Pattern Matching for Noncommutative Gröbner Basis Computation:

Project Description for CS6014, Summer I

Ben J. Keller

June 6, 1996

Gröbner basis computations in all algebraic settings require some way of determining whether one polynomial term divides another, or if two terms have a nontrivial common multiple. In the noncommutative setting, terms are words in a free semigroup where the words cannot be normalized to a form on which testing for division or common multiples is straight-forward. So, these tests are essentially standard pattern matching searches with a set of patterns.

Pattern matching with a set of patterns is called *dictionary matching*, and a solution using pattern matching automata is described by Aho and Corasick [1]. This approach seems ideally suited to pattern matching in Gröbner basis computations, and, in fact, a similar approach is described by Sims for use in the Knuth-Bendix Completion algorithm for string rewriting [3]. The problem with the Aho and Corasick approach in Gröbner basis computation (and in completion) is that the data structures are static and must be rebuilt each time a new pattern is introduced (or an old one deleted). These problems are overcome in a *dynamic* dictionary matching approach described by Amir *et al* [2].

I have already shown that the dynamic dictionary matching approach can be adapted to the pattern matching problems of Gröbner basis computations. The goal of this project is to formally describe the adaptations, how dictionary matching is used in the Gröbner basis computation, and to analyze the complexity of the search algorithms.

References

[1] A. V. Aho and M. J. Corasick, *Efficient string matching: an aid to bibliographic search*, Communications of the ACM, 18 (1975), pp. 333–340.

[2] A. Amir, M. Farach, Z. Galil, R. Giancarlo, and K. Park, *Dynamic dictionary matching*, Journal of Computer and Systems Sciences, 49 (1994).

[3] C. C. Sims, *Computation with Finitely Presented Groups*, Cambridge University Press, New York, 1994.