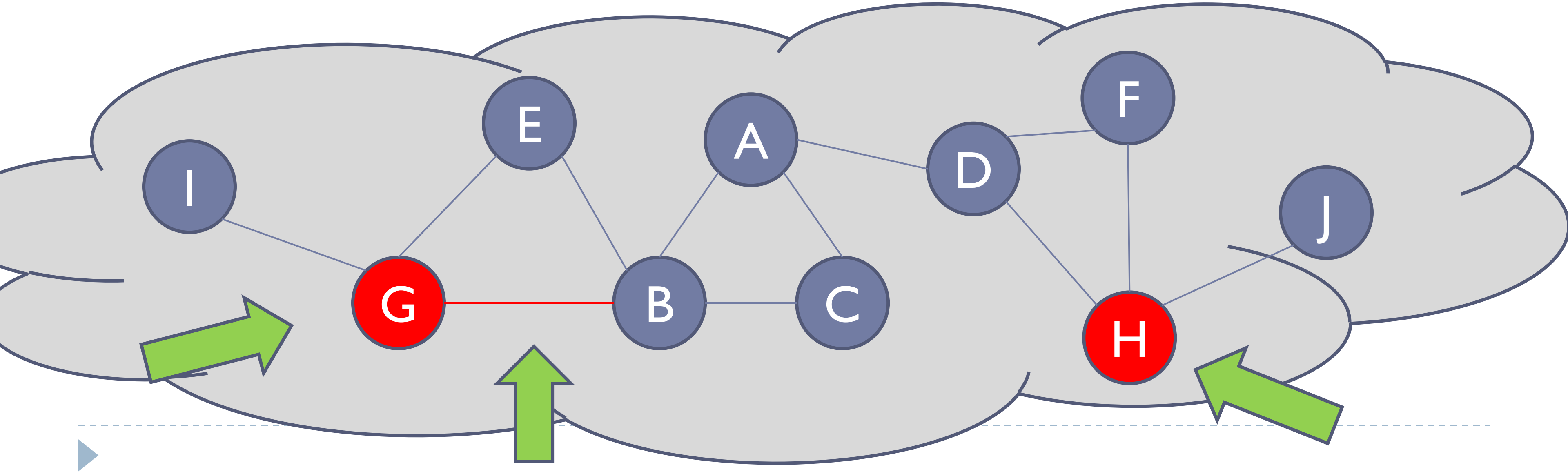
Using Substructure Mining to Identify Misbehavior in Network Provenance Graphs

David DeBoer, Georgetown University
Wenchao Zhou, Georgetown University
Lisa Singh, Georgetown University

June 23, 2013, GRADES Workshop, SIGMOD 2013
New York, NY

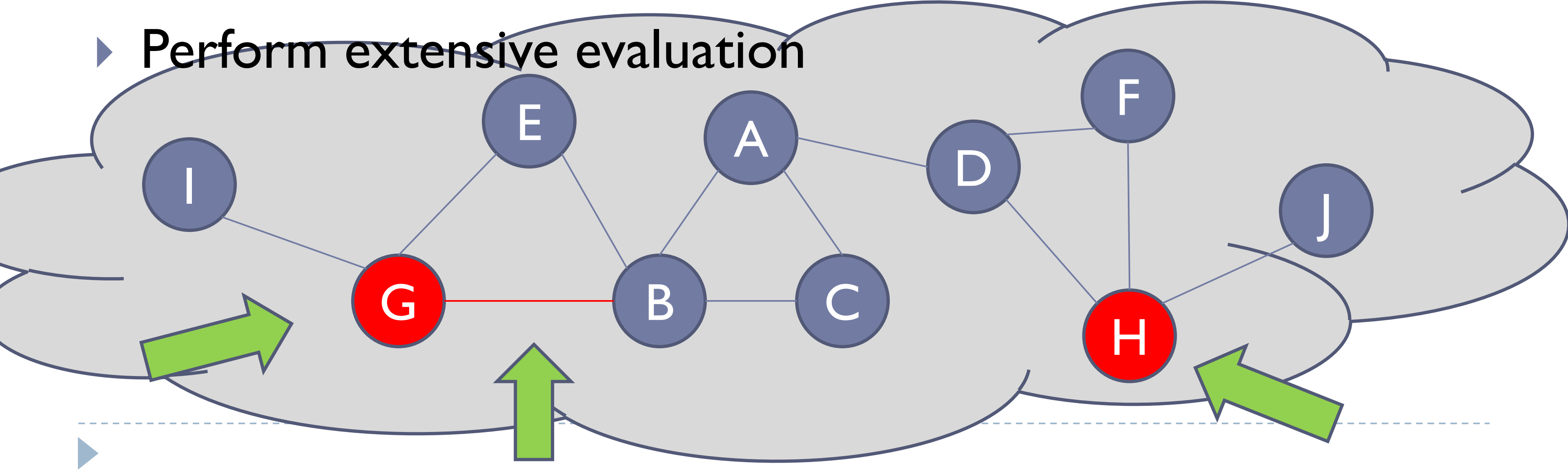
Distributed Systems

- Distributed systems have seen huge success
- They touch many parts of our daily lives
- Faults are costly
- Monitoring and maintenance is difficult
- Network Provenance is a proposed solution

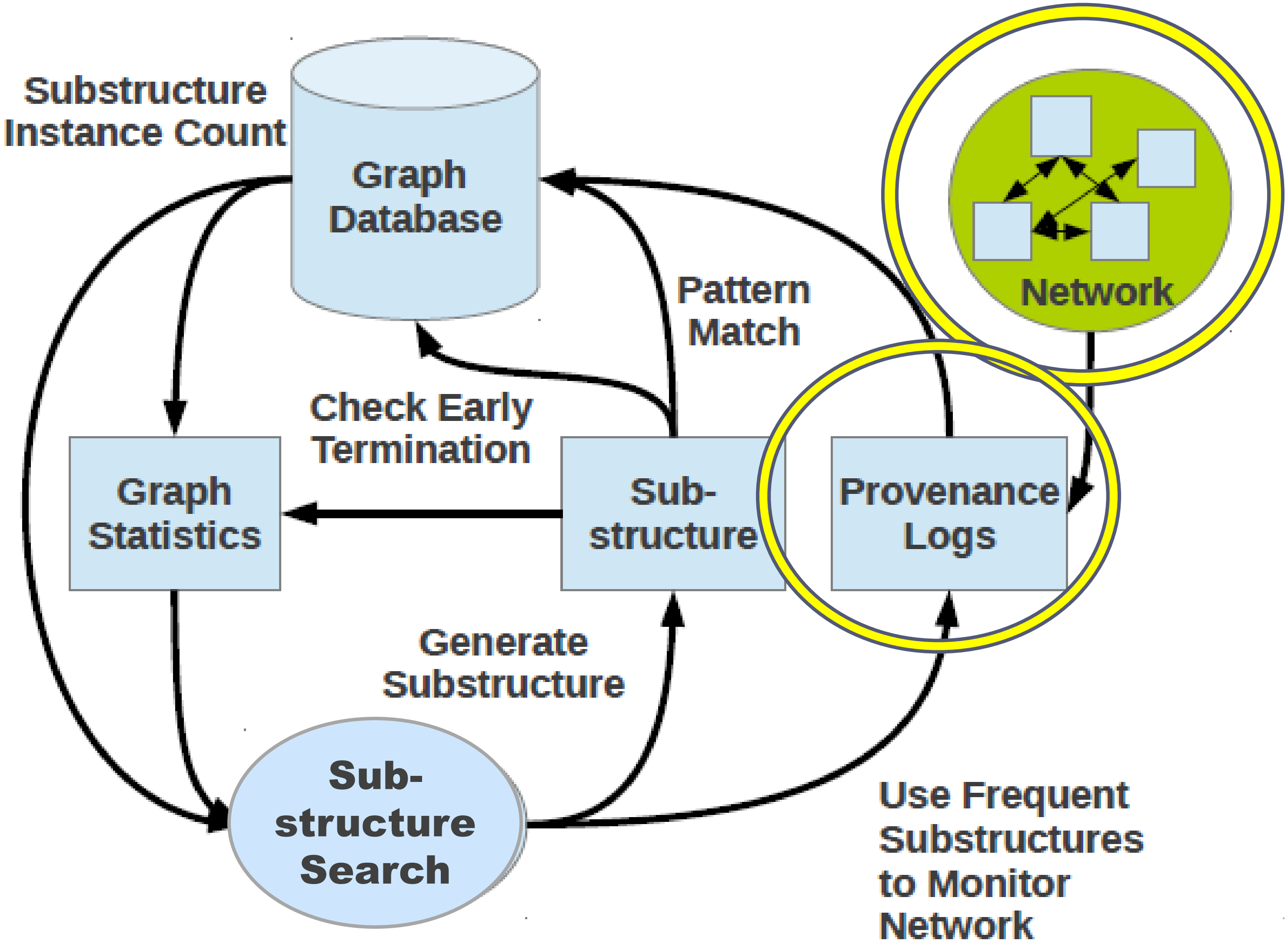


Our Contribution

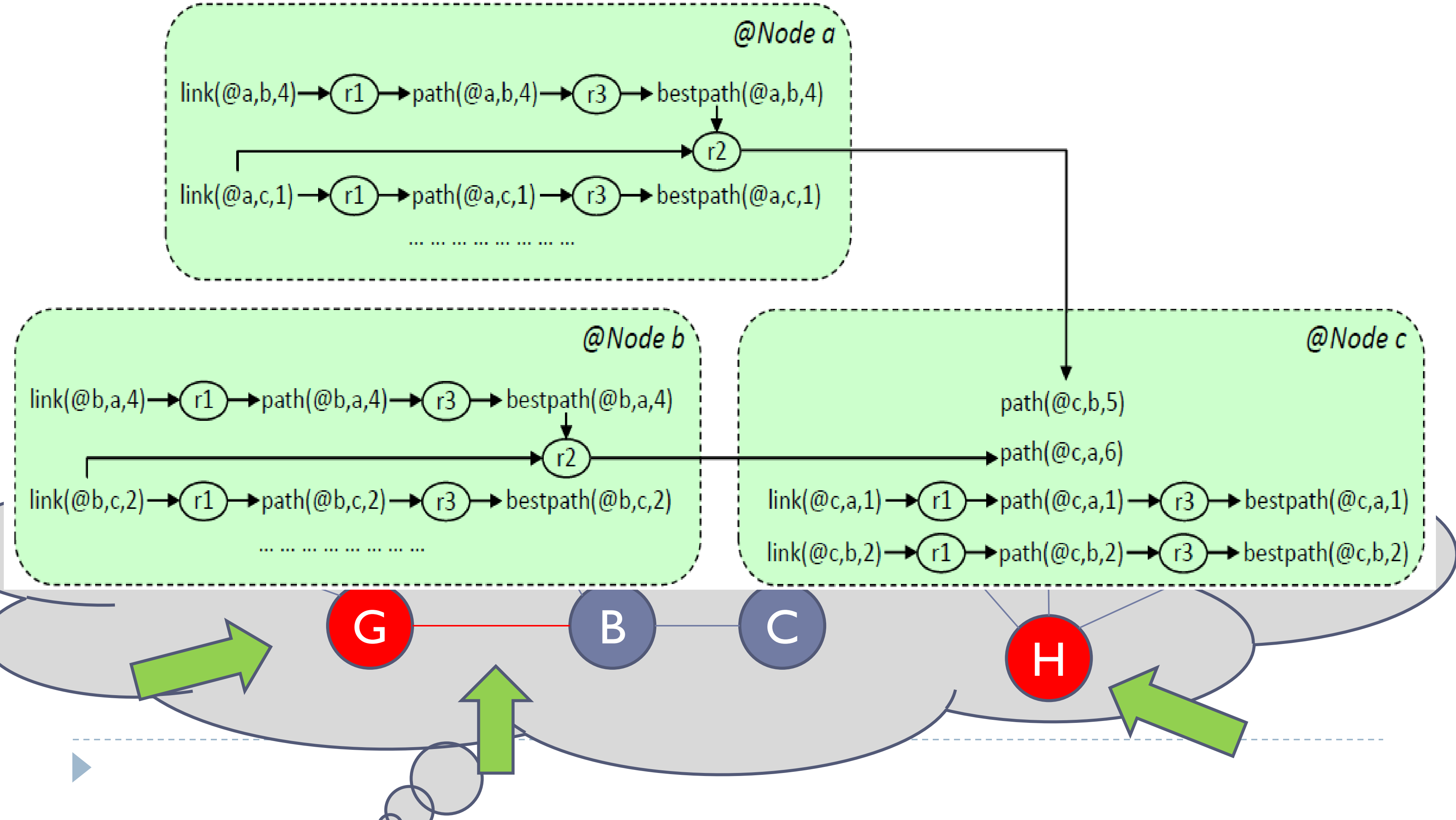
- Leverage the dependency graph of network provenance for a substructure mining application
- Find common execution patterns
- Use them as a feature set to identify misbehaving nodes
- Use heuristics to find substructures more quickly
- Implement with a graph database, neo4j
- Perform extensive evaluation



