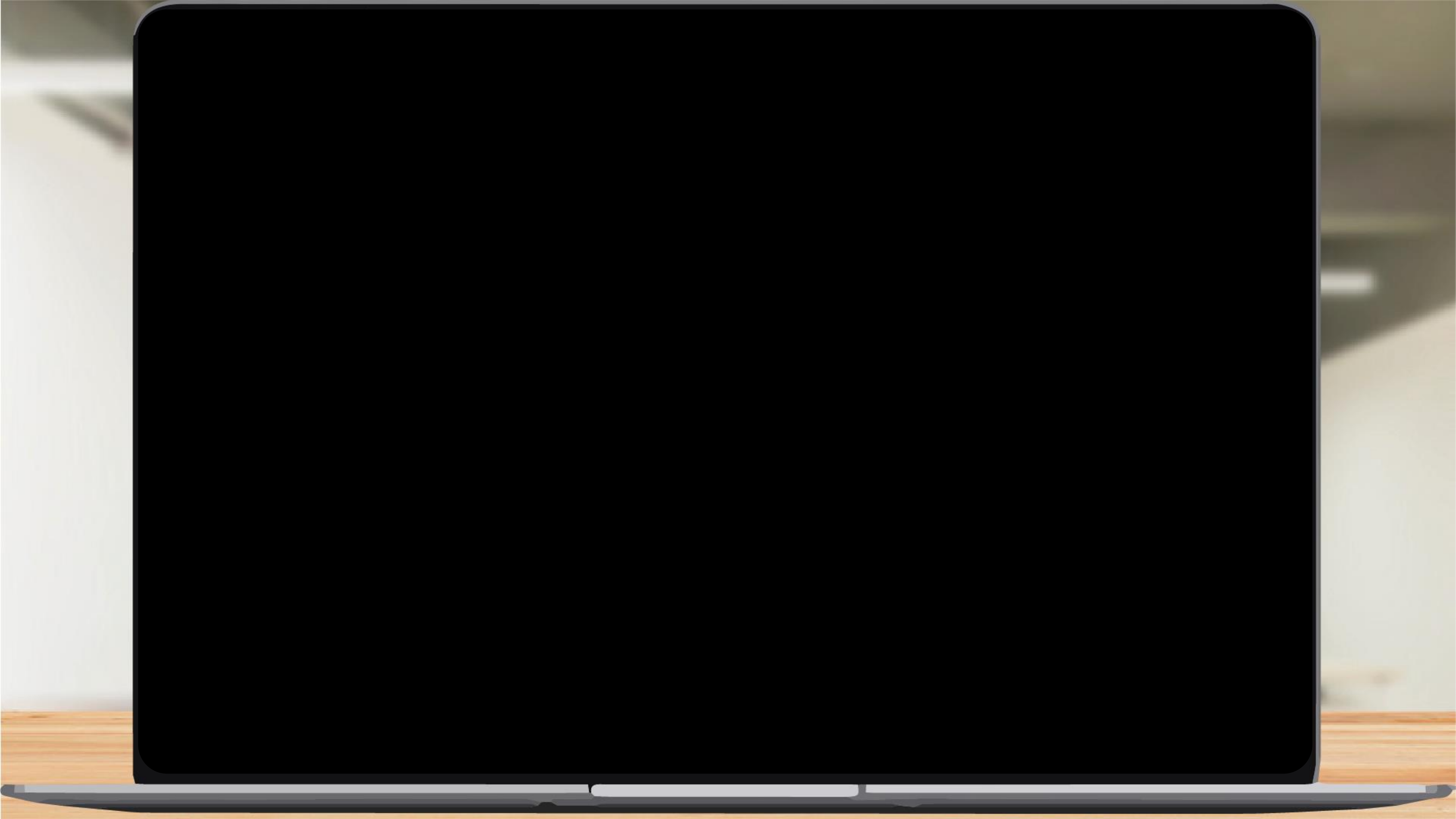
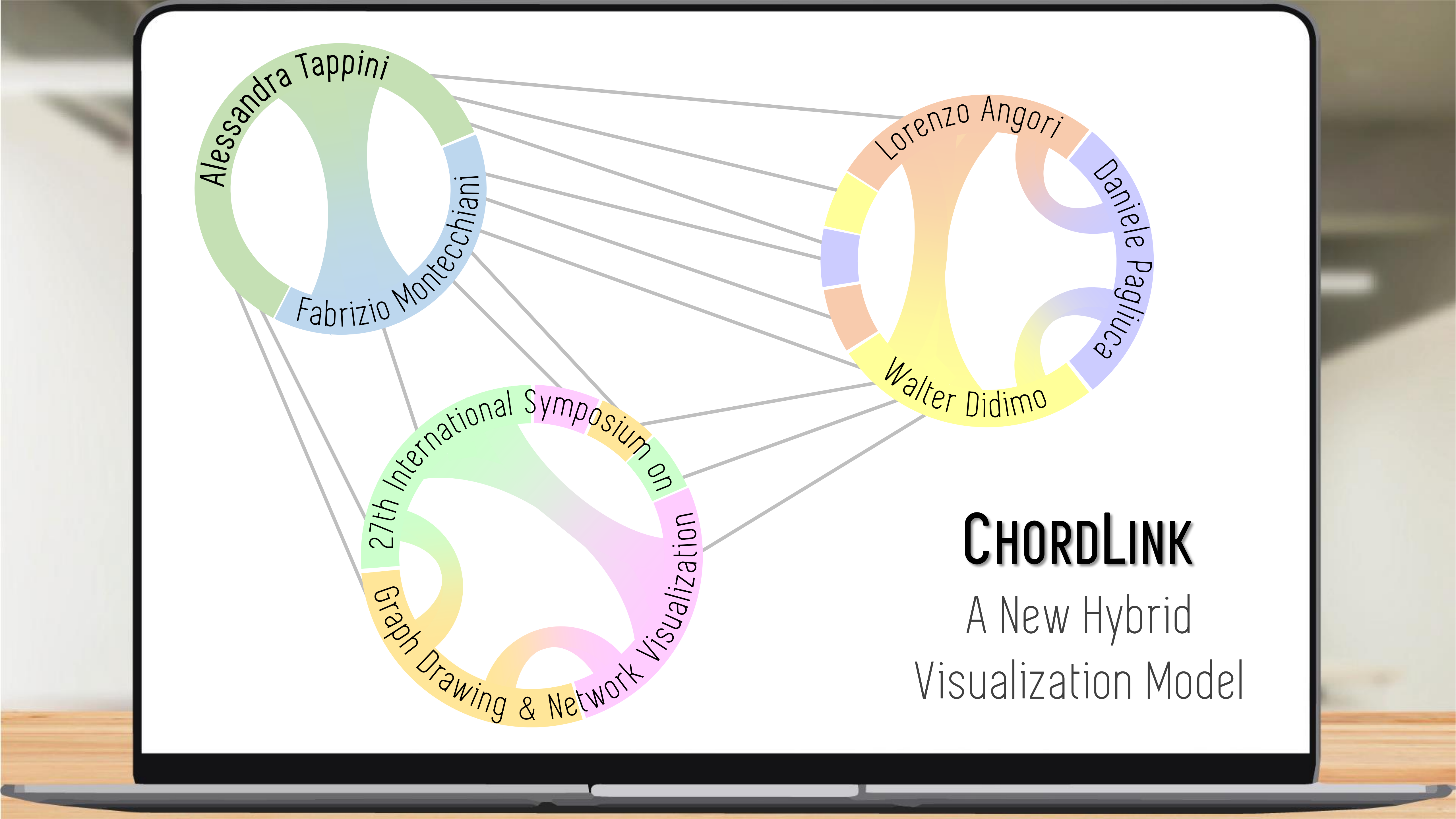



Alessandra Tappini
Fabrizio Montecchiani
Lorenzo Angori
Daniele Pagliuca
Walter Didimo
27th International Symposium on Graph Drawing & Network Visualization
ChordLink
A New Hybrid Visualization Model

Hybrid Visualizations of Graphs

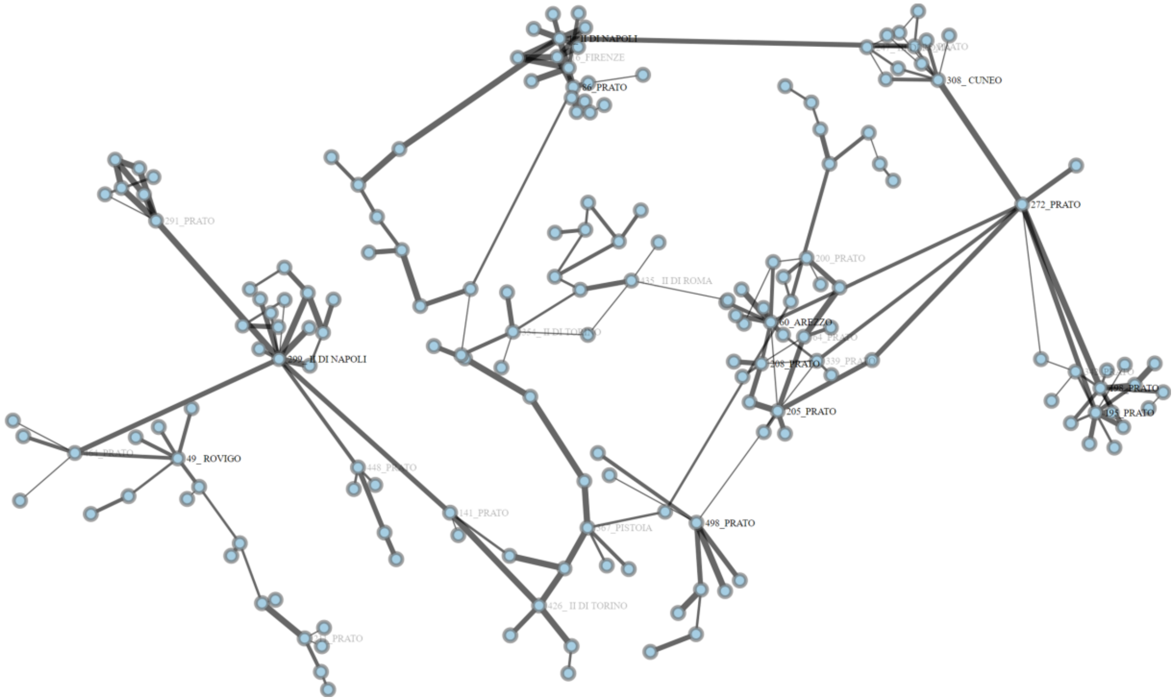
Real-world networks are **globally sparse** but **locally dense**
(Social networks, biological networks, financial networks)

- Communities (**clusters**) contain highly connected sets of nodes
- Clusters are loosely connected to each other

Visual exploration tasks:

(T1) Get an overview of the network

(T2) Analyze the communities in detail



Hybrid Visualizations of Graphs

Problem: How to support both global and local tasks?

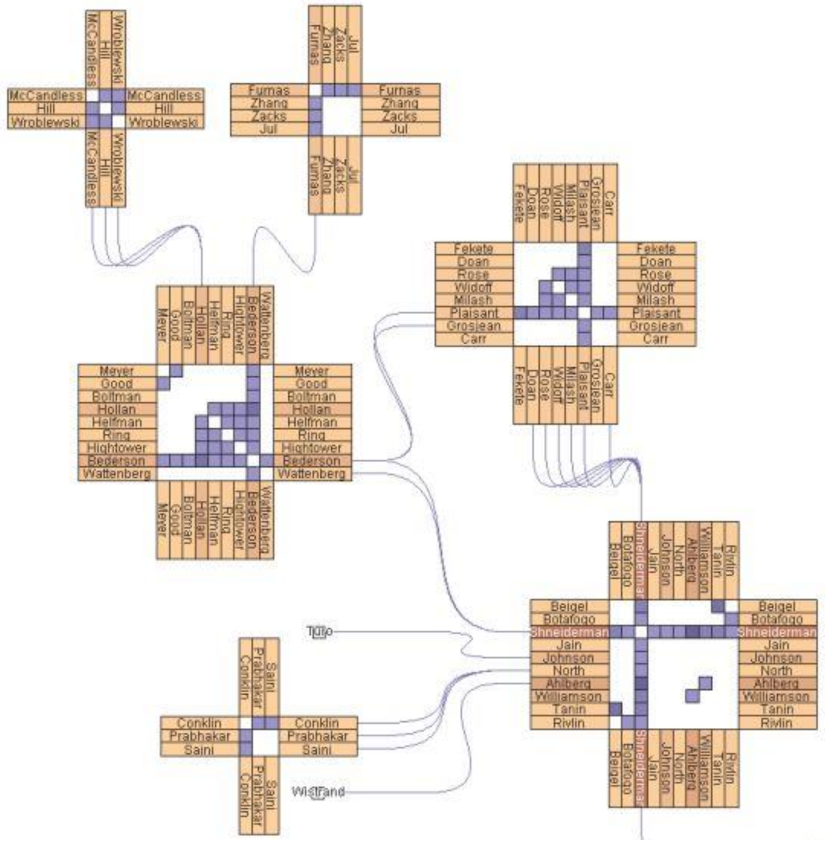
Idea: Combine different drawing styles ⟶ **Hybrid visualizations**

NodeTrix model [Henry et al., 2007]

- Global structure ⟶ Node-link paradigm
- Clusters ⟶ Adjacency matrices
- Interaction ⟶ The user can select the portions to be represented as adjacency matrices

Drawbacks

- Users less familiar with matrices
- Paths harder to follow



Contribution

- Design a new hybrid visualization model
- Integrate this model into an interactive visual analytics system

