Proposta de Otimização do Tamanho de Batch em Aplicações de Stream para Multicores usando Aprendizado de Máquina

Claudio Scheer[1], **Dalvan Griebler**[1], **Luiz G. Fernandes**[1]

[1] Escola Politécnica, Grupo de Modelagem de Aplicações Paralelas (GMAP), Pontifícia Universidade Católica do Rio Grande do Sul (PUCRS), Porto Alegre, Brasil

claudio.scheer@edu.pucrs.br,dalvan.griebler@edu.pucrs.br

Resumo. *Este trabalho apresenta uma proposta de estudo e avaliação de features e algoritmos de aprendizado de máquina visando melhorar a desempenho através do ajuste/regulagem do tamanho do batch em aplicações paralelas de stream para arquiteturas multicore.*

Proposta de Pesquisa

Aplicações para processamento de *stream* são amplamente utilizadas. Formalmente, uma *stream* se caracteriza como um fluxo contínuo de dados, geralmente gerado por sensores, câmeras, entre outros [Andrade et al. 2014]. Na maioria dos casos, o processamento de *stream* está intimamente ligado com aplicações críticas, que processam e analisam dados em tempo real. Portanto, aplicações para processamento de *stream* precisam ser capazes de processar grande quantidade de dados com resposta rápida.

Uma aplicação de processamento paralelo de *stream* é estruturada como um *pipeline* de operadores, onde cada operador representa uma transformação que será aplicada sobre o dado proveniente da *stream*, também chamado de tupla. Determinados operadores precisam de um conjunto de dados finitos. Nestes casos, se usa a técnica de *batching*.

Batching consiste em agrupar tuplas com base em unidades de tempo ou quantidade de tuplas. Ao usar *batching*, a aplicação fica sensível ao *trade-off* entre latência e taxa de transferência. *Batches* menores tendem a diminuir a latência e o taxa de transferência. Por outro lado, *batches* maiores aumentam a latência e a taxa de transferência, reduzindo a capacidade da aplicação de reagir em tempo real aos dados recebidos pela *stream*.

Alguns trabalhos já propuseram algoritmos inteligentes que são capazes de encontrar o tamanho mais adequado de *batch*, considerando necessidades da aplicação de processamento de *stream*. [Das et al. 2014] propuseram um algoritmo para definir o tamanho do próximo *batch* com base nos últimos *batches* processados. [Zhang et al. 2016] usou regressão isotônica para definir o tamanho do *batch* de acordo com a carga de trabalho recebida pela aplicação. Similarmente, [Stein et al. 2020] propuseram a alteração do tamanho do *batch* com base na carga de trabalho e com o objetivo de atender os requisitos de latência da aplicação.

Diferente dos trabalhos já presentes na literatura, esta proposta foca no estudo de algoritmos de aprendizado de máquina que possam auxiliar a definição do melhor tamanho de *batch*. Como aplicações de processamento de *stream* geralmente são críticas, os modelos de aprendizado de máquina treinados precisam apresentar um *trade-off*

adequado entre acurácia e desempenho. Neste contexto, o desempenho do modelo de aprendizado de máquina é dado pelo custo para se fazer uma inferência.

O desempenho de algoritmos de aprendizado de máquina está diretamente ligado aos dados oferecidos como entrada para o treinamento do modelo. Portanto, para esta proposta também se fará a análise e a discussão das *features* que são mais adequadas para o treinamento de modelos de aprendizado de máquina no contexto de processamento de *stream*.

A proposta foca especificamente no processamento de *streams* em arquiteturas *multicore*. Portanto, além de *features* relacionadas à carga de trabalho e à aplicação, esta proposta também considerará *features* relacionadas à arquitetura disponível para o processamento de *streams*. Para o melhor do conhecimento do autores desta proposta, não há na literatura trabalho que faça uma análise quantitativa de quais *features* têm maior impacto no processamento de *streams* em arquiteturas *multicore* e também quais os algoritmos de aprendizado de máquina com melhor performance para aplicações críticas.

Os trabalhos presentes na literatura usualmente utilizam apenas *features* relacionadas à aplicação ou à carga de trabalho. Com uma avaliação mais ampla de *features*, incluindo o sistema *multicore* disponível, espera-se que o sistema tenha maior capacidade de adaptação à plataforma onde o processamento está sendo realizado, explorando melhor os recursos. Portanto, nosso objetivo

Para atingir o objetivo proposto, primeiramente se faz necessária a execução de experimentos para identificar *features* com maior impacto no desempenho de aplicações para processamento de *stream*. Finalizada esta etapa, será criado um *dataset* para treinamento dos algoritmos de aprendizado de máquina a partir de dados extraídos de aplicações de *stream* executadas. O *dataset* será composto por aplicações de diferentes domínios, o que irá auxiliar na generalização dos modelos. Inicialmente, a generalização focará apenas nas aplicações e não no sistema *multicore*.

Os algoritmos serão selecionados com base nas *features* disponíveis no *dataset*, já que elas podem ser categóricas, numéricas, entre outras. O modelo de aprendizado de máquina com melhor acurácia e menor custo de inferência será usado para determinar o melhor tamanho de *batch*.

Referências

Andrade, H. C. M., Gedik, B., and Turaga, D. S. (2014). *Fundamentals of Stream Processing: Application Design, System and Analytics*. Cambridge University Press, Cambridge CB2 8BS, United Kingdom.

Das, T., Zhong, Y., Stoica, I., and Shenker, S. (2014). Adaptive stream processing using dynamic batch sizing. In Lazowska, E., Terry, D., Arpaci-Dusseau, R. H., and Gehrke, J., editors, *Proceedings of the ACM Symposium on Cloud Computing, Seattle, WA, USA, November 3-5, 2014*, pages 16:1–16:13. ACM.

Stein, C. M., Rockenbach, D. A., Griebler, D., Torquati, M., Mencagli, G., Danelutto, M., and Fernandes, L. G. (2020). Latency-aware adaptive micro-batching techniques for streamed data compression on graphics processing units. *Concurrency and Computation: Practice and Experience*, na(na):e5786.

Zhang, Q., Song, Y., Routray, R., and Shi, W. (2016). Adaptive block and batch sizing for batched stream processing system. In Kounev, S., Giese, H., and Liu, J., editors, *2016 IEEE International Conference on Autonomic Computing, ICAC 2016, Wuerzburg, Germany, July 17-22, 2016*, pages 35–44. IEEE Computer Society.