Escalonamento de Tarefas

Dr. Osmar Marchi dos Santos

Escalonamento (*Scheduling*)

- Identifica o procedimento de ordenar tarefas que encontram-se disponíveis para execução no sistema
- O escalonador (*scheduler*) é o componente responsável por escolher qual a próxima tarefa a ser executada no processador
- A política de escalonamento define a forma que o escalonador irá escolher as próximas tarefas a serem executadas

Escalonamento (*Scheduling*)

- Principais abordagens de escalonamento:
  - Garantia em tempo de projeto (*off-line guarantee*):
    - Leva em consideração a carga do sistema e garante que no sistema existe reserva o suficiente para executar as tarefas considerando o pior caso
  - Garantia em tempo de execução (*on-line guarantee*):
    - Não é possível prever a carga do sistema, logo, testes são realizados durante a execução do sistema com o objetivo de garantir restrições temporais – tem de lidar com sobrecargas do sistema
  - Melhor esforço (*best-effort*):
    - Tentam encontrar uma escala do sistema sem testes ou com testes ainda mais fracos – mais usadas em sistemas de tempo real "soft"

Teste de Escalonamento

- Tem como objetivo verificar se a carga imposta pelo sistema irá conseguir ser executada pelo sistema, de forma que tarefas não percam seu(s) deadline(s)
  - Testes suficiente – bastante utilizado:
    - Conjuntos de escalonamento aceitos nesse teste são escalonáveis, porém um subconjunto descartado também pode ser escalonável
  - Testes necessários:
    - Descarta conjuntos de escalonamento não escalonáveis
  - Testes exatos (suficientes e necessários) – alta complexidade:
    - Identifica exatamente o conjunto de escalonamentos escalonáveis e não escalonáveis

Teste de Escalonamento

- Testes de escalonamento baseados em utilização são um exemplo de teste suficiente:
  - Utilização de uma tarefa: é a medida de ocupação de processador da tarefa, calculada como:
    - $U_i = C_i / P_i$
    - $U_i = C_i / mit_i$

- A utilização do processador é dada pelo somatório de todas as tarefas executadas:
  - $U = \sum_{i}^{n} U_i < 1$ (considerando um único processador)

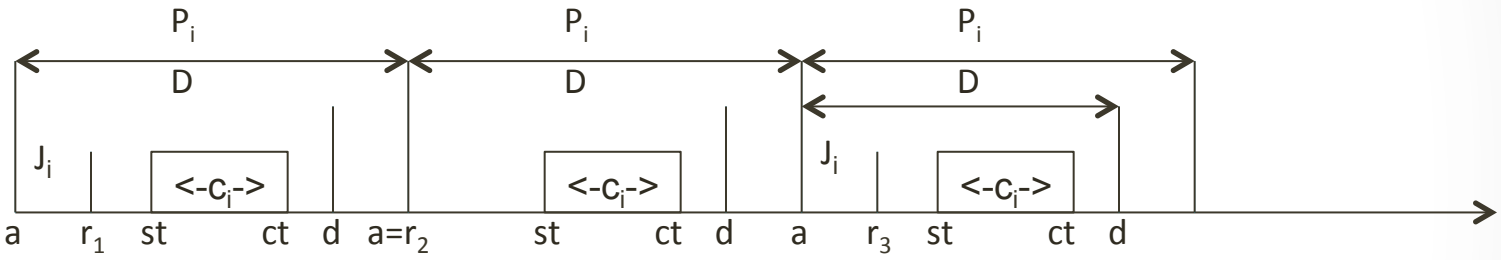
Exercício

- Qual a utilização total (*U*) de uma aplicação composta por 2 tarefas periódicas onde:
  - Tarefa 1 – maior prioridade:
    - Computação = 5 ms
    - Período = 10 ms
  - Tarefa 2 – menor prioridade:
    - Computação = 6 ms
    - Período = 15 ms

- Esse sistema é escalonável?

