

Tree-Decomposition based Algorithms for Abstract Argumentation Frameworks

Masterstudium:
Software Engineering & Internet Computing

Günther Charwat

Technische Universität Wien
Institut für Informationssysteme
Arbeitsbereich: Datenbanken und Künstliche Intelligenz
Betreuer: Privatdoz. Dipl.-Ing. Dr.techn. Stefan Woltran
Mitwirkung: Dipl.-Ing. Wolfgang Dvořák

Motivation

Overview

- Argumentation Frameworks (AFs) important research field in Artificial Intelligence
- Selection of 'appropriate' arguments from AF defined by a semantics
- Many different semantics
- Selection oftentimes computationally hard (intractable)
 - Identify tractable fragments
 - For AFs: Tree-Width, defined on Tree Decompositions

Goal

- Development of novel algorithms for stable, complete and admissible semantics, following up the work of Dvořák et.al. [1]
- Based on Tree Decompositions, tractable
- Evaluation of different types of Tree Decompositions

Basic Definitions

Argumentation Framework [2]

Pair $F = (A, R)$ where A is a set of arguments and $R \subseteq A \times A$ is the attack relation.

Tree Decomposition [3]

Tree where each node has a bag that contains a set of vertices from the original graph such that:

- every vertex in at least one bag
- connected vertices together in bag
- nodes containing a vertex are connected upwards the tree.

Admissible Semantics [2]

Set S admissible if conflict-free and each argument in S is defended.

Stable Semantics [2]

Set S stable if conflict-free and each argument not in S is attacked by S .

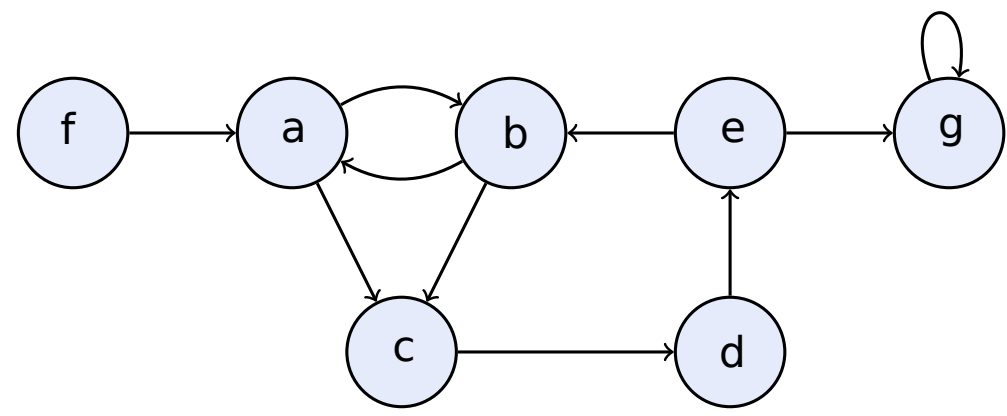
Complete Semantics [2]

Set S complete if admissible and each defended argument is in S .