REQUIREMENTS

(R1) Support the drawing stability during the user interaction

- Preserve the geometry of nodes and edges
- Maintain the user's mental map

(R2) Use drawing styles that are intuitive for non-expert users

- Not so different from the node-link style

The ChordLink Model

ChordLink

- Global structure ⟶ Node-link paradigm
- Clusters ⟶ Chord diagrams

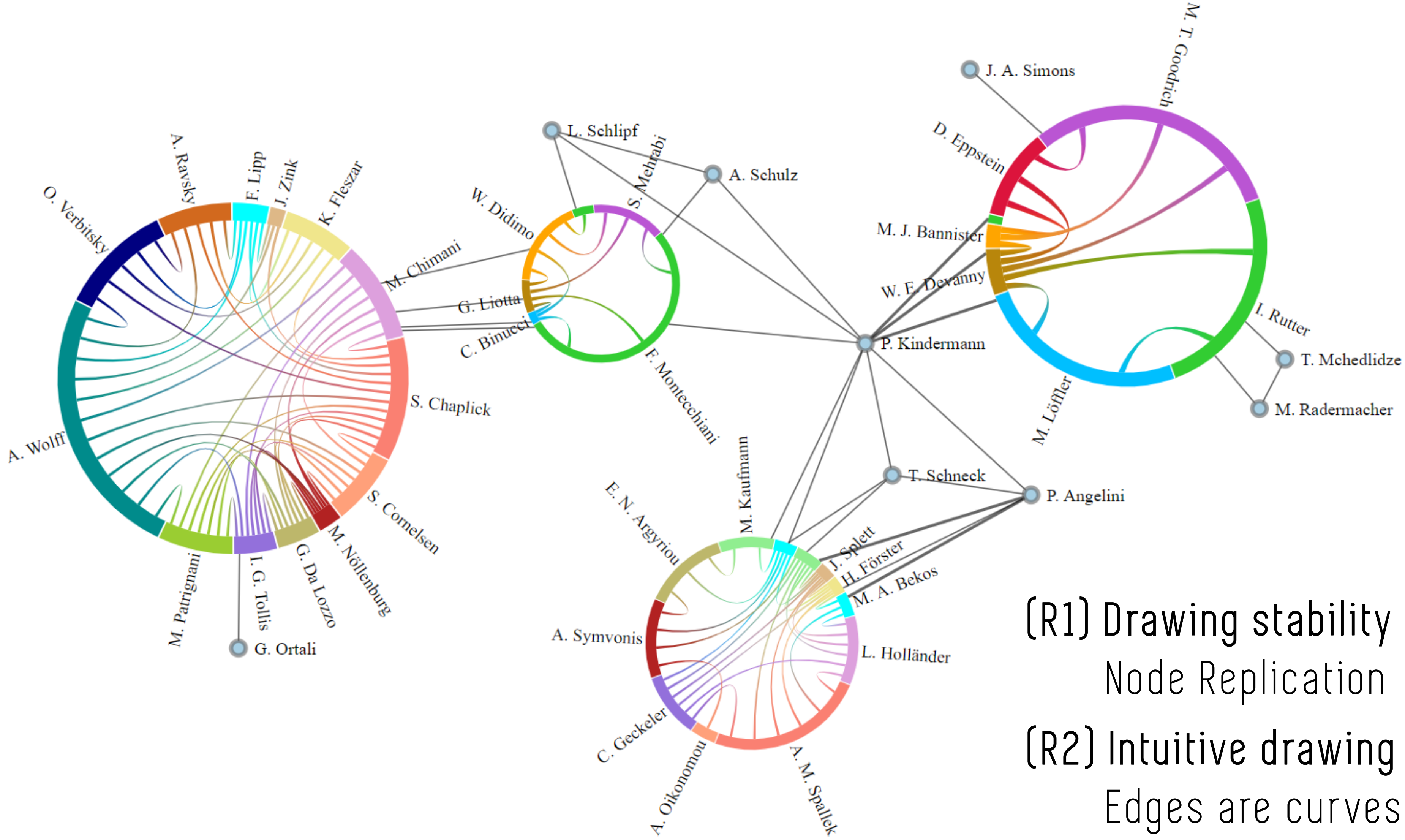
Chord diagrams

- Extension of circular drawings
- Nodes are circular arcs instead of points
- Edges are curves (**chords**)

The user can

- Define clusters interactively
- Analyze clusters while the system ensures the drawing stability

The ChordLink Model



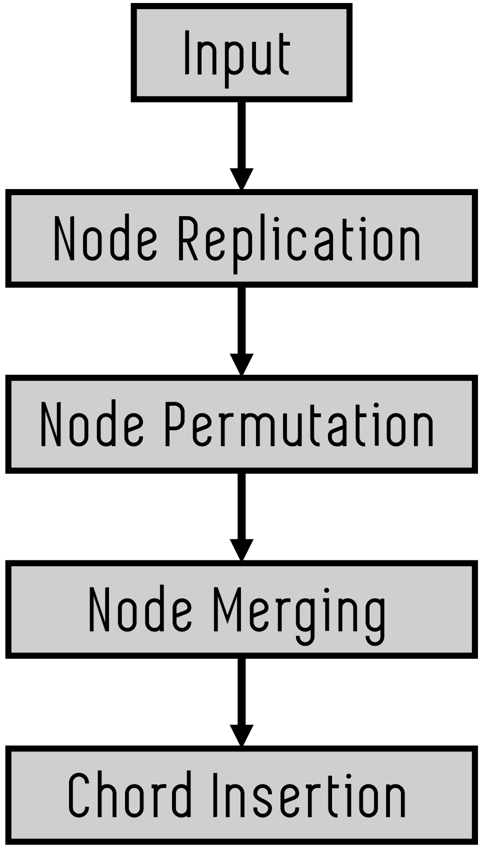
(R1) Drawing stability

Node Replication

(R2) Intuitive drawing styles

Edges are curves

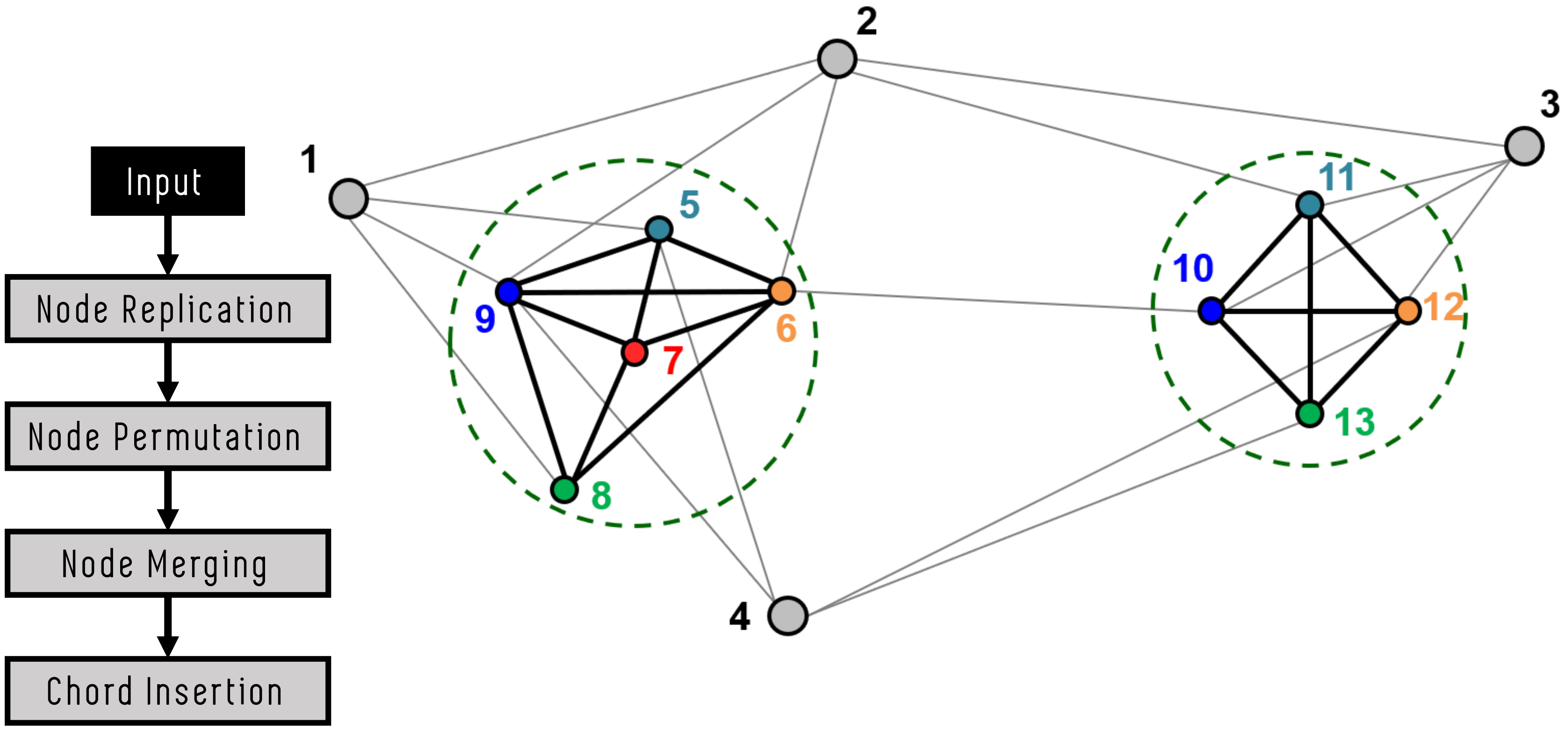
General Strategy



General Strategy

Input: Node-link straight-line drawing of a clustered graph

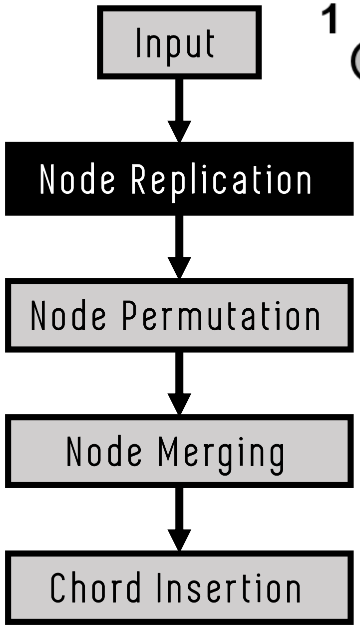
Assume that the nodes of each cluster lie in a circular (restricted) region

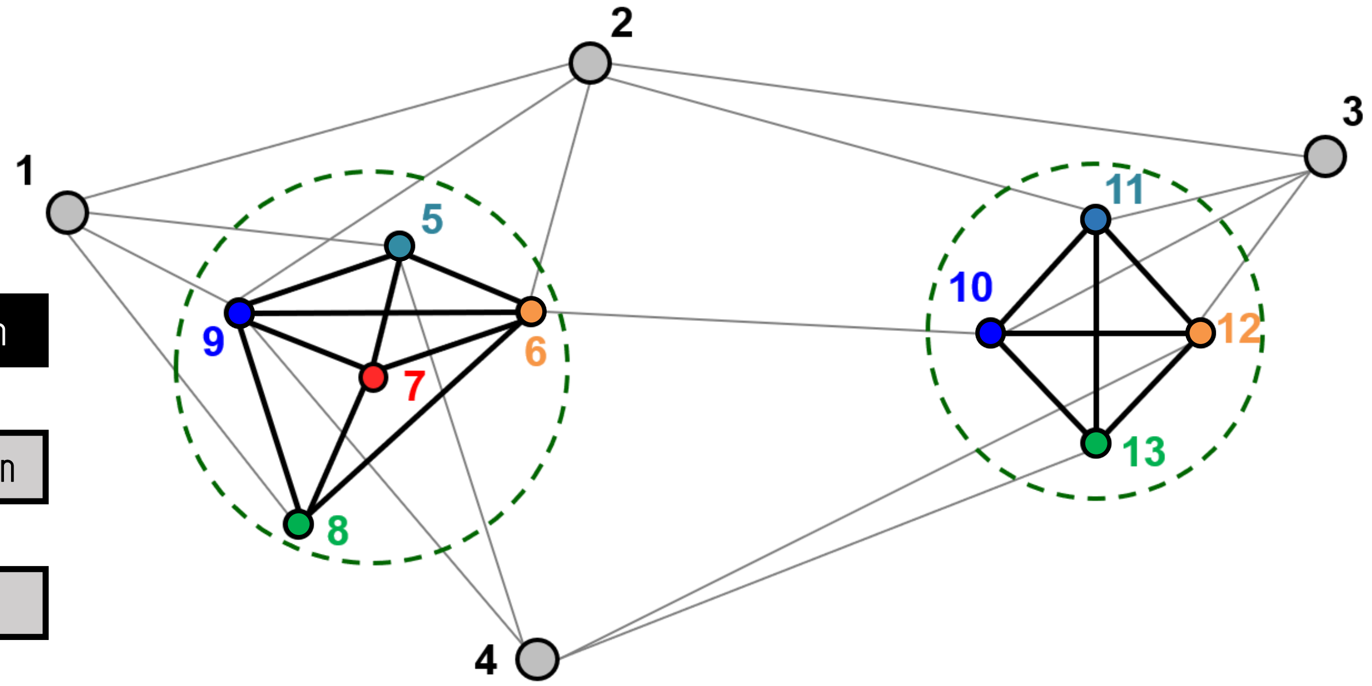


General Strategy

Node Replication: Create (multiple) copies of the nodes of a cluster

Project the nodes on the boundary of their region, following the circular order induced by the external edges. Remove the elements inside the region.

