Efficient Extraction of Multiple Kuratowski Subdivisions

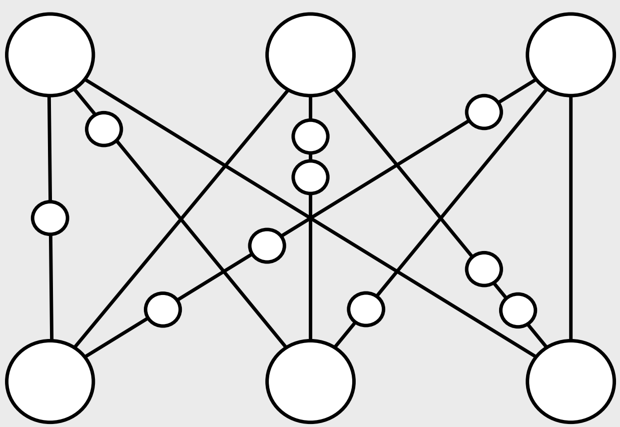
Jens Schmidt

Outline

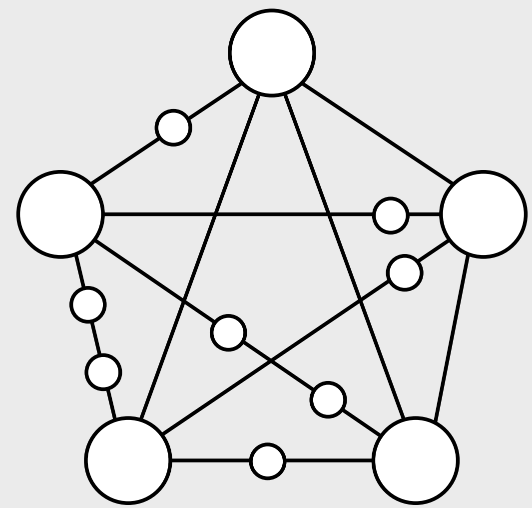
1. Introduction
2. The planarity test of Boyer and Myrvold
3. Extracting multiple Kuratowski subdivisions
4. Experimental results

Planarity

- **Definition:**
  A graph G=(V,E) is planar if and only if it can be embedded in the plane with no edge intersections.



$K_{3,3}$-subdivision



K_5-subdivision

Kuratowski (1930):
A graph is planar if and only if it contains neither a $K_{3,3}$-subdivision nor a K_5-subdivision.

Motivation

Why multiple Kuratowski subdivisions?

Motivation:

Generation of cut constraints for Branch-and-Cut approaches:

- Crossing Minimization problem (NP-hard)
- Maximum Planar Subgraph problem and variants (NP-hard)

Planarity Tests

- **Hopcroft and Tarjan (1974):**
  - First planarity test in linear running time O(n)
  - Complex
  - No Kuratowski subdivision for non-planar graphs

⋮

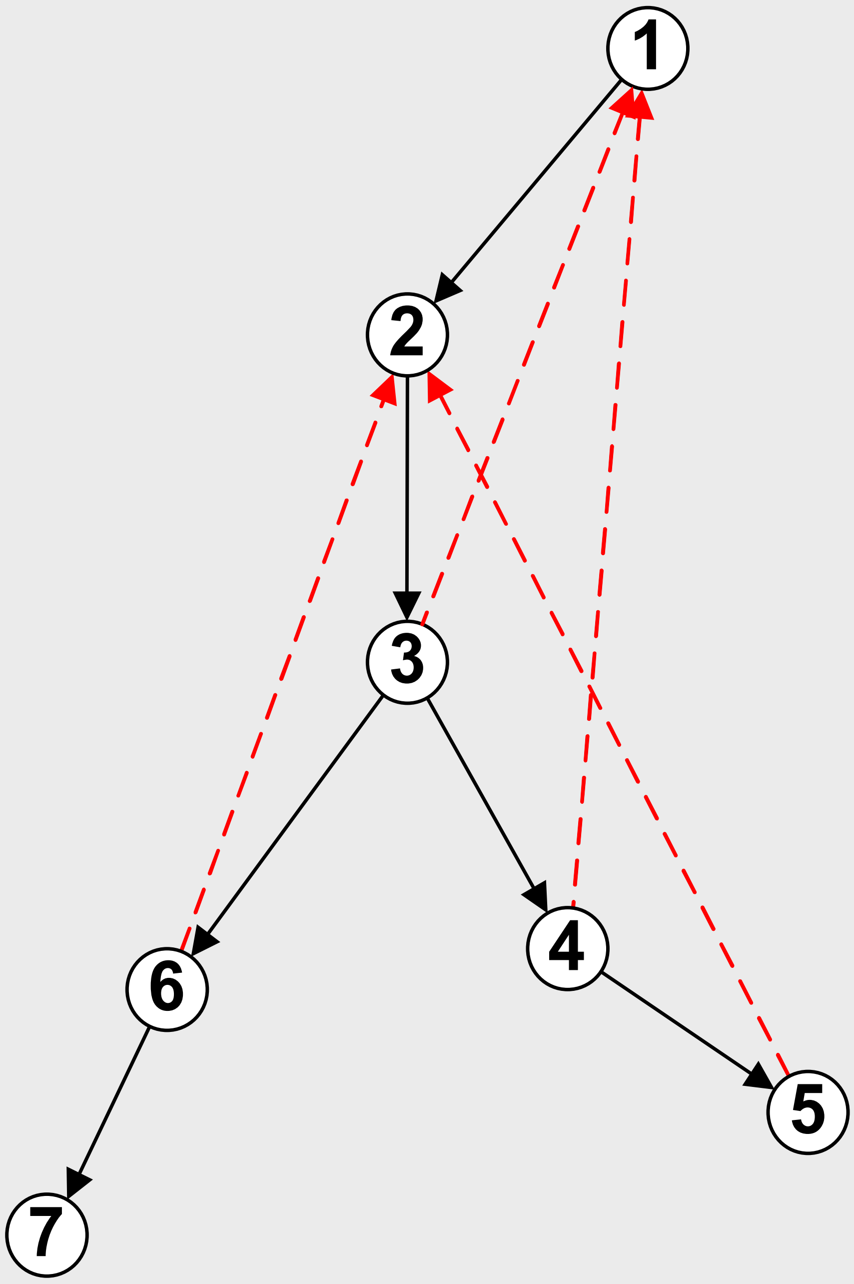
- **Boyer and Myrvold (2004):**
  - Linear running time, very small constant factor
  - Computes planar embedding or Kuratowski subdivision (but only one)
  - Yet quite involved (though not as complex as former planarity tests have been)

Outline

1. Introduction
2. The planarity test of Boyer and Myrvold
3. Extracting multiple Kuratowski subdivisions
4. Experimental results

