Alta Disponibilidade aplicada a Computação Móvel

Hélio Antônio Miranda da Silva , Ingrid Jansch-Pôrto

[1]Instituto de Informática – Programa de Pós-Graduação em Ciência da Computação
Universidade Federal do Rio Grande do Sul – UFRGS
Caixa Postal 15064 – 91501-970 Porto Alegre, RS

hamsilva@inf.ufrgs.br, ingrid@inf.ufrgs.br

Abstract. *This paper argues on the attainment of fault tolerance in distributed mobile environments. Mobile computation in distributed environments requires the use of techniques that add dependability to these applications. However, achieving fault tolerance in these environments is still a challenge, due to the characteristics of the network connections, as the their mobility, the low bandwidths and the high fault rates. In this paper, mechanisms for recovery using checkpoints are considered to achieve fault tolerance.*

Resumo. *Este artigo discute a obtenção de tolerância a falhas em ambientes móveis distribuídos. A computação móvel em ambientes distribuídos requer o uso de técnicas que adicionem disponibilidade à execução de aplicações. Contudo, a obtenção de tolerância a falhas nesses ambientes ainda é um desafio, devido às características de conectividade dessas redes, como a mobilidade, as baixas larguras de banda e as altas taxas de falhas. Neste artigo, é proposta a utilização de recuperação por checkpoints para obter tolerância a falhas.*

1. Introdução

A computação móvel oferece uma nova perspectiva no desenvolvimento de aplicações distribuídas, onde a dinâmica da mobilidade permite que os *hosts* [1] móveis se comuniquem de qualquer lugar a qualquer instante, mesmo estando em movimento. E assim como em redes estáticas, se faz necessário a utilização de técnicas que tornem a execução de aplicação distribuídas mais confiável e segura. Contudo, a obtenção de tolerância a falhas em ambientes móveis ainda é um desafio, devido às características das redes *wireless*, como a dinâmica da mobilidade, as baixas larguras de banda e as altas taxas de falhas. Assim, esquemas tradicionais de tolerância a falhas [Jalote, 1994] não podem ser diretamente aplicados nesses ambientes [Laamanen et al., 1999].

Neste artigo, são focados esquemas que utilizam recuperação através *checkpoints*. Este esquemas baseiam-se na realização, de tempos em tempos, de *checkpoints* globais que são constituídos por um conjunto de *checkpoints* locais. Assim, após uma falha, o sistema pode retornar até um estado consistente e retomar a execução.

O artigo está disposto da seguinte maneira: na seção 2, é definido modelo de sistema móvel que será utilizado. Na seção 3, serão vistas algumas características relacionadas à computação móvel. Na seção 4, é definida a recuperação de erros através de

[1]Host: em português significa ”hospedeiro”.

checkpoints. Nesta seção, também são apresentadas várias estratégias de realização de *checkpoints* em ambientes móveis. E por fim, a seção 5 mostra as conclusões do artigo.

2. Modelo de Sistema Móvel

Um modelo de sistema móvel é formado por *hosts* móveis (HM) [2] e também por *hosts* estáticos [Acharya and Badrinath, 1994]. Os *hosts* móveis podem se mover enquanto mantêm ativa a sua conexão. Os *hosts* estáticos são conectados uns aos outros por redes estáticas. Alguns dos *hosts* estáticos possuem uma interface *wireless* e cada um destes é considerado uma *estação de suporte à mobilidade* (ESM) [3], e têm como objetivo fazer a ligação entre a rede *wireless* e a rede estática. Considera-se que todas as ESMs são estáticas e estão interconectadas por redes estáticas. Por causa do limitado alcance das antenas *wireless*, um HM pode se comunicar com uma ESM somente dentro de uma determinada região, e a esse raio de cobertura da ESM é dado o nome de *célula*. Um HM, em dado instante, pode estar dentro de apenas uma célula. A mobilidade permite aos HMs passar de célula em célula de forma transparente. Quando ocorre passagem de um *host* para uma outra célula, a ESM responsável por este *host* também muda. Assim a ESM da nova célula passa a ser a mediadora entre o HM e a rede estática. Este processo de troca de ESM é chamado *handoff* [4].

