Sistemas de Tempo Real:

Servidores de Aperiódicas

Rômulo Silva de Oliveira
Departamento de Automação e Sistemas - DAS – UFSC

romulo@das.ufsc.br
http://www.das.ufsc.br/~romulo
Outubro/2010

Referências

- J.-M. Farines, J. da S. Fraga, R. S. de Oliveira. “Sistemas de Tempo Real”. Escola de Computação 2000, IME-USP, São Paulo-SP, julho/2000.
  - Capítulo 2
- J. Liu. “Real-Time Systems”. Prentice-Hall, 2000.
  - Capítulo 7
- G. Buttazzo, “Hard Real-Time Computing Systems – Predictable Scheduling Algorithms And Applications”. 2nd edition, Springer Verlag, 2005.
  - Capítulos 5 e 6

Descrição do Problema

- Testes de escalonabilidade podem garantir deadlines
- Mas precisam assumir carga limitada
  - Tarefas periódicas
  - Tarefas esporádicas
- Em muitos sistemas existem tarefas aperiódicas
  - Nada pode ser dito sobre seus padrões de chegada
- Existem tarefas aperiódicas sem restrições de tempo real
  - Tentar minimizar o tempo médio de resposta
  - Exemplo: Transferência de arquivos de configuração, log
- Existem tarefas aperiódicas com restrições de tempo real
  - Tentar garantir o deadline do job quando ele chega (garantia dinâmica)
  - Exemplo: Interface humano-máquina

Descrição do Problema

- Testes de escalonabilidade podem garantir deadlines
- Mas precisam assumir carga limitada
  - Tarefas periódicas
  - Tarefas esporádicas
- Em muitos sistemas existem tarefas aperiódicas
  - Nada pode ser dito sobre seus padrões de chegada
- Como executar tarefas aperiódicas

  sem comprometer a garantia dada para os deadlines

  das tarefas periódicas/esporádicas ?

Definição

- Sempre devem ser cumpridos os deadlines garantidos
- O que sobra de processador é fornecido para uma tarefa especial:
  o Servidor de Aperiódicas
- O servidor de aperiódicas:
  - Executa quando isto não compromete as garantias já dadas
  - Não é uma tarefa de verdade
  - Usa seu tempo para executar os jobs aperiódicos que chegam
  - Jobs aperiódicos formam uma fila que é atendida pelo servidor de aperiódicas
  - Podem existir apenas um ou vários servidores de aperiódicas
- A questão central é:
  quando executar o servidor de aperiódicas ?

Servidores de Aperiódicas

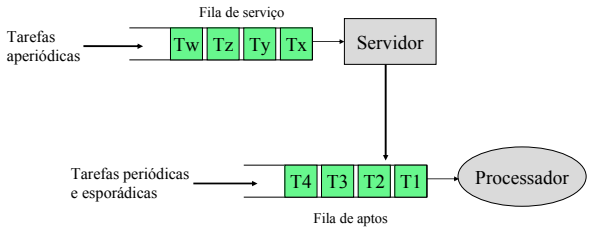
- Quando executar o servidor de aperiódicas ?
- Existem muitos tipos de servidores na literatura
- Alguns para prioridade fixa outros para prioridade variável
- Alguns são capazes de fornecer garantia dinâmica mais facilmente
- Principal diferença está em como as sobras de tempo de processamento são detectadas

  ou seja

  Como a capacidade (budget) do servidor é reabastecida

Servidores de Aperiódicas

- Duas filas são normalmente usadas



Background Server

- Executa quando o processador está idle
- Pode ser usado com prioridade fixa ou variável facilmente
- Simples de implementar
- Não afeta a escalonabilidade do sistema

- Problema: tempos de resposta elevados para os jobs aperiódicos
  - É possível melhorar isto

Background Server

tarefas	C_i	P_i	D_i	p_i
tarefa periódica A	4	10	10	1
tarefa periódica B	8	20	20	2
tarefa aperiódica C	1	-	-	3
tarefa aperiodica D	1	-	-	3

