WSCAD-CTD 2018

Concurso de Teses e Dissertações em Arquitetura de Computadores e Computação de Alto Desempenho

Conteúdo

WSCAD-CTD Abstracts

Aurora: Seamless Optimization of OpenMP Applications

Author: Arthur Francisco Lorenzon
Advisor: Prof. Dr. Antonio Carlos Schneider Beck

Thesis presented in partial fulfillment of the requirements for the degree of Doctor in Computer Science, Universidade Federal Do Rio Grande Do Sul, Instituto De Informática, Programa De Pós-graduação Em Computação (2018).

Abstract

Efficiently exploiting thread-level parallelism has been challenging for software developers. As many parallel applications do not scale with the number of cores, blindly increasing the number of threads may not produce the best results in performance or energy. However, the task of rightly choosing the ideal amount of threads is not straightforward: many variables are involved (e.g. off-chip bus saturation and overhead of data-synchronization), which will change according to different aspects of the system at hand (e.g., input set, micro-architecture) and even during execution. To address this complex scenario, this thesis presents Aurora. It is capable of automatically finding, at run-time and with minimum overhead, the optimal number of threads for each parallel region of the application and re-adapt in cases the behavior of a region changes during execution. Aurora works with OpenMP and is completely transparent to both designer and end-user: given an OpenMP application binary, Aurora optimizes it without any code transformation or recompilation. By executing fifteen well-known benchmarks on four multi-core processors, Aurora improves the trade-off between performance and energy by up to: 98% over the standard OpenMP execution; 86% over the built-in feature of OpenMP that dynamically adjusts the number of threads; and 91% over a feedback-driven threading emulation.
Keywords: Parallel computing. energy and performance optimization. software tuning. OpenMP.

Resumo

A exploração eficiente do paralelismo no nível de threads tem sido um desafio para os desenvolvedores de softwares. Como muitas aplicações não escalam com o número de núcleos, aumentar cegamente o número de threads pode não produzir os melhores resultados em desempenho ou energia. No entanto, a tarefa de escolher corretamente o número ideal de threads não é simples: muitas variáveis estão envolvidas (por exemplo, saturação do barramento off-chip e sobrecarga de sincronização de dados), que mudam de acordo com diferentes aspectos do sistema (por exemplo, conjunto de entrada, micro-arquitetura) e mesmo durante a execução da aplicação. Para abordar esse complexo cenário, esta tese apresenta Aurora. Ela é capaz de encontrar automaticamente, em tempo de execução e com o mínimo de sobrecarga, o número ideal de threads para cada região paralela da aplicação e se readaptar nos casos em que o comportamento de uma região muda durante a execução. Aurora trabalha com o OpenMP e é completamente transparente tanto para o programador quanto para o usuário final: dado um binário de uma aplicação OpenMP, Aurora o otimiza sem nenhuma transformação ou recompilação de código. Através da execução de quinze benchmarks conhecidos em quatro processadores multi-core, mostramos que Aurora melhora o trade-off entre desempenho e energia em até: 98% sobre a execução padrão do OpenMP; 86% sobre o recurso interno do OpenMP que ajusta dinamicamente o número de threads; e 91% quando comparado a uma emulação do feedback-driven threading.
Palavras-chave: Computação paralela, otimização de desempenho e energia, software tuning, OpenMP.

Algoritmos Paralelos e Eficientes para Consultas IP no Intel(R) Xeon Phi(tm) e CPUs Multi-Core

Autor: Alexandre Lucchesi Alencar
Orientador: Prof. Dr. George Luiz Medeiros Teodoro

Dissertação apresentada como requisito parcial para conclusão do Mestrado em Informática, Instituto de Ciências Exatas, Departamento de Ciência da Computação, Universidade de Brasília (2017)

