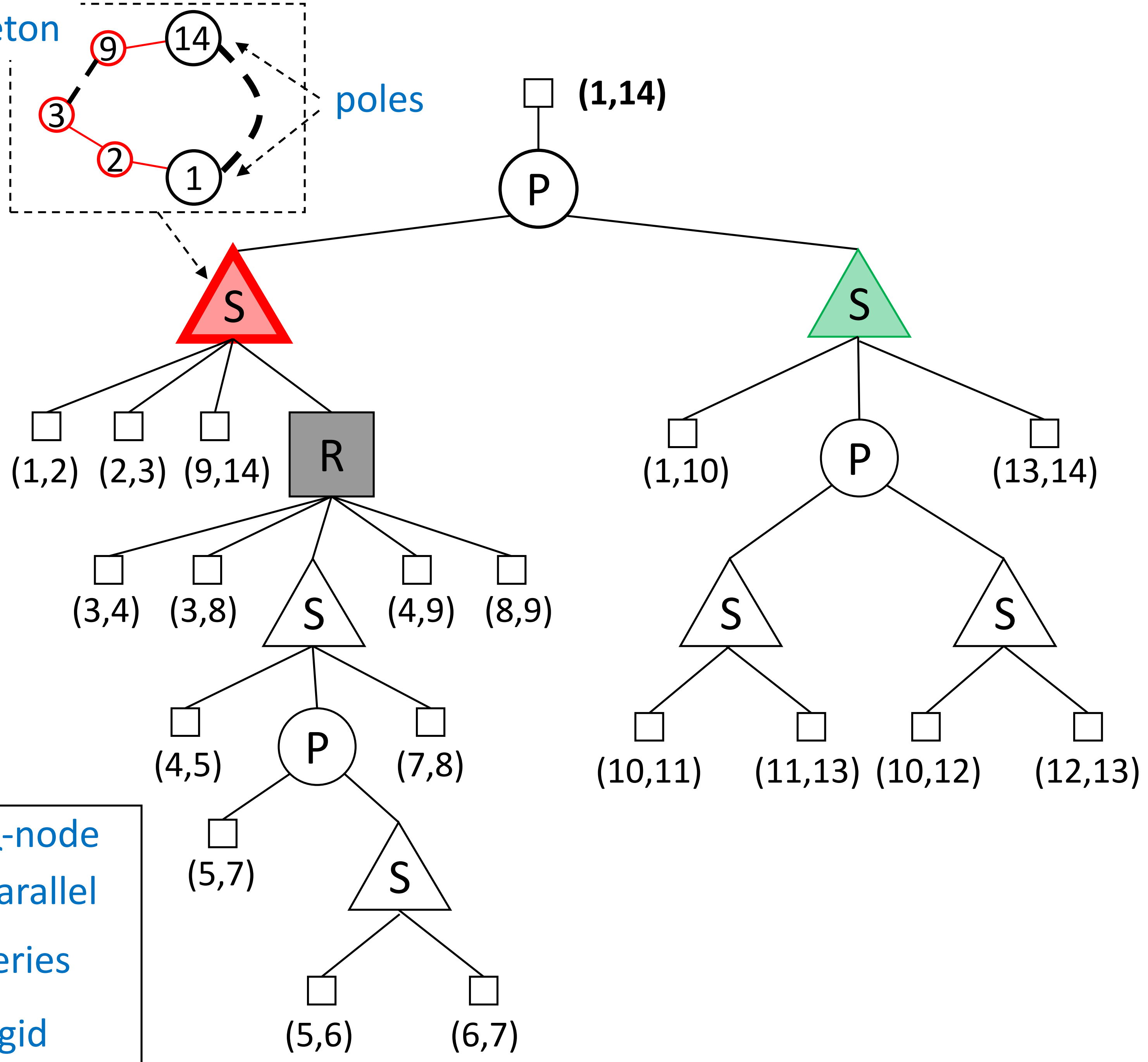

pertinent graph
(P-component)
G_μ
14
9
8
7
6
5
4
3
2
1
13
11
12
10

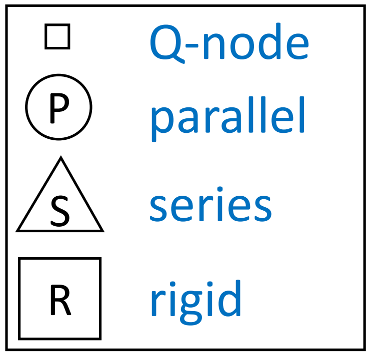

skeleton
poles
14
1
(1,14)
μ
P
S
S
(1,2)
(2,3)
(9,14)
R
(3,4)
(3,8)
S
(4,9)
(8,9)
(4,5)
P
(7,8)
(5,7)
S
(5,6)
(6,7)
(1,10)
P
(13,14)
S
S
(10,11)
(11,13)
(10,12)
(12,13)
Q-node
P
parallel
S
series
R
rigid

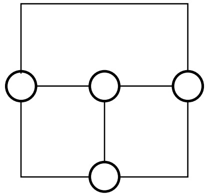


SPQR-trees

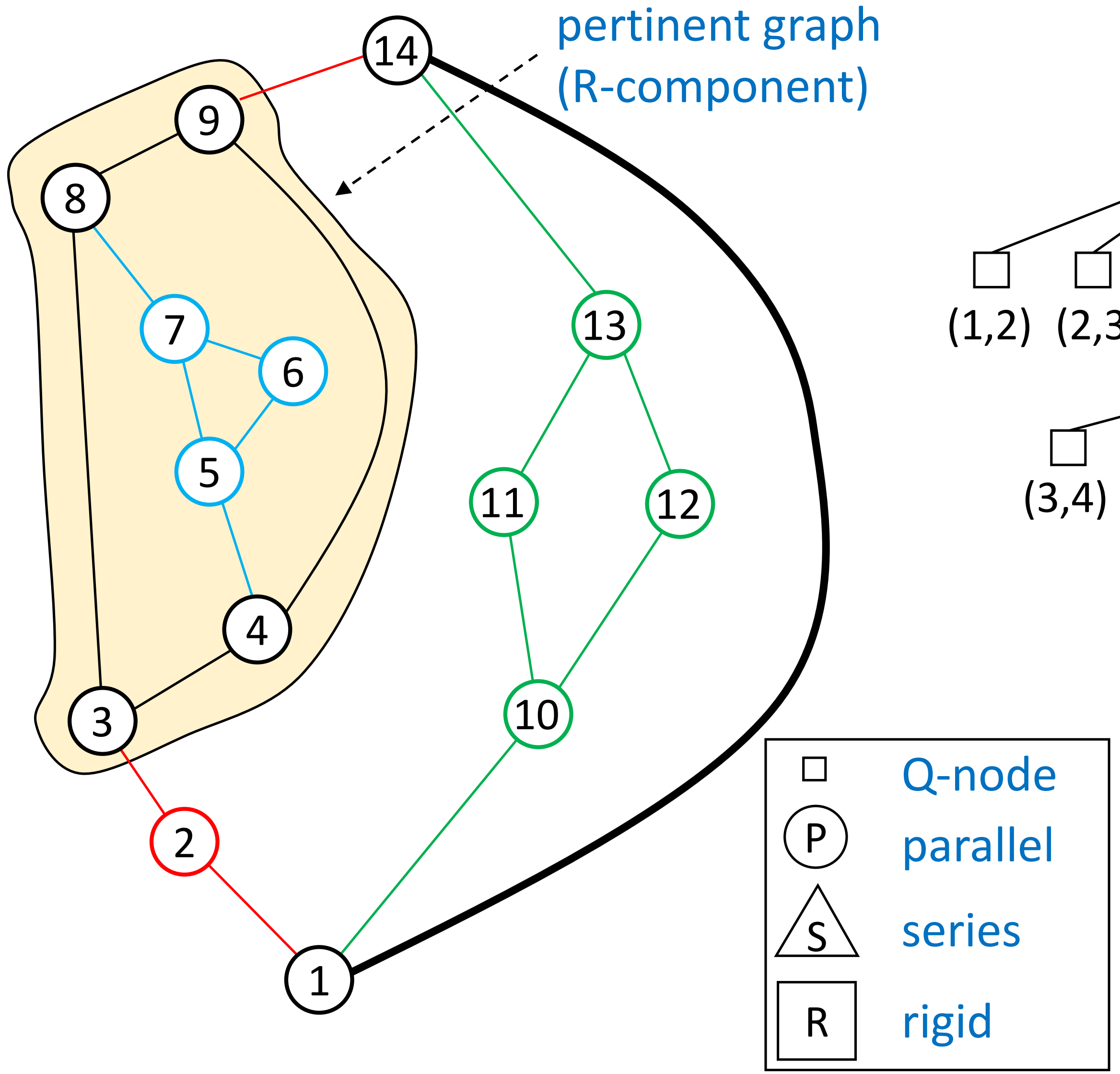

pertinent graph
(S-component)
14
9
8
7
6
5
4
3
2
1
13
11
12
10


skeleton
poles
9
14
3
2
1
(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)