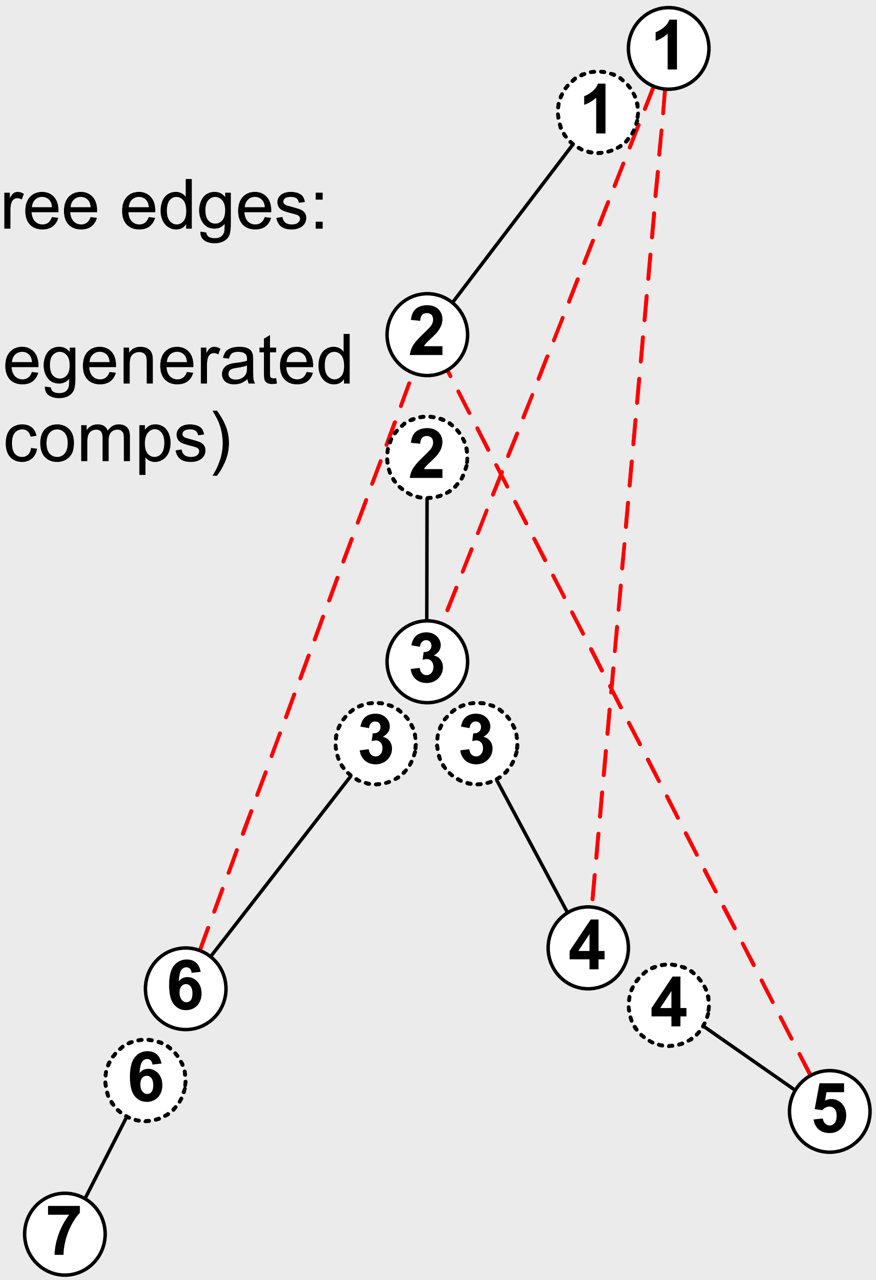
The Boyer-Myrvold Planarity Test

- DFS-based



The Boyer-Myrvold Planarity Test

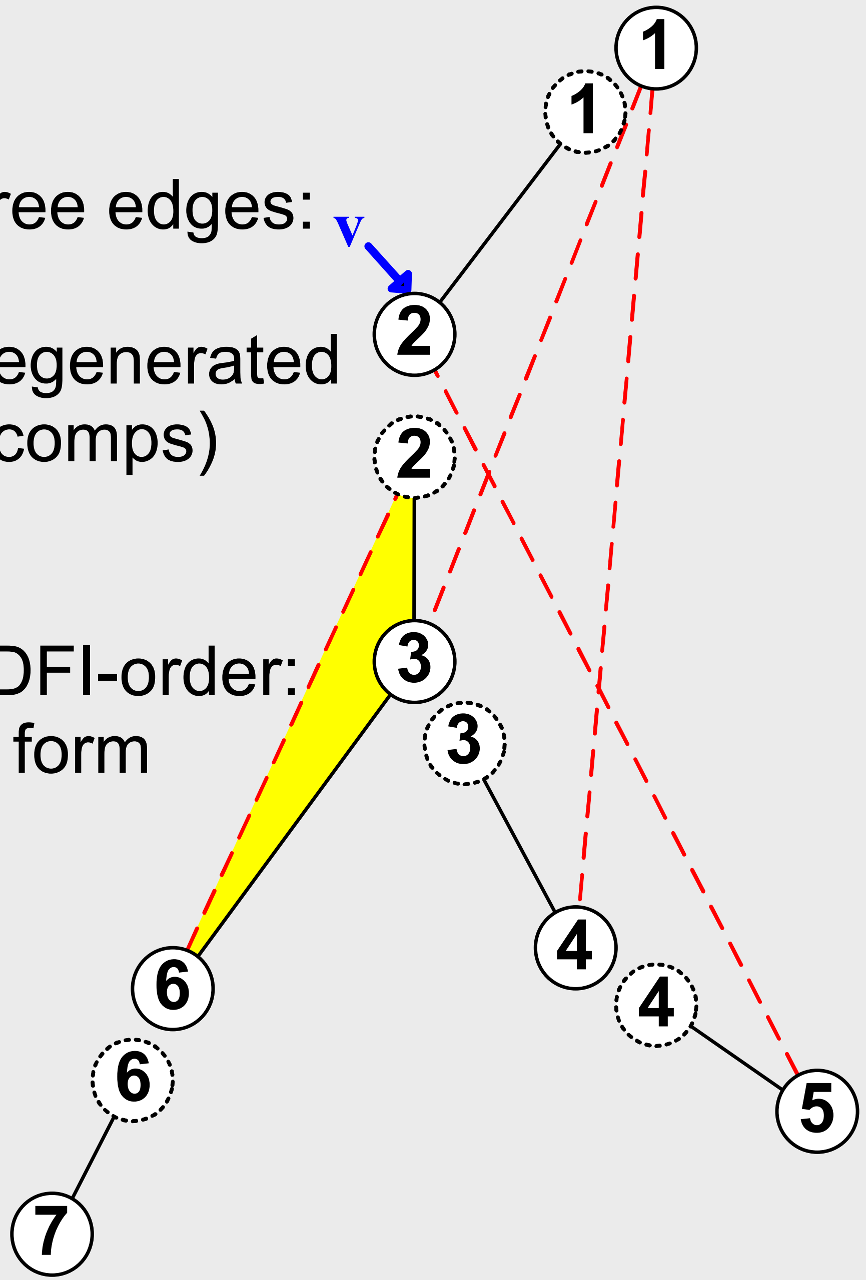
- DFS-based
- Starts with separating all DFS-tree edges:
  - Backedges are ignored
  - Tree edges now represent degenerated biconnected components (bicomps)



