Programa de Pós-Graduação em Engenharia de Automação e Sistemas
Universidade Federal de Santa Catarina

Emprego do Linux Perf na Medição e Análise do Tempo de Resposta de Aplicações de Tempo Real

Projeto de Dissertação de Mestrado
Orientador: Rômulo Silva de Oliveira
Co-Orientador: Karila Palma Silva (IFC)

1. Contexto

Os sistemas submetidos a requisitos temporais, também chamados de sistemas de tempo real, possuem requisitos que variam muito com relação ao tamanho, complexidade e criticalidade. No contexto da automação industrial, são muitas as possibilidades (ou necessidades) de empregar sistemas com requisitos de tempo real. Exemplos são os sistemas de controle embutidos em equipamentos industriais, os sistemas de supervisão e controle de células de manufatura e os sistemas responsáveis pela supervisão e controle de plantas industriais completas.

Na literatura os sistemas de tempo real são, em geral, classificados conforme a criticalidade dos seus requisitos temporais. Enquanto o não atendimento dos requisitos temporais nos sistemas de tempo real não críticos (soft real-time systems) resulta apenas na redução da utilidade destes sistemas (ou qualidade do serviço prestado), o não atendimento de um requisito temporal em sistemas de tempo real críticos (hard real-time systems) pode ter como resultado a perda de vidas humanas.

Sistemas de tempo real são uma tecnologia fundamental para a instrumentação necessária no contexto da Indústria 4.0. O desenvolvimento de sistemas para condicionamento analógico de sinais visando a realização de medições, o desenvolvimento de sistemas embarcados integrando instrumentos e microcontroladores, e a integração desses com a Internet/nuvem, são atividades fundamentais. A evolução industrial pressupõe a disseminação de sistemas distribuídos que trocam informação entre si através de redes de comunicação digital de alto desempenho que são conhecidos na literatura como Sistemas de Controle em Rede (NCS – Networked Control Systems).

Nos sistemas em geral (que não são do tipo tempo real), a única preocupação é com a qualidade dos resultados. Embora uma execução rápida seja desejável, a abordagem é sempre do tipo "fazer o trabalho usando o tempo que for necessário". Sistemas tempo real possuem uma abordagem diferente, pois o tempo é limitado. É preciso garantir que será possível atender aos prazos, geralmente impostos pelo ambiente do sistema. Logo, a preocupação é “fazer o trabalho usando o tempo disponível”. Esta preocupação tem como consequência um problema básico encontrado na construção de sistemas tempo real (críticos ou não): a alocação e o escalonamento das tarefas nos recursos computacionais disponíveis. Existe uma dificuldade intrínseca em compatibilizar dois objetivos fundamentais: garantir que os resultados serão produzidos no momento desejado e dotar o sistema de flexibilidade para adaptar-se a um ambiente dinâmico e, assim, aumentar sua utilidade.

Qualquer abordagem que pretenda fornecer garantia para os *deadlines* precisa conhecer o comportamento do sistema no pior caso, tanto do *software* quanto do *hardware*. Isso porque os *deadlines* deverão ser cumpridos mesmo nos cenários mais desfavoráveis para a aplicação. Isso significa supor o pior caso, incluindo: o pior fluxo de controle para cada tarefa, o pior cenário de sincronização entre tarefas (exclusão mútua, etc.), os piores dados de entrada, a pior combinação de eventos externos (interrupções, sensores, etc.), o pior comportamento das *caches* internas e externas do *hardware*, o pior comportamento do processador (*pipeline*, barramentos, etc.), ou seja, um

cenário composto pela pior combinação possível de eventos. No cenário de pior caso, teremos o tempo de execução no pior caso (*WCET – Worst-Case Execution Time*) para cada tarefa individualmente, ou seja, apenas a sua própria computação, tarefa executando sem interrupções. Assim como o tempo de resposta no pior caso (*WCRT – Worst-Case Response Time*), esse último combinando a execução das várias tarefas, ou seja, sua própria computação, interferências, bloqueios, qualquer atraso que a tarefa possa sofrer.