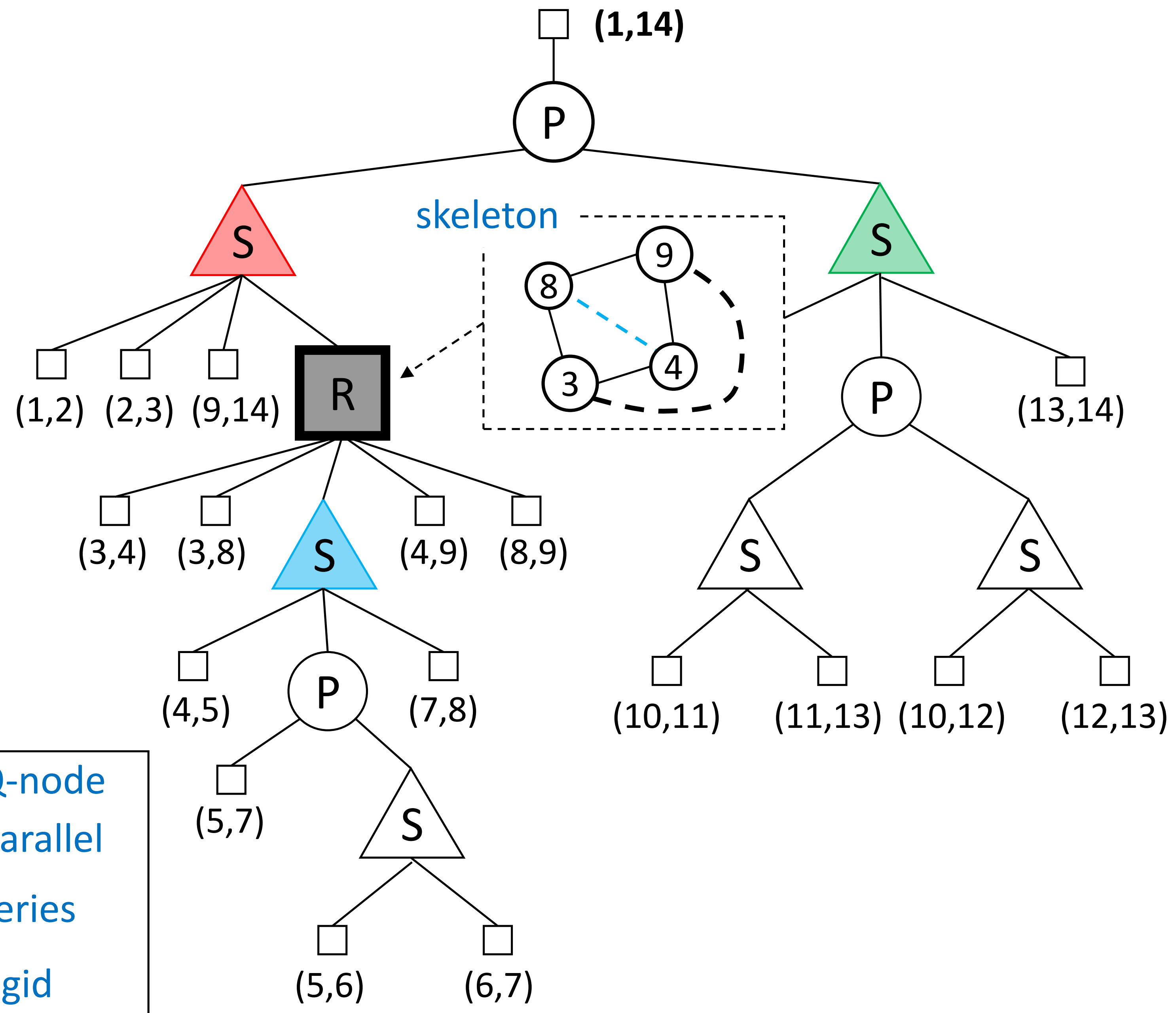

Q-node
P
parallel
S
series
R
rigid



SPQR-trees

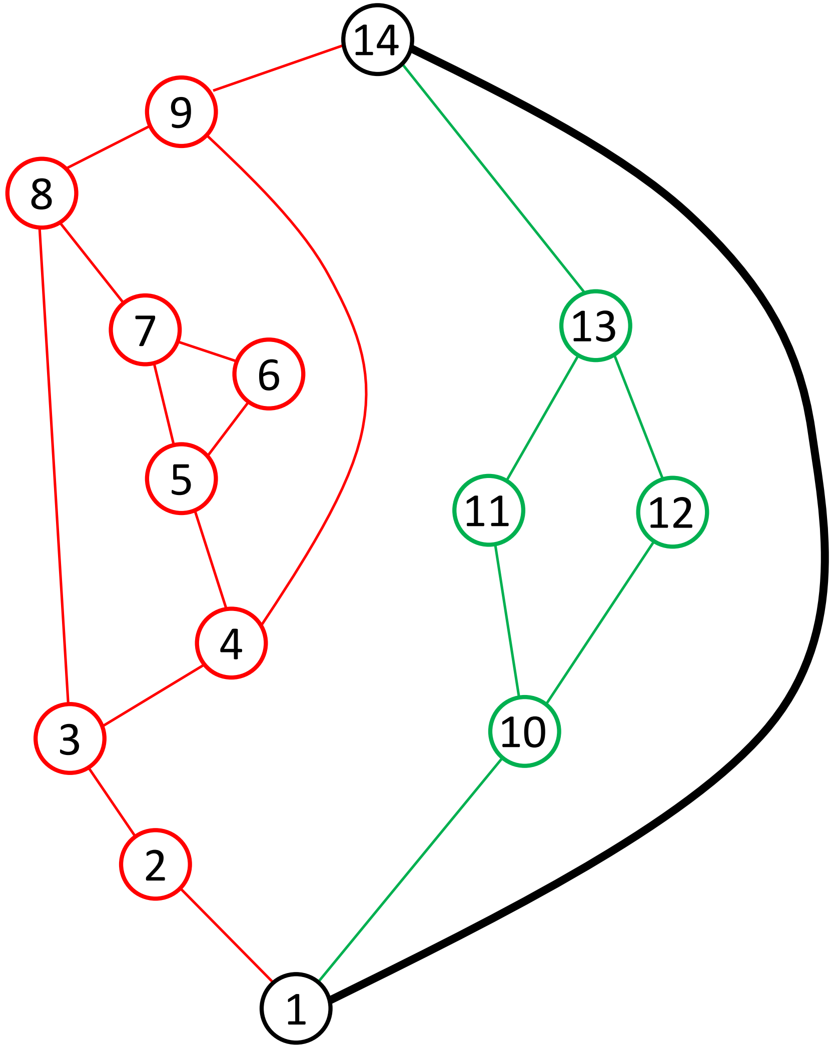


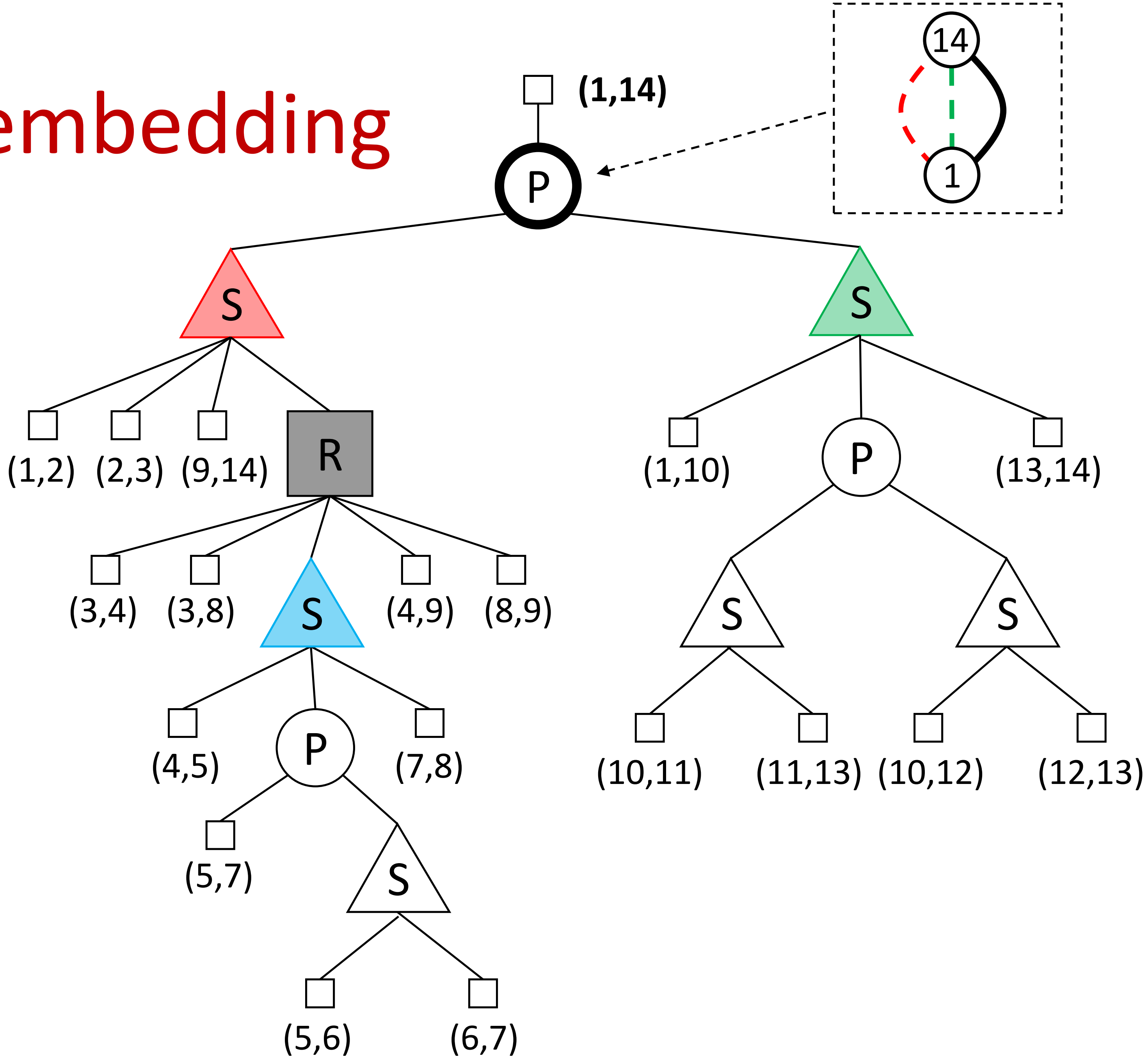
pertinent graph
(R-component)
14
9
8
7
6
5
4
3
2
1
13
11
12
10
Q-node
P
parallel
S
series
R
rigid

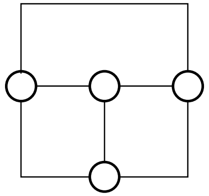


(1,14)
P
S
skeleton
9
8
4
3
S
(1,2)
(2,3)
(9,14)
R
P
(13,14)
(3,4)
(3,8)
S
(4,9)
(8,9)
S
S
(4,5)
P
(7,8)
(10,11)
(11,13)
(10,12)
(12,13)
(5,7)
S
(5,6)
(6,7)

