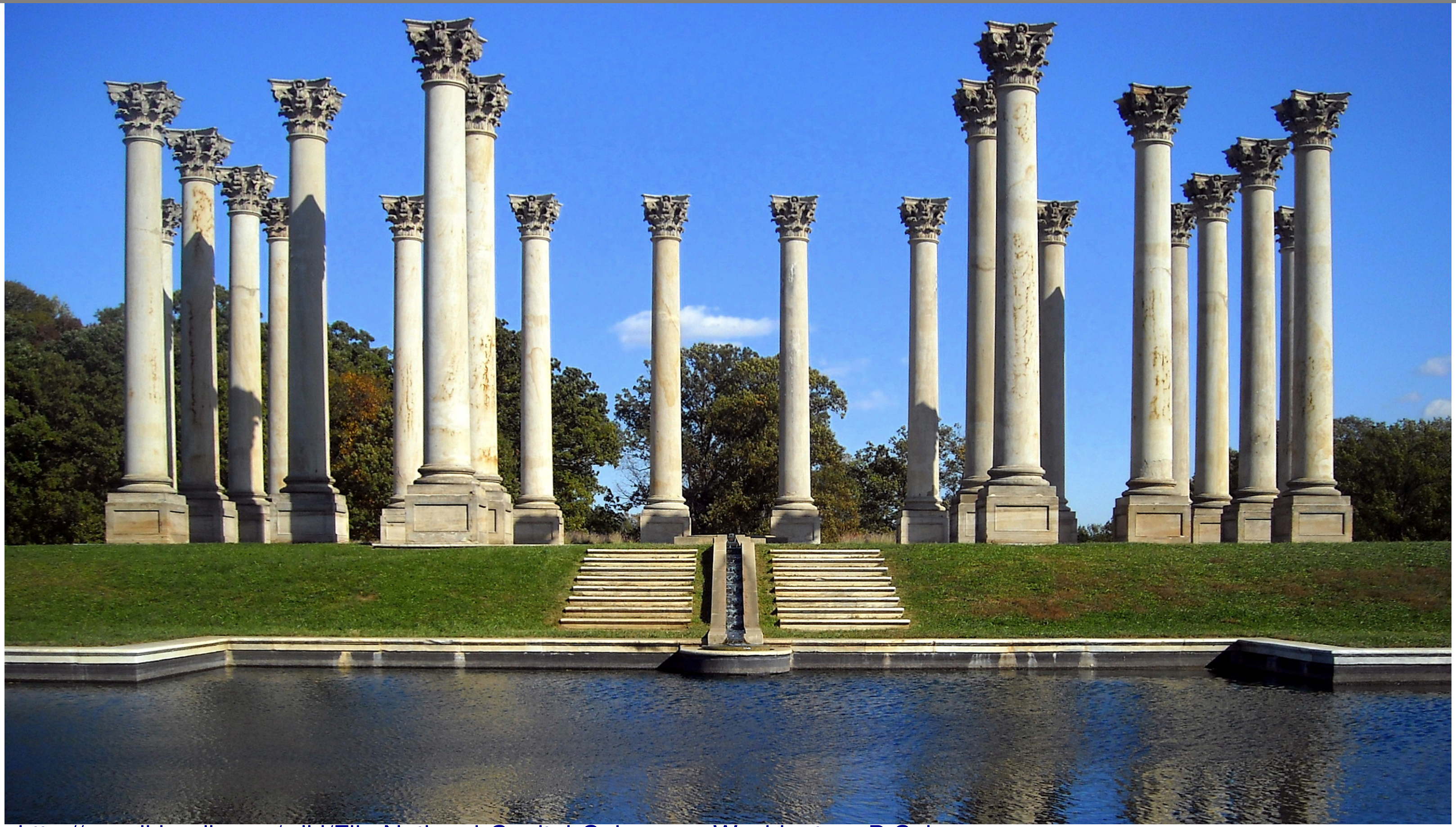
Column-based Graph Layouts

Gregor Betz, *Andreas Gemsa*, Christof Mathies, Ignaz Rutter, Dorothea Wagner

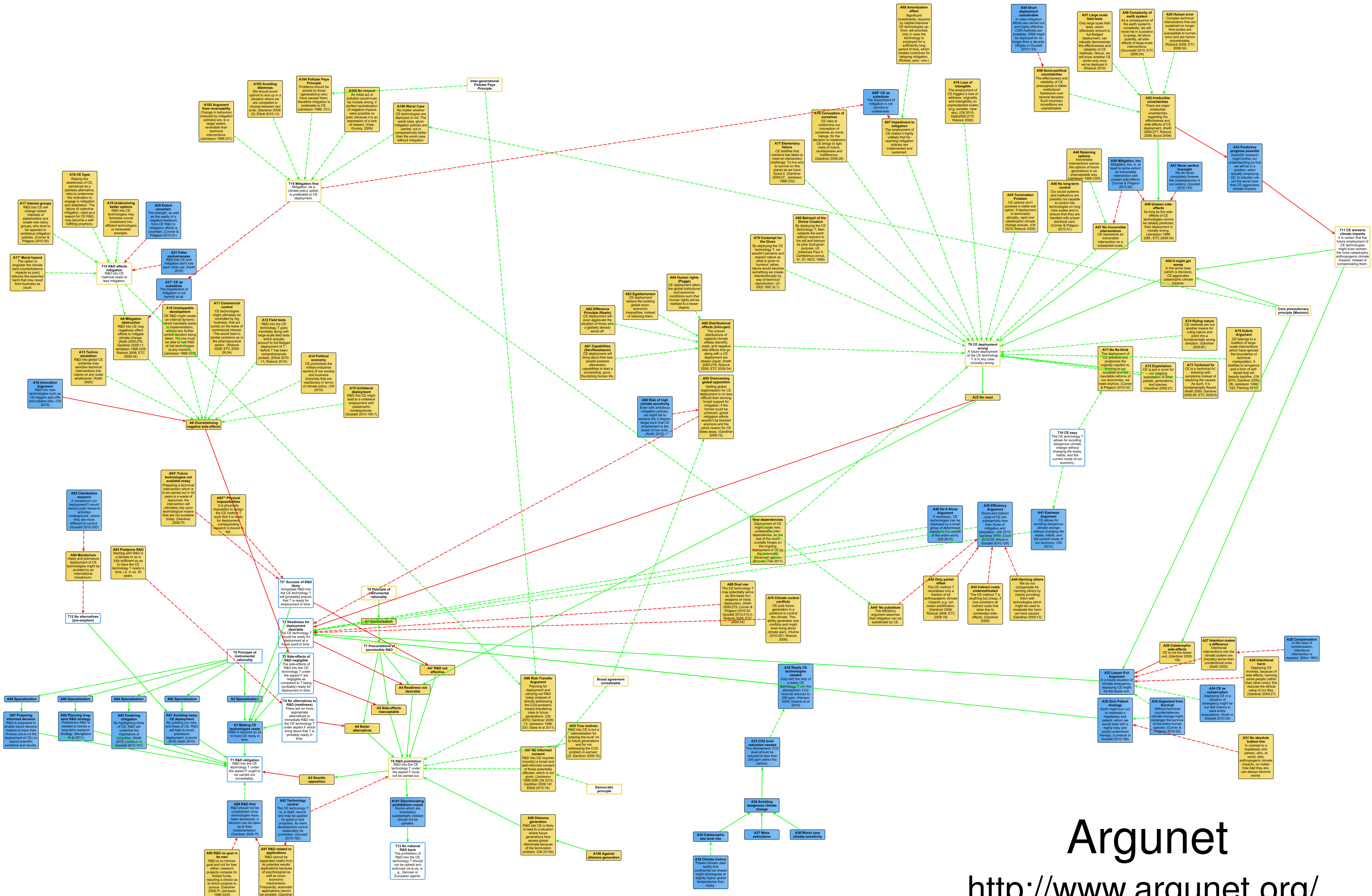
Karlsruhe Institute of Technology (KIT),



http://en.wikipedia.org/wiki/File:National_Capitol_Columns_-_Washington,_D.C..jpg

Motivation

Karlsruhe Institute of Technology



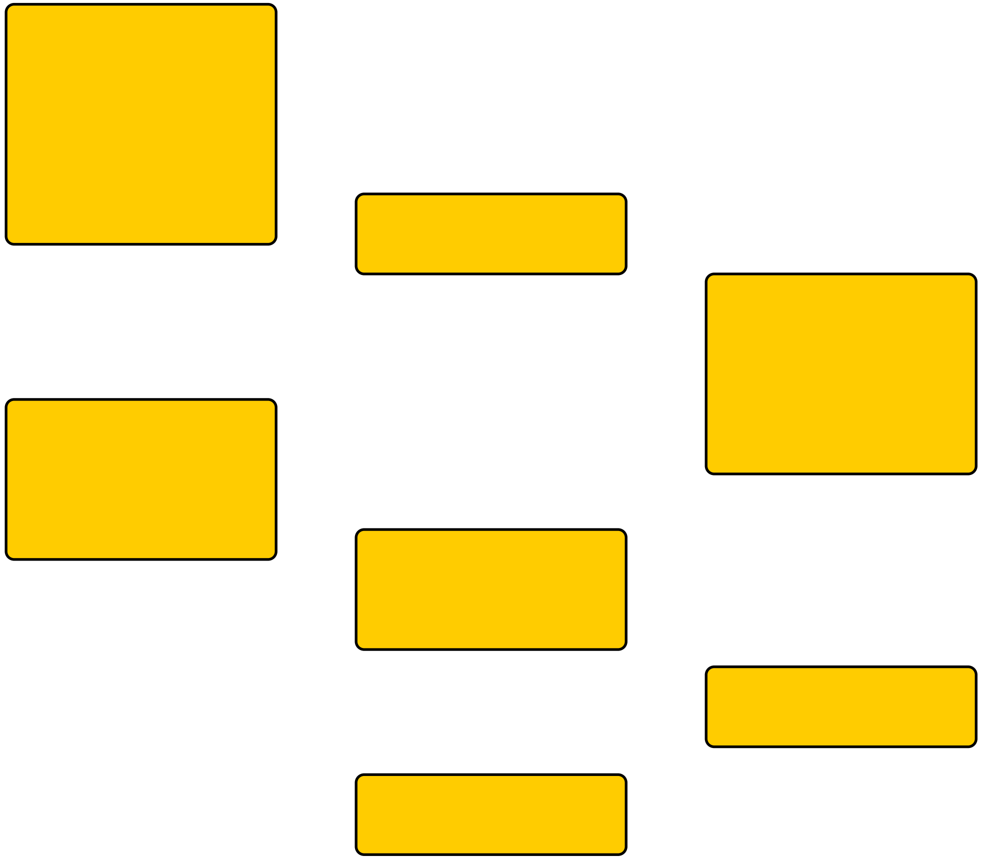
Argunet

http://www.argunet.org/

Introduction

Drawing Style

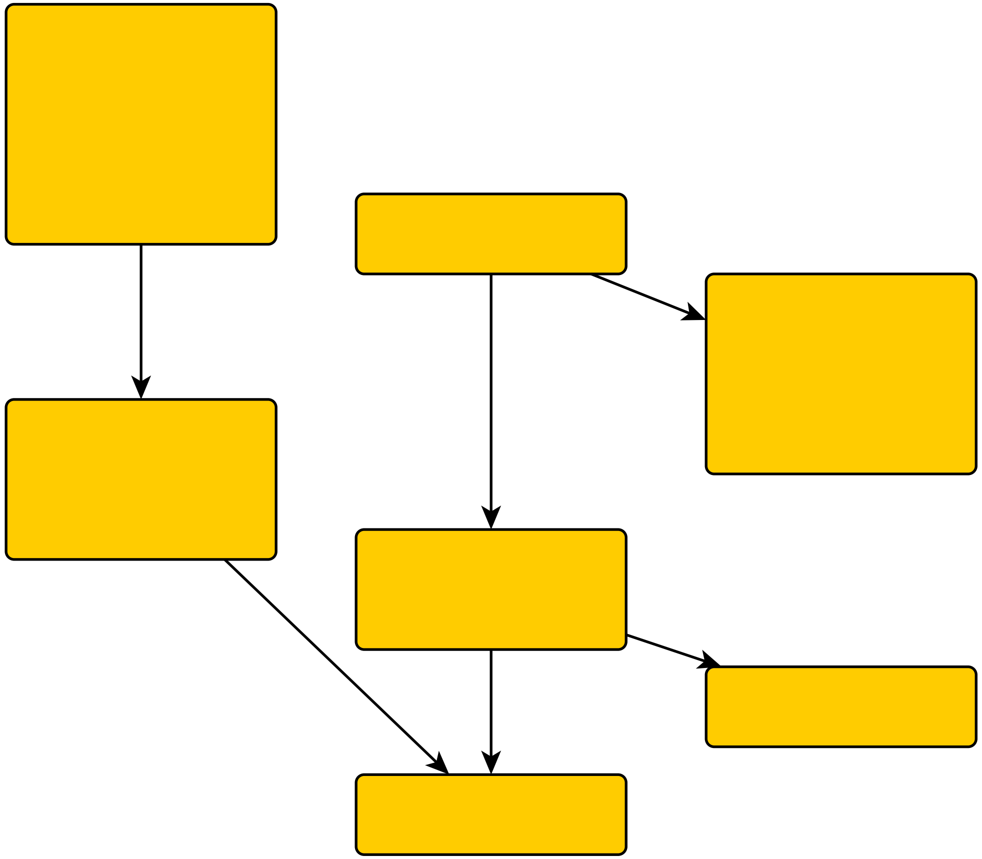
- boxes of uniform width
- upward drawings
- ingoing edges at the top & outgoing edges at the bottom
- orthogonal edges
- directed acyclic graphs



Introduction

Drawing Style

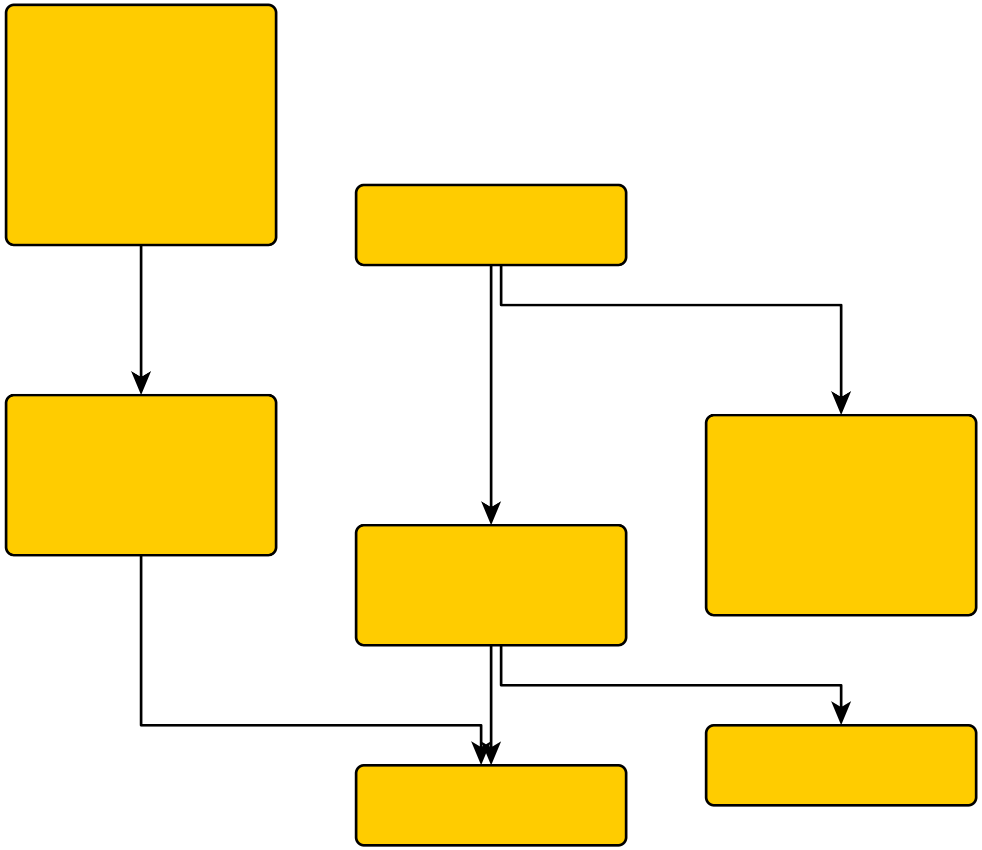
- boxes of uniform width
- upward drawings
- ingoing edges at the top & outgoing edges at the bottom
- orthogonal edges
- directed acyclic graphs



Introduction

Drawing Style

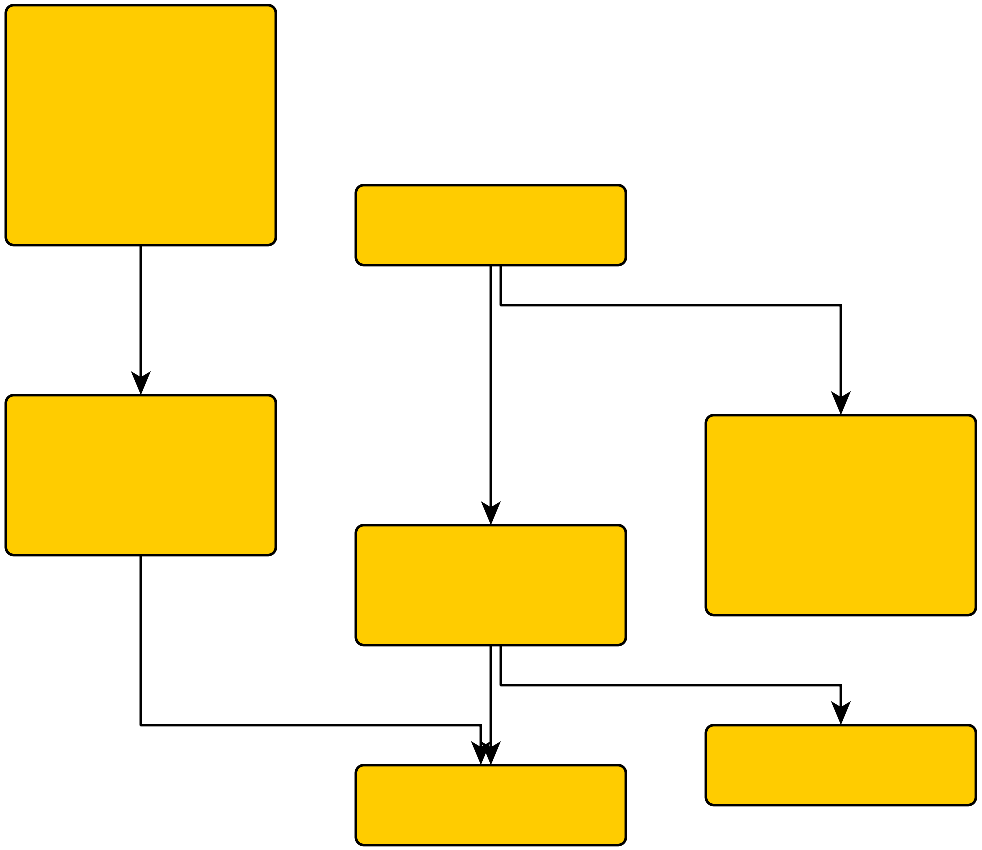
- boxes of uniform width
- upward drawings
- ingoing edges at the top & outgoing edges at the bottom
- orthogonal edges
- directed acyclic graphs



Introduction

Drawing Style

- boxes of uniform width
- upward drawings
- ingoing edges at the top & outgoing edges at the bottom
- orthogonal edges
- directed acyclic graphs



Sugiyama Framework ?

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Introduction

Sugiyama Framework

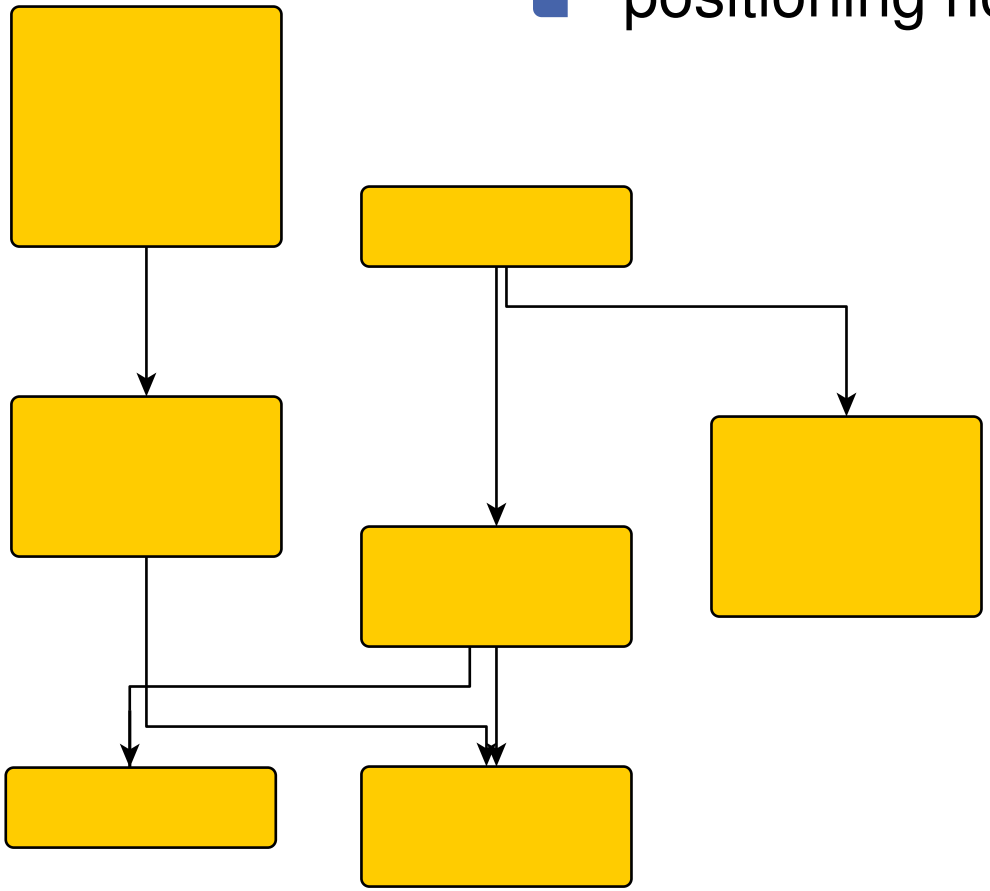
three steps:

- layer assignment
- determining positions of nodes in each layer
- positioning nodes and edges

Sugiyama Framework

three steps:

- layer assignment
- determining positions of nodes in each layer
- positioning nodes and edges

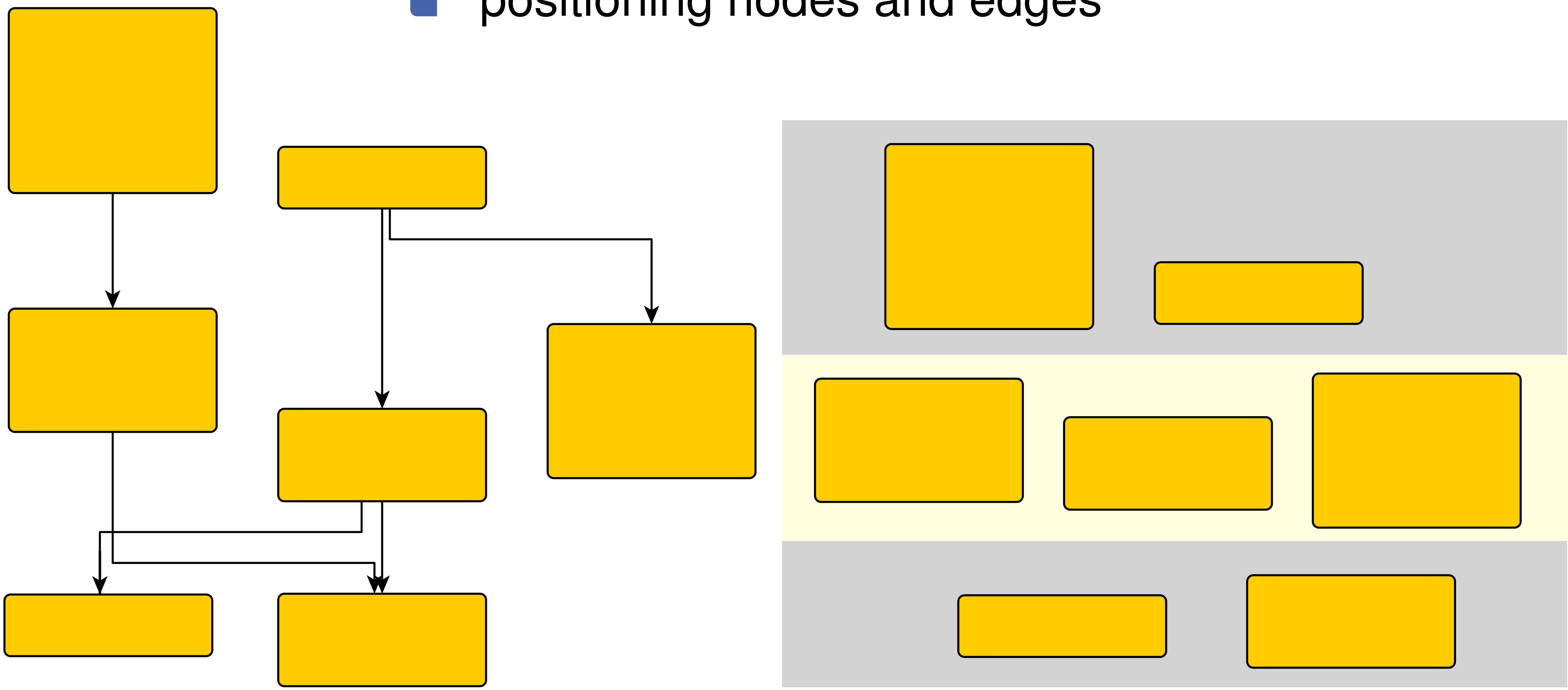


Introduction

Sugiyama Framework

three steps:

