Implementação de Tolerância a Falhas na Biblioteca de Programação Paralela - BPP

Cristiano Berni, Gabriel Pedebos, Iuri Fiedoruk, Rogério C. Turchetti, Ana Paula Canal

Curso de Ciência da Computação – Centro Universitário Franciscano (UNIFRA)
Rua dos Andradas, 1614 – Centro - Santa Maria - RS – Brasil
{cberni, gpedebos, iuri}@gmail.com, {turchetti, apc}@unifra.br

Introdução

Sistemas paralelos e distribuídos foram criados para distribuir as tarefas e aumentar o poder computacional através do uso de vários processadores como também prover o compartilhamento de recursos. Eles permitem que uma aplicação seja dividida em diferentes partes que podem ser processadas em sistemas independentes que se comunicam através de canais de comunicação [RIB 05]. Porém a implementação destes sistemas em ambientes sujeitos a falhas torna-se uma tarefa complexa frente a diversidade de falhas que podem ocorrer no sistema. Um sistema distribuído tolerante a falhas é projetado para fornecer um serviço contínuo, confiável e livre de defeitos de seus componentes [JAL 94].

Neste sentido o presente trabalho propõe a implementação de um detector de defeitos não confiável [CHA 96], como ferramenta de auxílio a tolerância a falhas, e acrescentando maior funcionalidade à Biblioteca de Programação Paralela - BPP. A BPP é uma biblioteca para programação paralela, como o MPI (*Message Passing Interface)* e a PVM (*Parallel Virtual Machine*), tendo como objetivo prover as primitivas básicas para a comunicação através de troca de mensagens, bem como para a sincronização dos processos, simplificando a implementação de programas paralelos [CIO 05], visto que as bibliotecas utilizadas para este fim são em geral, complexas.

O detector de defeitos monitora os processos e informa a aplicação em execução caso um dos nós do sistema falhe. Esse serviço é essencial, pois evita com que o processamento de um grande volume de informações seja perdido em caso de falha de um dos nós ou no canal de comunicação que o interliga com os demais.

Configuração

O detector de defeitos apresentado neste trabalho monitora os processos através da utilização de *timeouts*, ou seja, caso o processo monitorado deixe de responder ao processo monitor em um limite tempo, o mesmo passa a ser considerado suspeito. A utilização de um *timeout* se faz necessária, pois desta forma limita o tempo de espera por uma resposta, caso contrário, se tornaria infinita violando a propriedade de terminação de algoritmos determinísticos.

Definir um *timeout* para detectores de defeitos em um ambiente assíncrono, é uma tarefa complexa. A dificuldade está em estimar com precisão o tempo de resposta de um dado processo. Para reduzir a complexidade de tentar encontrar um valor

adequado para o *timeout* o presente serviço faz uso de modelos matemáticos para a predição do *timeout* [NUN 03] de forma dinâmica. Estes modelos baseiam-se no tempo de resposta dos processos em rodadas anteriores para a previsão do próximo *timeout*. Entre os preditores utilizados neste trabalho estão: *Last*, *Mean*, *Winmean* e *LPF*. O preditor *Last* utiliza como parâmetro de inferência somente o último valor amostrado, enquanto que os preditores *Mean*, *Winmean* e *LPF* baseiam-se na média aritmética ou ponderada dos tempos de resposta dos processos em rodadas anteriores, ou sobre um conjunto dos mesmos [NUN 03].

Ambiente de testes

O ambiente de testes do modelo constitui-se de um *cluster* de computadores, para o qual a BPP foi inicialmente desenvolvida. O *cluster* Unifra é composto por três máquinas *Pentium* IV 2.4 *Ghz*, onde o nó principal possui 512 MB de RAM e os demais, 256 MB. O serviço detector de defeitos utiliza o módulo de rede (*sockets*) para realizar a comunicação com os processos que compõem o sistema. É importante salientar que o detector de defeitos informa o estado de um determinado processo, sendo que a decisão sobre o que fazer em caso de falhas fica a cargo da aplicação cliente, representada neste trabalho pela BPP.

Considerações Finais

O serviço detector de defeito encontra-se em fase de desenvolvimento. Para a implementação do detector está sendo utilizada a linguagem *C*, pois a biblioteca de programação paralela – BPP, havia sido anteriormente implementada nesta linguagem. Na fase atual de desenvolvimento o serviço detecta processos que deixam de responder até o tempo limite espera fixo. A partir deste momento o processo é considerado suspeito, caso o mesmo volte a responder em um tempo futuro, este retorna ao estado operacional. Para analisar o comportamento do serviço foram realizados vários testes, nos quais foram forçadas falhas nos processos monitorados, e nos canais de comunicação, obtendo-se sucesso na detecção de todas as falhas. Foi possível perceber também que os preditores *Winmean* e *LPF* apresentam uma maior taxa de acerto na previsão do próximo *timeout*.

Referências

[RIB 05] RIBEIRO, U. E. **Sistemas Distribuídos – Desenvolvendo Aplicações de Alta Performance no Linux.** Rio de Janeiro – RJ: Axcel Books. 2ª ed., 2005.

[JAL 94] JALOTE, P. **Fault Tolerance in Distributed Systems**. Prentice Hall, 1994.

[CHA 96] CHANDRA, T. D.; TOUEG, S. **Unreliable failure detectors for reliable distributed systems**. Journal of the ACM, [S.l.], v.43, n.2, p.225.267, jan 1996.

[CIO 06] CIOCARI, J. F. C. **Desenvolvimento de uma Biblioteca para Programação Paralela – BPP**. ERAD - Escola Regional de Alto Desempenho 2006.

[NUN 03] NUNES, R. C. **Adaptação Dinâmica do Timeout de Detectores de Defeitos Através do Uso de séries Temporais**. 2003. Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre.