Escalonamento para Sistemas de Tempo Real Tolerantes a Falhas: Um Estudo Empírico

Edinaldo O. de Jesus , George M. de A. Lima

Laboratório de Sistemas Distribuídos (LaSiD)
Departamento de Ciência da Computação, Universidade Federal da Bahia (UFBA)
Salvador - Bahia - Brasil

edinaldo.jesus@uol.com.br, gmlima@ufba.br

Abstract. *Some scheduling approaches for fault-tolerant hard real-time systems are evaluated by simulation. The obtained results illustrate some advantages of using dynamic scheduling criteria in static schedulers for fault tolerance purposes. Up to now such a kind of evaluation has not been addressed.*

Resumo. *Algumas abordagens de escalonamento para sistemas de tempo real críticos tolerantes a falhas são avaliadas por simulação. Os resultados obtidos ilustram vantagens relacionadas ao uso de critérios dinâmicos em escalonadores estáticos para propósitos de tolerância a falhas. Até então tal tipo de avaliação não havia sido realizada.*

1. Introdução

Dois aspectos são de fundamental importância na construção de sistemas de tempo real críticos (*hard*): escalonamento de tarefas e tolerância a falhas. Ambos devem ser considerados de maneira integrada para que as tarefas do sistema sejam escalonadas respeitando suas restrições temporais (*deadlines*) e para que rotinas de detecção e recuperação de erros, quando executadas, não comprometam a correção do sistema. Para tanto, várias abordagens de escalonamento têm sido propostas (para uma descrição de algumas delas consultar [Lima 2003]). O principal objetivo destas abordagens é modelar e equacionar os efeitos causados pela execução de rotinas de recuperação durante a execução do sistema (*análise de escalonamento*). A principal dificuldade para alcançar este objetivo está relacionada à imprevisibilidade intrínseca à ocorrência de falhas. De fato, ao contrário das tarefas de um sistema de tempo real crítico, que possuem características e atributos conhecidos, é impossível prever quando e como alguma tarefa em execução irá falhar. Para contornar esta dificuldade, vários autores têm considerado modelos de falhas que podem ser, de certa forma, equacionados durante a análise de escalonamento. Por exemplo, alguns assumem que uma tarefa não pode falhar mais que k vezes [Liberato et al. 2000], outros consideram um mínimo intervalo de tempo entre dois erros [Burns et al. 1996, Lima and Burns 2003].

O objetivo do trabalho aqui apresentado é diferente. Não há preocupação com a análise de escalonamento. Assumimos um modelo de falhas probabilístico para que seja possível comparar, através de *experimentação*, algumas abordagens de escalonamento. As possibilidades avaliadas vão desde as abordagens estáticas, usando a política DM (*Deadline Monotonic*), até as abordagens dinâmicas, usando EDF (*Earlier Deadline First*). Ao todo, quatro abordagens de escalonamento foram avaliadas.

2. Abordagens de Escalonamento para Tolerância a Falhas

Assumimos que o sistema de tempo real é crítico, uniprocessado e composto por um conjunto conhecido de tarefas. Cada tarefa τ_i é descrita por seu período T_i, seu *deadline* D_i e seu tempo máximo de computação C_i. Em caso de falhas, a própria tarefa τ_i é responsável por detectar o erro e acionar rotinas responsáveis pelas ações de compensação ou recuperação do sistema, chamadas aqui de *tarefas alternativas*, representadas por $\overline{\tau}_i$. O maior custo de execução de $\overline{\tau}_i$ é $\overline{C}_i$. Se $\overline{\tau}_i$ for escalonada, ela deve terminar sua execução sem violar o *deadline* de τ_i. Se outros erros ocorrem em $\overline{\tau}_i$, assumimos que ela deve ser re-executada.

O modelo de escalonamento aqui considerado contém duas filas de tarefas, `FilaP` e `FilaA`. A primeira contém as tarefas principais, que são executadas quando não há erros. Se algum erro é detectado em alguma tarefa, seja ela principal ou alternativa, uma tarefa alternativa é inserida na `FilaA`. Cada uma destas filas possui seu próprios critérios de ordenação. O escalonador, examinando ambas as filas, decide, de acordo com alguma política de escolha, qual tarefa irá executar. Variações das políticas de ordenação das filas e das possibilidades de escolhas do escalonador serão descritas na próxima seção. A idéia deste modelo é possibilitar o ajuste do nível de dinamismo do escalonamento do sistema a fim de avaliar os efeitos causados sua capacidade de tolerância a falhas.

