CALL SUBSUMPTION MECHANISMS FOR TABLED LOGIC PROGRAMS

Flávio Manuel Fernandes Cruz

Projecto/Dissertação realizado sob a orientação do *Prof. Ricardo Rocha*
no *CRACS / FCUP*

1. Estado da Arte

A tabulação é um método de resolução particularmente bem sucedido que resolve algumas das limitações do método de avaliação SLD encontrado em sistemas Prolog [**?**], no tratamento de computações recursivas e/ou redundantes. Em comparação com o método SLD, a tabulação consegue reduzir o espaço de procura, evitar computações redundantes e tem melhores propriedades de terminação [**?**]. Na tabulação, as primeiras chamadas a subgolos tabelados são avaliadas normalmente através da execução do código do programa, enquanto que *chamadas similares* são avaliadas através do consumo das respostas guardadas na *tabela* e que foram geradas pelo subgolo similar correspondente. Em geral, podemos distinguir entre duas formas de determinar se um subgolo A é similar a um subgolo B: *tabulação por variantes* e *tabulação por subsumpção*. Na tabulação por variantes, A é similar a B quando eles são iguais por renomeação das variáveis. Na tabulação por subsumpção, A é similar a B quando A é mais específico do que B (ou B é mais geral do que A). Isto acontece pelo simples princípio de que se A é mais específico do que B e S_A e S_B são os respectivos conjuntos de respostas, então $S_A \subseteq S_B$. Embora a tabulação por subsumpção consiga atingir maiores ganhos em termos do tempo de execução, devido à maior partilha de respostas, a implementação eficiente dos mecanismos necessários para seu suporte é bastante mais difícil em comparação com tabulação por variantes, o que faz com que este último seja bastante mais popular entre os motores de tabulação disponíveis.

2. Objectivos

Esta tese descreve a migração e integração do mecanismo de *Time Stamped Tries* (TST) do motor de tabulação SLG-WAM [**?**] no motor de tabulação YapTab [**?**]. Este mecanismo foi proposto por Ernie Johnson *et al.* [**?**, **?**] e implementa os algoritmos e estruturas de dados que suportam tabulação por subsumpção. A técnica TST é baseada na ideia de estender a tabela com informação temporal para se distinguir respostas novas das antigas.

Na segunda parte desta tese apresenta-se o desenho, implementação e avaliação de uma nova extensão baseada na tabulação por subsumpção chamada *Tabulação por Subsumpção Retroactiva* (TSR) [**?**]. A TSR resolve algumas limitações da tabulação por subsumpção tradicional, nomeadamente, o facto de a ordem da chamada dos subgolos poder afectar o seu sucesso e aplicação. A TSR permite uma partilha completa e bidireccional de respostas entre subgolos, independentemente da sua ordem de chamada através do corte da avaliação dos golos mais específicos. Para implementar tabulação retroactiva foram desenvolvidas algumas novas ideias: (1) um algoritmo eficiente para pesquisar, na tabela, subgolos mais específicos; (2) uma nova organização da tabela, onde as respostas são representadas apenas uma vez; (3) uma nova estratégia de execução capaz de cortar e transformar a execução de um subgolo gerador em consumidor.

3. Resultados

Os nossos resultados mostram que a integração dos mecanismos e algoritmos da TST da SLG-WAM para o YapTab foram bem sucedidos, com desempenhos comparáveis aos da SLG-WAM na execução entre tabulação por variantes e tabulação por subsumpção.

Para a TSR, os nossos resultados mostram ganhos consideráveis para os programas que conseguem tirar partido do novo mecanismo, enquanto que o custo associado aos programas que dele não conseguem beneficiar é quase insignificante.

4. Conclusões

As contribuições desta tese são as seguintes: (1) suporte para tabulação por subsumpção implementada no Yap Prolog; (2) a nova técnica TSR que permite partilha bidireccional de respostas; (3) uma nova organização da tabela que permite uma melhor partilha entre subgolos de um predicado; e (4) um motor de tabulação capaz de usar diferentes técnicas de execução, tais como tabulação por variantes, por subsumpção e por subsumpção retroactiva. O nosso sistema final permite que o programador escolha a melhor estratégia por predicado, o que torna a tabulação mais prática e usável em problemas reais.

Referências

[1] W. Chen e D. S. Warren. Tabled Evaluation with Delaying for General Logic Programs. *Journal of the ACM*, 43(1):20–74, 1996.

[2] H. Tamaki e T. Sato. OLDT Resolution with Tabulation. Em *International Conference on Logic Pro-*

gramming, número 225 em LNCS, páginas 84–98. Springer-Verlag, 1986.

[3] K. Sagonas. *The SLG-WAM: A Search-Efficient Engine for Well-Founded Evaluation of Normal Logic Programs*. Tese de doutoramento, Department of Computer Science, State University of New York, 1996.

[4] R. Rocha, F. Silva, e V. Santos Costa. YapTab: A Tabling Engine Designed to Support Parallelism. Em *Conference on Tabulation in Parsing and Deduction*, páginas 77–87, 2000.

[5] E. Johnson, C. R. Ramakrishnan, I. V. Ramakrishnan, e P. Rao. A Space Efficient Engine for Subsumption-Based Tabled Evaluation of Logic Programs. Em *Fuji International Symposium on Functional and Logic Programming*, número 1722 em LNCS, páginas 284–300. Springer-Verlag, 1999.

[6] Ernie Johnson. Interfacing a Tabled-WAM Engine to a Tabling Subsystem Supporting Both Variant and Subsumption Checks. Em *Conference on Tabulation in Parsing and Deduction*, 2000.

[7] F. Cruz e R. Rocha. Retroactive Subsumption-Based Tabled Evaluation of Logic Programs. Em T. Janhunen e I. Niemelä, editores, *Proceedings of the 12th European Conference on Logics in Artificial Intelligence, JELIA'2010*, LNCS, Helsinki, Finland, September 2010. Springer-Verlag.