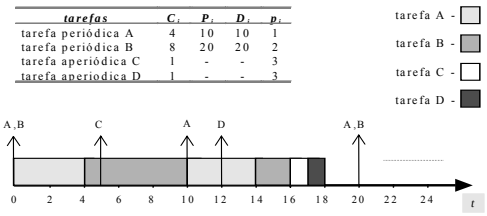


Figura 2.14: Servidora de "*Background*"

Polling Server

- Uma tarefa periódica é criada para atender a carga aperiódica
- A tarefa servidora possui um período P_{PS} e um tempo máximo de execução C_{PS} a cada período (capacidade nominal)
- Ela é escalonada como uma tarefa periódica normal
- Análise de escalonabilidade para tarefas periódicas pode ser usada
- Em cada ativação
  - A tarefa servidora executa as requisições aperiódicas pendentes dentro do limite de sua capacidade, com sua prioridade natural (fixa ou variável)

- Quando não houver requisições aperiódicas pendentes
  - A tarefa servidora suspende-se até o início do próximo período
  - Neste caso, a sua capacidade é zerada até o próximo período
  - Sua capacidade é reabastecida com C_{PS} no início do próximo período

Polling Server

tarefas	C_i	P_i	D_i	p_i
tarefa periódica A	4	10	10	3
tarefa periódica B	8	20	20	2
tarefa servidora PS	1	5	-	1
tarefa aperiódica C	1	-	-	-
tarefa aperiodica D	0.5	-	-	-

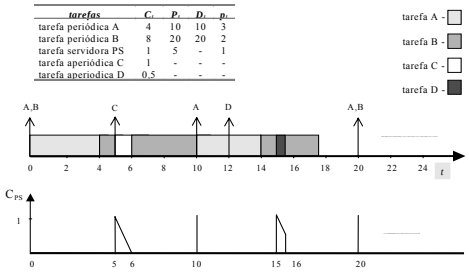
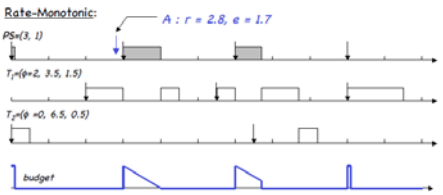


Figura 2.15: Algoritmo "Polling Server"

Polling Server

- Exemplo do livro da Jane Liu



Polling Server – Garantia Dinâmica

- Como fornecer garantia dinâmica ?
- Job aperiódico chega em um instante qualquer
  - Tempo de computação C_a e deadline D_a
- Pior caso:
  - Job aperiódico encontra servidor com capacidade zerada
  - Precisa esperar próxima reabastecimento do servidor
  - Para cada P_{PS} o servidor dispõe de C_{PS}
  - Vários períodos do servidor podem ser necessários

$$R_a = C_a + (P_{PS} - C_{PS}) + \lceil C_a / C_{PS} \rceil \cdot (P_{PS} - C_{PS})$$

  - Supondo fila do servidor vazia no momento da chegada do job aperiódico

Polling Server – Garantia Dinâmica

- Caso existam N jobs aperiódicos na fila do servidor, na frente do job a
- C_{Na} representa toda a carga aperiódica na frente do job a, incluindo job a

$$R_a = C_{Na} + (P_{PS} - C_{PS}) + \lceil C_{Na} / C_{PS} \rceil \cdot (P_{PS} - C_{PS})$$

- Caso o servidor tenha a prioridade mais alta, não existe interferência no último período

$$R_a = C_{Na} + (P_{PS} - C_{PS}) + (\lceil C_{Na} / C_{PS} \rceil - 1) \cdot (P_{PS} - C_{PS})$$

- Análise semelhante pode ser feita para outros servidores

Deferrable Server

- Uma tarefa periódica é criada para atender a carga aperiódica
- Recebe uma prioridade fixa conforme a política usada
- Jobs aperiódicos são atendidos no nível de prioridade da tarefa servidora
  - enquanto a sua capacidade C_{DS} não se esgotar no período correspondente
- No início de cada período do servidor, a sua capacidade de processamento é restaurada para C_{DS}
- Ao contrário dos servidores anteriores, servidor conserva a sua capacidade mesmo quando não existem jobs aperiódicos pendentes