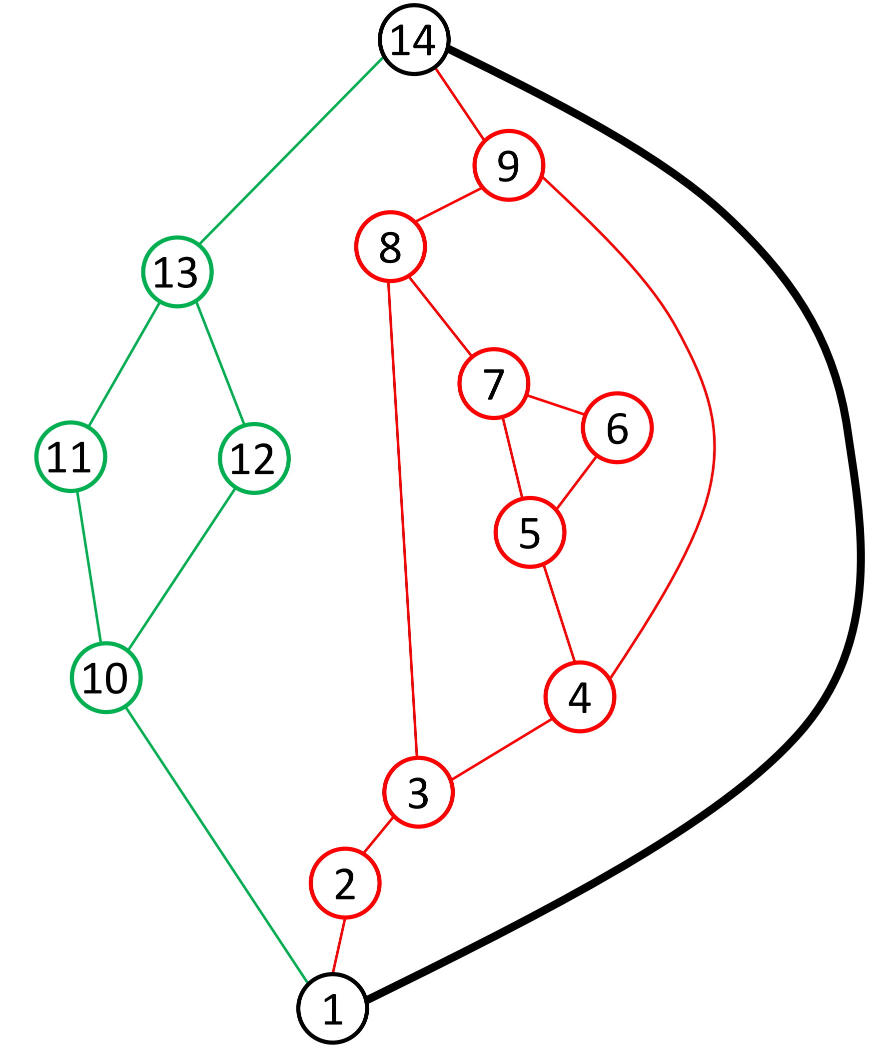
Changing the embedding

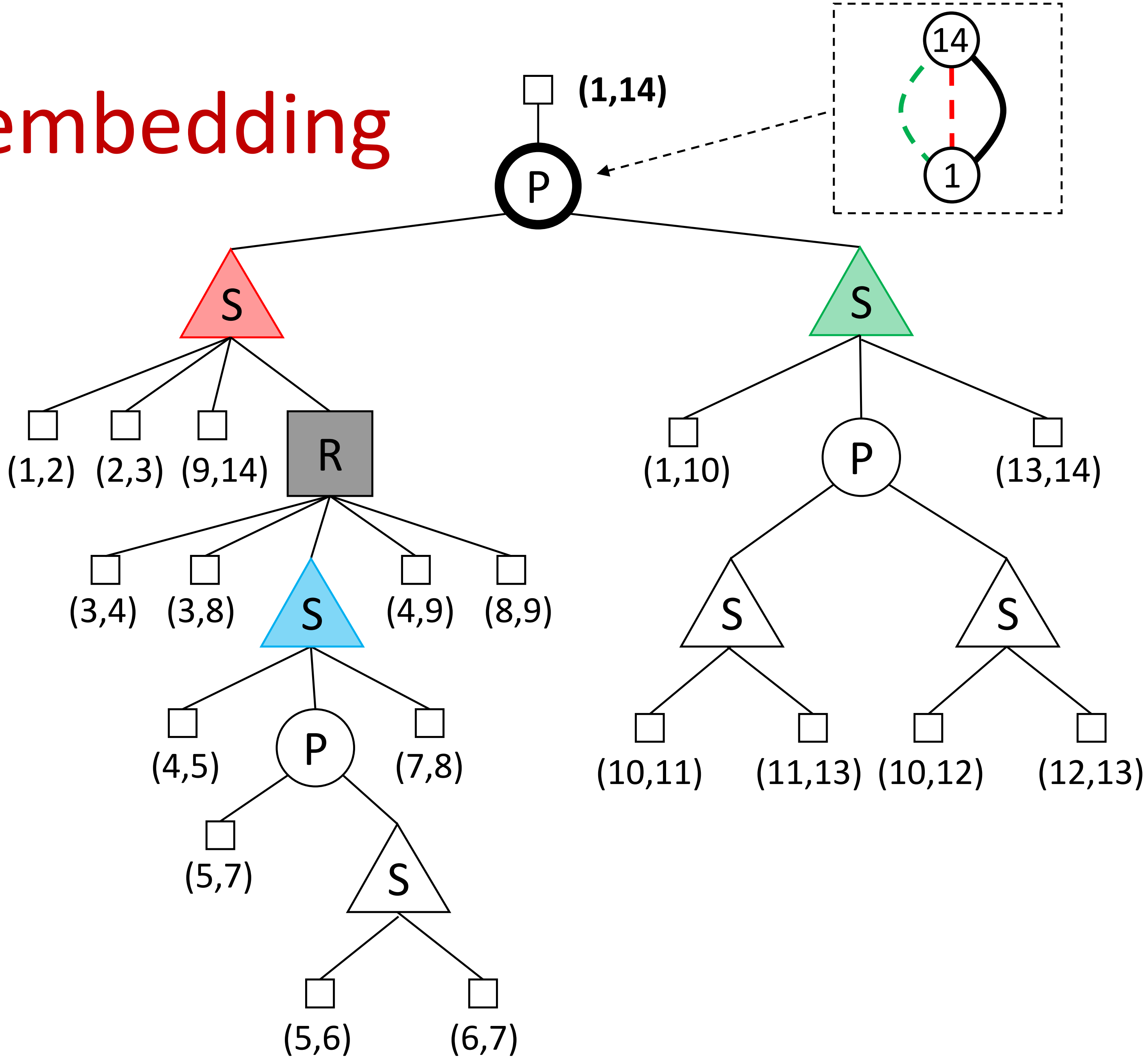




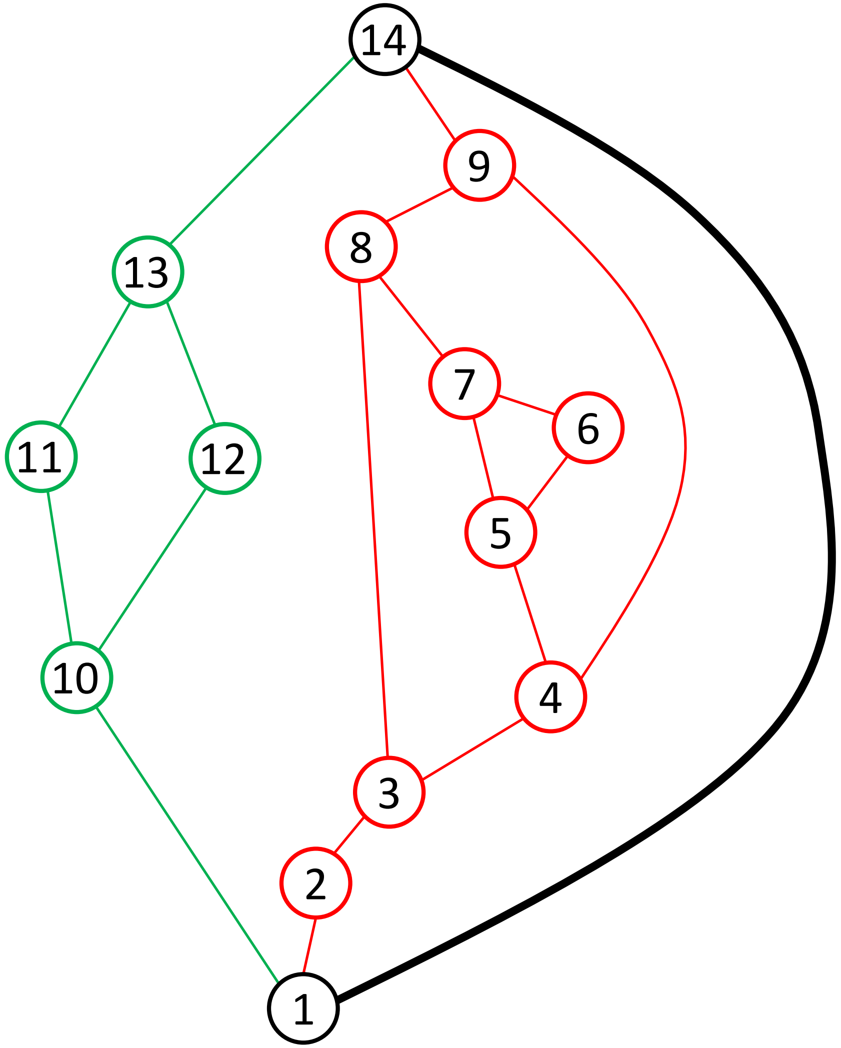


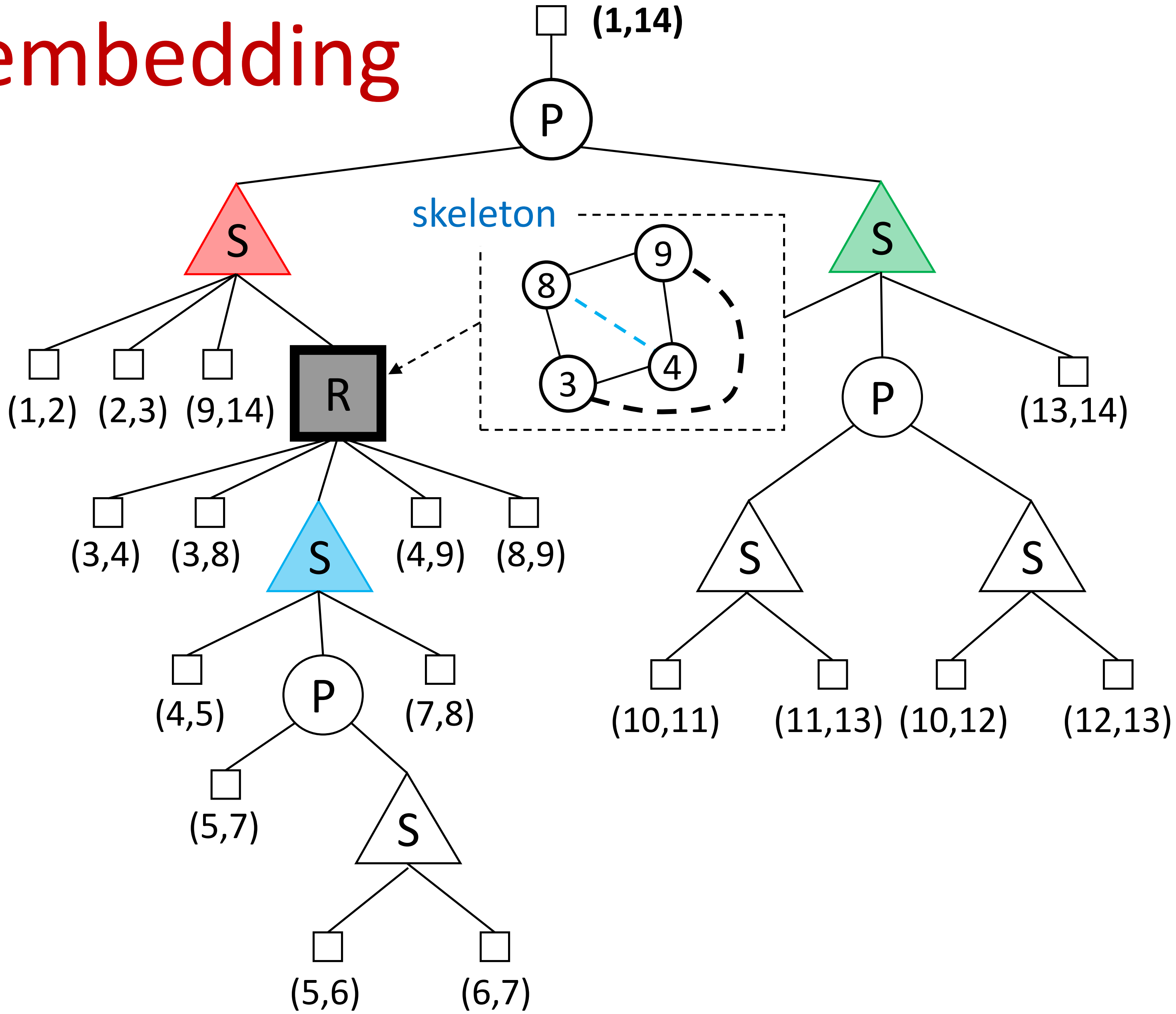
Changing the embedding



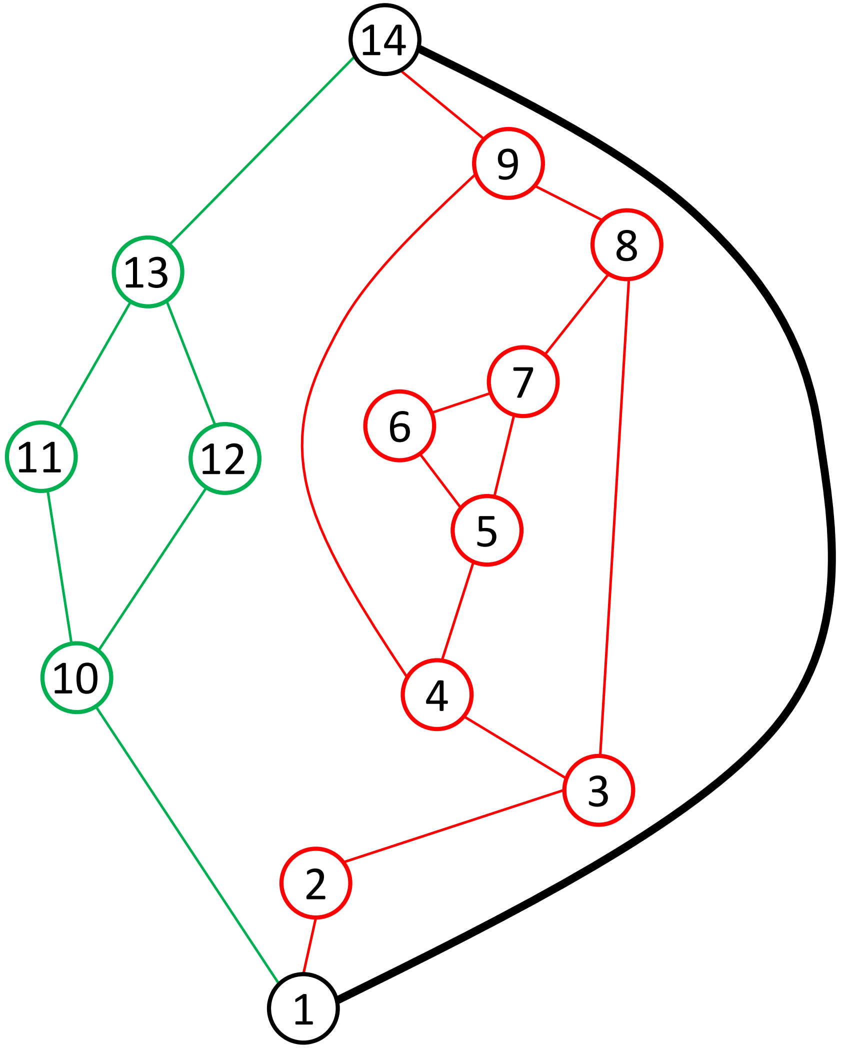


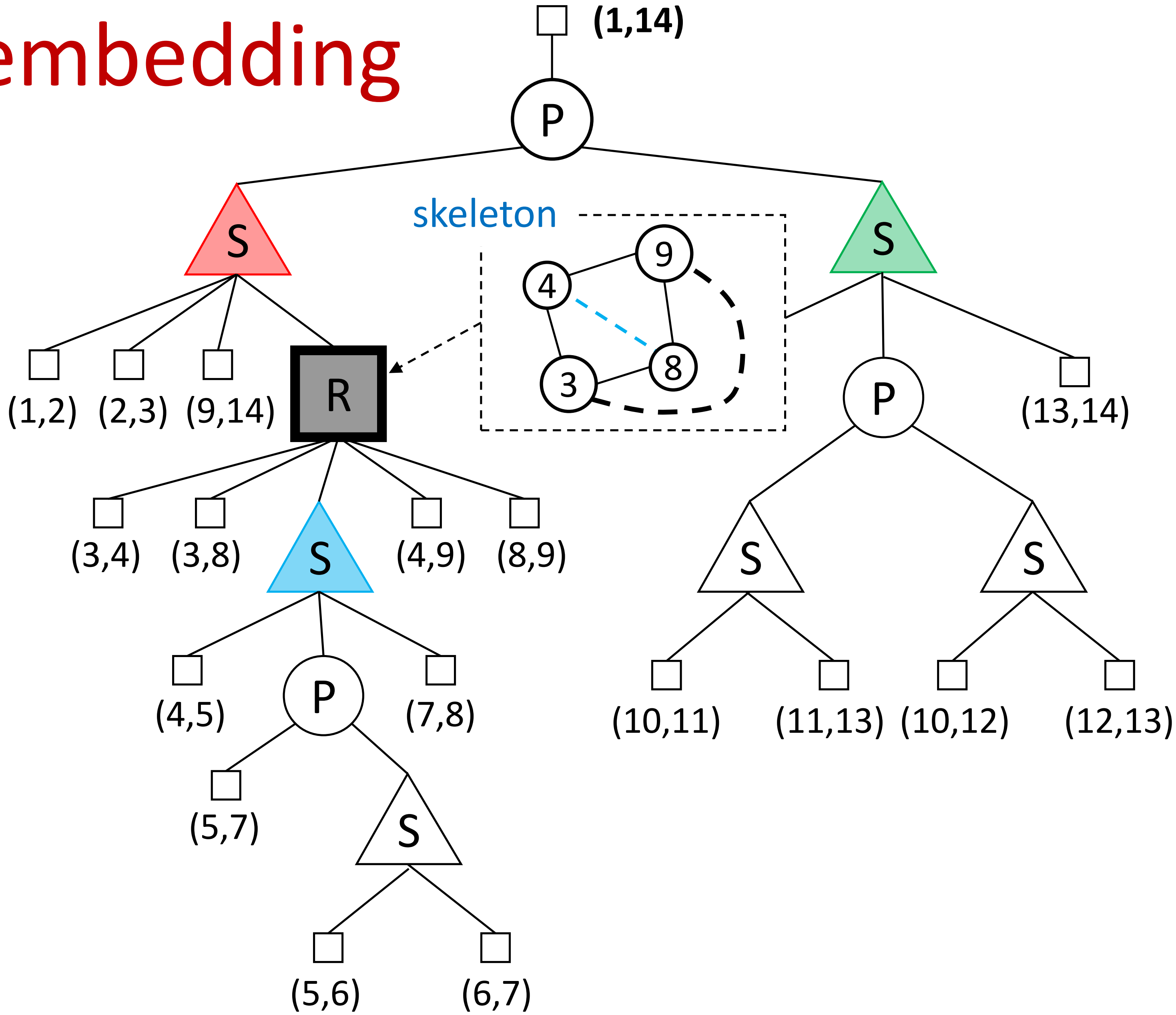
Changing the embedding

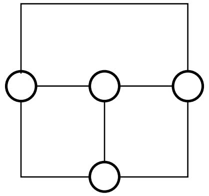




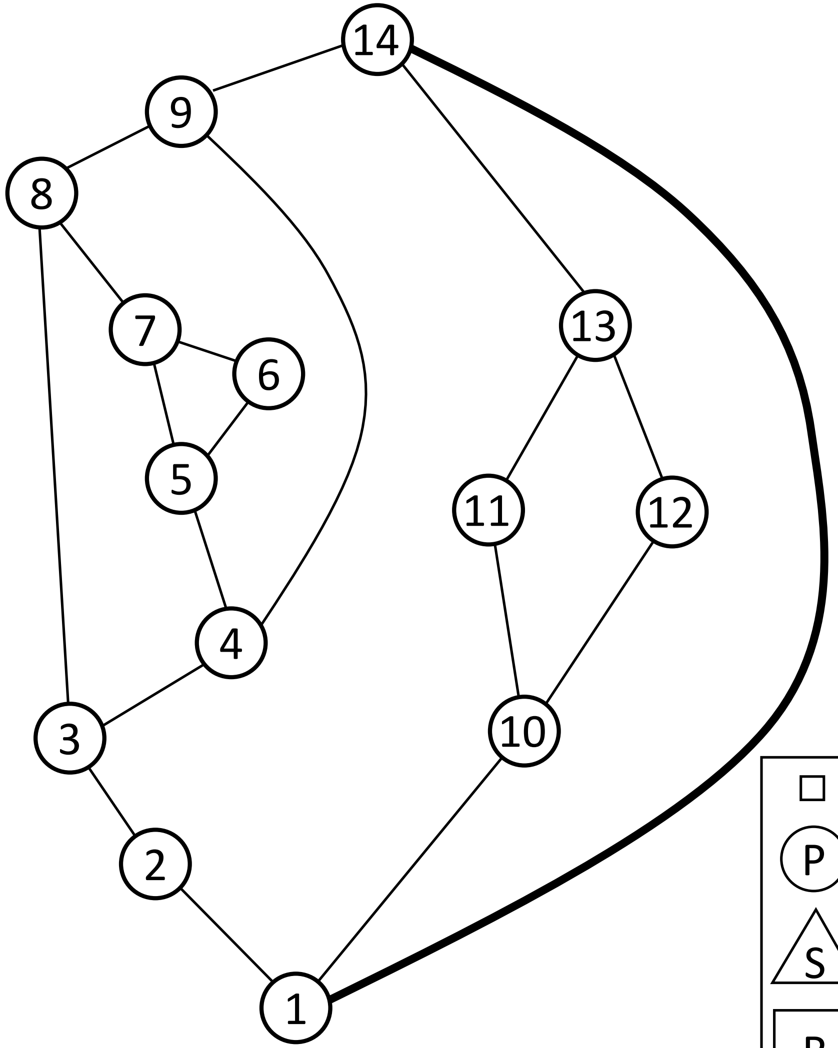
Changing the embedding