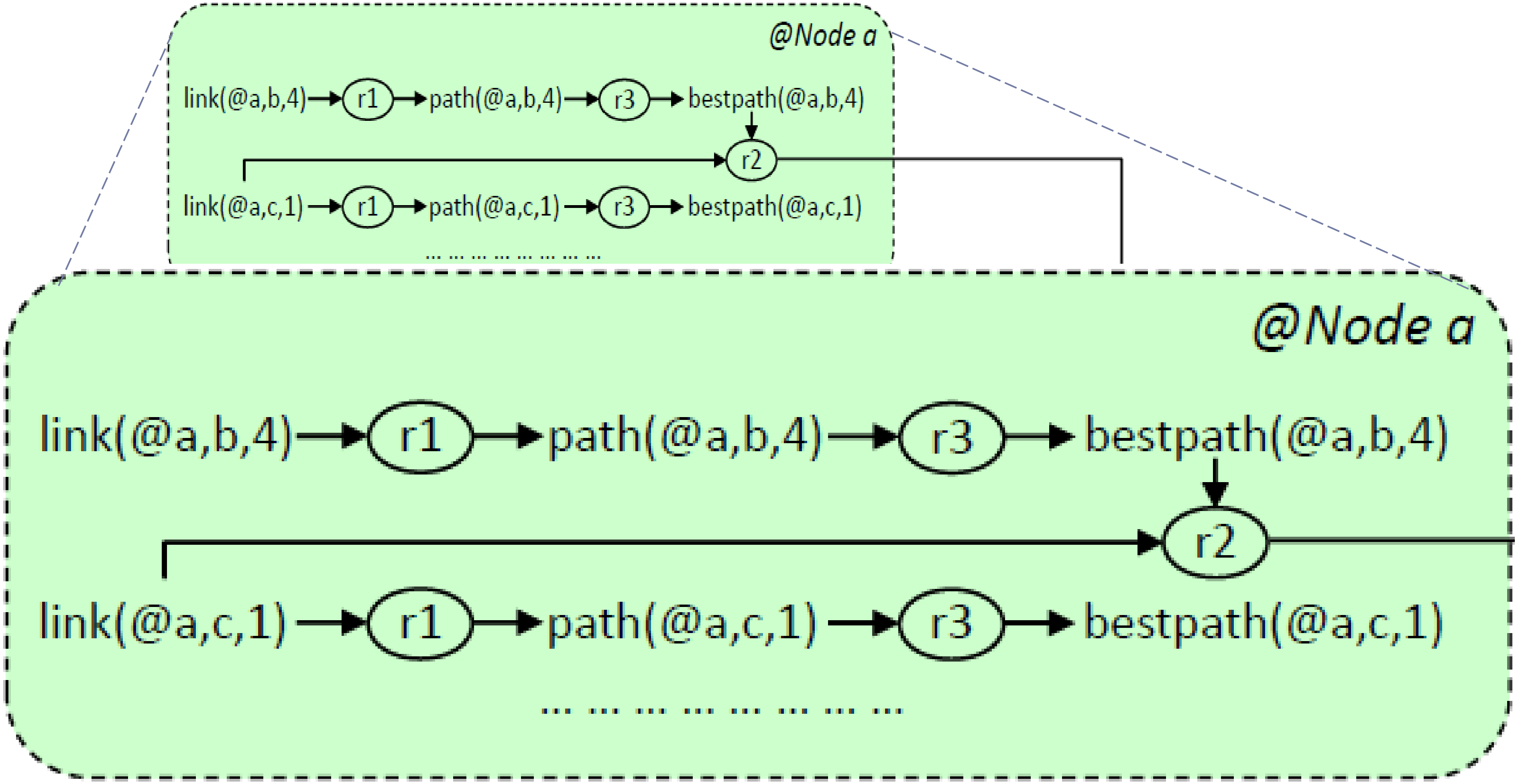
Proposed System Architecture



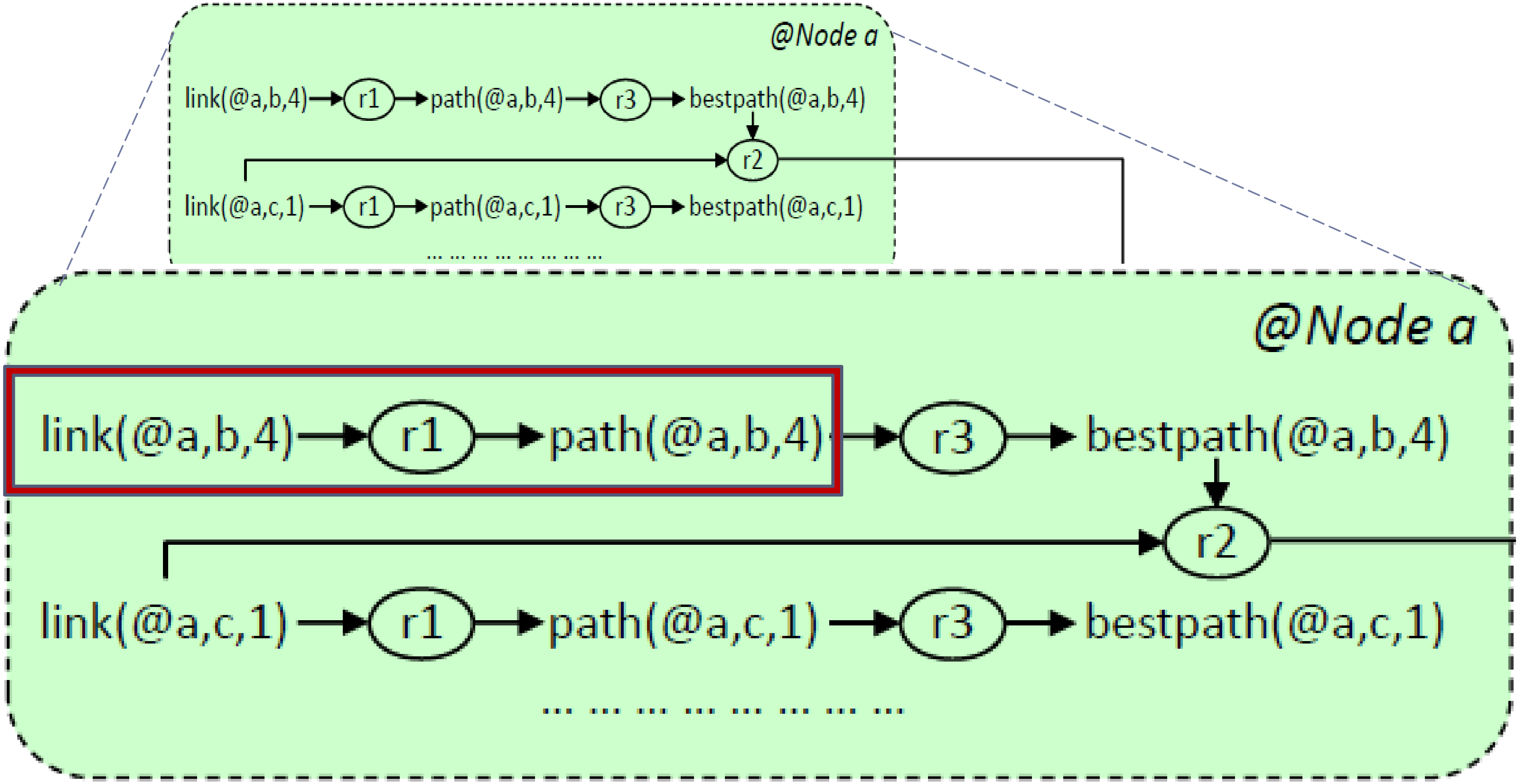
Example: Network Provenance



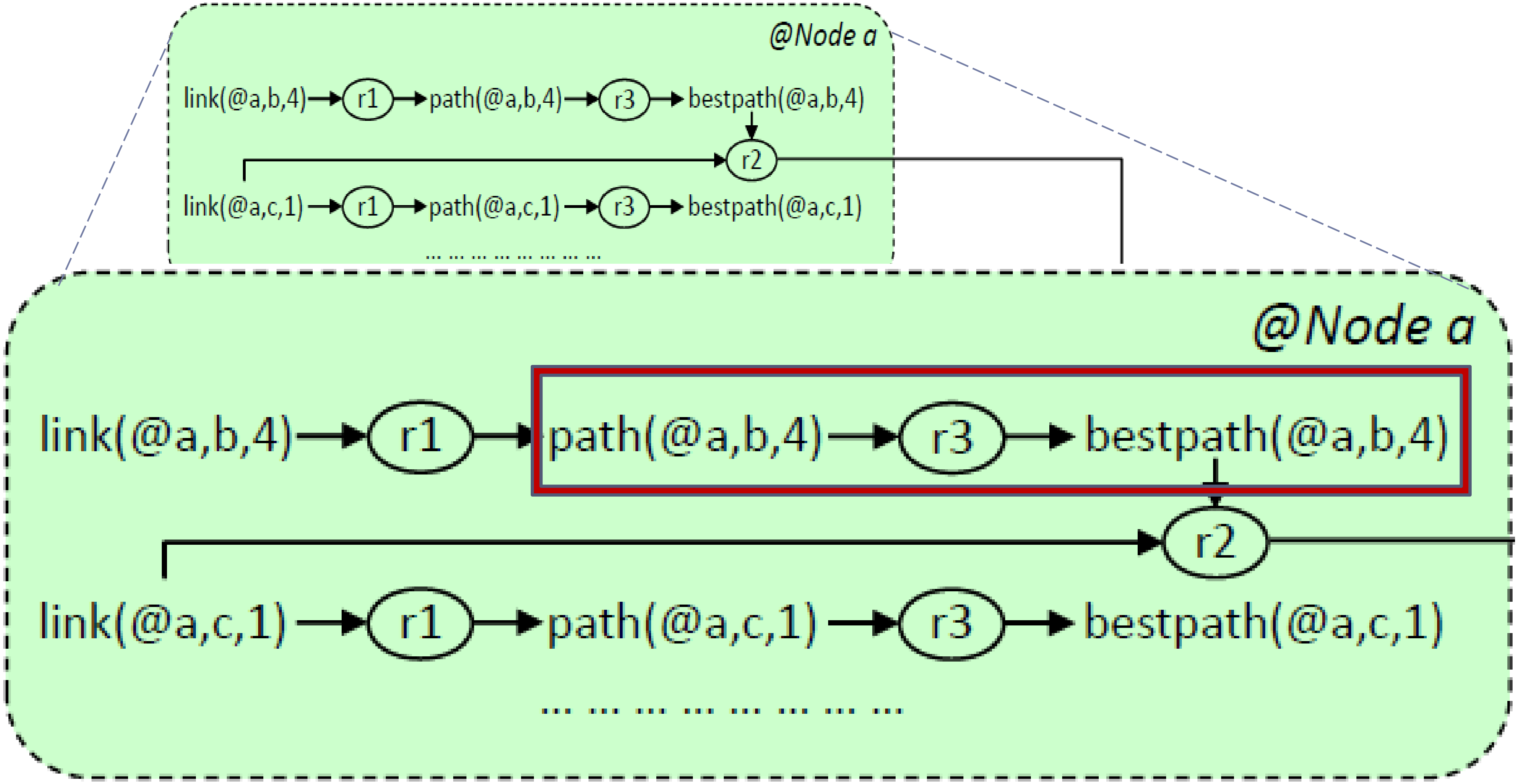
Example: Provenance Graph



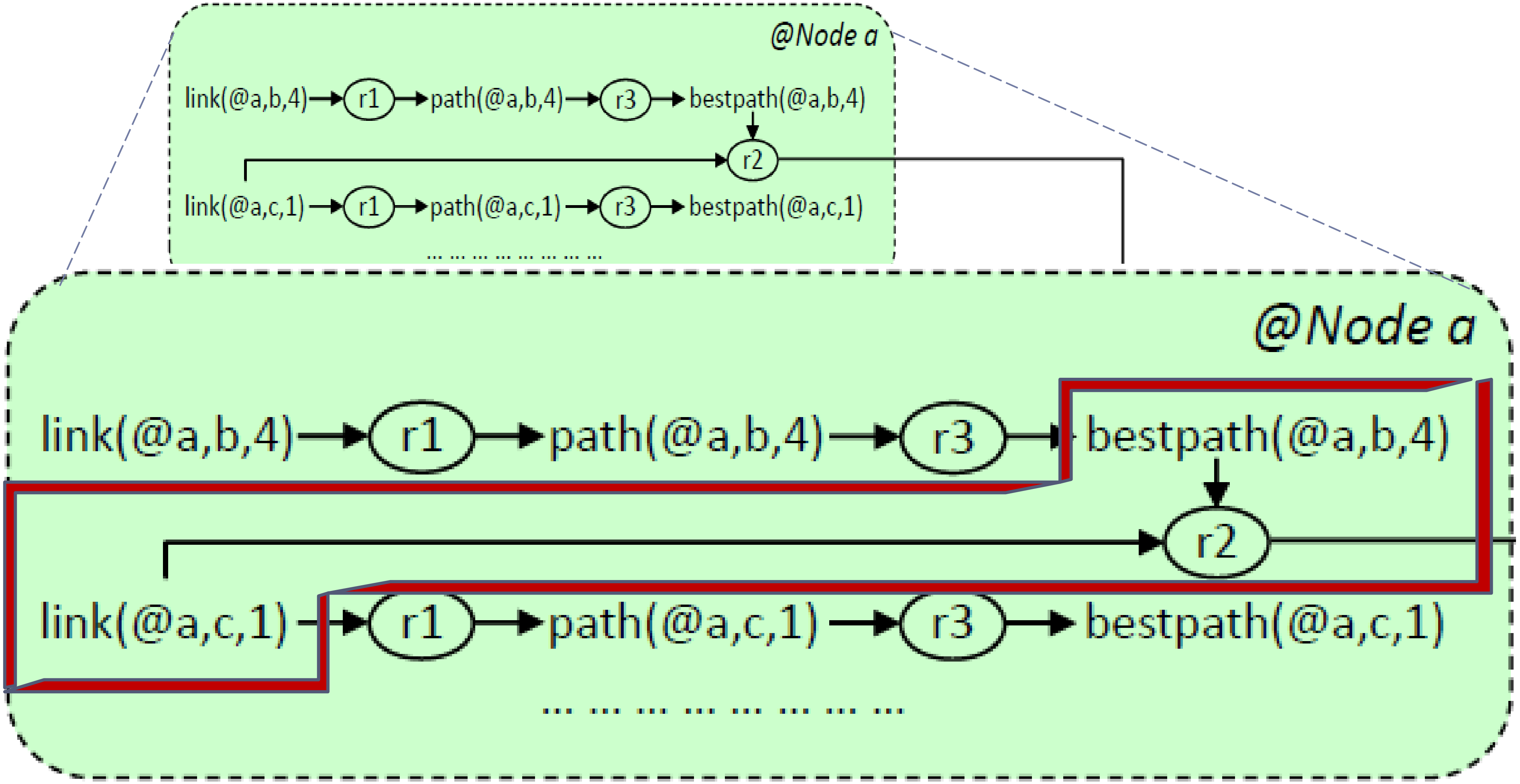
Example: Provenance Graph



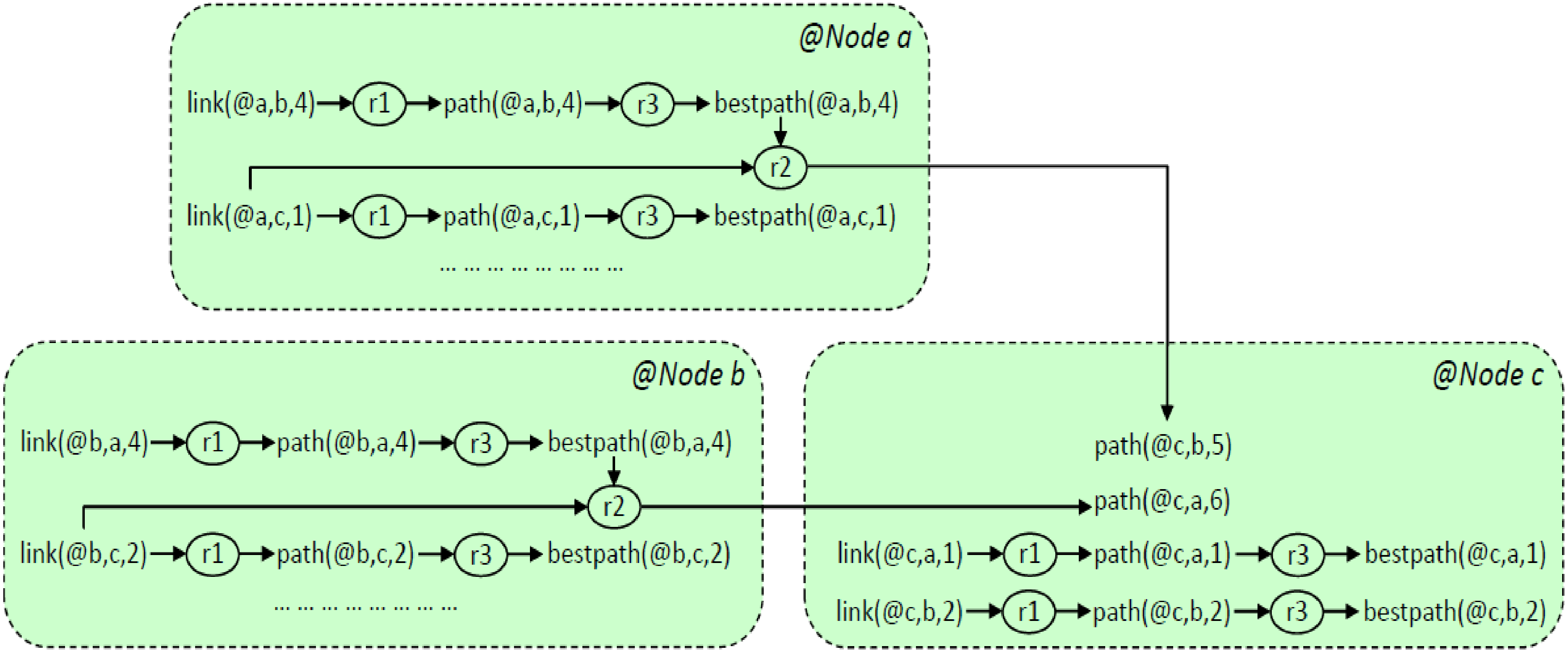
Example: Provenance Graph



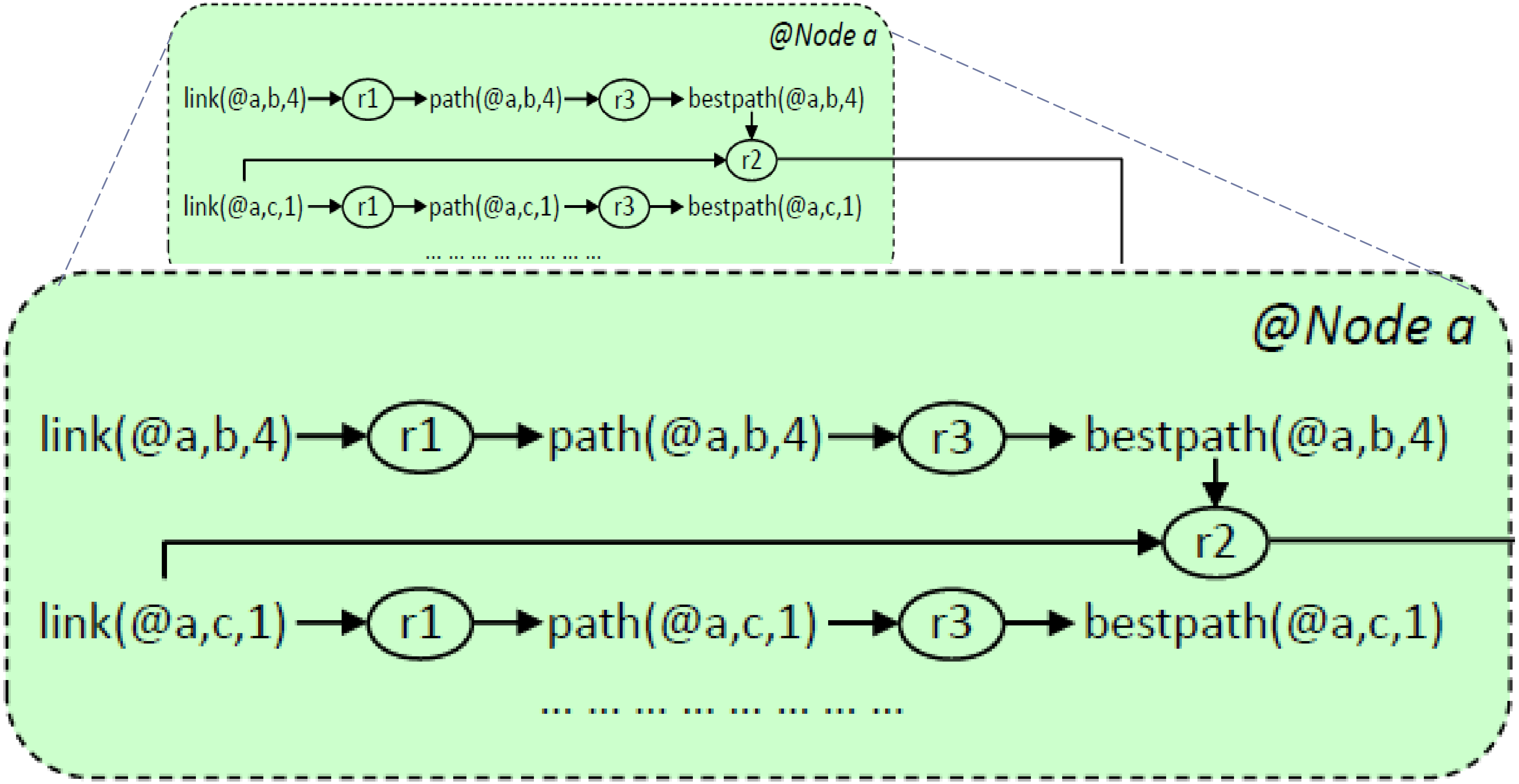
