UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM COMPUTAÇÃO

GABRIEL DILLENBURG MARTINS

Desempenho de aplicações OpenMP em ambientes HPC: uma abordagem detalhada com rastreamento de tarefas

Proposta de Projeto de Pesquisa

Orientador: Prof. Dr. Lucas Mello Schnorr

Porto Alegre
2024

SUMÁRIO

1 INTRODUÇÃO

1.1 High Performance Computing (HPC)

High Performance Computing (HPC) é um campo da ciência da computação focado na busca por soluções para problemas complexos que exigem poder computacional massivo. Essa área tem um impacto significativo em diversas áreas de aplicação, como física, biomedicina, engenharia e inteligência artificial (Chapman; Jost; Pas, 2007). Os supercomputadores, máquinas que representam a vanguarda da tecnologia computacional, são ferramentas essenciais para HPC. Eles reúnem milhares de unidades de processamento em um único sistema, possibilitando a execução de tarefas que antes eram muito difíceis (Geimer et al., 2010).

No centro do HPC reside a busca incessante por alto desempenho computacional. Para alcançar esse objetivo, é necessário explorar ao máximo o potencial dos recursos de hardware disponíveis. Arquiteturas com diversas unidades de processamento, presentes em supercomputadores modernos, oferecem uma oportunidade para aumentar o desempenho através da execução simultânea de múltiplas tarefas pertecentes a mesma aplicação.

1.2 OpenMP e Programação Paralela

1.2.1 OpenMP

Nesse contexto, o OpenMP (Open Multi-Processing) surge como um modelo de programação paralela de grande relevância para HPC (Dagum; Menon, 1998). Através de diretivas de programação simples e intuitivas, o OpenMP permite que os programadores tirem proveito da capacidade de processamento multi-core, dividindo tarefas em unidades independentes que podem ser executadas simultaneamente (Schmidl et al., 2013).

No desenvolvimento de aplicações que utilizam OpenMP, uma análise de desempenho eficaz é fundamental para identificar gargalos de desenmpenho e otimizar a execução paralela. A interface OMPT (OpenMP Tools Interface) representa uma evolução significativa nesse aspecto, oferecendo um conjunto de ferramentas e métodos para registrar o estado de execução de programas OpenMP com baixa sobrecarga e intrusão. A OMPT habilita as ferramentas de desempenho a coletar informações detalhadas sobre a execução de aplicações e interpretar comportamentos em nível de usuário, ocultando

particularidades de implementações específicas do OpenMP. Por meio de funcionalidades como o rastreamento de estado em tempo de execução, callbacks e funções de consulta, a OMPT permite uma análise precisa de desempenho, atribuindo custos tanto ao código fonte da aplicação quanto ao sistema de runtime do OpenMP. Esta capacidade é essencial para a otimização de aplicações paralelas, pois permite aos desenvolvedores entender e melhorar aspectos críticos de desempenho, como a gestão de tarefas e a sincronização entre threads (Eichenberger et al., 2013).

1.2.2 Paralelismo por Tarefas e Adaptabilidade

Uma das principais técnicas de paralelismo utilizadas em aplicações OpenMP é o paralelismo por tarefas (Chapman; Jost; Pas, 2007). Essa abordagem divide o trabalho em unidades de execução dependentes, denominadas tarefas, que podem ser facilmente mapeadas para diferentes núcleos de processamento. Essa flexibilidade permite que as aplicações se adaptem a diferentes configurações de hardware, otimizando o desempenho e a eficiência do processamento. Outro ponto crucial é a adaptabilidade em aplicações HPC, pois permite que elas se ajustem a flutuações na disponibilidade de recursos computacionais. Essa característica é fundamental para garantir a eficiência e o bom funcionamento das aplicações em ambientes dinâmicos e em constante mudança. Dominar a arte do HPC envolve a integração harmoniosa de todos esses elementos: alto desempenho computacional, processadores multi-core, OpenMP, paralelismo por tarefas e adaptabilidade. Através do conhecimento nesse conjunto de habilidades, cientistas e programadores podem explorar o poder dos supercomputadores para realizar computação avançada, facilitando a simulação, modelagem e análise de dados em grande escala. Esse processo contribui diretamente para a aceleração de pesquisas, o desenvolvimento de novas tecnologias e a resolução de questões complexas em várias disciplinas científicas.