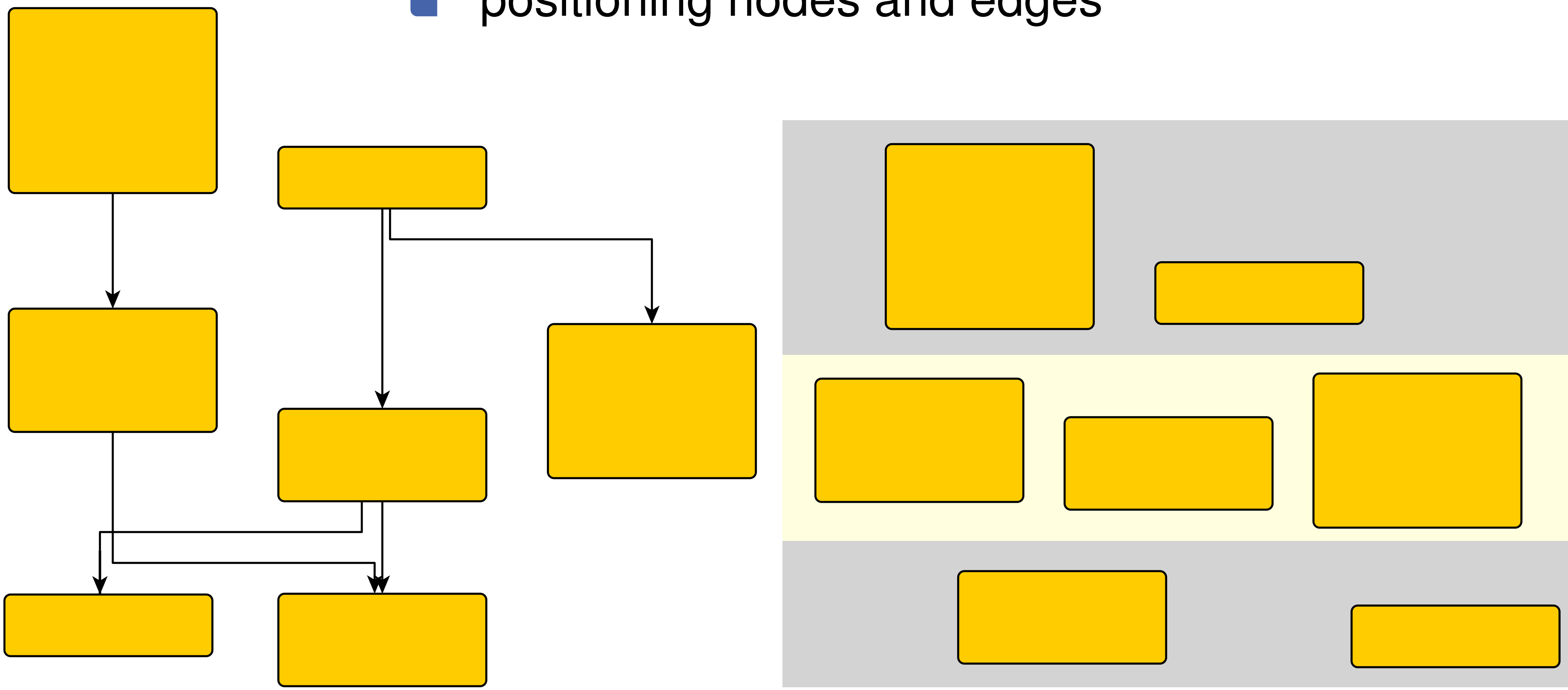
- layer assignment
- determining positions of nodes in each layer
- positioning nodes and edges



Sugiyama Framework

three steps:

- layer assignment
- determining positions of nodes in each layer
- positioning nodes and edges

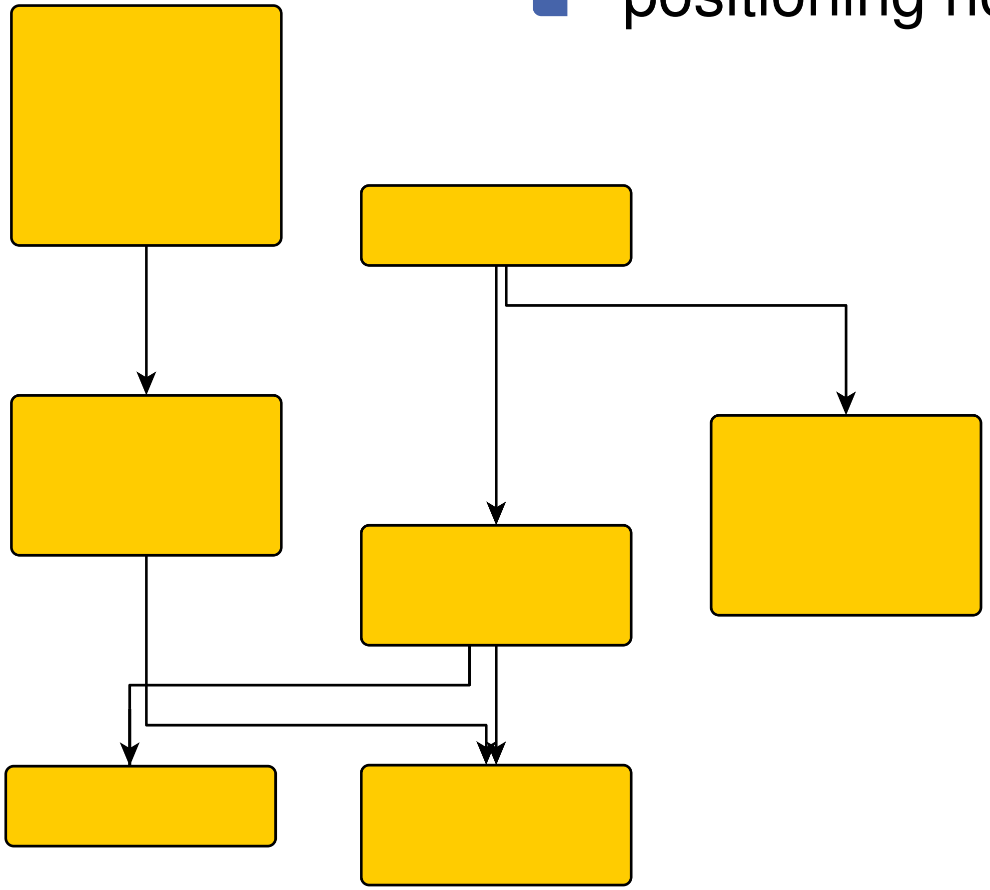


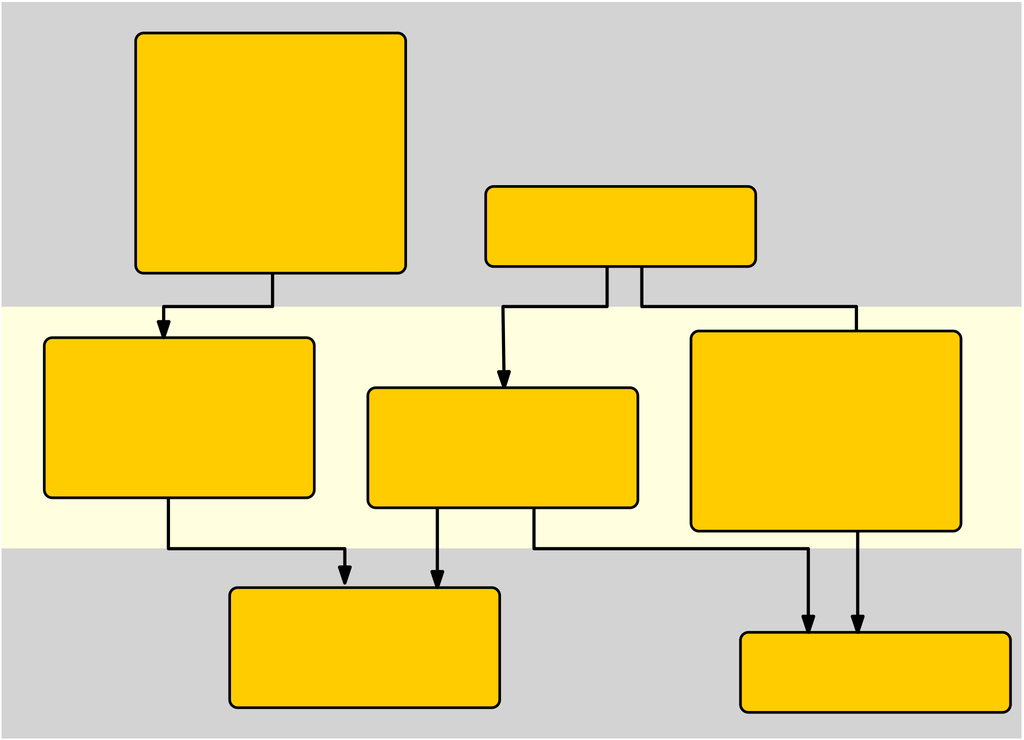
Introduction

Sugiyama Framework

three steps:

- layer assignment
- determining positions of nodes in each layer
- positioning nodes and edges





Introduction

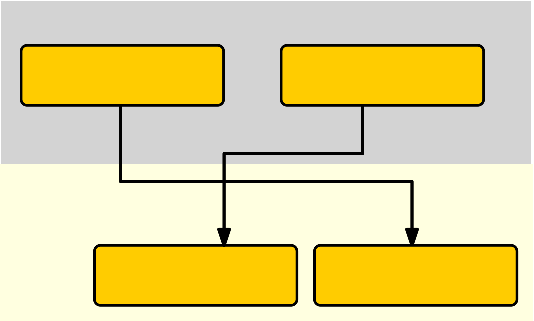
Sugiyama Framework

three steps:

- layer assignment
- determining positions of nodes in each layer
- positioning nodes and edges

two drawbacks:

- unfortunate layering
  $\Rightarrow$ many unncessary crossings



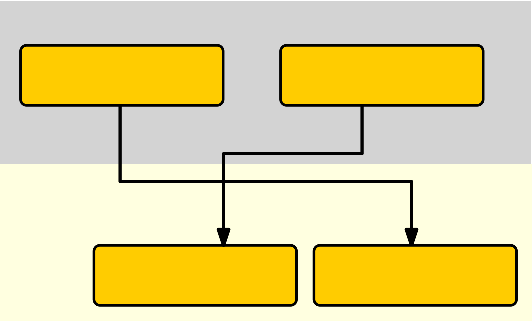
- tall nodes can lead to non-compact layouts

Introduction

Sugiyama Framework

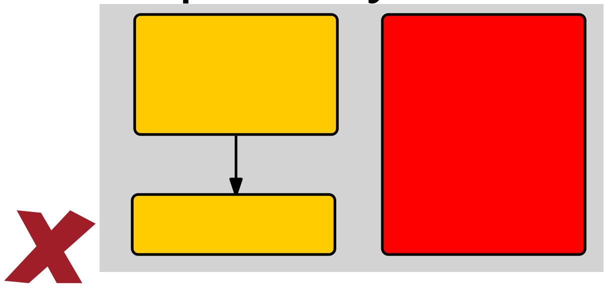
three steps:

- layer assignment
- determining positions of nodes in each layer
- positioning nodes and edges



two drawbacks:

- unfortunate layering
  ⇒ many unncessary crossings
- tall nodes can lead to non-compact layouts



Introduction

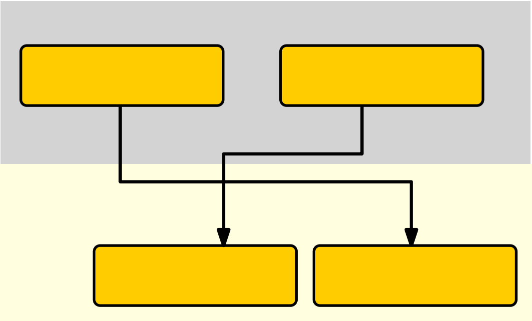
Sugiyama Framework

three steps:

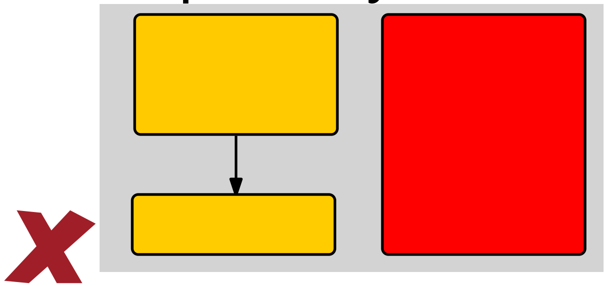
- layer assignment
- determining positions of nodes in each layer
- positioning nodes and edges

two drawbacks:

- unfortunate layering
  $\Rightarrow$ many unncessary crossings



- tall nodes can lead to non-compact layouts



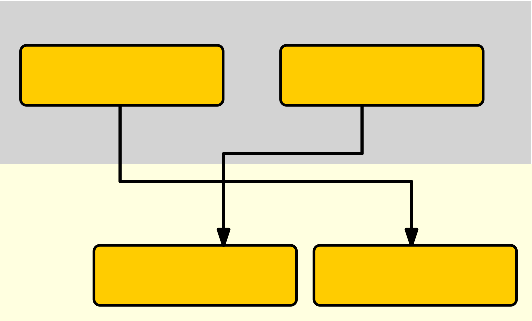
layer-free upward crossing minimization
[Chimani et al. ’10]

Introduction

Sugiyama Framework

three steps:

- layer assignment
- determining positions of nodes in each layer
- positioning nodes and edges

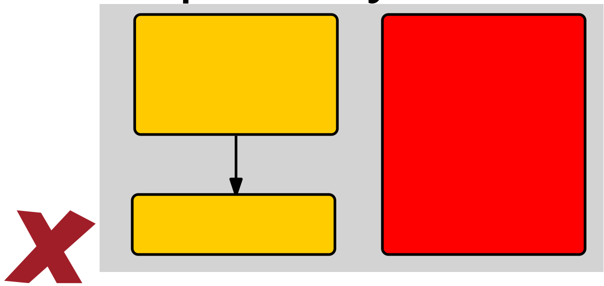


two drawbacks:

- unfortunate layering
  $\Rightarrow$ many unncessary crossings
- tall nodes can lead to non-compact layouts

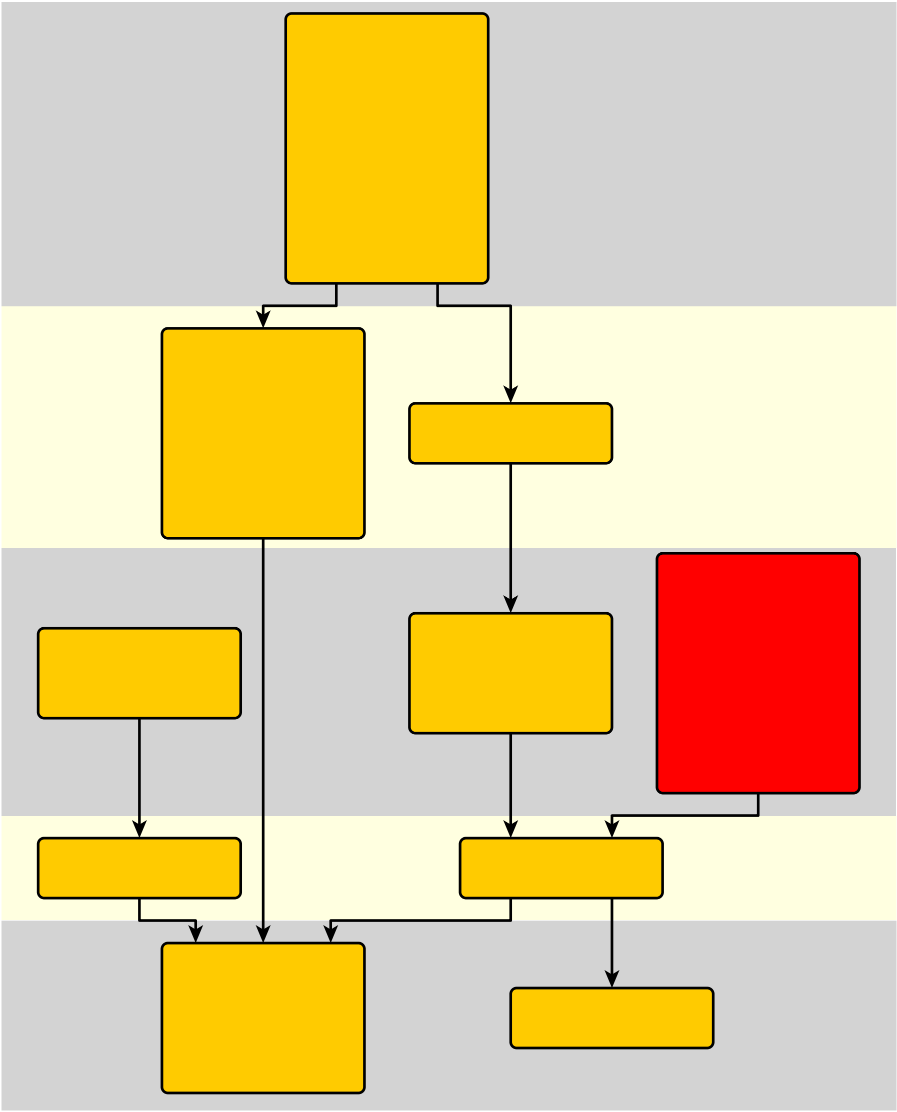
layer-free upward crossing minimization
[Chimani et al. ’10]

?



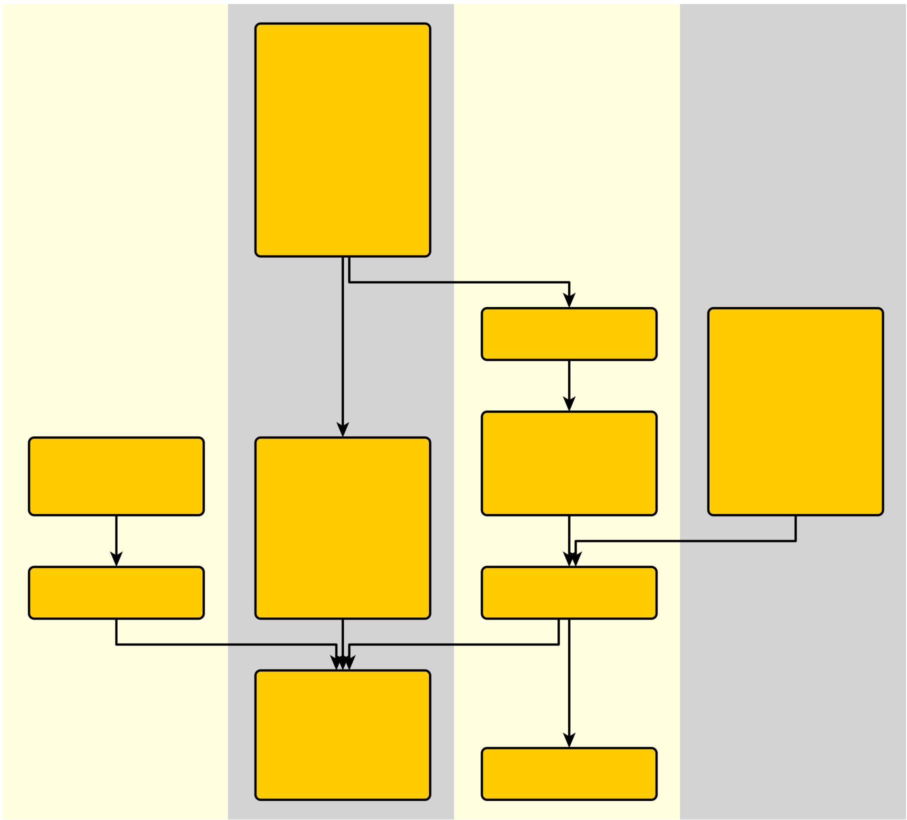
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Introduction