Resumo:
Roteadores em software são uma solução promissora para lidar com o encaminhamento de pacotes devido ao seu bom custo-benefício e flexibilidade. Contudo, é desafiador o desenvolvimento de roteadores em software capazes de atingir as taxas de encaminhamento de pacotes necessárias. O uso de sistemas e técnicas de computação paralela pode ser uma abordagem viável para melhorar o desempenho dessas soluções. A fase de consulta IP constitui uma operação central no encaminhamento de pacotes, que é implementada através de um algoritmo de Casamento de Maior Prefixo (CMP). Assim, este trabalho propõe e avalia o uso de técnicas e processadores paralelos no desenvolvimento de um algoritmo otimizado que emprega filtros de Bloom (BFs) e tabelas *hash* para a execução de consultas IP. Especificamente, tem-se como alvo a implementação desse algoritmo no coprocessador *many-core* Intel® Xeon Phi™ (Intel Phi), mas também avalia-se o seu desempenho em CPUs *multi-core* e em um modelo de execução cooperativa que usa ambos os processadores com várias otimizações. Os resultados experimentais mostram que foi possível atingir altas taxas de consultas IP — até 182,7 Mlps (milhões de pacotes por segundo) ou 119,9 Gbps para pacotes IPv6 de 84B — em um único Intel Phi. Este desempenho indica que o Intel Phi é uma plataforma promissora para a implantação de algoritmos de consultas IP. Além disso, comparou-se o desempenho do algoritmo BFs com uma abordagem eficiente baseada na Multi-Index Hybrid Trie (MIHT), na qual o algoritmo BFs foi até 5,39x mais rápido. Esta comparação mostra que o algoritmo sequencial mais eficiente pode não ser a melhor opção em uma configuração paralela. Alternativamente, é necessário avaliar as características dos processadores, as demandas de computação/dados dos algoritmos e as estruturas de dados empregadas para analisar como os algoritmos podem se beneficiar de um dispositivo de computação paralelo, potenciais limitações na escalabilidade e oportunidades de otimização. Estas descobertas também são importantes para novos esforços no desenvolvimento de algoritmos nessa área, os quais têm sido, em sua maioria, focado sem soluções sequenciais.
Palavras-chave: Consultas IP, Roteadores em Software, Intel® Xeon Phi™, Casamento de Maior Prefixo.

Abstract
Software routers are a promising solution to deal with packet forwarding because of their good cost benefit and flexibility. However, it is challenging to develop software routers that can attain the required packet forwarding rates. The use of parallel computing systems and techniques may be a viable approach to improve the performance of these solutions.The IP lookup phase is a core operation in packet forwarding, which is implemented viaa Longest Prefix Matching (LPM) algorithm to find the next hop address for every input packet. Therefore, this work proposes and evaluates the use of parallel processors and techniques in the development of an optimized algorithm that employs Bloom filters (BFs) and hash tables to the IP lookup problem. Specifically,

we target the implementation on the Intell® Xeon Phi™ (Intel Phi) many-core coprocessor, but we also evaluate its performance on multi-core CPUs and on a cooperative execution model that uses both processors with several optimizations. The experimental results show that we were able to attain high IP lookup throughputs — up to 182.7 Mlps (million packets per second) or 119.9 Gbps for 84B IPv6 packets — on a single Intel Phi. This performance indicates that the Intel Phi is a very promising platform for deployment of IP lookup algorithms.We have also compared the BFs algorithm to an efficient approach based on the Multi-Index Hybrid Trie (MIHT) in which the BFs algorithm was up to 5.39x faster. This comparison shows that the most efficient sequential algorithm may not be the best option in a parallel setting. Instead, it is necessary to evaluate the processors characteristics, algorithms compute/data demands, and data structures employed to analyze how the algorithms will benefit from parallel computing devices, potential limitations on scalability and opportunities for optimizations. These findings are also important to new efforts in algorithmic developments in the topic, which have been highly focused on sequential solutions.
Keywords: IP Lookup, Software Routers, Intel® Xeon Phi™, Longest Prefix Matching.