Deferrable Server

tarefas	C_i	P_i	D_i	p_i
tarefa periódica A	4	10	10	3
tarefa periódica B	8	20	20	2
tarefa servidora PS	1	5	-	1
tarefa aperiódica C	1	-	-	-
tarefa aperiodica D	0,5	-	-	-

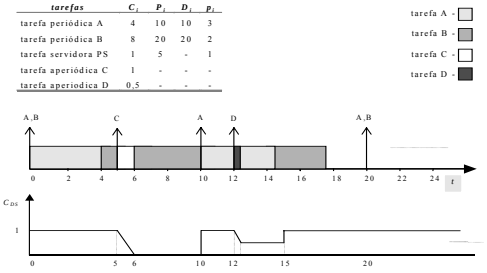
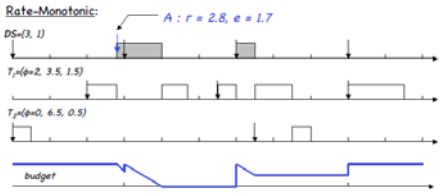


Figura 2.16: Algoritmo "Deferrable Server"

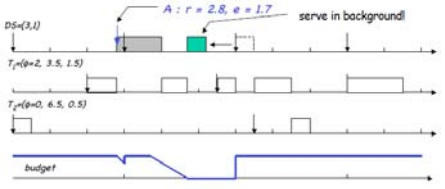
Deferrable Server

- Exemplo do livro da Jane Liu



Deferrable Server

- Deferrable Server com Background Server



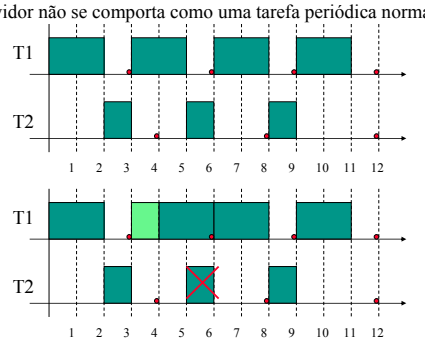
Deferrable Server

- Por preservar sua capacidade, o Deferrable Server fornece melhores tempos de resposta para as tarefas aperiódicas que o Polling Server
- Implementação semelhante ao Polling Server
- Entretanto, o comportamento do servidor com prioridade mais alta não é captado pelos testes de escalonabilidade usuais
- Por exemplo, no teste do Rate Monotonic, é suposto que a tarefa periódica de mais alta prioridade necessita executar em seu tempo de chegada, e não depois

- Não se comporta como uma tarefa periódica normal
- Necessário derivar novos testes de escalonabilidade quando Deferrable Server é usado

Deferrable Server

- Servidor não se comporta como uma tarefa periódica normal



Deferrable Server

- Rate Monotonic
- Tarefas periódicas, P=D, independentes
- Servidor com utilização U_s
- Servidor com prioridade mais alta (menor período)
- Teste de Liu&Layland pode ser adaptado

$$U_p + U_S \le U_S + n\left(\left(\frac{U_S + 2}{2U_S + 1}\right)^{1/n} - 1\right)$$

Deferrable Server

- Análise do tempo de resposta no pior caso para T_i:
  - Servidor tem prioridade mais alta
  - Executa C_{DS} imediatamente na chegada de T_i
    - Esta foi uma execução no final de um ciclo de reabastecimento
  - Imediatamente após C_{DS} tem sua capacidade reabastecida
  - A partir deste ponto, executa C_{DS} a cada P_{DS}
- Interferência recebida do servidor será

  $I_{DS} = C_{DS} + \lceil (R_i - C_{DS}) / P_{DS} \rceil . C_{DS}$

- Tempo de resposta de Ti:
  $R_i = C_i + C_{DS} + \lceil (R_i - C_{DS}) / P_{DS} \rceil . C_{DS} + SOMA_{k \in hp(i)} \lceil R_i / P_k \rceil . C_k$