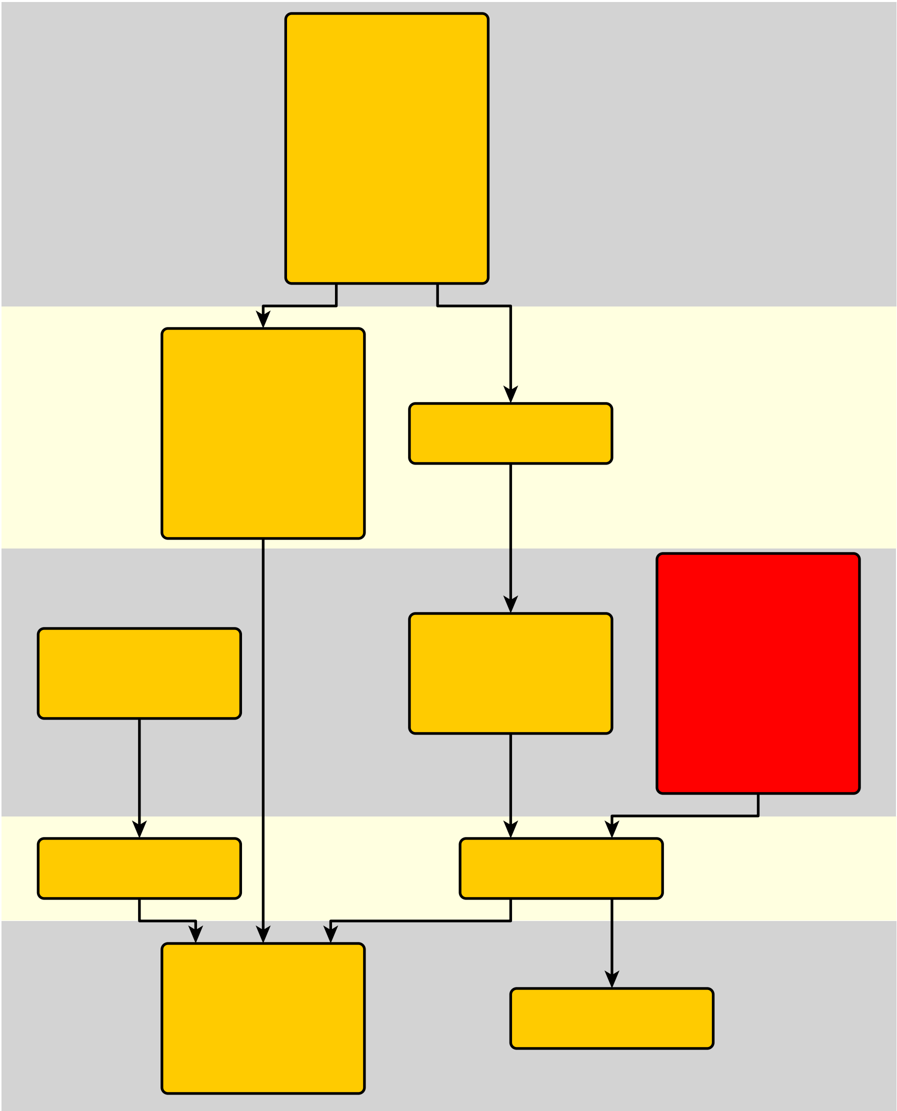


Sugiyama-Framework:
non-compact layouts

Introduction



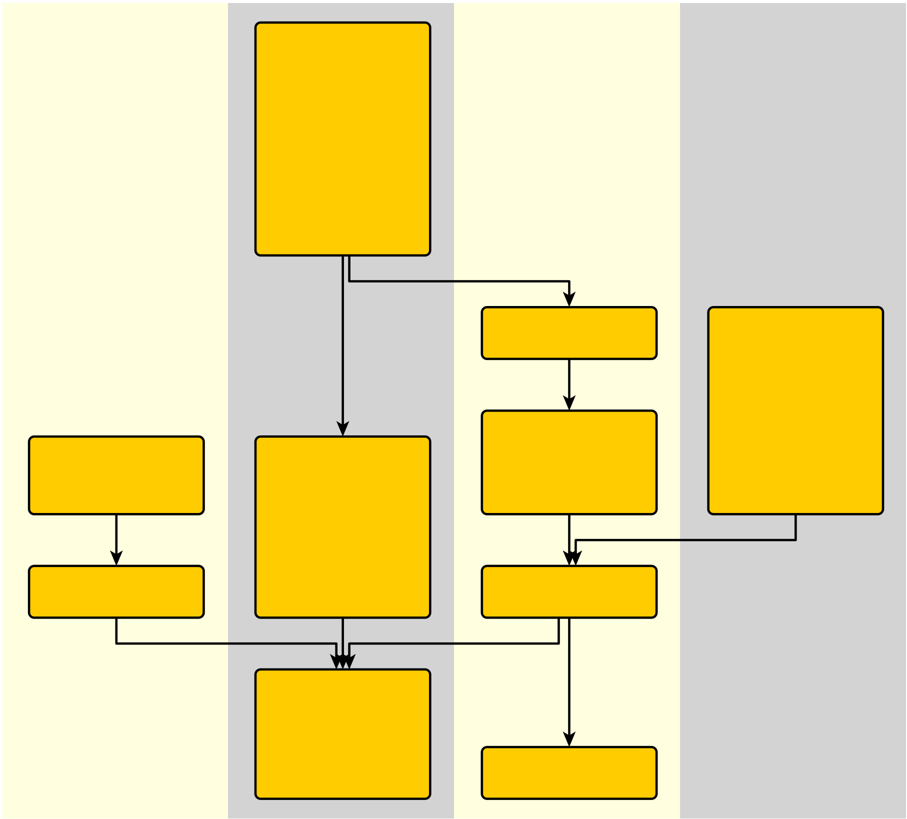
our approach



Sugiyama-Framework:
non-compact layouts

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

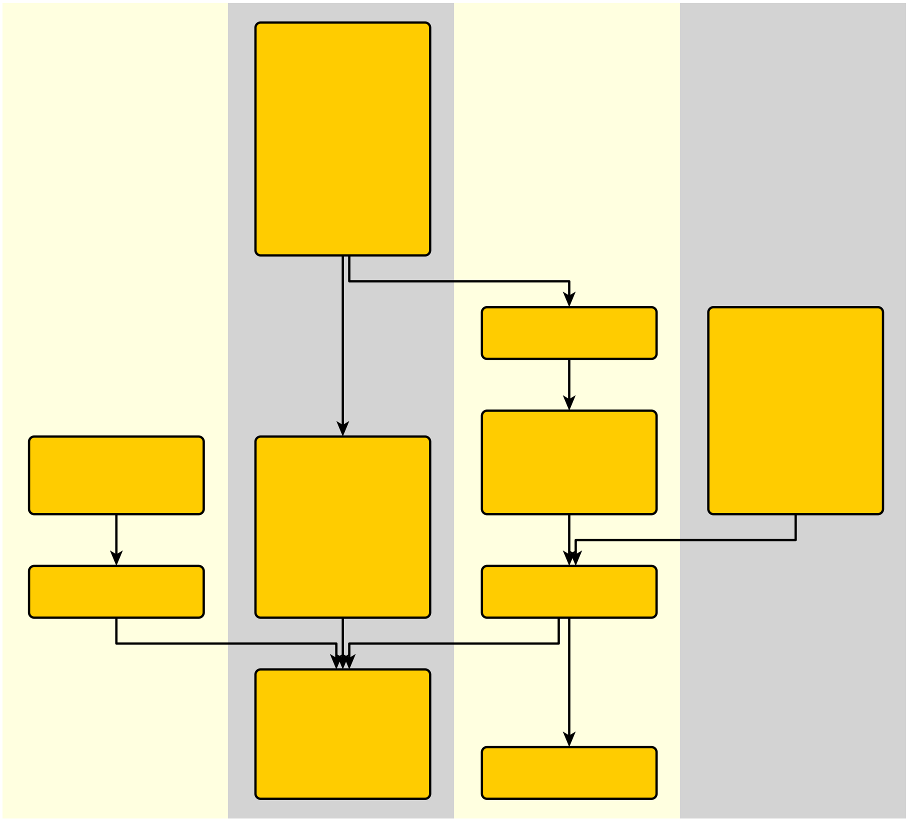
Column-based Graph Layouts



our approach

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Column-based Graph Layouts



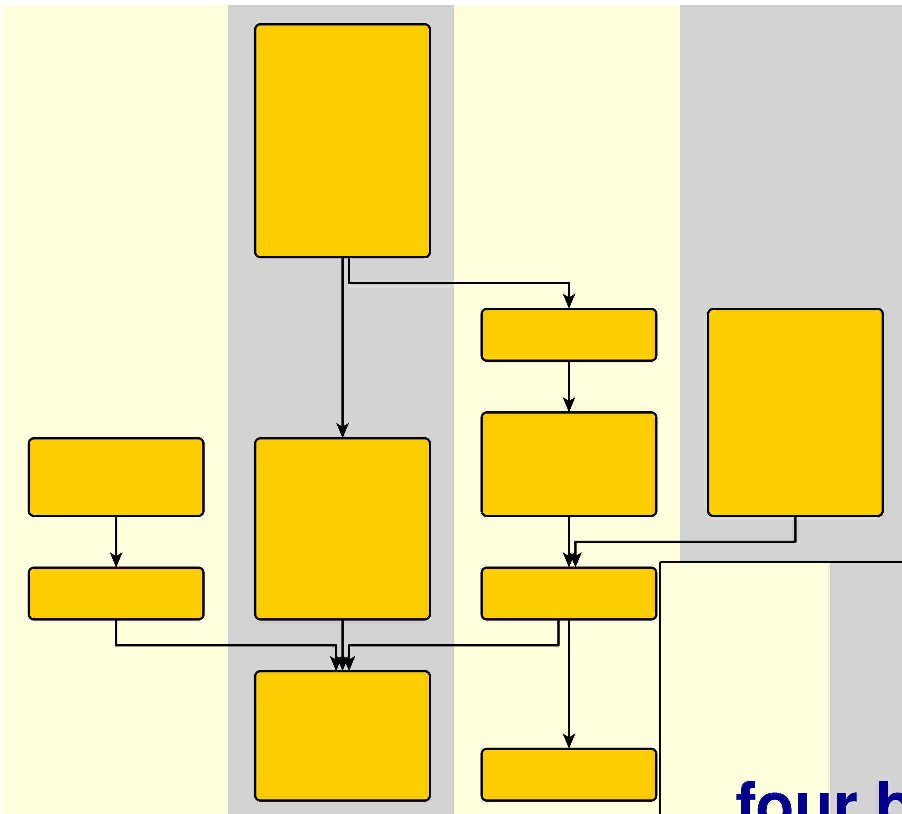
our approach

Quality Guarantees

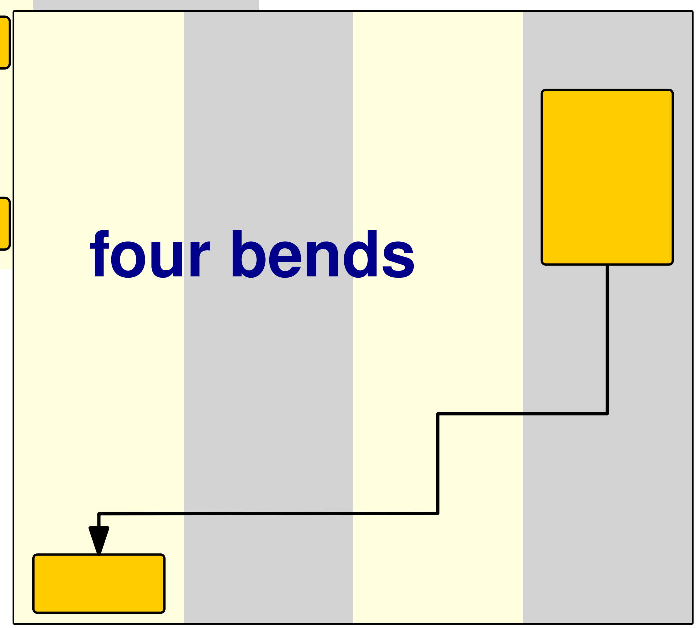
- Vertical alignment of predecessors
- At most four bends per edge
- Local symmetry
- Spacing constraints
- Edge Bundling

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Column-based Graph Layouts



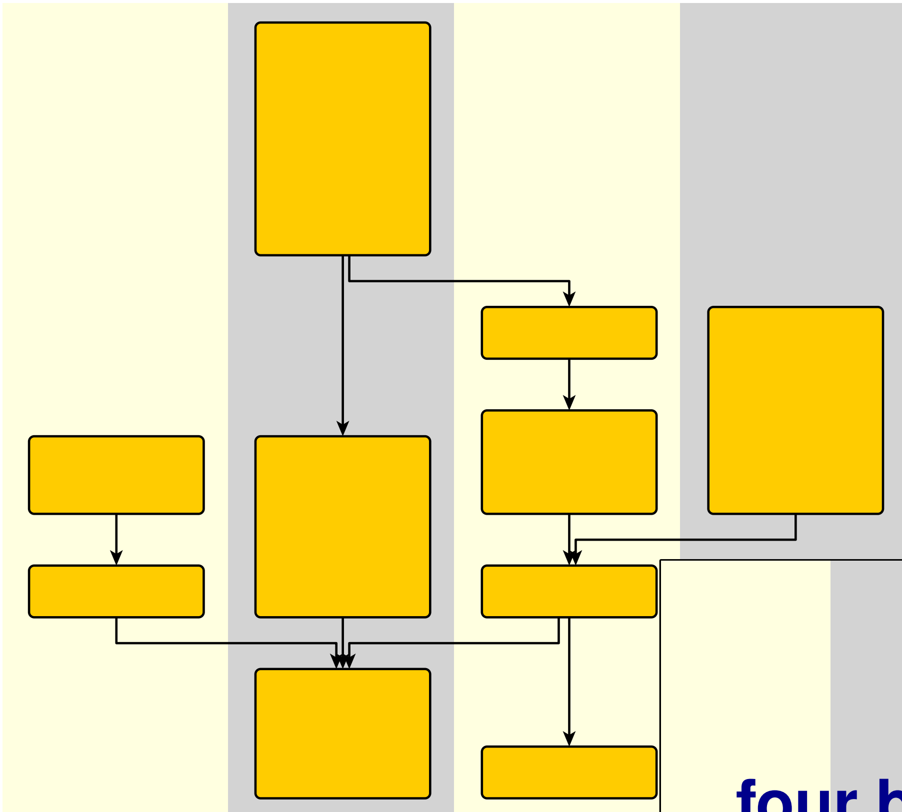
our approach



Quality Guarantees

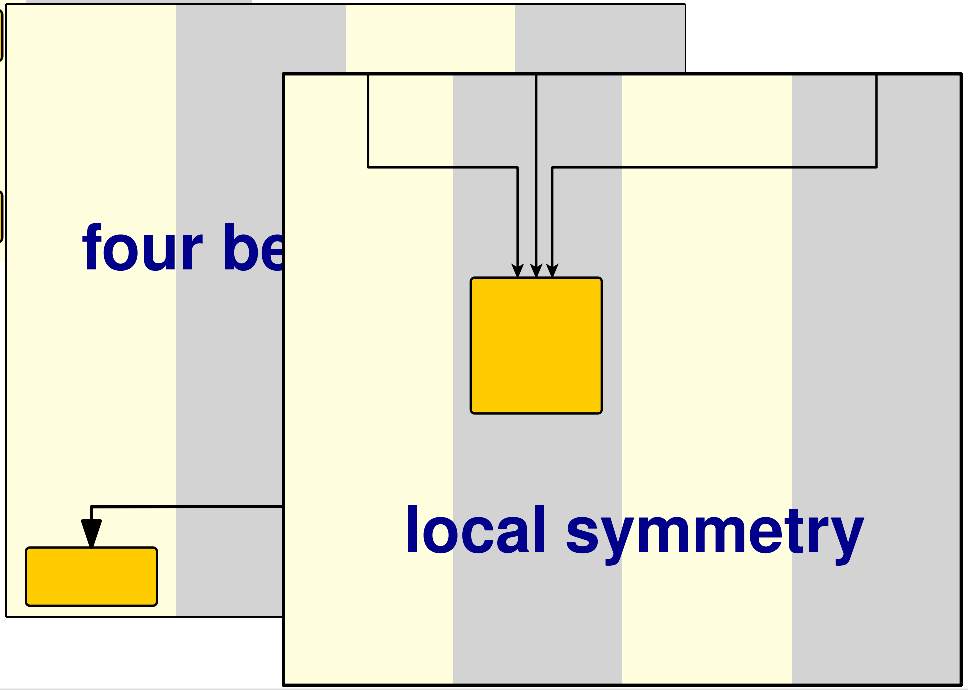
- Vertical alignment of predecessors
- At most four bends per edge
- Local symmetry
- Spacing constraints
- Edge Bundling

Column-based Graph Layouts



Quality Guarantees

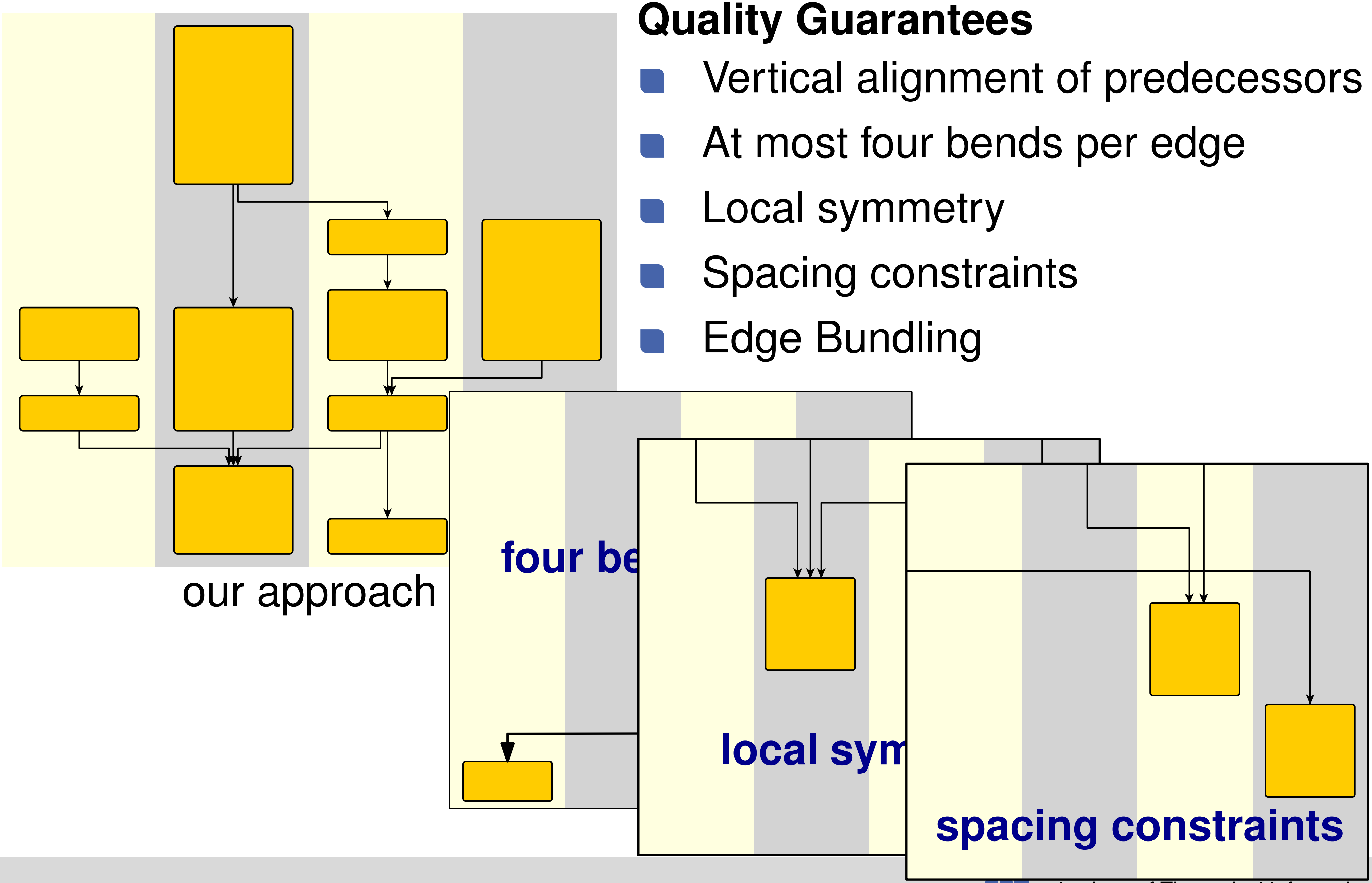
- Vertical alignment of predecessors
- At most four bends per edge
- Local symmetry
- Spacing constraints
- Edge Bundling



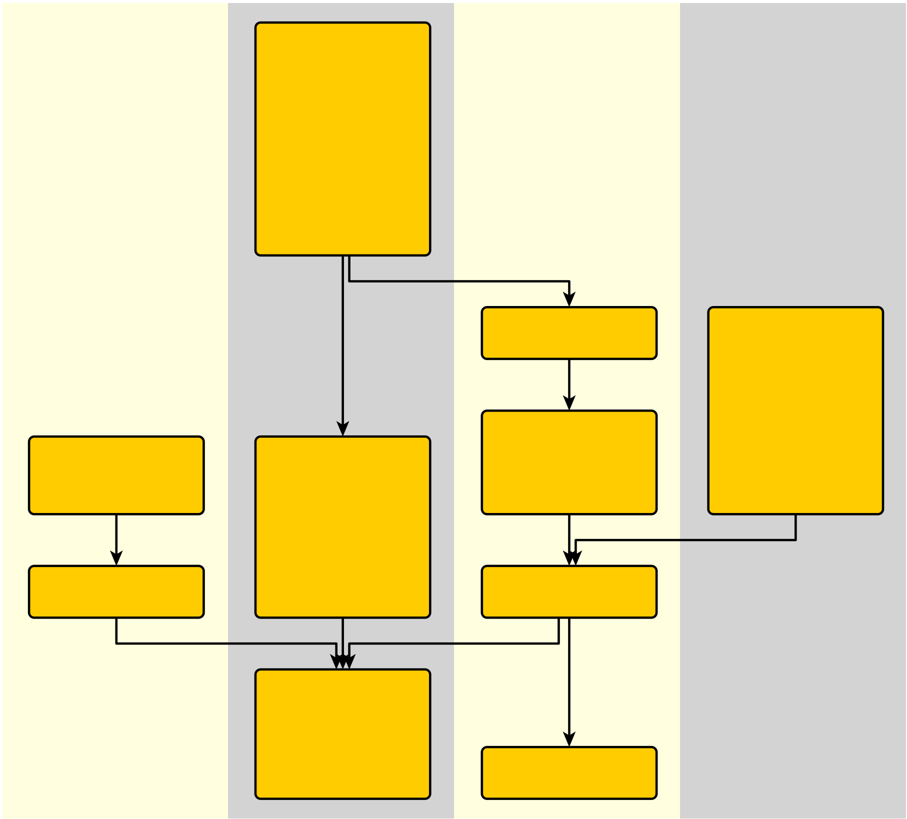
Column-based Graph Layouts

Quality Guarantees

- Vertical alignment of predecessors
- At most four bends per edge
- Local symmetry
- Spacing constraints
- Edge Bundling



Column-based Graph Layouts



our approach

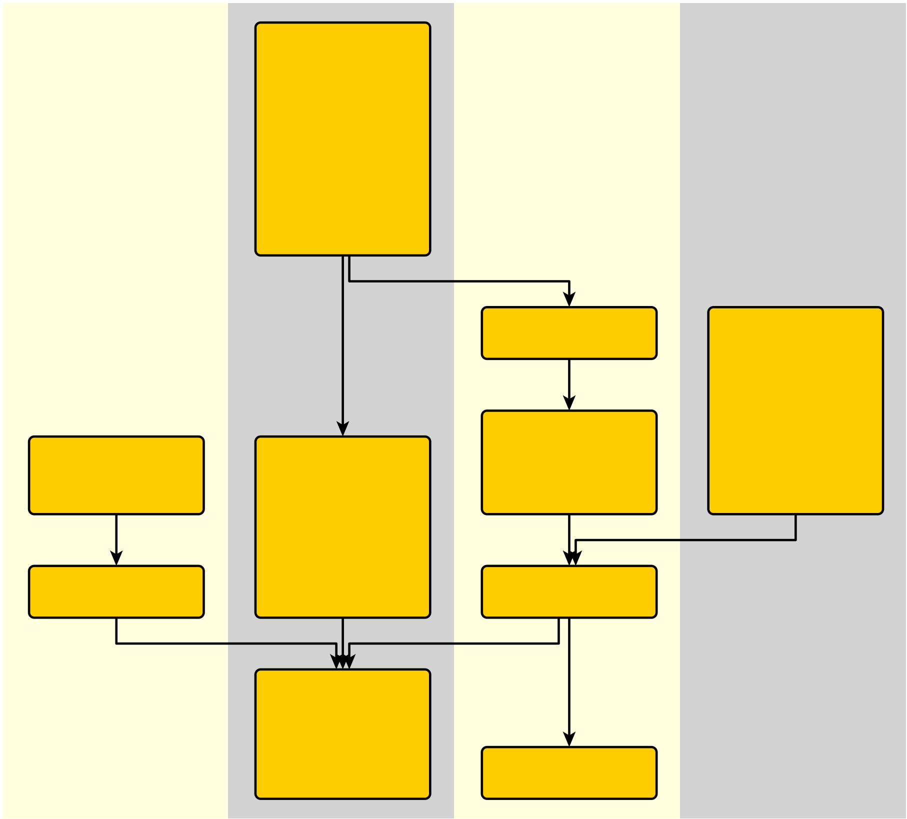
Quality Guarantees

- Vertical alignment of predecessors
- At most four bends per edge
- Local symmetry
- Spacing constraints
- Edge Bundling

Optimization Criteria:

- minimize number of crossings
- minimize number of bends
- minimize total edge length

Column-based Graph Layouts



our approach

Quality Guarantees

- Vertical alignment of predecessors
- At most four bends per edge
- Local symmetry
- Spacing constraints
- Edge Bundling

Optimization Criteria:

- minimize number of crossings
- minimize number of bends
- minimize total edge length

How do we achieve this?

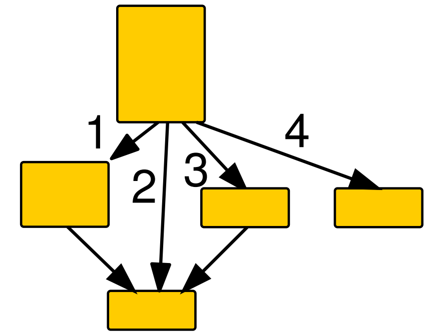
integrate *layer-free upward crossing minimization* in the *topology-shape-metric framework*

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Our Approach - TSM + Layer-free ...

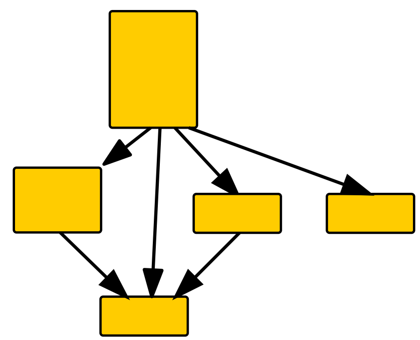
topology compute embedding of the graph

$G = (V, E)$ $\Longrightarrow$

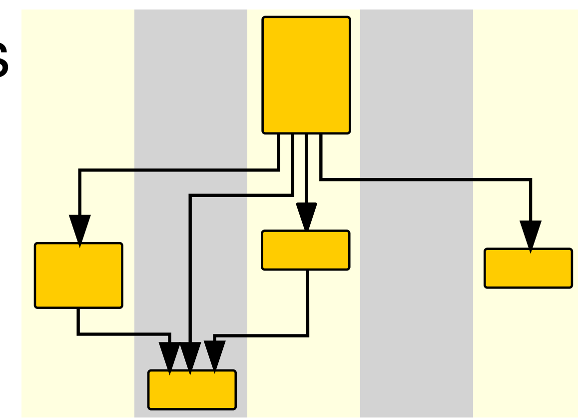


shape assign bends to edges

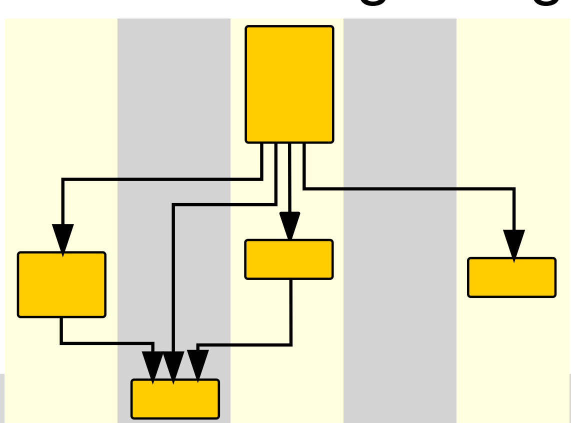
assign nodes/edges/bends to columns



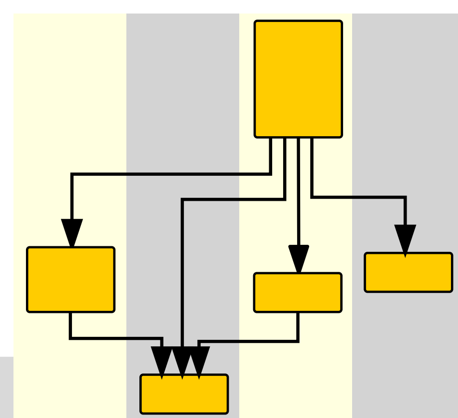
$\Longrightarrow$



metric determine edge lengths & node positions



$\Longrightarrow$



Topology – Compute Embedding

Layer-free upward crossing minimization [Chimani et al. ’10]

two steps:

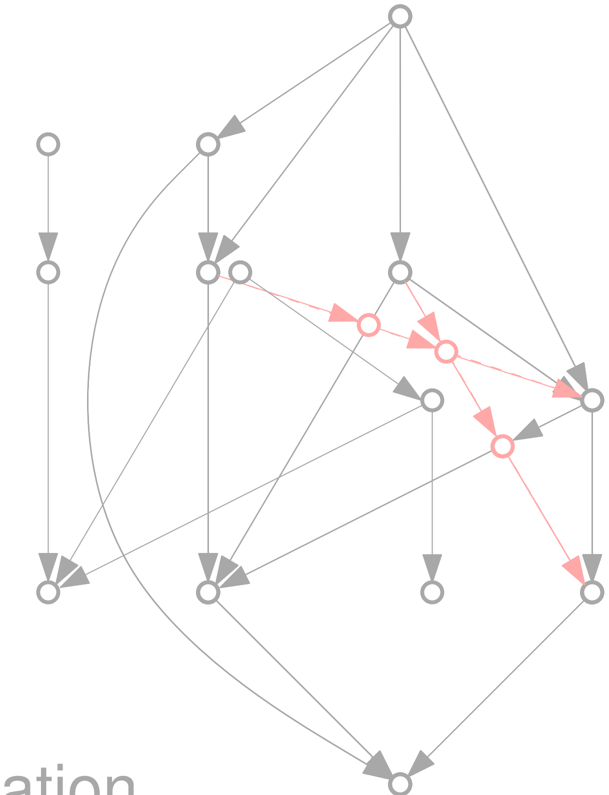
(i) delete edges until the remaining graph is upward planar