Approach for Algorithms based on Tree Decompositions

Preparation

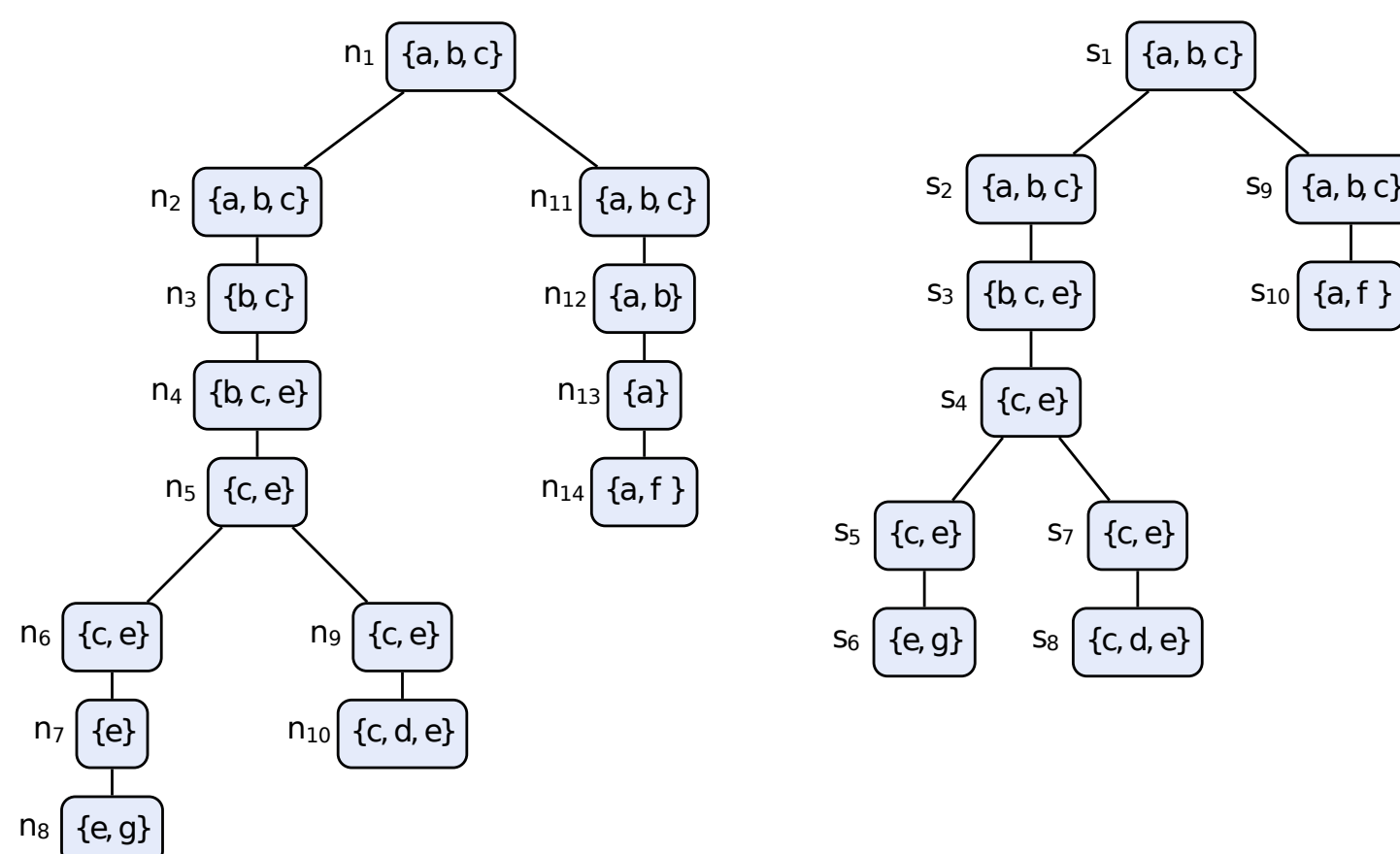
Input Instance: Argumentation Framework



Example Argumentation Framework

Obtain Tree Decomposition

- Makes use of heuristics (finding decomposition of minimal width itself intractable)
- Handled by an existing purpose-built framework
- Either normalized or semi-normalized
 - Semi-norm contains less nodes
 - Several arguments introduced or removed in one node
 - Branch node the same



Tree Decomposition:
Normalized (left), Semi-normalized (right)