Performance Prediction of Applications Executed on GPUs using a Simple Analytical Model and Machine Learning Techniques

Author: Marcos Tulio Amarís González
Advisors: Prof. Dr. Alfredo Goldman and Prof. Dr. Raphael Yokoingawa de Camargo

Thesis presented in partial fulfillment of the requirements for the degree of Doctor in Computer Science, University of São Paulo, Institute of Mathematics and Statistics (2018).

Abstract
The parallel and distributed platforms of High Performance Computing available today have become more and more heterogeneous (CPUs, GPUs, FPGAs, etc). Graphics Processing Units (GPU) are specialized co-processor to accelerate and improve the performance of parallel vector operations. GPUs have a high degree of parallelism and can execute thousands or millions of threads concurrently and hide the latency of the scheduler. GPUs have a deep hierarchical memory of different types as well as different configurations of these memories. Performance prediction of applications executed on these devices is a great challenge and is essential for the efficient use of resources in machines with these co-processors. There are different approaches for these predictions, such as analytical modeling and machine learning techniques. In this thesis, we present an analysis and characterization of the performance of applications executed on GPUs. We propose a simple and intuitive BSP-based model for predicting the CUDA application execution times on different GPUs. The model is based on the number of computations and memory accesses of the GPU, with additional information on cache usage obtained from profiling. We also compare three different Machine Learning (ML) approaches: Linear Regression, Support Vector Machines and Random Forests with BSP-based analytical model. This comparison is made in two contexts, first, data input or features for ML techniques were the same than analytical model, and, second, using a process of feature extraction, using correlation analysis and hierarchical clustering. We show that GPU applications that scale regularly can be predicted with simple analytical models, and an adjusting parameter. This parameter can be used to predict these applications in other GPUs. We also demonstrate that ML approaches provide reasonable predictions for different cases and ML techniques required no detailed knowledge of application code, hardware characteristics or explicit modeling. Consequently, whenever a large data set with information about similar applications are available or it can be created, ML techniques can be useful for deploying automated on-line performance prediction for scheduling applications on heterogeneous architectures with GPUs.
Keywords: Performance Prediction, Machine Learning, BSP model, GPU Architectures, CUDA.

Resumo
As plataformas paralelas e distribuídas de computação de alto desempenho disponíveis hoje se tornaram mais e mais heterogêneas (CPUs, GPUs, FPGAs, etc). As Unidades de processamento gráfico são co-processadores especializados para acelerar operações vetoriais em paralelo. As GPUs tem um alto grau de paralelismo e conseguem executar milhares ou milhões de threads concorrentemente e ocultar a latência do escalonador. Elas têm uma profunda hierarquia de memória de diferentes tipos e também uma profunda configuração da memória hierárquica. A predição de desempenho de aplicações executadas nesses dispositivos é um grande desafio e é essencial para o uso eficiente dos recursos computacionais de máquinas com esses co-processadores. Existem diferentes abordagens para fazer essa predição, como técnicas de modelagem analática e aprendizado de máquina. Nesta tese, nós apresentamos uma análise e caracterização do desempenho de aplicações executadas em Unidades de

Processamento Gráfico de propósito geral. Nós propomos um modelo simples e intuitivo fundamentado no modelo BSP para predizer a execução de funções kernels de CUDA sobre diferentes GPUs. O modelo está baseado no número de computações e acessos à memória da GPU, com informação adicional do uso das memórias caches obtidas do processo de profiling. Nós também comparamos três diferentes enfoques de aprendizado de máquina (ML): Regressão Linear, Máquinas de Vetores de Suporte e Florestas Aleatórias com o nosso modelo analítico proposto. Esta comparação é feita em dois diferentes contextos, primeiro, dados de entrada ou features para as técnicas de aprendizado de máquinas eram as mesmas que no modelo analítico, e, segundo, usando um processo de extração de features, usando análise de correlação e clustering hierarquizado. Nós mostramos que aplicações executadas em GPUs que escalam regularmente podem ser preditas com modelos analíticos simples e um parâmetro de ajuste. Esse parâmetro pode ser usado para predizer essas aplicações em outras GPUs. Nós também demonstramos que abordagens de ML proveem predições aceitáveis para diferentes casos e essas abordagens não exigem um conhecimento detalhado do código da aplicação, características de hardware ou modelagens explícita. Consequentemente, sempre e quando um banco de dados com informação de profiling esteja disponível ou possa ser gerado, técnicas de ML podem ser úteis para aplicar uma predição automatizada de desempenho para escalonadores de aplicações em arquiteturas heterogêneas contendo GPUs.