(ii) reinsert the deleted edges one by one

- add super source/sink

I’ll skip the details

- add deleted edges
- insert dummy nodes



result:

upward planar represantation

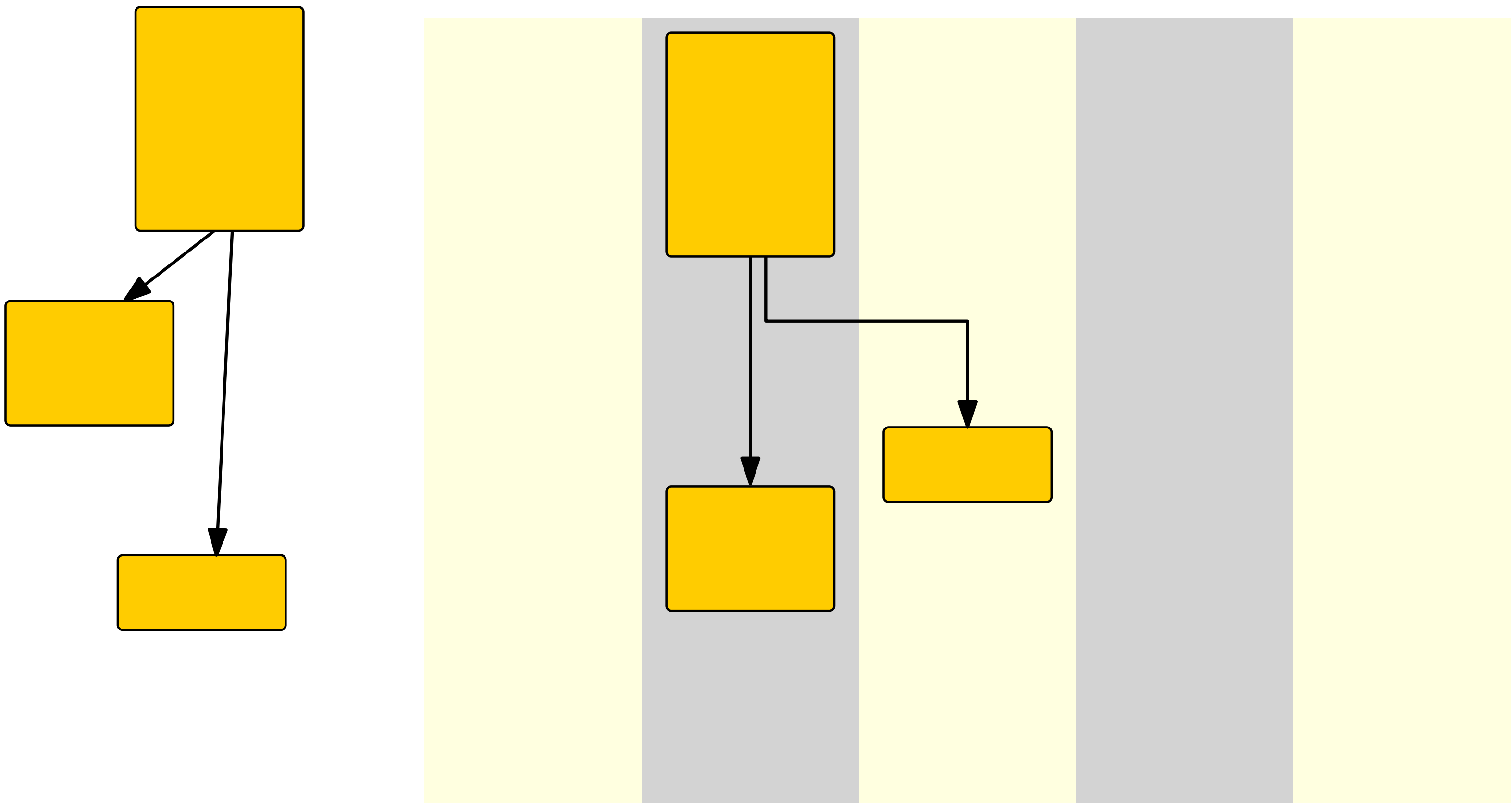
add super source/sink

Shape – Assign Each Node & Edge a Column

goal: find realizeable column assignment for each node and edge

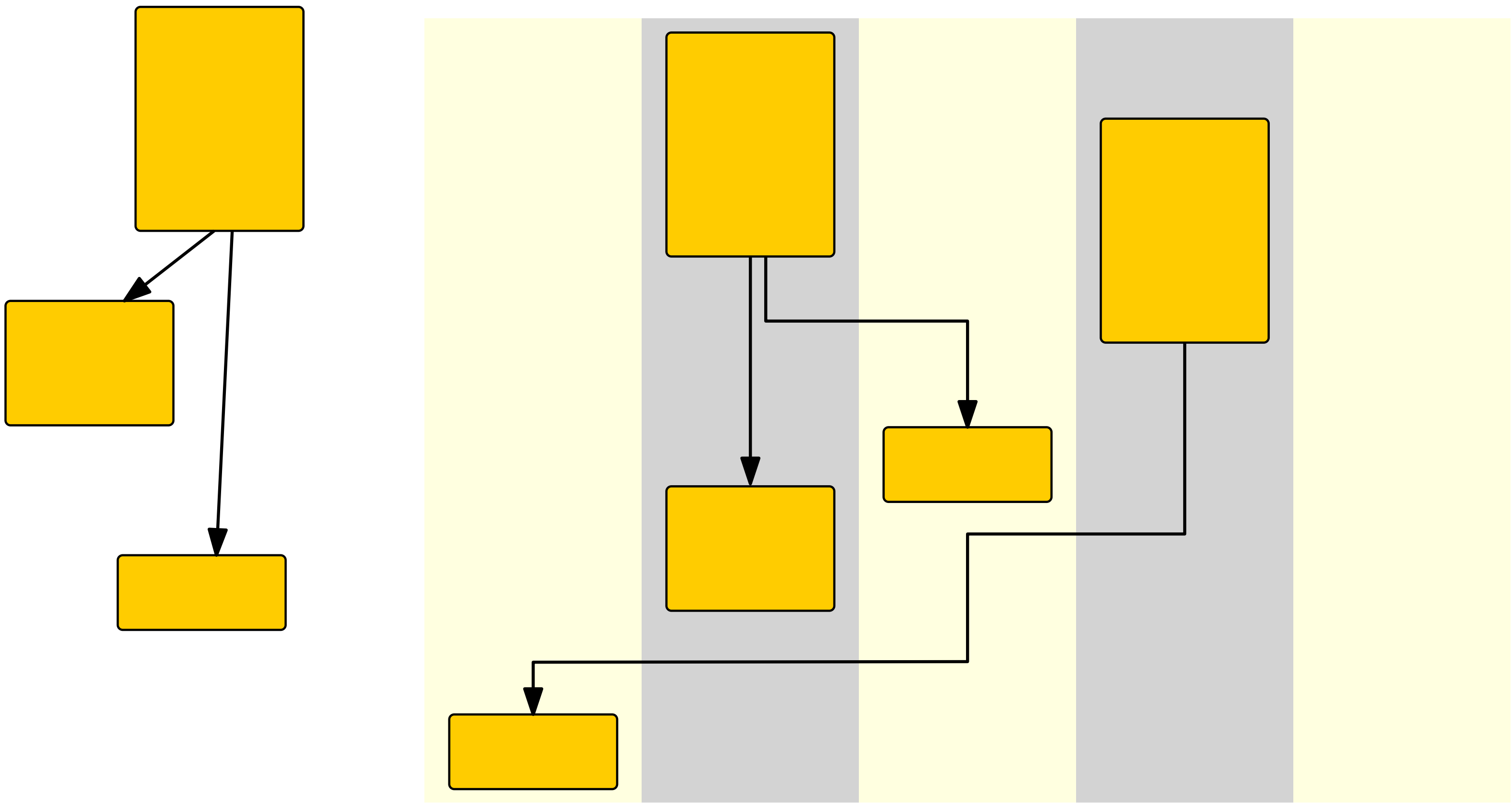
Shape – Assign Each Node & Edge a Column

goal: find realizeable column assignment for each node and edge



Shape – Assign Each Node & Edge a Column

goal: find realizeable column assignment for each node and edge



Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant ’98]

goal: find realizeable column assignment for each node and edge

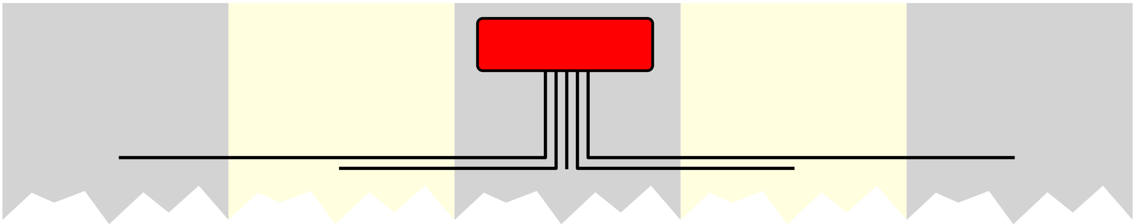
Theorem 1
There exists an algorithm that computes a realizable column assigment in $O(|E|)$.

Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant '98]

goal: find realizeable column assignment for each node and edge

draw from top to bottom



- start with super source
- draw vertical and horizontal part of edges

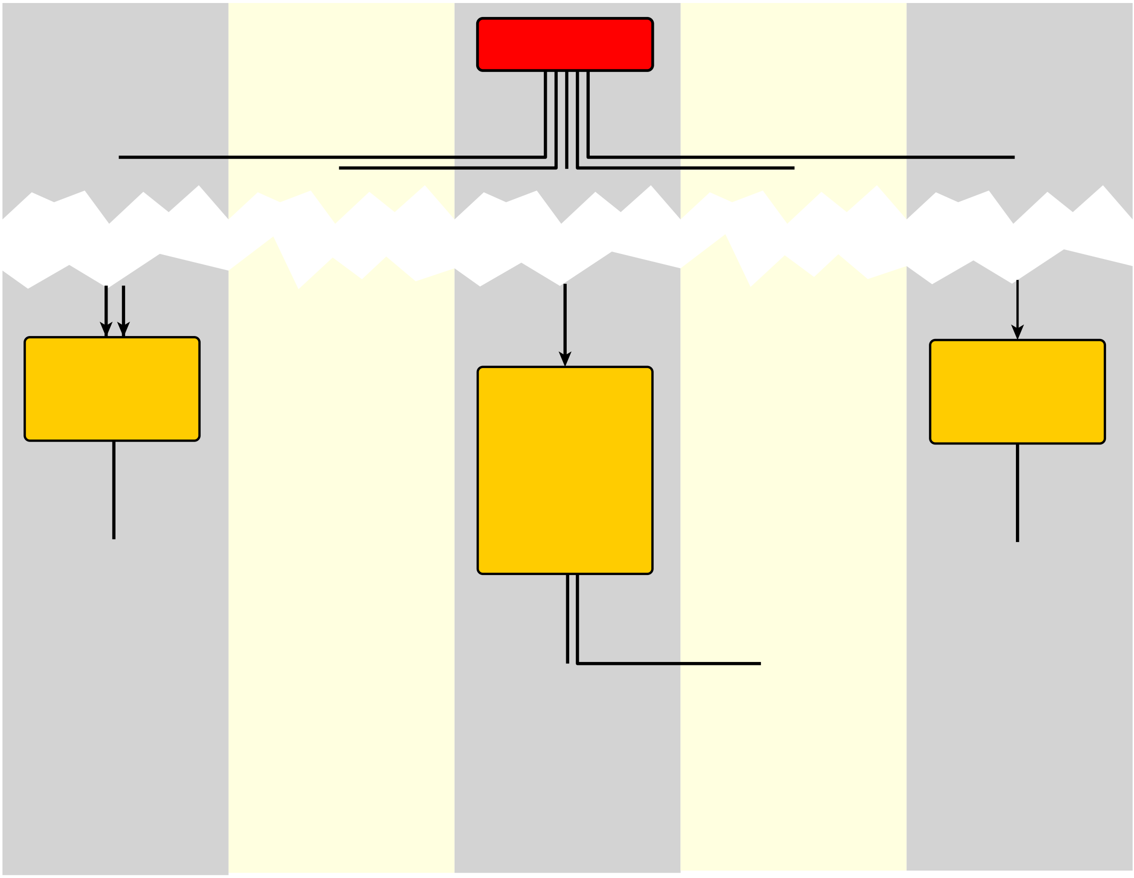
Theorem 1
There exists an algorithm that computes a realizable column assigment in $O(|E|)$.

Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant ’98]

goal: find realizeable column assignment for each node and edge

draw from top to bottom

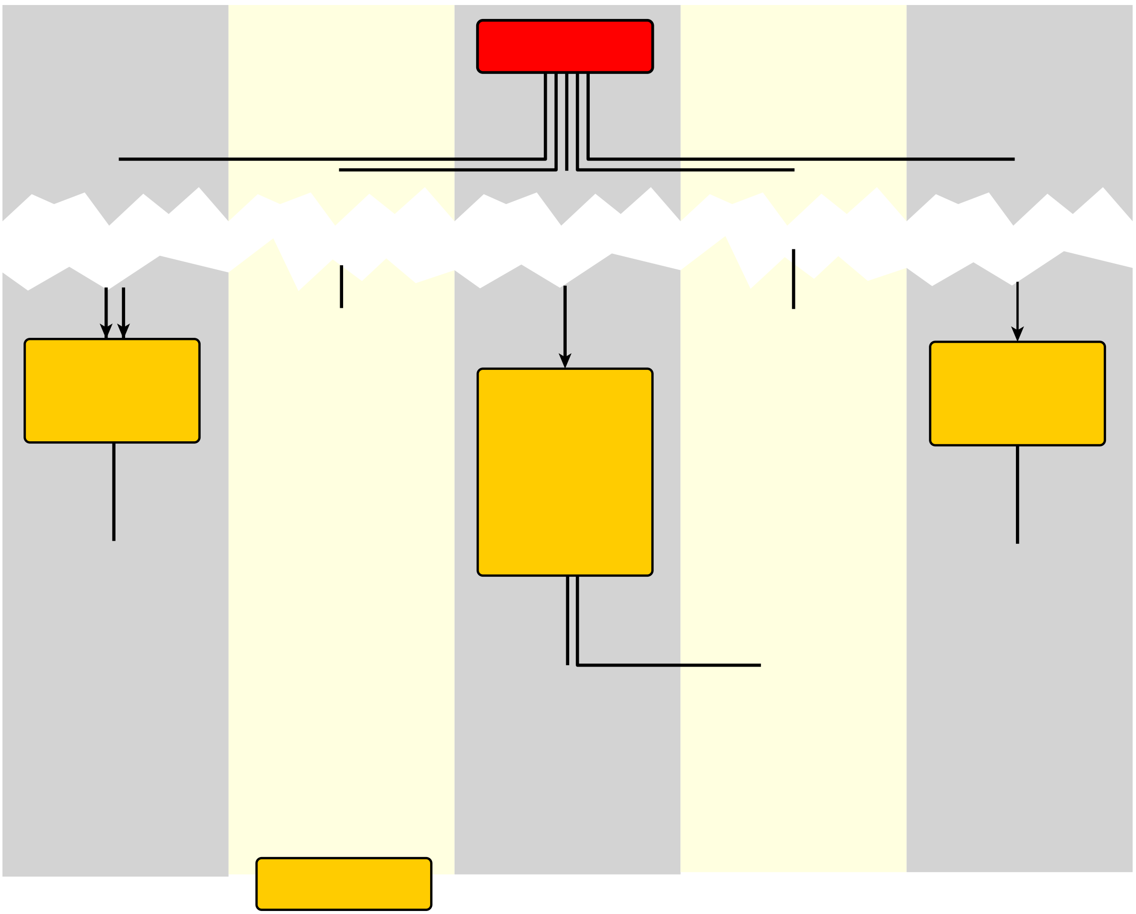


Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant '98]

goal: find realizeable column assignment for each node and edge

draw from top to bottom

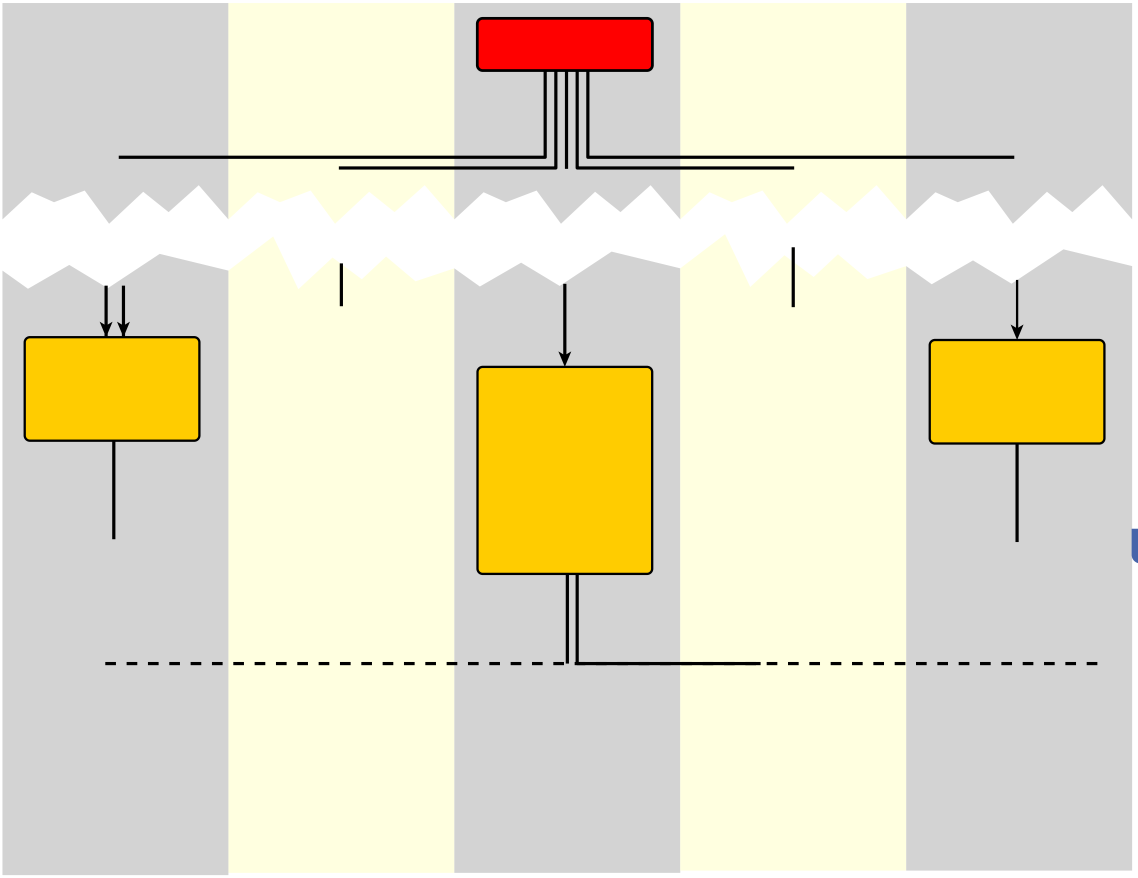


Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant '98]

goal: find realizeable column assignment for each node and edge

draw from top to bottom



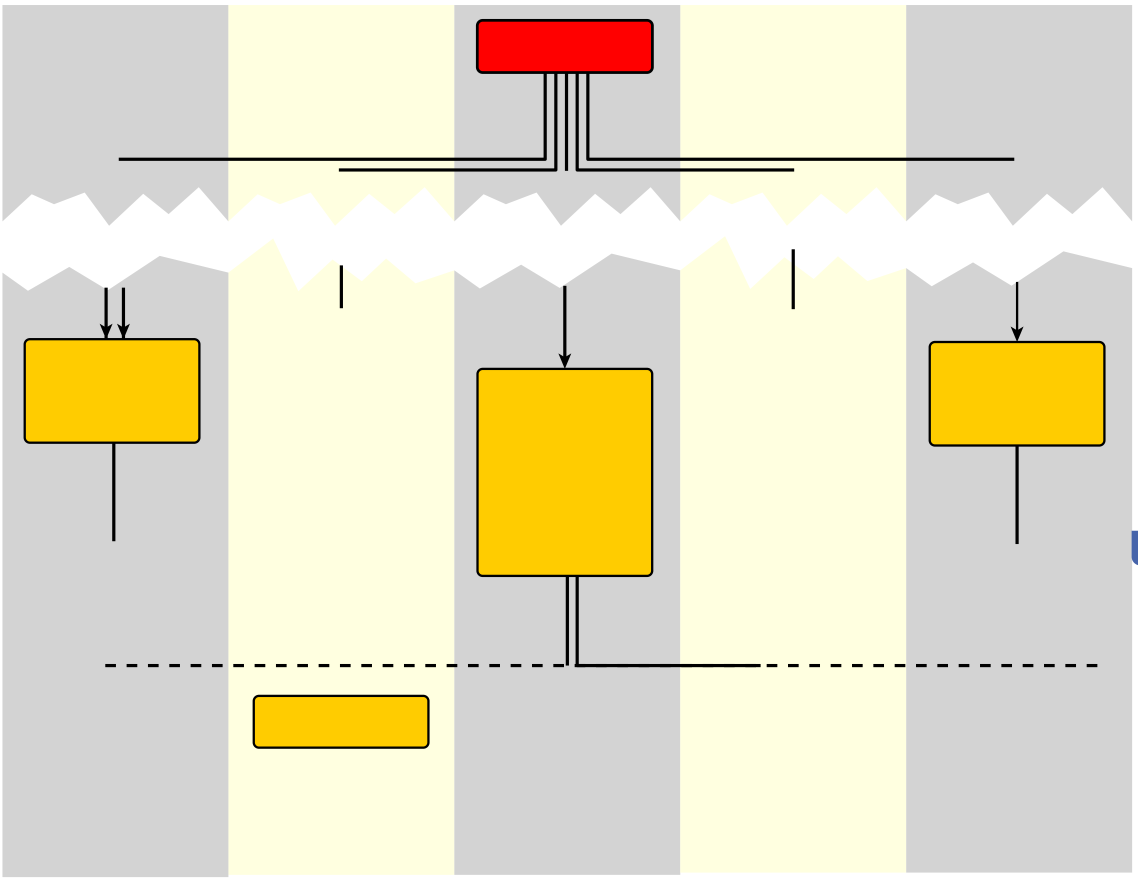
- find highest position

Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant ’98]

goal: find realizeable column assignment for each node and edge

draw from top to bottom



- find highest position

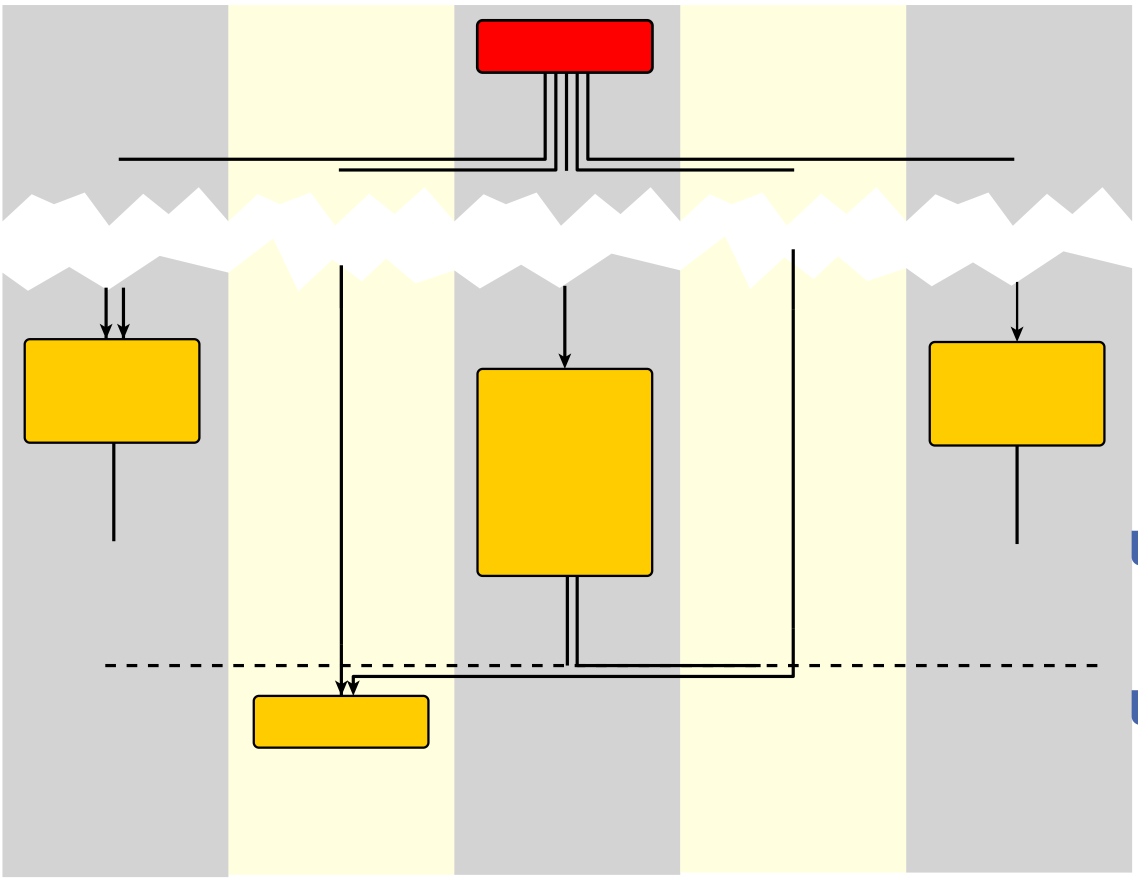
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant ’98]

goal: find realizeable column assignment for each node and edge

draw from top to bottom



- find highest position
- draw inedges

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

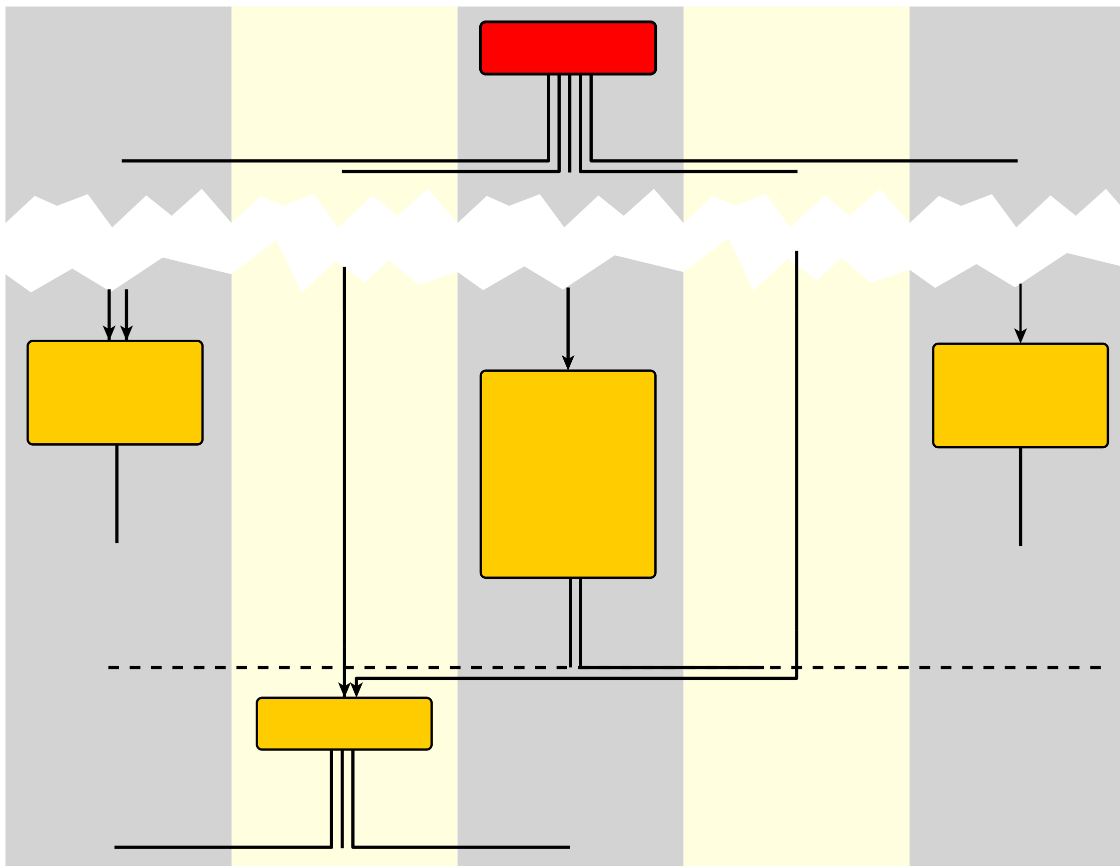
Shape – Assign Each Node & Edge a Column