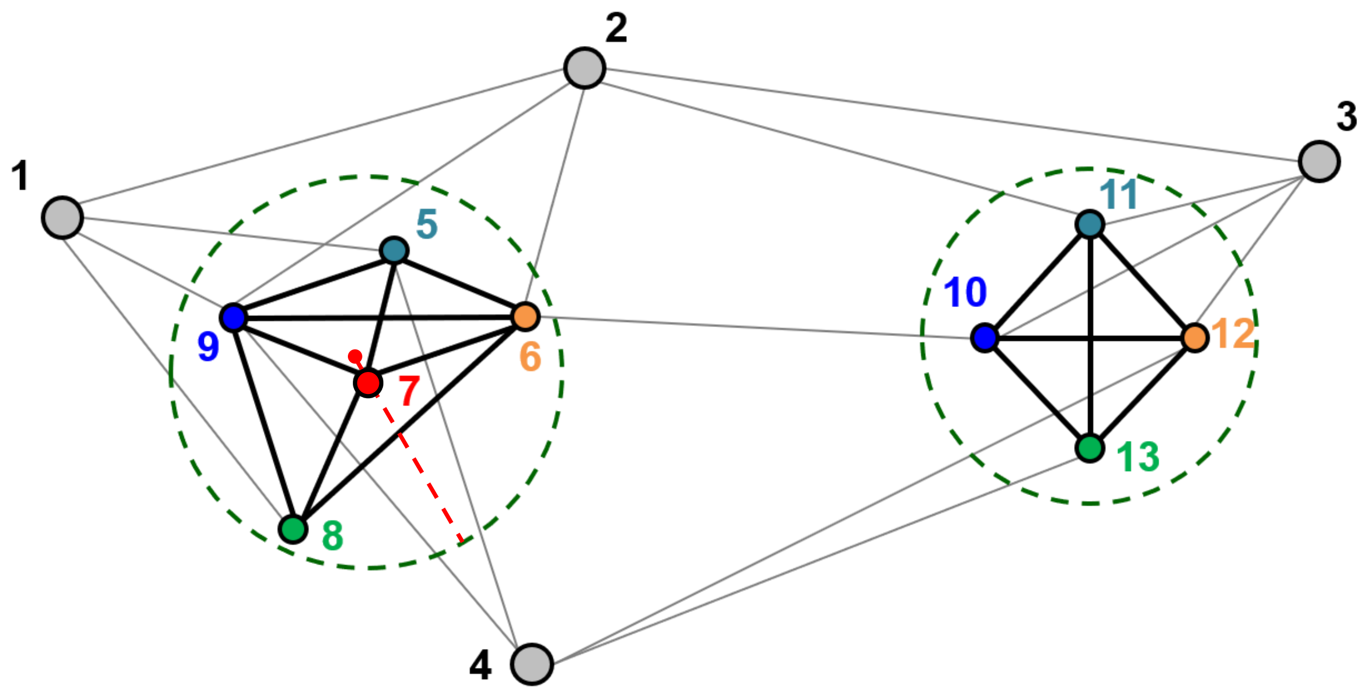




General Strategy

Node Replication: Create (multiple) copies of the nodes of a cluster

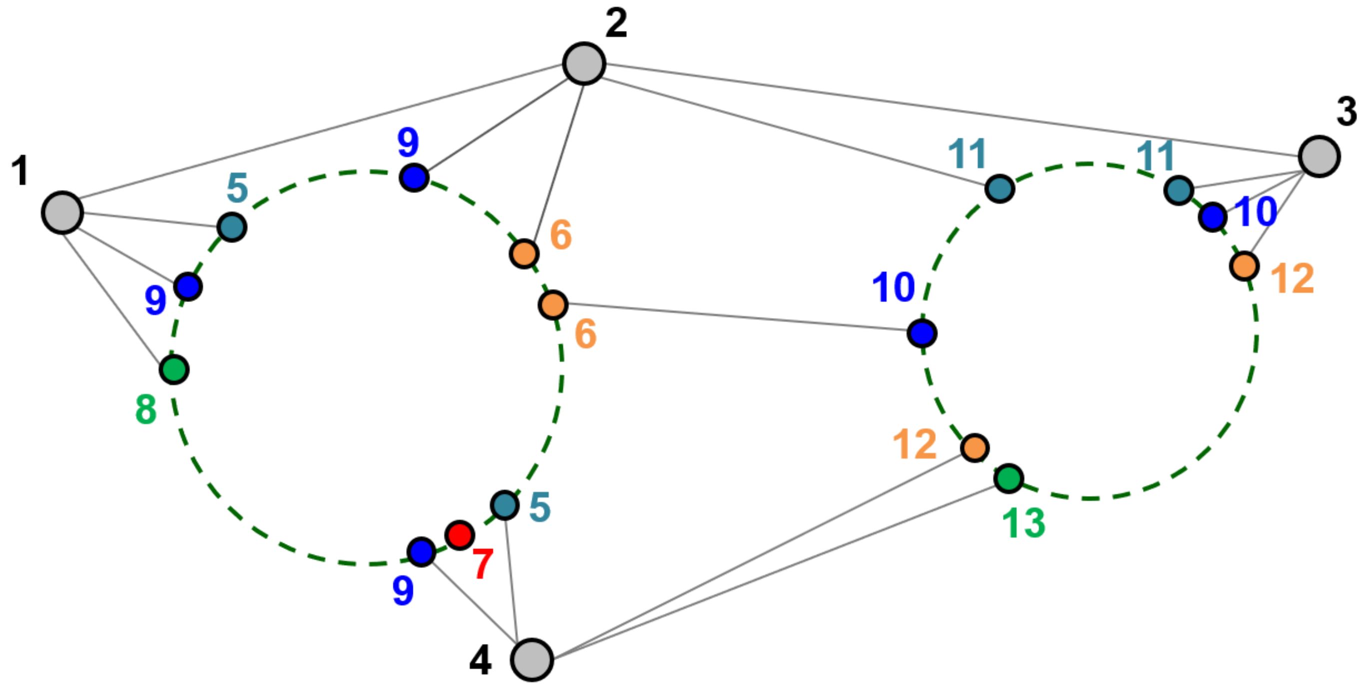
Project the nodes on the boundary of their region, following the circular order induced by the external edges. Remove the elements inside the region.



General Strategy

Node Replication: Create (multiple) copies of the nodes of a cluster

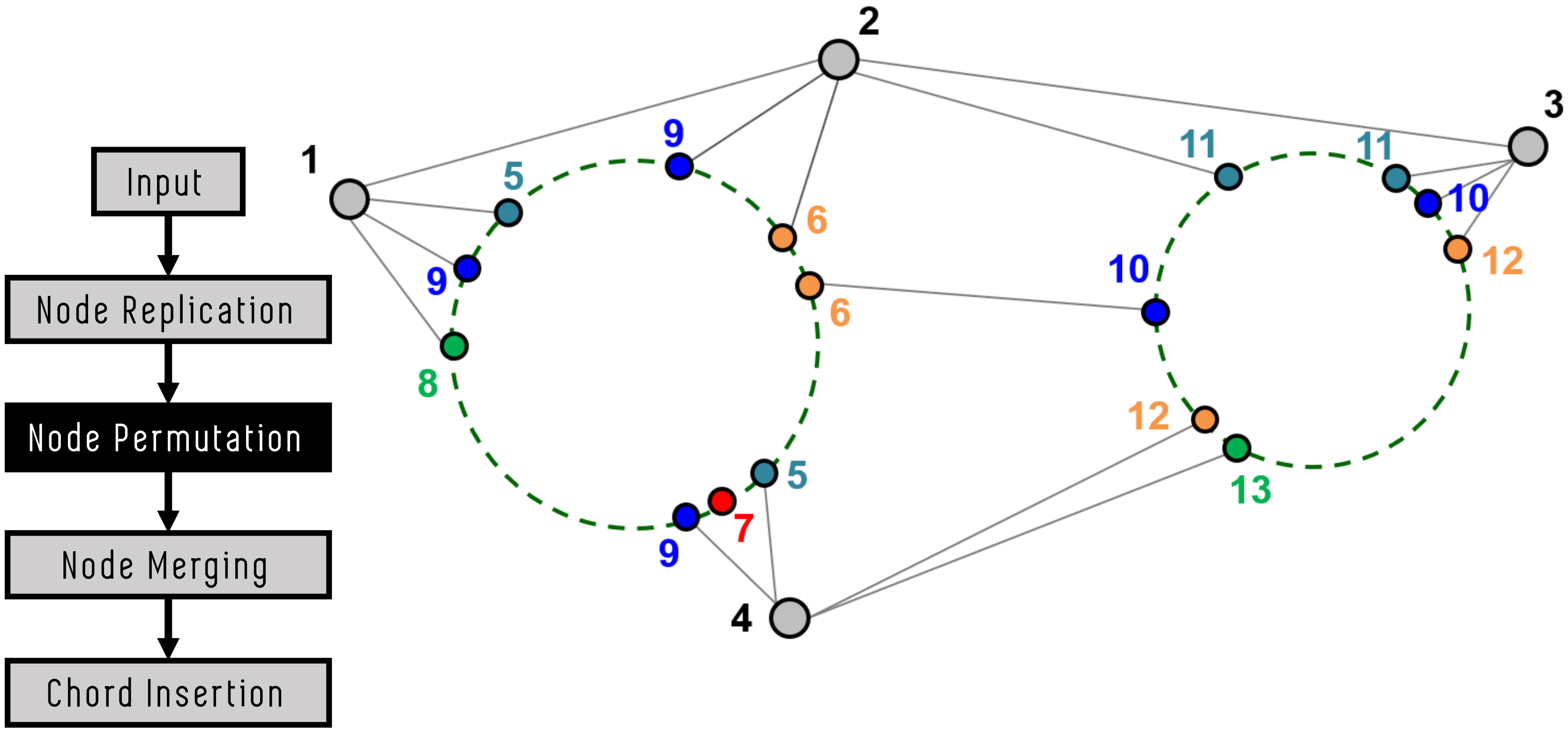
Project the nodes on the boundary of their region, following the circular order induced by the external edges. Remove the elements inside the region.



General Strategy

Node Permutation: Permute the copies of the nodes (only if they are adjacent to the same external node)

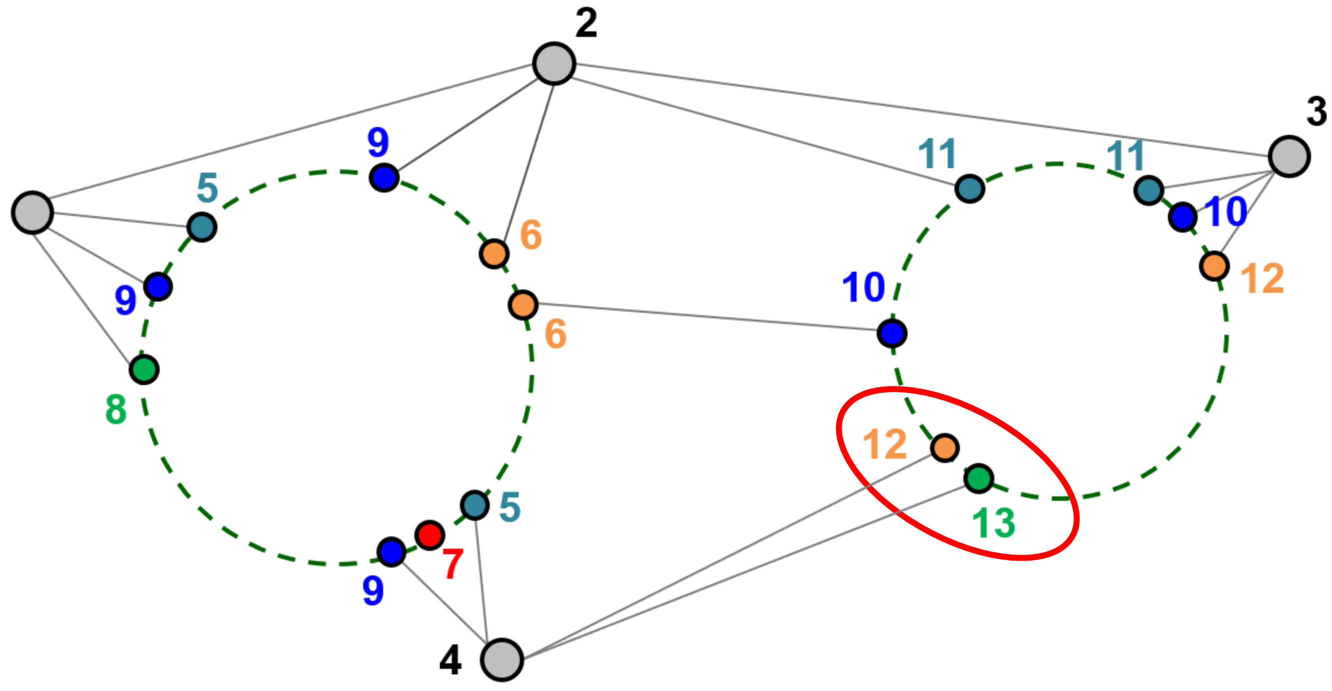
<u>Optimization Goal 1</u>: Minimize non-consecutive copies of the same node



General Strategy

Node Permutation: Permute the copies of the nodes (only if they are adjacent to the same external node)

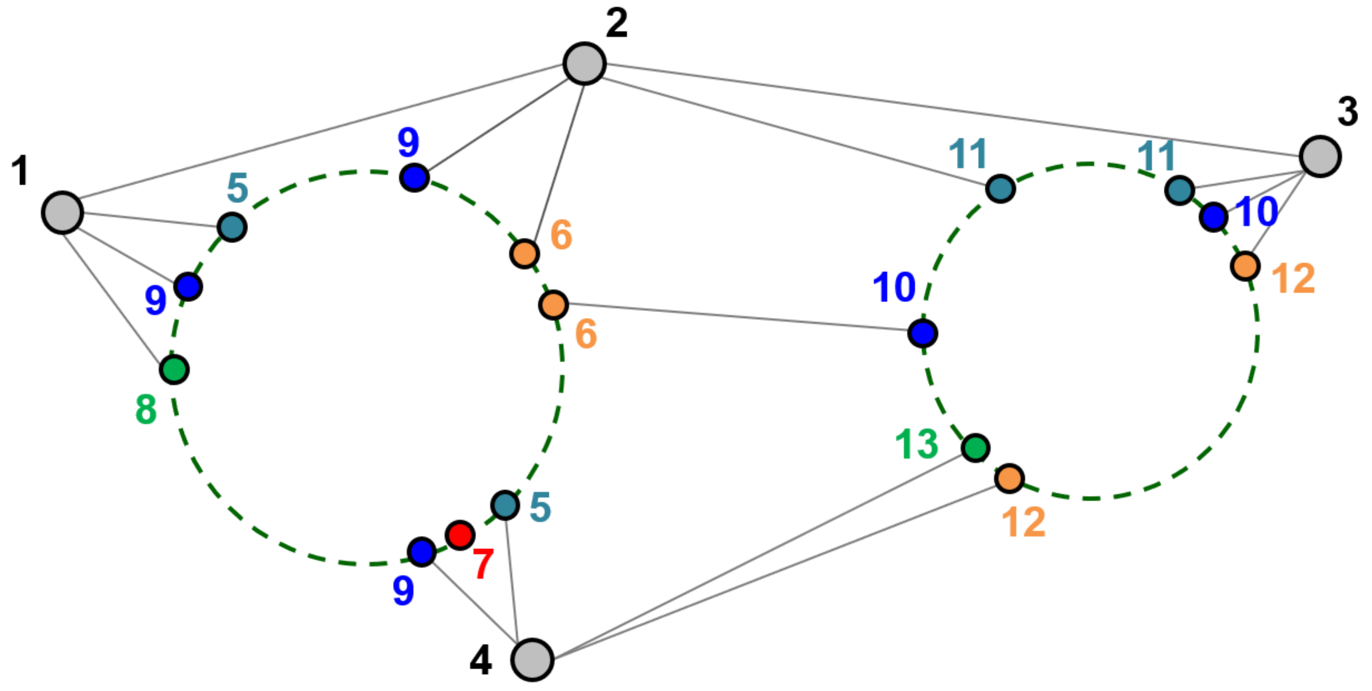
Optimization Goal 1: Minimize non-consecutive copies of the same node



General Strategy

Node Permutation: Permute the copies of the nodes (only if they are adjacent to the same external node)

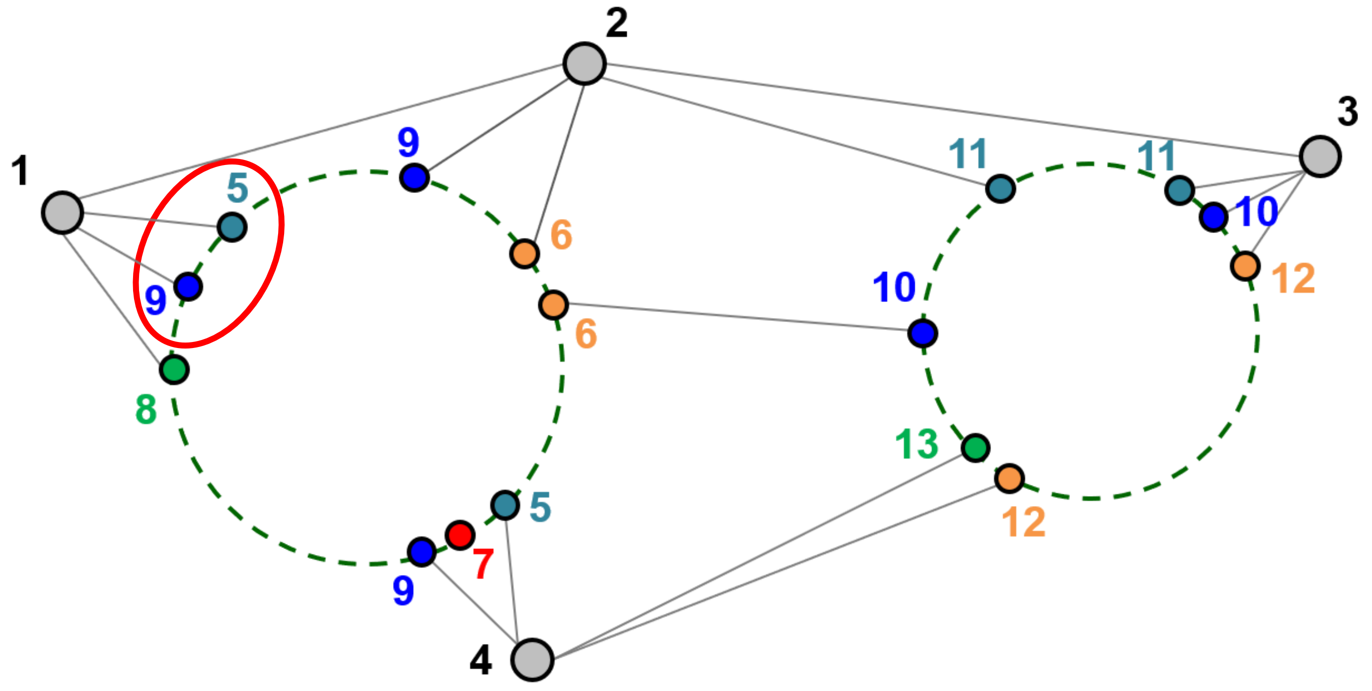
Optimization Goal 1: Minimize non-consecutive copies of the same node



General Strategy

Node Permutation: Permute the copies of the nodes
(only if they are adjacent to the same external node)

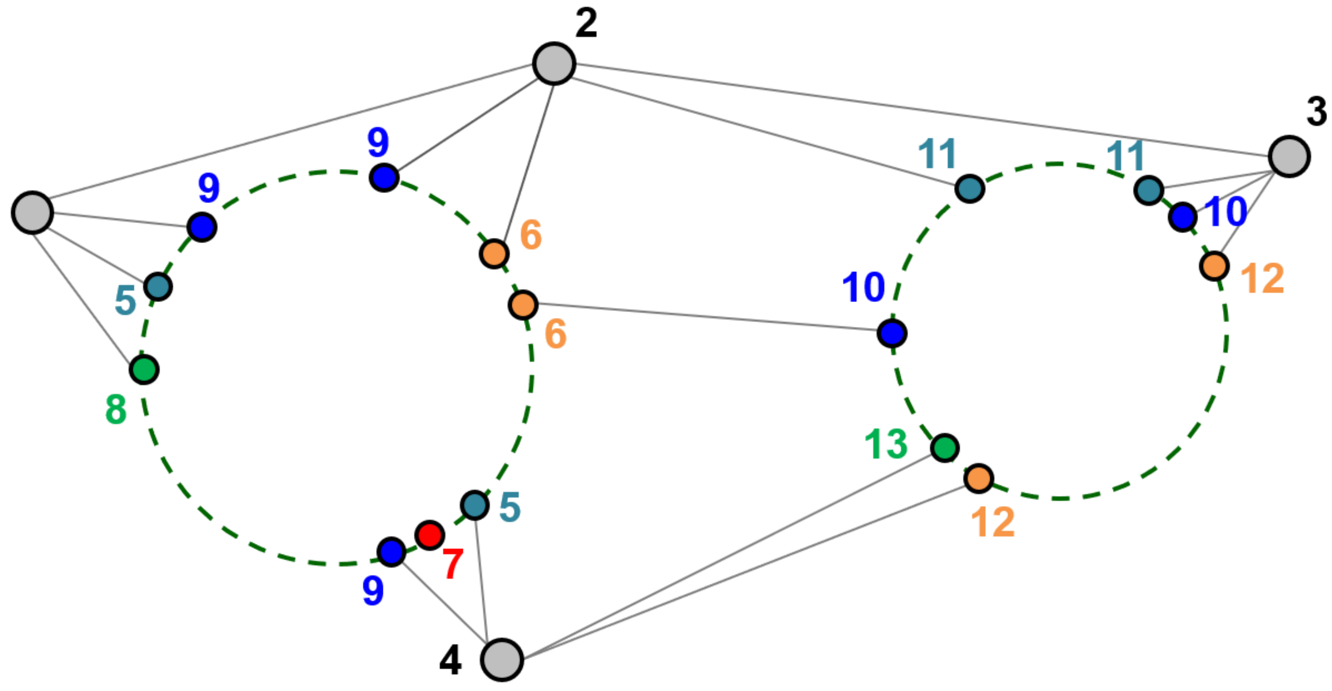
Optimization Goal 1: Minimize non-consecutive copies of the same node



General Strategy

Node Permutation: Permute the copies of the nodes (only if they are adjacent to the same external node)

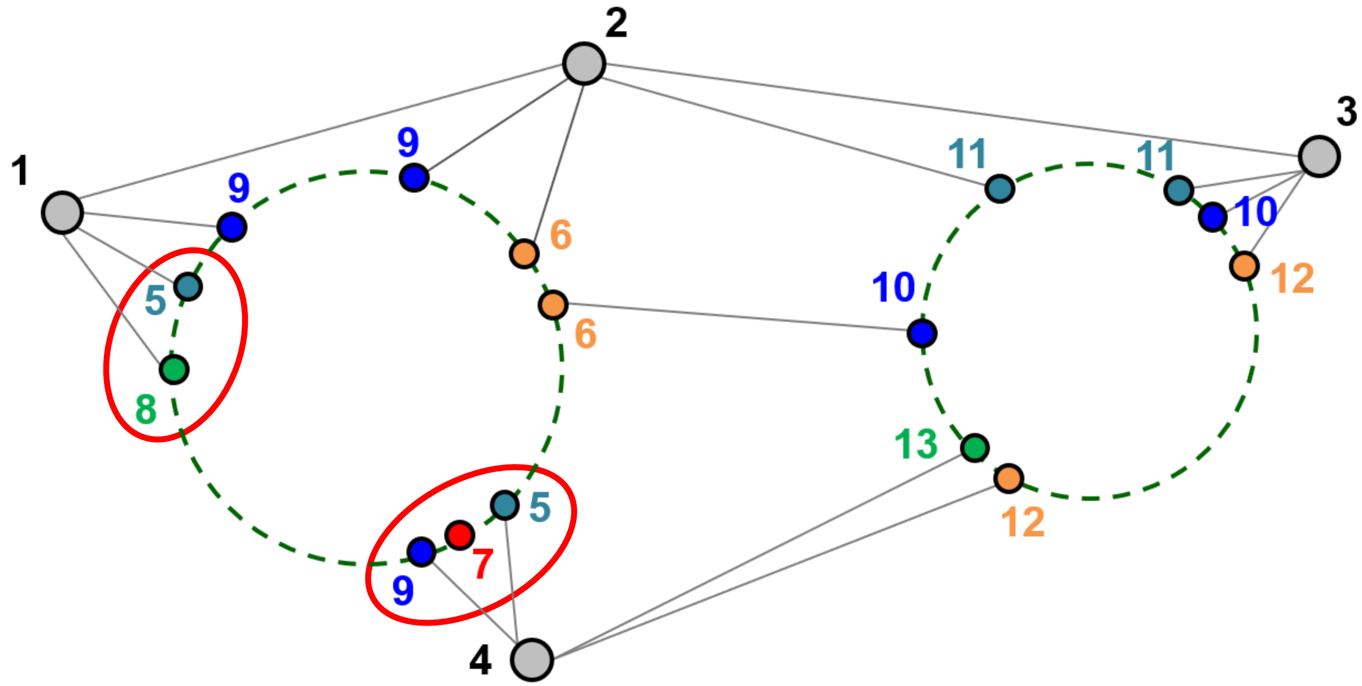
Optimization Goal 1: Minimize non-consecutive copies of the same node



General Strategy

Node Permutation: Permute the copies of the nodes (only if they are adjacent to the same external node)