Quatro abordagens de escalonamento foram avaliadas. Primeiramente, consideramos que qualquer tarefa alternativa seria executada com a mesma prioridade da tarefa falha. Assim, tanto `FilaA` quanto `FilaP` são ordenadas de acordo com DM. O escalonador escolhe, de ambas as filas, a tarefa de maior prioridade para ser executada. Esta abordagem, já considerada por outros autores [Burns et al. 1996], foi chamada aqui de **FPA** (*fixed priority approach*).

A segunda abordagem considerada, **DSA** (*defferable server approach*), é definida da seguinte forma. `FilaP` continua sendo ordenada de acordo com DM. A `FilaA` é ordenada de acordo com EDF e suas tarefas são executadas por um servidor DS (*defferable server*) [Lehoczky et al. 1987], que possui a mais alta prioridade. Portanto, as tarefas da `FilaA` terão sempre prioridade maior que as da `FilaP` desde que haja capacidade do servidor disponível.

As outras duas políticas de escalonamento simuladas consideram critérios dinâmicos de escalonamento em diferentes níveis. Na abordagem **DPA** (*dynamic priority approach A*) as filas `FilaP` e `FilaA` são ordenadas de acordo com DM e EDF, respectivamente. O escalonador **DPA** avalia, segundo EDF, todas as tarefas da `FilaP` em relação à tarefa mais prioritária da `FilaA` caso haja tarefas em ambas as filas. Nesse caso, há uma promoção de prioridade das tarefas da `FilaP`, dependendo de seus *deadlines* absolutos. Desta forma, tarefas na `FilaA` são apenas executadas após a execução de todas as tarefas da `FilaP` que possuem menores *deadlines* absolutos. Por fim, a abordagem DPB (*dynamic priority B*) usa EDF em ambas as filas, sendo eqüivalente, portanto, a um escalonador EDF.

3. Simulação e Resultados Obtidos

As experimentações foram feitas a partir de simulações da execução de conjuntos de de tarefas, todas escalonáveis num cenário livre de falhas segundo a abordagem DM.

Foram considerados 30 conjuntos, cada um constituído de 10 tarefas. Os valores de C_i e $\overline{C}_i$ ($\overline{C}_i \leq C_i$) foram gerados de acordo com uma distribuição exponencial com média $U/10$, onde U é o percentual de utilização do conjunto de tarefas considerado. Períodos e *deadlines* foram gerados de acordo com uma distribuição uniforme no intervalo $[1, 500]$ sendo que $D_i \leq T_i$. Não foram considerados conjuntos de tarefas com $U > 0,8$, pois estes possuem pouca capacidade ociosa para executar rotinas de recuperação.

Durante a execução dos experimentos, erros foram gerados aleatoriamente de tal forma a estarem uniformemente distribuídos ao longo do período de simulação, que foi fixado em $3 \max(T_i)$, período considerado satisfatório para os propósitos deste trabalho. Desta forma, assumimos que não existe tendência para ocorrências de erros em funcão do tempo. A fim de romper os limites de escalonabilidade dos conjuntos de tarefas considerados, fixamos uma taxa média de geração de erros relativamene alta, $\lambda = 0,3$. Caso houvesse alguma tarefa em execução no instante em que um erro fosse gerado, ela seria considerada falha e a tarefa alternativa correspondente seria inserida na `FilaA`. Se o processador estivesse ocioso no instante da ocorrência de um erro, o mesmo seria desconsiderado. Cada abordagem de escalonamento foi avaliada considerando os mesmos conjuntos de tarefas e os mesmos instantes de geração de erros.

A comparação entre as abordagens de escalonamento foi realizada em termos de duas métricas: a taxa de recuperação (TR) e a taxa de interferência (TI). TR mede o percentual de tarefas que falharam e foram efetivamente recuperadas sem violação de seus *deadlines*. TI é definida como o quociente entre o número de *deadlines* não cumpridos por causa da execução de tarefas alternativas e o número total de *deadlines* violados durante a simulação. Se não há *deadlines* violados, $\mathrm{TI} = 0$.