- Para **m** servidores DS basta somar as **m** intereferências

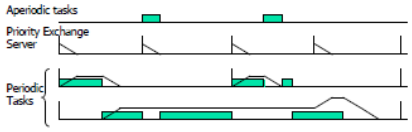
Priority Exchange Server

- Da mesma forma que o Deferrable Server, preserva sua capacidade (budget) quando não usa
- Entretanto, troca sua prioridade com qualquer tarefa de mais baixa prioridade que execute quando sua capacidade não está zerada
- No início do período a capacidade é reabastecida para C_{ES}

- Nenhum "direito novo" é criado
- Apenas os direitos são trocados entre tarefas de diferentes prioridades
- Os testes de escalonabilidade normais continuam válidos

- Implementação mais complicada

Priority Exchange Server



Sporadic Server

- O Sporadic Server corresponde a uma tarefa que atua em um só nível de prioridade para executar jobs aperiódicos
- O comportamento do Sporadic Server é equivalente a uma tarefa esporádica
  - Pode ser analisado por testes de escalonabilidade para tarefas esporádicas

- C_{SS} : Capacidade nominal do servidor
- P_{SS} : Período de recarga do servidor

- p_s : nível de prioridade em execução no processador
- p_i : um dos níveis de prioridades do sistema
- Intervalo Ativo : uma prioridade pi é dita em um intervalo ativo quando $p_i \leq p_s$
- Prioridade Desativada: uma prioridade pi é dita desativada quando $p_i > p_s$

- Tempo de Preenchimento RT_i : define o instante no qual se dá a restauração da capacidade consumida durante o intervalo em que a prioridade pi estava ativa

Sporadic Server

- O Sporadic Server preserva sempre a sua capacidade quando não está executando um job aperiódico
- Difere dos servidores anteriores quanto à forma do preenchimento de sua capacidade

- Se o servidor tem parte de sua capacidade (budget) consumida em um certo intervalo de tempo
  - O preenchimento correspondente ocorrerá no seu tempo de preenchimento RT_i
  - RT_i é determinado adicionando o valor do período do servidor ao tempo de início do intervalo onde pi era ativo e ocorreu o consumo considerado

- A quantidade a ser preenchida é igual a capacidade do servidor consumida no intervalo ativo

Sporadic Server

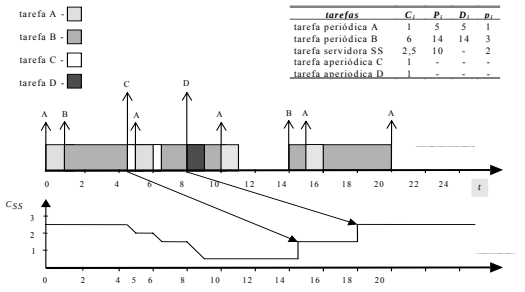


Figura 2.17: Algoritmo "Sporadic Server"

Sporadic Server

- É mais complexo de implementar do que o Polling Server e o Deferrable Server
- É mais simples de implementar do que o Priority Exchange Server
- Seu comportamento é melhor do que o Polling Server
- Testes de escalonabilidade para tarefas esporádicas podem ser usados

- Adaptação de Liu&Layland quando o servidor tem a prioridade mais alta do conjunto de tarefas (menor período):

$$U_p + U_S \leq U_S + n\left(\left(\frac{2}{U_S + 1}\right)^{1/n} - 1\right)$$

Comparação entre os Servidores para Prioridade Fixa

- 10 tarefas periódicas, períodos entre 54 e 1200
- Utilização do sistema 0,69
- Jobs aperiódicos
  - Chegadas segundo uma distribuição de Poisson
  - Tempo médio entre chegadas de 18
- Resultados em relação ao Background Server
  - 1,0 significa mesmo tempo de resposta que o Background Server fornece
  - 0,5 significa metade do tempo de resposta que o background Server fornece