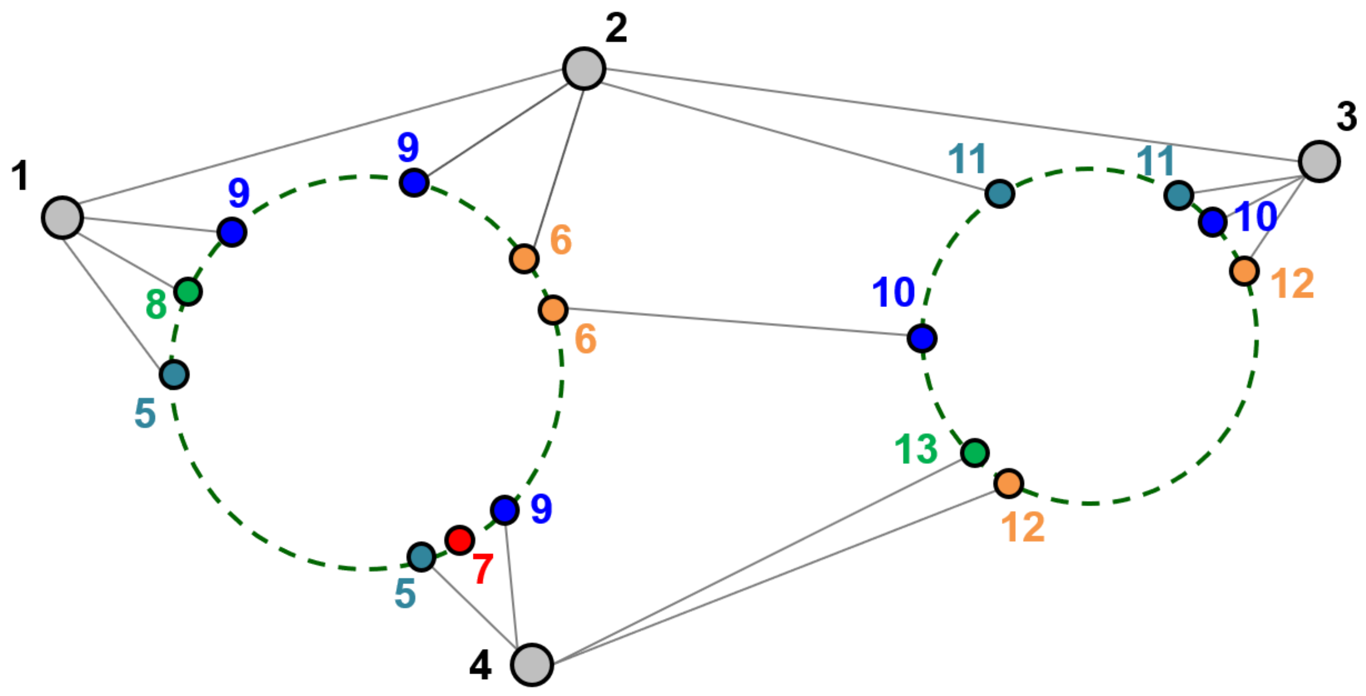
Optimization Goal 1: Minimize non-consecutive copies of the same node



General Strategy

Node Permutation: Permute the copies of the nodes (only if they are adjacent to the same external node)

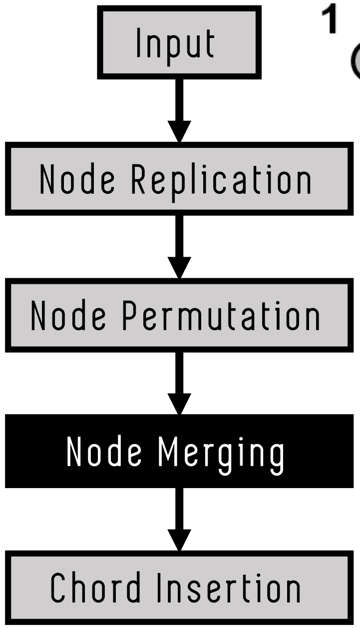
<u>Optimization Goal 1</u>: Minimize non-consecutive copies of the same node

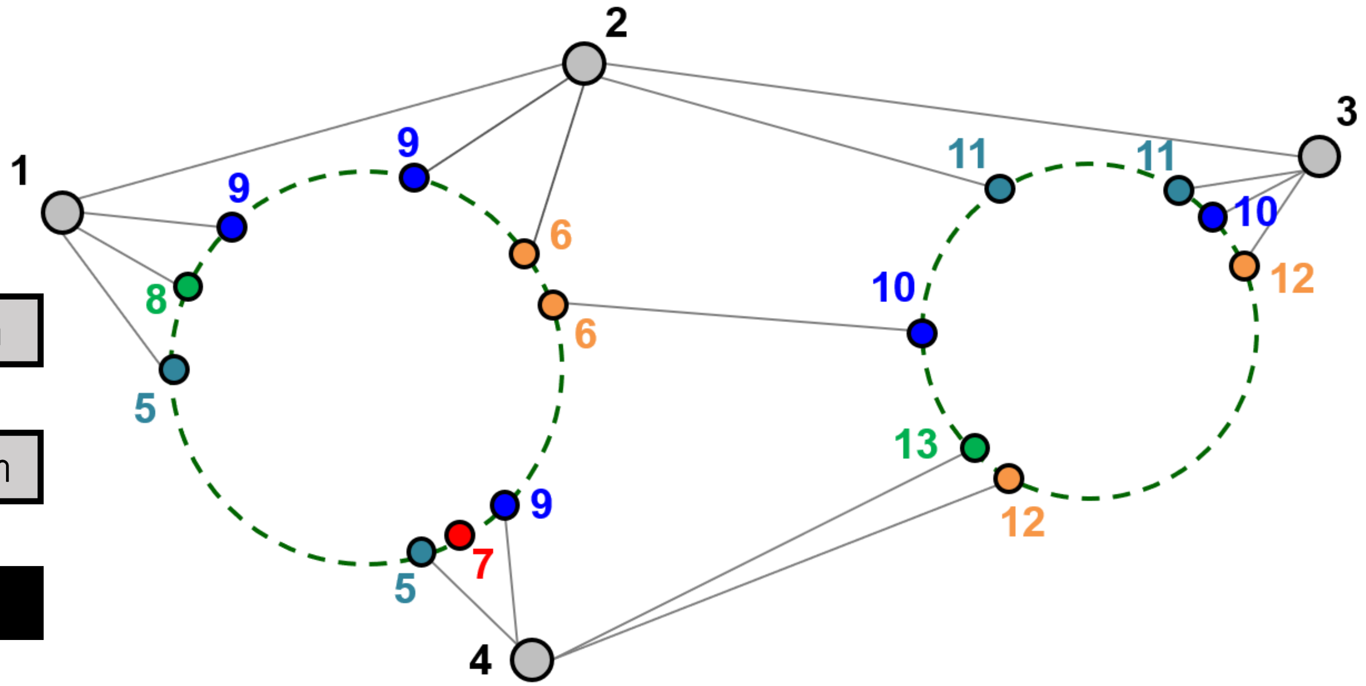


General Strategy

Node Merging: Replace nodes by circular arcs

Consecutive copies of the same node are replaced by the same arc
Arcs of the same node have the same color

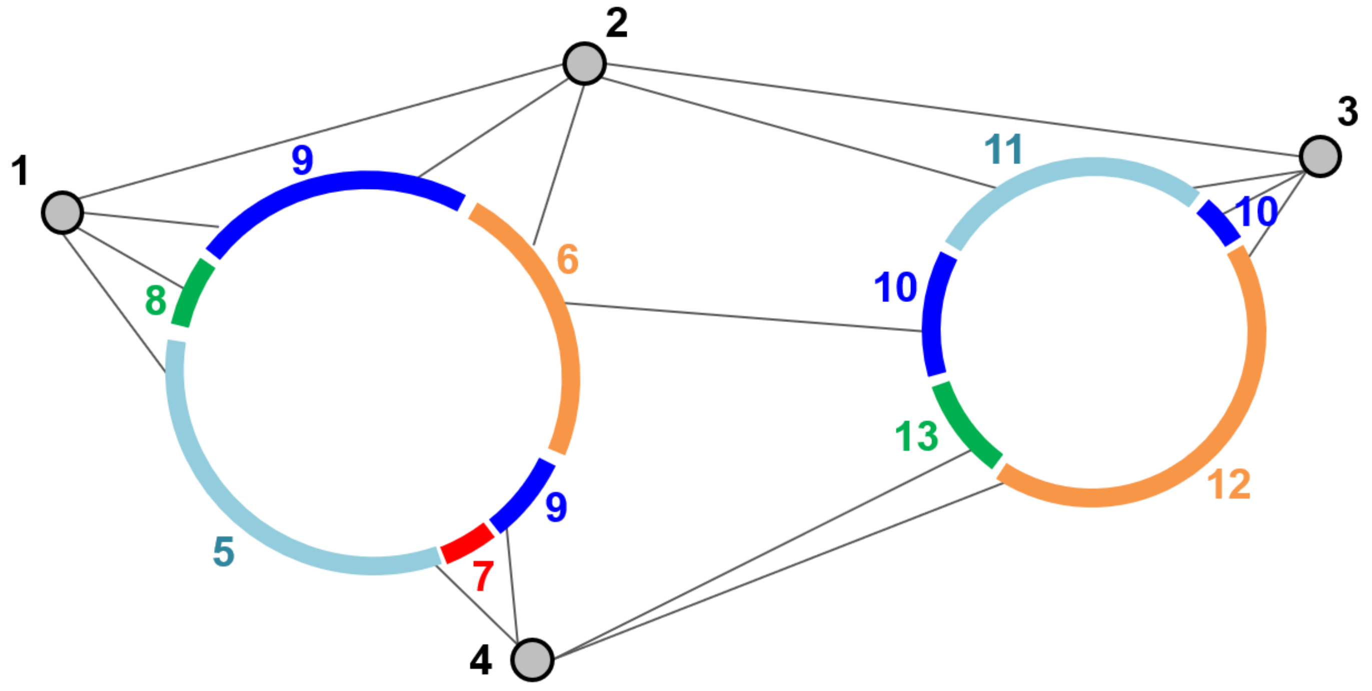




General Strategy

Node Merging: Replace nodes by circular arcs

Consecutive copies of the same node are replaced by the same arc
Arcs of the same node have the same color



General Strategy

Chord Insertion: Insert chords in the diagram

Optimization Goal 2: Minimize the number of crossings and maximize the crossing angle

