1.03.04 - Ciência da Computação / Sistemas de Computação.

Computação Paralela em JavaScript.

Clauber P. Stipkovic Halic[1], Calebe de Paula Bianchini[2]

1. Estudante de Ciência da Computação na Faculdade de Computação e Informática - FCI - da Universidade Presbiteriana Mackenzie; *clauber.halic@gmail.com

2. Professor Pesquisador na Faculdade de Computação e Informática - FCI - da Universidade Presbiteriana Mackenzie

Palavras Chave: *JavaScript, Computação Paralela, SIMD.*

Introdução

Devido a crescente utilização dos browsers como primeira escolha para a criação de aplicações para várias áreas, surgiu a necessidade de uma JavaScript Engine (motor de interpretação de códigos JavaScript) mais rápida e robusta. Assim, os browsers têm a opção de melhorar sua capacidade de executar instruções seguindo as premissas da Computação Paralela.

Este trabalho tem como objetivo apresentar o uso de computação paralela na linguagem JavaScript, iniciando o estudo pelo funcionamento de uma JavaScript Engine, os passos de interpretação e execução de um código JavaScript no browser. Em especial, o estudo será feito na Engine SpiderMonkey da Fundação Mozilla e como ele aplica os conceitos de Just-in-Time Compiler. Por fim, este trabalho mostra o uso de uma biblioteca de computação paralela de granularidade fina chamada SIMD.js e os resultados de sua aplicação.

Resultados e Discussão

Foi verificado que o JavaScript Engine do SpiderMonkey segue os princípios básicos de uma máquina virtual, seguindo os passos clássicos de interpretação de uma código-fonte em JavaScript [1; 2]. Além disso, ele também aplica os conceitos de compilação JIT (Just-in-time Compiler) [3].

Foi estudado também a utilização de arquiteturas com multiprocessadores e multi-cores para a execução de instruções de forma paralela utilizando também granularidade fina por meio do conceito de SIMD, definido por Flynn [4].

A partir desses estudos, foi avaliado o comportamento da biblioteca SIMD.js em aplicações JavaScript no SpiderMonkey Engine. Em especial, na aplicação de renderização de um Conjunto de Mandelbrot, foi observado um desempenho maior na mesma interpretação, demonstrando que o uso de SIMD aumenta o desempenho da interpretação de JavaScript.

O resultado da execução do Conjunto de Mandelbrot pode ser observado quando comparado a quantidade de frames por segundo (FPS) resultante da utilização dos recursos da biblioteca SIMD.js e, consequentemente, dos conceitos de SIMD. Neste caso, o desempenho foi 3 vezes maior quando utilizada a biblioteca SIMD.js.

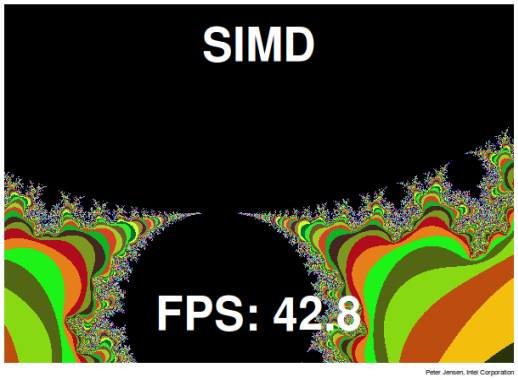

Figura 1. Renderização do Conjunto de Mandelbrot utilizando a SIMD.js.

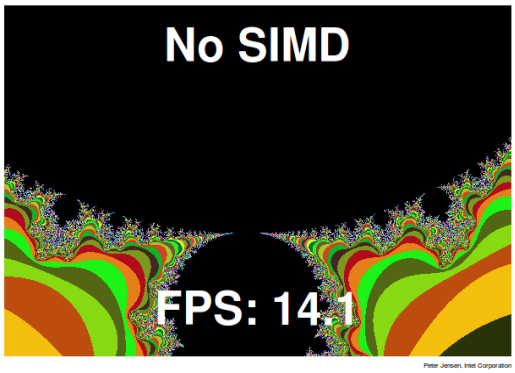

Figura 2. Renderização do Conjunto de Mandelbrot sem a utilização de SIMD.js.

Conclusões

Com a utilização da computação paralela, em especial de granularidade fina, e da biblioteca SIMD.js juntamente com o JavaScript Engine SpideMonkey, fica evidente que é possível ter ganhos de mais de 50% de desempenho e tempo de execução de algoritmos clássicos da computação. Isso também revela a necessidade de ampliar a atualização não só dos recursos nas JavaScript Engine atuais, mas também das aplicações que são construídas nessa linguagem.

[1] SPIDERMONKEY. SpiderMonkey. Disponível em: <https://developer.mozilla.org/en-US/docs/Mozilla/Projects/SpiderMonkey>. Acesso em 15 de Agosto de 2015.

[2] APPEL, A.; PALSBERG, J. Modern Compiler Implementation in Java, Nova York, NY: Cambridge University Press; 2ª edição, 2002.

[3] AHO, Alfred V. et al. Compilers: Principles, Technique, and Tools, Boston, MA: Pearson Education, 2007, 1009 p.

[4] FLYNN, Michael J. Some Computer Organizations and Their Effectivness. IEEE Transactions on Computers. Vol. c-21, No.9, September 1972. Disponível em: <http://www.cs.utah.edu/~kirby/classes/cs6230/Flynn.pdf>. Acesso em: 15 de Agosto de 2015.