Karlsruhe Institute of Technology

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant '98]

goal: find realizeable column assignment for each node and edge

draw from top to bottom



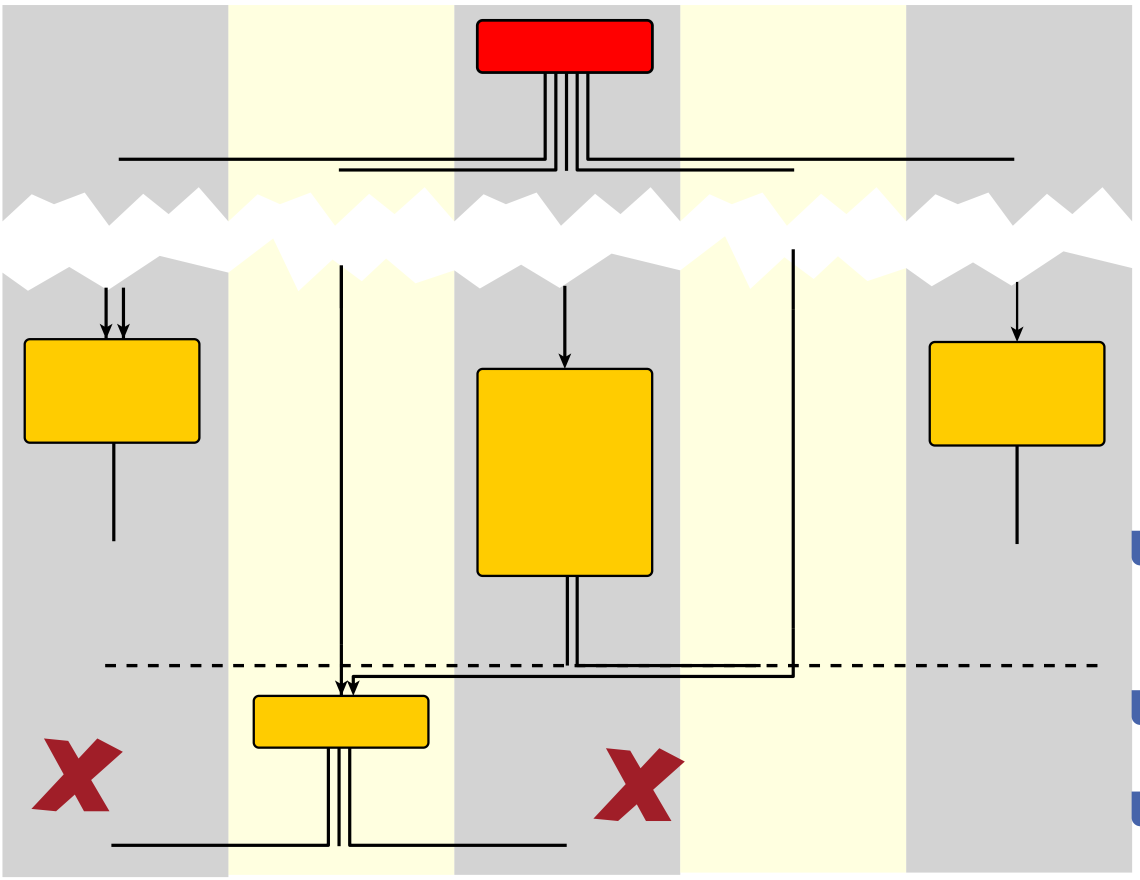
- find highest position
- draw inedges
- draw outedges

Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant '98]

goal: find realizeable column assignment for each node and edge

draw from top to bottom



- find highest position
- draw inedges
- draw outedges

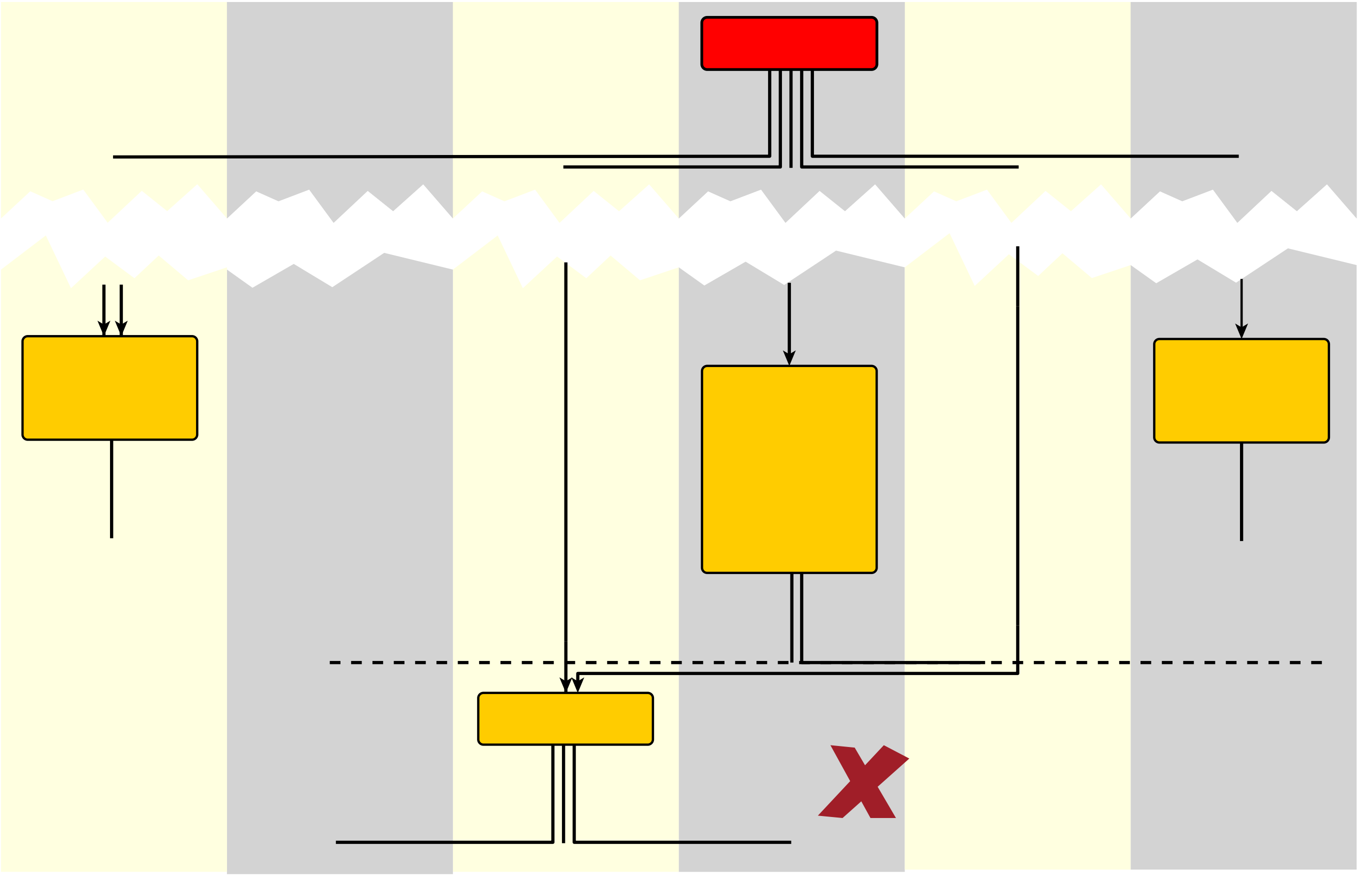
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant ’98]

goal: find realizeable column assignment for each node and edge

draw from top to bottom

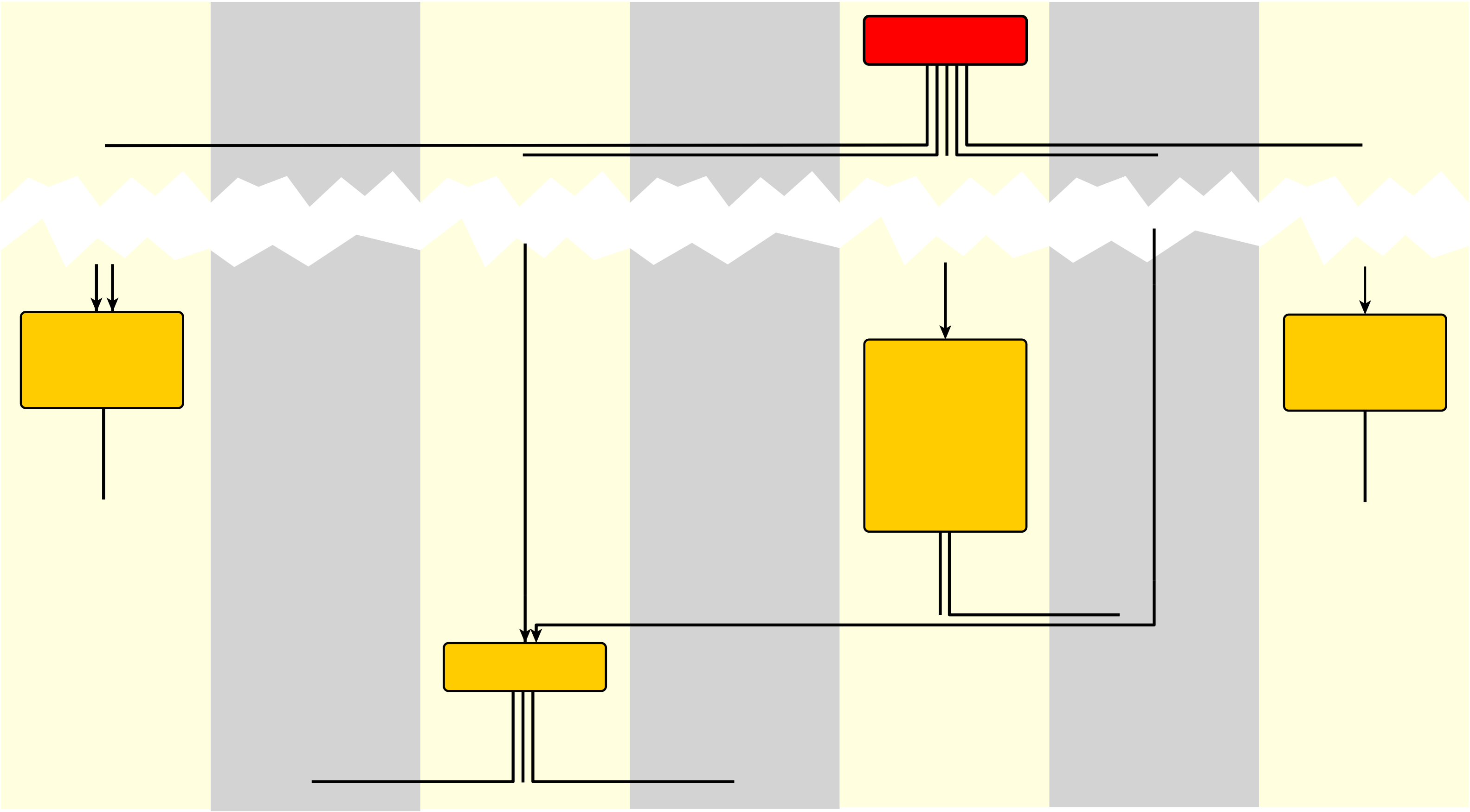


Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant '98]

goal: find realizeable column assignment for each node and edge

draw from top to bottom



Shape – Assign Each Node & Edge a Column

Better Heuristic for Orthogonal Graph Drawings [Biedl, Kant ’98]

goal: find realizeable column assignment for each node and edge

Quality Guarantees

- Vertical alignment of predecessors
- Local symmetry ✓
- At most four bends per edge ✓
- Spacing constraints
- Edge Bundling

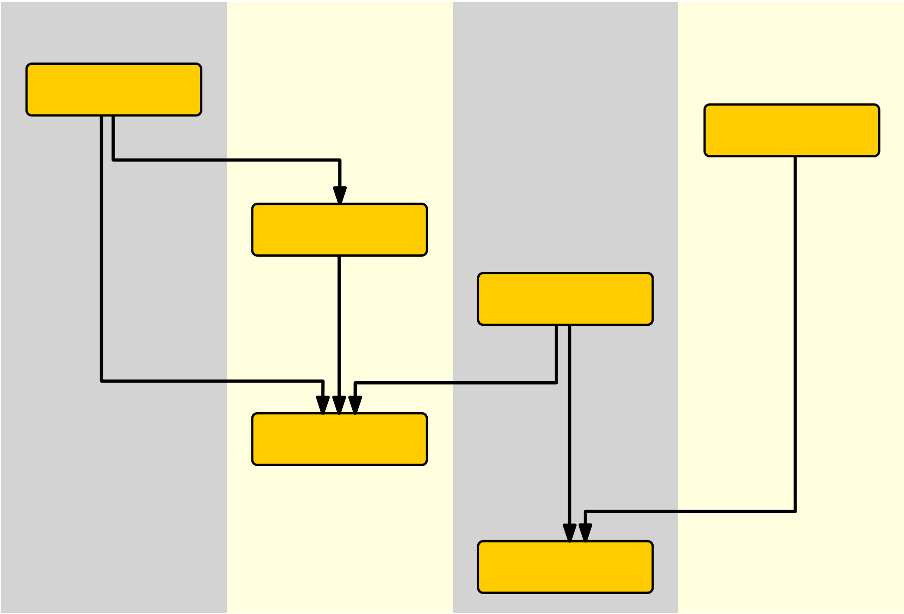
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Metrics – Compute final coordinates

goal: minimize total vertical edge length

Theorem 2

Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.



Metrics – Compute final coordinates

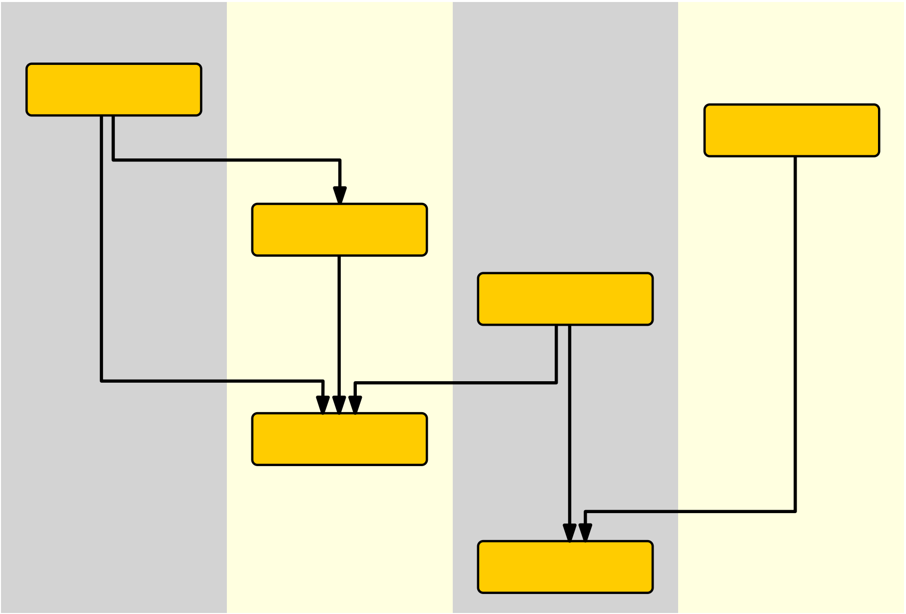
goal: minimize total vertical edge length

Theorem 2

Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:

(i) assign every node to a group [vertical alignment of predecessors]



Metrics – Compute final coordinates

Karlsruhe Institute of Technology

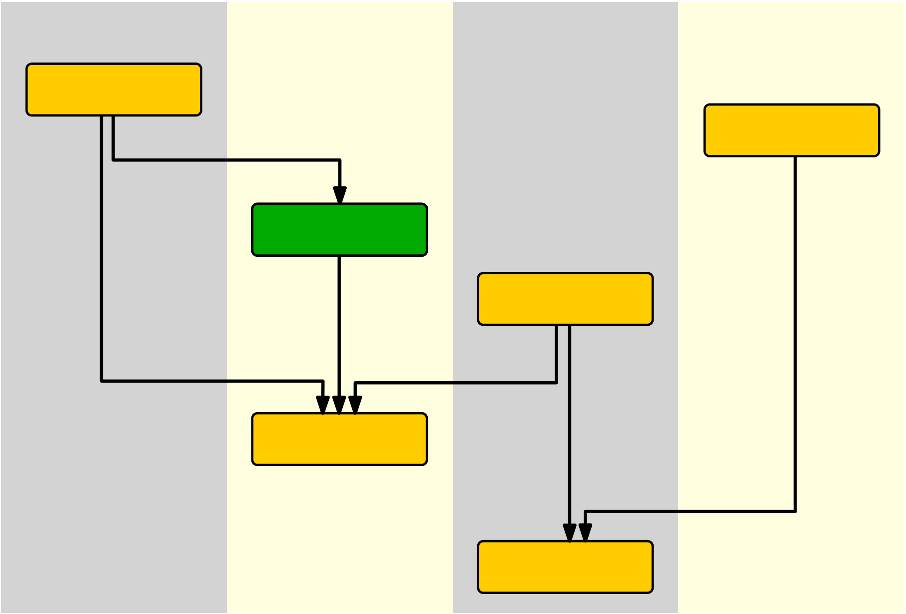
goal: minimize total vertical edge length

Theorem 2

Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:

(i) assign every node to a group [vertical alignment of predecessors]



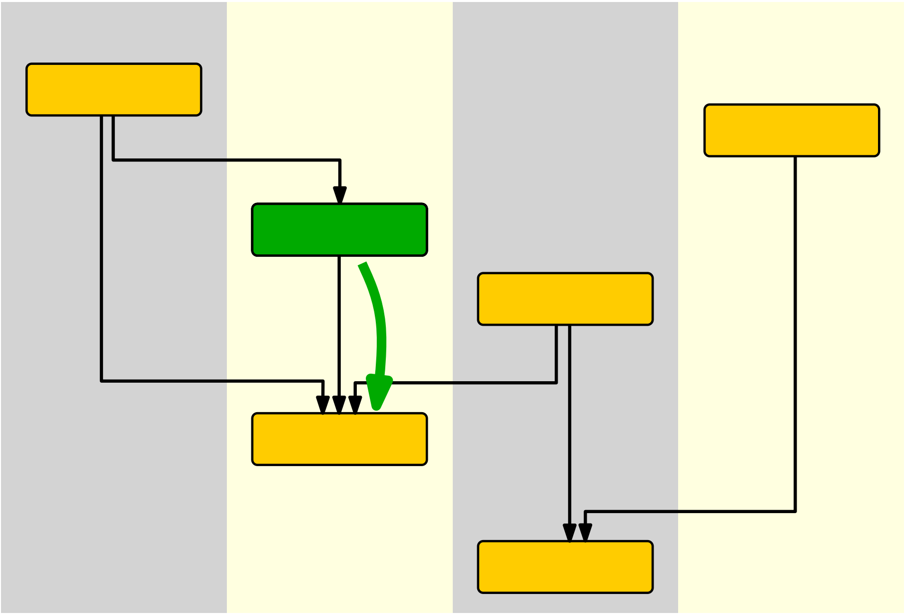
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Metrics – Compute final coordinates

goal: minimize total vertical edge length

Theorem 2
Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:
(i) assign every node to a group [vertical alignment of predecessors]

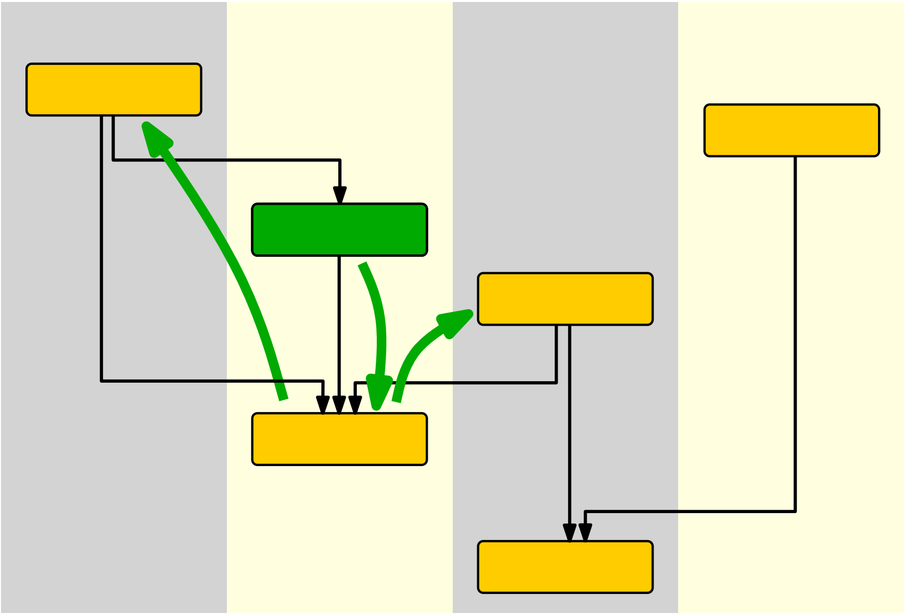


Metrics – Compute final coordinates

goal: minimize total vertical edge length

Theorem 2
Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:
(i) assign every node to a group [vertical alignment of predecessors]



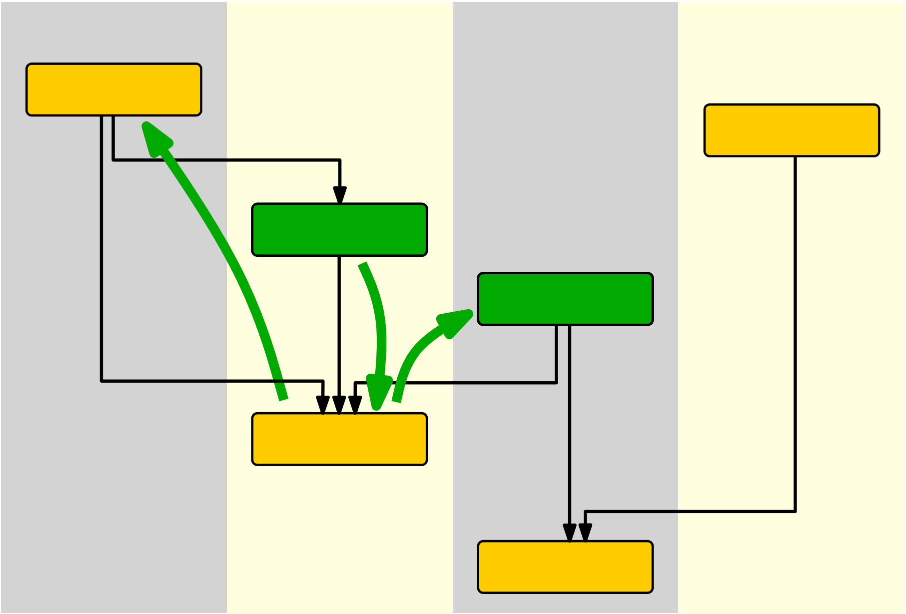
Metrics – Compute final coordinates

goal: minimize total vertical edge length

Theorem 2
Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:

(i) assign every node to a group [vertical alignment of predecessors]

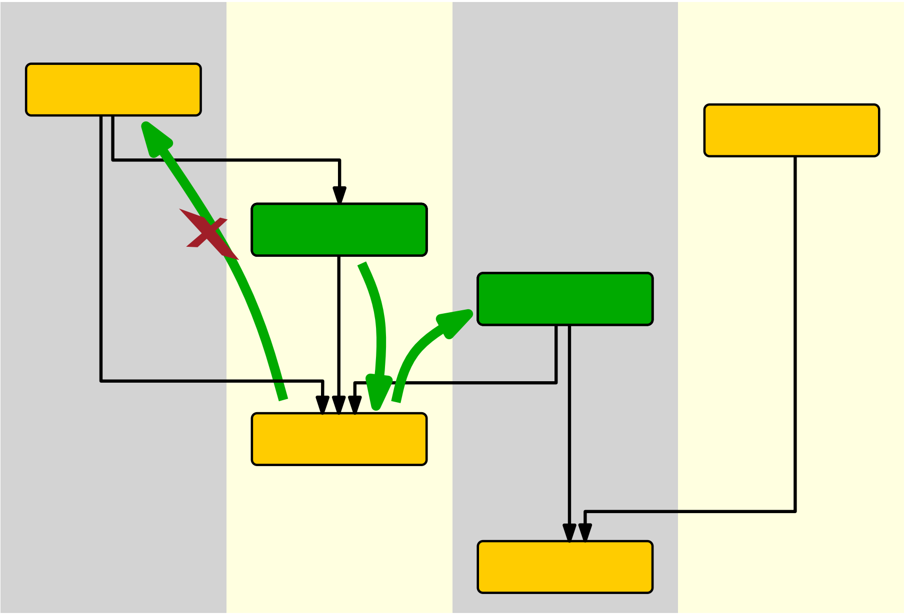


Metrics – Compute final coordinates

goal: minimize total vertical edge length

Theorem 2
Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:
(i) assign every node to a group [vertical alignment of predecessors]



Metrics – Compute final coordinates

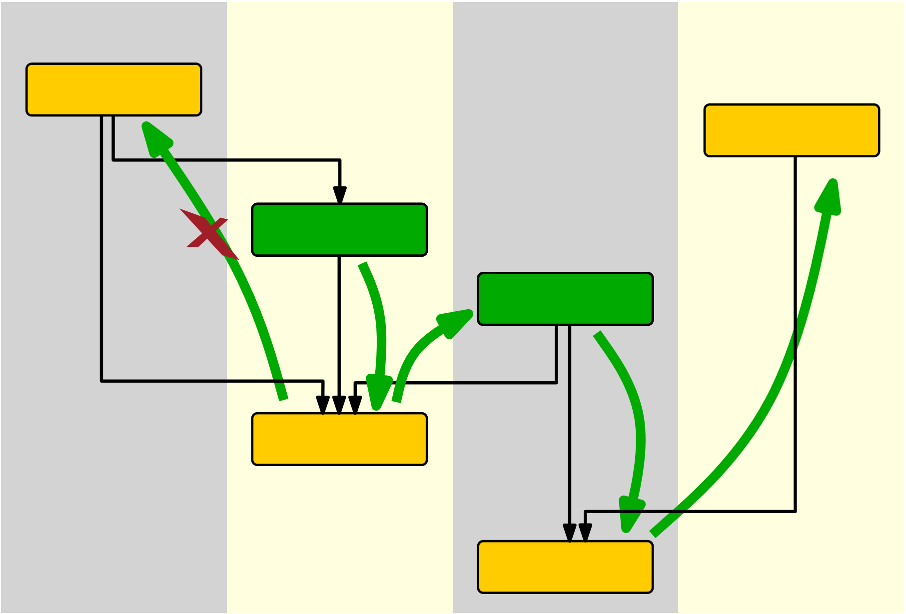
goal: minimize total vertical edge length

Theorem 2

Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:

(i) assign every node to a group [vertical alignment of predecessors]



Metrics – Compute final coordinates

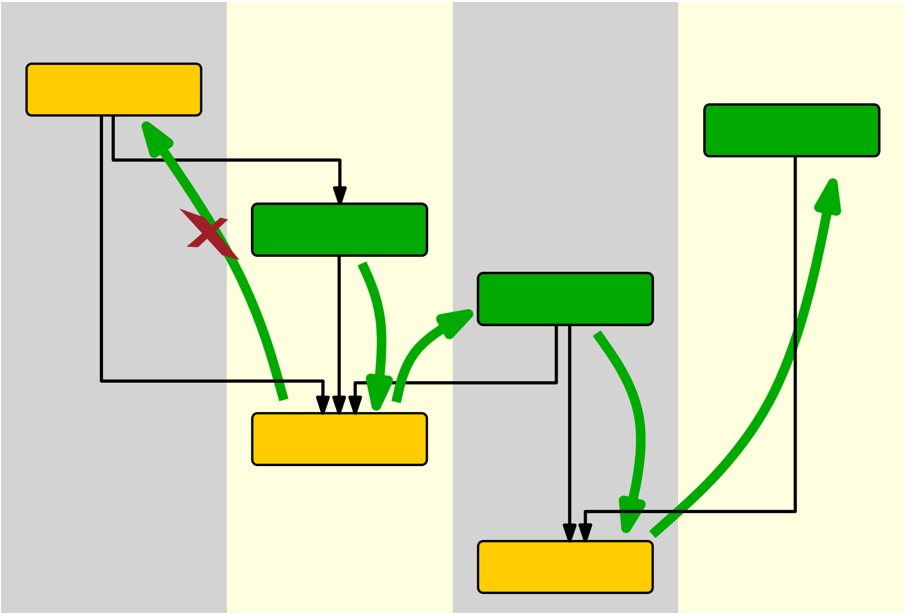
goal: minimize total vertical edge length