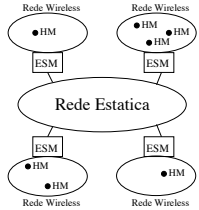

Figura 1: Modelo de Sistema Móvel

3. Aspectos Relacionados à Computação Móvel

Devido às características das redes *wireless*, técnicas tradicionais de obtenção de tolerância a falhas não podem ser diretamente aplicadas. Um dos motivos disso é que muitas destas técnicas foram desenvolvidas tendo-se em mente redes estáticas. Portanto, estes esquemas não prevêem a mobilidade. Características dos ambientes móveis, como a baixa largura de banda, quando comparados com redes estáticas, exigem a minimização da troca de mensagens entre *hosts*, pois altas taxas de comunicação podem causar um grande custo à aplicação distribuída. A probabilidade de avarias em equipamentos aumenta já que os *hosts* são carregados pelos usuários de um lugar para outro. Além disso, falhas transientes como falta de alimentação de energia e desconexões passam a ser freqüentes [Neves and Fuchs, 1997].

[2] No decorrer do artigo, freqüentemente, referências a hosts móveis serão feitas através da sigla HM.

[3] No decorrer do artigo, freqüentemente, referências as estações de suporte a mobilidade serão feitas através da sigla ESM.

[4] Handoff: em português significa "ato de delegar".

Quanto à utilização de métodos de recuperação por *checkpoints*, surge um novo problema, pois os *checkpoints* não podem ser salvos nos próprios *hosts* como em redes estáticas, já que *hosts* móveis, além de nem sempre terem à disposição recursos como espaço em disco, possuem uma maior probabilidade de falhas [Acharya and Badrinath, 1994]. Por isso, os *checkpoints* devem ser salvos nas ESMs. Contudo, é importante lembrar que se torna necessário adotar critérios na escolha das ESMs onde os *checkpoints* serão armazenados, pois os *hosts*, devido a sua mobilidade, não possuem uma ESM fixa.

4. Estratégias de Recuperação

Na recuperação baseada em *checkpoints*, cada *checkpoint* consiste em um estado da execução da aplicação. Periodicamente cada *host* participante da execução da aplicação distribuída realiza um *ckeckpoint*, formando dessa forma, um *checkpoint* global que contém o estado da execução da aplicação como um todo. Na ocorrência de falhas, o último *checkpoint* consistente é carregado e a aplicação pode recomeçar sua execução a partir desse ponto.

A realização dos *checkpoints* pode ser síncrona ou assíncrona [Acharya and Badrinath, 1994]. Em *checkpoints* síncronos, ocorre coordenação entre todos os participantes para garantir a consistência. Porém, esse esquema apresenta uma sobrecarga significativa gerada pela sincronização entre os processos. Em esquemas assíncronos, cada participante realiza seus *checkpoints* de forma independente dos outros participantes. Nesse, a desvantagem é que um dado conjunto de *checkpoints* locais não necessariamente forma um *checkpoint* global consistente.

Uma estratégia de recuperação por *ckeckpoints* possui duas componentes: uma estratégia para o salvamento de estado, que determina quando serão realizados os *checkpoints*, e uma estratégia que define onde ficarão armazenados os *checkpoints* quando o *host* trocar de ESM. As estratégias de recuperação por *checkpoint* em *hosts* móveis, definidos por Pradhan et al.(1996), e abordadas a seguir, enquadram-se no conceito de *checkpoints* assíncronos.

4.1. Salvamento de Estado

Conforme visto anteriormente, os *checkpoints* devem ser armazenados em um local seguro. Portanto, o armazenamento nos próprios HM é inadequado, pois, além de estarem mais propensos a falhas do que *hosts* estáticos, os HM podem ter uma capacidade de armazenamento muito limitada. Assim, os *checkpoints* devem ser armazenados na atual ESM do *host*, já que esta, por ser estática, possivelmente terá um maior poder de armazenamento e processamento.

Os *checkpoints* locais são realizados de forma independente para cada HM participante da aplicação. Porém, é necessário determinar com que periodicidade eles devem ser realizados. A estratégia de salvamento de estados pode ocorrer de duas maneiras: sem *logs* ou com *logs*.

