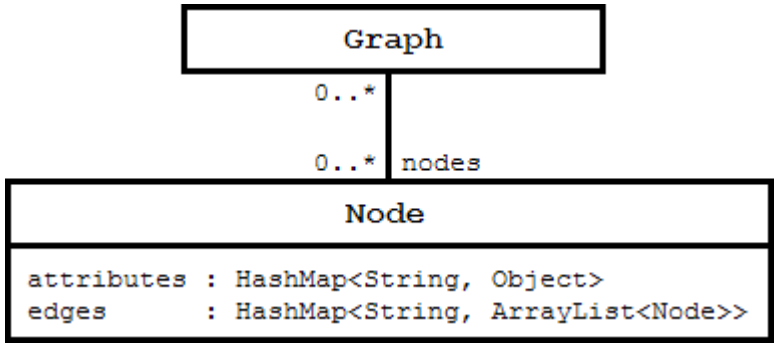
UNIKASSEL
VERSITÄT

Yage Solution to the State Elimination Case

Christoph Eickhoff, Simon-Lennert Raesch, Philipp Kolodziej

Yage – Yet Another Graph Engine




```
Graph ferrymansGraph = new Graph();
Node wolf = new Node(), goat = new Node(), cabbage = new Node();
ferrymansGraph.addNode(wolf).addNode(goat).addNode(cabbage);
wolf.setAttribute("species", "Wolf").addEdge("eats", goat);
goat.setAttribute("species", "Goat").addEdge("eats", cabbage);
cabbage.setAttribute("species", "Cabbage");
```

Basic idea

- read and load data
- eliminate states:
  - look for states not visited
  - if it should deleted
    - create new edge
    - mark as used
    - delete used states

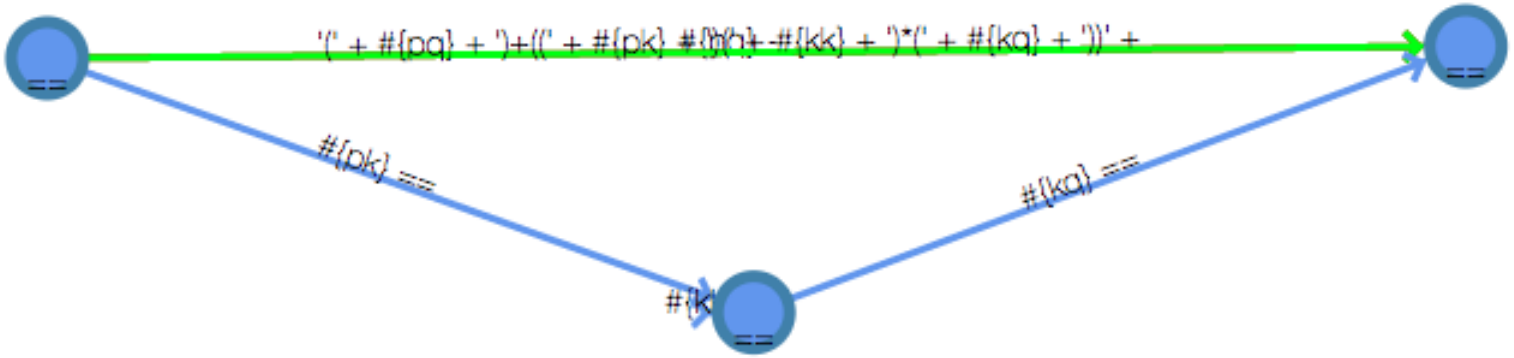
Algorithm

```
Algorithm stateCaseTTC2017 = new Algorithm("TTC 2017 State Case");

Algorithm eliminateState = new Algorithm("eliminate state");
Algorithm handleSourceNode = new Algorithm("handle source node");
Algorithm redirectRoute = new Algorithm("redirect route");

stateCaseTTC2017.addAlgorithmStep(getStateCasePreparationAlgorithmTTC2017());
stateCaseTTC2017.addAlgorithmStep(eliminateState, true);
eliminateState.addAlgorithmStep(getMarkStateForEliminationPattern());
eliminateState.addAlgorithmStep(handleSourceNode, true);
handleSourceNode.addAlgorithmStep(getMarkWithCurrentPattern());
handleSourceNode.addAlgorithmStep(getMarkFallbackWithCurrentPattern());
handleSourceNode.addAlgorithmStep(redirectRoute, true);
redirectRoute.addAlgorithmStep(getPrepareStateWithPqPkKkKqPattern(), true);
redirectRoute.addAlgorithmStep(getPrepareStateWithPkKkKqPattern(), true);
redirectRoute.addAlgorithmStep(getPrepareStateWithPqPkKqPattern(), true);
redirectRoute.addAlgorithmStep(getPrepareStateWithPkKqPattern(), true);
redirectRoute.addAlgorithmStep(getPrepareStateWithPpPkKkKpPattern(), true);
redirectRoute.addAlgorithmStep(getPrepareStateWithPpPkKpPattern(), true);
redirectRoute.addAlgorithmStep(getPrepareStateWithPkKkKpPattern(), true);
redirectRoute.addAlgorithmStep(getPrepareStateWithPkKpPattern(), true);
handleSourceNode.addAlgorithmStep(getUnmarkCurrentPattern());
handleSourceNode.addAlgorithmStep(getRemoveMarksPattern(), true);
eliminateState.addAlgorithmStep(getEliminateMarkedStatePattern());
eliminateState.addAlgorithmStep(getUnmarkPastPattern(), true);
```