Example: Provenance Graph



Example: Provenance Graph

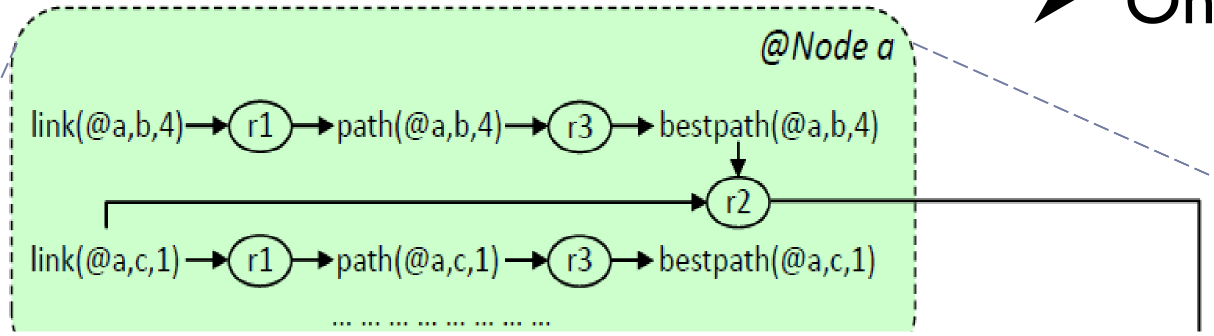


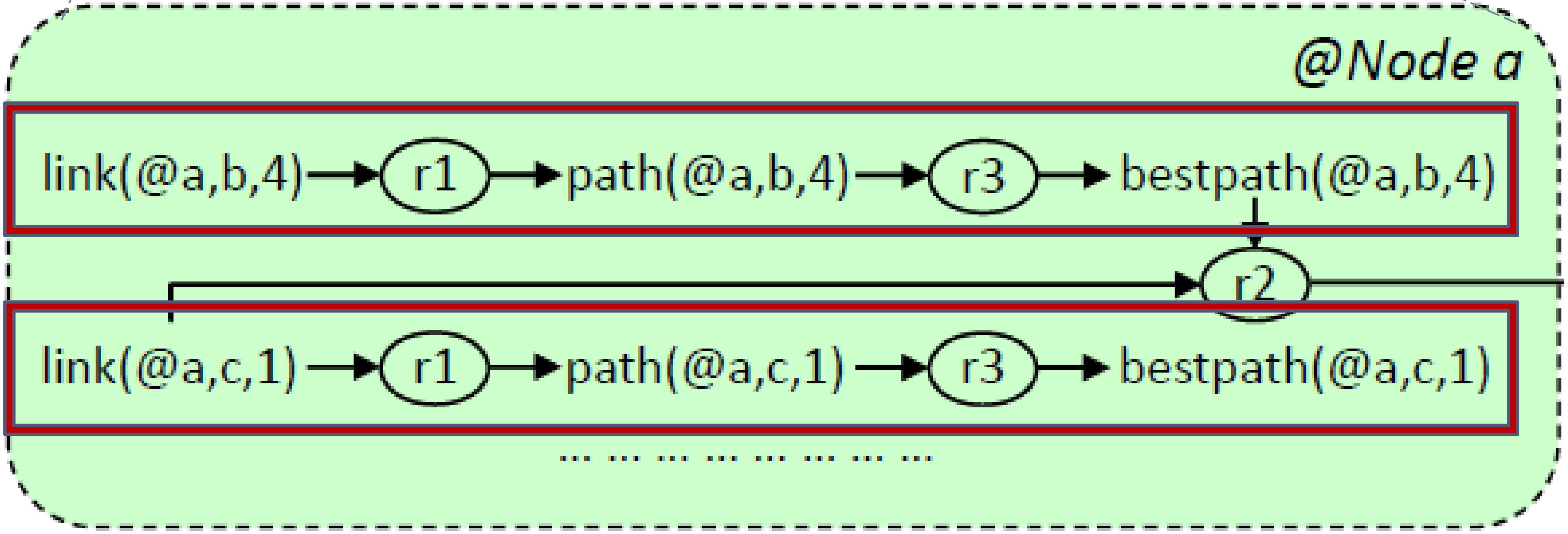
Example: Provenance Graph



Example: Provenance Graph

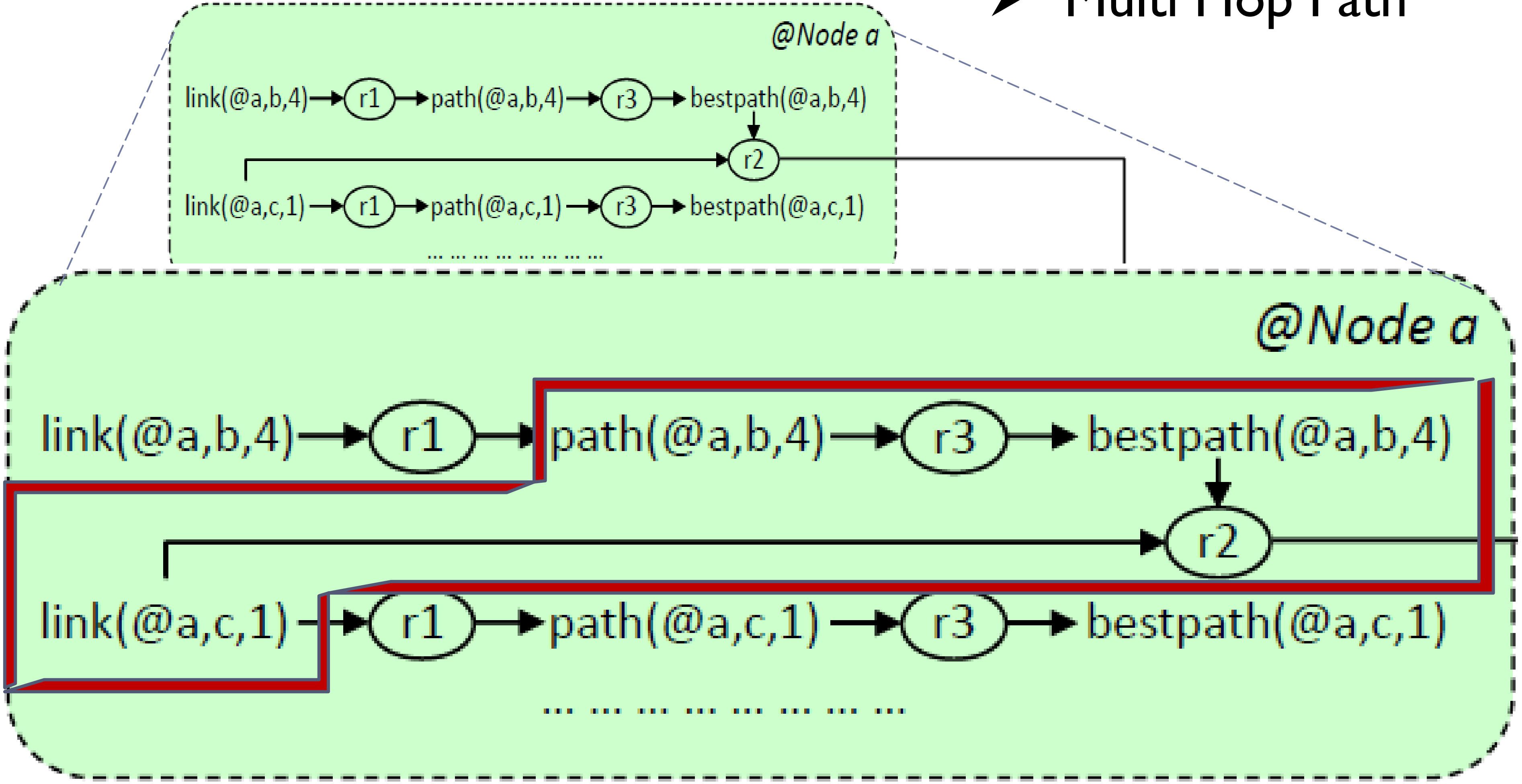
- One Hop Path



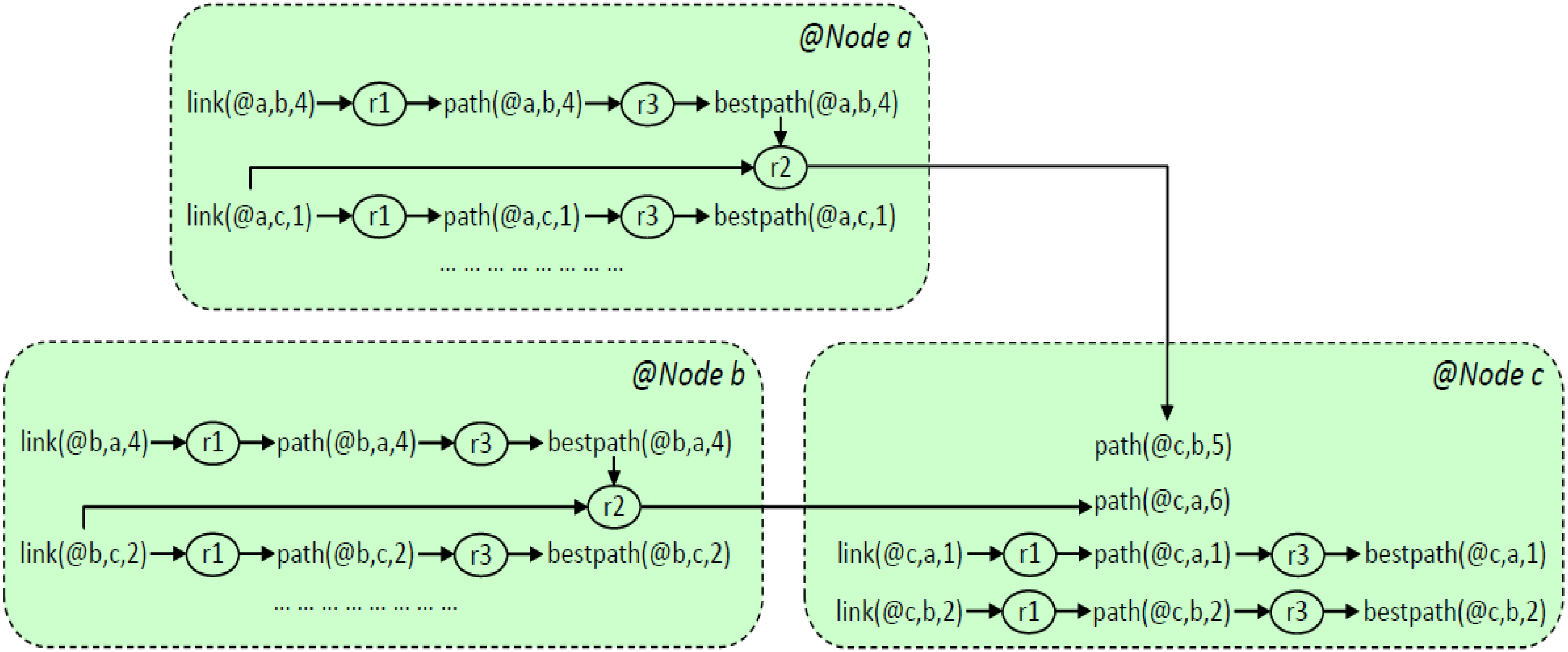


Example: Provenance Graph

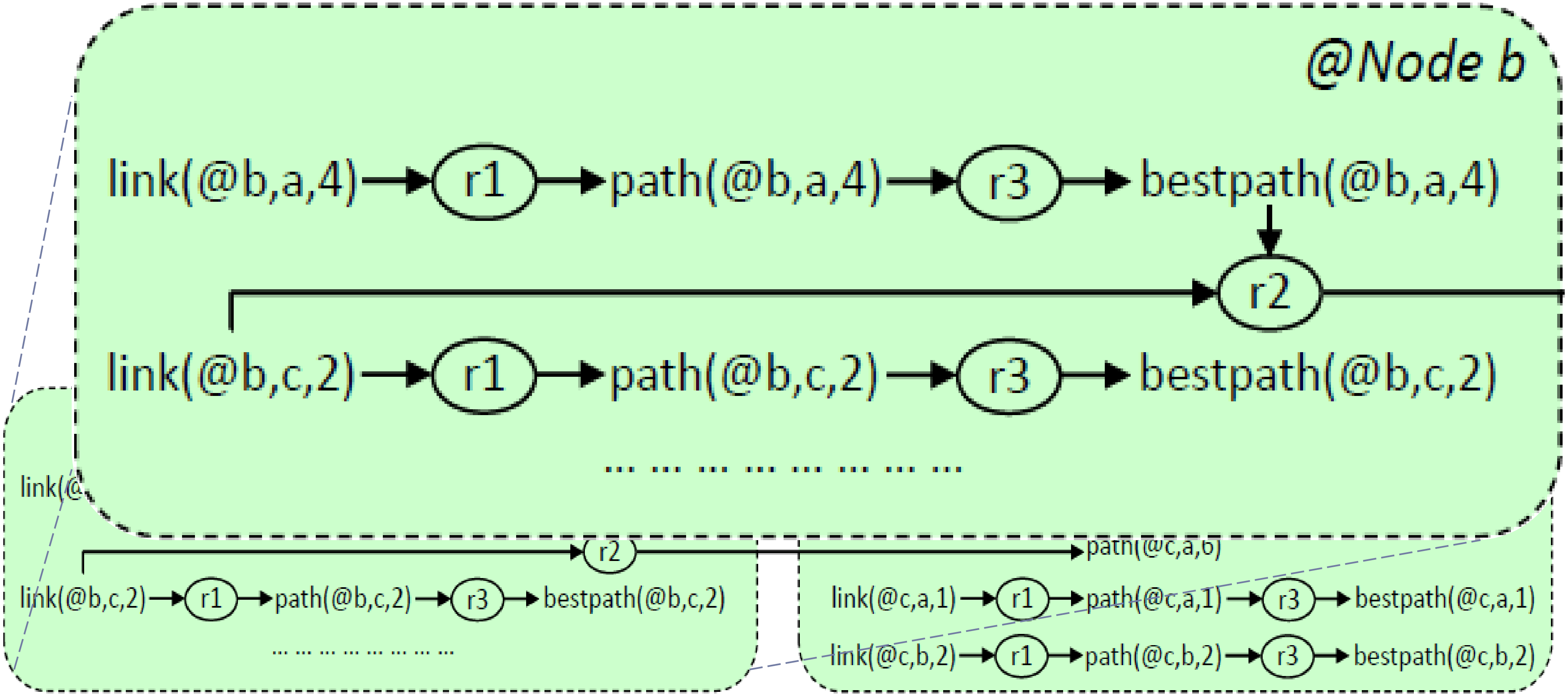
- Multi Hop Path



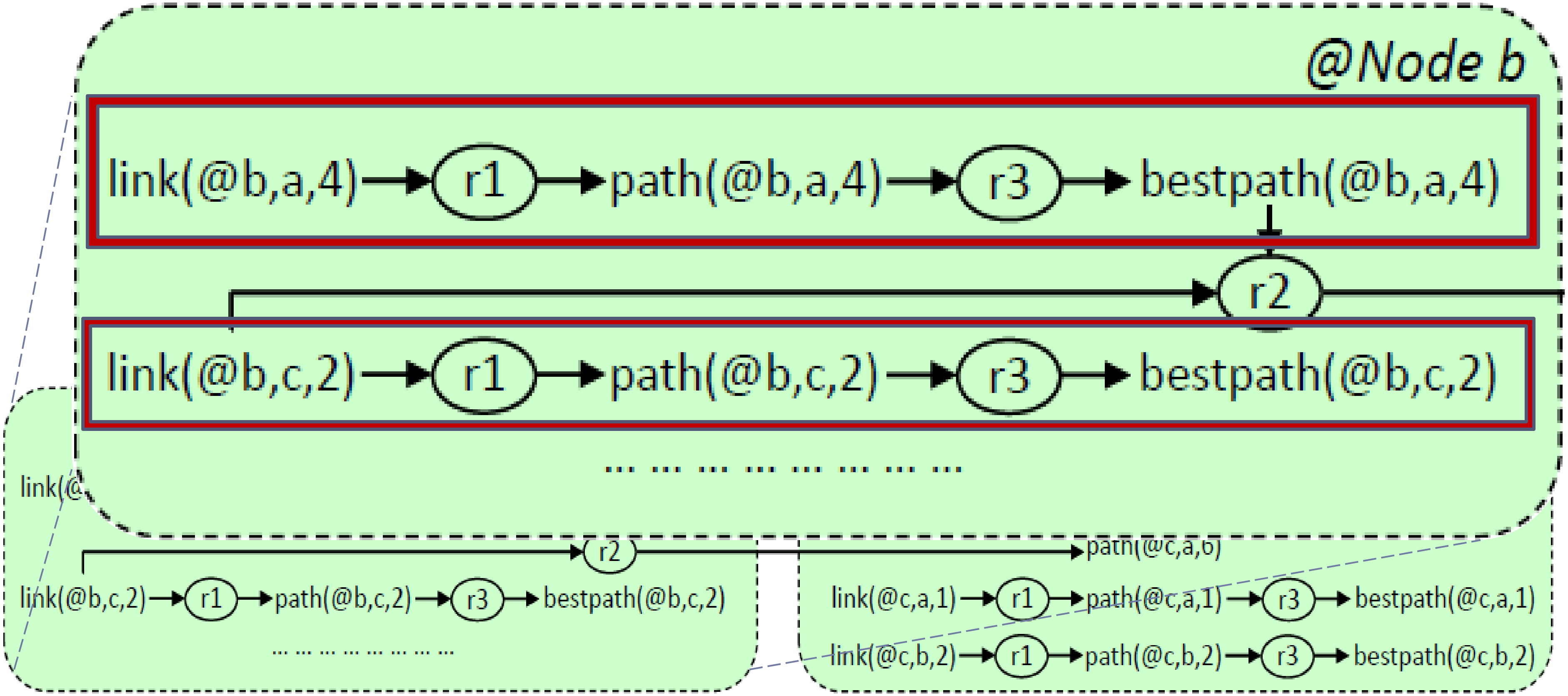