4.1.1. Sem registros de *logs*

Nessa estratégia, cada vez que um evento ocorre, seja o recebimento ou envio de uma mensagem ou uma entrada de dados do usuário, um novo *checkpoint* deve ser realizado. O *checkpoint* é armazenado na ESM corrente. Isto é, um novo salvamento de estado é realizado após ocorrência de qualquer evento que mude o estado do *host*. Após uma falha, quando a *host* móvel reinicia, ele pode buscar o último *checkpoint* realizado que está armazenado na sua atual ESM, diminuindo assim o tempo de recuperação. Porém, é importante lembrar que, a freqüente transmissão de estados através da rede é a desvantagem deste esquema.

4.1.2. Com registros de *logs*

Nesta estratégia, os *checkpoints* para cada HM são realizados em intervalos regulares de tempo. Os eventos, causadores de mudança de estado, que ocorrerem entre *checkpoints*, serão registrados através de *logs*. Quando um evento de mudança de estado ocorre, antes do processamento deste, é feito o registro de *log* que é enviado para a ESM corrente. Depois de armazenar o *log*, a ESM envia uma mensagem de confirmação para o HM. Os registros de *log* permanecem na ESM até que o próximo *checkpoint* seja realizado. Após uma falha, quando o *host* reinicia, ele pode retomar sua execução a partir do mais recente *ckeckpoint* e dos registros de *log* armazenados na ESM.

A vantagem desta abordagem é que transferir apenas um registro de *log* é menos oneroso do que transferir um novo *checkpoint*. Com isso, ocorre a diminuição da sobrecarga causada pela troca de informações entre o HM e a ESM. Porém, a desvantagem é que, no caso de falhas, a recuperação não é feita através da simples reativação do estado definido no *checkpoint* como na abordagem sem *logs*, já que também os registros de *log* definem ações a serem novamente realizadas.

4.2. *Handoff*

Em ambientes móveis, conforme pode ser observado na figura 2, um *host* pode movimentar-se para uma nova célula e passar a ter uma nova ESM. Assim, torna-se necessário adotar uma nova abordagem para realizar a recuperação de *checkpoint* e/ou registros de *logs* que potencialmente podem estar espalhados através de várias ESMs.

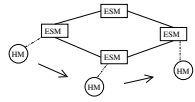

Figura 2: Ocorrência de *Handoffs*

Este problema pode ser resolvido através da troca de informações referentes ao estado da aplicação entre a antiga ESM e a nova ESM do HM. A seguir, são definidas três estratégias da troca de informações durante o processo de *handoff*: a pessimista, a preguiçosa [5] e a astuciosa [6] [Pradhan et al., 1996].

[5] Tradução utilizada para a palavra *lazy* da língua inglesa.
[6] Tradução utilizada para a palavra *trickle* da língua inglesa.

4.2.1. Estratégia Pessimista

Nesta estratégia, quando um *host* se move para uma nova célula, o seu *checkpoint* é transferido para a nova ESM. No caso de estarem sendo utilizados *logs*, estes também são transferidos. A velha ESM do *host*, após receber a confirmação de recebimento do *checkpoint* da nova ESM, pode excluir a sua cópia do *checkpoint*. A desvantagem desta abordagem é o grande volume de informações que é trocado durante o processo de *handoff*.

4.2.2. Estratégia Preguiçosa

Na estratégia preguiçosa, os *checkpoints* não são transferidos durante o *handoff*; em vez disso, são criadas listas que determinam quais ESMs o *host* visitou, e a partir destas listas o *checkpoint* e os *logs* podem ser recuperados. Após o *handoff*, a nova ESM recebe uma mensagem que contém a localização da ESM anteriormente visitada pelo *host*. Já que o *host* tem liberdade para mover-se através de várias células, como resultado disso surge efetivamente uma lista de ligação entre as várias ESMs visitadas. Porém, quando um novo *checkpoint* é realizado, a atual ESM informa este evento para a antiga ESM, para que esta possa apagar a sua cópia do *checkpoint* e mais todas informações referentes àquele *host*. Este processo ocorre até que todas as ESMs da lista de ligação excluam as informações sobre aquele HM.