A introdução de elementos cada vez mais complexos e cujo comportamento temporal não é determinístico nas arquiteturas computacionais modernas, como memórias *cache*, *pipelines* superescalares e mecanismos de predição de fluxo (*branch prediction*), acaba por dificultar ou até mesmo impossibilitar a determinação do tempo máximo de execução de uma tarefa. O uso de sistemas operacionais complexos como o Linux impede métodos analíticos para a determinação do tempo de resposta no pior caso.

O método de medições consiste em executar uma tarefa de *software*, ou partes dela, para um conjunto de entradas, em um *hardware* específico ou em um simulador. Assim, múltiplos tempos de operação dos diferentes componentes e procedimentos definidos em cada tarefa são avaliados. O tempo de execução mínimo e máximo, a distribuição desses tempos, ou também uma combinação entre os tempos de operação de diferentes segmentos de código são calculados.

Utilizando medição é difícil determinar se os valores obtidos são seguros, ou seja, é difícil garantir que o maior tempo de execução ou resposta medido seja o verdadeiro pior caso. Isso porque o cenário de pior caso pode ocorrer com pouca frequência, e as condições para que ele aconteça são normalmente desconhecidas.

Existe um conjunto de métodos de análise que utilizam métodos estatísticos para fornecer estimativas de pior caso que serão excedidas apenas com uma probabilidade máxima que é previsível. Esses métodos são baseados no ramo da estatística chamado Teoria dos Valores Extremos (TVE).

Uma estimativa do tempo de resposta de uma tarefa no pior caso pode ser obtida a um determinado nível de confiança estatística, por meio do uso de medição para obter dados e da TVE para generalizar os processos que produzem essas medições. A previsão de um evento, ou a estimativa de um valor particular, pode ser feita usando provas obtidas por meio de medição. A TVE é um ramo da estatística que permite realizar inferências para comportamentos extremos e atípicos de fenômenos sujeitos a aleatoriedade.

O objetivo da TVE é analisar valores extremos observados, e prever possíveis valores ainda mais extremos, inferindo sobre eventos cujas probabilidades são menores do que a probabilidade de qualquer evento observado anteriormente.

A principal dificuldade é que há uma exigência de generalizar o comportamento além do maior valor visto no conjunto de dados. Embora existam vários trabalhos na literatura que empregam TVE na estimação do *WCET* em sistemas simples, seu uso para estimar o *WCRT* em sistemas complexos ainda é debatido.

1.1 Linux Perf

Perf é uma ferramenta profiler para sistemas baseados em Linux 2.6+ que abstrai as diferenças de hardware em medições de desempenho Linux e apresenta uma interface de linha de comando simples. Perf é baseado na interface perf_events exportado por versões recentes do kernel do Linux.

A ferramenta perf oferece um rico conjunto de comandos para coletar e analisar os dados de desempenho e de rastreamento. O uso de linha de comando é uma reminiscência do git em que há uma ferramenta genérica, perf, que implementa um conjunto de comandos.

A ferramenta perf suporta uma lista de eventos mensuráveis. A ferramenta e a interface do kernel subjacente podem medir eventos provenientes de diferentes fontes. Por exemplo, alguns eventos são contadores de kernel puros, neste caso chamados de eventos de software. Os exemplos incluem: Context-switches, pequenas faltas.

Outra fonte de eventos é o próprio processador e sua Unidade de Monitoramento de Desempenho (PMU - Performance Monitoring Unit) do processador. Ele fornece uma lista de eventos para medir eventos micro-arquitetônicos, tais como o número de ciclos, instruções, erros de cache L1 e assim por diante. Esses eventos são chamados de eventos de hardware PMU ou eventos de hardware. Eles variam de acordo com cada tipo de processador e modelo.

2. Objetivo

Neste trabalho será empregado o método de medições para estimar o *WCRT* de tarefas que executam em sistemas de tempo real complexos. Será usado como estudo de caso o sistema operacional LINUX. Medições serão feitas com o uso da ferramenta Perf.

O objetivo geral deste projeto de mestrado é a avaliação de métodos e técnicas para estimar o tempo de resposta no pior caso, em aplicações de tempo real. Busca-se métodos e técnicas estatísticas que sejam estruturados de tal forma a permitir lidar com arquiteturas de *hardware* e sistemas operacionais modernos e complexos, de forma a obter estimativas mais confiáveis.