Example: Provenance Graph



Example: Provenance Graph

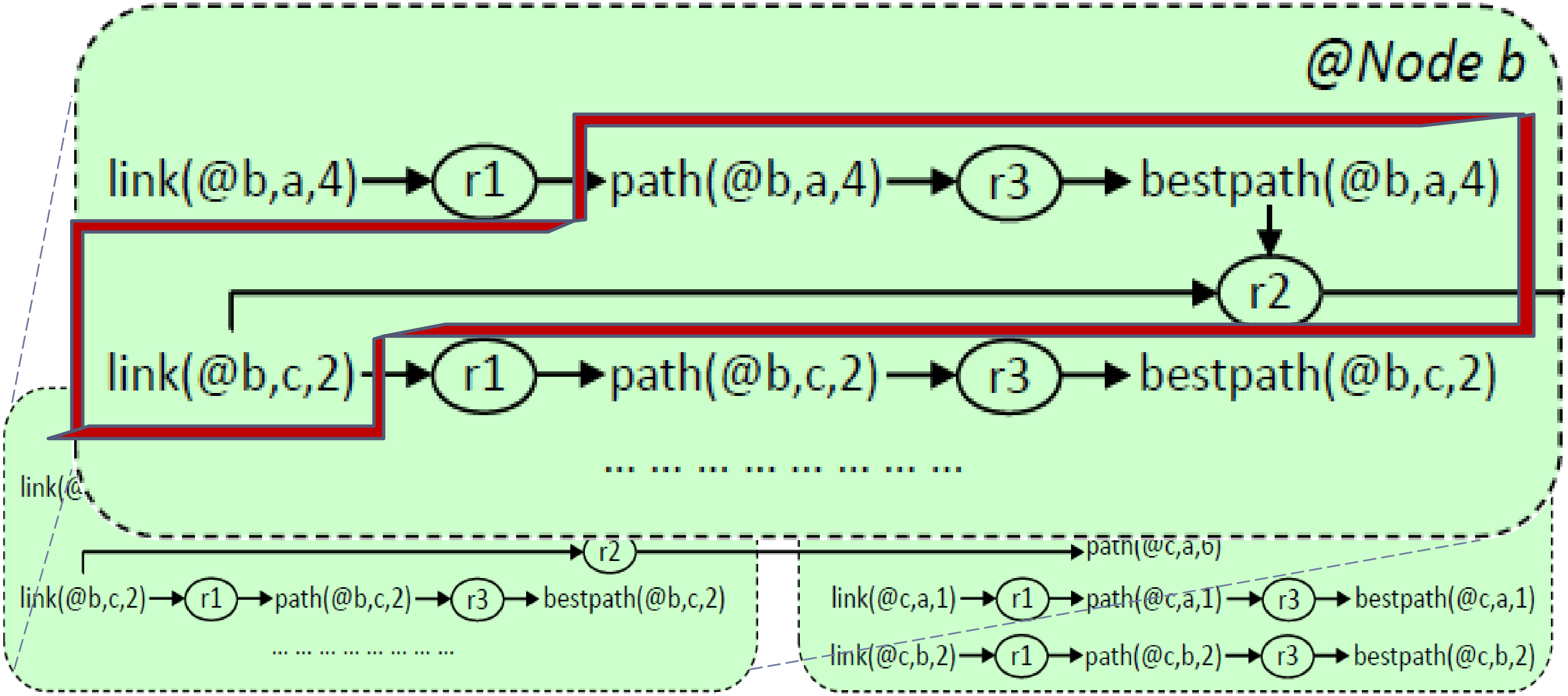


Example: Provenance Graph



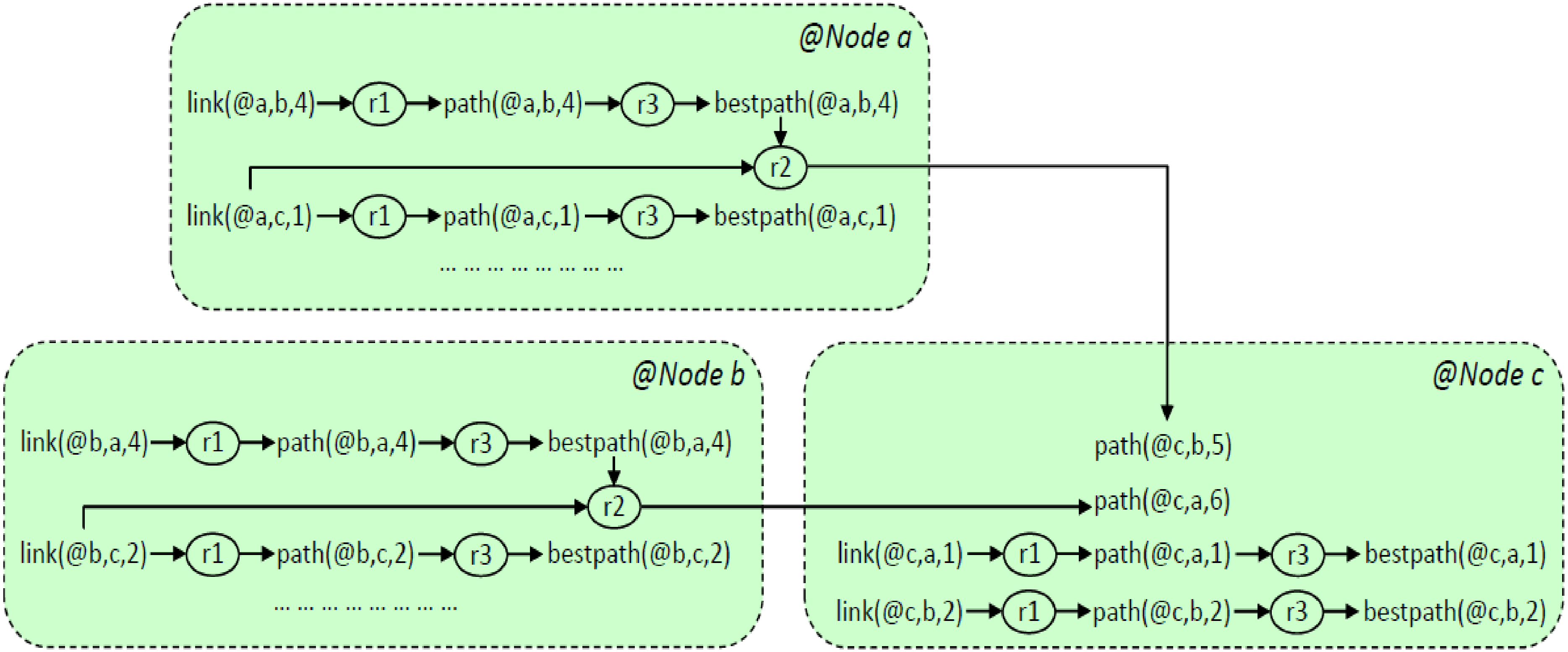
- One Hop Path

Example: Provenance Graph

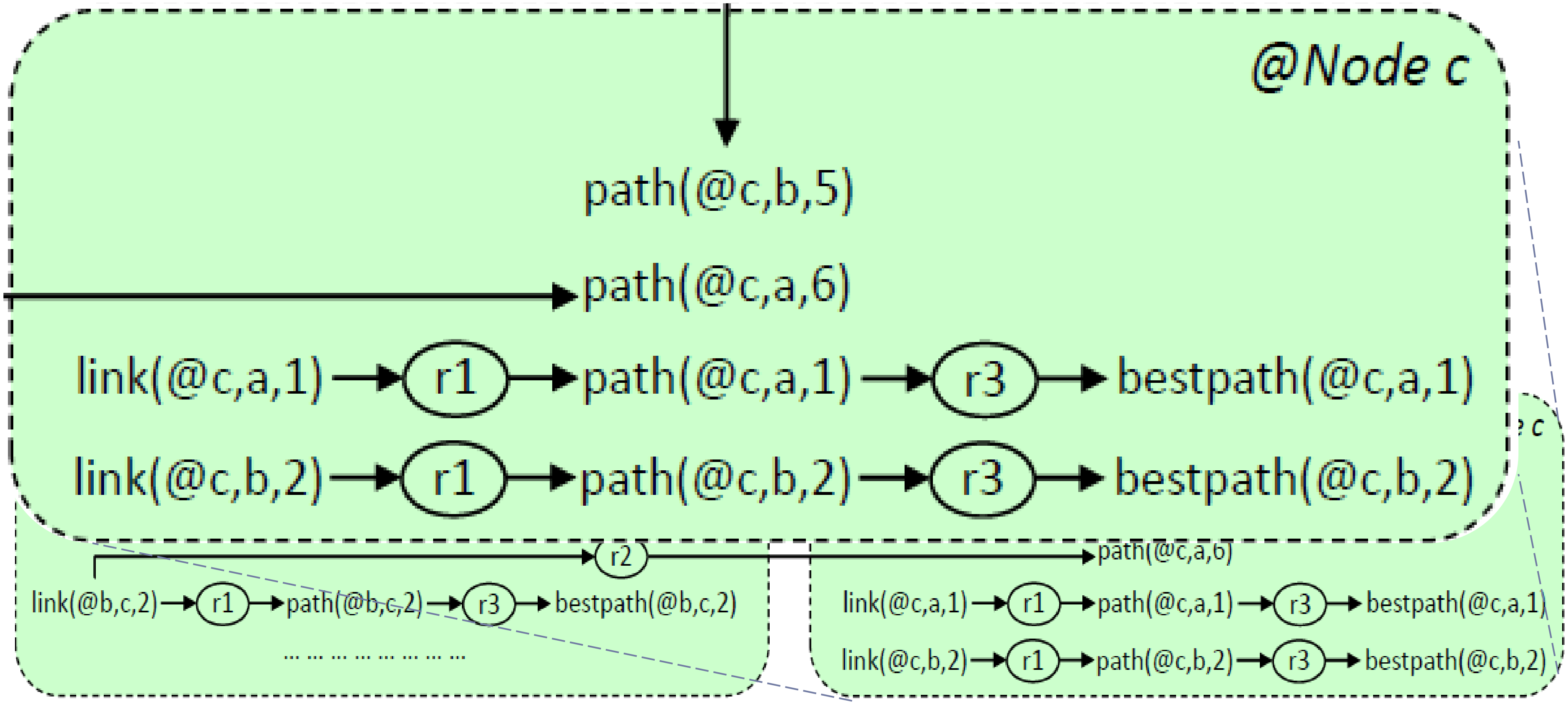


- Multi Hop Path

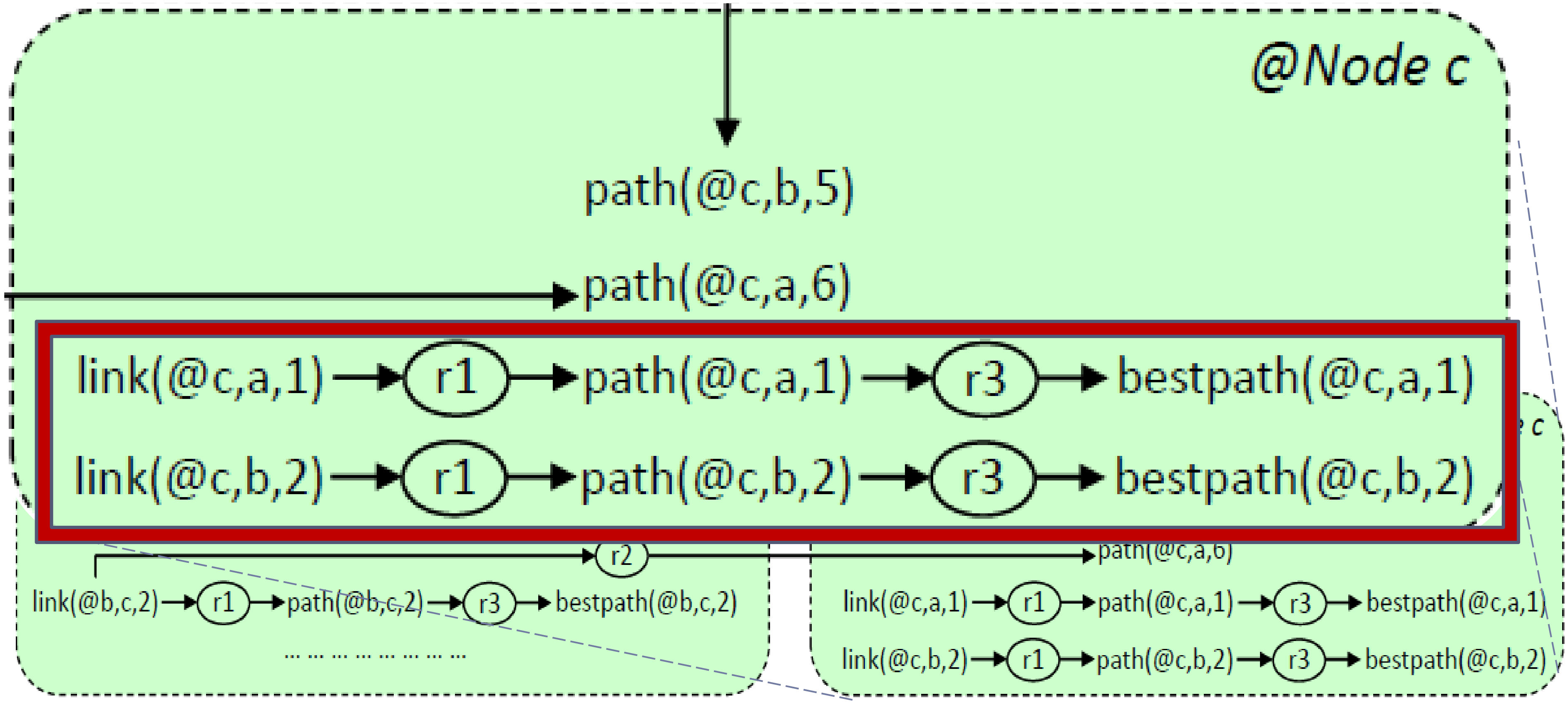
Example: Provenance Graph



Example: Provenance Graph

