Proposta de tema
MAC0499 - Trabalho de Formatura Supervisionado

Aluno: Marcelo Nascimento Dos Santos Junior

Orientador: Alfredo Goldman vel Lejbman
Orientador: Pedro Henrique Rocha Bruel

Resumo

Neste projeto pretende-se explorar técnicas de computação de alto desempenho com o objetivo de paralelizar trechos de códigos na linguagem C. A API escolhida para esta tarefa foi o OpenMP e a configuração de cada trecho será realizada por meio de aprendizado de máquina. Diversos exemplos serão coletados para a fase de treinamento e ao final serão realizadas análises de desempenho dos resultados.

1 Introdução

O entendimento deste projeto passa por conceitos que serão abordados a seguir, a começar pelo aprendizado de máquina. Aprendizado de máquina (em inglês, *machine learning* ou ML) é um tipo de inteligência artificial que tem o objetivo de obter padrões com o auxílio de algum algoritmo que interprete os dados de uma aplicação. O foco é permitir que uma máquina consiga inferir um resultado futuro sem que esse estado tenha sido efetivamente programado.

O algoritmo recebe uma lista de exemplos de treinamento de casos anteriores, onde cada caso é composto por um conjunto de dados que geram seu respectivo resultado, ao analisar cada elemento da lista, o algoritmo tenta encontrar um padrão que será aplicado na inferência de um conjunto de dados novo que não está presente na lista de treinamento.

Existem várias componentes que influenciam no desempenho do algoritmo, dois que se destacam são a quantidade de variáveis envolvidas e a qualidade dos exemplos de treinamento. Os dados que geram um determinado resultado são divididos em variáveis, quanto maior a quantidade, mais combinações destas resultam no mesmo resultado, deste modo, aumenta o espaço de possíveis padrões que representam o comportamento da lista. A qualidade dos exemplos de treinamento também impacta o desempenho, visto que uma lista com poucos exemplos ou com vários casos parecidos não generaliza a aplicação.

A programação paralela é a divisão de uma determinada aplicação em diferentes partes a fim de serem executadas simultaneamente em vários elementos de processamento. Cada elemento deve cooperar com os outros de forma a substituir o modelo sequencial, isto se dá por meio de primitivas de comunicação e sincronização. Para este projeto, as tarefas serão

divididas em *threads*.

O OpenMP é uma interface de programação de aplicação (em inglês, *application programming interface* ou API) que se baseia no modelo de programação paralela de memória compartilhada e permite que o desenvolvedor insira configurações na forma de comentários em trechos de códigos com o intuito de informar ao OpenMP como executar a próxima instrução de forma sincronizada. As linhas de comentário se chamam diretivas e podem informar, por exemplo, como o trabalho deve ser compartilhado entre as *threads* e o escopo das variáveis envolvidas.

2 Motivação

Com o avanço da engenharia de *software*, as soluções implementadas estão cada vez mais complexas e demandam mais linhas de código. Otimizar qualquer tarefa em um ambiente com grande volume de informações pode ser extremamente custoso e muitas vezes é necessário recorrer a técnicas de aprendizado de máquina. O OpenMP é uma ferramenta prática para paralelizar trechos de códigos, suas diretivas são facilmente configuráveis e como se encontram na forma de comentários, o código original não é alterado. A ideia deste projeto é unir essas duas componentes com o auxílio de aprendizado de máquina.

3 Objetivos

A proposta é criar um método que analise o código de uma aplicação a fim de detectar um escopo de trabalho, neste caso, os *loops for*, e ao detectá-lo, converter seus atributos em *tokens* para que um modelo de inferência possa devolver as configurações necessárias para sua operação em paralelo. Isto só é possível, pois os *loops for* possuem variáveis indexadoras que associam o seu valor a cada iteração do *loop*, o que, por sua vez, são frequentemente associadas as tarefas contidas no bloco de instruções do *loop*, ao virtualizar essas variáveis, o bloco pode ser executado simultaneamente para cada valor possível do indexador.

4 Metodologia

A principal referência desse projeto é o artigo[1] "A machine learning method to variable classification in OpenMP", inicialmente, as ferramentas e dados utilizados serão seguidos, porém, podem ser alterados ao encontrar alternativas com desempenhos melhores.

A construção do projeto se inicia na compreensão do OpenMP e seu modo de compilação, como cada atributo pode impactar no desempenho do programa, quais problemas podem ocorrer com uma inferência incorreta, limitações e estrutura, disponíveis na documentação da ferramenta. O próximo passo é coletar um grande volume de exemplos de treinamento para que depois sejam comparados e analisados com as ferramentas de *Tokenização* e com os modelos de inferência, tarefa esta que ocorrerá até o final da monografia. Abaixo é apresentado um exemplo do resultado pretendido para um *loop*.

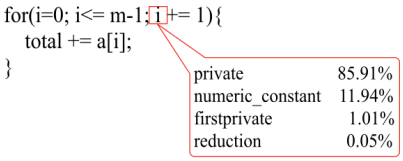

Figura 1: Exemplo da probabilidade de uma declaração de variável em relação aos atributos do OpenMP.

5 Cronograma

	Abr	Mai	Jun	Jul	Ago	Set	Out	Nov	Dez
1	x	x							
2	x	x							
3			x	x	x	x	x		
4			x	x	x	x	x	x	x
5								x	x

1. Compreender o funcionamento do OpenMP: como funcionam as diretivas e seu impacto no desempenho do código, opções disponíveis e problemas que podem ocorrer devido a uma inferência incorreta.
2. Estudar modelos de ML que se aplicam ao OpenMP: encontrar algoritmos que realizam as inferências das diretivas e as bibliotecas ou frameworks que os suportam. Esta tarefa deve ser alinhada com a tarefa 1 por se tratar dos fundamentos teóricos do projeto. Em dois meses é possível coletar a maioria das informações necessárias para iniciar a próxima fase.
3. Realizar experimentos: esta fase também inclui coletar códigos de treinamento e definir métricas de desempenho. O tempo difere da fase 4, pois ao início da monografia, não há mais tempo de fazer grandes alterações.
4. Definir modelos de ML e análise dos resultados: cada possível modelo e suas configurações devem ser enviados a fase 3 e analisados.
5. Monografia: As informações necessárias serão coletadas durante todas as fases anteriores, em dois meses é possível reagrupar e construir todo o conteúdo que atenda os requisitos do documento.

Referências

1. Yuanyuan Shen, Manman Peng, Qiang Wu, Renfa Li, A machine learning method to variable classification in OpenMP, Future Generation Computer Systems, Volume 140, 2023, Pages 67-78,https://doi.org/10.1016/j.future.2022.10.010.
2. Barbara Chapman, Gabriele Jost, Ruud van der Pas, Using OpenMP: Portable Shared Memory Parallel Programming, 2008