Concorrência e Paralelismo 2016-17 - HW 4 - Transactional Memory

Aluno: Gonçalo Sousa Mendes, n.º 42082, Turno p.3
Prof: João Lourenço
Dept. de Informática
FCT/NOVA

24 de Dezembro de 2016

1 Introdução

Neste trabalho era necessário paralelizar o programa dado pelo projeto desta cadeira usando a técnica de transações, ou memória transacional.

2 Implementação

A implementação desta solução foi bastante simples, sendo apenas necessário identificar os locais onde os acessos teriam de ser protegidos, isto é, onde é que tinha de haver exclusão mútua e colocar a anotação *@atomic* antes dos métodos. Este métodos foram o da inserção, remoção e leitura de artigos.
Foram testados todos os algoritmos para conseguir se conseguir descobrir qual o melhor para este problema, sendo o *tl2inplace* o que apresentou mais operações por segundo. Faz-se ainda o reparo que todos os algoritmos passam os testes de validação.

3 Conclusão

A primeira coisa que se consegue perceber com esta solução é a simplicidade da implementação, mesmo quando esta fornece garantias fortes das propriedades ACID, o que com uma implementação baseada em *locks* pode ser difícil de se conseguir. No entanto, esta solução apresenta uma grande desvantagem quando comparada com a solução encontrada no projeto, usando *locks* de granularidade média, o número de operações por segundo diminui bastante.
Esta é uma das principais preocupações quando se quer encontrar uma maneira de tornar um programa sequencial num programa concorrente, entender bem quais as propriedades que queremos preservar, quais as invariantes do programa, avaliar as soluções existentes e fazer o balanço entre a simplicidade da solução e a sua eficiência. Só assim somos capazes de tomar uma decisão informada e garantir, com um grau de confiança elevado, que nos encontramos perante uma solução eficaz para o problema que nos foi apresentado.