Proposta de TM para Arquiteturas Distribuídas

Jerônimo da Cunha Ramos, André Rauber Du Bois, Maurício Lima Pilla

Programa de Pós-Graduação em Computação – Universidade Federal de Pelotas

{jdcramos, dubois, pilla}@inf.ufpel.edu.br

Resumo. *Memórias transacionais (TM) facilitam a programação concorrente, mas poucos trabalhos suportam Sistemas Distribuídos (SD). Estes apresentam maiores desafios de programação do que as arquiteturas centralizadas. Desta forma, este trabalho visa desenvolver uma arquitetura para transações com objetos distribuídos, levando a facilidade de programação das TMs para os SDs.*

1. Introdução

TM é uma abstração para programação concorrente baseada na ideia de transações, similares às de banco de dados. Uma transação de memória é uma sequência atômica de operações que modificam a memória e pode ser executada completamente ou abortada. Elas executam como se estivessem modificando uma área de memória isolada do acesso de outras transações, que só conseguem ver o resultado após o *commit*. Este modelo possui várias vantagens sobre o modelo clássico de programação, sendo a principal a facilidade de programação [Rigo et al. 2007][Du Bois 2008]. Ele tem sido largamente estudado para aplicação em máquinas multicore, mas também pode ser utilizado, com algumas adaptações, para facilitar a programação de arquiteturas distribuídas. O estudo destas arquiteturas continua de extrema importância, tanto por comporem uma grande parcela da computação de alto desempenho, quanto por cada vez mais dispositivos computacionais se comunicarem entre si. Além disso, com o aumento do uso de *clouds* computacionais, se torna importante o estudo de como sincronizar as ações de donos diferentes, utilizando as primitivas de TM. Sendo assim, a principal motivação deste trabalho é buscar meios para utilização de mecanismos de TM em sistemas distribuídos com características diversas.

2. Trabalhos Relacionados

Existem várias implementações de TM para multicore. Um levantamento sobre o tema encontra-se em [Harris et al. 2010]. Porém, sistemas de TM com suporte à SD não são numerosos, e.g., Cluster-STM [Bocchino et al. 2008]: propõe uma interface com as operações necessárias e compara diferentes estratégias, todas rodando sobre PGAS (*Partitioned Global Address Space*); DiSTM[Kotselidis et al. 2008]: propõe algoritmos de coerência cache; TFA[Saad and Ravindran 2012]: propõe um algoritmo *dataflow*, movendo os objetos através dos nodos, não utiliza nenhum tipo de DSM (*Distributed Shared Memory*) e é totalmente distribuído; e DCMTJava [Ramos et al. 2016]: prototipou uma linguagem de domínio especifico embutida em Java, provendo suporte a transações locais e distribuídas, baseando-se no TFA para transações distribuídas.

3. Proposta

Este trabalho tem como objetivo central desenvolver uma arquitetura para transações que envolvam objetos distribuídos. Será dividido em dois níveis: (i) **nível de linguagem**, onde serão investigadas como as primitivas de alto nível para expressar transações de

memória de maneira composicional (e.g. ”*retry*”e ”*or else*” [Harris et al. 2010]) se adaptariam em um contexto de transações distribuídas, além de primitivas adicionais que se mostrem necessárias; e (ii) **nível de execução**, onde será estudado como devem ser adaptados os algoritmo para transações concorrentes, de forma a agregar o suporte a um contexto distribuído. Além disso, devem ser avaliadas as soluções existentes para ambientes distribuídos, identificando formas de adaptá-las para solucionar problemas em nível de linguagem de programação. Também devem ser avaliadas técnicas para minimizar o impacto da latência de rede na comunicação.

4. Resultados Parciais

Até o momento foram realizadas grande parte da revisão bibliográfica e parte da modelagem do sistema transacional distribuído. Já foram definidas as principais interfaces de nível intermediário, que servirão para ”interligar”o nível de linguagem ao nível de execução.A interface `Transaction` contém os métodos responsáveis por inicializar os metadados de uma transação, voltar ao estado anterior e fazer as validações necessárias para efetivar uma transação. `TObject` pode ser clonado, para migração entre nodos e contém os manipuladores de cada atributo de um objeto transacional; estes métodos podem ser gerados sinteticamente pelo compilador e devem conter todas as ações necessárias para leitura e escrita dos atributos. `Directory` é responsável pelo registro e localização dos objetos transacionais distribuídos através dos nodos. Já `VClock` contém os métodos do relógio virtual global, que é utilizado para o versionamento dos dados. Uma versão do sistema transacional já está sendo implementada, utilizando o algoritmo TFA e, assim que pronta e testada, será submetida a *benchmarks* clássicos para transações e comparada à trabalhos relacionados.

5. Considerações Finais

Espera-se obter uma solução para o problema de transações utilizando objetos distribuídos, dando suporte tanto em nível de linguagem, com abstrações de alto nível para a especificação de transações, quanto em nível de sistema de tempo de execução, provendo execução eficiente dessas abstrações.

Referências

Bocchino, R. L., Adve, V. S., and Chamberlain, B. L. (2008). Software transactional memory for large scale clusters. In *Proceedings of the 13th ACM PPoPP*.

Du Bois, A. R. (2008). Memórias transacionais e troca de mensagens: Duas alternativas para a programação de máquinas multi-core. *SBC, P. A. (Ed.). ERAD-RS*, pages 43–76.

Harris, T., Larus, J., and Rajwar, R. (2010). *Transactional Memory, 2nd Edition*. Morgan and Claypool Publishers, 2nd edition.

Kotselidis, C., Ansari, M., Jarvis, K., Luján, M., Kirkham, C., and Watson, I. (2008). Distm: A software transactional memory framework for clusters. In *IEEE ICPP*.

Ramos, J. d. C., Du Bois, A. R., and Pilla, M. L. (2016). An embedded domain specific language for distributed memory transactions in java. In *ACM SAC*.

Rigo, S., Centoducatte, P., and Baldassin, A. (2007). Memórias transacionais: Uma nova alternativa para programação concorrente. In *Minicursos do VIII WSCAD*.

Saad, M. and Ravindran, B. (2012). Transactional forwarding: Supporting highly-concurrent stm in asynchronous distributed systems. In *SBAC-PAD*.