REDUÇÃO DO TEMPO DE RESPOSTA DOS SISTEMAS DE MITIGAÇÃO DO PROBLEMA DE SELEÇÃO DE OTIMIZAÇÕES

Marcos Yukio Siraichi (PIBIC/CNPq), Anderson Faustino da Silva (Orientador), e-mail: ra67333@uem.br.

Universidade Estadual de Maringá / Centro de Tecnologia/Maringá, PR.

Ciências exatas e da terra – Ciência da Computação

Palavras-chave: estimativa de desempenho, compiladores, otimizações

Resumo:
Compiladores modernos tentam melhorar um código de entrada através da aplicação de combinações de transformações, chamadas otimizações. No entanto, a seleção de otimizações não é trivial, já que as combinações de otimizações disponíveis resulta em uma explosão combinatória. Assim, é possível encontrar várias pesquisas na literatura, que tentam explorar o espaço de otimizações, a partir de abordagens aleatórias, exaustivas, e guiadas por heurísticas e aprendizado de máquina. Visto que essas pesquisas utilizam-se da execução do programa para guiar a seleção de otimizações, tais abordagens tornam-se custosas demais, dependendo do tempo necessário para executar o código de entrada. Considerando esses problemas, este trabalho utiliza-se de uma estratégia, que estima o desempenho de um programa, proposta anteriormente, para guiar um seletor de otimizações baseado em aprendizado de máquina. Mostramos que a estimativa reduz o tempo necessário para encontrar um conjunto de otimizações, e consegue encontrar otimizações que superam conjuntos previamente testados, mesmo com várias restrições do algoritmo.

Introdução

O PSO (Problema de Seleção de Otimizações) consiste em encontrar o melhor conjunto de otimizações para um programa de entrada específico. Para isto, várias abordagens são encontradas na literatura, sejam elas algoritmos randômicos ou heurísticos(KISUKI et al., 1999; COOPER et al., 2002; TARTARA et al., 2013). Porém, elas utilizam, como método de

FUNDAÇÃO ARAUCÁRIA
CNPq Conselho Nacional de Desenvolvimento Científico e Tecnológico
PARANÁ GOVERNO DO ESTADO Secretaria da Ciência, Tecnologia e Ensino Superior

avaliação do conjunto, a execução do programa. Fazendo com que, para a análise de 100 conjuntos, o programa seja executado 100 vezes, tornando-o altamente custoso.

Portanto, este projeto utiliza-se da estratégia implementada em um projeto anterior para estimar o desempenho de um programa, eliminando a execução iterativa do programa. E aplica-o à um seletor de otimizações, visando, assim, conseguir um desempenho igual ou melhor que utilizando a execução do programa.

Materiais e métodos

A estimativa de desempenho utilizada, foi implementada no *framework* GESO. De uma maneira geral, elas são calculadas conforme mostra a Equação 1. A estimativa é calculada baseando-se no peso de cada instrução. Ela multiplica o peso pela frequência de execução do bloco básico que a instrução está inserida, e a soma de todas resulta no custo de executar o módulo.

$$Custo = \sum_{f \in F} \sum_{b \in f} \sum_{i \in b} peso(i) \cdot freq(b) \quad (1)$$

Como seletor de otimizações, foi implementado um algoritmo de aprendizagem contínua de longo prazo. De um modo geral, esse algoritmo se trata de um algoritmo evolutivo que visa encontrar boas formas para realizar decisões, dependendo apenas de características do programa, coletadas estaticamente, tais como o número de instruções e o número de blocos básicos.

Cada membro da população deste algoritmo é composto por N fórmulas, que representam as decisões que o compilador deve fazer. As decisões abordadas neste projeto consistem em: dada uma otimização X, deve-se ativá-la (aplicá-la ao programa), ou não.

Resultados e Discussão

Os experimentos foram realizados em uma máquina composta por um processador I7-4700MQ, com 4 núcleos físicos e 4 lógicos e 8GB de memória RAM.

Na Figura 1 consta um gráfico com o desempenho obtido com a estratégia de estimativa de desempenho. É possível observar que apenas em alguns casos, a abordagem proposta conseguiu um desempenho comparável à obtida pela aplicação do conjunto “O3”.

Ainda que esse resultado esteja longe do ideal, é possível observar, em alguns casos, o aprendizado do algoritmo. Uma das razões deste baixo ganho de desempenho é o fato dele ser um algoritmo “de longo prazo”, o qual foi analizado o desempenho após a compilação de, no máximo, apenas 20 programas anteriores.