Prepare elimination of state



Prepare elimination of state

```
public static PatternGraph getPrepareStateWithPqPkKqPattern() { // #2.3.3 (all matches; don't repeat) - could also be repeated
   // gtr for adding new calculated labels
   PatternGraph gtr = new PatternGraph("prepare elimination of state (with just pq, pk, kq)");
   PatternNode p = new PatternNode("#{current}");
   PatternNode k = new PatternNode("#{eliminate}");
   PatternNode q = new PatternNode("!(#{used})").addPatternAttribute(new PatternAttribute().setAction("+").setName("used").setValue(true));
   gtr.addPatternNode(p, k, q);
   /* CASE 3: there's just pq, pk and kq */
   p.addPatternEdge("==", "#{pk}", k);
   k.addPatternEdge("==", "#{kq}", q);
   p.addPatternEdge("-", "#{pq}", q);
   p.addPatternEdge("+", "'(' + #{pq} + ')+((' + #{pk} + ')(' + #{kq} + '))'", q);
   return gtr;
}
```

Conclusion

Task	Test file	YAGE runtime	JFLAP runtime
task-main	leader3_2.xmi	0.818903705 s	0.09 s
task-main	leader4_2.xmi	2.244564188 s	0.14 s
task-main	leader3_3.xmi	2.761800781 s	0.49 s
task-main	leader5_2.xmi	8.766485675 s	3.46 s
task-main	leader3_4.xmi	9.005200941 s	4.37 s
task-main	leader4_3.xmi	30.331827838 s	57.78 s
task-main	leader3_5.xmi	30.478661809 s	58.6 s
task-main	leader6_2.xmi	45.064825973 s	143.12 s
task-main	leader3_6.xmi	86.166533262 s	461.64 s
task-main	leader4_4.xmi	271.480884718 s	4786.58 s
task-main	leader5_3.xmi	452.646586775 s	timeout
task-main	leader3_8.xmi	474.932934305 s	timeout
task-main	leader4_5.xmi	1600.522432121 s	timeout
task-main	leader6_3.xmi	6137.742556275 s	timeout
task-main	leader4_6.xmi	6876.856106731 s	timeout
task-main	leader5_4.xmi	7779.731544311 s	timeout
task-main	leader5_5.xmi	?	timeout