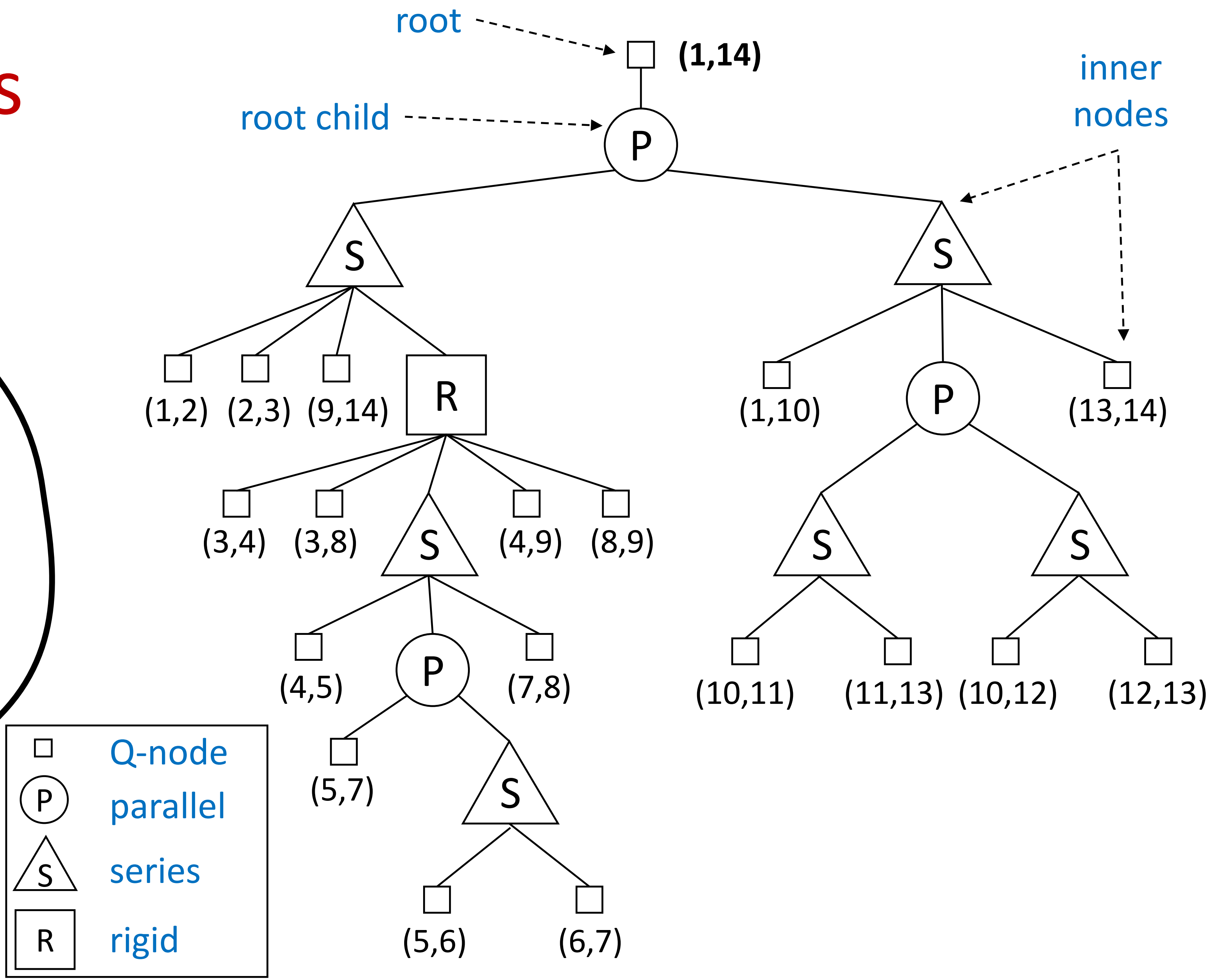


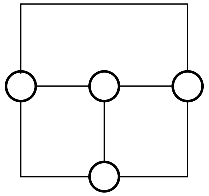




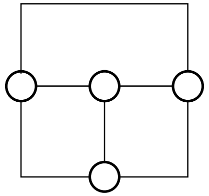
SPQR-trees


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



\end{SPQR-trees}



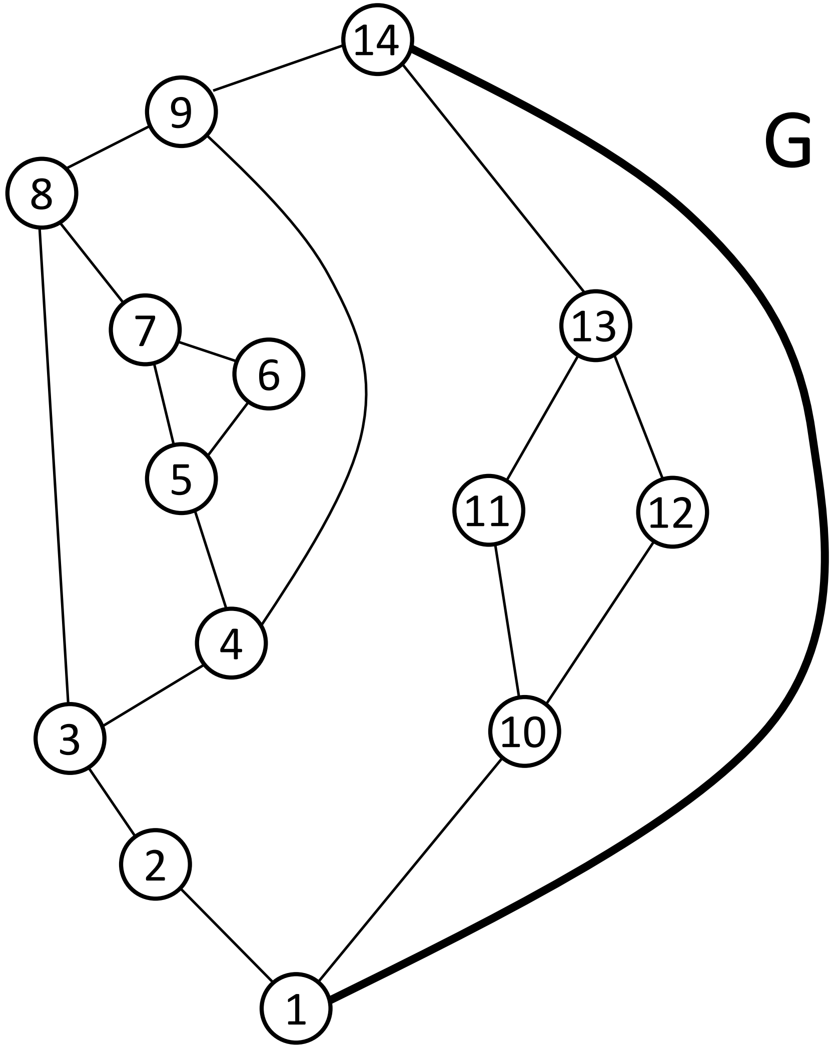
Orthogonal representations

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

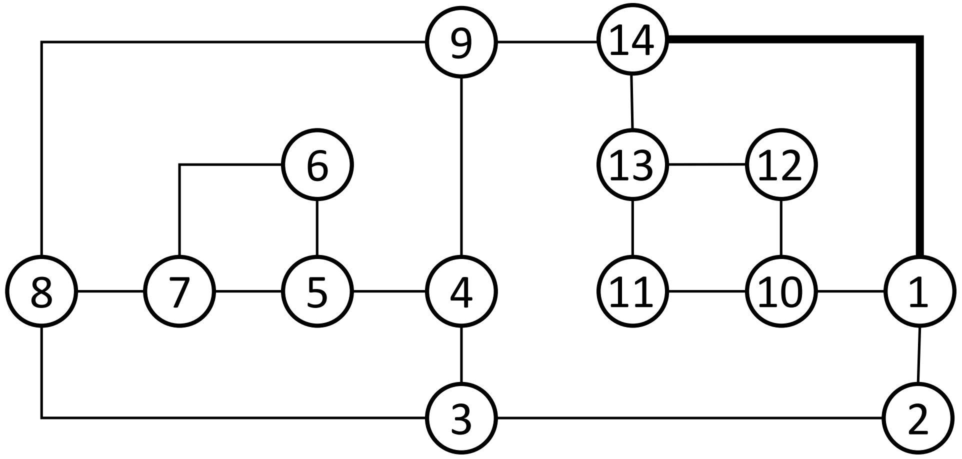
- a drawing of an orthogonal representation can be computed in linear time [Tamassia '97]