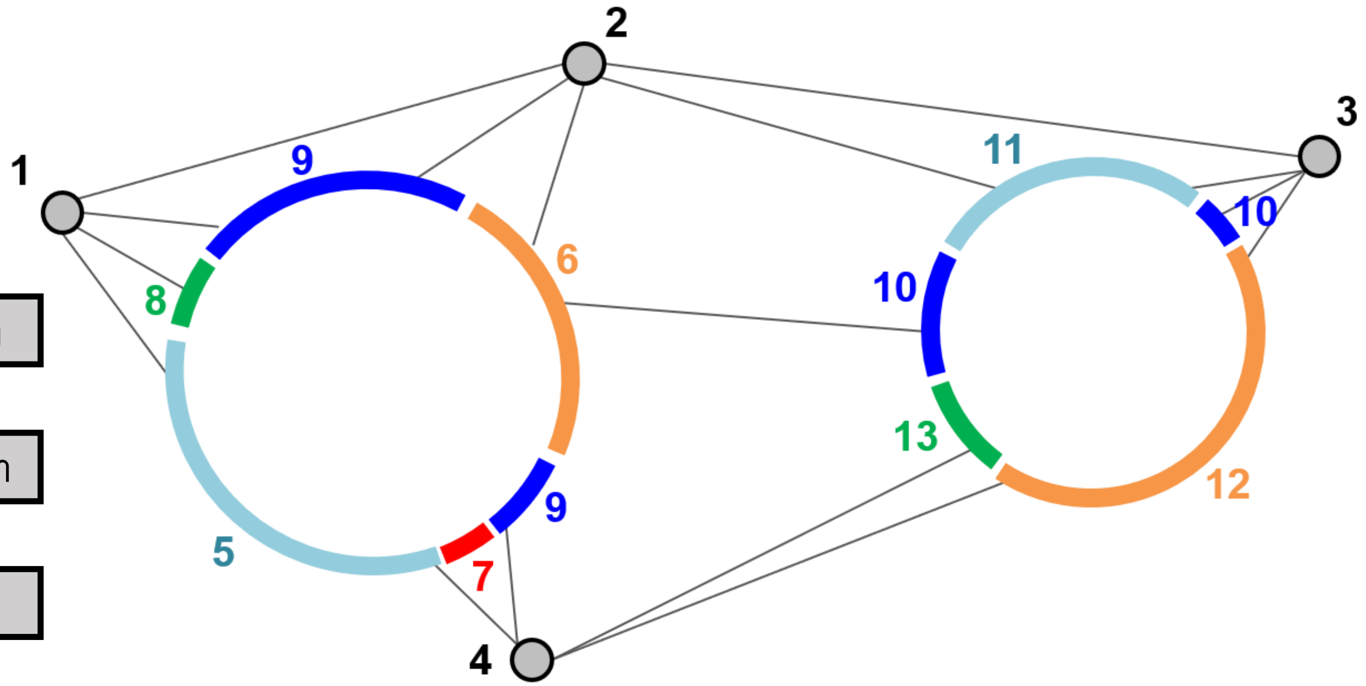
Input

Node Replication

Node Permutation

Node Merging

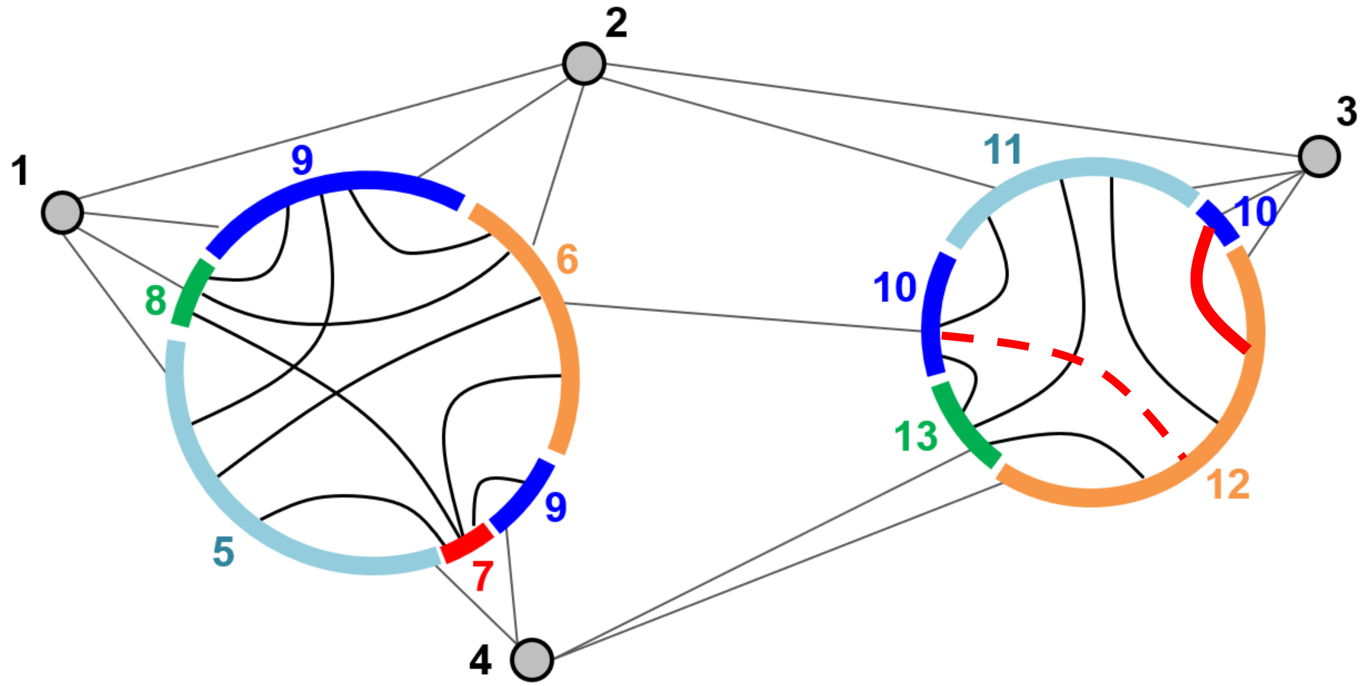
Chord Insertion



General Strategy

Chord Insertion: Insert chords in the diagram

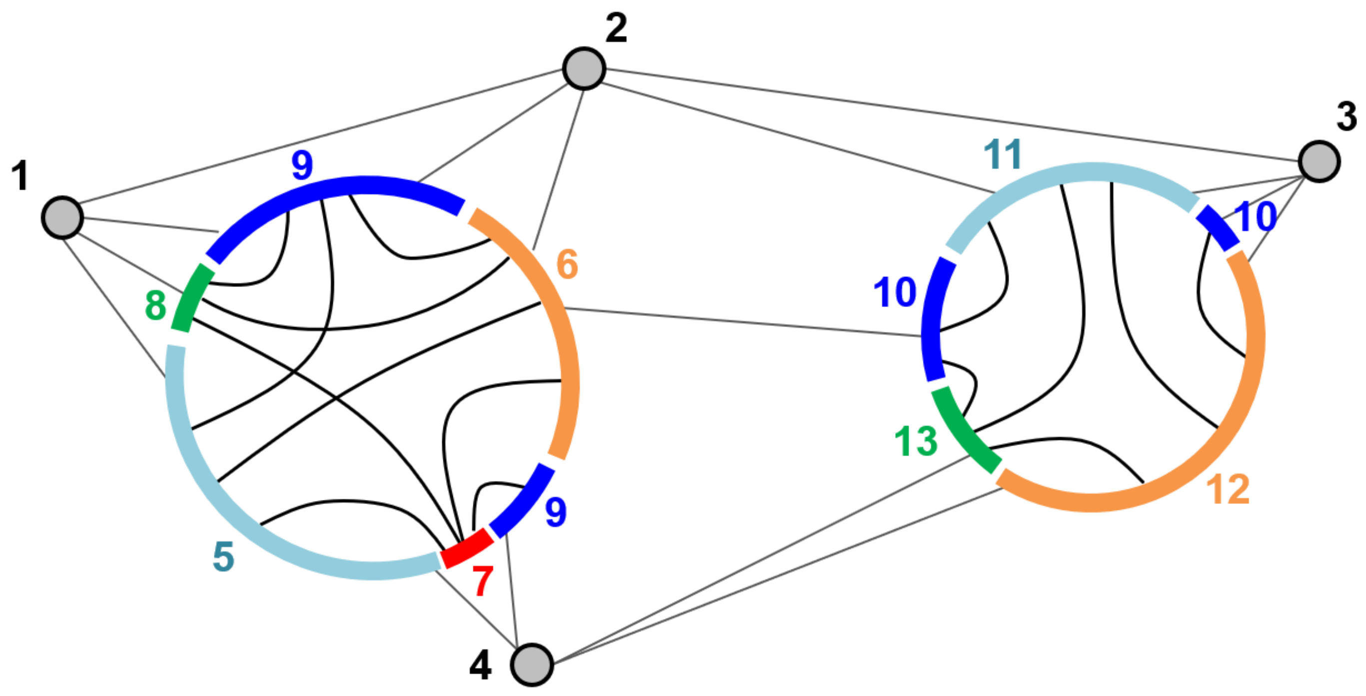
<u>Optimization Goal 2</u>: Minimize the number of crossings and maximize the crossing angle



General Strategy

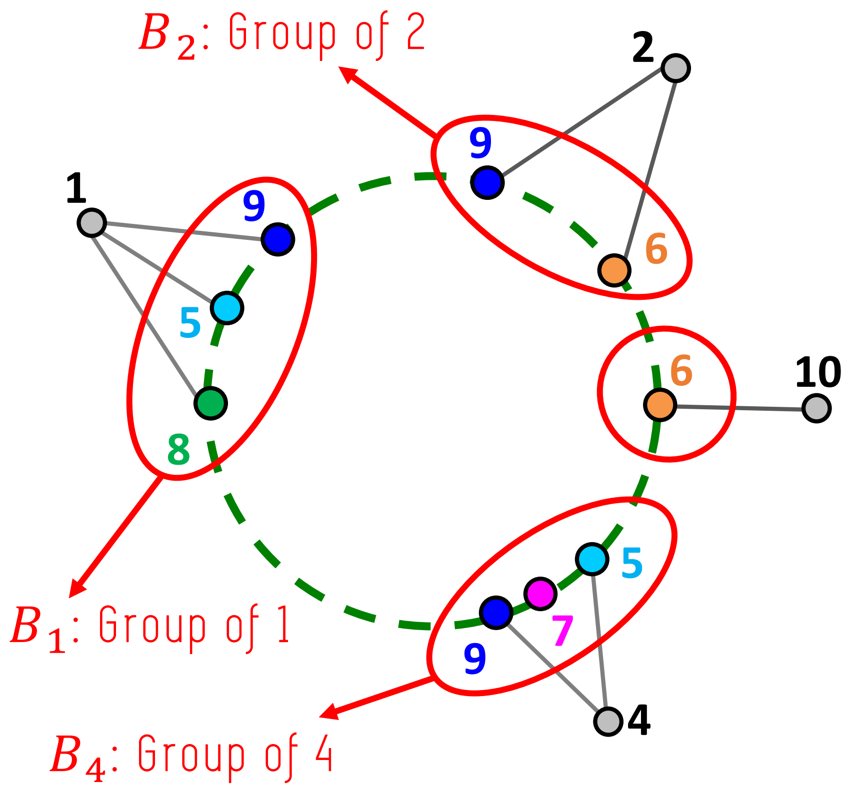
Chord Insertion: Insert chords in the diagram

<u>Optimization Goal 2</u>: Minimize the number of crossings and maximize the crossing angle



Optimization Goal 1 – Node Permutation

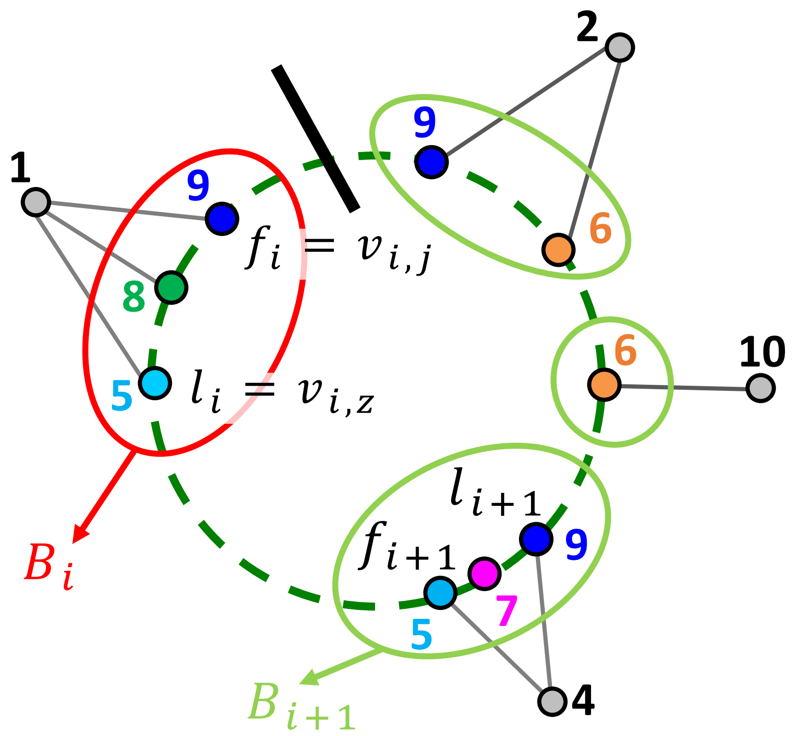
Minimize non-consecutive copies of the same node



Dynamic programming approach

- $B_0, \dots, B_{k-1}$: Clockwise sequence of groups
- The cost of a permutation only depends on the two extreme elements in each group
- f_i: First element of group B_i
- l_i: Last element of group B_i

Optimization Goal 1 – Node Permutation

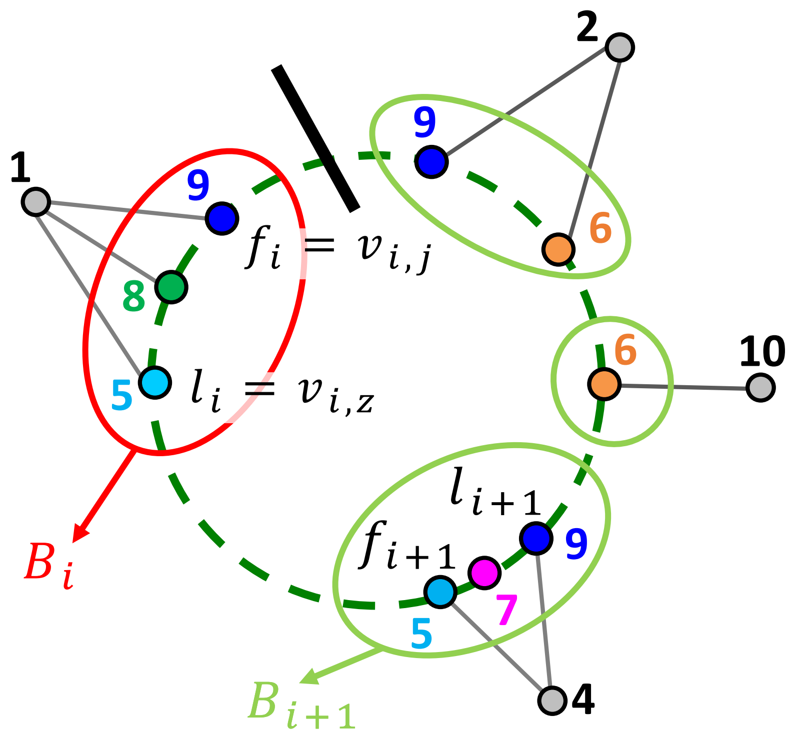


Dynamic programming approach

- Consider a linear sequence of groups
- Suppose to have chosen the first and the last element for $B_{i+1}, \dots, B_{k-1}$ (indices taken modulo k)
- $O_i(v_{i,j}, v_{i,z})$: Cost of choosing $f_i = v_{i,j}$ and $l_i = v_{i,z}$

- $$O_i(\underbrace{v_{i,j}}_{f_i}, \underbrace{v_{i,z}}_{l_i}) = O_{i+1}(\underbrace{v_{i+1,j'}}_{f_{i+1}}, \underbrace{v_{i+1,z'}}_{l_{i+1}}) + \begin{cases} 0, \text{if } v_{i+1,j'} = v_{i,z} \\ 1, \text{if } v_{i+1,j'} \neq v_{i,z} \end{cases}$$