Comparação entre os Servidores

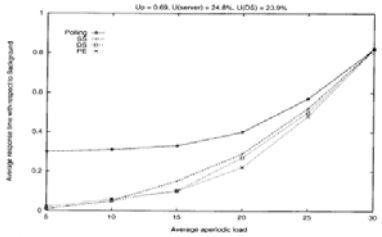


Figure from Giorgio Buttazzo, *Hard Real-time computing systems*, Kluwer Academic Publishers, 1997

Escalonamento Hierárquico

- Suponha um sistema com várias aplicações de tempo real que são independentes, cada uma com sua análise de escalonabilidade
  - Máquinas virtuais
  - Arinc 653
- Pode-se usar vários servidores independentes
- Cada servidor associado com uma aplicação/máquina virtual separada
- Cada aplicação/máquina virtual gerencia seu tempo de processador
- Escalonamento agora é feito em dois nível
  - Nível de baixo divide processador entre alguns servidores
  - Nível de cima, o tempo de cada servidor é dividido da sua própria forma
  - EDF, FCFS, RM, executivo cíclico, etc

Escalonamento Hierárquico

- Problemas:
- Como analisar a escalonabilidade das tarefas compondo o algoritmo de escalonabilidade da aplicação com a capacidade do seu servidor ?
- Qual a melhor divisão do tempo do processador entre os vários servidores ?
- Como fazer quando existem recursos que geram bloqueios entre aplicações associadas com diferentes servidores ?

Servidores de Aperiódicas para EDF

- Pode-se usar facilmente:
  - Background Server
  - Polling Server
- Existem servidores mais apropriados para EDF
- TBS – Total Bandwidth Server
- CBS – Constant Bandwidth Server

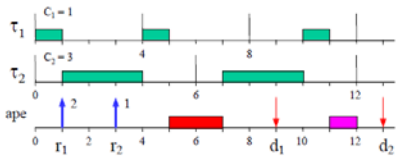
TBS – Total Bandwidth Server

- Servidor para escalonamento com EDF
- Executa jobs aperiódicos em um sistema EDF o mais cedo possível
- Mas mantém uma dada utilização (bandwidth) máxima
- Isto limita o impacto do servidor sobre as tarefas periódicas
- Quando um novo job aperiódico chega em r_a, com tempo de computação C_a, um deadline d_a é atribuído para ele

$$d_a = \max(r_a, d_{a-1}) + C_a / U_{TBS}$$

TBS – Total Bandwidth Server

- Exemplo com chegadas de aperiódicas em t=1 e t=3

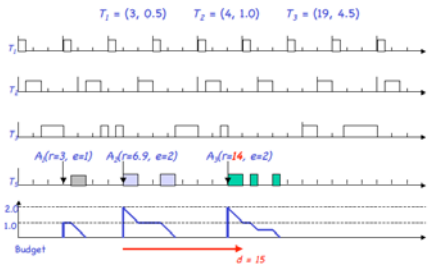


$$U_s = 1 - U_p = 1/4$$

$$\begin{cases} d_1 = r_1 + C_1 / U_s = 1 + 2 \cdot 4 = 9 \\ d_2 = \max(r_2, d_1) + C_2 / U_s = 9 + 1 \cdot 4 = 13 \end{cases}$$

TBS – Total Bandwidth Server

- Exemplo livro da Jane Liu



TBS – Total Bandwidth Server

- A dificuldade para implementar TBS é pequena
- Basta calcular um deadline absoluto sempre que um job aperiódico chega no sistema
  - E inseri-lo na fila de aptos junto com os jobs periódicos
- Tempo médio de resposta é menor do que com adaptações dos servidores estudados para prioridade fixa
- O impacto na escalonabilidade do sistema é o mesmo que uma tarefa periódica com a mesma utilização que o servidor
- Adaptação de Liu&Layland: $U_{periódicas} + U_{TBS} \leq 1$