orthogonal component = orthogonal representation H_μ of a component G_μ

Orthogonal components: example

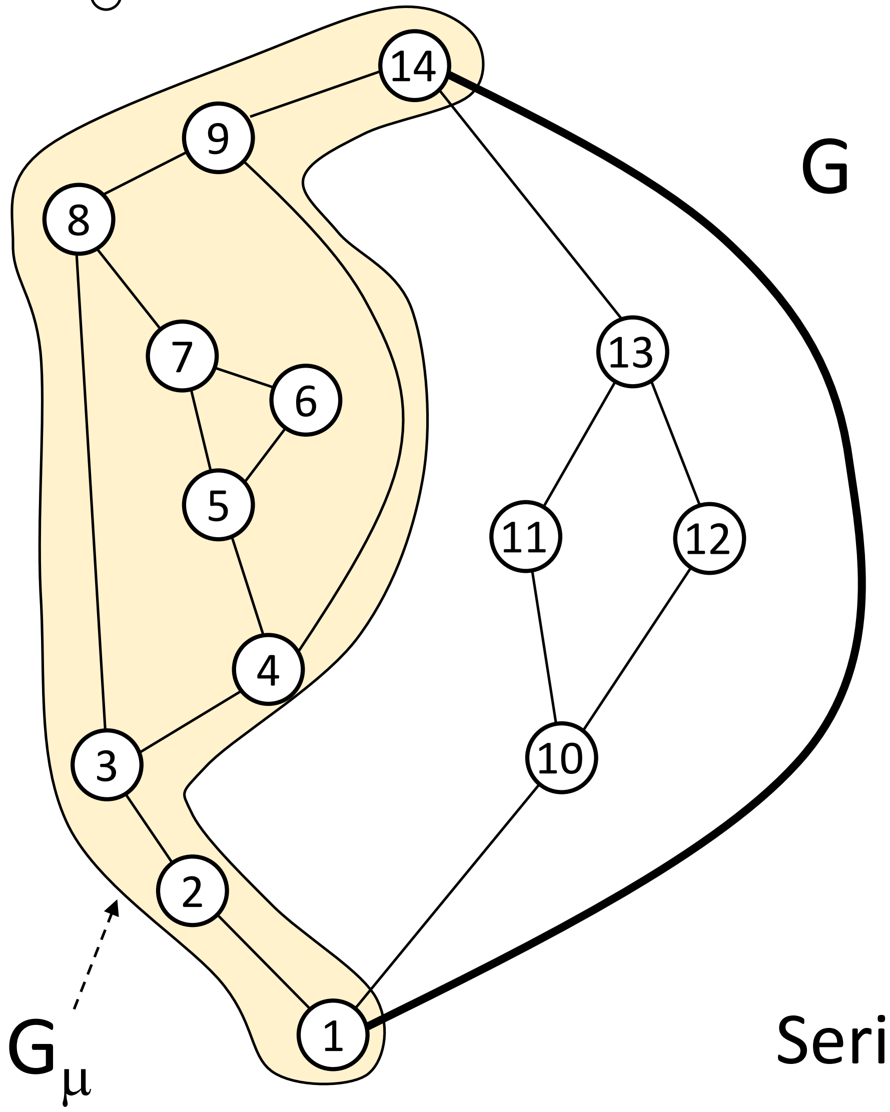


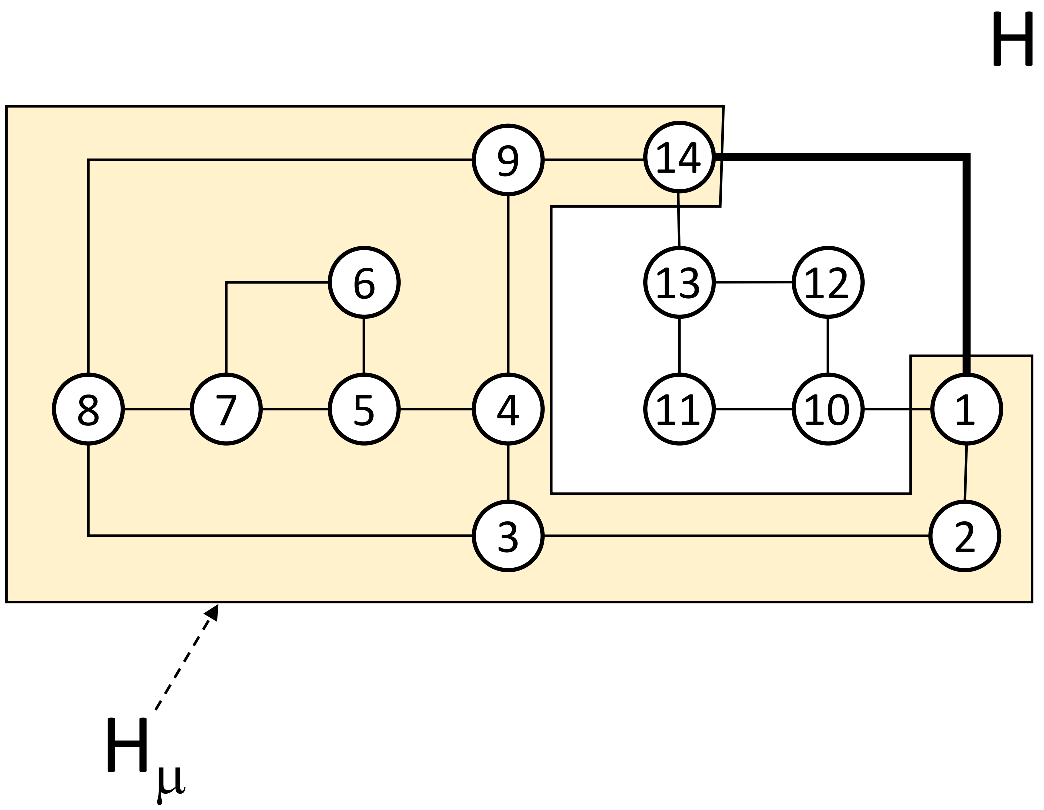
G



H

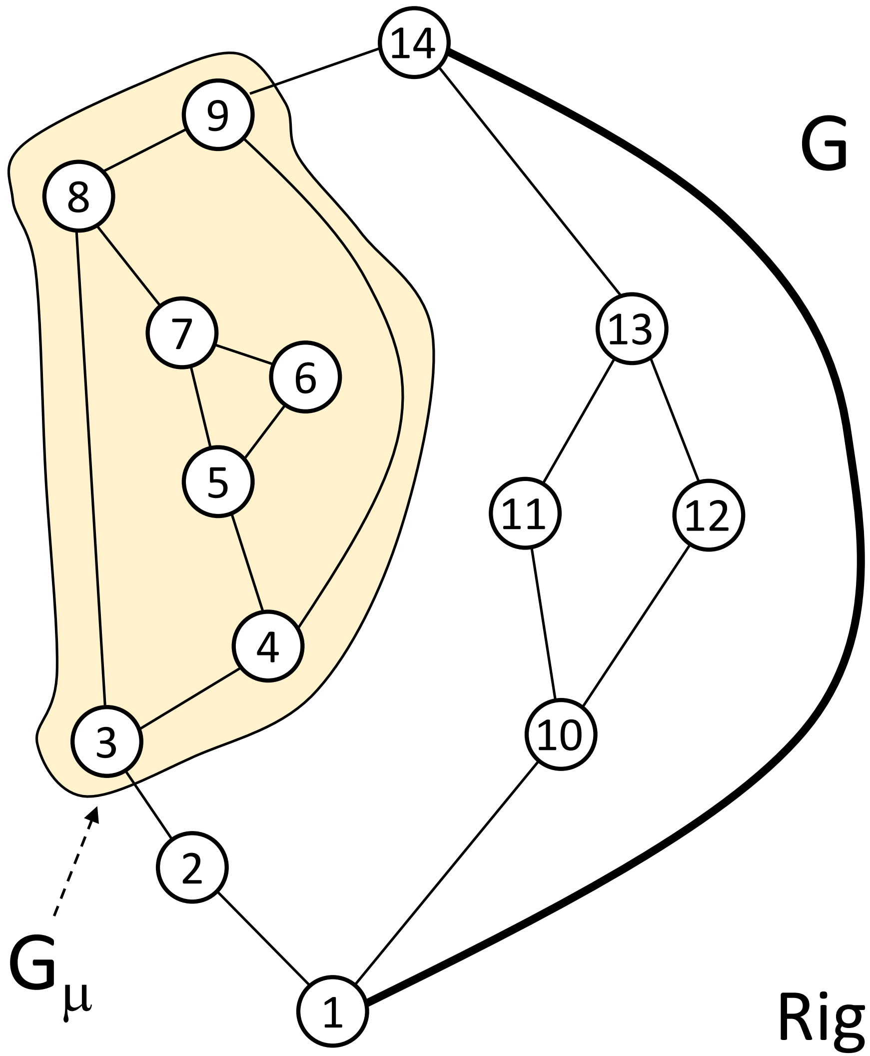
Orthogonal components: examples

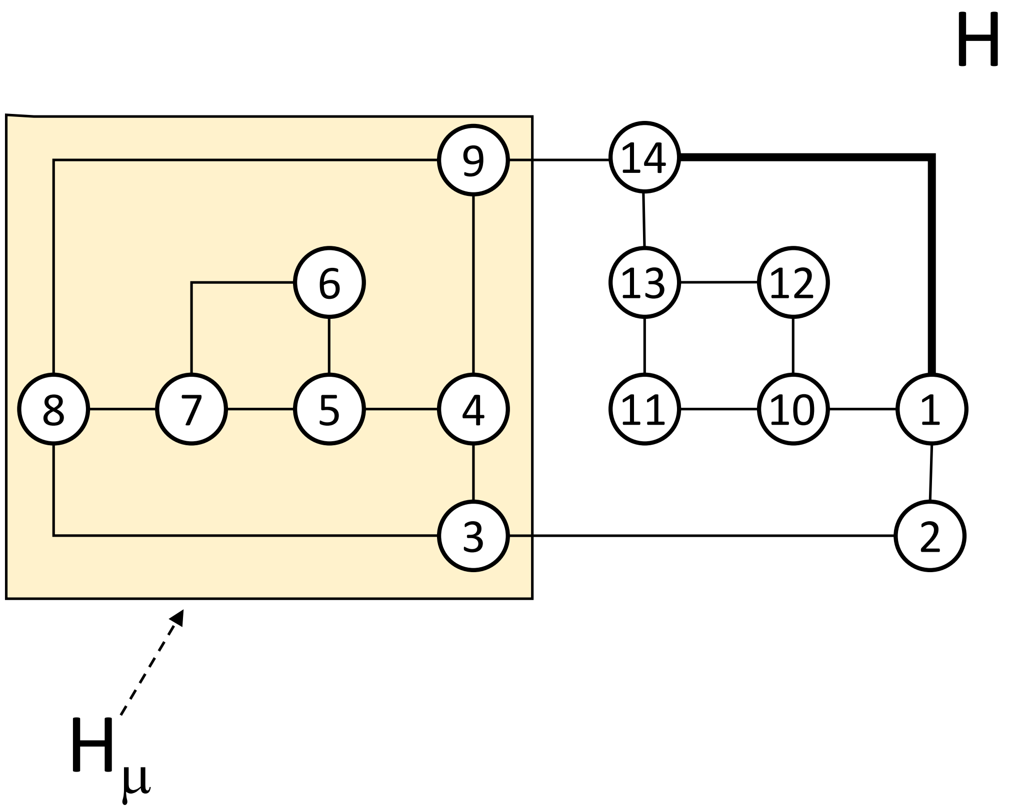




Series (orthogonal) component

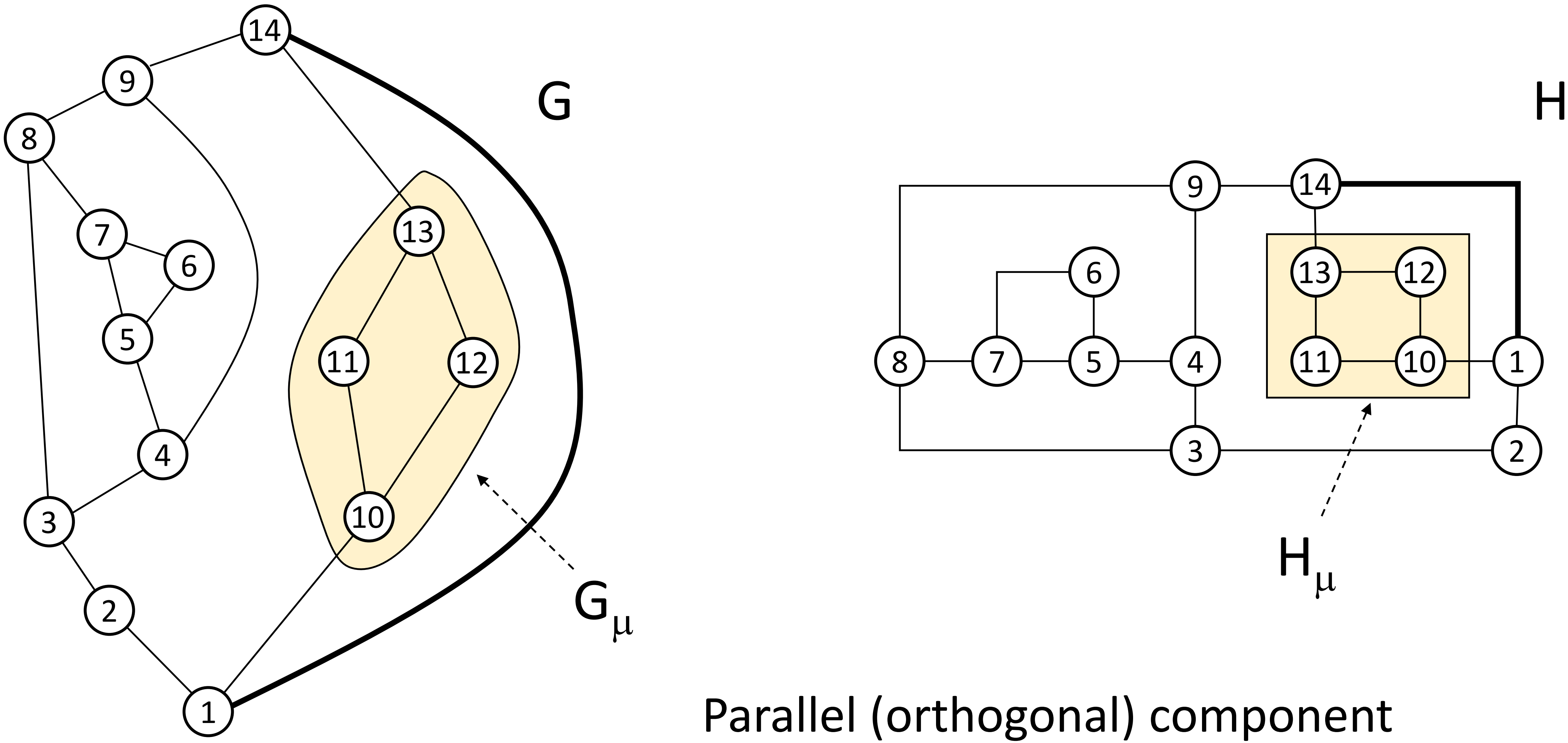
Orthogonal components: examples



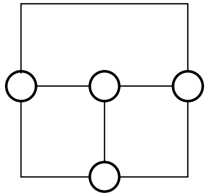


Rigid (orthogonal) component

Orthogonal components: examples

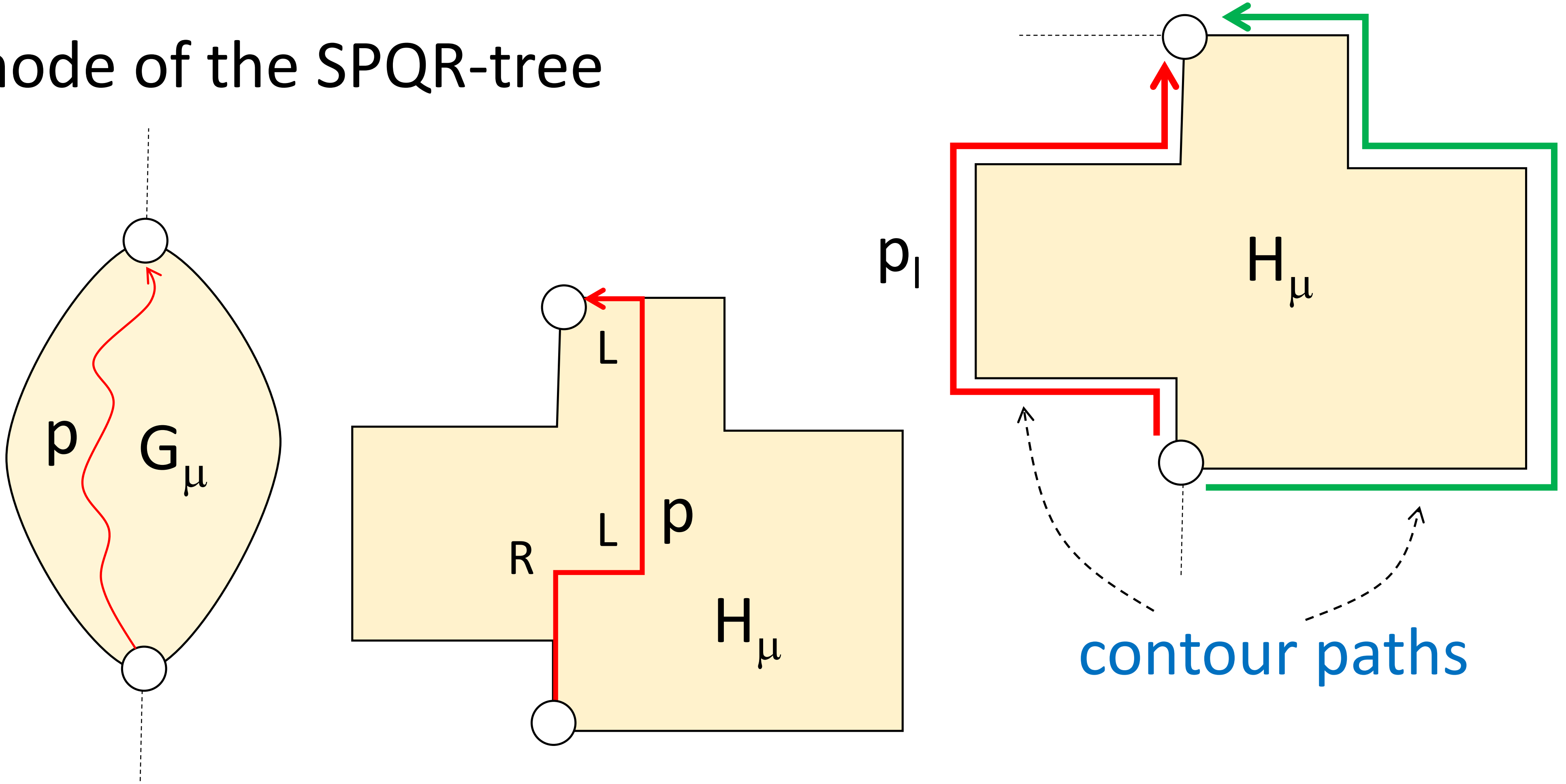


Parallel (orthogonal) component



Turn number and contour paths

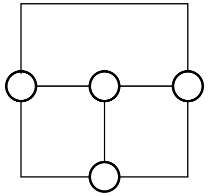
μ = node of the SPQR-tree



$t(p_l) = 0$

$t(p_r) = 2$

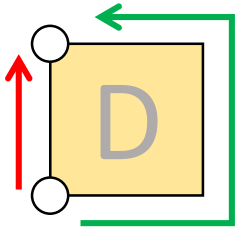
t(p) = turn number = |#left turns – # right turns| (along p)