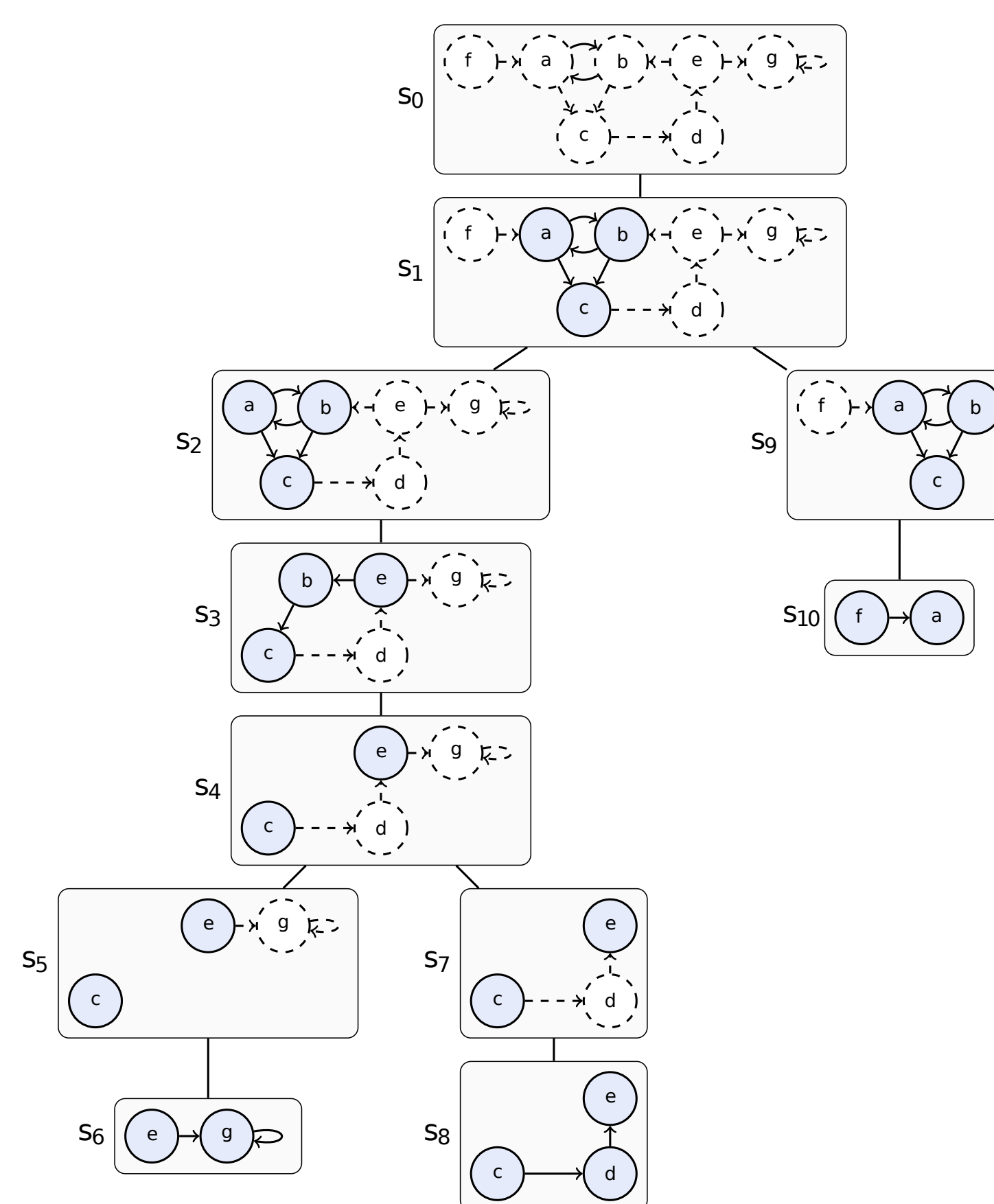
Computation

Traverse tree in bottom-up order

- Use information about vertices in current bag and colorings of vertices in the sub-tree

Restricted Sets: Contain arguments that were completely considered in the sub-tree of a node, arguments fulfill properties of semantics