1.3 Motivação

No desenvolvimento de aplicações paralelas com OpenMP, a necessidade de registrar e analisar o comportamento das tarefas se torna crucial para identificar gargalos de desempenho. A granularidade das tarefas, o desempenho geral da aplicação e a adequação do escalonamento de tarefas são aspectos que precisam ser cuidadosamente avaliados

para garantir o melhor desempenho possível. Este trabalho procura estabelecer uma metodologia para rastreamento de aplicações paralelas com OpenMP, utilizando a interface OMPT. Através da coleta e análise detalhada de dados sobre a execução das tarefas, a metodologia proposta oferece um fluxo de trabalho completo para rastrear tarefas e analisar o desempenho de aplicações.

1.4 Objetivos

O objetivo central deste projeto de pesquisa é desenvolver e propor uma metodologia para o rastreamento de diretivas de tarefas OpenMP, com o intuito de analisar e identificar possíveis gargalos de desempenho em aplicações paralelas. Para alcançar este objetivo, a interface OMPT será utilizada para coletar dados detalhados sobre a execução de tarefas OpenMP ao longo do tempo. Essa coleta é crucial para entender como as tarefas se executam e interagem dentro de uma aplicação.

Os dados coletados serão submetidos a uma análise meticulosa, incluindo diagramas de Gantt que serão utilizados para visualizar a sequência de execução e a duração das tarefas, auxiliando na identificação de gargalos de desempenho.

Através da interface OMPT, análise de tempo de execução das tarefas nas aplicações também será realizada, calculando indicadores como média, mediana e desvio padrão. Isso permitirá identificar tarefas com tempos excessivos e possíveis desequilíbrios de carga.

Por fim, as métricas serão coletadas de programas internos do benchmark KASTORS (GitLab, 2017), como Jacobi, Plasma, Sparselu e Strassen, os quais serão executados em diferentes máquinas e comparados. Essa comparação é fundamental para destacar discrepâncias no desempenho que podem surgir devido a diferenças arquitetônicas, permitindo identificar possíveis problemas de desempenho para cada arquitetura.

2 TRABALHOS RELACIONADOS

No âmbito do HPC, a utilização do OpenMP para processamento paralelo tem recebido significativa atenção. Nossa investigação se baseia em vários estudos fundamentais que contribuíram para a otimização e avaliação de aplicações OpenMP. Notavelmente, o trabalho de Virouleau et al. (2014) fornece uma análise perspicaz de como as dependências de tarefas podem ser gerenciadas para melhorar o desempenho de cálculos paralelos em ambientes OpenMP 4.0. Suas descobertas sublinham a importância de mecanismos de sincronização de tarefas eficientes na exploração eficaz de arquiteturas multi-core.

Ademais, a pesquisa apresentada por Chatarasi, Shirako and Sarkar (2015) introduz uma abordagem inovadora para a otimização de programas paralelos, incluindo aqueles desenvolvidos com OpenMP. Ao integrar modelos poliédricos (Feautrier; Lengauer, 2011), eles demonstram caminhos potenciais para alcançar níveis mais altos de eficiência computacional e escalabilidade, aspectos cruciais das simulações HPC.

Além disso, a ferramenta "ompTG: De Programas OpenMP para Grafos de Tarefas", desenvolvida por Sun et al. (2022), representa um avanço significativo para preencher a lacuna entre a pesquisa teórica e o desenvolvimento prático de software utilizando OpenMP. Esta ferramenta facilita a transformação de programas OpenMP em grafos de tarefas, oferecendo uma visão mais detalhada das dependências de tarefas e estruturas de controle de fluxo. Tal perspectiva é importante para a avaliação e otimização de algoritmos de agendamento em tempo real em aplicações HPC.

Esses estudos, coletivamente, enriquecem nossa compreensão dos desafios e oportunidades na otimização de aplicações OpenMP para ambientes HPC. Eles não apenas destacam os avanços no paralelismo e sincronização de tarefas, mas também oferecem ferramentas e metodologias práticas para melhorar o desempenho de simulações HPC. Nossa pesquisa visa construir sobre essas bases, explorando novas vias para identificar problemas de desempenho de simulações HPC baseadas em OpenMP.