Palavras-chaves: Predição de Desempenho, Máquinas de Aprendizado, Modelo BSP, Unidades de Processamento Gráfico, CUDA.

Adaptive degree of parallelism for the spar runtime

Author: Adriano Vogel
Advisors: Prof. Luiz Gustavo Fernandes and Prof. Dalvan Griebler

Thesis submitted to the Pontifical Catholic University of Rio Grande do Sul in partial fulfillment of the requirements for the degree of Master in Computer Science. Pontifical Catholic University of Rio Grande do Sul, School of Technology, Computer Science Graduate Program (2018).

Abstract

In recent years, stream processing applications have become a traditional workload in computing systems. They are traditionally found in video, audio, graphic and image processing. Many of these applications demand parallelism to increase performance. However, programmers must often face the trade-off between coding productivity and performance that introducing parallelism creates. SPar Domain-Specific Language (DSL) was created to achieve the optimal balance for programmers, with the C++-11 attribute annotation mechanism to ensure that essential properties of stream parallelism could be represented (stage, input, output, and replicate). The compiler recognizes the SPar attributes and generates parallel code automatically. The need to manually define parallelism is the crucial challenge for increasing SPAR's abstraction level, because it is time consuming and error prone. Also,executing several applications can fail to be efficient when running a non-suitable number of replicas. This occurs when the defined number of replicas in a parallel region is not optimal or when a static number is used, which ignores the dynamic nature of stream processing applications. In order to solve this problem, we introduced the concept of the abstracted and adaptive number of replicas for SPar. Moreover, we described our implemented strategy as well as transformation rules that enable SPar to generate parallel code with the adaptive degree of parallelism support. We experimentally evaluated the implemented adaptive strategies regarding their effectiveness. Thus, we used real-world applications to demonstrate that our adaptive strategy implementations can provide higher abstraction levels without significant performance degradation.
Keywords: Stream Parallelism, Abstracted and Adaptive Degree of Parallelism

Resumo

As aplicações de stream se tornaram cargas de trabalho representativas nos sistemas computacionais. Diversos domínios de aplicações representam stream, como vídeo, áudio, processamento de gráficos e imagens. Uma parcela significativa dessas aplicações demanda paralelismo para aumentar o desempenho. Porém, programadores frequentemente enfrentam desafios entre produtividade de código e desempenho devido às complexidades decorrentes do paralelismo. Buscando facilitar a paralelização, a DSL SPar foi criada usando atributos (stage, input, output, and replicate) do C++-11 para representar paralelismo. O compilador reconhece os atributos da SPar e gera código paralelo automaticamente. Um desafio relevante que ocorre em estágios paralelos da SPar é a demanda pela definição manual do grau de paralelismo, o que consome tempo e pode induzir a erros. O grau de paralelismo é definido através do número de réplicas em estágios paralelos. Porém, a execução de diversas aplicações pode ser pouco eficiente se executadas comum número de réplicas inadequado ou usando um número estático que ignora a natureza dinâmica de algumas aplicações. Para resolver esse problema, é

introduzido o conceito de um número de réplicas transparente e adaptativo para a SPar. Além disso, os mecanismos implementados e as regras de transformação são descritos para possibilitar a geração de código paralelo na SPar com um número adaptativo de réplicas. Os mecanismos adaptativos foram avaliados demonstrando a sua eficácia. Ainda, aplicações reais foram usadas para demonstrar que os mecanismos adaptativos propostos podem oferecer abstrações de alto nível sem significativas perdas de desempenho.
Palavras-Chave: Paralelismo de Stream, Paralelismo Adaptativo e Abstrato.