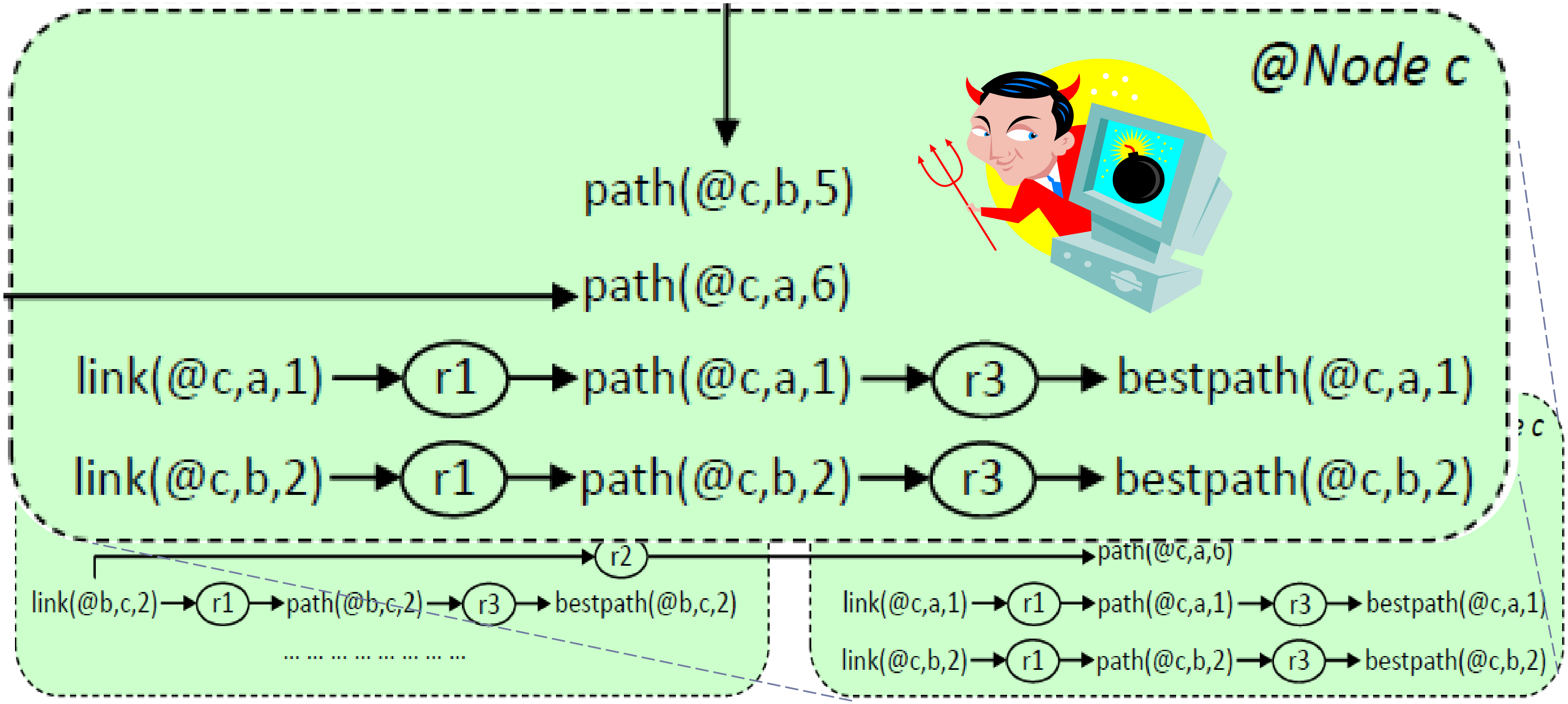


Example: Provenance Graph


@Node c
path(@c,b,5)
path(@c,a,6)
link(@c,a,1) → r1 → path(@c,a,1) → r3 → bestpath(@c,a,1)
link(@c,b,2) → r1 → path(@c,b,2) → r3 → bestpath(@c,b,2)
r2
path(@c,a,6)
link(@b,c,2) → r1 → path(@b,c,2) → r3 → bestpath(@b,c,2)
... ...
link(@c,a,1) → r1 → path(@c,a,1) → r3 → bestpath(@c,a,1)
link(@c,b,2) → r1 → path(@c,b,2) → r3 → bestpath(@c,b,2)

