Sistemas de Tempo-Real

Notas de curso realizado em Agosto de 2006 na
Universidade Federal do Rio Grande do Norte, Natal, Brasil

Francisco Vasques
Faculdade de Engenharia da
Universidade do Porto
http://www.fe.up.pt/~vasques

2. Escalonamento de Tempo-Real (parte 2)

Plano das Aulas

- **Introdução aos Sistemas de Tempo-Real**
- **Escalonamento de Tempo-Real**
  - Conceitos e Definições
  - Classificação de Algoritmos de Escalonamento
  - Algoritmos Clássicos de Escalonamento
  - Considerações Suplementares
  - Exclusão Mútua no Acesso a Recursos Partilhados
  - Exemplo de Escalonamento em Computação Industrial
- **Protocolos de Comunicação de Tempo-Real**

Plano das Aulas

- **Escalonamento de Tempo-Real**
  - Conceitos e Definições
  - Classificação de Algoritmos de Escalonamento
  - Algoritmos Clássicos de Escalonamento
  - Considerações Suplementares
  - Exclusão Mútua no Acesso a Recursos Partilhados
    - » Noções básicas de sincronização
    - » Herança de prioridades
    - » Protocolo de tecto de prioridades
    - » Exemplo: "*Mars PathFinder Mission*"
  - Exemplo de Escalonamento em Computação Industrial

Exclusão Mútua no Acesso a Recursos

- **Noções básicas de sincronização**
  - Objectivo da sincronização: esperar pela ocorrência de um evento antes de executar uma tarefa.

Relações inter-tarefas de 3 tipos:

1. Relação de precedência (sincronização por ocorrência de evento)
   - Noção de ordem entre a execução de instâncias de diferentes tarefas.
2. Partilha de variáveis / recursos
   - Troca de informação ou de resultados entre tarefas;
   - Necessidade de protecção através da utilização de semáforos, para impedir que uma variável partilhada seja acedida simultaneamente em escrita por diferentes tarefas.
3. Comunicação (sincronização por envio de mensagem)
   - Precedência + transferência de informação: relação do tipo produtor/consumidor;

Exclusão Mútua no Acesso a Recursos

- **Noções básicas de sincronização**

  Gestão de comunicação entre tarefas

  – Modelo funcional 

  – Classificação de sistemas de comunicação entre tarefas por mensagens

    » síncrona / assíncrona

      - síncrona: a tarefa receptora coloca-se explicitamente à espera da mensagem
      - assíncrona: um gestor de mensagem é activado à chegada da mensagem.

    » com / sem colocação em fila de espera

      - com: memorização numa fila de espera ("*mailbox*")
      - sem: escrita sobre mensagem anterior ("*buffer*")

    » com / sem bloqueio

      - sem bloqueio na emissão: possível perda de mensagens, se a fila de recepção tiver uma dimensão insuficiente.

Exclusão Mútua no Acesso a Recursos

- **Exclusão Mútua no Acesso a Recursos Partilhados**

  Partilha de variáveis / recursos

  – A garantia de exclusividade no acesso a recursos partilhados (secções críticas) tem que ser garantida, por forma a impedir que uma variável partilhada seja acedida simultaneamente em escrita por diferentes tarefas.

  – Num escalonamento não-preemptivo:

    » A exclusividade está garantida por inerência, visto que não existe preempção durante os intervalos de utilização/acesso aos recursos partilhados.

  – Num escalonamento preemptivo por prioridades:

    » A garantia de exclusividade no acesso a recursos partilhados pode ser obtida através da utilização de semáforos.

Exclusão Mútua no Acesso a Recursos

- **Exclusão Mútua no Acesso a Recursos Partilhados**

  Partilha de variáveis / recursos

  – A garantia de exclusividade no acesso a recursos partilhados (secções críticas) pode ser obtida através da utilização de semáforos;

  – Princípio de funcionamento:

  » Uma tarefa não pode aceder a um recurso partilhado, a menos que tenha bloqueado o semáforo que protege o acesso a esse recurso;

  » Para bloquear esse semáforo, tem que esperar que este fique livre;

  » Quando a tarefa termina o acesso ao recurso partilhado, deve libertar o semáforo de protecção.