Alinhamento Primário e Secundário de Sequências Biológicas em Arquiteturas de Alto Desempenho

Autor: Daniel Sundfeld Lima
Orientadora: Prof. Dr. Alba Cristina Magalhães Alves de Melo

Tese apresentada como requisito parcial para conclusão do Doutorado em Informática, Instituto de Ciências Exatas, Departamento de Ciência da Computação, Universidade de Brasília (2017).

Resumo

O alinhamento múltiplo primário de sequências biológicas é um problema muito importante em Biologia Molecular, pois permite que sejam detectadas similaridades e diferenças entre um conjunto de sequências. Esse problema foi provado NP-Completo e, por essa razão, geralmente algoritmos heurísticos são usados para resolvê-lo. No entanto, a obtenção da solução ótima é bastante desejada e, por essa razão, existem alguns algoritmos exatos que solucionam esse problema para um número reduzido de sequências. As sequências de RNA, diferente do DNA, não possuem dupla-hélice e podem dobrar-se, pois seus nucleotídeos podem formar pares de bases. É conhecido na Biologia Molecular que a função dessa estrutura está ligada à sua conformação espacial, e não à composição de seus nucleotídeos. Obter a estrutura secundária (2D) de uma sequência de RNA também exige uma grande quantidade de recursos computacionais, até mesmo para um pequeno número de sequências. Desta forma, as arquiteturas de alto desempenho são muito importantes para a obtenção dos resultados em um tempo factível. A presente tese visa investigar os problemas do alinhamento múltiplo primário e do alinhamento em pares secundário, utilizando arquiteturas de alto desempenho para acelerar a obtenção de resultados. Para o alinhamento primário ótimo de múltiplas sequências, propusemos na presente Tese o PA-Star, uma estratégia multithreaded baseada no algoritmo A-Star que usa uma política sensível à localidade de atribuição de trabalho às threads. De modo a lidar com o alto uso de memória, nossa estratégia PA-Star usa tanto memória RAM como disco. Para o alinhamento estrutural (2D) de sequências de RNA, propusemos o Foldalign 2.5, que é uma estratégia multithreaded heurística baseada no algoritmo exato de Sankoff, capaz de obter o alinhamento estrutural de grandes sequências em tempo reduzido. Finalmente, propusemos o CUDA-Sankoff, que é capaz de obter o alinhamento estrutural ótimo entre duas sequências de RNA em GPU (Graphics Processing Unit).
Palavras-chave: Alinhamento de Sequências, A-Estrela, Algoritmo de Sankoff, GPU

Abstract

The primary multiple sequence Alignment is a very important problem in Molecular Biology since it is able to detect similarities and differences in a set of sequences. This problem has been proven NP-Hard and, for this reason, heuristic algorithms are usually used to solve it. Nevertheless, obtaining the optimal solution is highly desirable and there are indeed some exact algorithms that solve this problem for a reduced number of sequences. The RNA sequences are different than the DNA, they do not have double helix, their nucleotides can form base pairs and the sequence can fold on itself. It is known in the Molecular Biology that, the function of the RNA is related to its spatial structure. Calculating the secondary structure of RNA sequences also demand a high amount of computational resources, even for a small number of sequences. The High Performance Computing (HPC) Platforms can be used in order to produce results faster. The current thesis aims to investigate the primary multiple sequence alignment

and the secondary pairwise sequence alignment, using High Performance Architectures to accelerate and obtaining results in reasonable time. For the primary multiple sequence alignment, we propose the PA-Star, a multithreaded solution based on the A-Star algorithm using a locality sensitive hash to distribute the workload among the threads. Due to the high RAM memory usage required by the algorithm, our strategy can also uses disk. For the RNA structural alignment, we proposed the Foldalign 2.5, a multithreaded solution that uses heuristics to reduce the Sankoff Algorithm complexity, and can obtain the pairwise structural alignment of large sequences in reduced time. Finally, we proposed CUDA-Sankoff, that obtains the optimal pairwise structural alignment for RNA sequences using a GPU (Graphics Processing Unit).
Keywords: Sequence Alignment, A-Star, Sankoff Algorithm, GPU