The Boyer-Myrvold Planarity Test

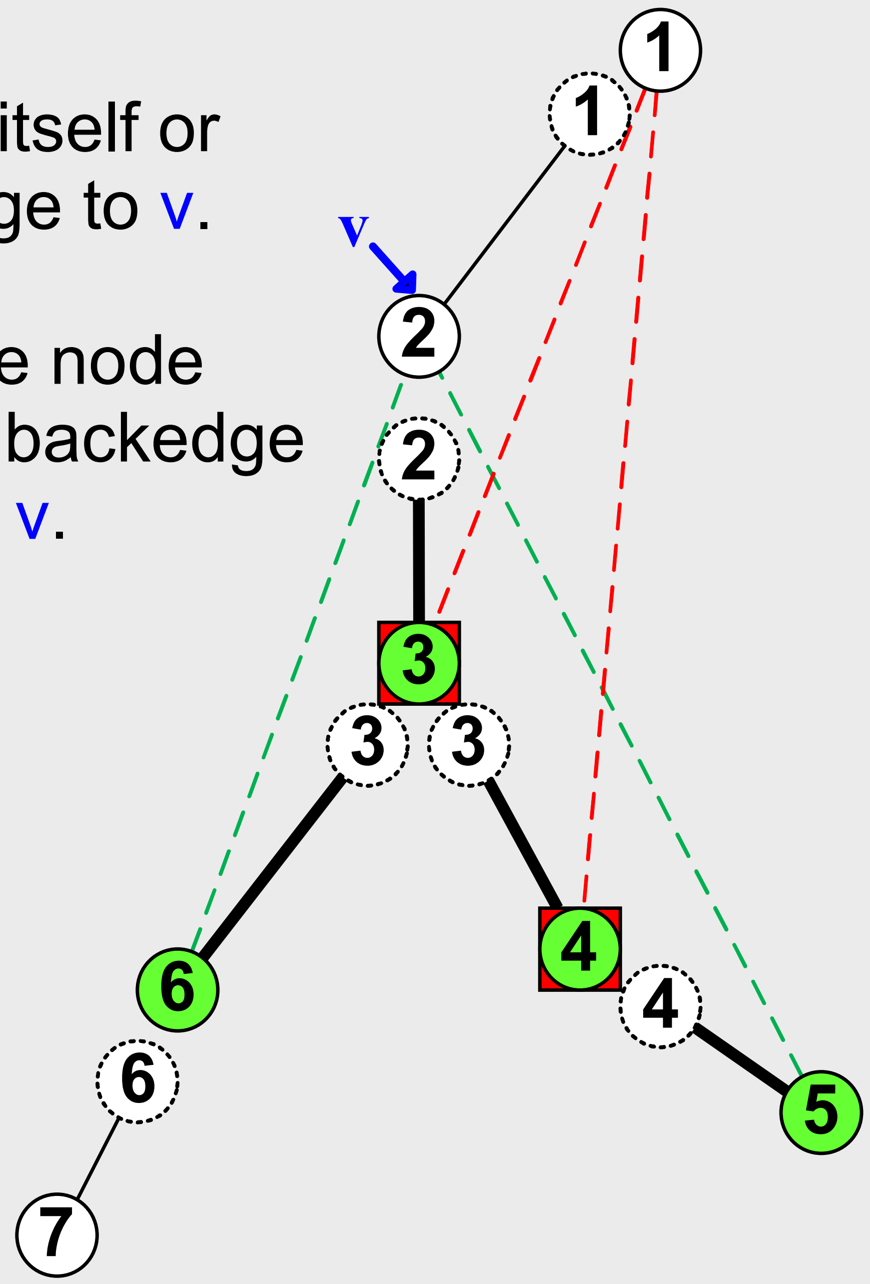
- DFS-based
- Starts with separating all DFS-tree edges:
  - Backedges are ignored
  - Tree edges now represent degenerated biconnected components (bicomps)

- **Idea:**
  For each node v in decreasing DFI-order:
  - Embed all backedges at v to form new, larger bicomps while preserving planarity



Pertinent vs. Externally Active

- A node is pertinent, if the node itself or any child bicomp has a backedge to v.
- A node is externally active, if the node itself or any child bicomp has a backedge to a node with smaller DFI than v.

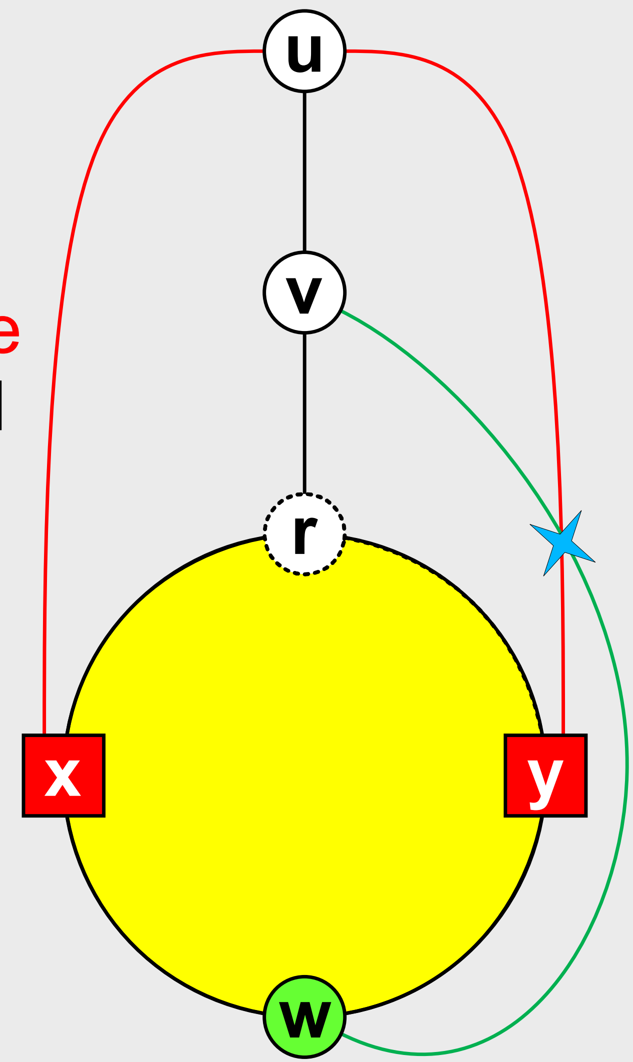


Stopping Configurations

- But how are non-planar graphs identified?

- **Stopping configuration:**
  - Bicomp with two externally active vertices on the external face and a pertinent vertex in between
  - Witness for non-planarity

Boyer & Myrvold:
A graph is planar if and only if no stopping configuration is found.



minor-type A ($K_{3,3}$)

Outline

1. Introduction
2. The planarity test of Boyer and Myrvold
3. Extracting multiple Kuratowski subdivisions
4. Experimental results

The Goal

Extraction of multiple Kuratowski subdivisions in efficient time

Solutions

- Simple approach:
  - Find one subdivision with a planarity test, delete edge of subdivision, iterate...
  - Number of unique subdivisions is limited by m, but may grow exponentially in general
  - $\Theta(mn)$
- Better approach:
  - $\Theta(n+m+\sum_{K \in S} |E(K)|)$, S = set of extracted subdivisions
  - Linear
  - Optimal in terms of output complexity
  - Extends the Boyer-Myrvold planarity test