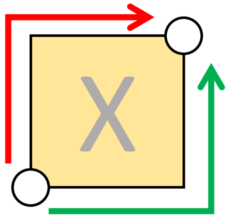
P- and R-components: shapes

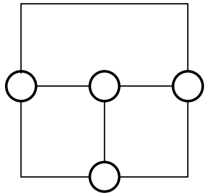
μ = P-node or R-node

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



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

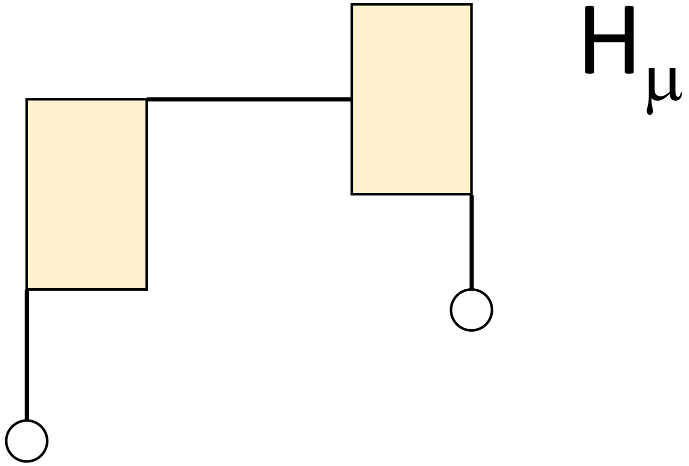


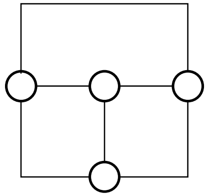


Inner S-components: spirality

μ = inner S-node

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

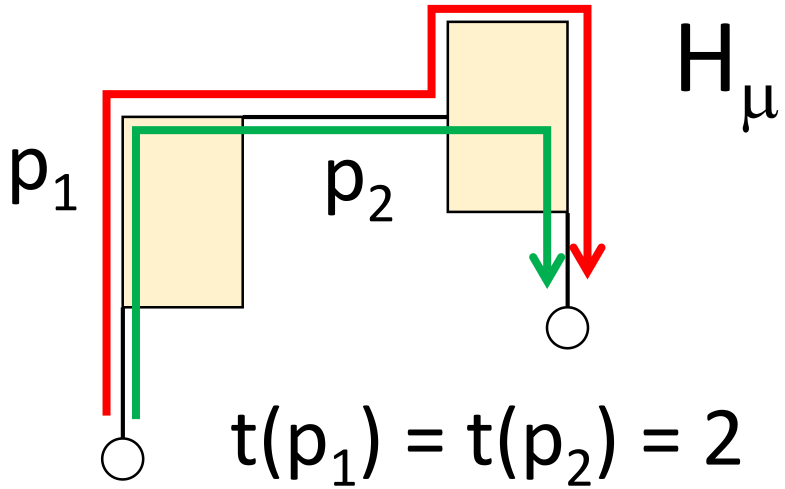




Inner S-components: spirality

μ = inner S-node

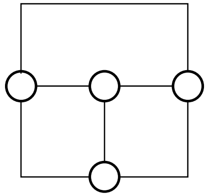
Lemma. All paths between the poles of a H_μ have the same turn number



$t(p) = k$

H_μ is k-spiral

H_μ has spirality k

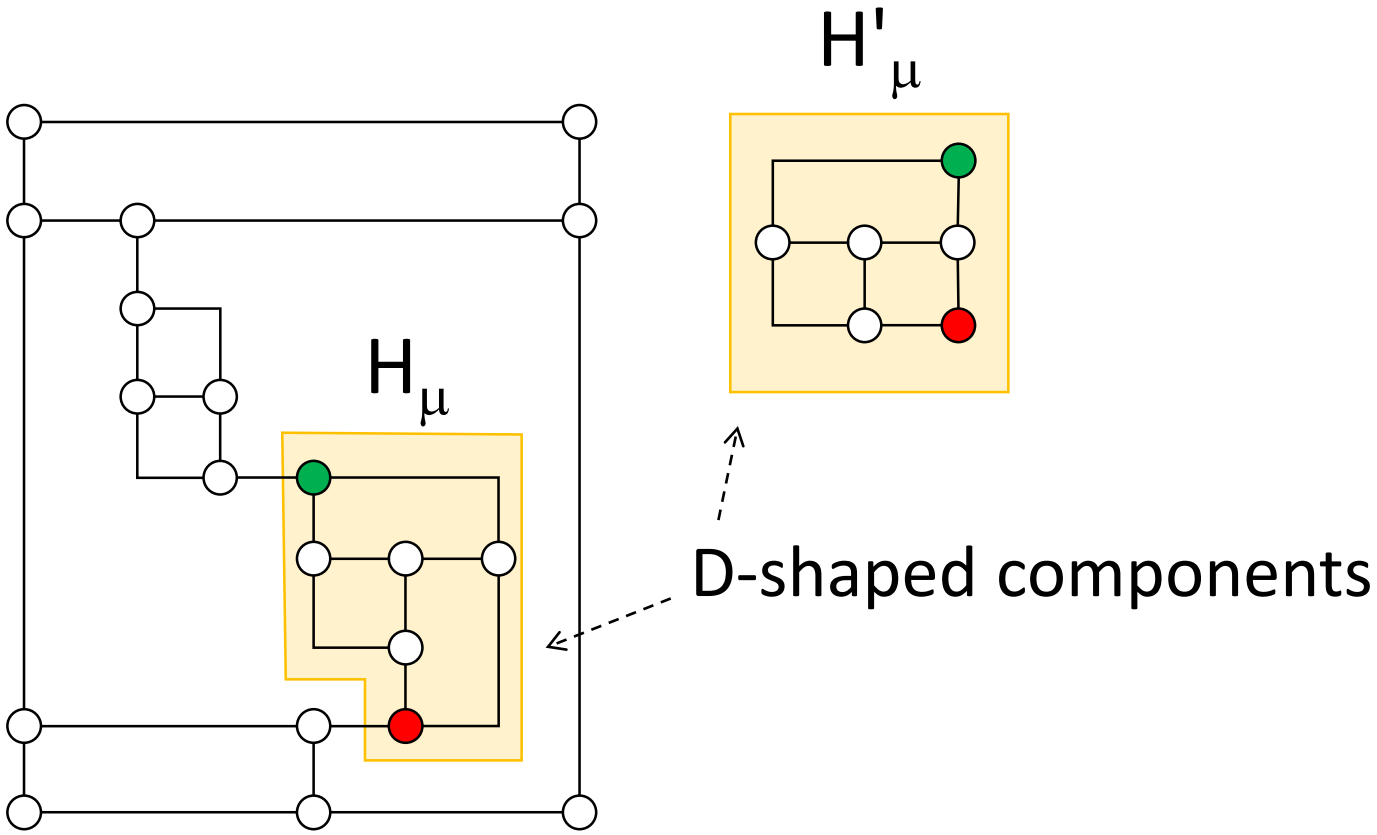


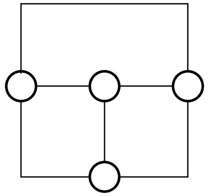
Equivalent orthogonal components

- 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 are both D-shaped or both X-shaped
  - μ is an S-node and H_μ, H'_μ have the same spirality

Equivalent orthogonal components

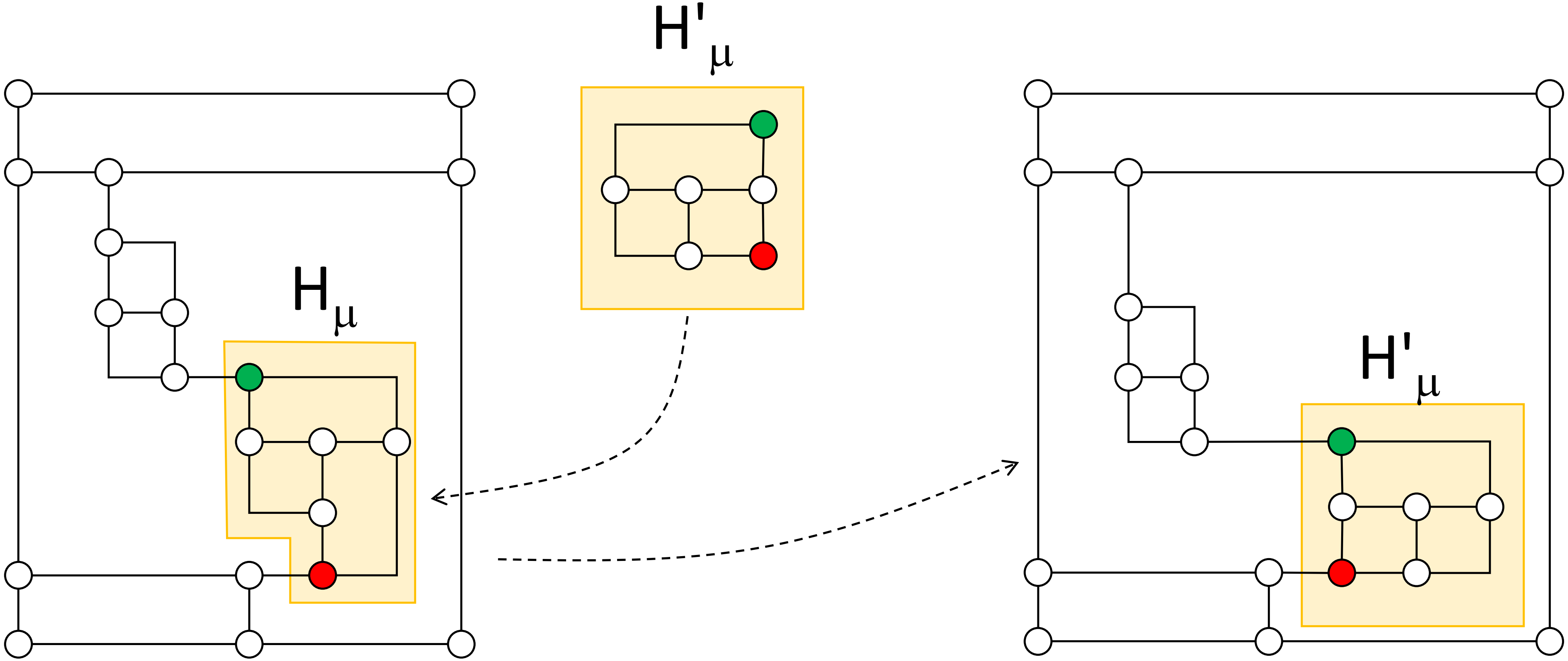
Theorem (substitution). Equivalent orthogonal components are interchangeable