Leveraging the Entity Matching Performance through Adaptive Indexing and Efficient Parallelization

Author:Demetrio Gomes Mestre
Advisor: Prof. Dr. Carlos Eduardo Santos Pires

Thesis presented in partial fulfillment of the requirements for the degree of Doctor in Computer Science, Universidade Federal de Campina Grande, Centro de Engenharia Elétrica e Informática Coordenação de Pós-Graduação em Ciência da Computação (2018).

Abstract

Entity Matching (EM), i.e., the task of identifying all entities referring to the same real-world object, is an important and difficult task for data sources integration and cleansing. A major difficulty for this task performance, in the Big Data era, is the quadratic nature of the task execution. To minimize the workload and still maintain high levels of matching quality, for both single or multiple data sources, the indexing (blocking) methods were proposed. Such methods work by partitioning the input data into blocks of similar entities, according to an entity attribute, or a combination of them, commonly called “blocking key”, and restricting the EM process to entities that share the same blocking key (i.e., belong to the same block). In spite to promote a considerable decrease in the number of comparisons executed, indexing methods can still generate large amounts of comparisons, depending on the size of the data sources involved and/or the number of entities per index (or block). Thus, to further minimize the execution time, the EM task can be performed in parallel using programming models such as MapReduce and Spark. However, the effectiveness and scalability of MapReduce and Spark-based implementations for data-intensive tasks depend on the data assignment made from map to reduce tasks, in the case of MapReduce, and the data assignment between the transformation operations, in the case of Spark. The robustness of this assignment strategy is crucial to achieve skewed data handling (large sets of data can cause memory bottlenecks) and balanced workload distribution among all nodes of the distributed infrastructure. Thus, considering that studies about approaches that perform the efficient execution of adaptive indexing EM methods, in batch or real-time modes, in the context of parallel computing are an open gap according to the literature, this work proposes a set of parallel approaches capable of performing efficient adaptive indexing EM approaches using MapReduce and Spark in batch or real-time modes. The proposed approaches are compared to state-of-the-art ones in terms of performance using
real cluster infrastructures and data sources. The results carried so far show evidences that the performance of the proposed approaches is significantly increased, enabling a decrease in the overall runtime while preserving the quality of similar entities detection.
Keywords: Entity Matching, EM Indexing, real-time EM, Parallel Computing, Load Balancing, MapReduce, Spark.

Resumo

Entity Matching (EM), ou seja, a tarefa de identificar entidades que se referem a um mesmo objeto do mundo real, é uma tarefa importante e difícil para a integração e limpeza de fontes de dados. Uma das maiores dificuldades para a realização desta tarefa, na era de Big Data, é o tempo de execução elevado gerado pela natureza quadrática da execução da tarefa. Para minimizar a carga de trabalho preservando a qualidade na detecção de entidades similares, tanto para uma ou mais fontes de dados, foram propostos os chamados métodos de indexação ou blocagem. Estes métodos particionam o conjunto de dados em subconjuntos (blocos) de entidades potencialmente similares, rotulando-as com chaves de bloco, e restringem a execução da tarefa de EM entre entidades pertencentes ao mesmo bloco. Apesar