3 METODOLOGIA

Na metodologia proposta para o estudo do rastreamento de aplicações paralelas utilizando OpenMP, nossa investigação focará na eficiência da aplicação e na granularidade das tarefas em ambientes HPC. Esta abordagem combina a análise de benchmarks específicos com KASTORS e a avaliação detalhada do desempenho sob diversas configurações de hardware, conforme apresentado na Tabela 3.1. A seleção dos benchmarks para este estudo incluirá os recomendados por Virouleau et al. (2014), que evidenciam a importância de utilizar benchmarks representativos para avaliar o desempenho de aplicações HPC. Para a execução dos experimentos, utilizaremos um ambiente de HPC que fornece uma ampla gama de configurações de núcleos e capacidades de processamento. A seleção das máquinas para este estudo, permite uma análise comparativa do desempenho das aplicações sob diferentes cargas de trabalho e estão detalhadas conforme Seção 3.2. Esta seleção é apoiada pela pesquisa de Schmidl et al. (2013), que destaca a relevância de avaliar o desempenho de aplicações em HPC em diversas configurações de hardware.

A configuração de software para a coleta de dados utilizará a versão mais atual do compilador GCC com suporte ao OpenMP, assegurando a disponibilidade das funcionalidades mais recentes do OMPT para o rastreamento, conforme recomendado por Dagum and Menon (1998). Este aspecto é crucial para a coleta de dados eficaz, como ilustrado pelo trabalho de Geimer et al. (2010), que enfatiza os avanços nas ferramentas de rastreamento para facilitar a análise detalhada do comportamento das tarefas em execução.

Nosso design experimental focará em testes específicos para cada benchmark, incluindo análises de tempo de execução, visualizações de gráficos de tarefas e estatísticas detalhadas do desempenho das tarefas ao longo do tempo. Este plano é inspirado na abordagem sugerida por Chapman, Jost and Pas (2007), demonstrando como uma análise meticulosa do comportamento das tarefas pode revelar percepções valiosas para possíveis otimizações de aplicações paralelas.

Em resumo, a metodologia deste estudo visa fornecer uma compreensão do desempenho de aplicações paralelas em ambientes HPC. Utilizando uma combinação de análise de benchmarks, avaliação de configurações de hardware e técnicas de rastreamento com OMPT, buscamos identificar possíveis problemas de desempenho e estratégias de paralelismo que possam ser aplicadas para melhorar a eficiência de aplicações que utilizam OpenMP.

3.1 Estudos de caso

A suite de benchmarks KASTORS, introduzida por Virouleau et al. (2014), é projetada para avaliar o desempenho das implementações de tarefas OpenMP em diferentes sistemas de runtime e arquiteturas. Os benchmarks já implementados do KASTORS visam especificamente explorar os recursos de paralelismo baseado em tarefas do OpenMP 4.0 e versões posteriores, oferecendo um conjunto diversificado de aplicações e kernels que utilizam tarefas assíncronas e dependências de tarefas. A inclusão do KASTORS na avaliação de desempenho é justificada pela sua abrangente cobertura do paradigma de tarefas e implementações de benchmarks com aplicações conhecidas, as quais são detalhadas na seção 3.1.1.

3.1.1 Benchmarks

No contexto do HPC, a seleção de benchmarks da suite KASTORS, com Jacobi, Plasma, Sparselu, e Strassen, desempenha um papel crucial em avaliar a eficácia das implementações OpenMP. Os benchmarks selecionados abordam desafios específicos dentro do paradigma da computação paralela, relacionados a problemas contemporâneos da comunidade científica, como álgebra linear densa, análise de métodos numéricos iterativos e multiplicação de matrizes.

O benchmark Plasma é particularmente relevante no âmbito da transição da biblioteca numérica PLASMA (PLASMA. . . , 2024) para o padrão OpenMP, como evidenciado por YarKhan et al. (2017). Este esforço destaca a necessidade crítica de otimizar a execução paralela de algoritmos de álgebra linear densa em processadores multi-core, aproveitando a gestão eficiente de dependências de dados para maximizar a eficiência computacional.

Em contrapartida, Jacobi e Sparselu representam benchmarks para a análise de métodos numéricos iterativos e operações com matrizes esparsas, respectivamente (Lee et al., 2016). O benchmark Jacobi, ao focar na resolução de sistemas lineares, exemplifica a aplicação de paralelismo em métodos iterativos, crucial para a solução de problemas complexos em física e engenharia. Por outro lado, Sparselu enfatiza a relevância da otimização de algoritmos para a fatorização LU de matrizes esparsas, um componente chave em muitas aplicações científicas que requerem a manipulação eficiente de estruturas de dados esparsas para acelerar a computação.