Esta estratégia diminui a sobrecarga decorrente da troca de informações entre ESMs durante o processo de *handoff*, se comparada com a estratégia pessimista. Contudo, o tempo de recuperação aumenta, já que possivelmente será necessário buscar informações espalhadas em várias ESMs.

4.2.3. Estratégia Astuciosa

Na estratégia preguiçosa, devido a uma grande mobilidade do *host*, o *checkpoint* e os *logs* podem ficar espalhados por várias ESMs o que aumentaria consideravelmente o tempo de recuperação. Portanto, visando atenuar este problema, e para manter os baixos custos de *handoff* da estratégia preguiçosa, é proposta a estratégia astuciosa. Esta estratégia garante que o *checkpoint* estará perto da atual ESM. Considerando que a distância entre a atual ESM e o anterior é de um ”salto”, esta estratégia garante que o *checkpoint* estará a no máximo um salto de distância da ESM atual.

Para conseguir isso, a nova ESM passa uma mensagem de controle para a ESM anterior do HM. Com recebimento desta mensagem, a ESM anterior trará para junto de si o último *checkpoint* e os possíveis *logs*. Assim, no caso de falha, para realizar a recuperação, a atual ESM pede o *checkpoint* e os *logs* que estão armazenados na ESM anterior. Quando um novo *checkpoint* for realizado e conseqüentemente for armazenado na ESM atual, a ESM atual enviará uma notificação autorizando a ESM anterior a excluir as informações referentes aquele HM.

4.3. Comparação entre estratégias

Segundo Pradhan et al.(1996), em esquemas tradicionais de recuperação, a taxa de falhas é o fator determinante na escolha do esquema de recuperação adequado. Já que em sistemas

wireless, não somente a taxa de falhas, mas também a taxa de mobilidade dos *hosts* e a largura de banda disponível devem ser levadas em consideração nessa escolha.

A tabela 1 mostra uma vinculação entre as estratégias e ambientes diversos, associados por Pradhan et al.(1996), de acordo com sua adaptabilidade. Segundo os próprios proponentes, é possível observar que não existe um esquema eficiente para todos os ambientes, embora a predominância seja de esquemas de salvamento que adotam *logs*.

Mobilidade	Largura de Banda	Taxa de Falhas	Esquema
Alta	Baixa	Baixa	Com *log* e preguiçosa
		Alta	Sem *log* e astuciosa
	Alta	Todas	Com *log* e astuciosa
Baixa	Todas	Todas	Com *log* e preguiçosa

Tabela 1: Esquemas de recuperação ótimos

5. Conclusão

Ambientes móveis apresentam-se como um desafio à área de tolerância a falhas graças às suas características inerentes como a mobilidade e baixa largura de banda. Uma maneira de obter-se a disponibilidade e a confiabilidade necessárias nesses sistemas é através da utilização de recuperação por *checkpoints*. Portanto, este artigo mostrou diferentes modos de realizar *checkpoints* em ambientes móveis. Entre os objetivos visados através das estratégias de recuperação propostas destacam-se a minimização da troca de informação entre *hosts* móveis e a adoção de uma estratégia de *handoff* que não cause sobrecarga devido à troca de mensagens e nem altos custos para a recuperação.

Referências

Acharya, A. and Badrinath, B. (1994). Checkpointing distributed applications on mobile computers. *Procedings of the Third International Conference on Parallel and Distributed Information Systems*.

Jalote, P. (1994). *Fault Tolerance in Distributed Systems*. Prentice Hall, Englewood Cliffs, New Jersey 07632.

Laamanen, H., Alanko, T., and Raatikainen, K. (1999). Dependability issues in mobile distributed system. *Pacific Rim International Symposium on Dependable Computing*.

Neves, N. and Fuchs, W. K. (1997). Adaptive recovery for mobile environments. *Communications of the ACM*, 40(1):68–74.

Pradhan, D. K., Krishna, P., and Vaidya, N. H. (1996). Recoverable mobile environment: Design and trade-off analysis. In *Symposium on Fault-Tolerant Computing*, pages 16–25, Texas - USA. IEEE.