Simplified Computation and Generalization of the Refined Process Structure Tree

Artem Polyvyanyy, Jussi Vanhatalo, and Hagen Völzer
Outline

1. Motivation
2. The Refined Process Structure Tree
3. Simplified Computation
4. Generalization
5. Conclusion
BPM: Bridging the Gap between Business and IT

- Align business and IT by aligning business process models and executable processes

Business Process Model and Notation (BPMN)  Business Process Execution Language (BPEL)

© Artem Polyvyanyy, Jussi Vanhatalo, Hagen Völzer, and IBM Corporation, 2010
Translate BPMN to BPEL Using the Refined Process Structure Tree

Refined Process Structure Tree (RPST)

Business Process Model and Notation (BPMN)

Business Process Execution Language (BPEL)

[BBK08]

© Artem Polyvyanyy, Jussi Vanhatalo, Hagen Völzer, and IBM Corporation, 2010
Applications for the Refined Process Structure Tree

- Translation between process languages
- **Control-flow** and data-flow analysis [FV10]
- Process comparison and merging
- Process abstraction
- Process comprehension
- Model layout
- Pattern application in process modeling
- **Process refactoring** [PGD10]
- **Reuse** [DB10]

© Artem Polyvyanyy, Jussi Vanhatalo, Hagen Völzer, and IBM Corporation, 2010
Limitations of the Previous Work

• Algorithm is fairly complex to implement
• Technique is restricted to a subset of directed graphs
  – That have a single source and a single sink such that adding an edge from the sink to the source makes the graph biconnected
Contributions

- **Simplified algorithm**
  - Linear time complexity
- **Generalization of the RPST**
  - Applicable to all directed graphs, where every node is in a path from a source to a sink
Outline

1. Motivation
2. The Refined Process Structure Tree
3. Simplified Computation
4. Generalization
5. Conclusion
The Refined Process Structure Tree

*Fragment* is a connected subgraph that has a single entry node and single exit node

The *RPST* is a hierarchy of all fragments that
  - do not overlap any fragment

Unique, modular, and computed in linear time
Computation of the RPST

- Triconnected component (TCC)
- Linear time algorithm
  - [VVK08], [Van09]
Outline

1. Motivation
2. The Refined Process Structure Tree (RPST)
3. Simplified Computation
4. Generalization
5. Conclusion
Simplified Algorithm for Normalized Graphs

• A graph is *normalized* if every node has at most one incoming or at most one outgoing edge

• Theorem 1 – For a normalized graph the RPST coincides with the tree of triconnected components

• Linear time algorithm
Not Applicable in General

- A fragment of the RPST is not necessarily a triconnected component
- A triconnected component is not necessarily a fragment of the RPST

[VVK08]

Fragments of the RPST

Triconnected components
Node Splitting

- A graph is *normalized* if every node has at most one incoming or at most one outgoing edge.
- What if we *split* the nodes that violate this restriction?
Let’s Split Nodes

1. Compute the Tree of the TCCs
2. Analyze the TCCs
3. Restructure the tree to the RPST

1. Split nodes
2. Compute the Tree of the TCCs
1. Compute the Tree of the TCCs
2. Analyze the TCCs
3. Restructure the tree to the RPST

1. Split nodes
2. Compute the Tree of the TCCs

4. Remove redundant fragments
3. Remove added edges
Outline

1. Motivation
2. The Refined Process Structure Tree (RPST)
3. Simplified Computation
4. Generalization
5. Conclusion
How to decompose the business process models below?

- Current restrictions
  - Completed graph is not biconnected
  - Multiple sources or multiple sinks
  - Disconnected graphs
Computation for Not Biconnected Graphs

1. Split nodes

2. Compute the Tree of the TCCs (the RPST)

3. Remove redundant fragments
Definition for Not Biconnected Graphs

1. Split nodes

2. Compute the Tree of the TCCs (the RPST)

3. Remove redundant fragments

4. Definition of the RPST
Examples for Not Biconnected Graphs

• Useful containment hierarchy

[TV80]
Computation for Graphs with Multiple Sources or Sinks

1. Add unique source and sink

2. Compute the RPST

3. Remove redundant fragments
Definition for Graphs with Multiple Sources or Sinks

1. Add unique source and sink
2. Compute the RPST
3. Remove redundant fragments
4. Definition of the RPST
Computation for Disconnected Graphs

1. Add unique source and sink

2. Compute the RPST

3. Remove redundant fragments
Definition for Disconnected Graphs

1. Add unique source and sink

2. Compute the RPST

3. Remove redundant fragments

4. Definition of the RPST
How to decompose the business process models below?

- These decompositions help us with the BPMN to BPEL translation
Outline

1. Motivation
2. The Refined Process Structure Tree (RPST)
3. Simplified Computation
4. Generalization
5. Conclusion
Conclusion

• **Simplified** linear time algorithm

• **Generalization** of the RPST
  
  – Applicable to all directed graphs, where every node is in a path from a source to a sink
Questions?