Já algoritmo de strassen é reconhecido por sua abordagem otimizada à multiplicação de matrizes, que ilustra a busca contínua por eficiência algorítmica (Kouya, 2020). A inclusão de Strassen na suite KASTORS enfatiza a importância de algoritmos avançados que reduzem a complexidade computacional, uma meta alinhada com os objetivos de pesquisa em HPC, que visa superar as barreiras de desempenho inerentes aos métodos convencionais.

Portanto, a escolha destes benchmarks sublinha uma abordagem sistemática para a avaliação de desempenho em HPC. Cada benchmark foi selecionado não apenas pela sua relevância teórica, mas também pela sua aplicabilidade prática na resolução de problemas de computação paralela. Através desta análise, a suite KASTORS serve como uma ferramenta para o avanço do conhecimento em HPC, promovendo uma investigação das estrat'egias e práticas de programação paralela.

3.2 Plataformas de Simulação do Parque Computacional de Alto Desempenho (PCAD) UFRGS

No contexto do Parque Computacional de Alto Desempenho (PCAD) (Parque. . . , 2024), diversas plataformas destacam-se pela sua configuração técnica específica, adaptando-se a diferentes tipos de aplicações paralelas de maior porte. Neste trabalho, focaremos na execução de simulações em quatro partições distintas dentro do PCAD, cada uma oferecendo um conjunto único de recursos e capacidades. As plataformas selecionadas são:

Tabela 3.1 – Plataformas selecionadas para execução de simulações

Plataforma	**Núcleos**	**Configuração**
Cei	24 núcleos	2 x Intel Xeon Silver 4116 Skylake (Q3'17), 2.1 GHz, 48 threads
Blaise	44 núcleos	2 x Intel Xeon E5-2699 v4 Broadwell (Q1'16), 2.2 GHz, 88 threads
Hype	20 núcleos	2 x Intel Xeon E5-2650 v3 Haswell (Q3'14), 2.3 GHz, 40 threads
Draco	16 núcleos	2 x Intel Xeon E5-2640 v2 Ivy Bridge (Q3'13), 2.0 GHz, 32 threads

Fonte: O Autor

Cada uma destas plataformas foi selecionada com o propósito de abordar diferentes aspectos e desafios no campo do HPC. A seleção abrange desde configurações com maior número de núcleos, propícias para aplicações altamente paralelizáveis, até opções que oferecem um equilíbrio entre desempenho e consumo energético. Esta diversidade permite não apenas a execução eficiente de uma ampla gama de simulações, mas também facilita uma análise comparativa detalhada sobre como diferentes configurações de

hardware influenciam o desempenho de aplicações paralelas.

Através da execução de simulações nestas plataformas específicas, é possível extrair percepções valiosas sobre a escalabilidade, eficiência e adaptabilidade de diferentes abordagens de paralelização frente aos desafios computacionais em HPC.

3.3 Cronograma do Projeto de Pesquisa

O cronograma proposto para este projeto de pesquisa, focado no rastreamento de aplicações paralelas utilizando OpenMP, inicia com uma fase de revisão da literatura. Esta etapa inicial é crucial para fundamentar o estudo nas descobertas e nas práticas já estabelecidas na área, estabelecendo uma base sólida para a pesquisa. Seguindo a revisão literária, o projeto avança por fases marcadas por objetivos específicos, culminando na redação do relatório final. A seguir o cronograma proposto:

Tabela 3.2 – Cronograma de Atividades para o Estudo do Rastreamento de Aplicações Paralelas com OpenMP

Mês	**Atividade**
Abril 2024	Revisão de literatura sobre HPC, OpenMP, OMPT, e benchmarks. Preparação do ambiente de teste. Seleção e preparação dos benchmarks.
Maio 2024	Execução dos benchmarks na primeira configuração de hardware (Plataforma Draco). Análise inicial dos dados. Início dos testes na segunda configuração de hardware (Plataforma Hype).
Junho 2024	Continuação dos testes nas configurações de hardware restantes (Plataformas Bleise e Cei). Análise dos dados de todas as configurações. Preparação do relatório inicial dos resultados.
Julho 2024	Finalização do relatório de resultados. Redação e revisão do artigo de pesquisa. Submissão do artigo para revisão.

