

DRP Contract:
An Algorithm for Determining Graph Properties
in Linear time via Reduction Rules

Peter Boothe

November 24, 2003

1 Background

Monadic second order logic is an generalization of first order logic that allows quantification over set variables as well as individual entities - but not over functions. A *graph reduction system* consists of a finite set of labeled graph pairs (*reductions*) and a finite set of irreducible graphs (*reducts*). A reduction system is applied to a graph by replacing the left hand side of a reduction with the right hand side, until no further reductions can be applied. If a graph that is isomorphic to one of the reducts results, then the reduction system accepts the graph. Otherwise it does not.

As shown in [1], for every property of a graph that can be written down in monadic second order logic, there exists a reduction system and a finite set of reducts. For every reduction system and set of reducts, there exists a linear time algorithm that will determine whether a particular graph can be reduced to one of the reducts via the reduction system. Unfortunately each step of this chain is non-constructive, so while the existence of the algorithm is not in doubt for a particular reduction system, its exact form currently has to be derived via non-algorithmic means.

2 Proposal

I propose to design and implement a general algorithm that, for any given reduction system, tests whether a given graph is accepted in $O(n)$ expected time, where n is the size of the input graph.

The reduction approach to determining graph properties generally implies an algorithm for each reduction system, and algorithms using this approach have been designed and implemented for many properties, including algorithms for recognizing partial 3-trees[1], and planar partial 3-trees[2]. These algorithms were specific to those properties, however, and were not generalizable to an arbitrary reduction system. As of yet a general algorithm that will work for all

reduction systems has not been discovered, and a program that will accept as inputs both a reduction system and a graph, and then output whether or not the graph has the property associated with that reductions system has been an open problem.

I have implemented one version of the algorithm, but have yet to formally prove its correctness and running time. It uses some concepts from parameterized complexity, including combinatorial explosion for subsets of bounded size, and reduction to the problem kernel[3]. As implemented, the system takes $O(n)$ memory and executes in $O(n)$ expected time. Preliminary results indicated that the same algorithm could be implemented in $O(n \lg n)$ time with $O(n)$ memory or $O(n)$ time with $O(n^k)$ memory (where k is roughly proportional to the size of the largest reduction rule) with a different choice of storage data structures.

3 Deliverables and Timeline

The deliverables for this project include:

Formal Description: A formal description of the algorithm including correctness and complexity results

Implementation Issues: A discussion section concerning different methods of implementing the algorithm as well as issues and challenges encountered along the way.

Implementation: An implementation of one of the forms of the algorithm.

Users Guide: A document explaining how to use the program.

Examples: Usage examples, including input graphs and output results.

My anticipated completion date is the end of Fall term 2003.

References

- [1] Stefan Arnborg, Bruno Courcelle, Andrzej Proskurowski, and Detlef Seese. An algebraic theory of graph reduction. *Journal of the ACM (JACM)*, 40(5):1134–1164, 1993.
- [2] Stefan Arnborg and Andrzej Proskurowski. Characterization and recognition of partial 3-trees. *SIAM J. Alg. Disc. Meth.*, 7:305–314, 1986.
- [3] R. G. Downey and M. R. Fellows. *Parameterized Complexity*. Springer, 1999.