Extensions

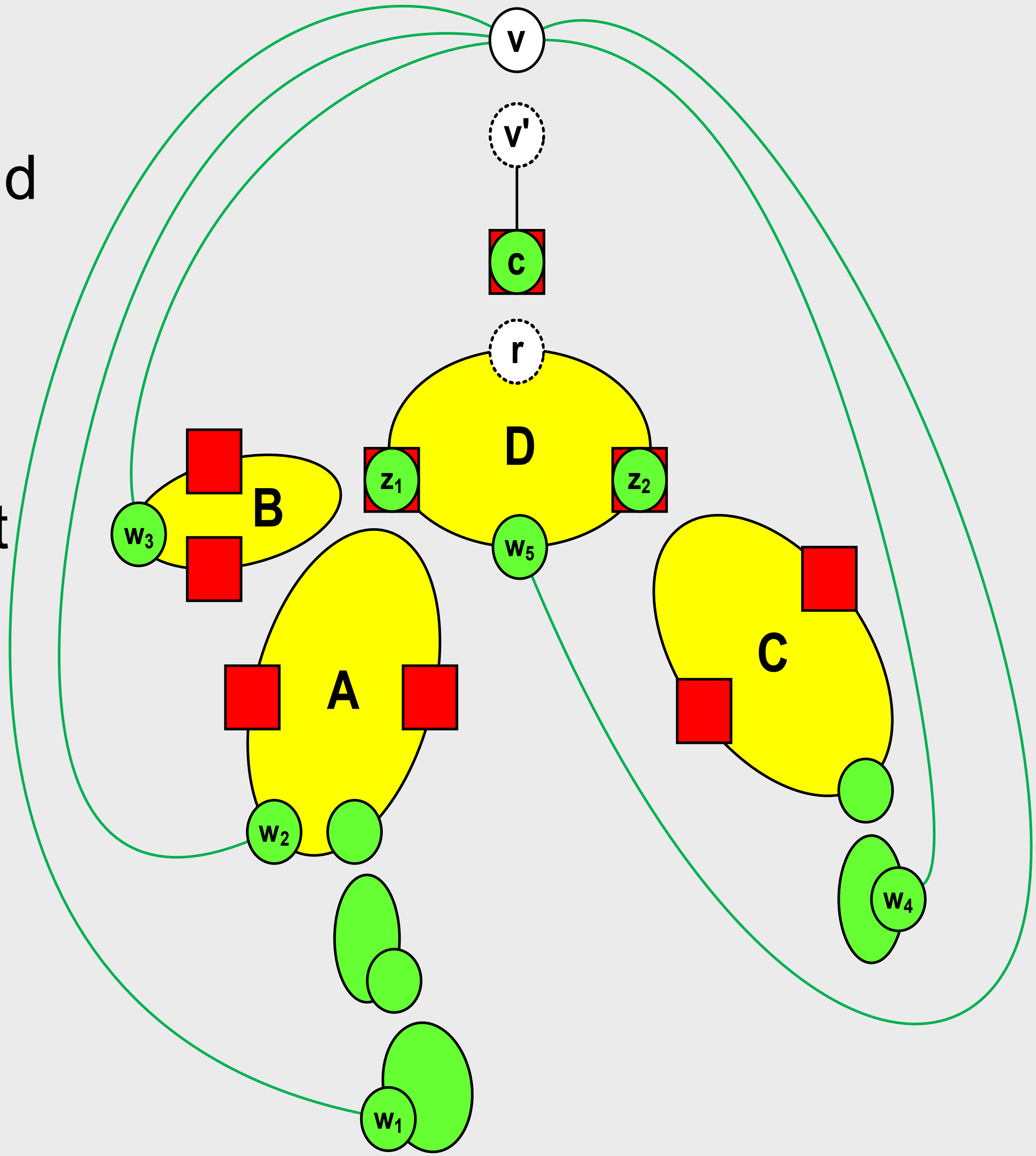
- Find multiple stopping configurations
  - Each stopping configuration contains several minor-types (at least one).
  - Each minor-type induces several Kuratowski subdivisions (at least one).
- Find additional minor-types
- Make the whole extraction efficient

↓

Many extensions and a heavily modified runtime analysis are necessary.

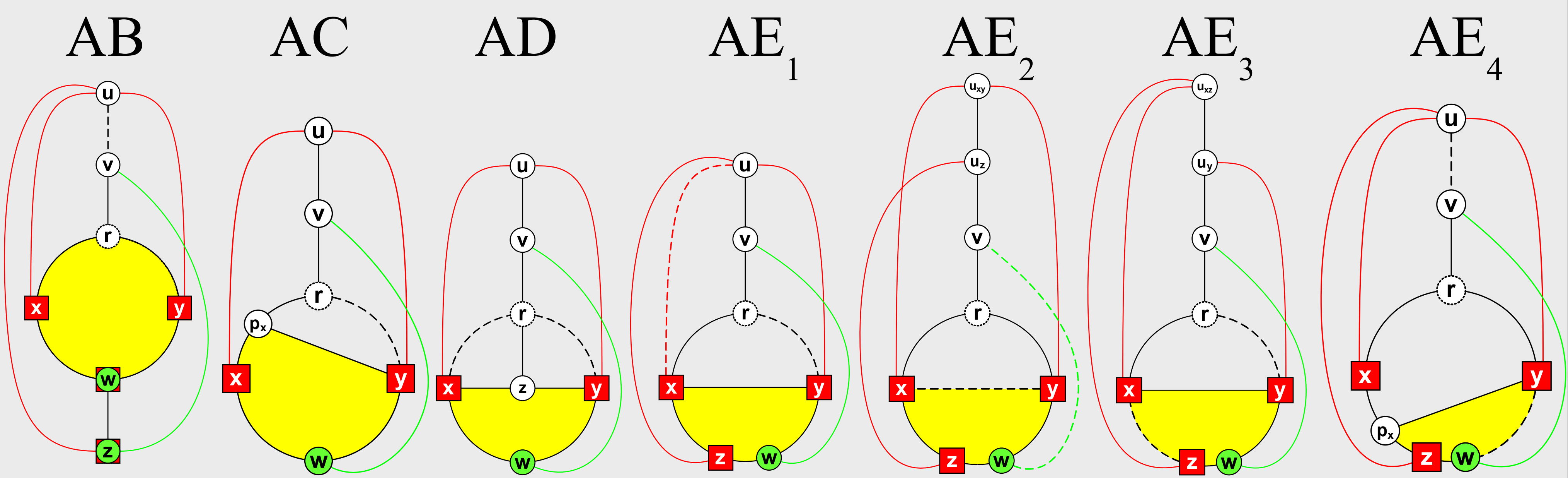
Multiple Stopping Configurations

- Assume a stopping configuration was found on bicomp A.
- Idea:
  - Delete pertinent edges in A
  - Iterate planarity test until next stopping configuration
- Problems:
  - Update underlying data structures efficiently
  - Find next node for reentry



Additional Minor-Types

- A stopping configuration may contain up to 9 different minor-types.
- The 7 additional minor-types below increase the number of extracted subdivisions.
- Delete the dotted lines to get $K_{3,3}$'s.