de promover uma diminuição considerável no número de comparações realizadas, os métodos de indexação ainda podem gerar grandes quantidades de comparações, dependendo do tamanho dos conjuntos de dados envolvidos e/ou do número de entidades por índice (ou bloco). Assim, para reduzir ainda mais o tempo de execução, a tarefa de EM pode ser realizada em paralelo com o uso de modelos de programação tais como MapReduce e Spark. Contudo, a eficácia e a escalabilidade de abordagens baseadas nestes modelos depende fortemente da designação de dados feita da fase de map para a fase de reduce, para o caso de MapReduce, e da designação de dados entre as operações de transformação, para o caso de Spark. A robustez da estratégia de designação de dados é crucial para se alcançar alta eficiência, ou seja, otimização na manipulação de dados enviesados (conjuntos de dados grandes que podem causar gargalos de memória) e no balanceamento da distribuição da carga de trabalho entre os nós da infraestrutura distribuída. Assim, considerando que a investigação de abordagens que promovam a execução eficiente, em modo batch ou tempo real, de métodos de indexação adaptativa de EM no contexto da computação distribuída ainda não foi contemplada na literatura, este trabalho consiste em propor um conjunto de abordagens capaz de executar a indexação adaptativas de EM de forma eficiente, em modo batch ou tempo real, utilizando os modelos programáticos MapReduce e Spark. O desempenho das abordagens propostas é analisado em relação ao estado da arte utilizando infraestruturas de cluster e fontes de dados reais. Os resultados mostram que as abordagens propostas neste trabalho apresentam padrões que evidenciam o aumento significativo de desempenho da tarefa de EM distribuída promovendo, assim, uma redução no tempo de execução total e a preservação da qualidade da detecção de pares de entidades similares.

Palavras-chave: Entity Matching, Métodos de indexação de EM, EM em tempo real, Computação Paralela, Balanceamento de Carga, MapReduce, Spark.

Uma adaptação do escalonador estático da biblioteca dataflow Sucuri para computação in-situ

Autor: Caio Bonfatti Gomes de Carvalho
Orientadores: Felipe Maia Galvão França e Leandro Augusto Justen Marzulo

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, Universidade Federal do Rio de Janeiro (2018).

Resumo

No modelo de computação por fluxo de dados, dataflow, as instruções ou tarefas são executadas de acordo com as dependências de dados, ao invés de seguir a ordem do programa, permitindo a exploração natural de paralelismo. A Sucuri é uma biblioteca dataflow em Python que permite aos usuários especificarem suas aplicações como um grafo de dependências, e as executa de maneira transparente em clusters de multicores, enquanto se encarrega também dos problemas de escalonamento. Avanços recentes em computação fog e in-situ assumem que dispositivos de armazenamento e de rede serão equipados com unidades de processamento, geralmente de baixo consumo de energia e baixo desempenho. Uma decisão importante em tais sistemas é a de migrar os dados para os processadores tradicionais, pagando os custos de comunicação, ou realizar a computação onde o dado está (in-situ), usando um processador de desempenho potencialmente menor. Este trabalho apresenta um estudo dos diferentes fatores que devem ser levados em consideração quando executando aplicações dataflow em um ambiente de computação in-situ. A Sucuri foi utilizada para gerenciar a execução em um pequeno sistema composto de um computador comum e uma placa Paralela simulando um disco inteligente, e uma série de experimentos foi realizada para mostrar como o tamanho dos dados a serem transferidos, a latência de rede, a perda de pacotes e a complexidade computacional da aplicação afetam o tempo de execução quando o processamento é deixado a cargo desse disco. Então, uma solução de escalonamento estático que leva em conta tais fatores é apresentada, dando a Sucuri poder de decisão sobre onde melhor realizar o processamento dos dados, evitando fazer a computação in-situ quando a mesma não trouxer ganhos de performance.
Palavras-chaves: Dataflow, In-situ, Edge, Fog, Escalonamento, Smart Storage.

