Proposta de reparticionamento contínuo em sistemas com estado particionado

Douglas Pereira Luiz[1], **Odorico Machado Mendizabal**[1]

[1]Departamento de Informática e Estatística
Universidade Federal de Santa Catarina (UFSC) – Florianópolis – SC – Brazil

douglas.pereira@grad.ufsc.br, odorico.mendizabal@ufsc.br

Resumo. *Estratégias de particionamento de estado combinadas com algoritmos de corte em grafos podem ser utilizados para balancear a carga em sistemas de alta vazão. Neste trabalho, propomos uma estratégia para a realização de reparticionamentos que evita a contenção do restante do sistema. A estratégia consiste na realização do particionamento de forma assíncrona e contínua, considerando a carga de trabalho mais recente.*

1. Introdução

Particionamento de estados é uma estratégia comum para aumentar vazão. Entretanto, prever um esquema de partições eficiente pode ser difícil, e estratégias que fixam o nível de paralelismo na inicialização do sistema podem ser pouco adequadas para cargas de trabalho dinâmicas [Alchieri et al. 2017]. Para maximizar os ganhos em função do paralelismo, o particionamento pode ser reconfigurado durante a execução, o que pode balancear a carga de trabalho entre *threads* e reduzir o impacto causado por sincronizações devido a comandos conflitantes [Trombeta and Mendizabal 2020].

Resultados experimentais indicam que rebalanceamentos baseados em corte em grafos podem oferecer ganhos de desempenho [Trombeta 2021, Goulart et al. 2023]. No entanto, estratégias que param a execução, ou que só são realizadas em períodos de ociosidade, podem ser pouco adequadas para sistemas de alta vazão nos quais paradas são indesejadas. Em vista disso, nosso objetivo é reduzir o custo do particionamento de forma que aplicações possam se beneficiar de reparticionamentos frequentes ou mesmo ininterruptos.

2. Estratégia Proposta

Consideramos sistemas com estado particionado, com um escalonador que toma decisões com base em um mapa que associa partes do sistema à *threads* trabalhadoras. A caracterização da carga de trabalho é feita com um grafo, cujo particionamento fornece um mapa de partições, tal como apresentado em [Trombeta and Mendizabal 2020]. Queremos que reparticionamentos sejam realizados continuamente e de forma assíncrona ao escalonador, sem provocar interrupções prolongadas da execução dos outros componentes do sistema.

Propomos duas novas linhas de execução, o *caracterizador de carga* e o *particionador*. O caracterizador deve receber do escalonador informações dos acessos às variáveis do sistema e atualizar o grafo que caracteriza a carga de trabalho. Enquanto isso, o particionador é responsável por particionar o grafo e disponibilizar ao escalonador uma novo

mapa de partições. Com isso, ao fim da execução do particionamento, o escalonador tem a sua disposição um mapa atualizado. O particionador pode então dar início à produção de um novo mapa que considera as atualizações feitas pelo caracterizador durante o particionamento anterior.

O caracterizador deve manter informações sobre uma porção recente da carga de trabalho. Para isso, pode ser por quanto tempo uma informação é mantida no grafo, ou seja, a atualização de um vértice ou aresta do grafo em um dado momento deve ser desconsiderada futuramente. A limitação do tempo em que uma informação é mantida pelo caracterizador limita o tamanho do grafo e torna previsível as durações particionamento. Espera-se que a estratégia reduza o tempo de particionamento mantendo uma boa qualidade do balanceamento mesmo em cargas de trabalho que sofrem variações frequentes ou nas quais a carga recente resume bem o padrão de acessos.

3. Próximos Passos

Como caracterização de carga e reparticionamentos são realizados em linhas de execução diferentes, os acessos ao grafo devem ser sincronizados. Uma forma de reduzir a contenção no caracterizador de carga é a realização do particionamento com uma *cópia do grafo* produzida pela caracterizador. Outra alternativa seria a investigação de algoritmos não bloqueantes para a realização do corte do grafo, que não impusessem bloqueios à atualização de vértices e arestas.

Outro aspecto da proposta que ainda deve ser explorado é o tempo em que uma informação sobre a carga permanece no grafo. Dentre as opções está a definição de um valor na inicialização do sistema que determina o tempo de permanência de uma informação ou o tamanho máximo do grafo. Esse valor também poderia ser adaptado em tempo de execução, de forma que se busque uma boa compensação entre tempo de particionamento e qualidade do balanceamento.

A técnica pode ser aplicada em sistemas com replicação máquina de estados e contribuir para aumentar o desempenho de serviços como os de bancos de dados, gerenciadores de *clusters* e *key-value stores*. Espera-se que implementações baseadas na estratégia proposta mantenham constantemente o balanceamento de carga entre as *threads* trabalhadoras, sem recorrer a paradas prolongadas do escalonador e sem sofrer perdas significativas de desempenho.

Referências

Alchieri, E., Dotti, F., Mendizabal, O. M., and Pedone, F. (2017). Reconfiguring parallel state machine replication. In *SRDS*.

Goulart, H., Trombeta, J., Franco, A., and Mendizabal, O. (2023). Achieving enhanced performance combining checkpointing and dynamic state partitioning. In *SBAC-PAD*.

Trombeta, J. G. (2021). *Análise do uso de particionamento balanceado de grafos para explorar paralelismo em Replicação Máquina de Estados Paralela*. Monografia, Ciências da Computação, Universidade Federal de Santa Catarina, Florianópolis, SC, Brasil.

Trombeta, J. G. and Mendizabal, O. M. (2020). Proposta para reparticionamento de estado em replicação máquina de estado paralela. In *COTB '20*.