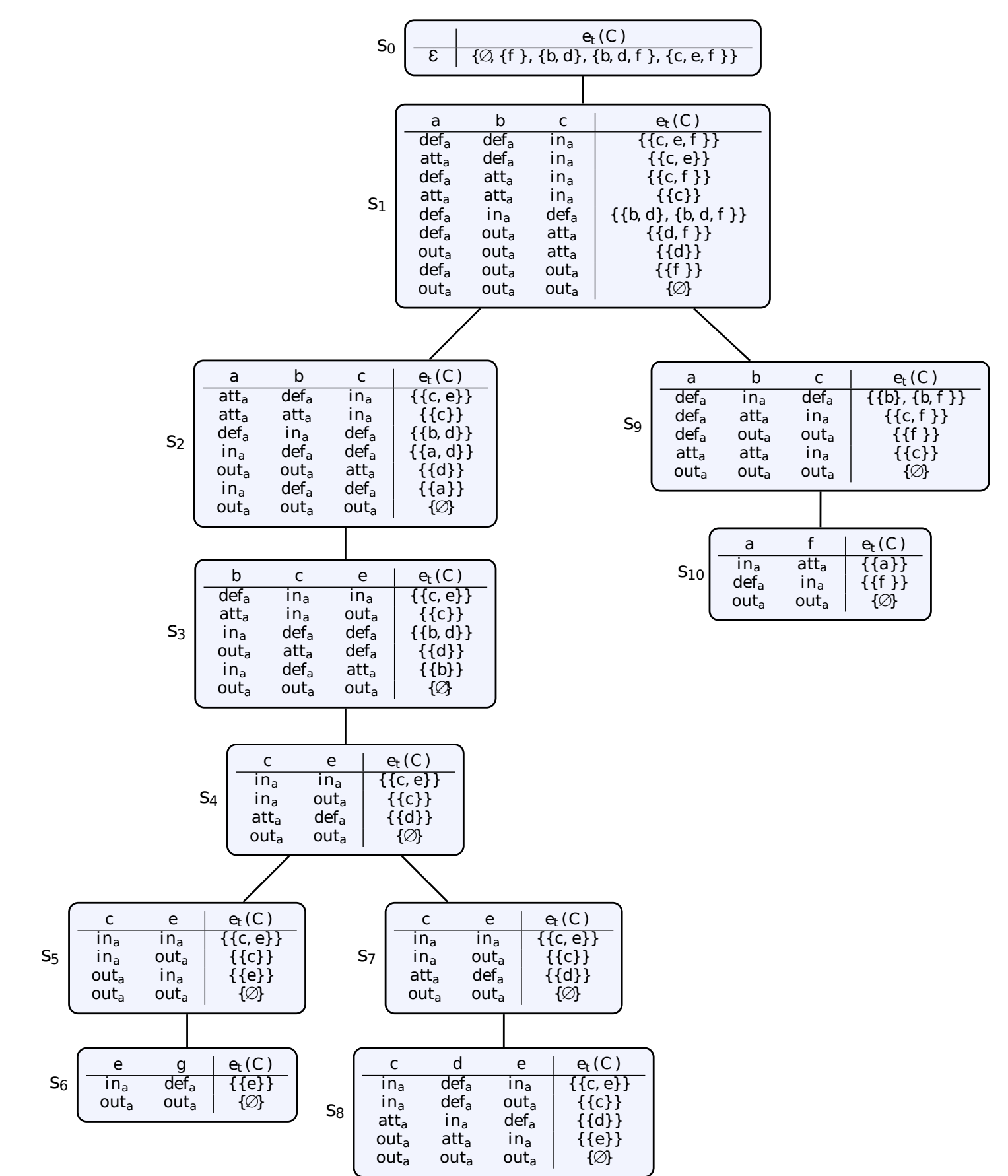
Colorings: Defined on restricted sets
Encode information about relations by assigning colors to each argument of current node



Semi-normalized Tree Decomposition with Sub-Frameworks

V-Colorings: Defined solely on current vertices and colorings of child-node, not on restricted sets

→ Fixed-parameter tractability achieved



Semi-normalized Tree Decomposition with V-Colorings for Admissible Semantics

Result Delivery

- At root node all arguments have been handled
- Restricted sets correspond to extensions of AF
- Computation in $f(tw) \cdot n^{O(1)}$