Optimization Goal 1 – Node Permutation



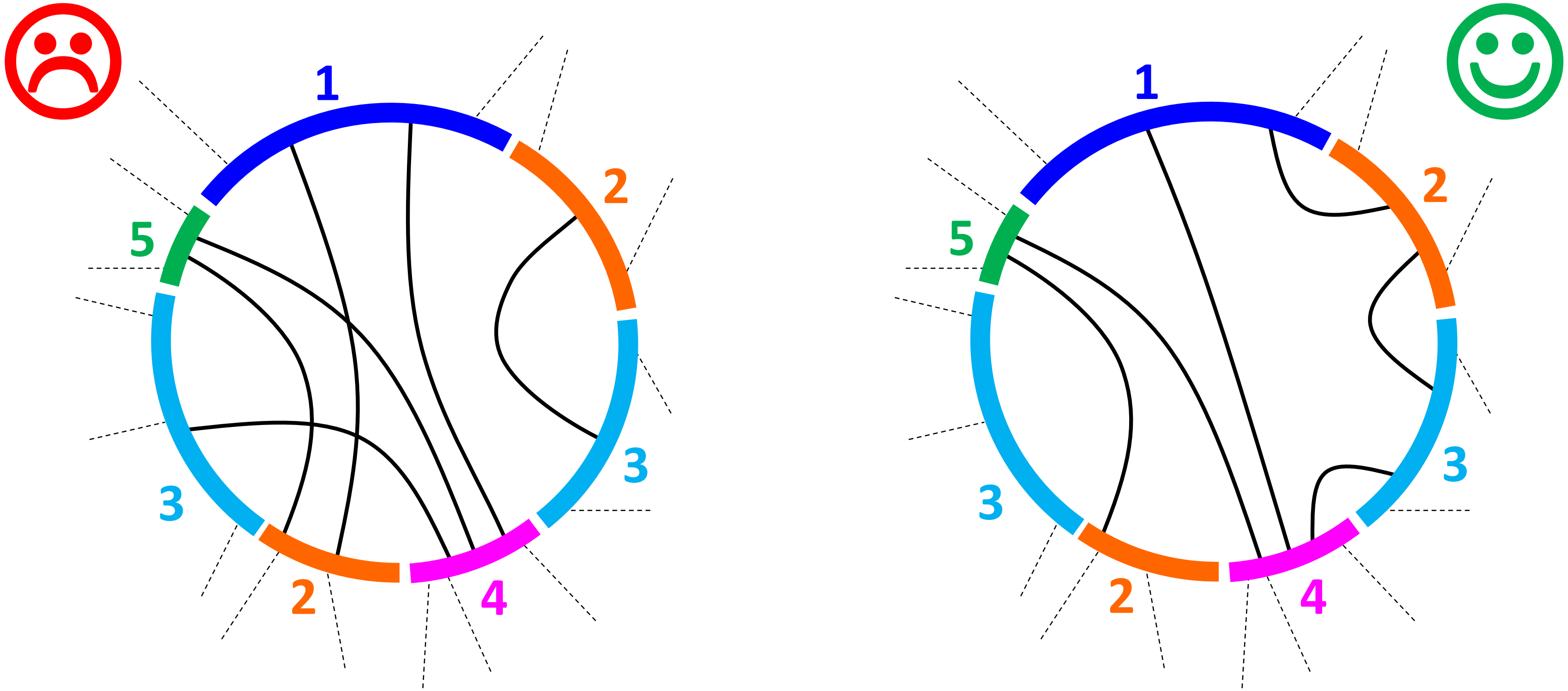
Dynamic programming approach

- Consider a linear sequence of groups
- Suppose to have chosen the first and the last element for $B_{i+1}, \ldots, B_{k-1}$ (indices taken modulo k)
- $O_i(v_{i,j}, v_{i,z})$: Cost of choosing $f_i = v_{i,j}$ and $l_i = v_{i,z}$

- $O_i(v_{i,j}, v_{i,z}) = O_{i+1}(v_{i+1,j'}, v_{i+1,z'}) + \begin{cases} 0, \text{if } v_{i+1,j'} = v_{i,z} \\ 1, \text{if } v_{i+1,j'} \neq v_{i,z} \end{cases}$
- $\chi_{\text{opt}} = \min_{v_{0,j}, v_{0,z} \in B_0} O_0(v_{0,j}, v_{0,z}) \longrightarrow$ OPTIMAL SOLUTION
- The algorithm requires $O(m^3)$ time

Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

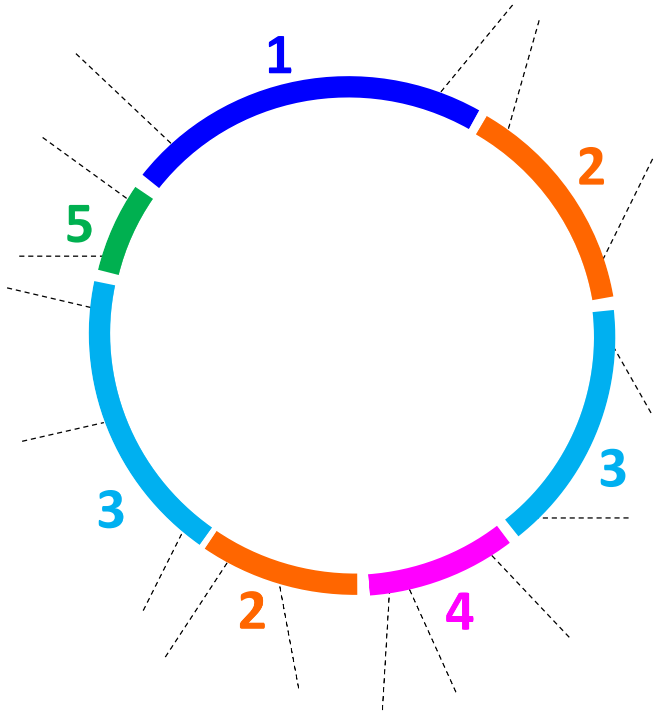


Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

Greedy Strategy

Edges: $(1,2)$, $(1,4)$, $(2,3)$, $(2,5)$, $(3,4)$, $(4,5)$



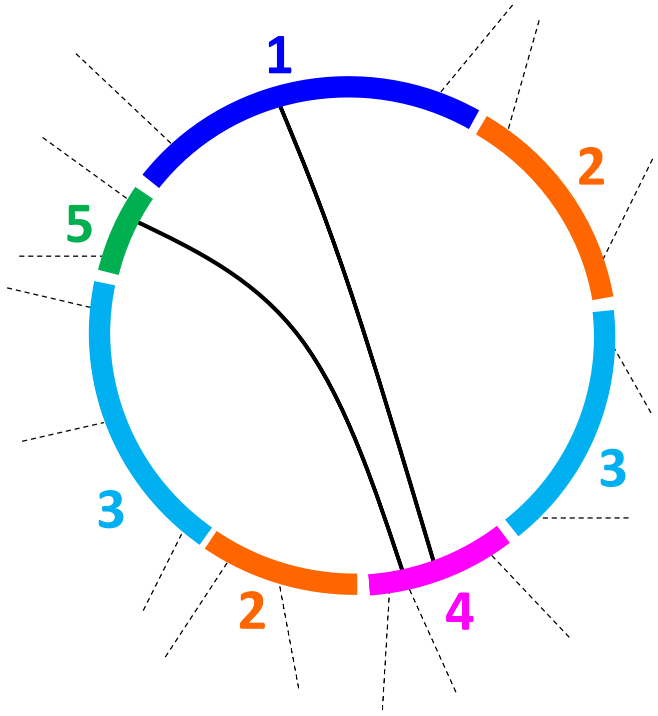
Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

Greedy Strategy

Edges: $(1,2), (2,3), (2,5), (3,4)$

1. Insert edges that have only one possible chord
   $(1,4), (4,5)$



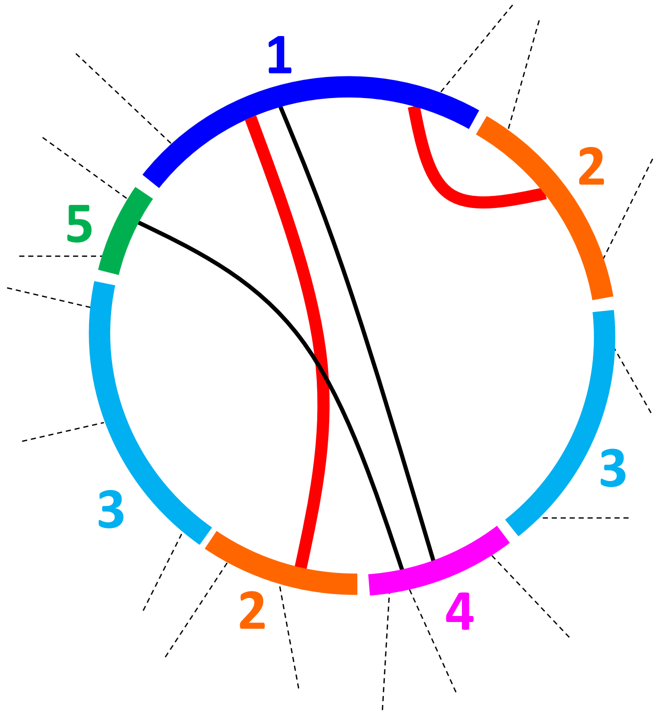
Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

Greedy Strategy

Edges: $(1,2)$, $(2,3)$, $(2,5)$, $(3,4)$

1. Insert edges that have only one possible chord
2. Pick an edge and compute the cost of each chord $(1,2)$



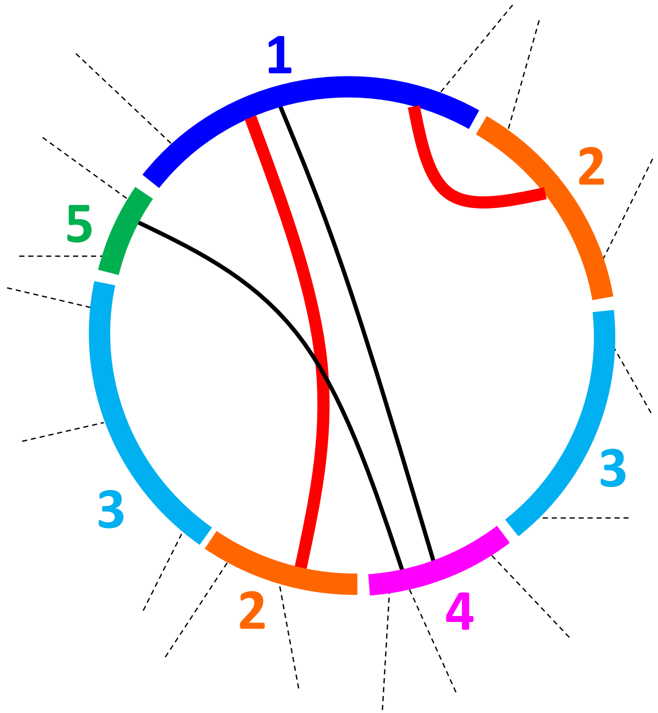
Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

Greedy Strategy

Edges: $(1,2), (2,3), (2,5), (3,4)$

1. Insert edges that have only one possible chord
2. Pick an edge and compute the **cost** of each chord
   $(1,2)$
   - Number of crossings
   - Crossing angle



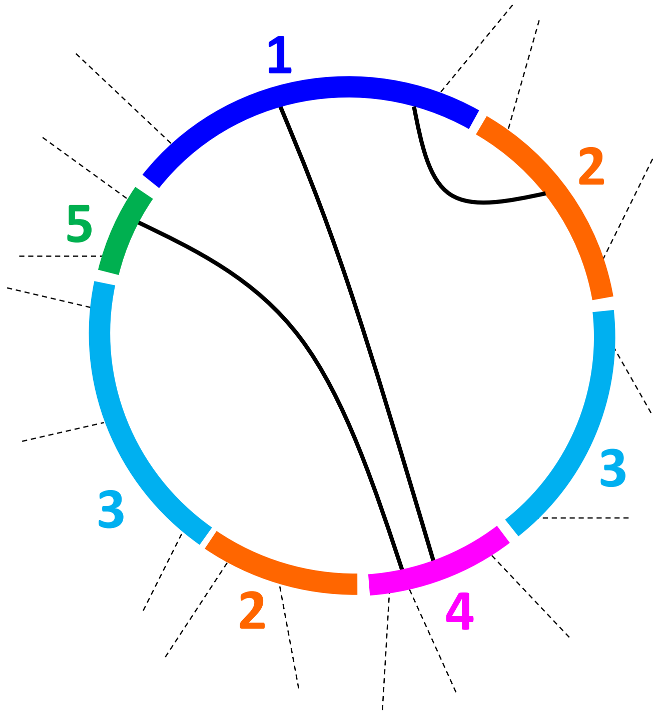
Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

Greedy Strategy

Edges: $(2,3)$, $(2,5)$, $(3,4)$

1. Insert edges that have only one possible chord
2. Pick an edge and compute the **cost** of each chord
   $(1,2)$
   - Number of crossings
   - Crossing angle
3. Choose the chord having the minimum cost



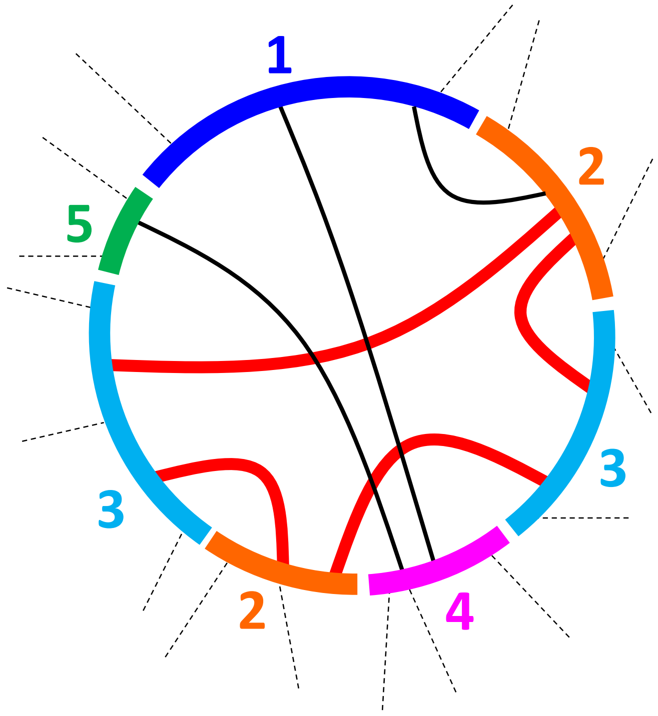
Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

Greedy Strategy

Edges: $(2,3), (2,5), (3,4)$

1. Insert edges that have only one possible chord:
2. Pick an edge and compute the **cost** of each chord $(2,3)$
   - Number of crossings
   - Crossing angle