Além disso, o algoritmo contou com algumas restrições não apresentadas em seu artigo original. Enquanto esse programa apenas habilita e desabilita transformações oferecidas pelo compilador, ainda é preciso resolver questões como a ordem das otimizações e seus parâmetros.

Dentre as limitações do algoritmos, encontramos: para cada rodada de testes, foi definida uma sequência fixa de otimizações, da qual o seletor apenas habilitava e desabilitava otimizações; não haviam otimizações repetidas; e a não parametrização das transformações oferecidas pelo compilador.

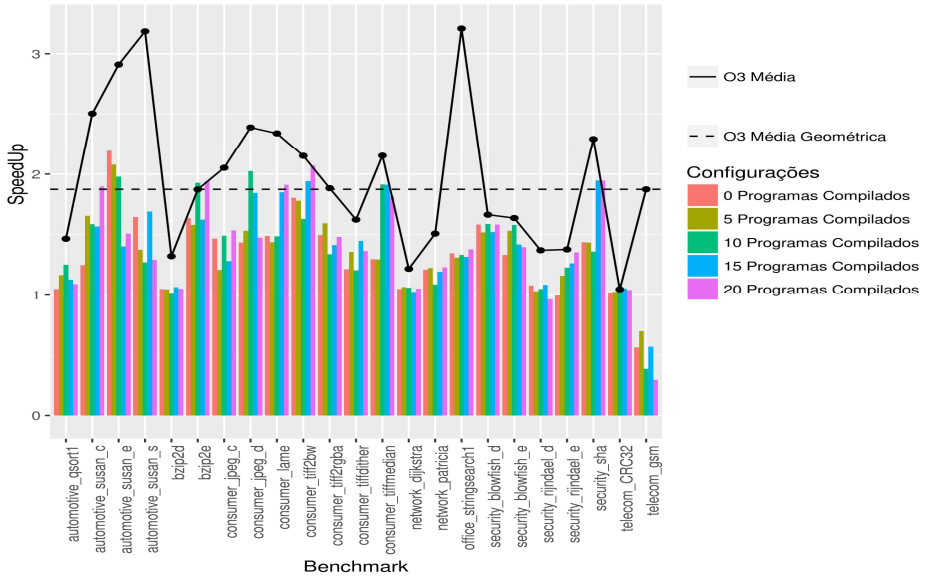

Figura 1 – Gráfico contendo o speedup obtido com o auxílio da ferramenta GESO. A figura mostra o *speedup* de cada programa após a compilação de 0, 5, 10,15 e 20 programas não relacionados.

Mesmo assim, com a utilização das estimativas de desempenho, foi possível reduzir a quantidade de execução do programa. Enquanto, sem a estimativa, cada programa era executado, no máximo, 18 vezes, com a estimativa, o programa era executado apenas 2 vezes.

Conclusões

Dessa forma, é possível observar que, apesar da deficiência de precisão da estratégia adotada, e das restrições oferecidas pelo algoritmo, foi possível obter uma redução significativa do tempo de compilação, com uma certa perda de desempenho.

É possível ver que, melhorando a precisão da estimativa e aumentando os casos de testes, o algoritmo seletor de otimizações é capaz de aprender com compilações anteriores.

Agradecimentos

Agradeço ao CNPQ pela oportunidade de realizar o presente projeto, assim como ao professor Anderson Faustino da Silva pela orientação.

Referências

COOPER, Keith D.; SUBRAMANIAN, Devika; TORCZON, Linda. **Adaptive optimizing compilers for the 21st century**. The Journal Of Supercomputing**,** Hingham, Ma, v. 23, n. 1, p.7-22, 2002. Springer Science

KISUKI, Toru; KNIJNENBURG, Peter M. W.; O'BOYLE, Michael F. P.. **A Feasibility Study in Iterative Compilation**. Ishpc '99 Proceedings Of The Second International Symposium On High Performance Computing, Verlag, London, p.121-132, out. 1999.

ROSARIO, Vanderson M. GEOS: **Guide for Exploration of the Otimization Space**. 2014. Disponível em: <http://github.com/vandersonmr/geos>.

TARTARA, Michele; REGHIZZI, Stefano Crespi. **Continuous learning of compiler heuristics**. Taco**,** Nova Iorque, Usa, v. 9, n. 4, p.1-25, 1 jan. 2013. Association for Computing Machinery (ACM).