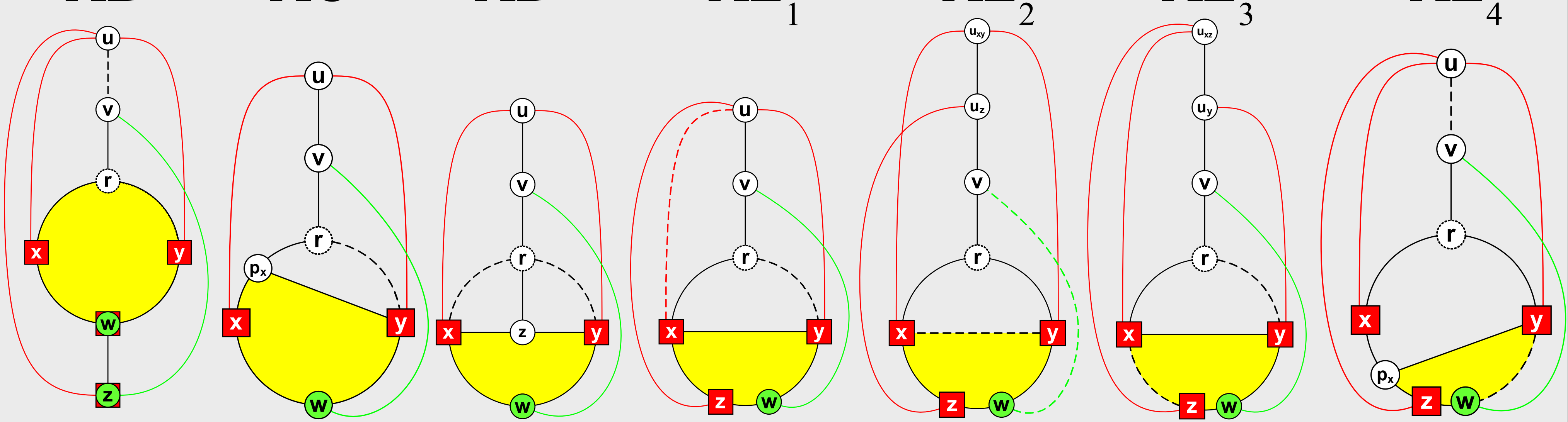


Additional Minor-Types

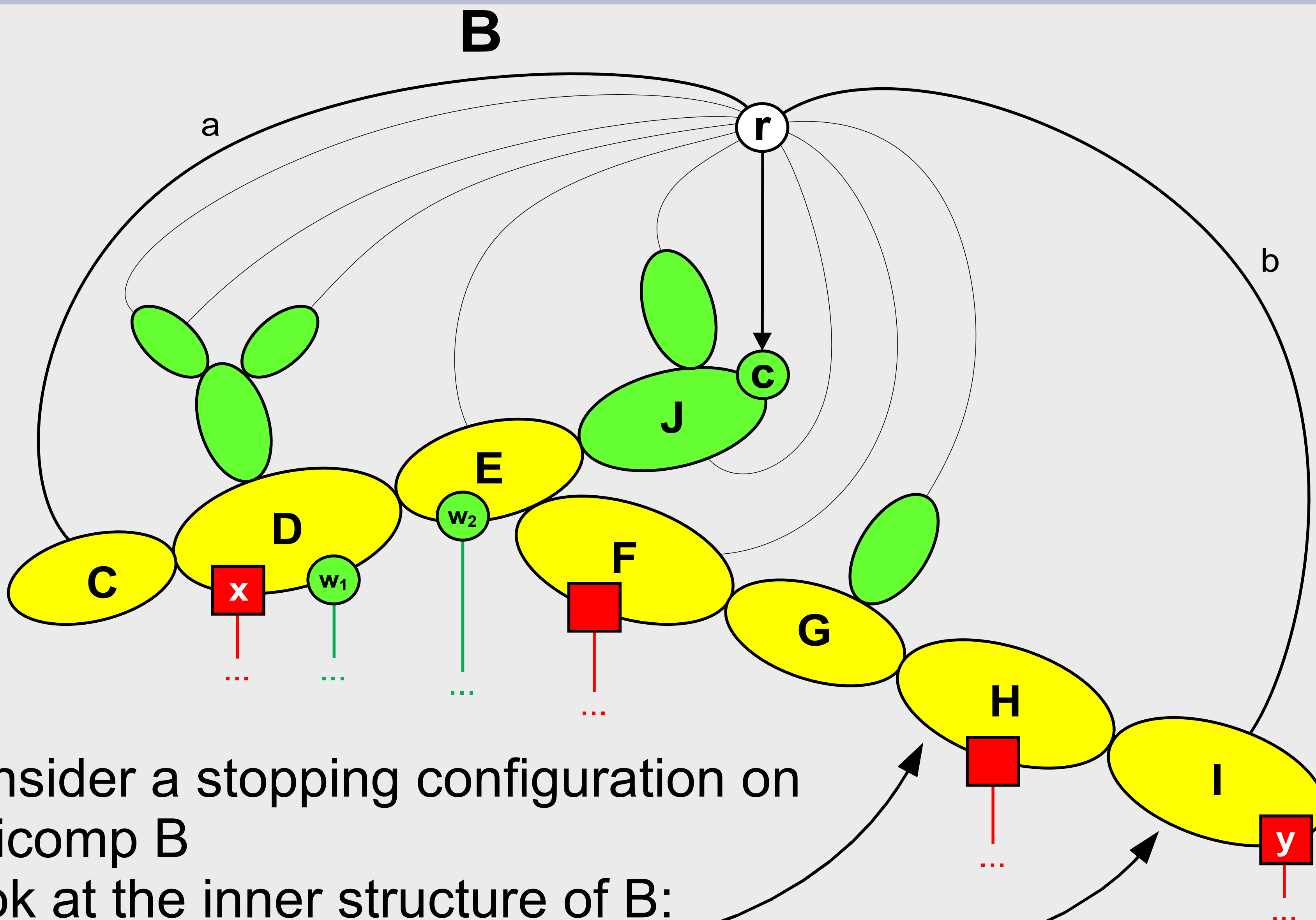
- A stopping configuration may contain up to 9 different minor-types.
- The 7 additional minor-types below increase the number of extracted subdivisions.
- Delete the dotted lines to get $K_{3,3}$'s.

- Most of them contain the so-called highest face path.