Deseja-se que os métodos propostos possibilitem obter estimativas de confiabilidade para os requisitos temporais e, consequentemente, garantir a segurança, a previsibilidade e a robustez do sistema, equilibrando esforços entre confiabilidade e custo. Ou seja, reduzir o custo testando o mínimo possível e garantir a confiabilidade maximizando a confiança de que os requisitos temporais são cumpridos.

Também deseja-se explorar a utilização da ferramenta Perf em soluções para a verificação de aplicações de tempo real executando sobre o sistema operacional Linux. O objetivo é monitorar o comportamento do sistema durante a fase de testes para levantar dados estatísticos e depois, com a aplicação no campo, monitorar seu comportamento para detectar desvios, os quais podem indicar violações de segurança, falha intermitente do hardware, ou ainda condições de execução (dados de entrada) diferentes daqueles previstos durante o desenvolvimento do sistema.

3. Atividades

(a) Estudar os conceitos básicos dos sistemas de tempo real.

(b) Estudar a determinação do *WCRT* através de medição.

(c) Estudar o conjunto de ferramentas associados com o perf.

(d) Implementar aplicações de tempo real com o propósito de usar o perf e consolidar o entendimento do que ele pode provê e como pode ser usado.

(e) Estudar o emprego de técnicas estatísticas e da Teoria dos Valores Extremos (TVE) para estimar o tempo de resposta das tarefas no pior caso.

(f) Montar um cenário de testes onde uma aplicação alvo será analisada, a fim de validar os métodos estatísticos.

(g) Redigir artigos para publicação em congressos e/ou revistas.

(h) Redigir a dissertação.

4. Bibliografia

[1] R. I. Davis and A. Burns, A survey of hard real-time scheduling algorithms and schedulability analysis techniques for multiprocessor systems, Systems Research, 2009.

[2] G. Buttazzo, Hard Real-Time Computing Systems: Predictable Scheduling Algorithms and Applications. Springer, 2 ed., 2005.

[3] J. W. S. Liu, Real-Time Systems. Prentice Hall, 1 ed., 2000.

[4] D. B. de Oliveira, R. S. de Oliveira, Timing Analysis of the PREEMPT RT Linux Kernel, Software Practice and Experience, Volume 46, Issue 6, pages 789–819, June 2016.

[5] N. C. Audsley, A. Burns, R. I. Davis, K. W. Tindell, A. J. Wellings, Fixed priority pre-emptive scheduling: An historical perspective, Real-Time Systems, March 1995, Volume 8, Issue 2–3, pp 173–198.

[6] R. I. Davis, A. Zabos, A. Burns, IEEE Transactions on Computers, Efficient Exact Schedulability Tests for Fixed Priority Real-Time Systems Volume 57, Issue 9, Sept. 2008.

[7] R. S. de Oliveira. Fundamentos dos Sistemas de Tempo Real. Edição do autor, 2018.

[8] R. Wilhelm, J. Engblom, A. Ermedahl, N. Holsti, S. Thesing, D. Whalley, G. Bernat, C. Ferdinand, R. Heckmann, T. Mitra, F. Mueller, I. Puaut, P. Puschner, J. Staschulat, Per Stenstrom. The worst-case execution-time problem - overview of methods and survey of tools. ACM Trans. Embed. Comput. Syst., 7(3), 1-36, May 2008.

[9] R. Wilhelm, T. Mitra, F. Mueller, I. Puaut, P. Puschner, J. Staschulat, P. Stenstr¨om, J. Engblom, A. Ermedahl, N. Holsti, S. Thesing, D. Whalley, G. Bernat, C. Ferdinand, and R. Heckmann. The Worst-Case Execution-Time Problem - Overview of Methods and Survey of Tools. ACM Transactions on Embedded Computing Systems (TECS), vol. 7, pp. 36:1–36:53, 2008.

[10] R. I. Davis and L. Cucu-Grosjean, A Survey of Probabilistic Timing Analysis Techniques for Real-Time Systems, Leibniz Trans. Embed. Syst., vol. 6, pp. 03:1-03:60, 2019.

[11] R. I. Davis and L. Cucu-Grosjean, A Survey of Probabilistic Schedulability Analysis Techniques for Real-Time Systems, Leibniz Trans. Embed. Syst., pp. 1–53, 2019.