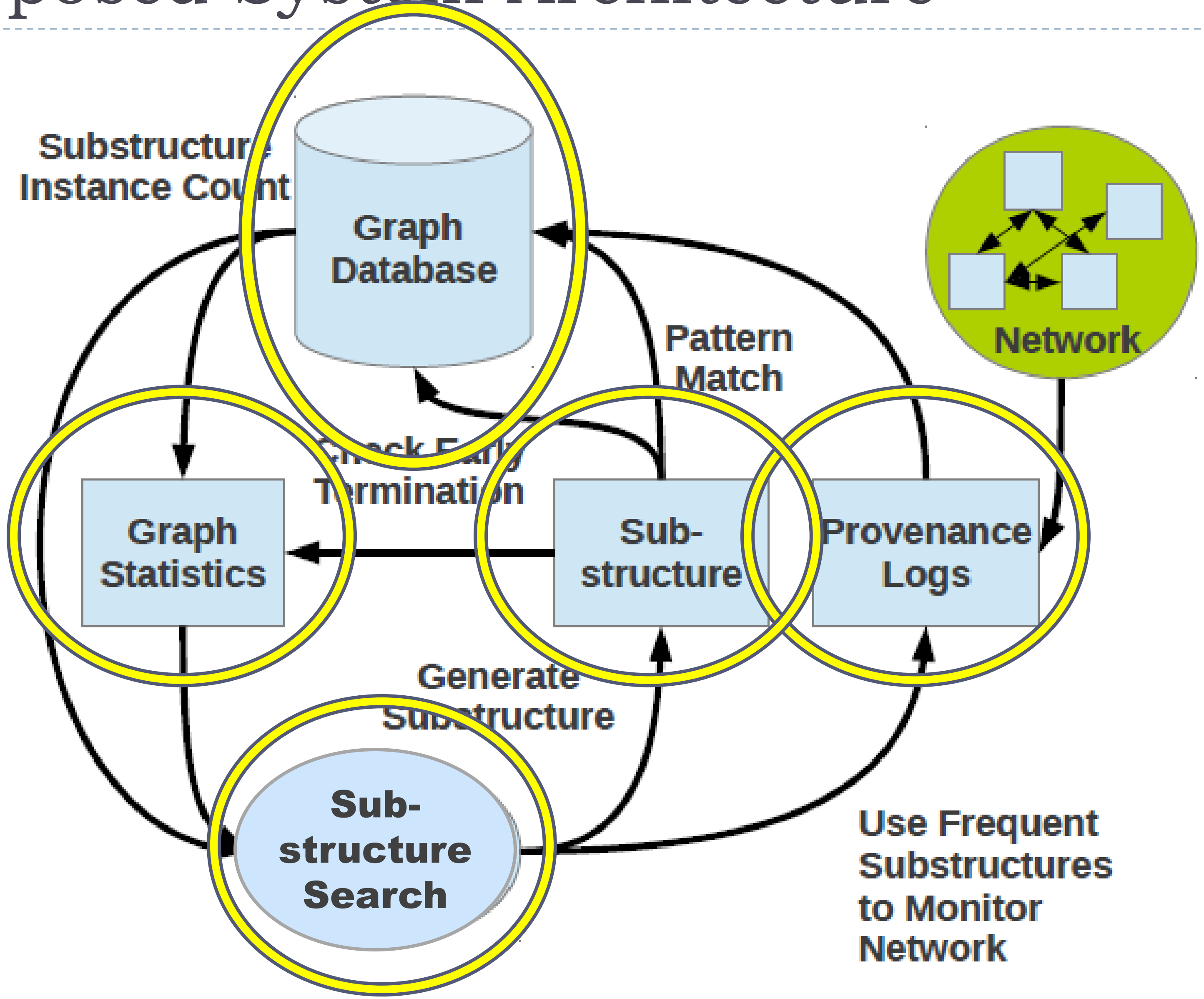
- One Hop Path

Example: Provenance Graph



- No Multi Hop Path

Proposed System Architecture



Substructure Mining

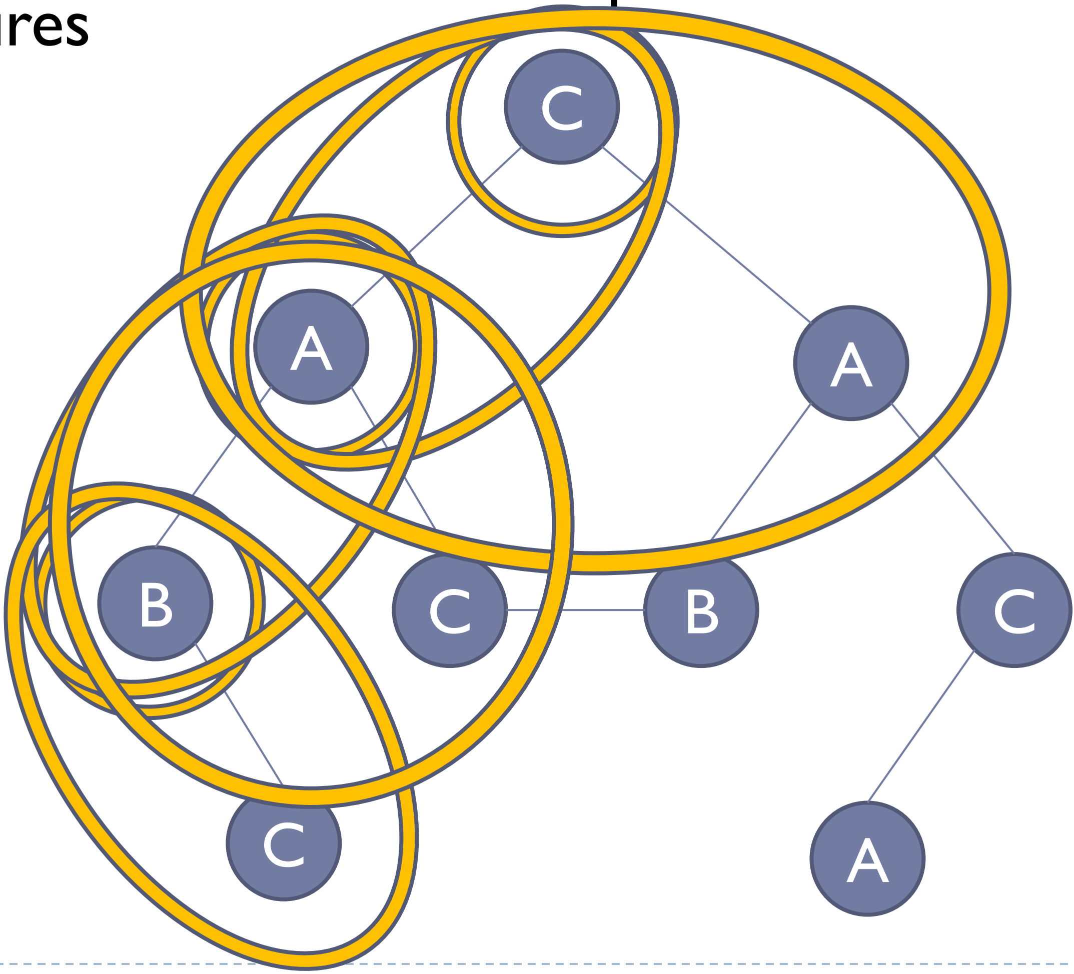
- Substructure mining is the search for "good" subgraphs within a graph or set of graphs

- Two parts:
  - Searching the space of possible substructures
  - Finding instances of an individual substructure

Substructure Mining: Substructures

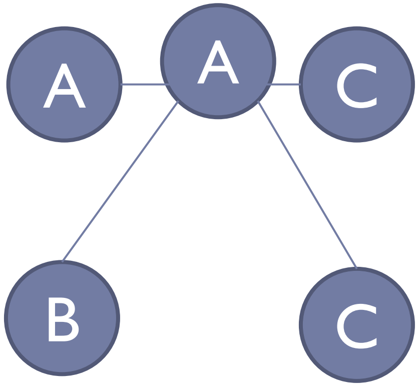
- Many Possible substructures

- Graph

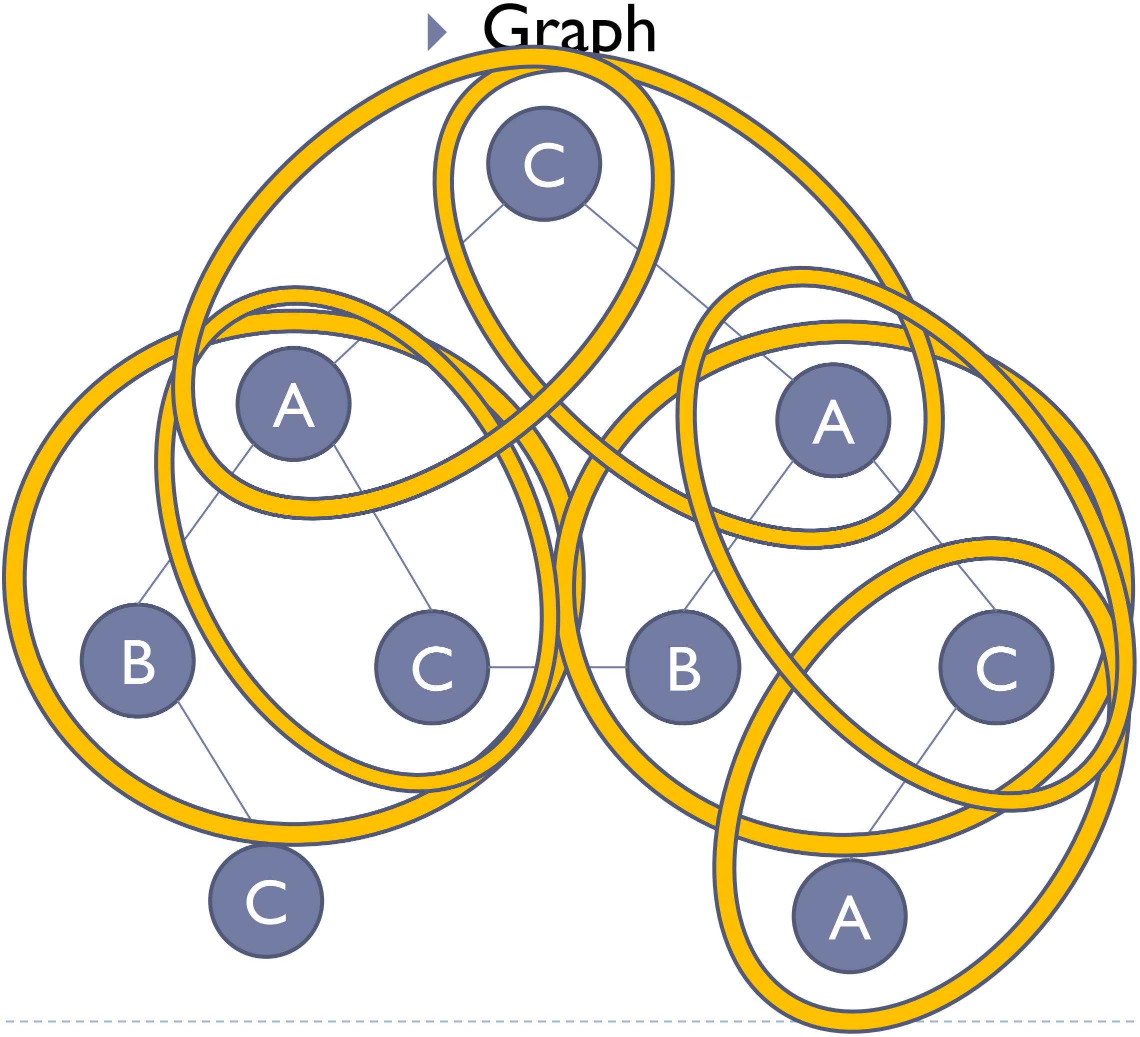


Substructure Mining: Instances

- Substructure



- Graph

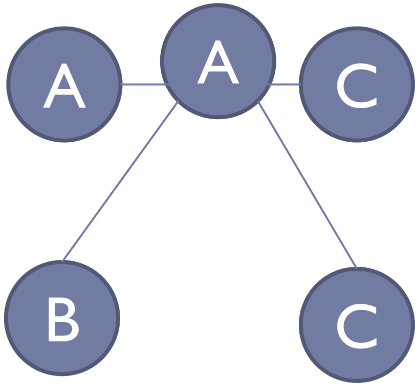


Subdue

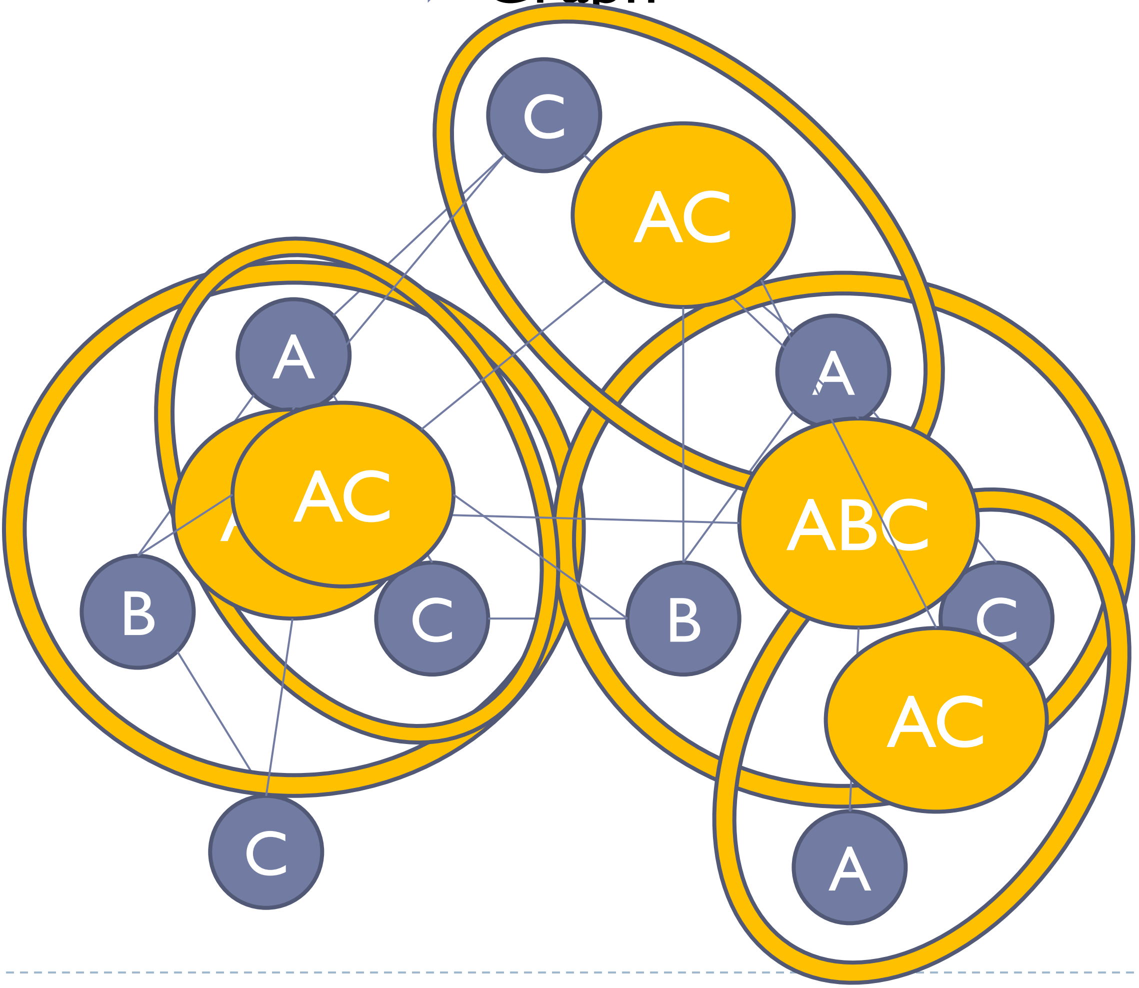
- Classical substructure mining algorithm (N.S.Ketkar et al., 2005)
- Substructures are evaluated based on how well they compress the full graph
  - Compression calculated based on non-overlapping instances
- Subdue uses a guided beam search to search the space of possible substructures
  - Structures from a previous iteration are expanded, tested, and only the best of the expanded go on to the next iteration (beam size = number of the best substructures)