AB AC AD AE_1 AE_2 AE_3 AE_4



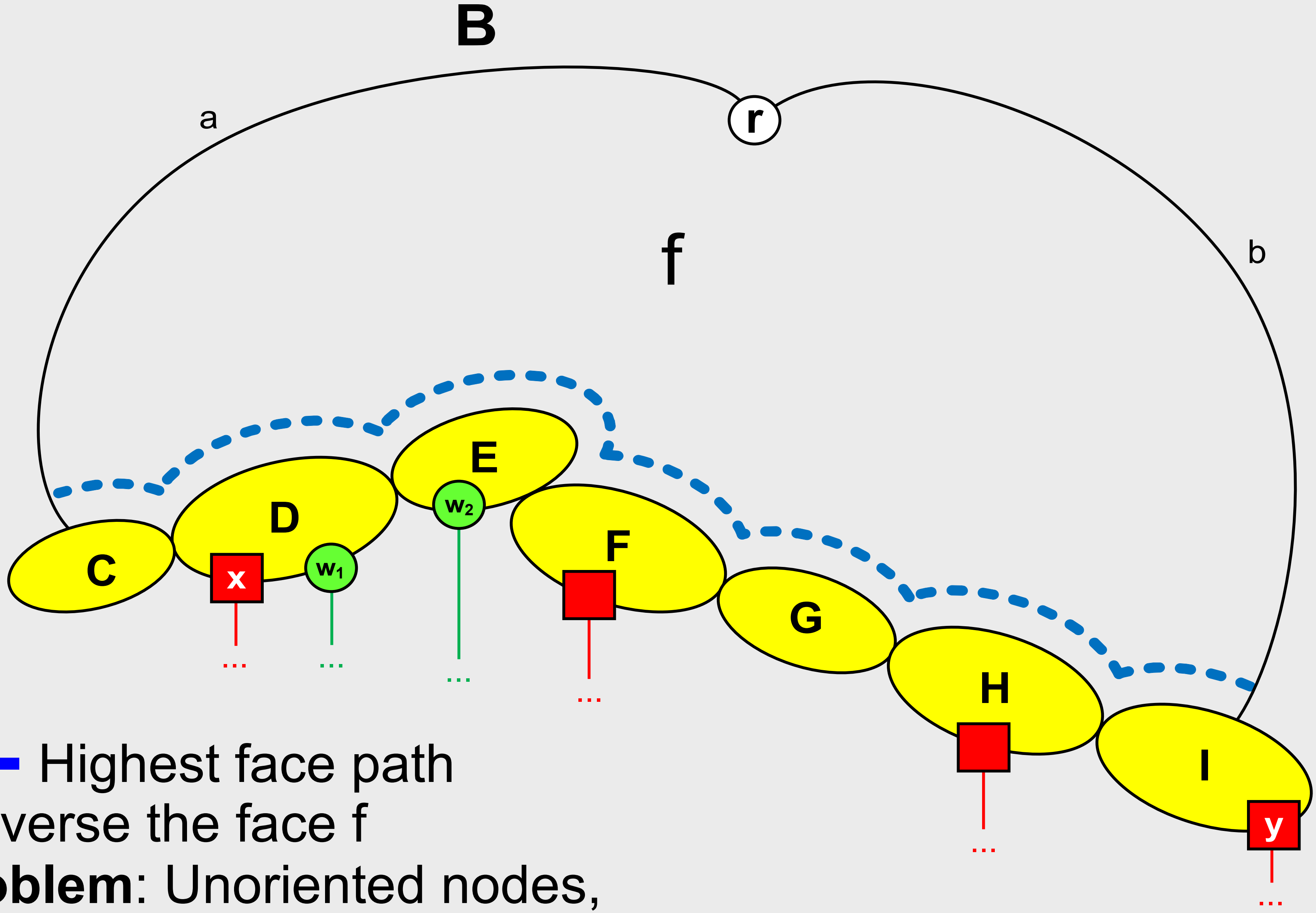
Highest Face Path



- Consider a stopping configuration on a bicomp B
- Look at the inner structure of B:

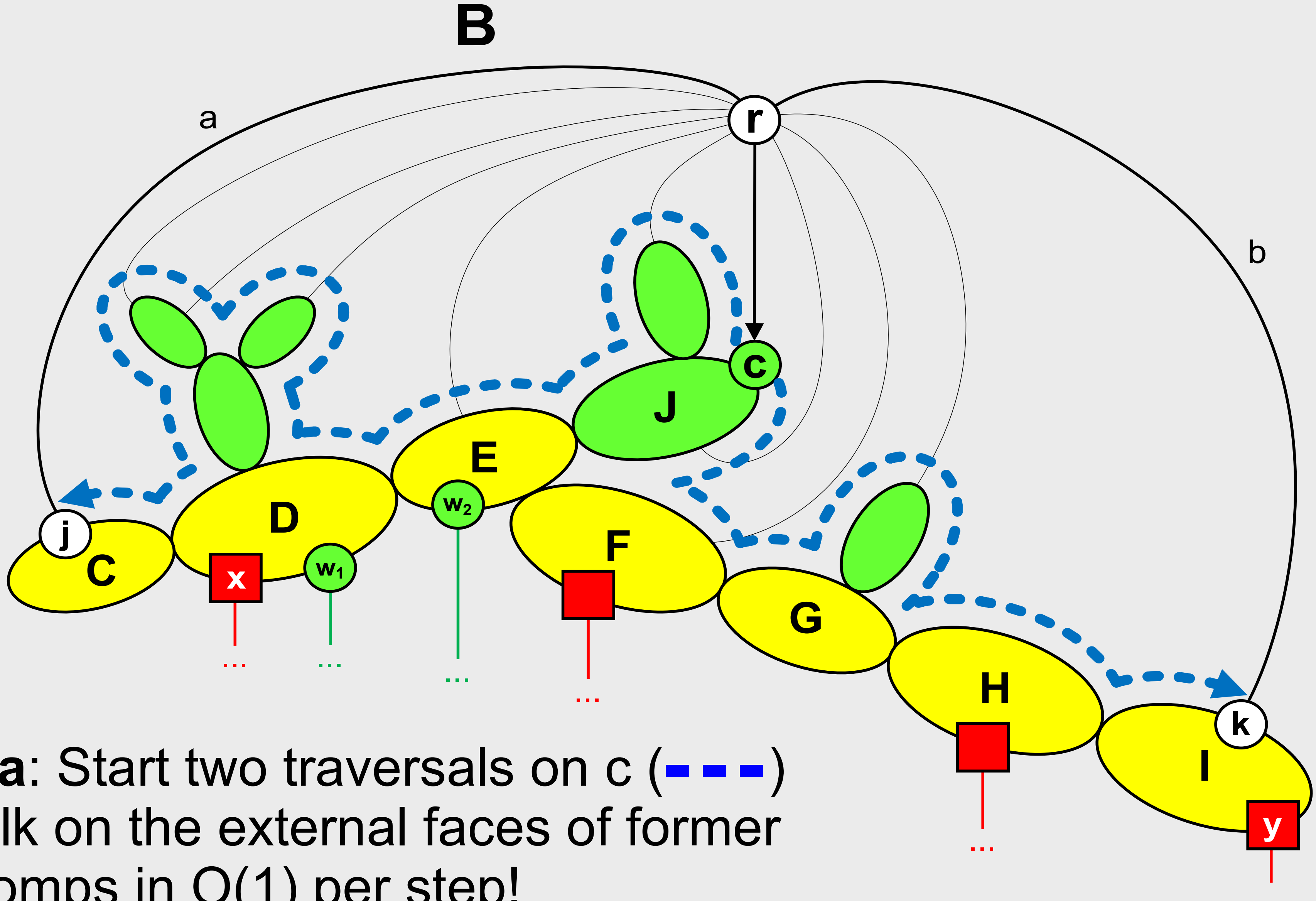
  Former bicomps

Highest Face Path



- Highest face path
- Traverse the face f
- **Problem**: Unoriented nodes, reorientation too costly

Highest Face Path



- **Idea**: Start two traversals on c (- - -)
- Walk on the external faces of former bicomps in O(1) per step!
- We can bound all green parts by a small term.

Obtaining Linear Running Time

Computation step	Overall running time
Extended Walkup	$O(n + m + \sum_{K \in S} \|E(K)\|)$
Extended Walkdown	$O(n + m + \sum_{K \in S} \|E(K)\|)$
– Short-circuit edges	$O(n + m)$
Additional backedgepaths	$O(\sum_{K \in S} \|E(K)\|)$
Classification of minor-types	$O(n + m + \sum_{K \in S} \|E(K)\|)$
Extraction of...	
– ...nodes v,r,x and y	$O(m)$
– ...critical backedgepaths	$O(n + m + \sum_{K \in S} \|E(K)\|)$
– ...external backedgepaths	$O(\sum_{K \in S} \|E(K)\|)$
– ...highest face path	$O(n + m)$
– ...position of the highest face path	$O(\sum_{K \in S} \|E(K)\|)$
– ...external z-nodes (E/AE)	$O(\sum_{K \in S} \|E(K)\|)$
Extraction of all minor-types	$O(\sum_{K \in S} \|E(K)\|)$

Overall running time: $$\Theta\left(n + m + \sum_{K \in S} |E(K)|\right)$$

(S = set of extracted Kuratowski subdivisions)

Outline

1. Introduction
2. The planarity test of Boyer and Myrvold
3. Extracting multiple Kuratowski subdivisions
4. Experimental results