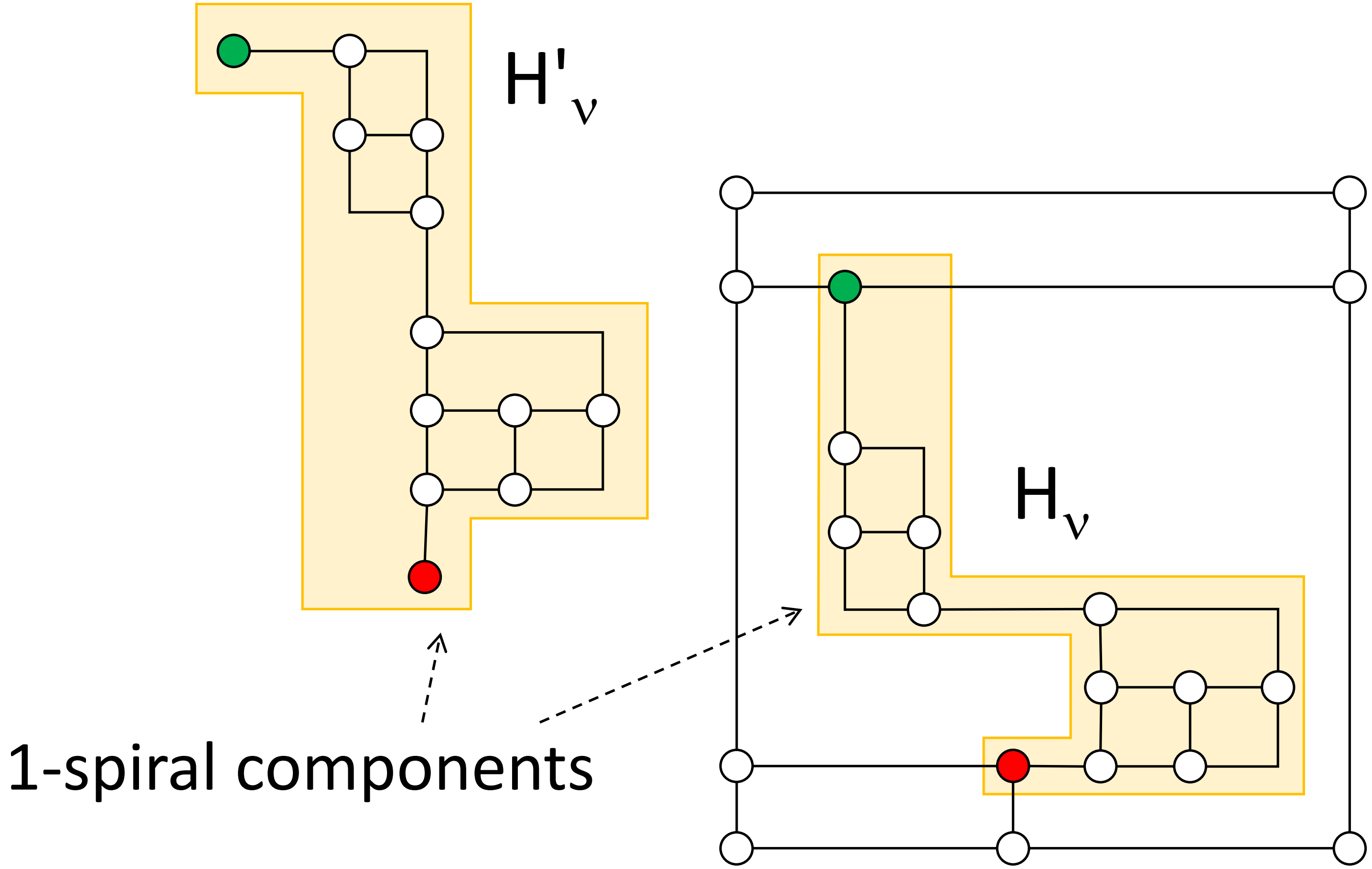
Equivalent orthogonal components

Theorem (substitution). Equivalent orthogonal components are interchangeable



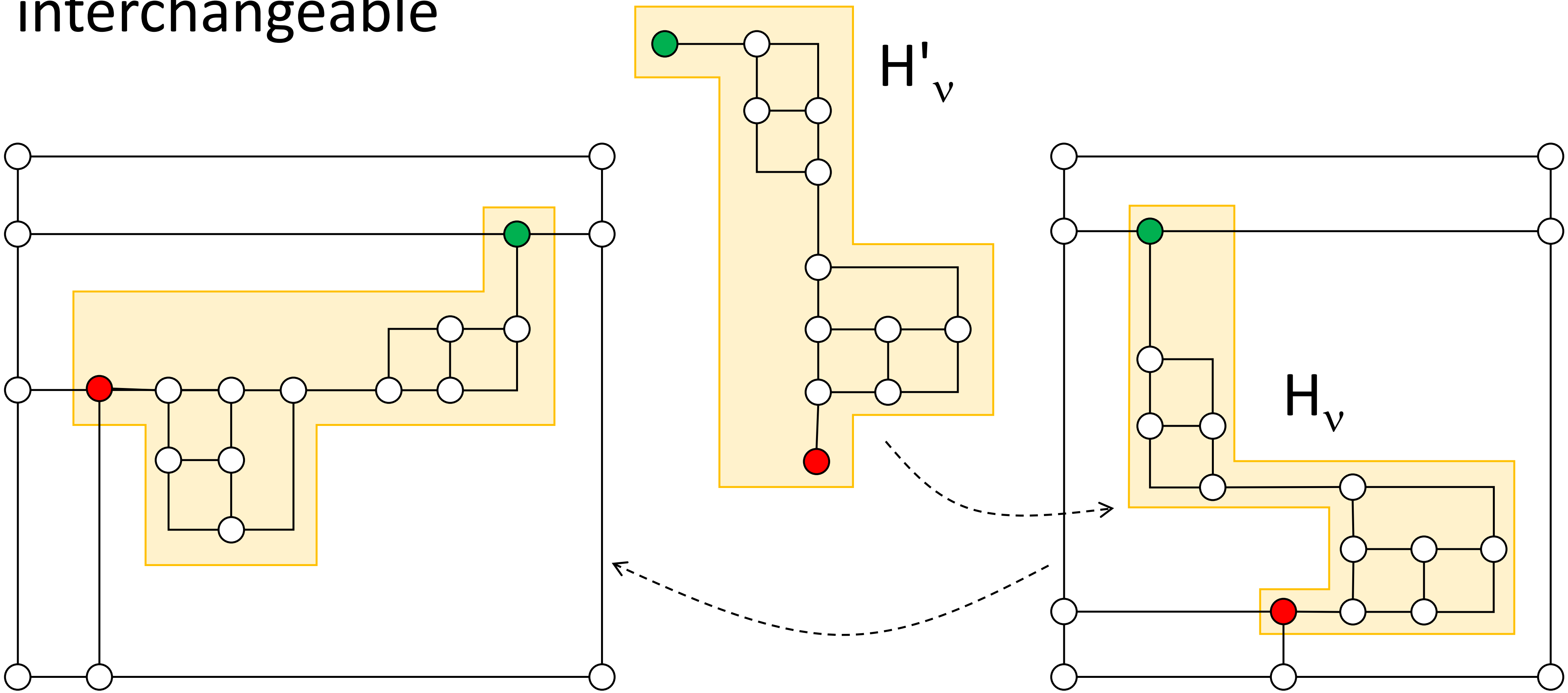
Equivalent orthogonal components

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
- **O3.** every inner 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
- **O3.** every inner S-component has spirality at most 4

proof: 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

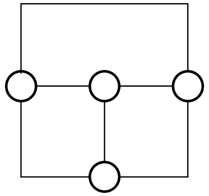
O3. every inner S-component has spirality at most 4

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

Algorithm

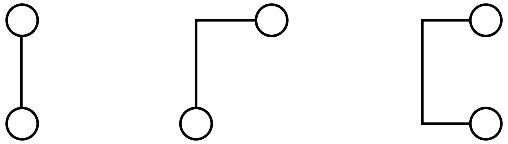
- **input**: biconnected planar 3-graph G with a reference edge *e*
- **output**: bend-min representation H of G with *e* on the external face

1. compute the SPQR-tree T of G with respect to *e*
2. visit the nodes μ of T bottom-up:
   - **μ inner node** $\Rightarrow$ store in μ a set of candidate bend-min representations of G_μ- one for each distinct equivalence class, thanks to the substitution theorem
   - **μ the root child** $\Rightarrow$ 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

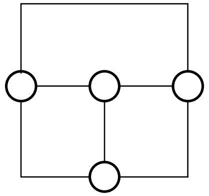


Candidate sets of inner nodes

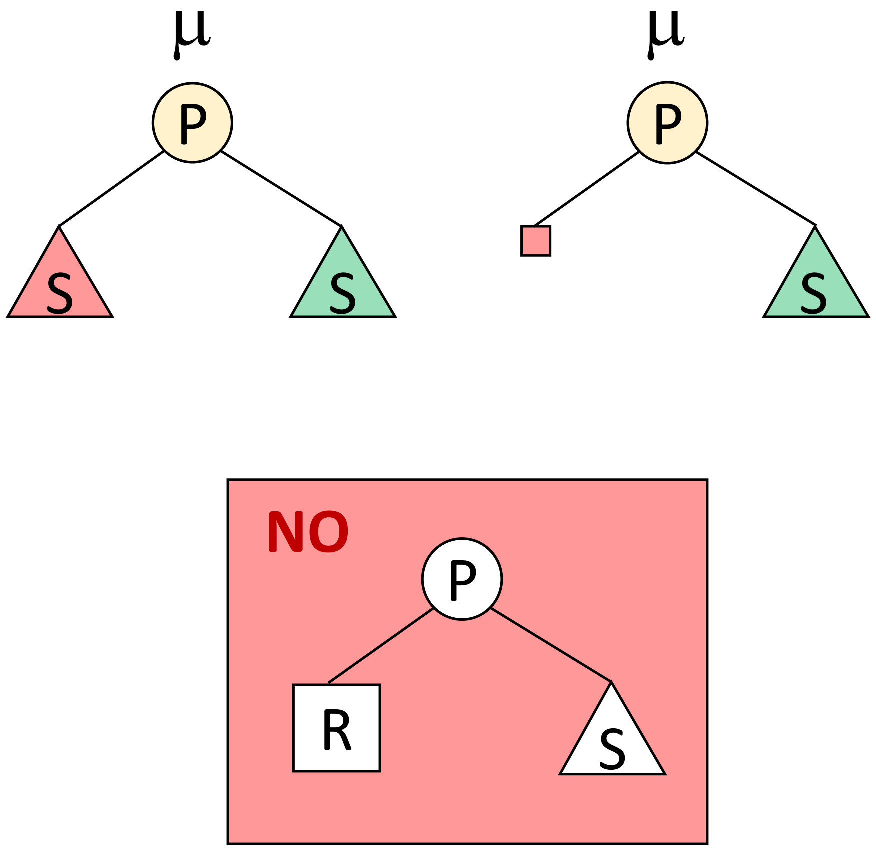
- **Q-node:** a representation for each number of bends in {0, 1, 2}
  - thanks to O1 of the key-lemma

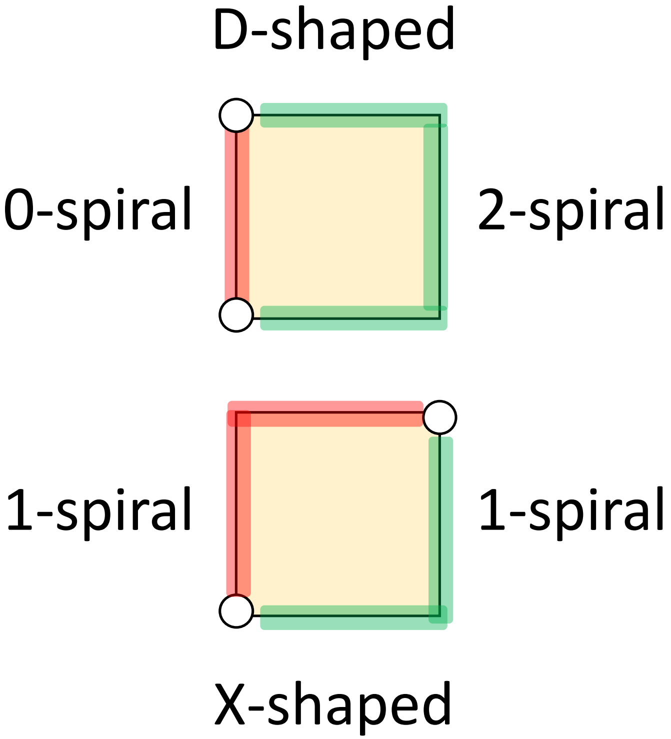


- **P/R-node:** the cheapest D-shaped and the cheapest X-shaped representations
  - thanks to O2 of the key-lemma
- **S-node:** the cheapest representation for each value of spirality in {0, 1, 2, 3, 4}
  - thanks to O3 of the key-lemma