Modelo de Tarefas - Periódico

- Possibilita determinar *a priori* a carga do sistema através dos tempos de ativação e computação (exemplo: $U_i = C_i / P_i$)



- Garantias em tempo de projeto sobre a "escalonabilidade" do conjunto de tarefas que compõem a aplicação

Escalonamento Baseado em Prioridades

- As prioridades das tarefas são derivadas "automaticamente" das restrições temporais das tarefas
  - Ao invés de utilizar outros atributos, como a importância de uma tarefa específica para o sistema
- Esse tipo de escalonamento apresenta melhor desempenho e flexibilidade
  - Por exemplo, comparado ao modelo executivo cíclico (*cyclic executive*)
- Discutiremos três abordagens de escalonamentos baseados em prioridades:
  - RM; EDF; DM

Escalonamento Taxa Monotônica (RM)

- *Rate Monotonic* (RM) *scheduling*
  - *Proposto em 1973 por Liu e Layland em* "Scheduling algorithms for multiprogramming in a hard real-time environment”

- Considerado um algoritmo ótimo em sua classe
  - Ou seja, nenhum outro algoritmo na mesma classe consegue escalonar um conjunto de tarefas que não seja escalonável utilizando RM

Escalonamento Taxa Monotônica (RM)

- As premissas para o funcionamento do RM incluem:
  - Tarefas são periódicas e independentes
  - O “deadline” das tarefas são iguais ao seu período ($D_i = P_i$)
  - Tempo de computação da tarefa C_i é conhecido e constante (derivação do WCET)
  - O tempo *overhead* para o chaveamento entre tarefas (*context switch*) é dito nulo

Escalonamento Taxa Monotônica (RM)

- Política de definição de prioridades:
  - Prioridades decrescem em função do aumento dos períodos
  - Ou seja, quanto maior a frequência de ativação da tarefa, maior será a sua prioridade

- Exemplo:
  - Tarefa X: $P_x = 10$
  - Tarefa Y: $P_y = 5$

- Prioridade de X > Y? Ou Y > X?

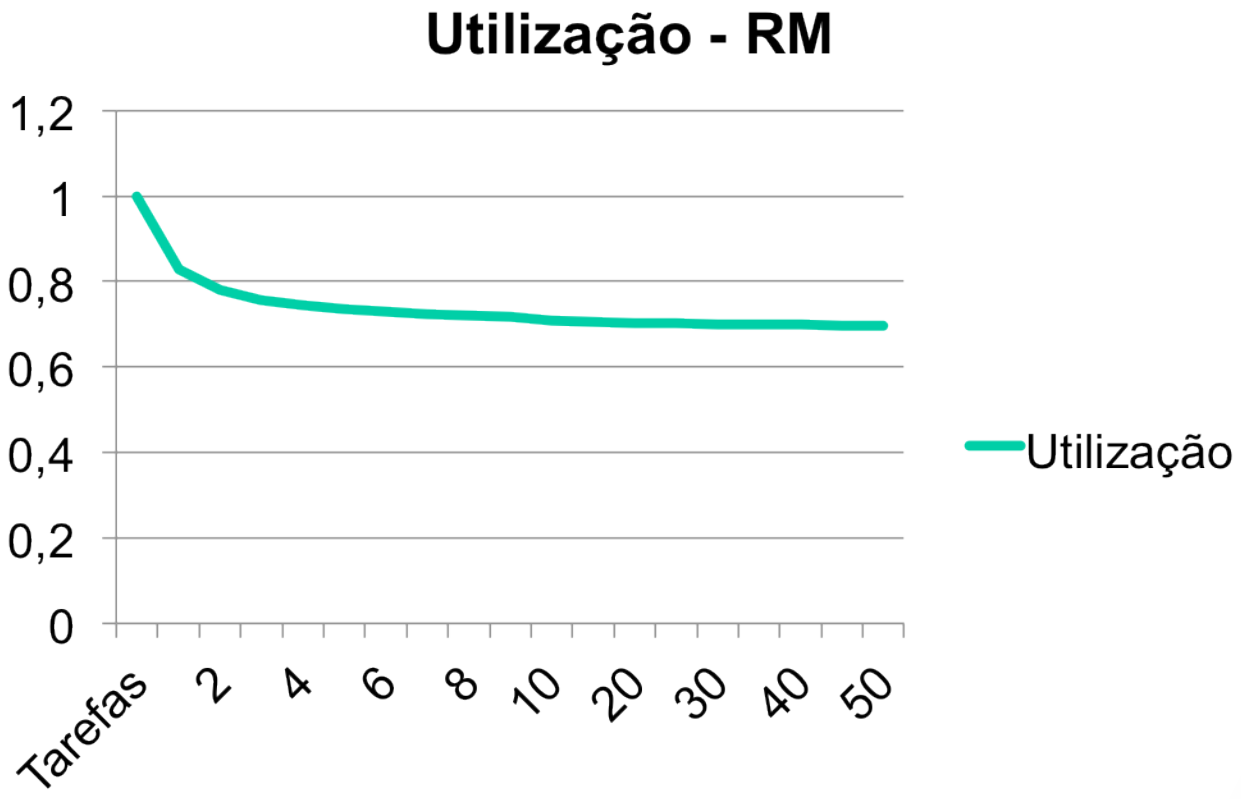
Escalonamento Taxa Monotônica (RM)