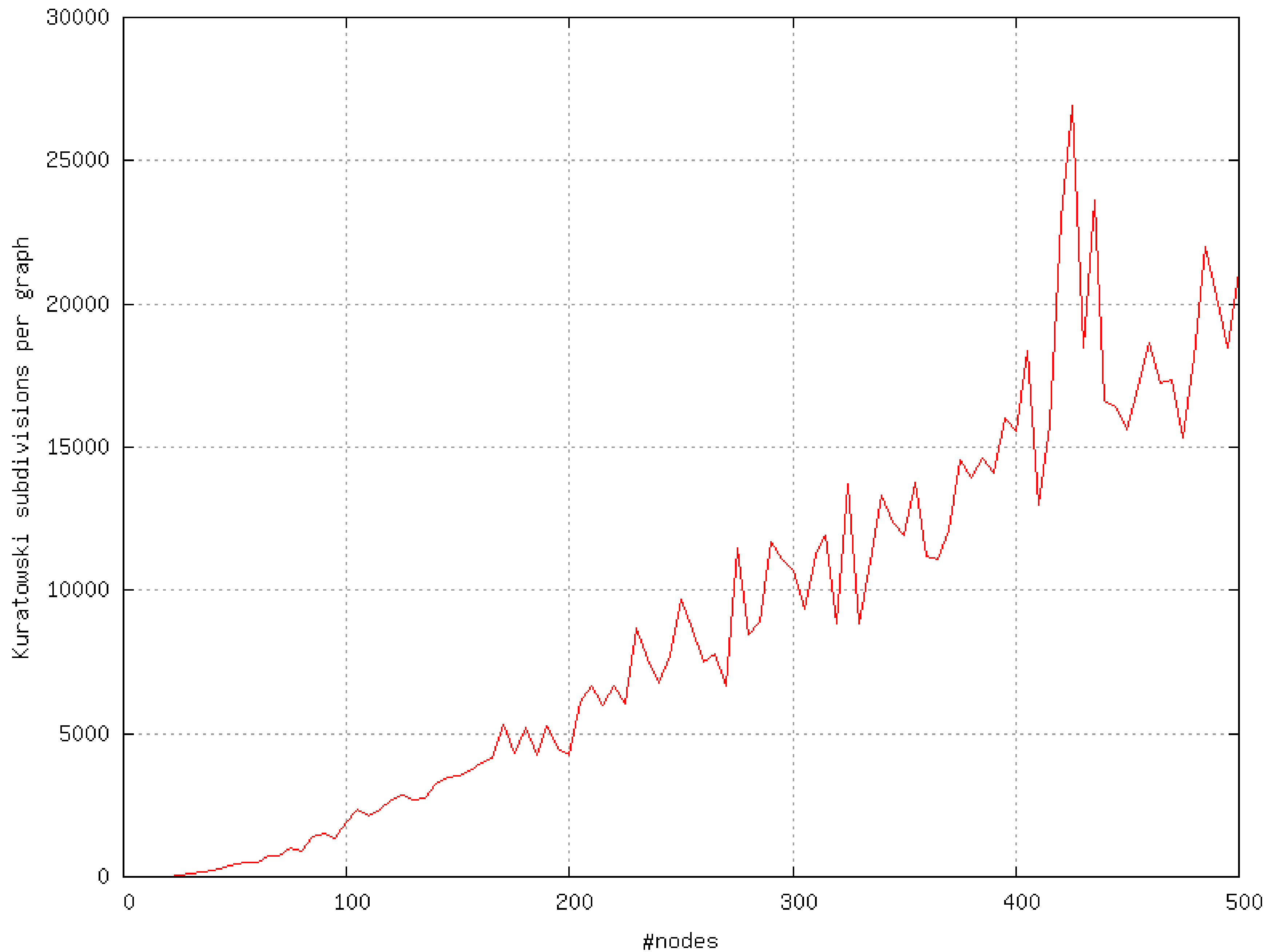
Experimental Setup

- Implemented as part of the Open Graph Drawing Framework (OGDF, open source, C++)

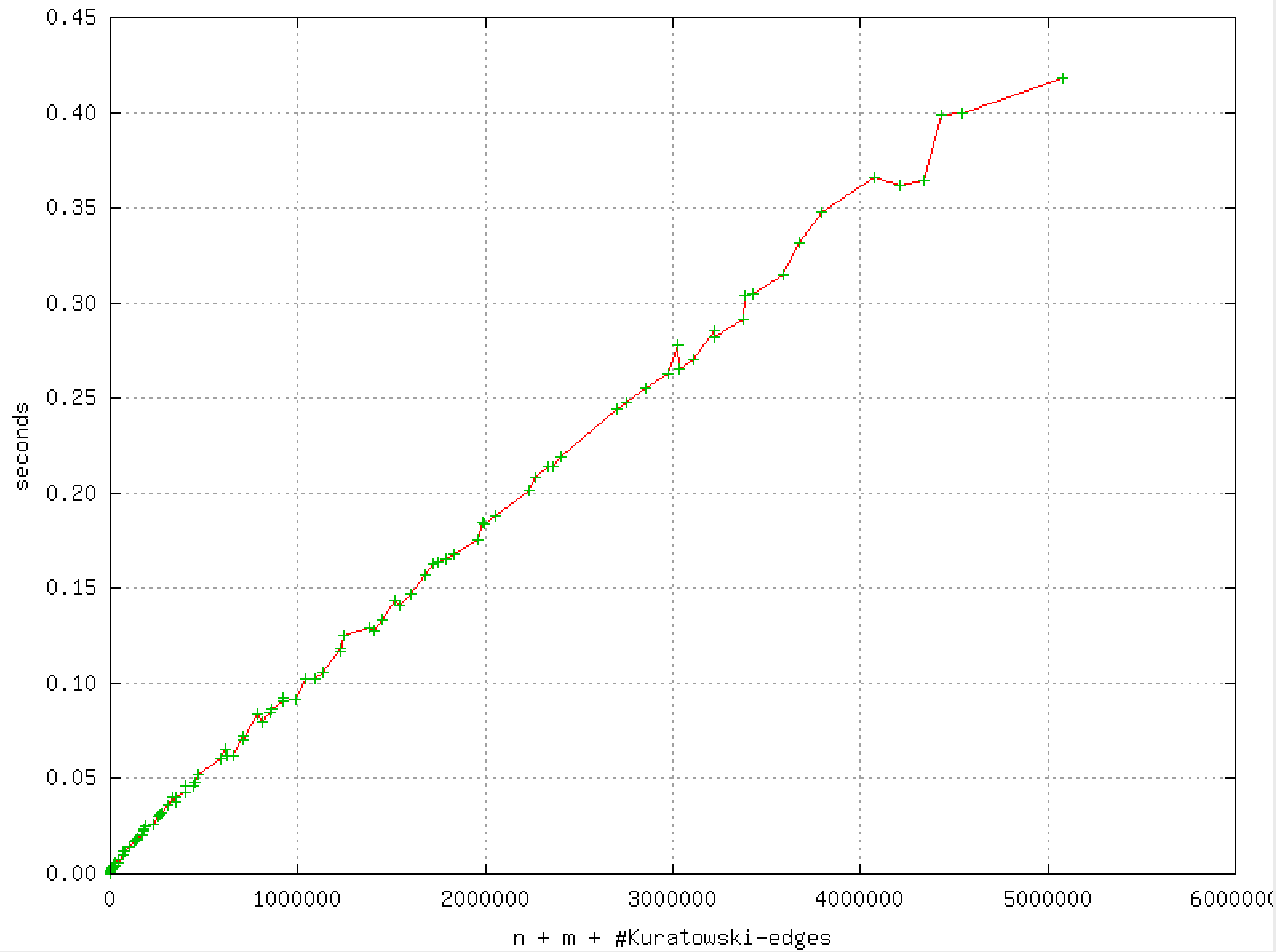
- Tested on DualCore 1.83GHz, gcc 3.4.4 -O1

- Random graphs:
  - 10-500 nodes
  - m = 2n
- Rome Library:
  - 10-100 nodes
  - Sparse
  - >8000 graphs of real world applications