Candidate set of an inner P-node

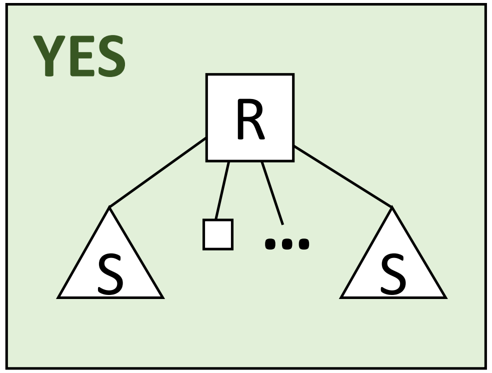


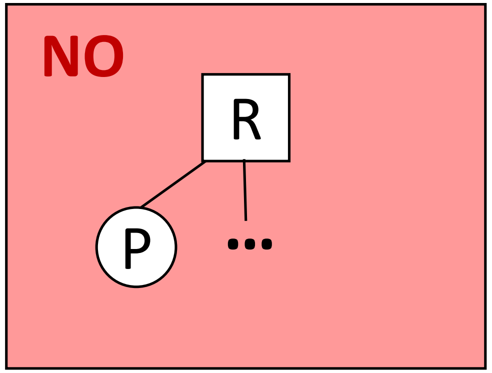


O(1) time

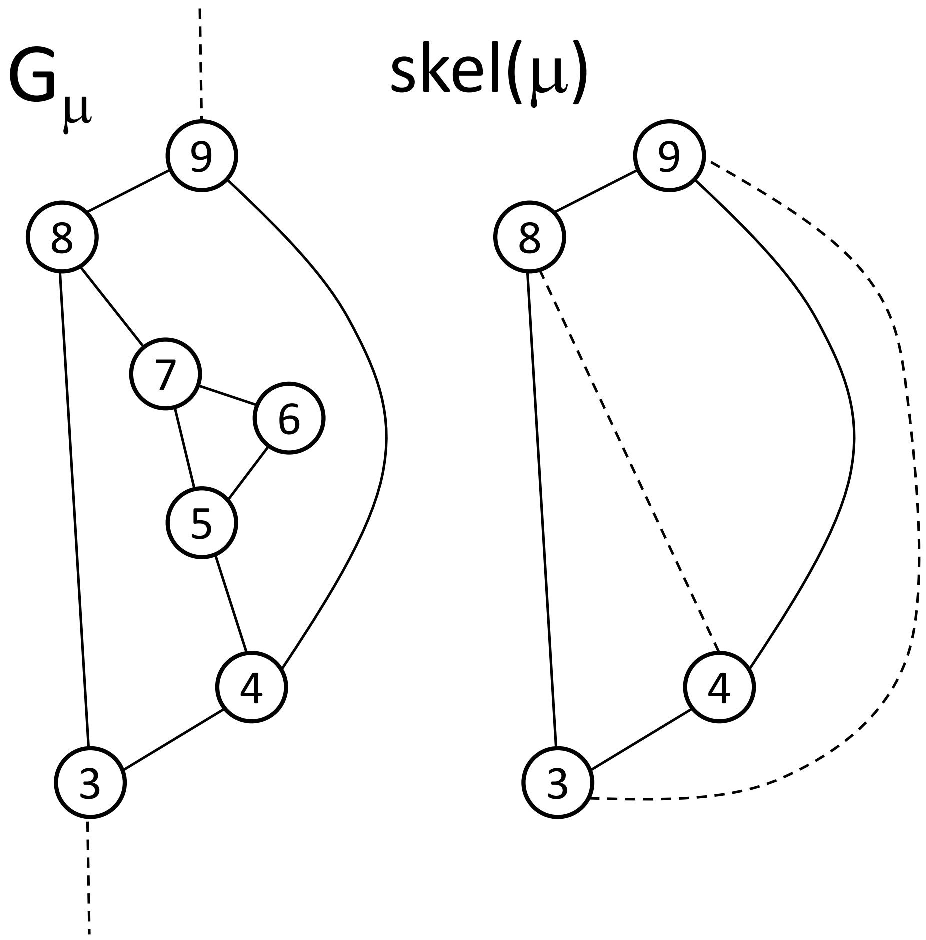
Candidate set of an inner R-node

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





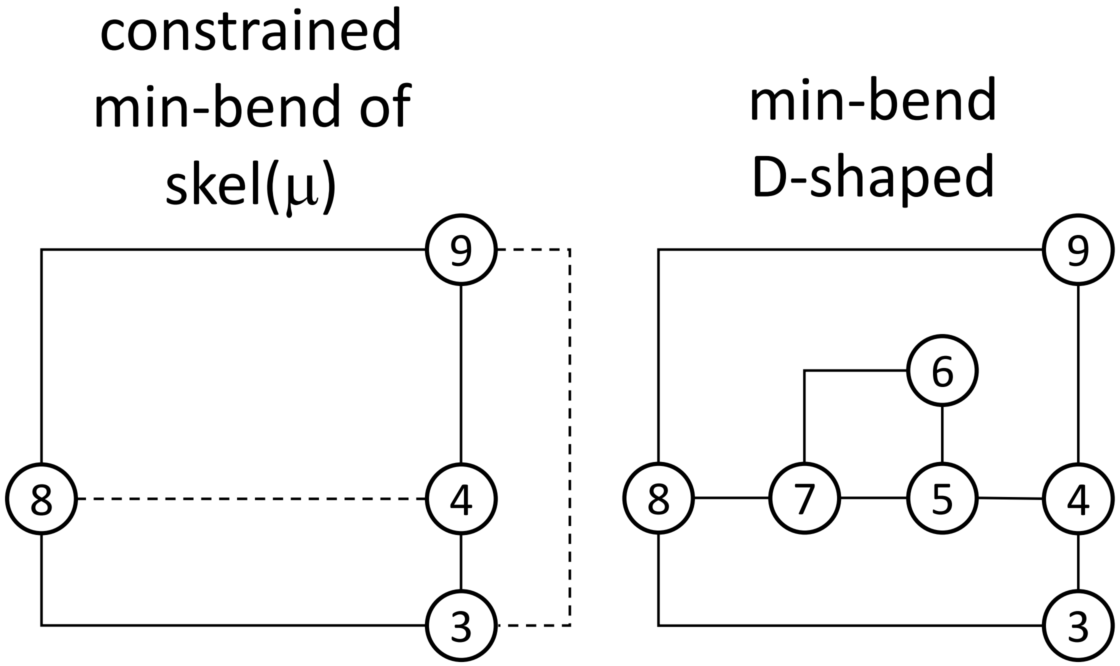
Candidate set of an inner R-node



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

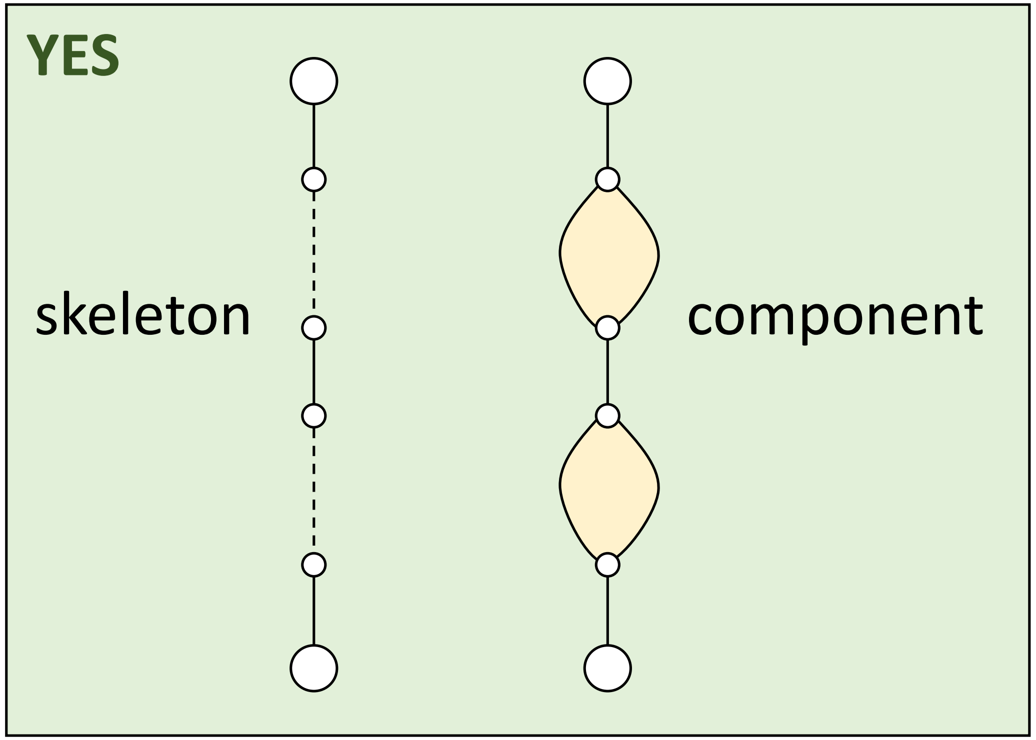
⇒

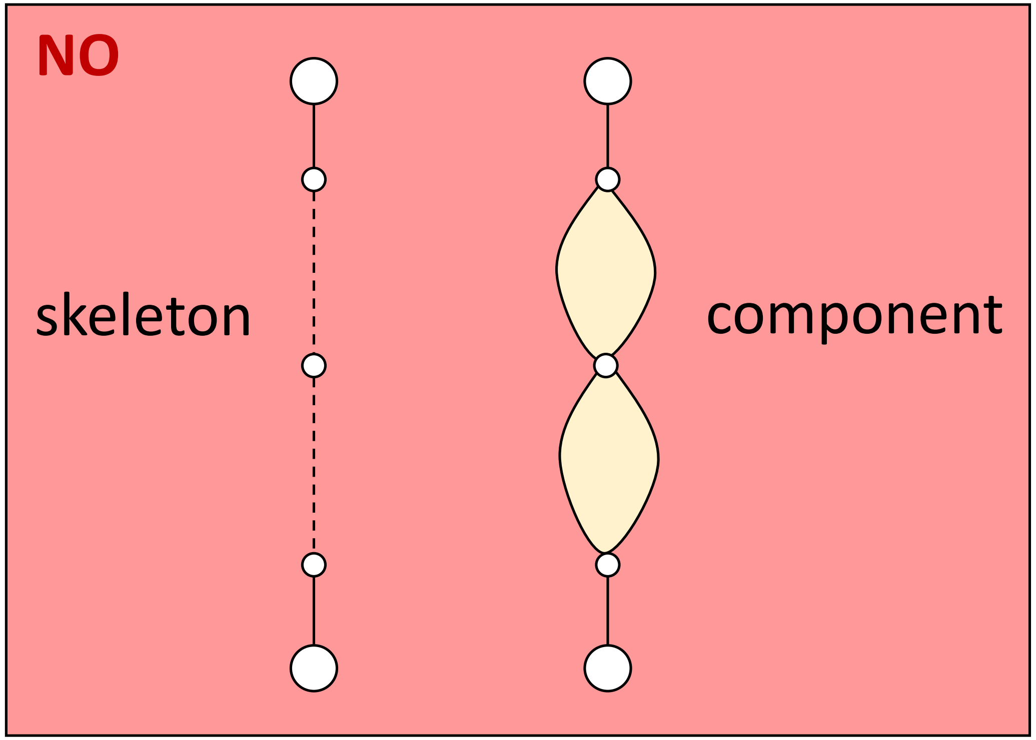
min-bend 3-connected cubic (with constraints)

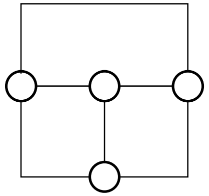


$O(n_\mu)$ time

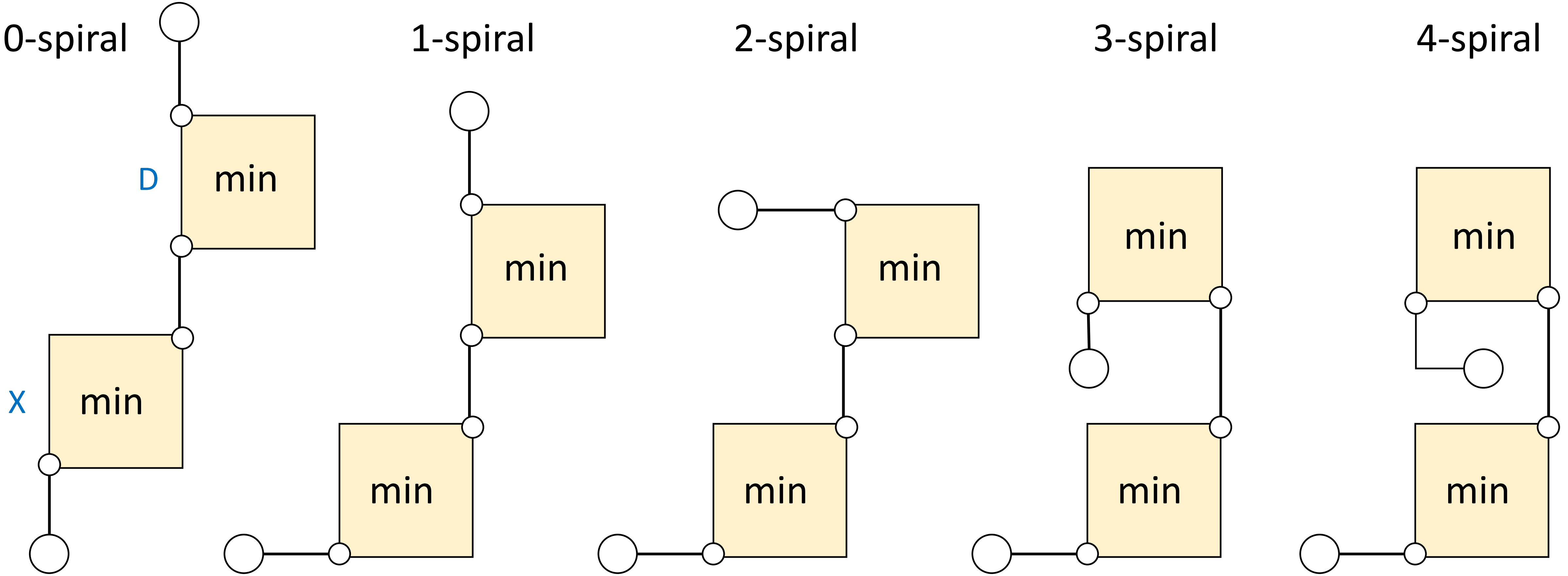
Candidate set of an inner S-node







Candidate set of an inner S-node

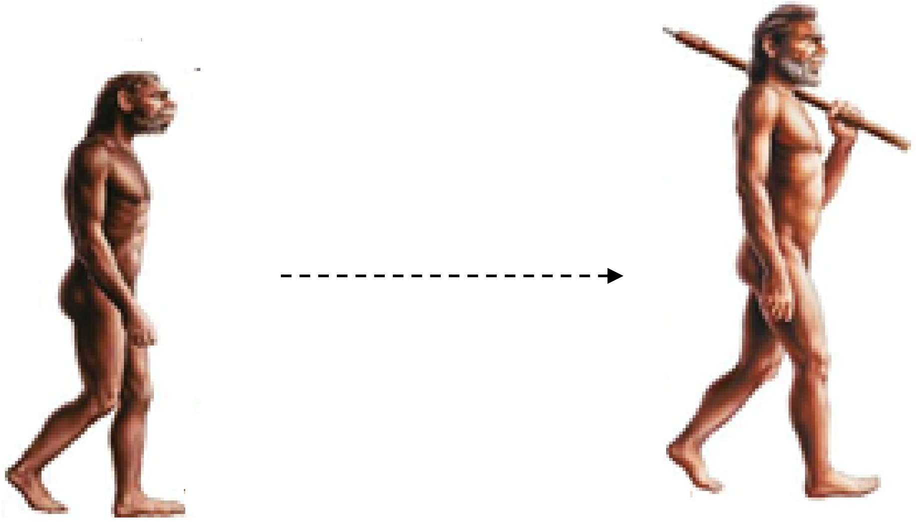


#(extra bends) = max{0, spirality – (#D-shaped + #Q-nodes – 1)}

O(n_μ) time

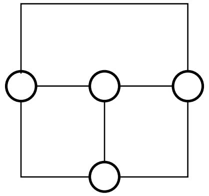
Open problems

- **Problem 1.** Is there a subquadratic-time algorithm to compute a bend-minimum orthogonal drawing of a planar 3-graph?



$O(n^2)$
our result

Can we do better?

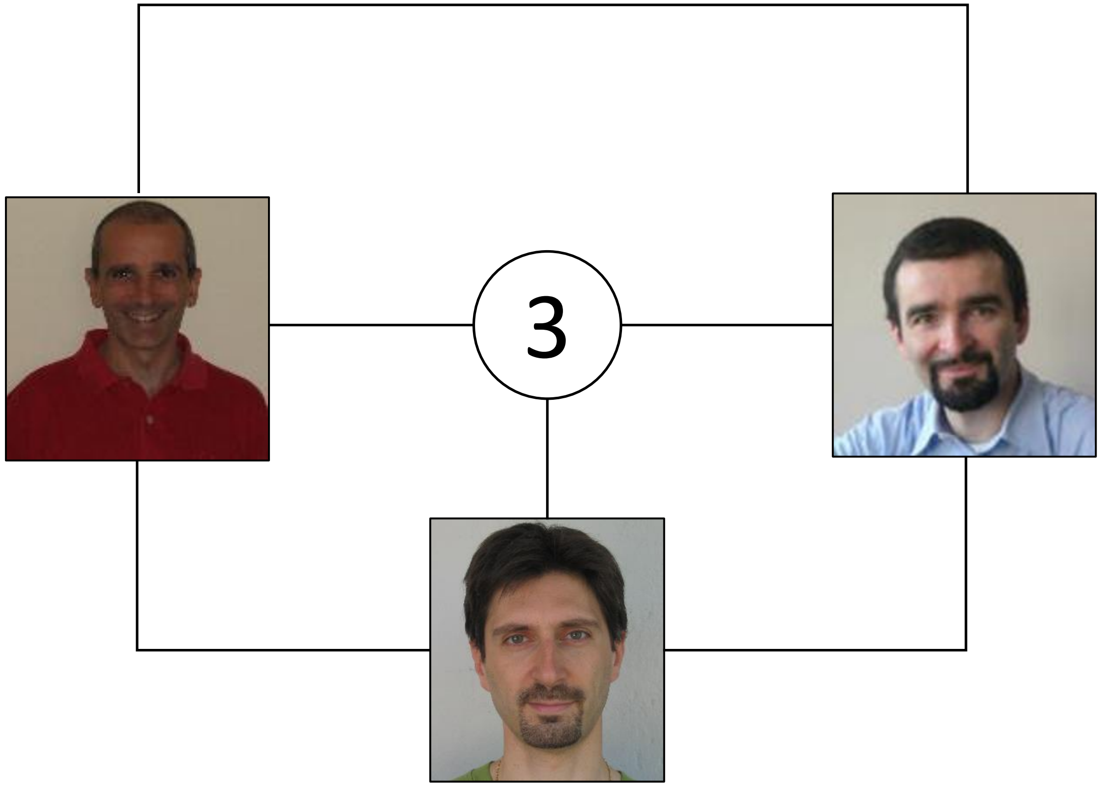


Open problems: even more prehistoric

- **Problem 2.** Is there a linear-time algorithm to compute a bend-minimum orthogonal drawing of a *plane* 4-graph (in the *fixed embedding setting*)?

Thank you!

Giuseppe Liotta



Maurizio Patrignani