

Simpler Linear-Time Modular Decomposition Via Recursive Factorizing Permutations

Marc Tedder¹, Derek Corneil^{1,*}, Michel Habib², and Christophe Paul^{3,**}

¹ Department of Computer Science, University of Toronto
{`mtedder,dgc`}@`cs.toronto.edu`

² LIAFA and the University of Paris 7 - Denis Diderot
`habib@liafa.jussieu.fr`

³ CNRS - LIRMM, Univ. Montpellier II France (part of this research was conducted while on sabbatical in the School of Computer Science at the University of McGill)
`christophe.paul@lirmm.fr`

Abstract. Modular decomposition is fundamental for many important problems in algorithmic graph theory including transitive orientation, the recognition of several classes of graphs, and certain combinatorial optimization problems. Accordingly, there has been a drive towards a practical, linear-time algorithm for the problem. This paper posits such an algorithm; we present a linear-time modular decomposition algorithm that proceeds in four straightforward steps. This is achieved by introducing the notion of factorizing permutations to an earlier recursive approach. The only data structure used is an ordered list of trees, and each of the four steps amounts to simple traversals of these trees. Previous algorithms were either exceedingly complicated or resorted to impractical data-structures.

1 Introduction

A natural operation to perform on a graph G is to take one of its vertices, say v , and replace it with another graph G' , making v 's neighbours universal to the vertices of G' . Modular decomposition is interested in the inverse operation: finding a set of vertices sharing the same neighbours outside the set – that is, finding a *module* – and contracting this module into a single vertex. A graph's modules form a partitive family [2], and as such, define a decomposition scheme for the graph with an associated decomposition tree composed of the graph's *strong modules* – those that don't overlap other modules. To compute this *modular decomposition tree* is to compute the *modular decomposition* (and vice versa), and with its succinct representation of a graph's structure, its computation is often a first-step in many algorithms. Indeed, since Gallai first noticed its importance to comparability graphs [12], modular decomposition has

* Marc Tedder and Derek Corneil wish to thank NSERC for partially funding this research.

** Michel Habib and Christophe Paul were supported by the French ANR project ANR-06-BLAN-0148-01, *Graph Decomposition and Algorithms* (GRAAL).