Substructure Mining: Subdue

- Substructure

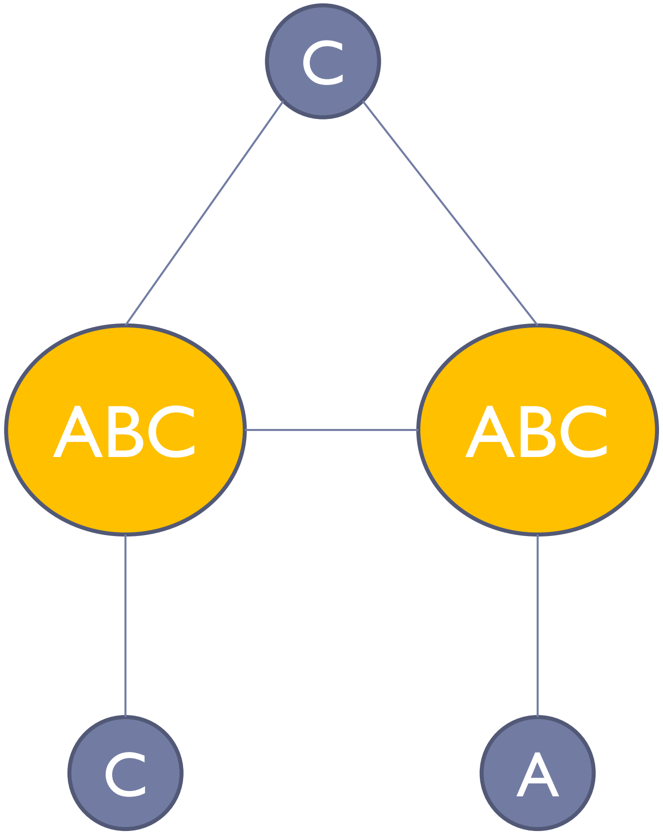


- Graph

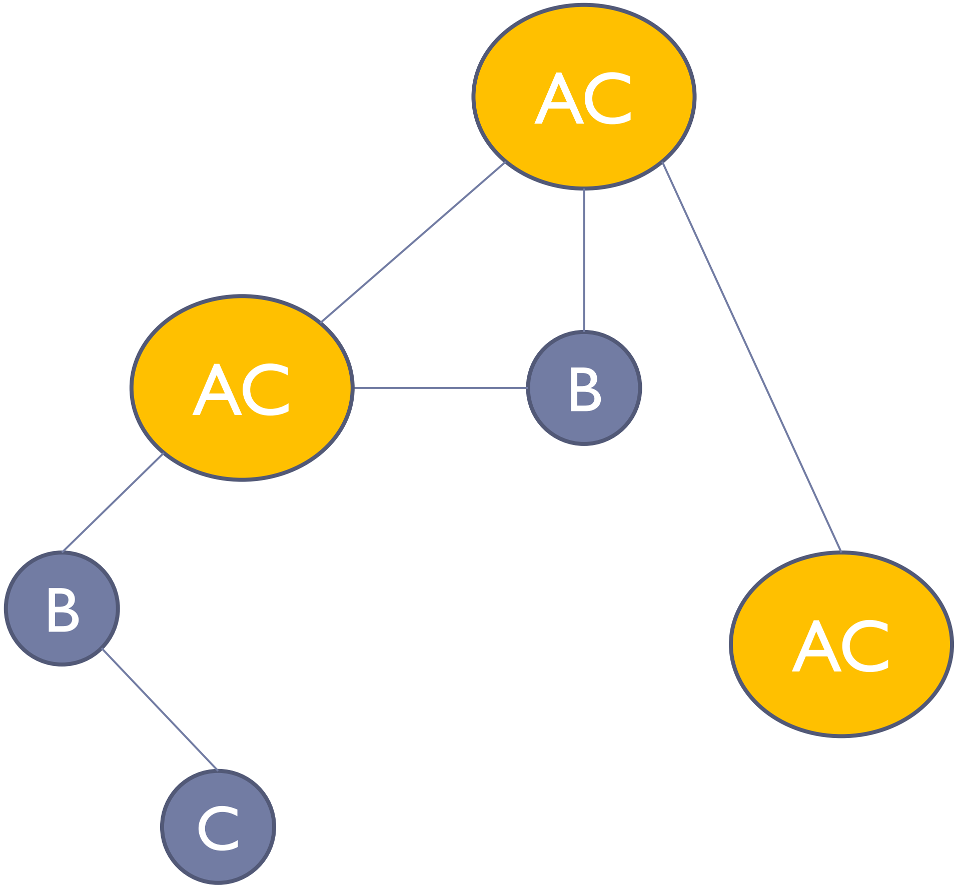


Substructure Mining: Subdue

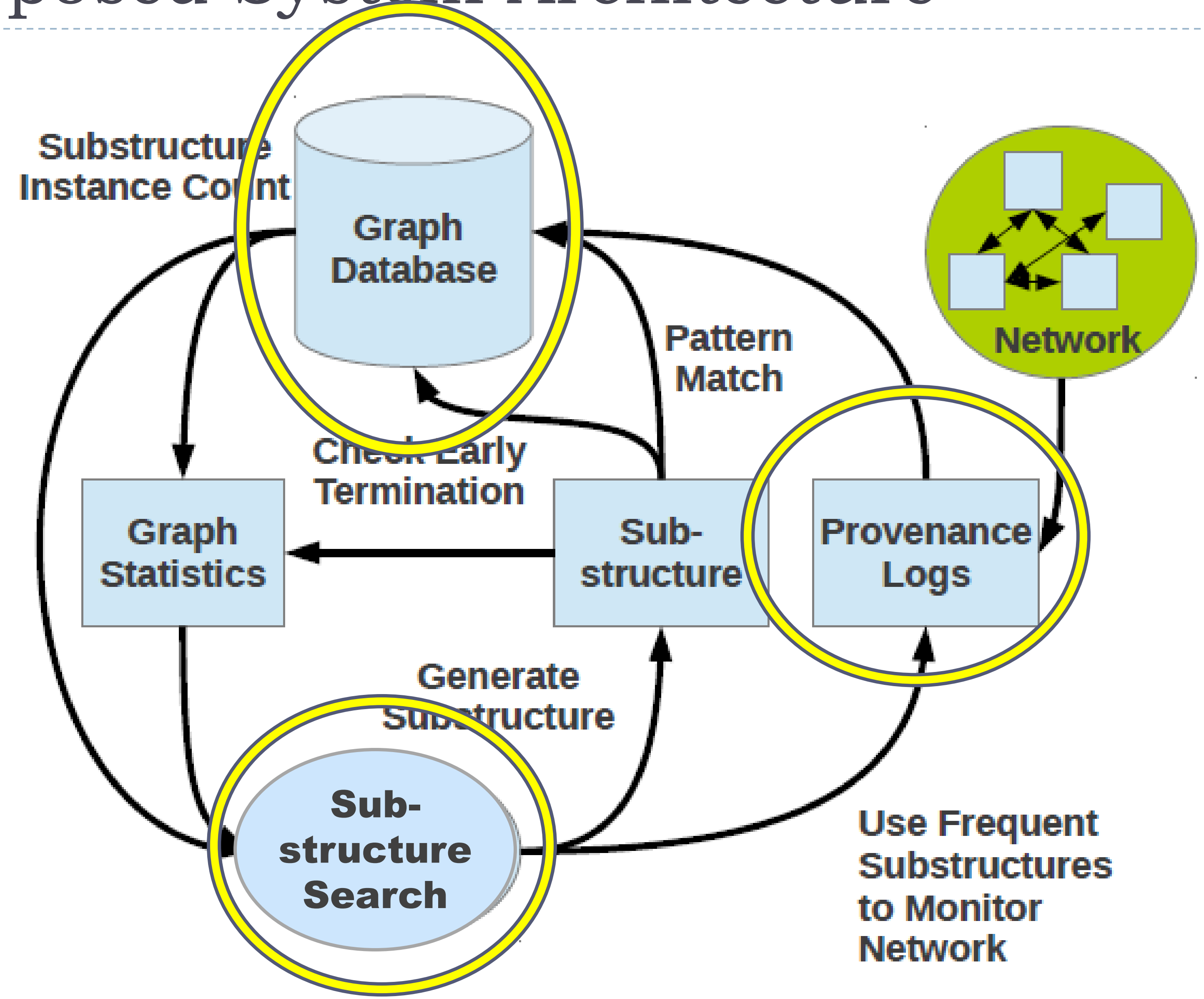
- Compressed Graph 1



- Compressed Graph 2



Proposed System Architecture



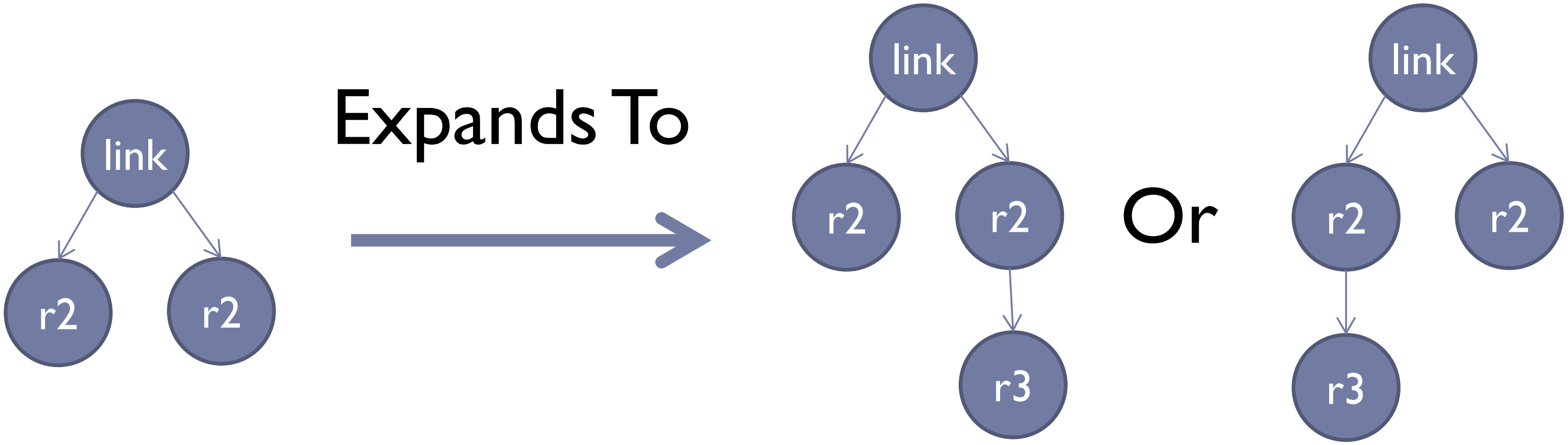
Heuristics

- Limiting the number of substructures to search
  - Duplicate Substructure Reduction
  - Outward Expansion

- Speeding up the search for substructure instances
  - Infrequent Start Vertex
  - Start Vertex Reuse

Duplicate Substructure Reduction

- During the expansion of substructures you duplicate substructures are created and tested.
- We incorporated aspects of Gspan (Yan and Han, 2003) to help reduce the number of duplicates

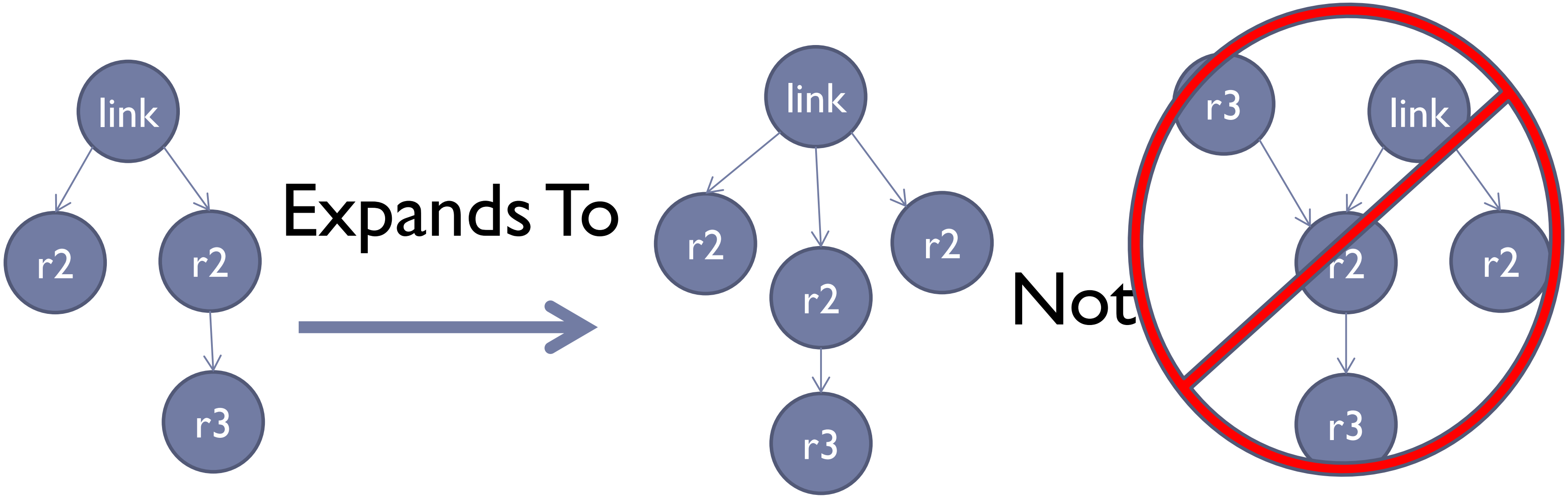


Expands To

Or

Outward Expansion

- When determining new substructures to search for, only expand using outgoing edges
- A possible problem is that certain types of substructures will be ignored.