Fonte: O Autor

REFERÊNCIAS

CHAPMAN, B.; JOST, G.; PAS, R. van der. **Using OpenMP: Portable Shared Memory Parallel Programming**. [S.l.]: The MIT Press, 2007. (Scientific and Engineering Computation).

CHATARASI, P.; SHIRAKO, J.; SARKAR, V. Polyhedral optimizations of explicitly parallel programs. In: **2015 International Conference on Parallel Architecture and Compilation (PACT)**. [s.n.], 2015. p. 213–26. Available from Internet: <https://doi.org/10.1109/PACT.2015.44>.

DAGUM, L.; MENON, R. Openmp: an industry standard api for shared-memory programming. **IEEE Computational Science and Engineering**, v. 5, n. 1, p. 46–55, Jan 1998. Available from Internet: <https://doi.org/10.1109/99.660313>.

EICHENBERGER, A. et al. Ompt: An openmp tools application programming interface for performance analysis. In: **OpenMP in the Multi-Core Era**. [s.n.], 2013. v. 8122. Available from Internet: <https://doi.org/10.1007/978-3-642-40698-0_13>.

FEAUTRIER, P.; LENGAUER, C. Polyhedron model. In: PADUA, D. (Ed.). **Encyclopedia of Parallel Computing**. Boston, MA: Springer US, 2011. p. 1581–1592. Available from Internet: <https://doi.org/10.1007/978-0-387-09766-4_502>.

GEIMER, M. et al. The scalasca performance toolset architecture. **Concurrency and Computation: Practice and Experience**, v. 22, p. 702–719, 2010. Available from Internet: <https://doi.org/10.1002/cpe.1556>.

GITLAB. **Openmp / Kastors**. 2017. <https://gitlab.inria.fr/openmp/kastors>. Acessado em 19 de dezembro de 2017.

KOUYA, T. Performance evaluation of strassen matrix multiplication supporting triple-double precision floating-point arithmetic. In: GERVASI, O. et al. (Ed.). **Computational Science and Its Applications ICCSA 2020**. Cham: Springer International Publishing, 2020. p. 163–176. Available from Internet: <https://doi.org/10.1007/978-3-030-58814-4_12>.

LEE, J. et al. Openmp extension for explicit task allocation on numa architecture. In: MARUYAMA, N.; SUPINSKI, B. R. de; WAHIB, M. (Ed.). **OpenMP: Memory, Devices, and Tasks**. Cham: Springer International Publishing, 2016. p. 89–101. Available from Internet: <https://doi.org/10.1007/978-3-319-45550-1_7>.

PARQUE Computacional de Alto Desempenho (PCAD). 2024. <https://gppd-hpc.inf.ufrgs.br/>. Acessado em: 25 de Março de 2024.

PLASMA - QUARK: Queuing And Runtime for Kernels. 2024. <https://icl.utk.edu/plasma/docs/group__QUARK.html>. Acesso em 4 de abril de 2024.

SCHMIDL, D. et al. Assessing the performance of openmp programs on the intel xeon phi. In: WOLF, F.; MOHR, B.; MEY, D. an (Ed.). **Euro-Par 2013 Parallel Processing**. Berlin, Heidelberg: Springer, 2013. p. 547–558. Available from Internet: <https://doi.org/10.1007/978-3-642-40047-6_56>.

SUN, J. et al. omptg: From openmp programs to task graphs. **Journal of Systems Architecture**, v. 126, n. 1, p. 102470, May 2022. ISSN 1383-7621. Available from Internet: <https://doi.org/10.1016/j.sysarc.2022.102470>.

VIROULEAU, P. et al. Evaluation of openmp dependent tasks with the kastors benchmark suite. In: **International Conference on OpenMP in the Multi-core Era**. [s.n.], 2014. Available from Internet: <https://doi.org/10.1007/978-3-319-11454-5_2>.

YARKHAN, A. et al. Porting the plasma numerical library to the openmp standard. **International Journal of Parallel Programming**, v. 45, n. 3, p. 612–633, Jun 2017. Available from Internet: <https://doi.org/10.1007/s10766-016-0441-6>.