Theorem 2

Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:

(i) assign every node to a group [vertical alignment of predecessors]

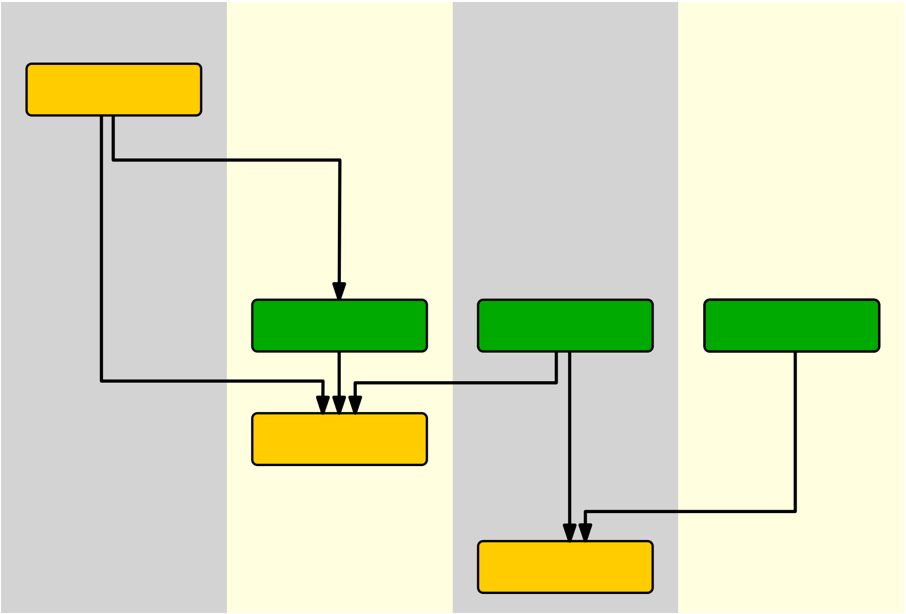


Metrics – Compute final coordinates

goal: minimize total vertical edge length

Theorem 2
Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:
(i) assign every node to a group [vertical alignment of predecessors]



Metrics – Compute final coordinates

goal: minimize total vertical edge length

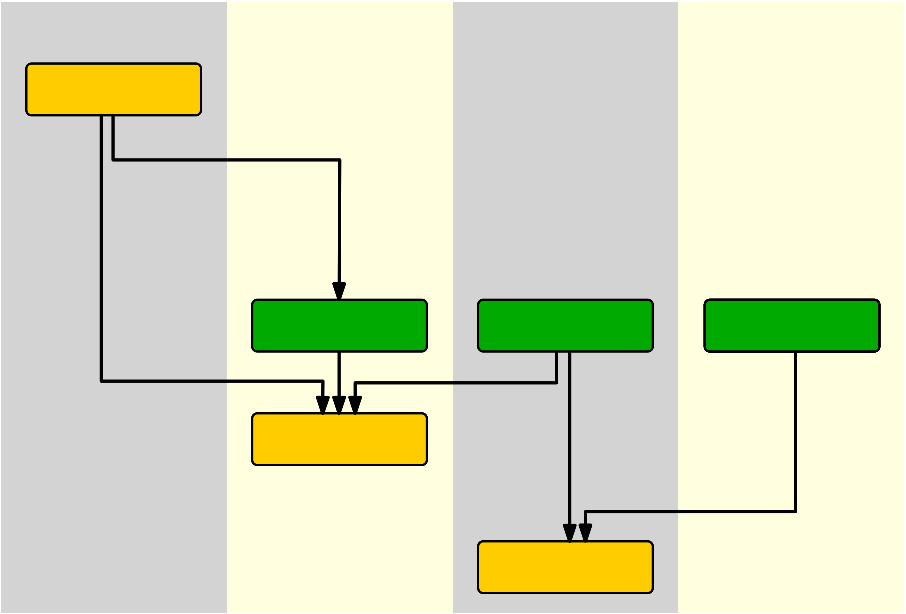
Theorem 2

Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:

(i) assign every node to a group [vertical alignment of predecessors]

(ii) draw bottom up



Metrics – Compute final coordinates

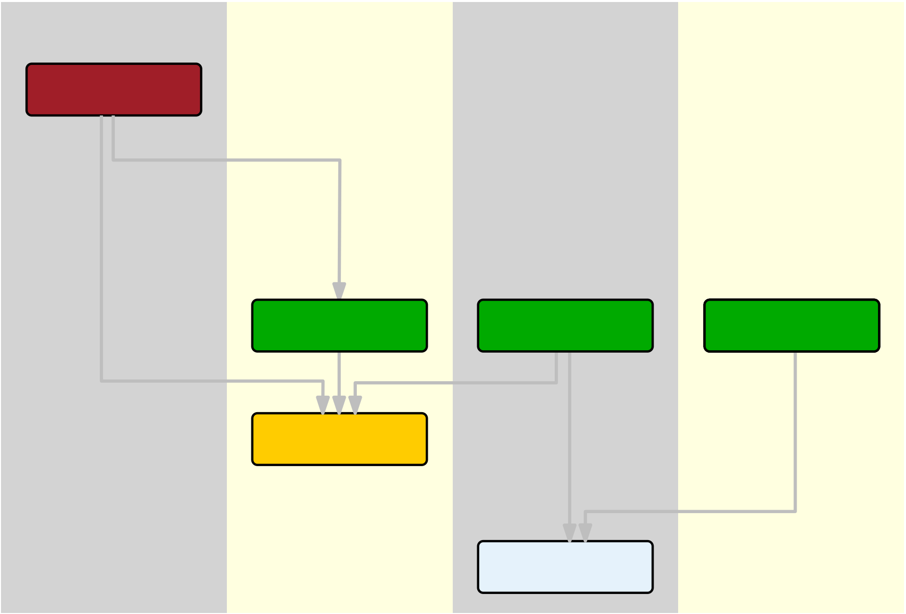
goal: minimize total vertical edge length

Theorem 2
Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:

(i) assign every node to a group [vertical alignment of predecessors]

(ii) draw bottom up



Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Metrics – Compute final coordinates

goal: minimize total vertical edge length

Theorem 2

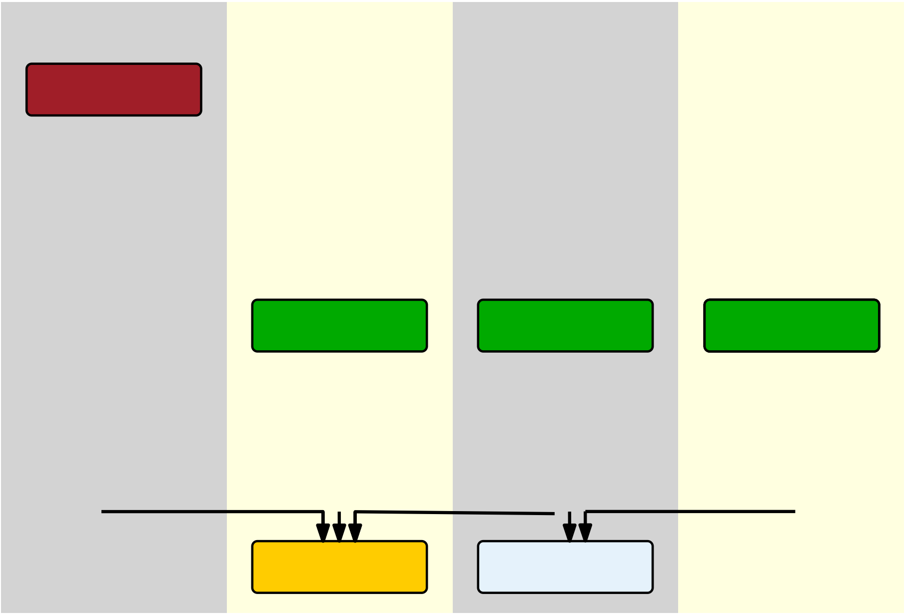
Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:

(i) assign every node to a group [vertical alignment of predecessors]

(ii) draw bottom up

- draw horizontal part of inedges



Metrics – Compute final coordinates

goal: minimize total vertical edge length

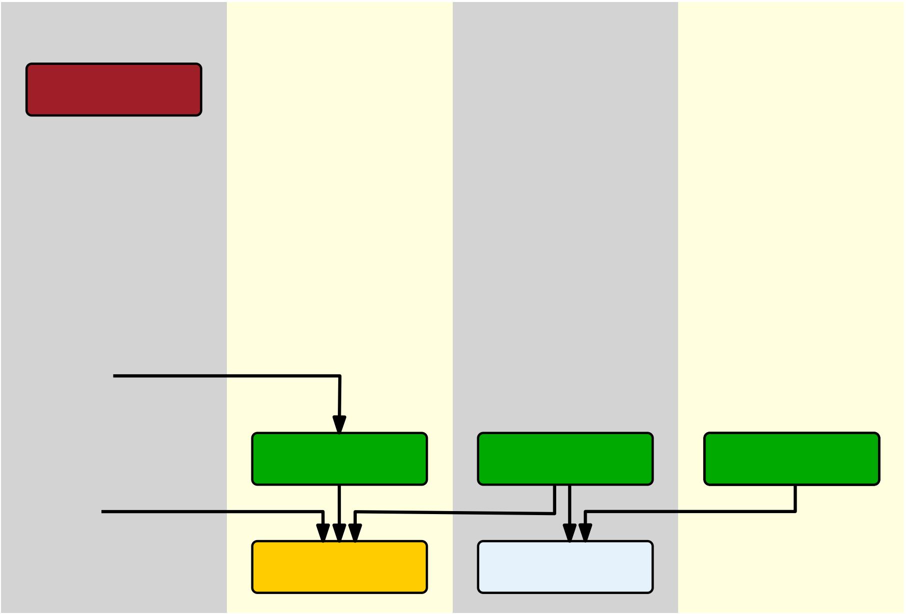
Theorem 2

Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:

(i) assign every node to a group [vertical alignment of predecessors]

(ii) draw bottom up



- draw horizontal part of inedges
- complete drawing of outedges

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Metrics – Compute final coordinates

goal: minimize total vertical edge length

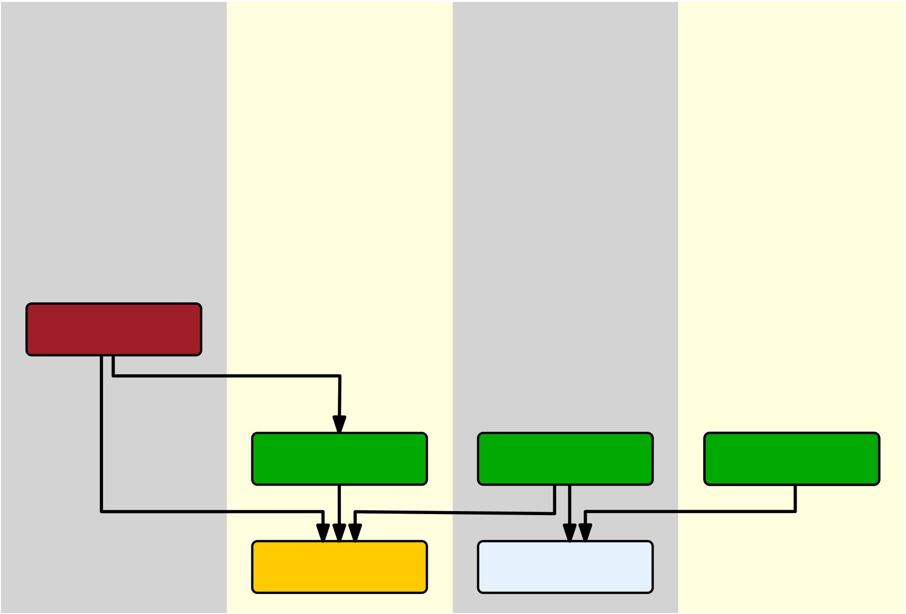
Theorem 2
Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:

(i) assign every node to a group [vertical alignment of predecessors]

(ii) draw bottom up

- draw horizontal part of inedges
- complete drawing of outedges



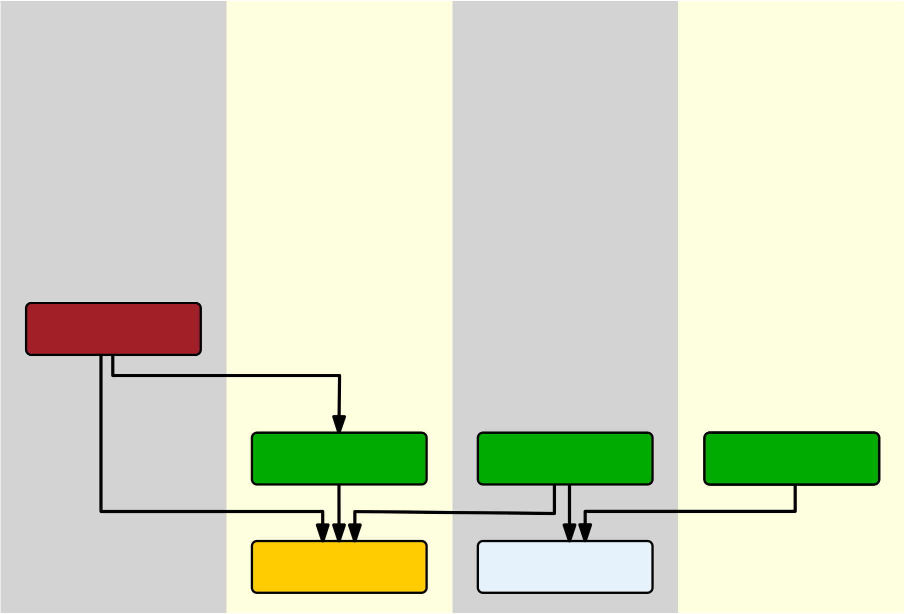
Metrics – Compute final coordinates

goal: minimize total vertical edge length

Theorem 2
Finding a realization of a column assignment with minimum total vertical edge length is $\mathcal{NP}$-complete.

greedy algorithm:
(i) assign every node to a group [vertical alignment of predecessors]

(ii) draw bottom up

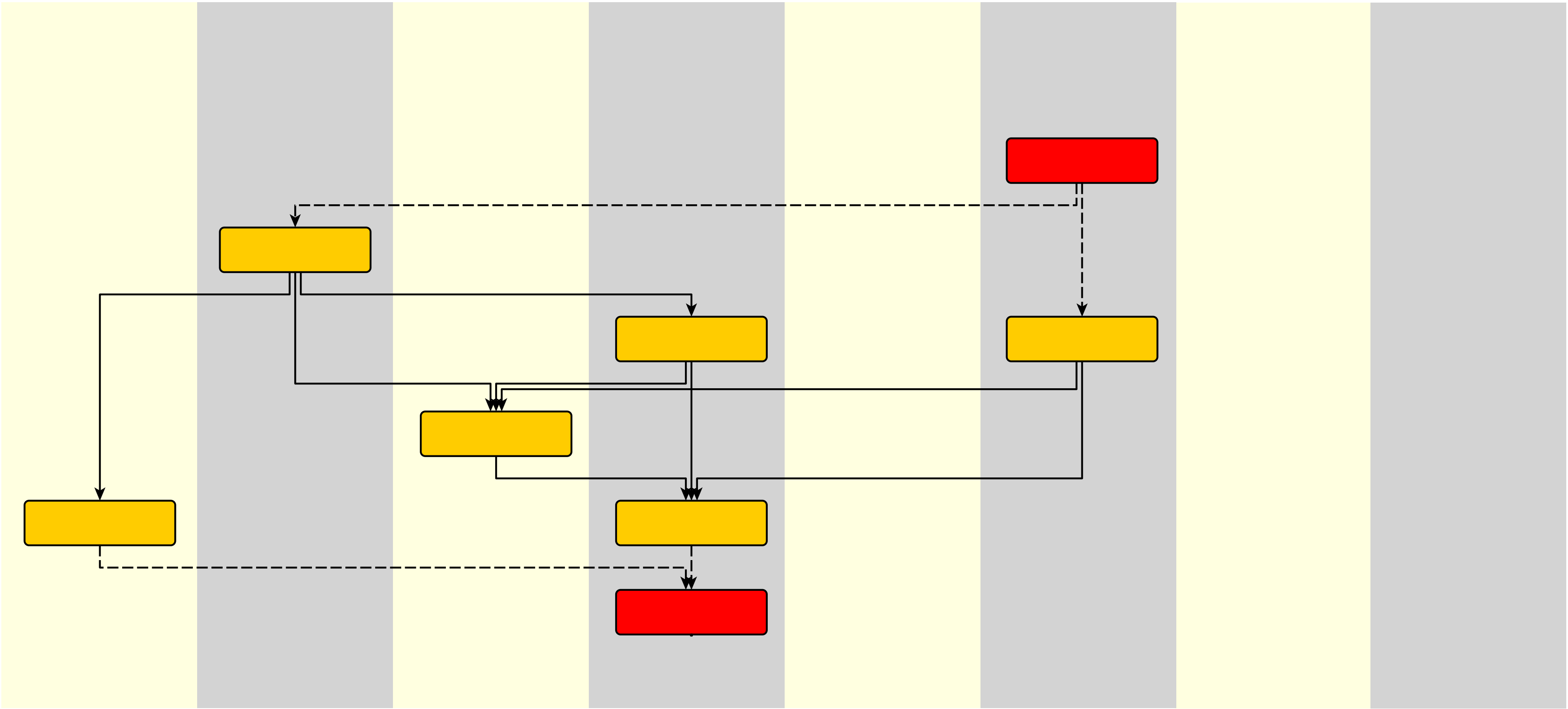


Theorem 3
The greedy algorithm requires $O((n+b)m)$ time.