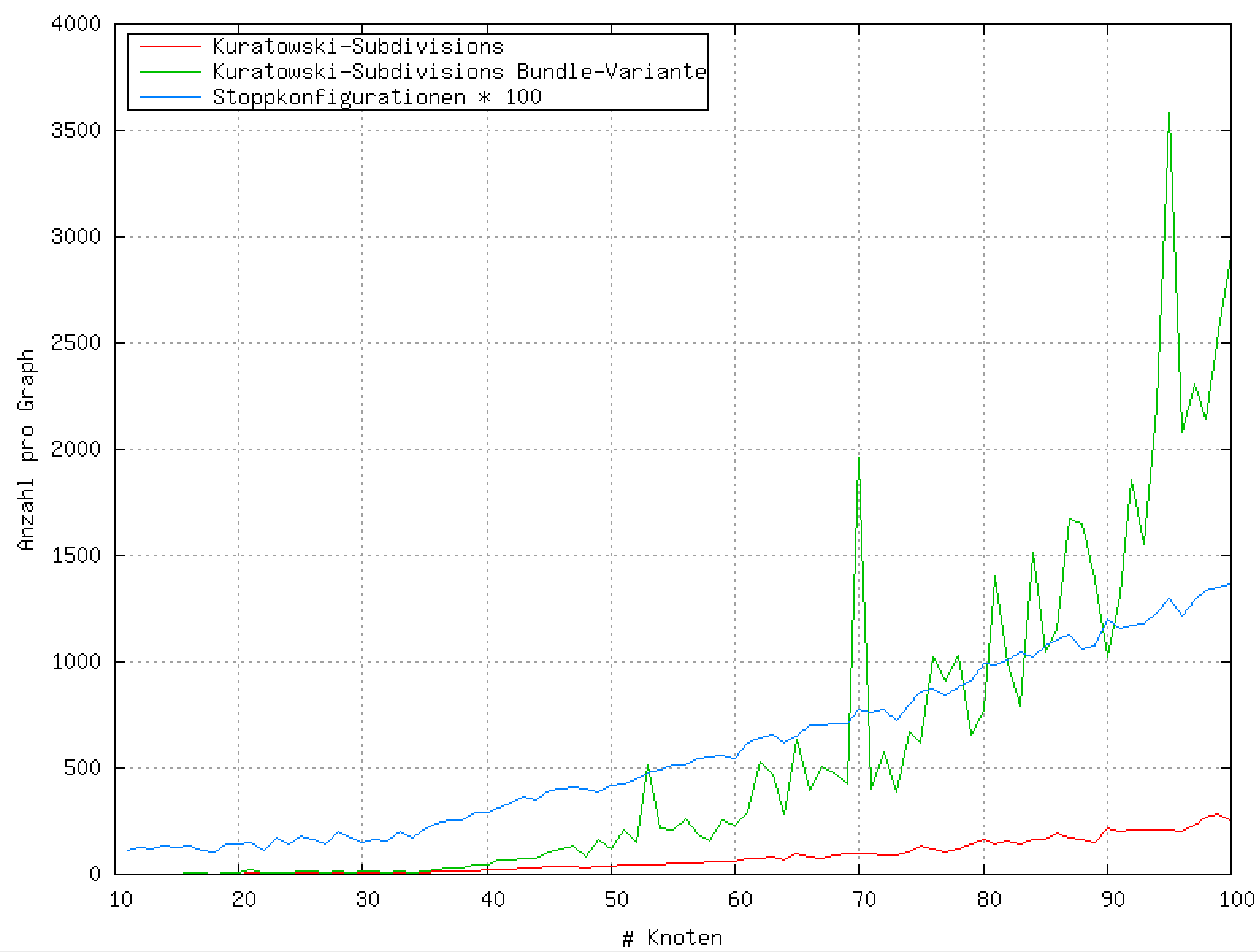
Random Graphs



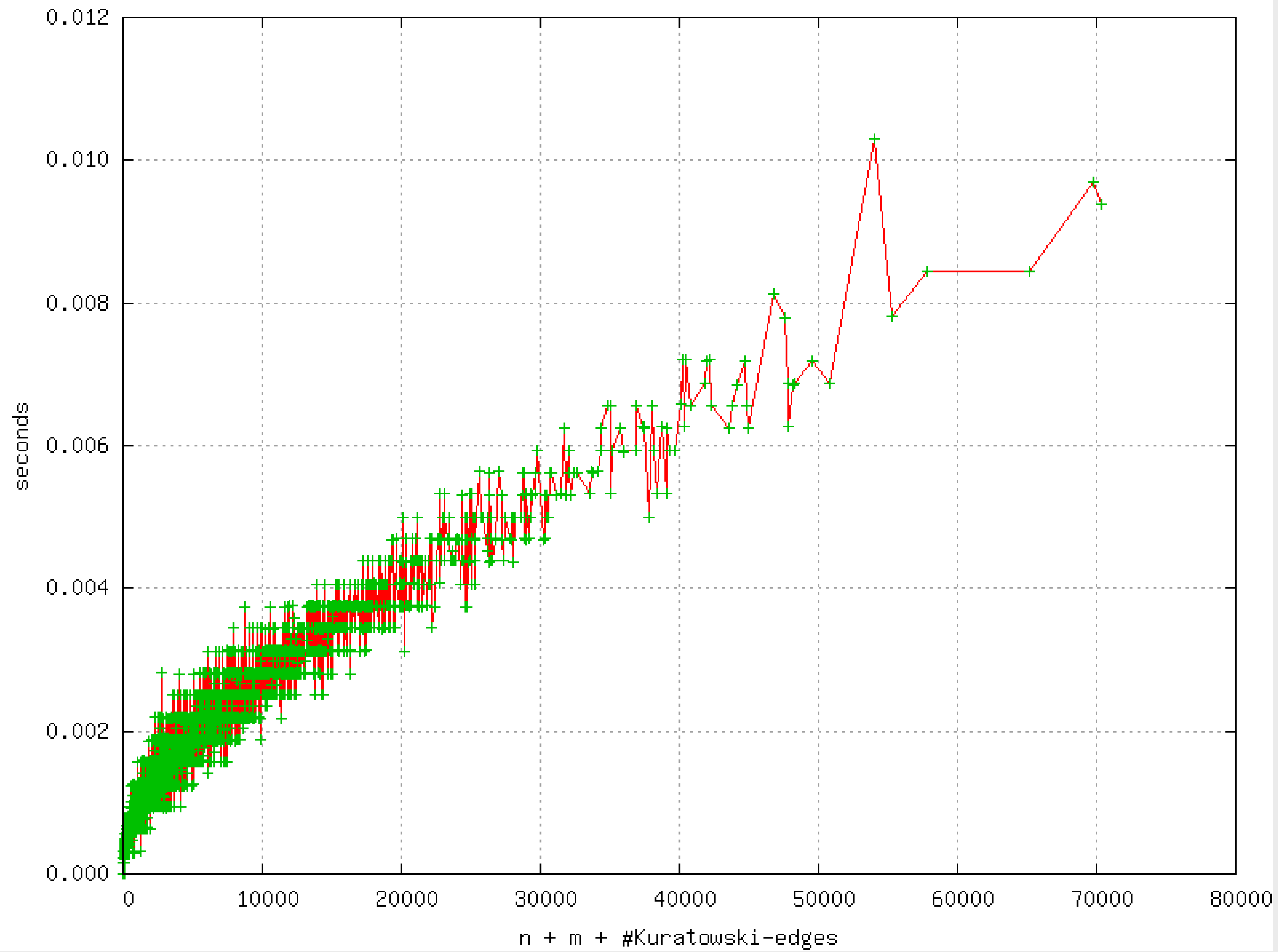
Random Graphs



Rome Library



Rome Library



The End