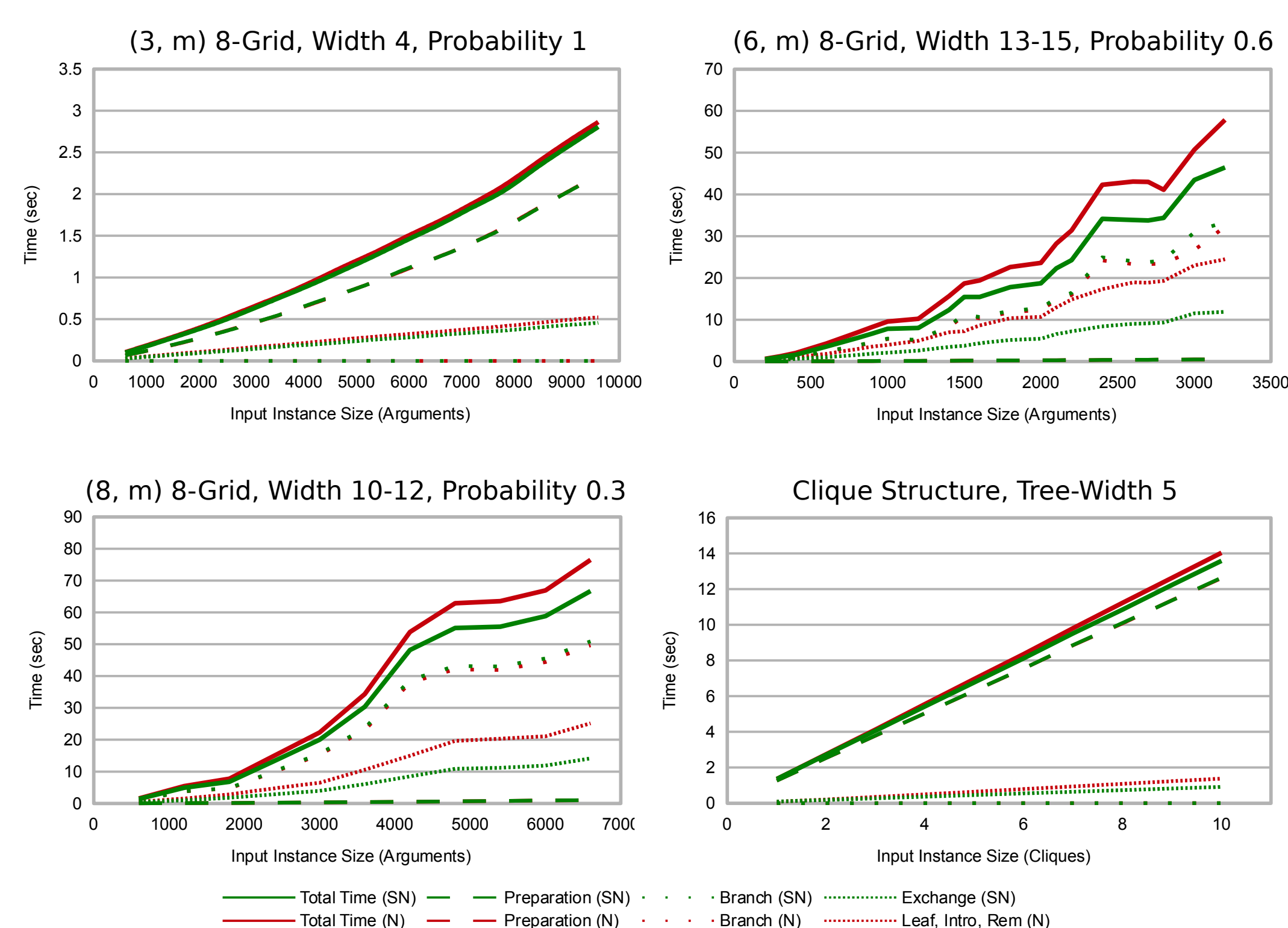
Experimental Results

Test Setup

- Comparison of already existing algorithm for admissible semantics on normalized tree decomposition [1] to novel on semi-normalization
- Different test instance types (Grid, Clique)
- Different width and edge probability

Analysis of Benchmarks

- Semi-normalized implementation outperforms normalized in every test case
- Relative performance gain significant (up to 50%)
- Absolute performance gain depends on cost for preparation and branch node evaluation
- In general: Less edges, performance gain more significant



Admissible Semantics: Normalized (N) vs. Semi-normalized (SN)

Conclusion

Contributions

- Novel algorithms for stable and complete semantics based on normalized tree decompositions
- Novel algorithm for admissible semantics based on semi-normalized tree decompositions
- Implementations and correctness proofs of the algorithms

- Experimental results show that algorithm on semi-normalized tree decompositions outperforms the existing one

Future Work

- Provide algorithms for further semantics based on tree decompositions
- Analysis of run-time on non-normalized tree decompositions
- Detailed complexity analysis