b = #columns

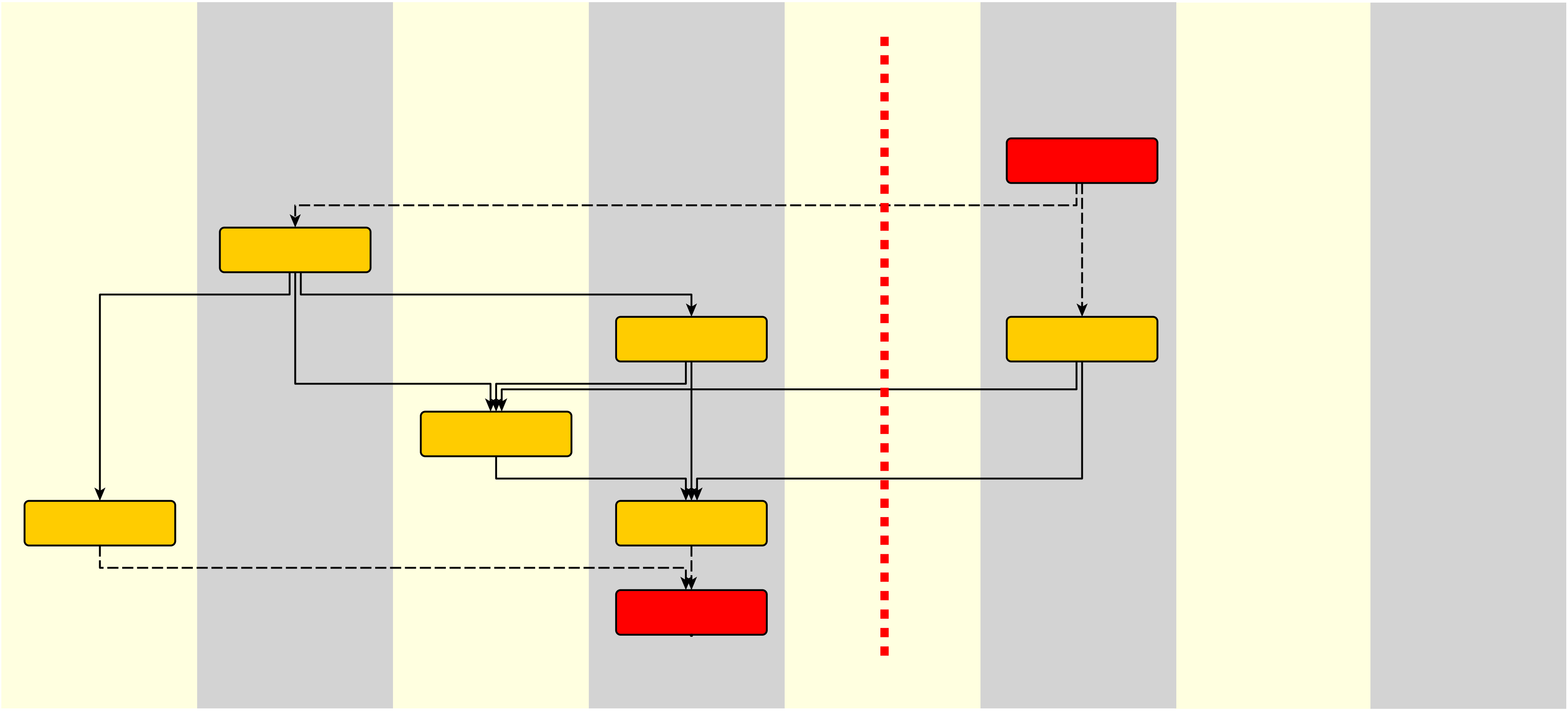
Metrics – Compute final coordinates

goal: minimize total horizontal edge length



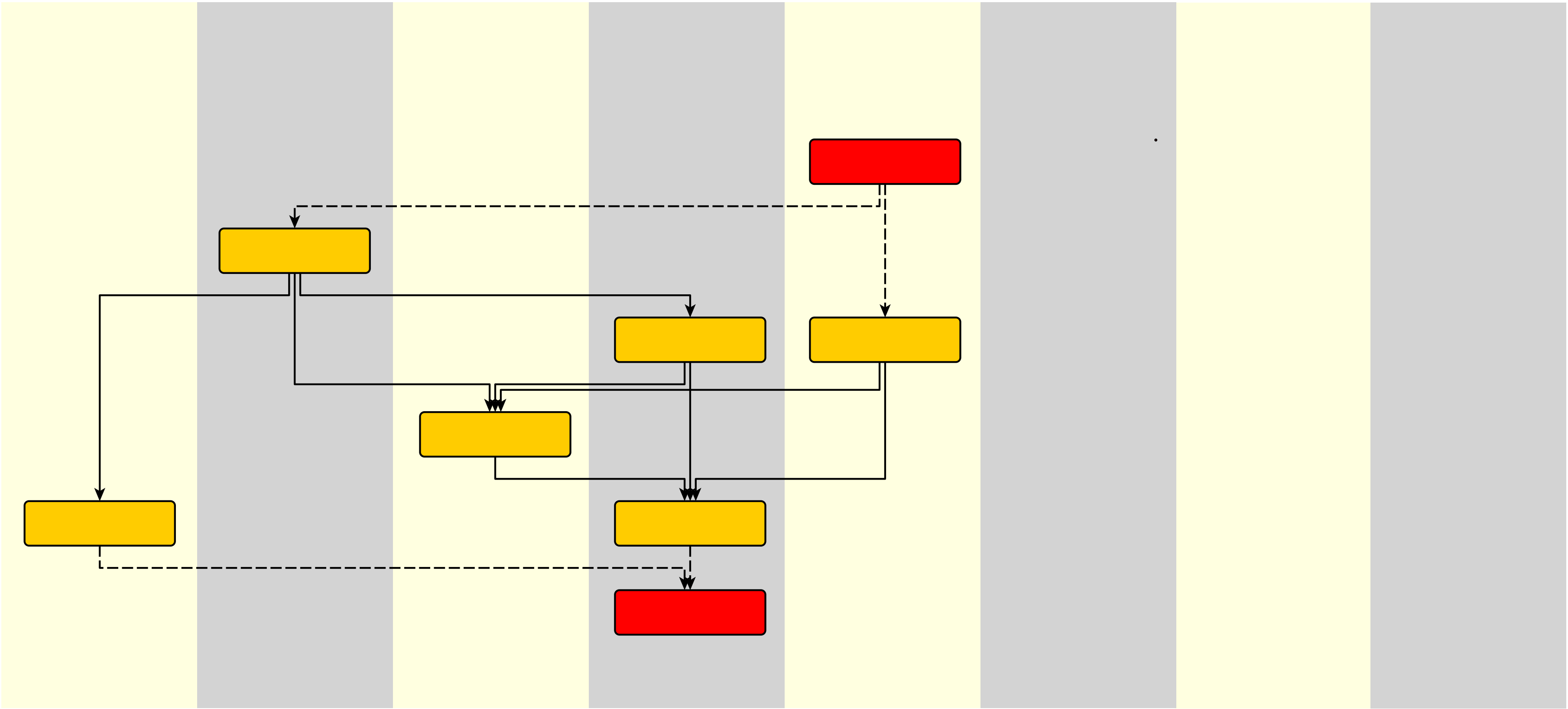
Metrics – Compute final coordinates

goal: minimize total horizontal edge length



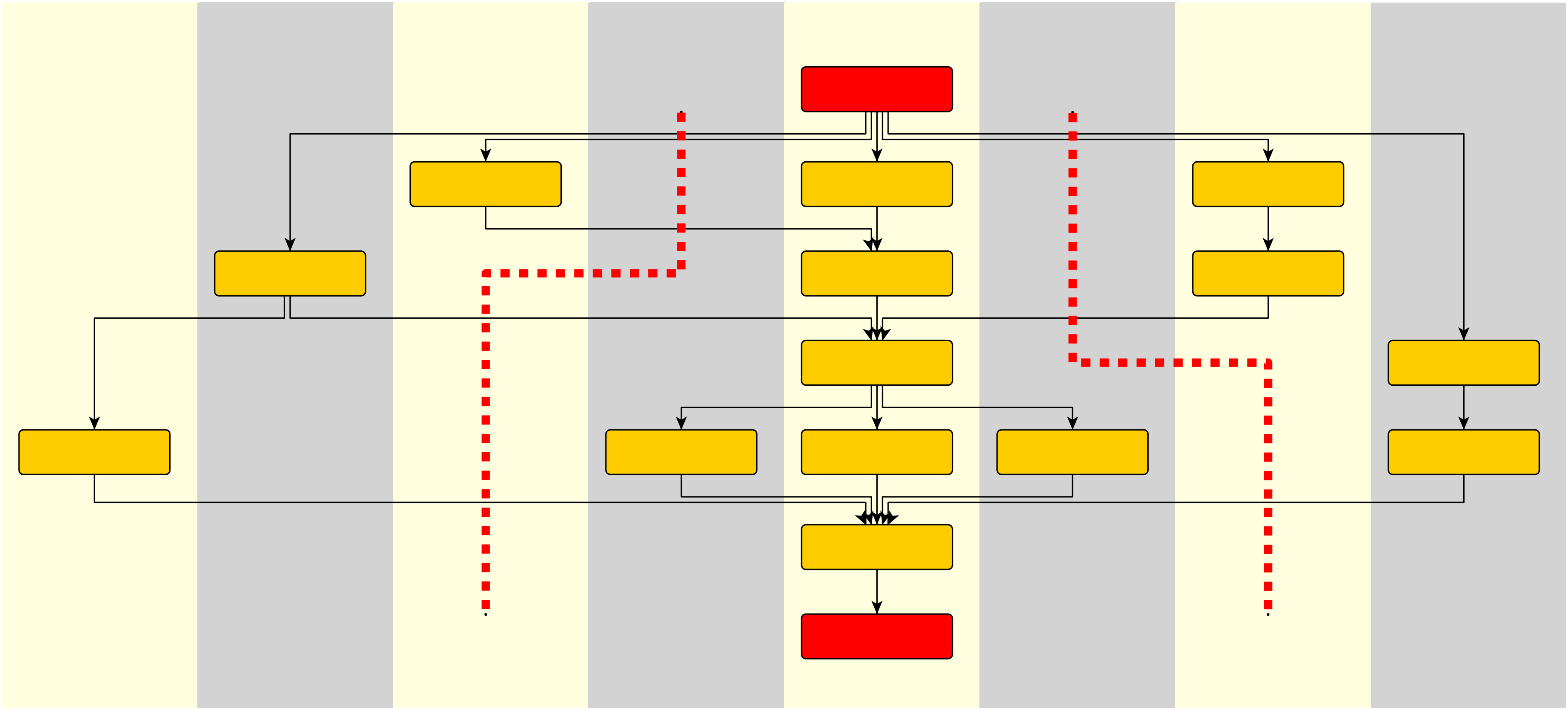
Metrics – Compute final coordinates

goal: minimize total horizontal edge length



Metrics – Compute final coordinates

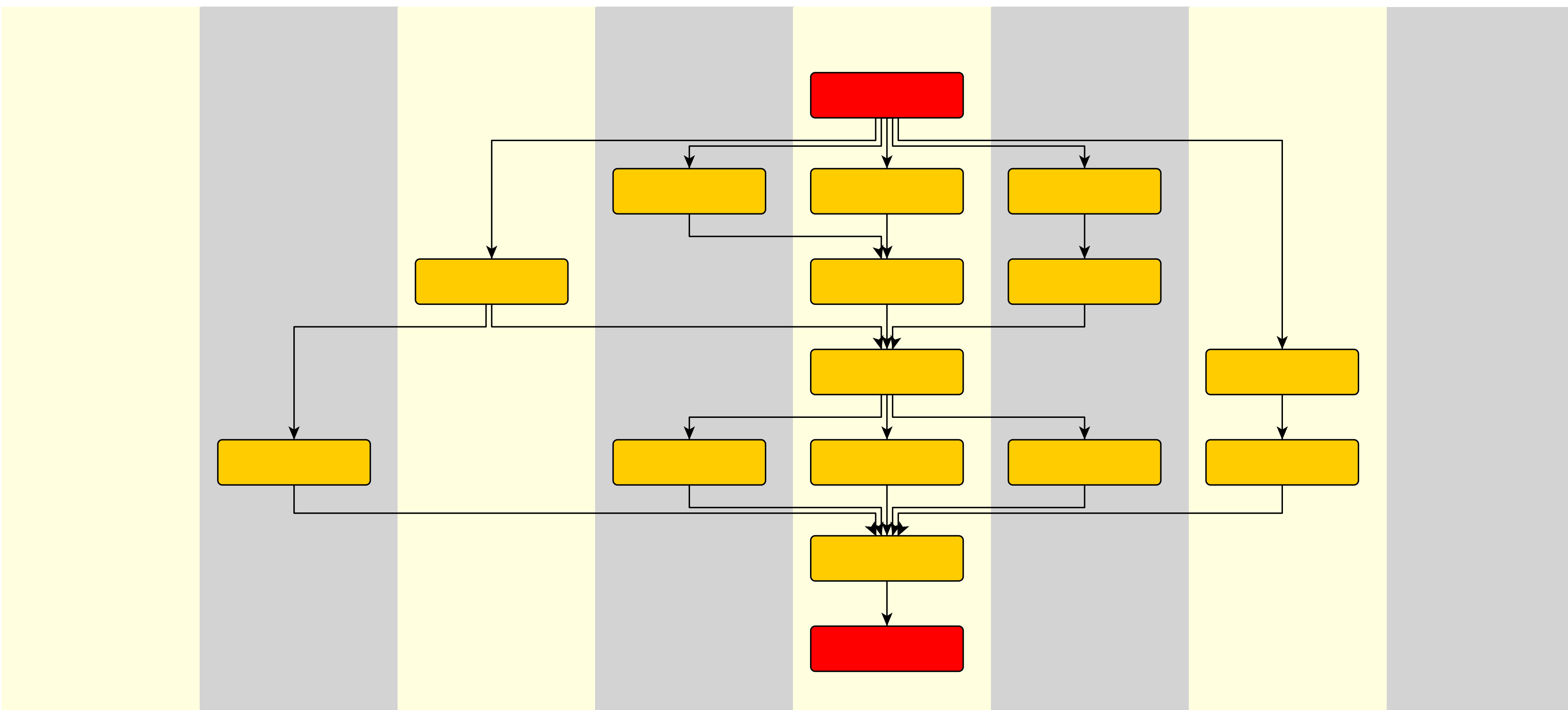
goal: minimize total horizontal edge length



Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

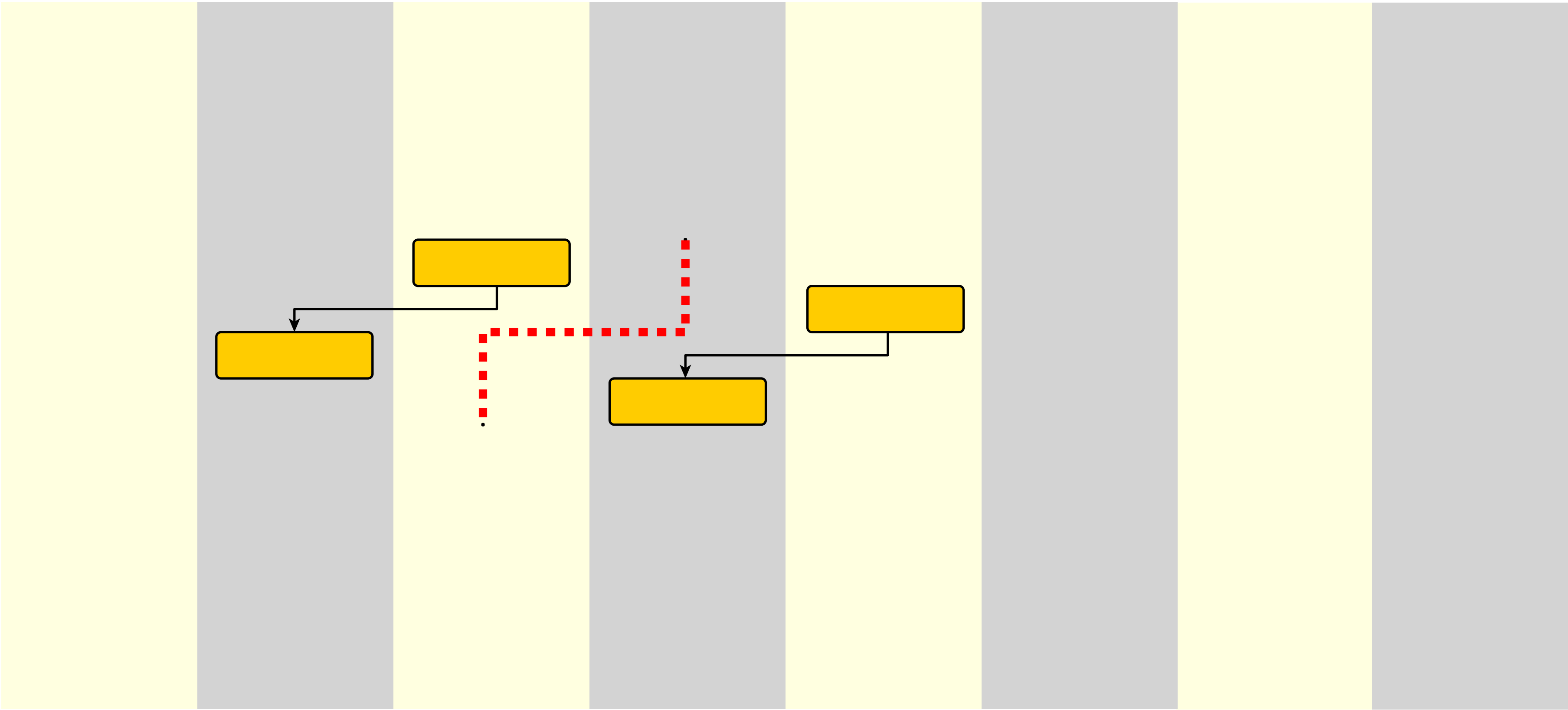
Metrics – Compute final coordinates

goal: minimize total horizontal edge length



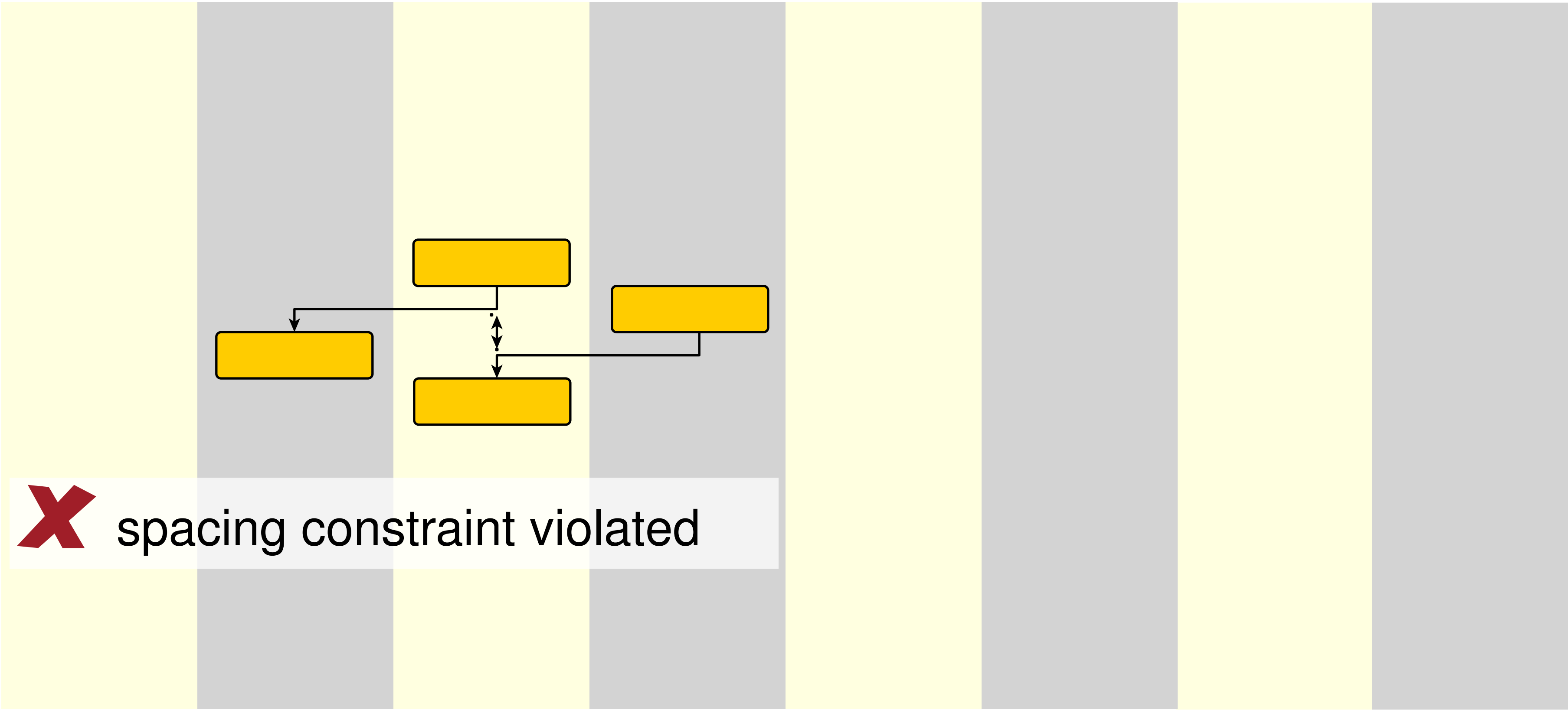
Metrics – Compute final coordinates

goal: minimize total horizontal edge length



Metrics – Compute final coordinates

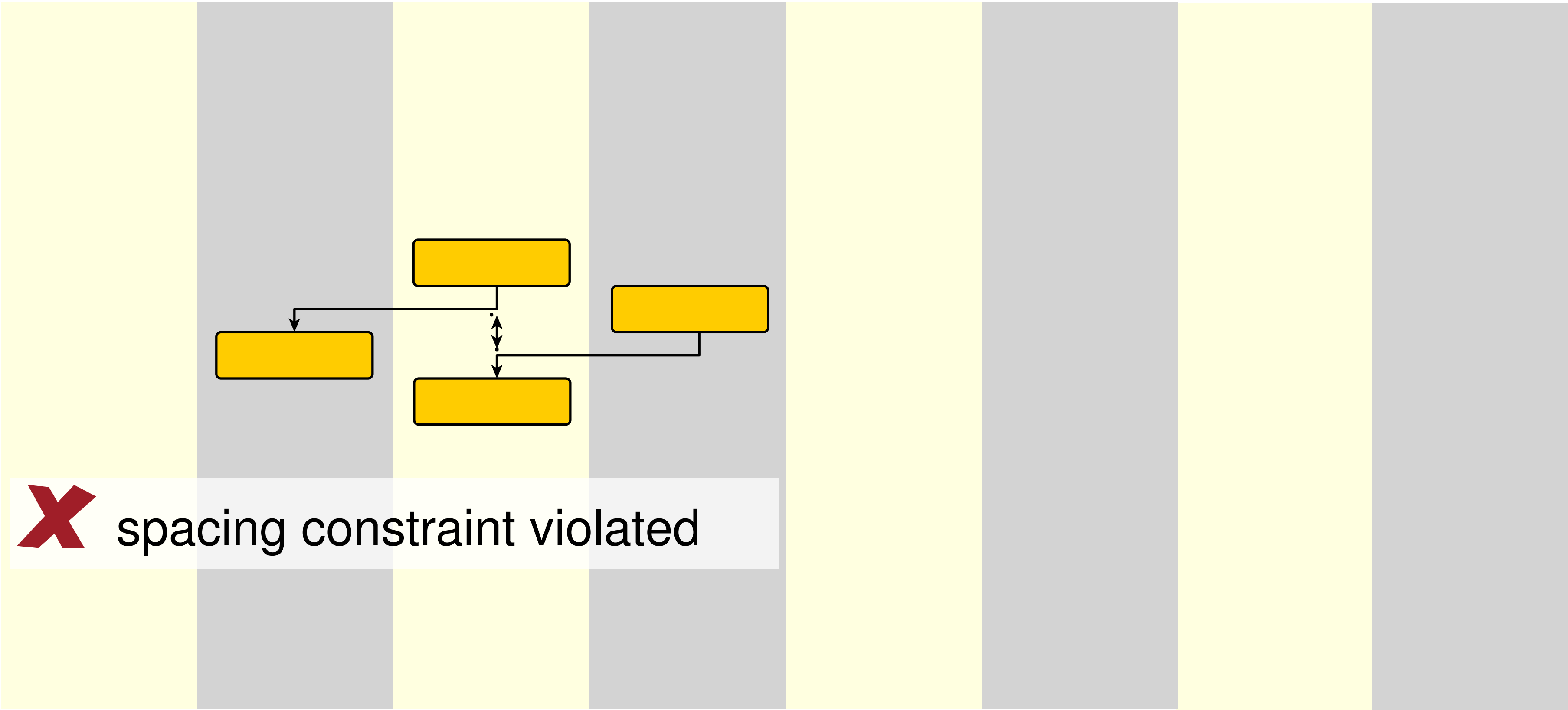
goal: minimize total horizontal edge length



Metrics – Compute final coordinates

goal: minimize total horizontal edge length

idea:
- determine compaction paths
- compact along them

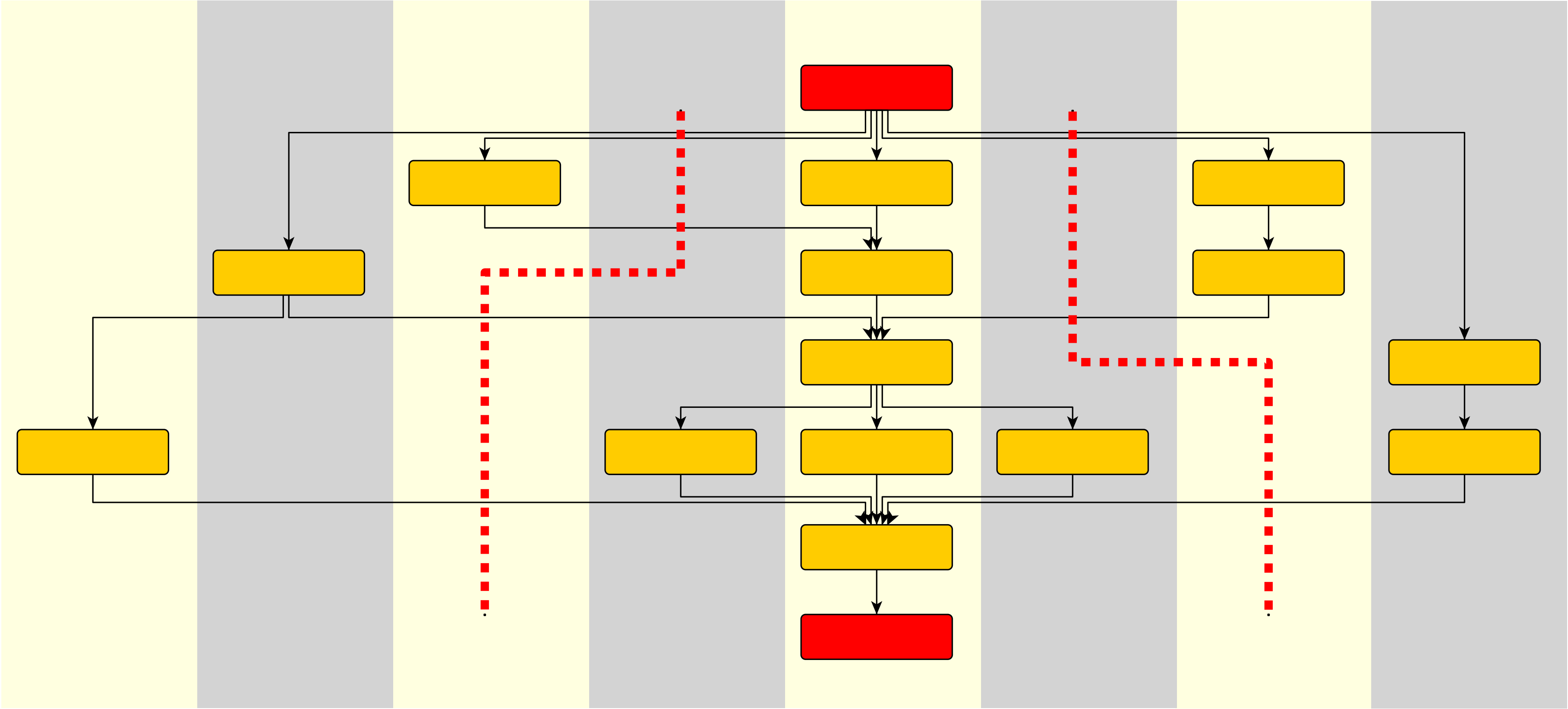


Metrics – Compute final coordinates

goal: minimize total horizontal edge length

idea:

- determine compaction paths
- compact along them

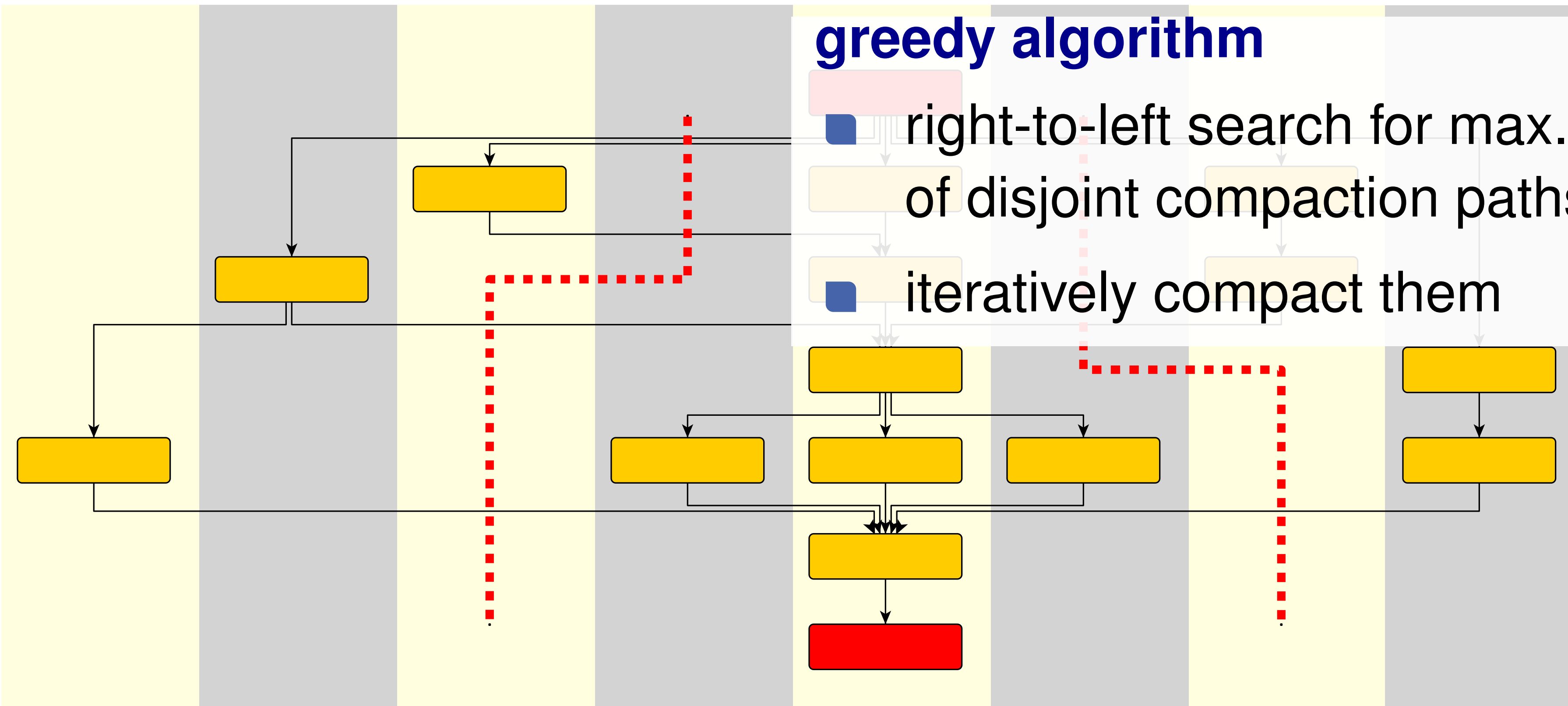


Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Metrics – Compute final coordinates

goal: minimize total horizontal edge length

idea:
- determine compaction paths
- compact along them

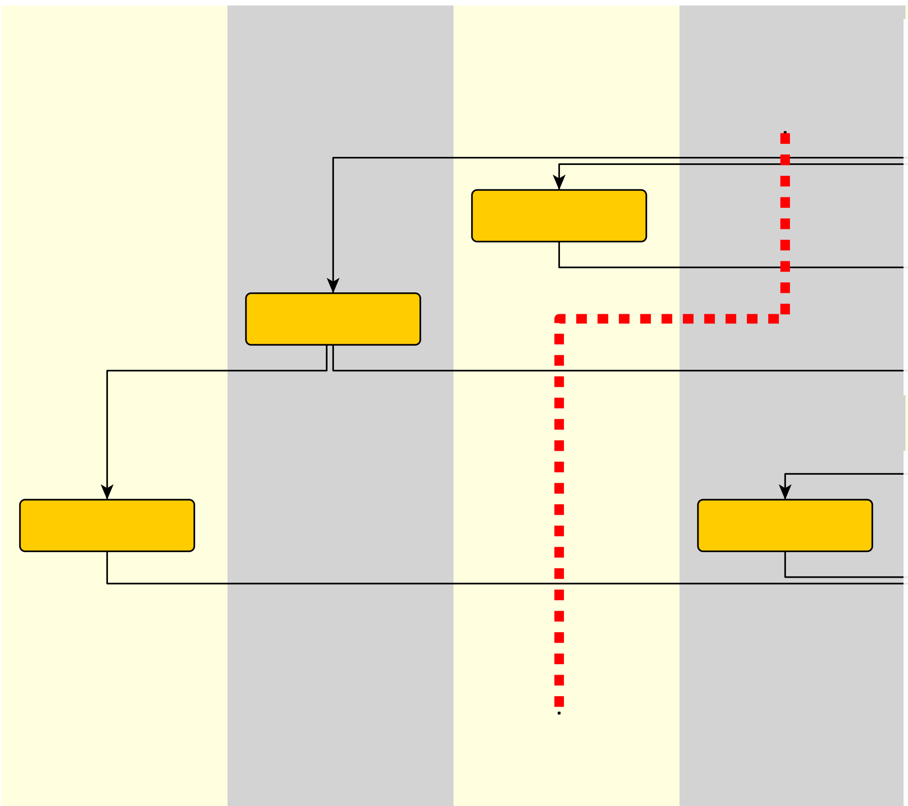


greedy algorithm
- right-to-left search for max. set of disjoint compaction paths
- iteratively compact them

Metrics – Compute final coordinates

goal: minimize total horizontal edge length

idea:
- determine compaction paths
- compact along them



greedy algorithm
- right-to-left search for max. set of disjoint compaction paths
- iteratively compact them

Theorem 4
The greedy algorithm requires $O(mb(b + \log n))$ time.

b = #columns

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Our Approach – Theoretical Results