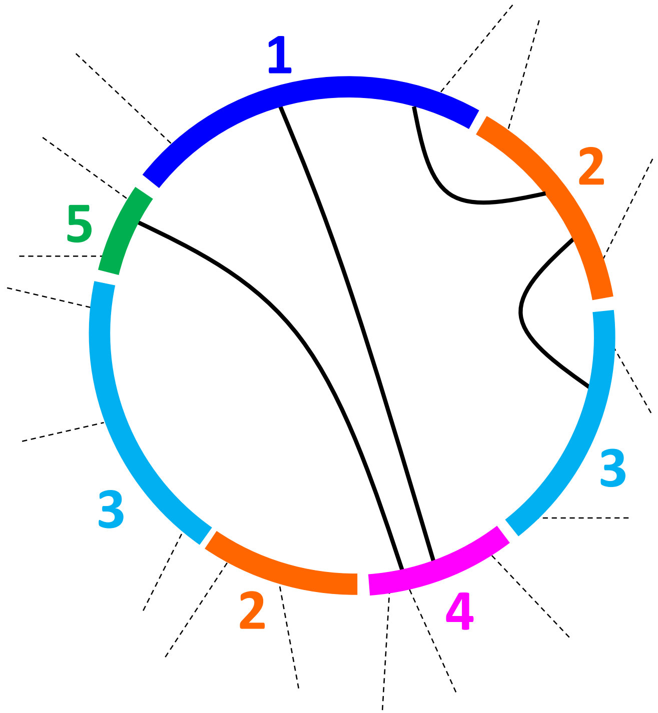
Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

Greedy Strategy

Edges: $(2,5),(3,4)$

1. Insert edges that have only one possible chord:
2. Pick an edge and compute the **cost** of each chord $(2,3)$
   - Number of crossings
   - Crossing angle
3. Choose the chord having the minimum cost



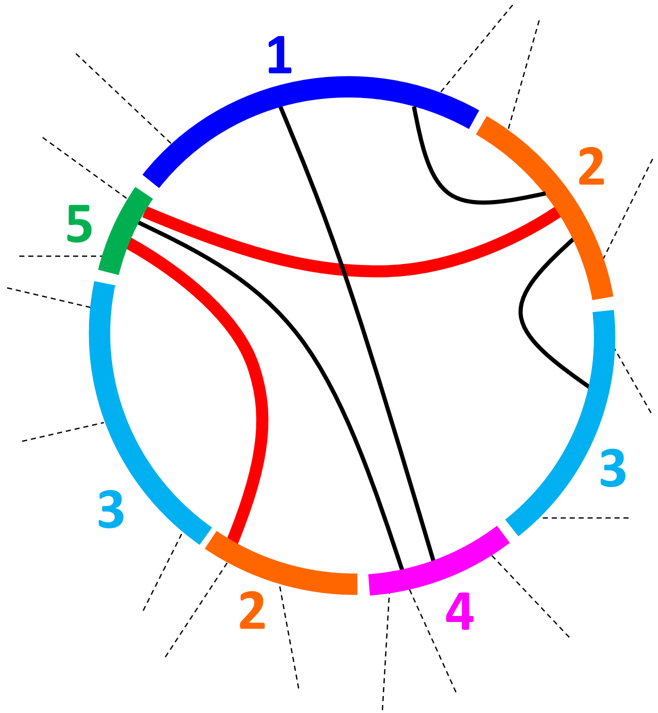
Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

Greedy Strategy

Edges: $(2,5),(3,4)$

1. Insert edges that have only one possible chord
2. Pick an edge and compute the **cost** of each chord $(2,5)$
   - Number of crossings
   - Crossing angle



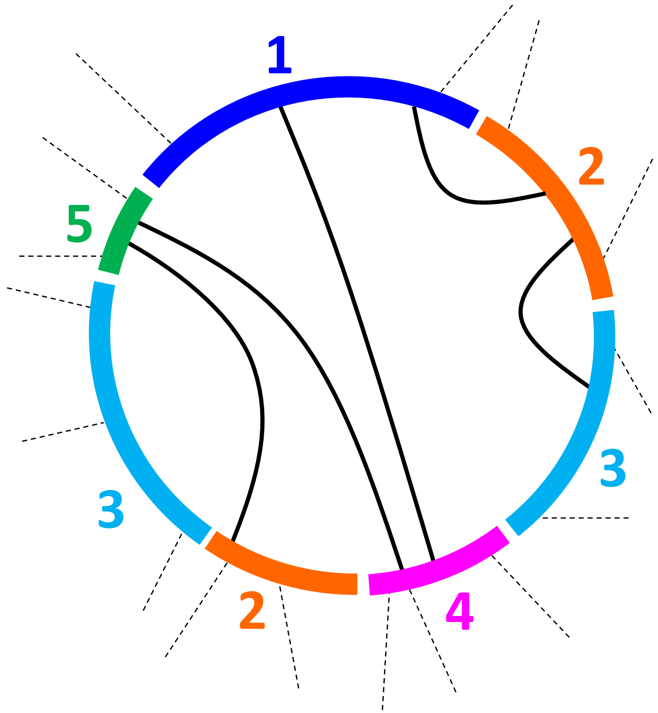
Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

Greedy Strategy

Edges: $(3,4)$

1. Insert edges that have only one possible chord
2. Pick an edge and compute the **cost** of each chord
   $(2,5)$
   - Number of crossings
   - Crossing angle
3. Choose the chord having the minimum cost



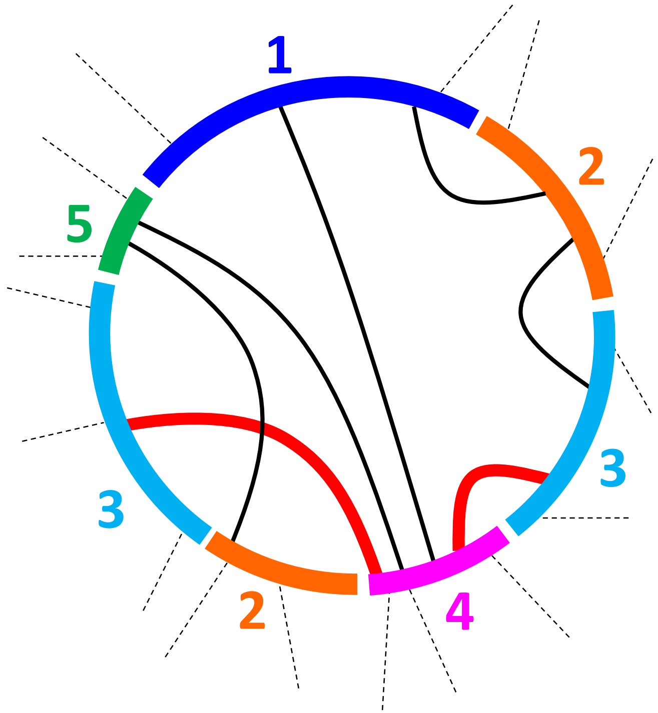
Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

Greedy Strategy

Edges: $(3,4)$

1. Insert edges that have only one possible chord
2. Pick an edge and compute the **cost** of each chord $(3,4)$
   - Number of crossings
   - Crossing angle



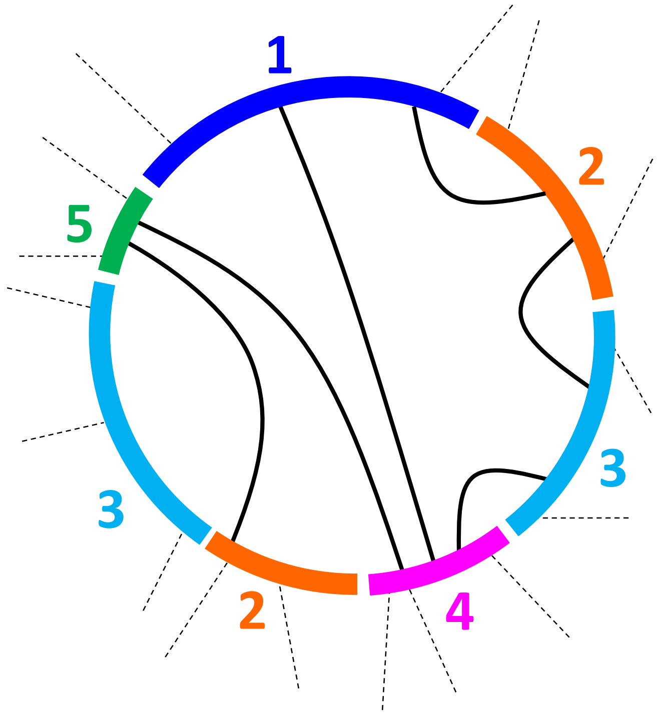
Optimization Goal 2 – Chord Insertion

Minimize number of crossings – Maximize crossing angle

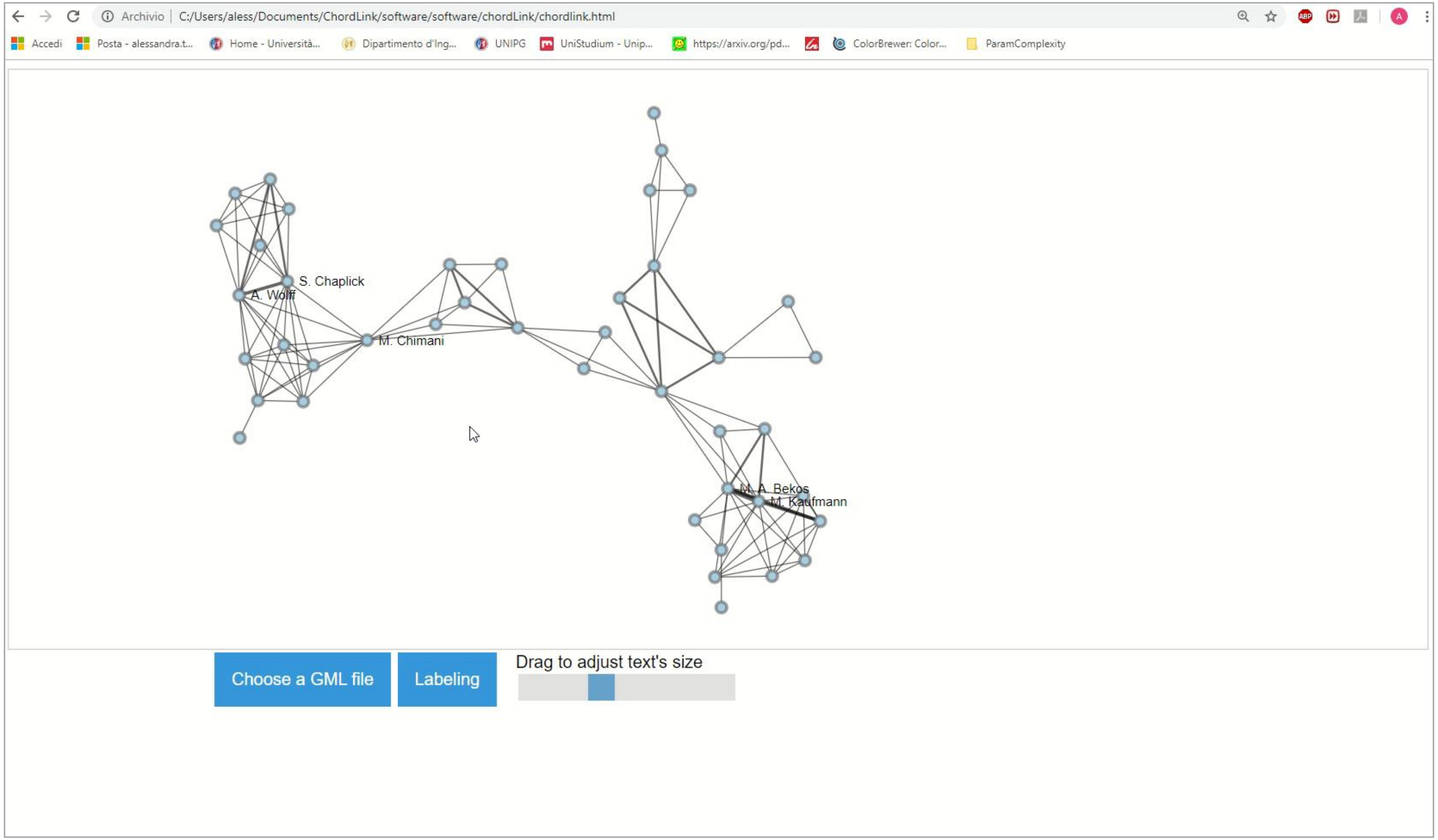
Greedy Strategy

Edges:

1. Insert edges that have only one possible chord
2. Pick an edge and compute the **cost** of each chord
   $(3,4)$
   - Number of crossings
   - Crossing angle
3. Choose the chord having the minimum cost



Our System


Archivio
C:/Users/aless/Documents/ChordLink/software/software/chordLink/chordlink.html
Accedi
Posta - alessandra.t...
Home - Università...
Dipartimento d'Ing...
UNIPG
UniStudium - Unip...
https://arxiv.org/pd...
ColorBrewer: Color...
ParamComplexity
S. Chaplick
A. Wolff
M. Chimani
M. A. Bekos
M. Kaufmann
Choose a GML file
Labeling
Drag to adjust text's size

Final Remarks and Future Work

- We introduced ChordLink, a new hybrid visualization model
- ChordLink keeps the visualization stable during the interaction
- The readability of a chord diagram may degrade for clusters with more than 25-30 nodes

Intriguing research directions:

- Computational complexity of the optimization problems
- Design new algorithms to compare with our heuristics
- Combine ChordLink and NodeTrix models
- Exploit an automatic clustering algorithm

Thank you
for your attention