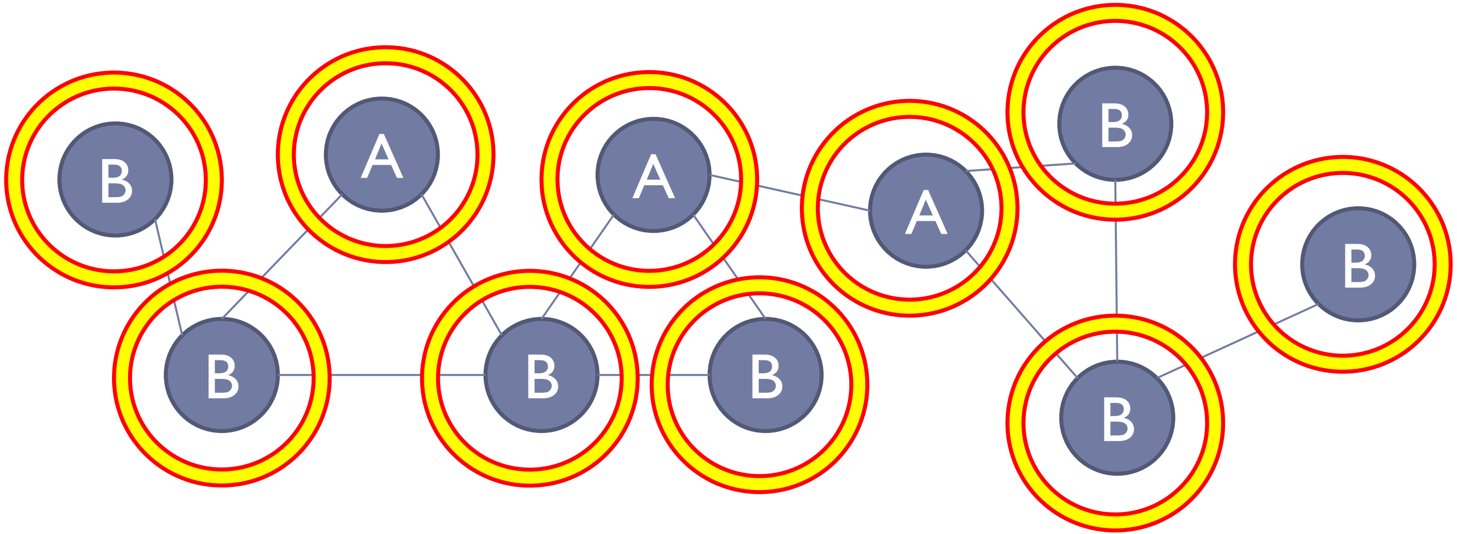


Expands To

Not

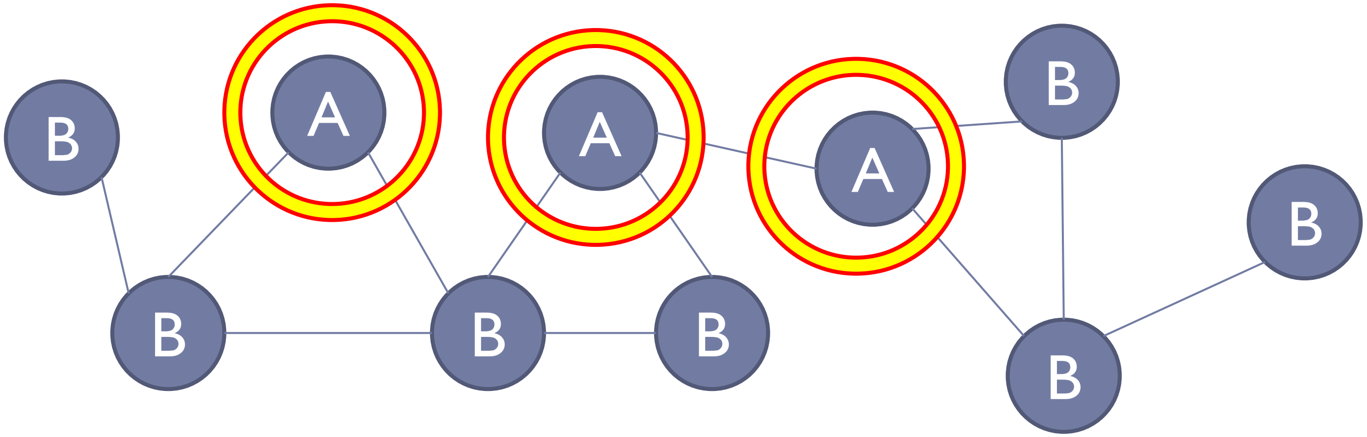
Infrequent Start Vertex

- Testing a substructure instance starts with a single vertex
- Pick start vertices based on the least frequently occurring vertex type in the substructure



Start Vertex Reuse

- Good substructures get expanded to new substructures
- Save the subset of start vertices which have a match
- New substructures can take advantage of the information from the previous substructure



Experimental Setup

- Use 5 different inferred intra-domain topologies from the Rocketfuel project (Spring et al., 2002)

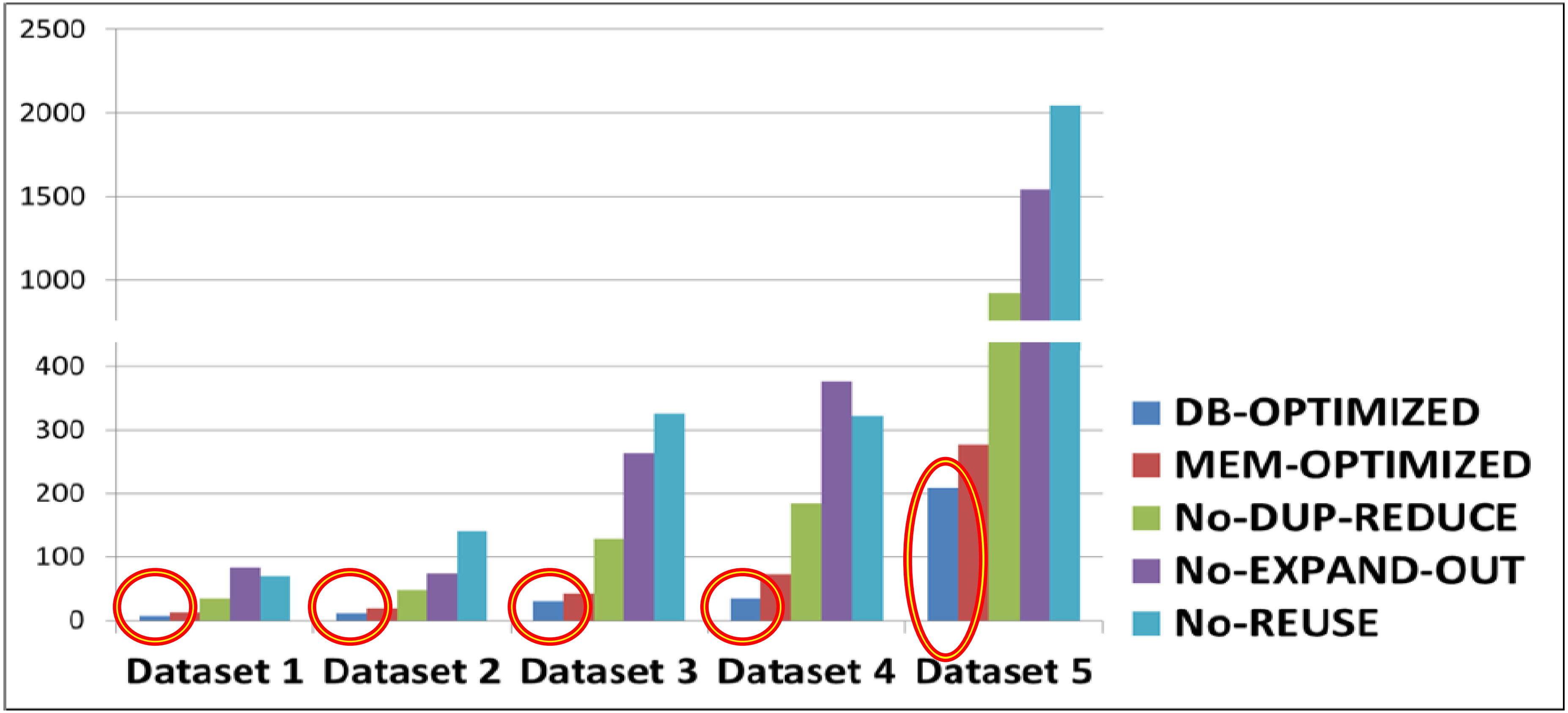
Dataset	ASN	Nodes	Links	|V(G)|	|E(G)|
1	1221	108	306	16,227	28,090
2	1755	87	322	23,015	40,725
3	3257	161	656	52,848	94,568
4	6461	141	748	73,316	134,072
5	1239	315	1,944	317,066	592,038

- Use a beam size of 10 with 100 expansions maximum
- Evaluate run time, quality of substructures, and effect of beam size

Experimental Runs

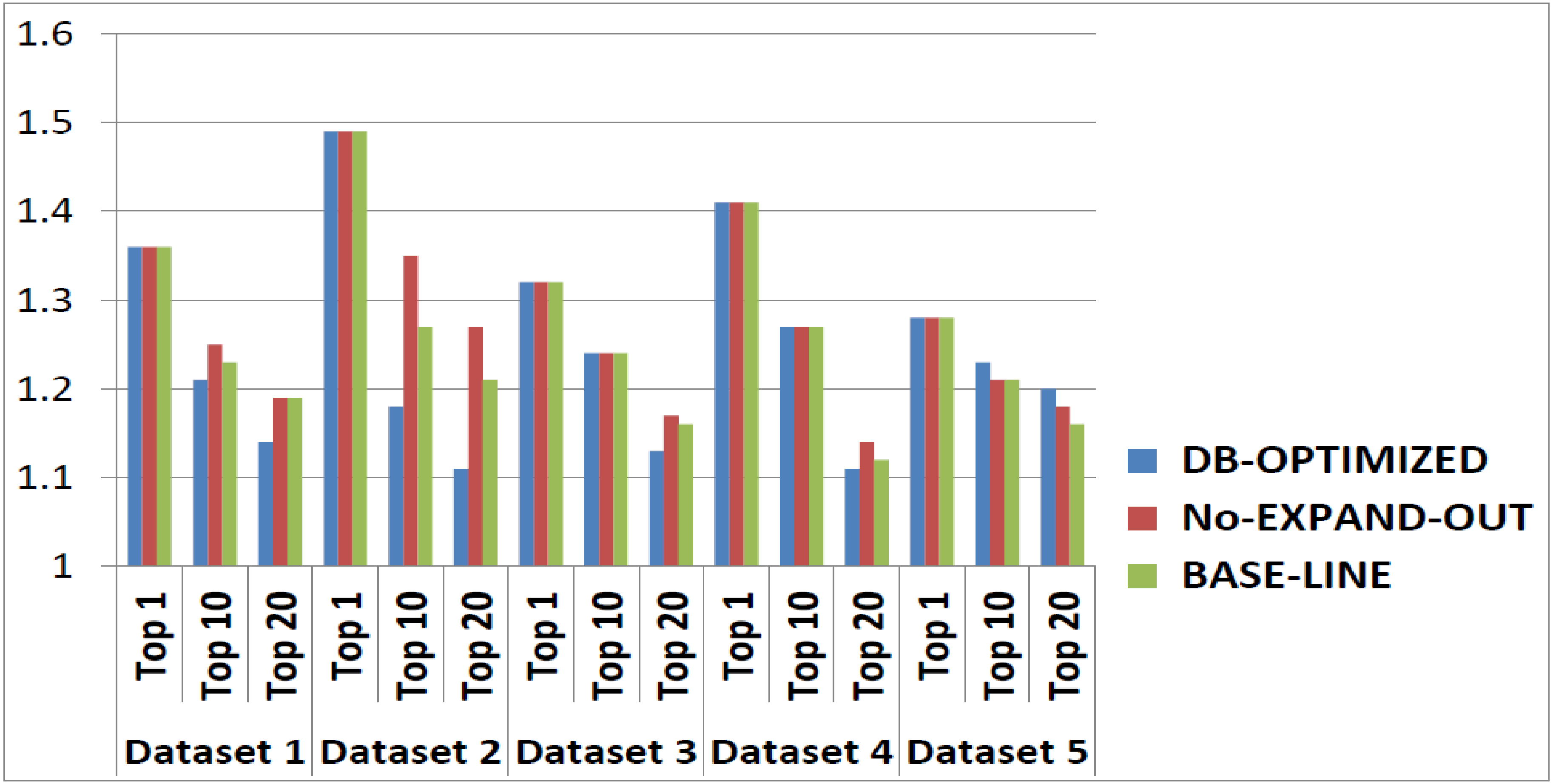
- DB-OPTIMIZED: all heuristics using Neo4j
- MEM-OPTIMIZED: all heuristics using in memory version
- No-DUP-REDUCE: all heuristics except duplication reduction
- No-EXPAND-OUT: all heuristics except outward expansion
- No-REUSE: all heuristics except reuse of start vertices
- BASE-LINE: no heuristics

Results (Run Time)



- Each heuristic improves the run time
- DB version consistently outperforms the memory version

Results (Compression)



- Top compression results the same for each run

Conclusion

- Contributions
  - Apply substructure mining to network provenance
  - Implement algorithm using the neo4j graph database
  - Propose heuristics which take advantage of provenance structure
  - Perform extensive evaluation that shows strength of our approach
- Future Work
  - Try other protocols
  - Use more advanced substructure mining techniques
  - Take advantage of the tree like structure of our graphs
  - Explore substructure mining for dynamic provenance graphs
  - Implement a complete system to test using misbehaving nodes

References

- N.S. Ketkar, L.B. Holder, and D.J. Cook. Subdue: compression-based frequent pattern discovery in graph data. In *Proc. OSDM*, 2005.
- N. Spring, R. Mahajan, and D. Wetherall. Measuring isp topologies with rocketfuel. *ACM SIGCOMM CCR*, 32(4), 2002.
- X. Yan and J. Han. Closegraph: mining closed frequent graph patterns. *In Proc. SIGKDD*, 2003.
- W. Zhou, M. Sherr, T. Tao, X. Li, B. T. Loo, and Y. Mao. Efficient querying and maintenance of network provenance at Internet-scale. In *Proc. SIGMOD*, 2010.