Quality Guarantees

- Vertical alignment of predecessors ✓
- Local symmetry ✓
- At most four bends per edge ✓
- Spacing constraints ✓
- Edge Bundling ✓

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Our Approach – Theoretical Results

Quality Guarantees

- Vertical alignment of predecessors ✓
- Local symmetry ✓
- At most four bends per edge ✓
- Spacing constraints ✓
- Edge Bundling ✓

Theorem 5
There exists an algorithm that computes a column-based layout in $O(m^2(n+b))$ time.

b = #columns

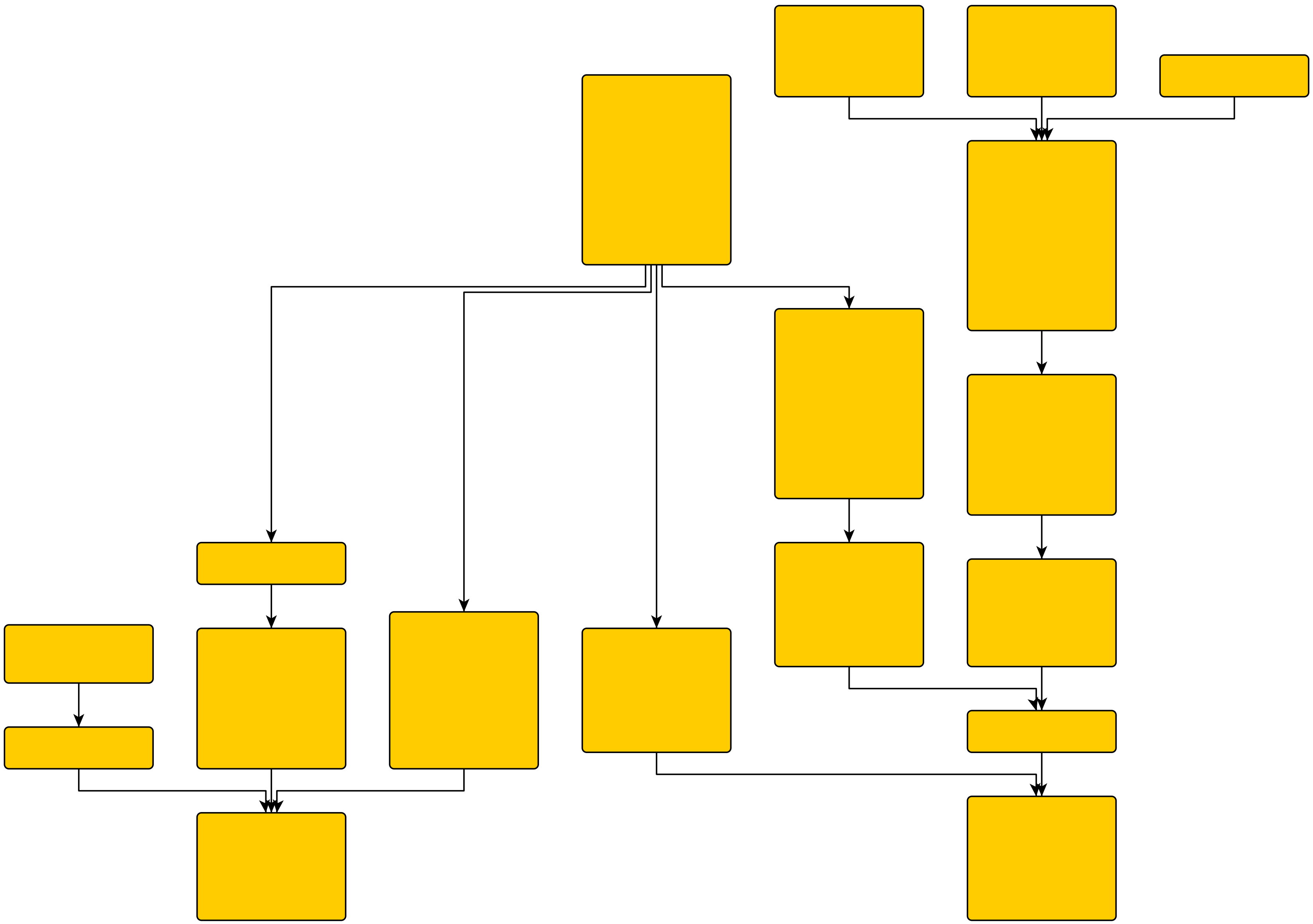
Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Experimental Evaluation

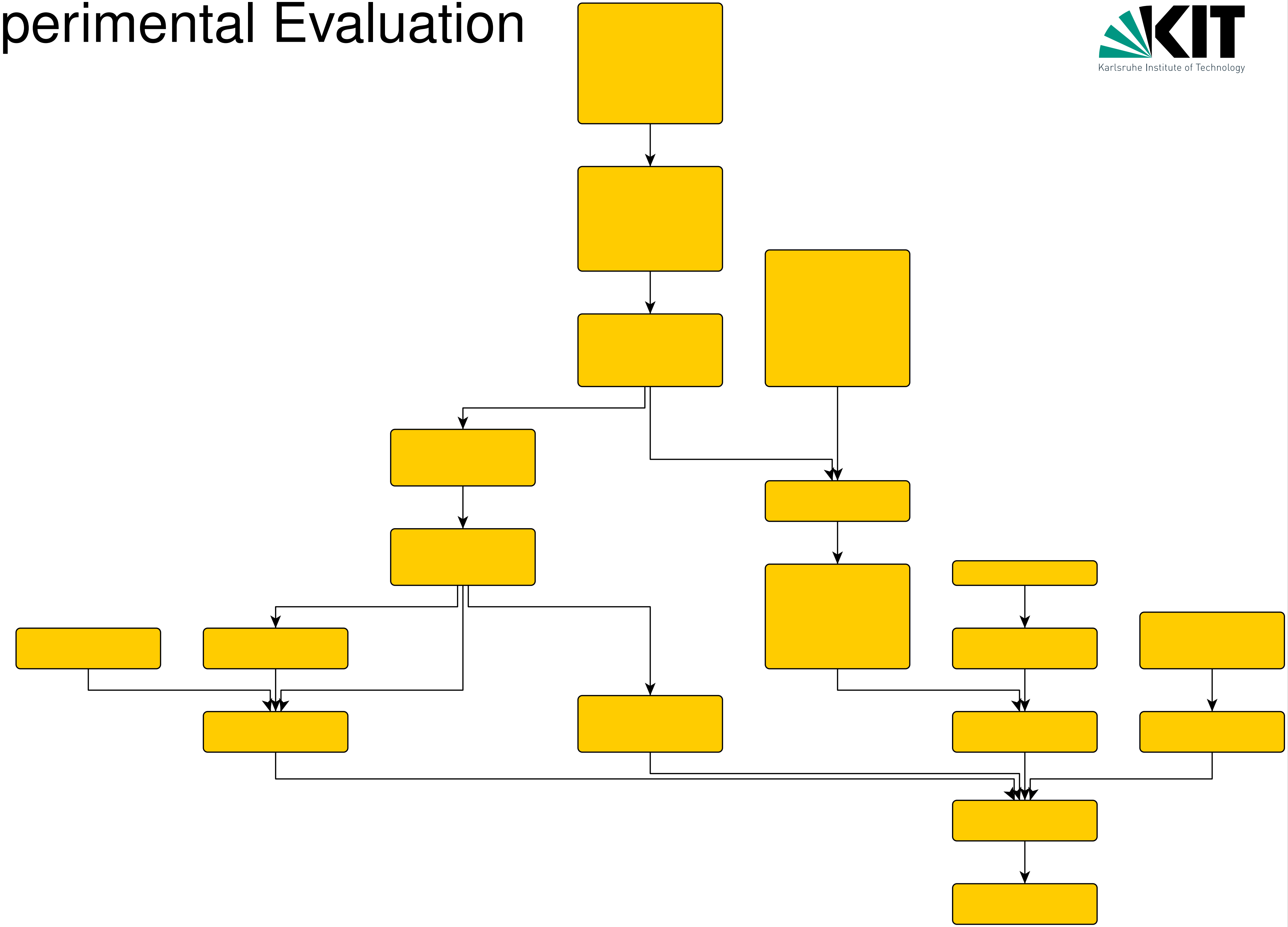
Experimental evaluation with 51 argument maps

- avg. number of nodes $\leq$ 40 nodes
- max. degree $\leq$ 6
- 93% planar und acyclic graphs

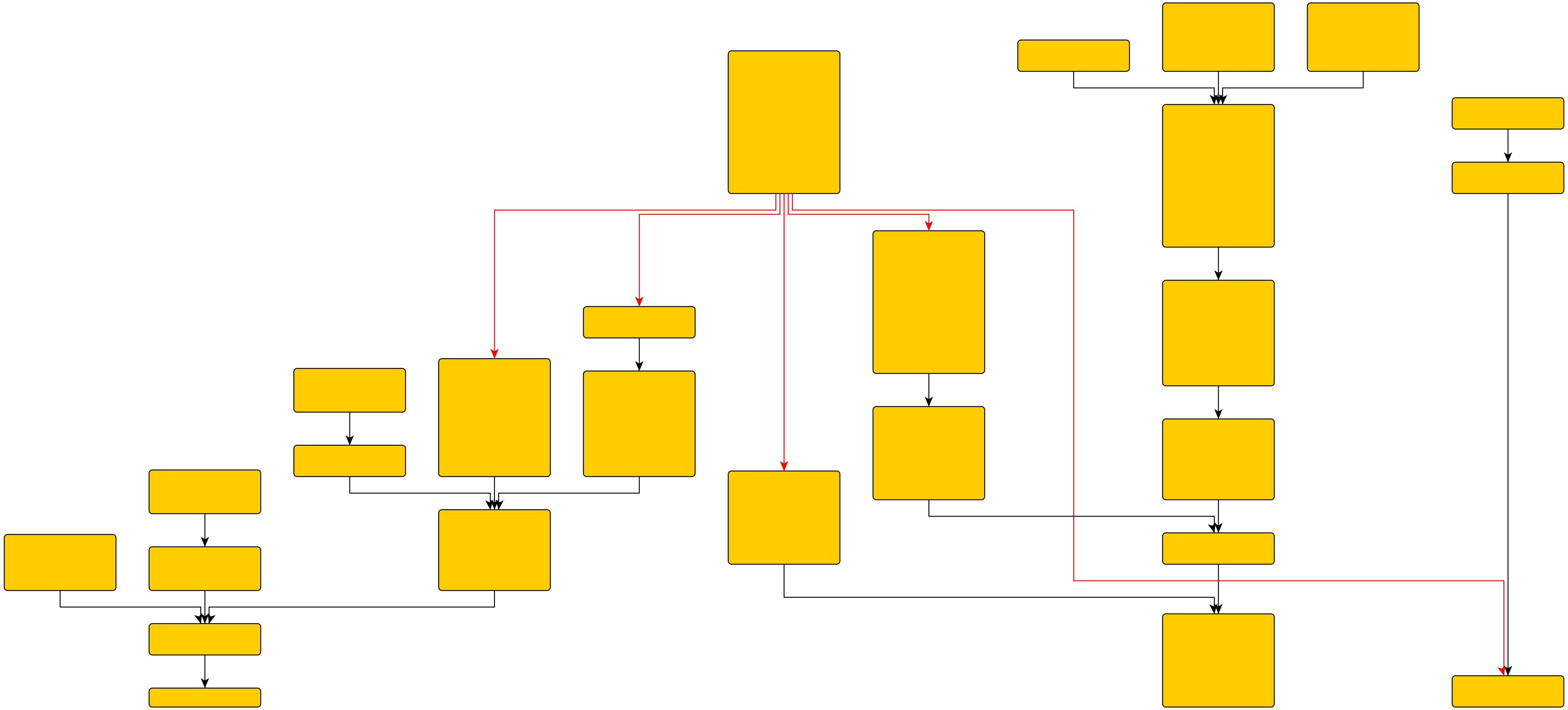
Experimental Evaluation



Experimental Evaluation

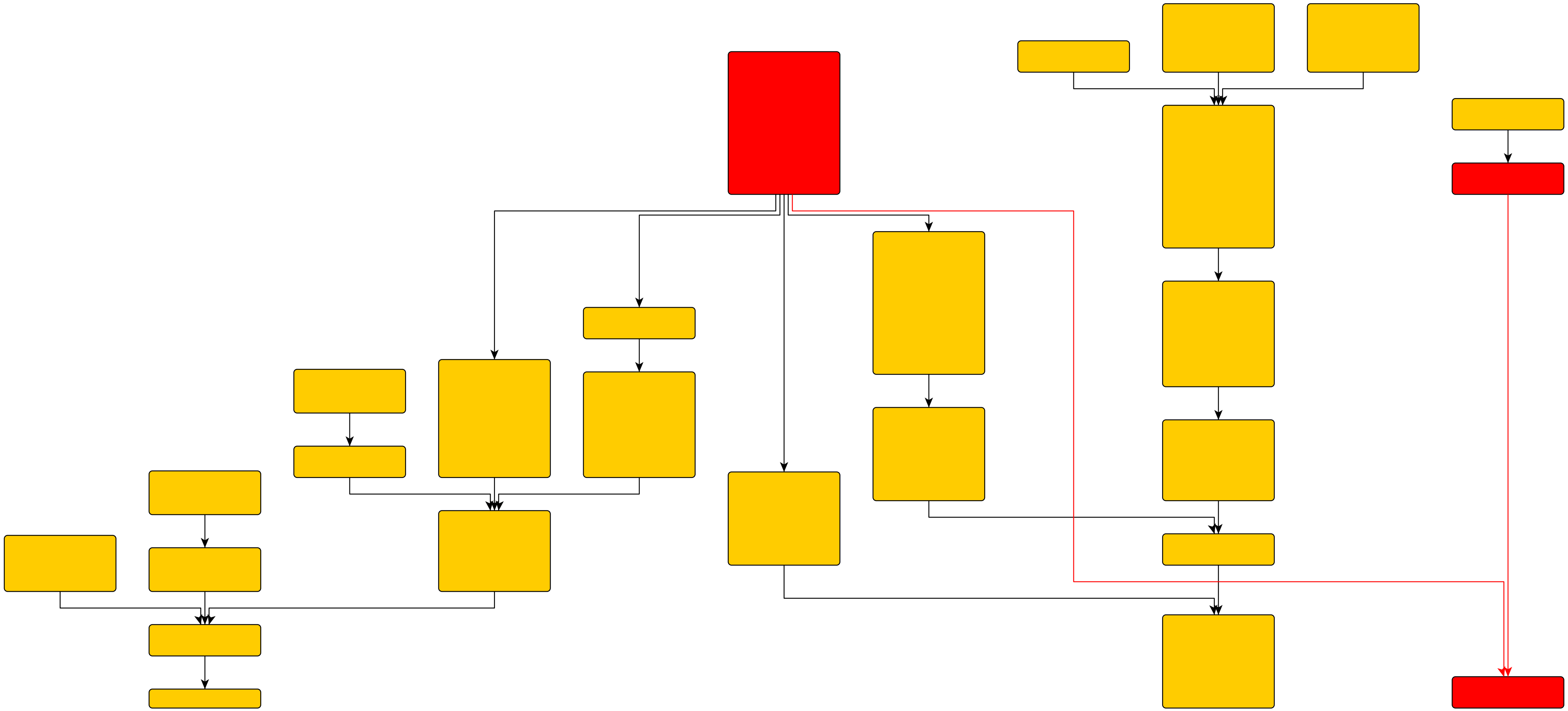


Experimental Evaluation



Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Experimental Evaluation



Experimental Evaluation

Experimental evaluation with 51 argument maps

- avg. number of nodes $\leq$ 40 nodes
- max. degree $\leq$ 6
- 93% planar und acyclic graphs

Statistics:

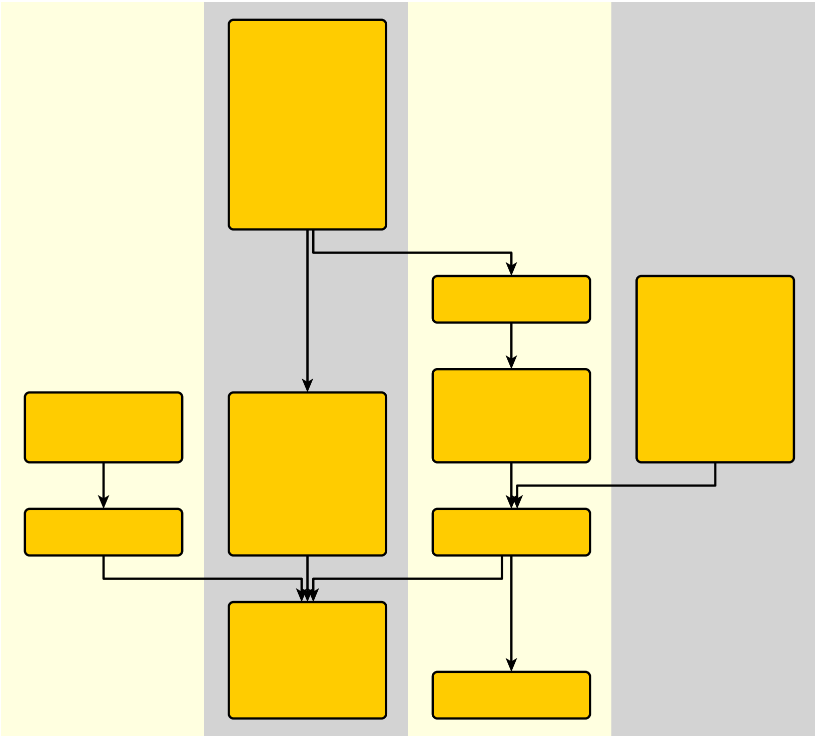
- avg. number of bends per edge: 1.06
- max. runtime: 25s [134 nodes, 158 edges]
- avg. runtime: 0.77s

the topology step requires [on avg.] 99.6% of the runtime

Conclusion

Topology-Shape-Metrics [Tamassia ’87]

topology	compute embedding of the graph
shape	assign bends to edges
metric	determine edge lengths & node positions



Column-based Graph Layouts

Institute of Theoretical Informatics
Prof. Dr. Dorothea Wagner

Conclusion

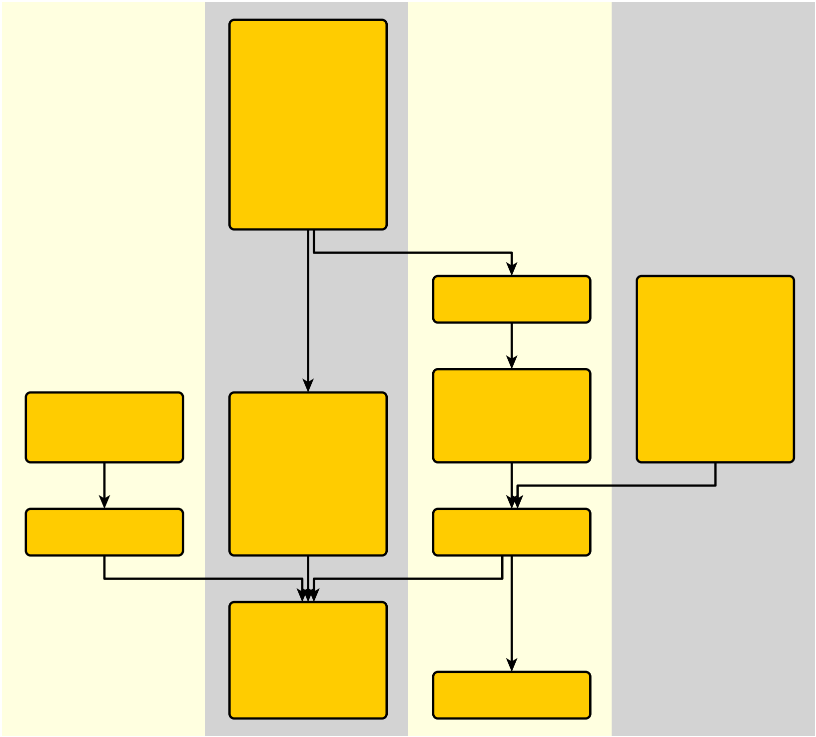
Topology-Shape-Metrics [Tamassia ’87]

topology compute emb… of the graph

shape ass… to edges

metric …etermine edge lengths & node positions

relaxation of topology



Column-based Graph Layouts

Conclusion

Topology-Shape-Metrics [Tamassia '87]

relaxation of topology

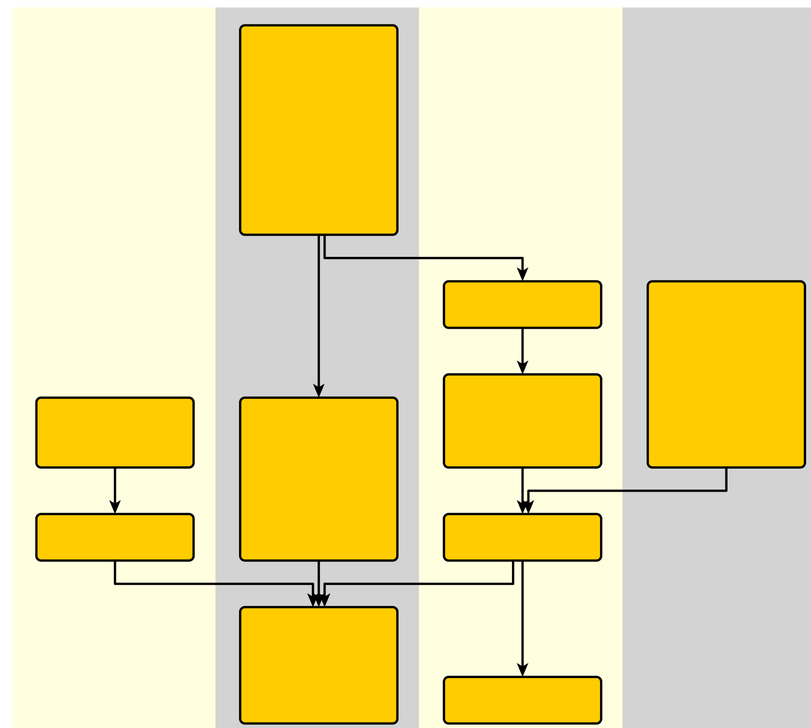
topology compute emb… of the graph

shape ass… to edges

metric …termine edge lengths & node positions

Our Approach

- clean and structured layouts
- reasonable running time



Column-based Graph Layouts

Conclusion

Topology-Shape-Metrics [Tamassia ’87]

topology compute emb … of the graph

shape ass … to edges

metric … termine edge lengths & node positions

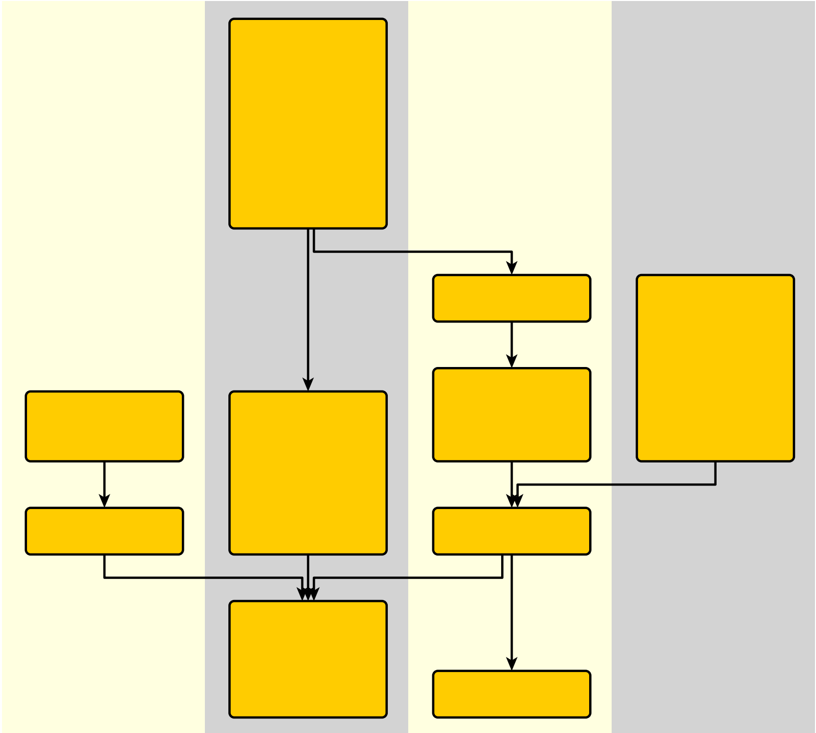
relaxation of topology

Our Approach

- clean and structured layouts
- reasonable running time

Future Work

- incremental layouts



Column-based Graph Layouts

Conclusion

Karlsruhe Institute of Technology

Topology-Shape-Metrics [Tamassia '87]

topology compute emb… of the graph

shape assi… to edges

metric …etermine edge lengths & node positions

relaxation of topology

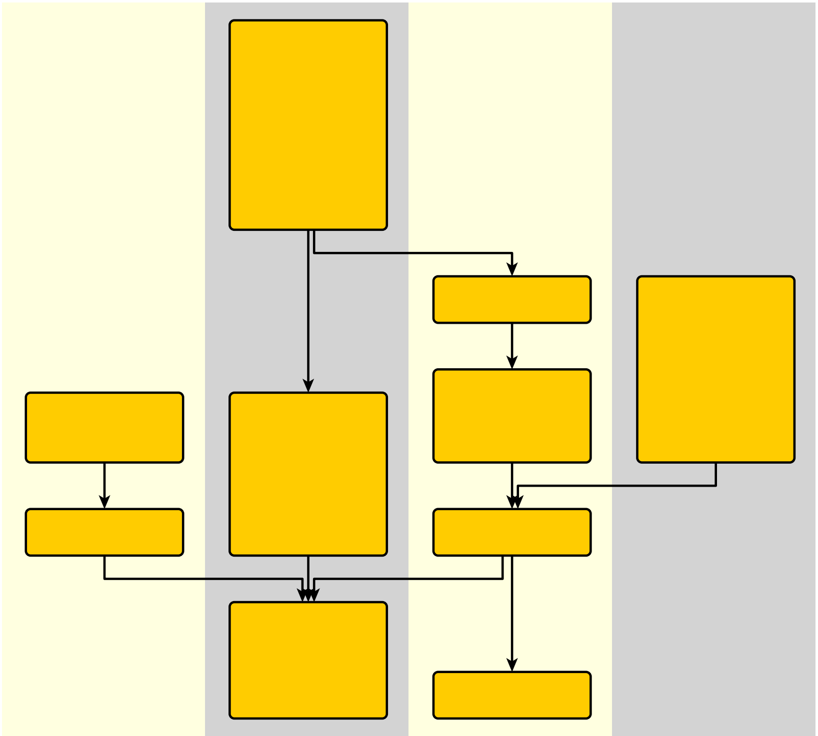
Our Approach

- clean and structured layouts
- reasonable running time

Future Work

- incremental layouts

Thank you!



Column-based Graph Layouts