Versatile and Declarative Dynamic Programming Using Pair Algebras (Abstract)

Robert Giegerich and Peter Steffen

Faculty of Technology, Bielefeld University
Postfach 10 01 31
33501 Bielefeld, Germany
{robert,psteffen}@techfak.uni-bielefeld.de
Fax: +49 (521) 106 6411

Dynamic programming (DP) solves combinatorial optimization problems by recursive decomposition and tabulation of intermediate results. The recently developed discipline of algebraic dynamic programming (ADP) helps to make program development and implementation in nontrivial applications much more effective [1, 2]. It raises dynamic programming to a declarative level of abstraction, separates the search space definition from its evaluation, and thus yields more reliable and versatile algorithms than the traditional DP recurrences. Here we extend this discipline by a pairing operation `***` on evaluation algebras, whose clue lies with an asymmetric combination of two different optimization objectives. This leads to a surprising variety of applications without additional programming effort.

Each dynamic programming algorithm uses an optimization algebra `opt`, which scores solutions to subproblems and applies the objective function. Backtracing is normally used to extract also a concrete solution that yields the optimum score. Little is normally known about uniqueness of optimal solutions, and the degree of redundancy of the near-optimal solution space.

For each ADP problem we introduce three further algebras in a generic form, `count`, `enum` and `pretty`. The combination of two algebras by `***` is not universally correct, but comes with certain proof obligations. We show that running the (unmodified) ADP algorithm with the algebra `(opt *** enum)` substitutes backtracing via a forward computation, and can, if desired, also report co-optimal solutions. Algebra `(opt *** count)` computes multiplicity of optimal solutions, and on demand, computes a histogram of near-optimal scores. Algebra `(opt *** pretty)` provides visualization of results, and algebra `pretty` by itself, when chosen isomorphic to the problem domain, can be used for testing the (non-)redundancy of the search space. (We cannot do better than testing here, since this question is undecidable.)

These techniques are presented using the problem of finding approximate palindromic patterns in strings, while the real-world application behind this work is the analysis of RNA secondary structures [4, 3].

References

1. R. Giegerich. A systematic approach to dynamic programming in bioinformatics. *Bioinformatics*, 16:665–677, 2000.
2. R. Giegerich, C. Meyer, and P. Steffen. A discipline of dynamic programming over sequence data. *Science of Computer Programming*, in Press, 2004.
3. J. Reeder and R. Giegerich. Design, implementation and evaluation of a practical pseudoknot folding algorithm based on thermodynamics. submitted, 2004.
4. A. Sczyrba, J. Krüger, H. Mersch, S. Kurtz, and R. Giegerich. RNA-related tools on the Bielefeld Bioinformatics Server. *Nucl. Acids. Res.*, 31(13):3767–3770, 2003.