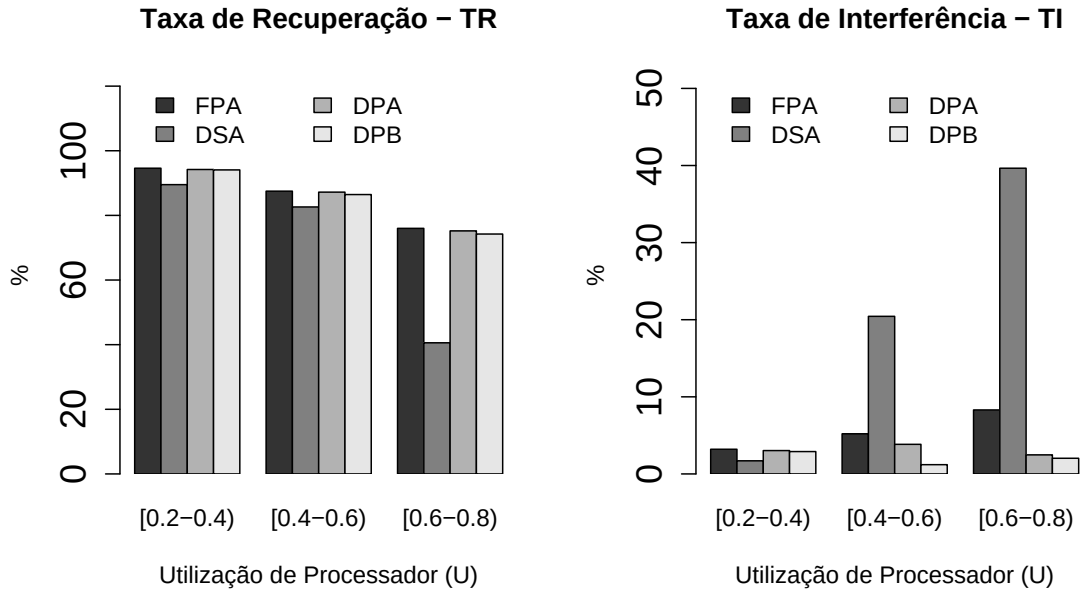

Figura 1. Comparação entre as abordagens.

O comportamento das abordagens está resumido nos gráficos da figura 1, que contém os valores médios de TR e TI para diferentes faixas de U. A abordagem DSA é a menos eficiente em termos de tolerância a falhas. Este fato já era esperado, pois há forte dependência da capacidade e do período do servidor. Tais parâmetros foram ajustados a fim de se obter o menor número de *deadlines* violados. De fato, a execução das tarefas sofre interferência do servidor executando na mais alta prioridade, o que aumenta o valor de TI. Para melhorar estes valores tanto a capacidade do servidor poderia ser reduzida quanto o seu período poderia ser aumentado. No entanto, isto faria com que os valores de TR diminuíssem significativamente.

Em termos de TR, o desempenho da abordagem FPA é comparável com as que usam algum critério dinâmico de escalonamento (DPA ou DPB). No entanto, pode-se perceber que há um aumento de TI para valores altos de U. Como a recuperação de tarefas pode sofrer preempção de tarefas mais prioritárias, esta recuperação pode se alongar muito. Isto acaba interferindo no tempo de resposta de tarefas menos prioritárias e estas podem ter *deadlines* absolutos menores.

Os desempenhos observados para as abordagens DPA e DPB são similares (para TR e TI). É importante notar que ambas as abordagens baseiam-se na variação de prioridades, DPA em cenários de falhas e DPB equivalente ao escalonador EDF. Portanto, DPB conserva o caráter estático da `FilaP` em cenários sem falhas; considera todas as tarefas do sistema ao escalonar procedimentos de recuperação; e promove tarefas na `FilaP`, caso seja necessário, a fim de reduzir TI e aumentar TR.

Concluindo, pode-se observar potenciais benefícios quando abordagens mistas de escalonamento, que incorporam políticas dinâmicas a escalonadores estáticos, são usadas. Tais abordagens, exemplificadas aqui por DPA, podem inserir um alto nível de adaptação no sistema em cenários de falhas. Derivar a escalonabilidade do sistema em situações de falhas usando algumas das abordagens aqui descritas deverá fazer parte de trabalhos futuros.

Referências

Burns, A., Davis, R. I., and Punnekkat, S. (1996). "Feasibility Analysis of Fault-Tolerant Real-Time Task Sets". In *Proc. of the Euromicro Real-Time Systems Workshop*, pp. 29–33.

Lehoczky, J. P., Sha, L., and Strosnider, J. (1987). "Enhanced Aperiodic Responsiveness in Hard Real-time Environment". In *Proc. of the 8th IEEE RTSS*, pp. 110–123.

Liberato, F., Melhem, R. G., and Mossé, D. (2000). "Tolerance to Multiple Transient Faults for Aperiodic Tasks in Hard Real-Time Systems". *IEEE Transactions on Computers*, 49(9):906–914.

Lima, G. M. A. (2003). *"Fault Tolerance in Fixed-Priority Hard Real-Time Distributed Systems"*. PhD thesis, Deptartment of Computer Science, University of York.

Lima, G. M. A. and Burns, A. (2003). "An Optimal Fixed-Priority Assignment Algorithm for Supporting Fault Tolerant Hard Real-Time Systems". *IEEE Transaction on Computers*, 52(10):1332–1346.