Abstract

In the dataflow computation model, instructions or tasks are executed according to data dependencies, instead of following program order, thus allowing parallelism to be exposed naturally. Sucuri is a dataflow library for Python that allows each user to specify their application as a dependency graph and execute it transparently in clusters of multicores, while taking care of scheduling issues. Recent trends in Fogand In-situ computing assume that storage and network devices will be equipped with processing elements that usually have lower power consumption and performance. An important decision for such systems is whether to move data to traditional processors (paying the communication costs), or to perform the computation where the data sits, using a potentially slower processor. Hence, runtime environments that deal with that trade-off are of extreme importance. This work presents a study on different factors that should be considered when running dataflow applications in a In-situ environment. We use Sucuri to manage the execution in a small system with a regular PC and a Parallella board, emulating a smart storage,and a set of experiments was performed to show how data transfer size, network

latency, packet loss rates and computational complexity affect execution time when outsourcing computation to the smart storage. Then, a static scheduling solution is presented, allowing Sucuri to take the best decision where to execute the application,avoiding outsourcing when there would be no performance gains.

Escalonamento de Tarefas e Alocação de Arquivos de Dados de Workflows Científicos em Nuvens Computacionais

Author: Luan Teylo Gouveia Lima
Orientadores: Lúcia Maria de Assumpção Drummond e Yuri Abitbol de Menezes Frota

Dissertação de mestrado apresentada ao Programa de Pós-Graduação em Computação da Universidade Federal Fluminense como requisito parcial para a obtenção do Grau de Mestre em Computação. Área de concentração: Sistemas de Computação (2017).

Resumo

Na última década, um número crescente de experimentos computacionalmente intensivos envolvendo grandes volumes de dados têm sido modelados na forma de workflows científicos. Ao mesmo tempo, as nuvens computacionais surgem como um ambiente promissor para executar esse tipo de aplicação. Neste cenário, a investigação de estratégias de escalonamentos se tornaram essenciais, sendo este um campo de pesquisa extremamente popular. No entanto, poucos trabalhos consideram o problema da alocação de dados durante a resolução do problema de escalonamento de tarefas. Um workflow é geralmente representado como um grafo, no qual os nós equivalem às tarefas e, neste caso, o problema de escalonamento consiste em alocar essas tarefas a máquinas que as executarão em um tempo pré definido. O objetivo é reduzir o tempo total de execução de todo o workflow. Neste trabalho é mostrado que o escalonamento de workflows científicos pode ser melhorado quando os problemas de escalonamento de tarefa e alocação de dados são tratados de forma conjunta. Para isso, uma nova representação, na qual os nós do grafo representam tanto tarefas como dados, é proposta. Além disso, o problema de Escalonamento de Tarefas e Alocação de Dados é definido, considerando esse novo modelo. Esse problema foi formulado como um problema de programação inteira. Por fim, um algoritmo evolucionário híbrido capaz de escalonar tarefas e alocar os dados em ambientes de nuvens computacionais também é apresentado.
Palavras-chave: Problema de escalonamento, Alocação de Dados, Workflow Científico, Metaheurística.

Abstract

A growing number of data- and compute-intensive experiments have been modeled as scientific workflows in the last decade. Meanwhile, clouds have emerged as a prominent environment to execute this type of applications. In this scenario, the investigation of workflow scheduling strategies, aiming at reducing its execution times, became a top priority and a very popular research field. However, few works consider the problem of data file assignment when solving the task scheduling problem. Usually, a workflow is represented by a graph where nodes represent tasks and the scheduling problem consists in allocating tasks to machines to be executed at a predefined time aiming at reducing the makespan of the whole workflow. In this work, we show that the scheduling of scientific workflows can be improved when both task scheduling and the data file assignment problems are treated together. Thus, we propose a new workflow representation, where nodes of the workflow graph represent either tasks or data files, and define the Task Scheduling and Data Allocation Problem, considering this new model. We formulated this problem as an integer programming problem. Moreover, a hybrid evolutionary algorithm for solving it is also introduced.
Keywords: Scheduling Problem, Data Allocation, Scientific Workflow, Metaheuristic.

Lista de Autores