- Atribuição é feita de forma estática, em tempo de projeto, antes da execução do sistema
  - Com o uso de escalonamento baseado em filas de prioridades, em conjunto com preempção, o sistema irá se comportar (do ponto de vista temporal) de acordo com sua análise anterior

- Utilizando o escalonamento de Taxa Monotônica, é possível impor uma condição suficiente (baseada em utilização) para a análise de escalonamento:
  - $U = \sum_{i}^{n} U_i < n\,(2^{1/n} - 1)$, onde *n* é o número de tarefas da aplicação

Escalonamento Taxa Monotônica (RM)

- Utilização do processador converge para 0,69 (+- 70%)



Escalonamento Taxa Monotônica (RM)

- Considerando o seguinte sistema, veja se ele é escolanável ou não:

Tarefa	P_i	C_i	Prio. p_i	U_i
A	100	20	1	0,2
B	150	50	2	0,33
C	200	50	3	0,25

Escalonamento Taxa Monotônica (RM)

- Considerando o seguinte sistema, veja se ele é escolanável ou não:

Tarefa	P_i	C_i	Prio. p_i	U_i
A	50	10	1	0,2
B	100	30	2	0,3
C	150	50	3	0,33

Escalonamento Taxa Monotônica (RM)

- A condição suficiente pode ser “relaxada” quando as tarefas do conjunto apresentam períodos múltiplos da tarefa mais prioritária
  - Tarefas harmônicas

- Nesse caso, o limite de utilização se aproxima do máximo teórico (1 – 100%)

Escalonamento “Earliest Deadline First” (EDF)

- *Earliest Deadline First* (EDF) *scheduling*
  - *Também proposto em 1973 por Liu e Layland em* "Scheduling algorithms for multiprogramming in a hard real-time environment”

- O escalonamento ocorre dinamicamente, durante o tempo de execução através de um escalonador preemptivo e baseado em prioridades

Escalonamento “Earliest Deadline First” (EDF)

- As premissas para o funcionamento do EDF incluem:
  - Tarefas são periódicas e independentes
  - O “deadline” das tarefas são iguais ao seu período ($D_i = P_i$)
  - Tempo de computação da tarefa C_i é conhecido e constante (derivação do WCET)
  - O tempo *overhead* para o chaveamento entre tarefas (*context switch*) é dito nulo

- Mesmas descritas anteriormente para o RM

Escalonamento “Earliest Deadline First” (EDF)

- Política de escalonamento:
  - No EDF a atribuição dinâmica de prioridades ocorre a cada momento que uma tarefa entra na fila de processos prontos para escalonamento
  - Nesse momento, ocorre uma “reordenação” das tarefas na fila para encontrar o processo cujo deadline absoluto está mais próximo (considerando o tempo atual do sistema)
  - Esse processo, com o deadline absoluto mais próximo é escolhido para execução

- Exemplo:
  - Tarefa X: $P_x = 7$
  - Tarefa Y: $P_y = 5$

Escalonamento “Earliest Deadline First” (EDF)

- Uma das grandes vantagens do EDF é que seu teste suficiente e necessário é o limite teórico:
  - $U = \sum_{i}^{n} U_i <= 1$, onde *n* é o número de tarefas da aplicação
- Problemas podem ser:
  - Maior overhead de escalonamento
  - Possíveis imprevisibilidade na ocorrência de falhas

Comparação

Tarefa	P_i	C_i	D_i
A	20	10	20
B	50	25	50

Escalonamento Deadline Monotônico (DM)

- *Deadline Monotonic* (DM) *scheduling*
  - Estende o RM que limitava períodos para deadlines
  - Também considerado um algoritmo ótimo em sua classe

- As premissas para o funcionamento do DM incluem:
  - Tarefas são periódicas e independentes
  - O "deadline" das tarefas são iguais ou menores ao seu período ($D_i <= P_i$)
  - Tempo de computação da tarefa C_i é conhecido e constante (derivação do WCET)
  - O tempo *overhead* para o chaveamento entre tarefas (*context switch*) é dito nulo

- Análise "exata" fica para a próxima aula