TBS – Total Bandwidth Server

- Não existe gerência de capacidade
- Não existe proteção contra *overrun* por parte do job aperiódico
- Se um job aperiódico executa mais do que o C_a esperado, tarefas periódicas poderão perder o seu deadline
- Cada tarefa não deveria demandar mais do que sua utilização declarada $U_i = C_i / P_i$
- Se uma tarefa executa mais do que o declarado, sua prioridade deveria ser reduzida (deadline adiado no caso de EDF), para não prejudicar as demais

CBS – Constant Bandwith Server

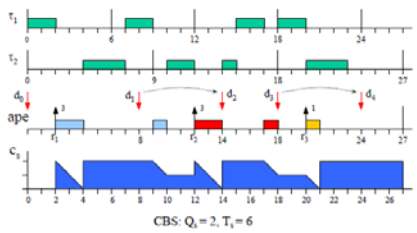
- Servidor de aperiódicas para EDF
- Inclui o conceito de capacidade nominal Q_{CBS} a cada período P_{CBS}
- A utilização (bandwidth) deste servidor será $U_{CBS} = Q_{CBS} / P_{CBS}$
- Mantém C_S (capacidade atual) e d_S (deadline do servidor)
  - Inicializados com zero
- Quando um job aperiódico chega no instante r_a, seu deadline absoluto d_a é calculado:

  Se $r_a + (C_S / U_{CBS}) \leq d_S^{atual}$
  então $d_a = d_S^{atual}$
  senão $d_a = d_S^{atual} = r_a + P_{CBS}$ e $C_S = Q_{CBS}$

- Quando a capacidade do servidor termina (C_S=0) ela é imediatamente reabastecida, mas d_S aumenta para manter uma utilização máxima: $d_S = d_S + T_{CBS}$ e $C_S = Q_{CBS}$

CBS – Constant Bandwith Server

- Melhora o TBS pois impõe isolamento entre utilizações



CBS – Constant Bandwith Server

- Implementar CBS é mais complicado do que implementar TBS
  - Necessário monitorar a capacidade do servidor
- Mais simples que servidores com prioridade fixa
  - Fila única para jobs periódicos e aperiódicos, conforme deadlines
- Tempo de resposta médio semelhante ao TBS
- Para fins de análise de escalonabilidade, equivale a tarefa periódica com utilização semelhante:

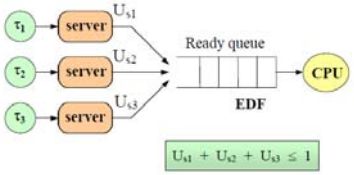
  $U_{periódicas} + U_{CBS} \leq 1$

CBS – Constant Bandwith Server

- Principal motivação para CBS:
  - Prove isolamento entre utilizações (bandwidth isolation)
- Impõe o limite de utilização do servidor independentemente da carga aperiódica que o servidor realmente executa
- Se um job aperiódico executar mais tempo que o esperado, seu deadline absoluto é automaticamente aumentado, o que na prática significa diminuir sua prioridade
- Se uma tarefa é atendida por um servidor CBS with utilização U_{CBS}, em qualquer intervalo de tempo Δ múltiplo de seu período esta tarefa (servidor) jamais demandará mais processador que $\Delta \cdot U_{CBS}$

CBS – Constant Bandwith Server

- Isolamento entre tarefas pode ser obtido através de reserva de utilização (bandwidth reservation)
- Cada tarefa é gerenciada por um servidor dedicado para ela, com utilização U_{si}
- O servidor atribui deadlines para os jobs de forma que eles não excedam



$U_{s1} + U_{s2} + U_{s3} \leq 1$

Resumo

- Servidores de aperiódicas
  - Não comprometem as garantias dadas para tarefas periódicas/esporádicas
  - Usam tempo que sobra para executar jobs aperiódicos
- Servidores para (principalmente) prioridade fixa
  - Background Server
  - Polling Server
  - Deferrable Server
  - Priority Exchange Server
  - Sporadic Server
- Escalonamento Hierárquico
- Servidores para prioridade variável
  - Total Bandwidth Server
  - Constant Bandwidth Server