Efficient Production Match Algorithm and Its Implication for Dynamic Constraint Satisfaction Problems

Bonghan Cho and **Paul Rosenbloom** and **Milind Tambe**
University of Southern California
Computer Science Department and Information Sciences Institute
4676 Admiralty Way, Marina Del Rey, CA 90292, {bcho,rosenbloom,tambe}@ISI.EDU

Introduction[1] Production systems have been a successful paradigm in Artificial Intelligence (AI) programming and have been extensively used in building AI systems including problem-solving systems, cognitive models, expert systems and real-time systems. Despite this extensive use, there remains a key performance bottleneck that limits the applicability of production systems - the *combinatoric* match process. Algorithms that handle the match combinatorics efficiently are increasingly needed as (1) knowledge bases grow in size and (2) real time performance becomes more important.

There has in fact been much progress in production match since Rete (Forgy 1982) was developed, resulting in orders of magnitude of improvements in production system performance. However, despite the current match technology, programmers of production systems experience that they spend substantial amounts of time in rewriting productions just in order to improve production system efficiency.

The goal of our research is to develop an efficient match algorithm for large data sets in working memory in real time environments. The match algorithm we have developed, called ERMA, attempts to eliminate match combinatorics. Due to the elimination of such combinatorics (though not fully eliminated yet), ERMA is able to scale well with large data sets in working memory.

ERMA: New matching algorithm The key idea of ERMA is to avoid duplicating match effort for similar working memory elements (or wmes) by organizing working memory via equivalence relations. These equivalence relations organize working memory into equivalence classes, called *match sets*, such that all wmes in the same match set are involved in generating the same results when productions fire.

In ERMA, the matching units are these match sets, rather than individual wmes in the match sets as in conventional match algorithms. This helps to reduce the match combinatorics since the multiplication factor (i.e. the base of the exponential) is reduced to the number of match sets in working memory from the number of all elements. Furthermore, *when a new wme is added to an existing match set, no additional matching effort is needed.*

We have mathematically analyzed production match and developed a match algebra to carry out a mathematical analysis of ERMA. The algebra, when applied to previous set-based match algorithms (Acharya & Tambe 1993), reveals their key problems — (1) expensive management of the sets and (2) redundant responsiveness to any single change of working memory — which ERMA alleviates. ERMA was implemented within the SOAR system (Laird, Newell, & Rosenbloom 1987) as its matching engine and applied to real tasks as well as common testbeds. ERMA demonstrates orders of magnitude speedup over the existing state-of-the-art Rete match algorithm.

Implications for Dynamic CSP Based on previous work, we established a mapping between production match and Dynamic CSP (DCSP). Based on this mapping, ERMA can be cast as a new DCSP algorithm. ERMA as a DCSP algorithm exploits equivalence relations in constraints and organizes constraints into sets. A match set in ERMA can be operated as an *interchangeable set* in the CSP literature. This match set, however, can be dynamically formed while an interchangeable set is usually formed via preprocessing. Changes in constraints can also be effectively handled in ERMA. A newly added constraint can be instantly checked as to whether it involves a new solution by simply checking existence of a corresponding set.

References

Acharya, A., and Tambe, M. 1993. Collection-oriented match. In *Proceedings of the Second International Conference on Information and Knowledge Management.*

Forgy, C. L. 1982. Rete: A fast algorithm for the many pattern/many object pattern match problem. *Artificial Intelligence* 19(1):17–37.

Laird, J. E.; Newell, A.; and Rosenbloom, P. S. 1987. Soar: An architecture for general intelligence. *Artificial Intelligence* 33(1):1–64.