Universidade do Porto
Faculdade de Engenharia

Exclusão Mútua no Acesso a Recursos

- **Exclusão Mútua no Acesso a Recursos Partilhados**

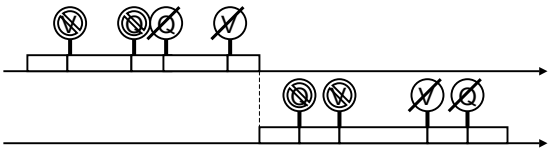
  – Consequência:

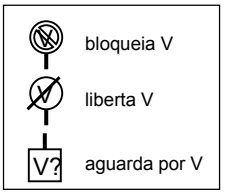
  » existência de *intervalos de bloqueio*, durante os quais tarefas de menor prioridade bloqueiam a execução de tarefas de maior prioridade.

  » É fundamental que estes intervalos de bloqueio sejam limitados e mensuráveis, caso contrário a execução das tarefas não poderá ter garantias de tempo-real.

Exclusão Mútua no Acesso a Recursos

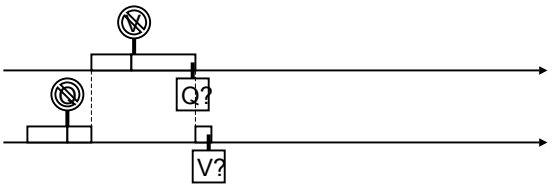
- **Exemplo 1:**
  - Execução de 2 tarefas, sendo que cada uma das tarefas efectua acessos a 2 recursos partilhados protegidos pelos semáforos V e Q (respectivamente).





Exclusão Mútua no Acesso a Recursos

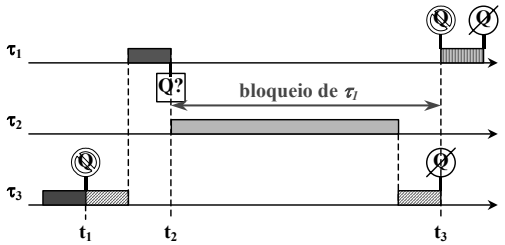
- **Exemplo 2:**
  - Situação de *impasse (“deadlock”)* devido a bloqueios cruzados
    - Cada uma das tarefas tenta executar sobre recursos bloqueados pela outra tarefa.



  - A ocorrência de *bloqueios cruzados* pode ser considerada como consequência de uma prática de desenvolvimento deficiente, visto este tipo de bloqueios poder ser facilmente evitado na fase de concepção do software.

Exclusão Mútua no Acesso a Recursos

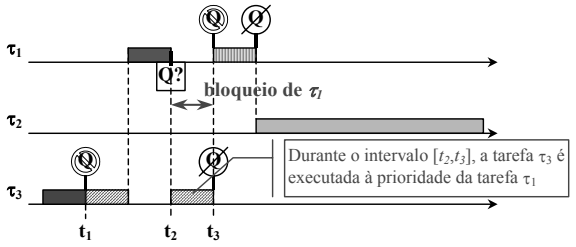
- **Situação de bloqueio prolongado**
  - ... devido à possibilidade que as tarefas de prioridade intermédia têm de bloquear tarefas de prioridade mais elevada;



  - O bloqueio ocorre não só durante o intervalo de tempo em que a tarefa τ_3 acede ao recurso protegido por Q, mas também durante um *intervalo de tempo prolongado* durante o qual tarefas de prioridade intermédia executam sobre o processador.

Exclusão Mútua no Acesso a Recursos

- **Herança de prioridades [SRL90]**
  - Através de um mecanismo de *herança de prioridades* é possível eliminar situações de bloqueio prolongado.



  - A herança de prioridades impõe que:
    "sempre que uma tarefa de prioridade inferior bloqueia uma tarefa de prioridade superior, então a tarefa que provoca o bloqueio passa a ser executada com a prioridade da tarefa bloqueada".

Exclusão Mútua no Acesso a Recursos

- **Herança de prioridades [SRL90]**
  - O tempo máximo de bloqueio é limitado superiormente (excepto se houver situações de impasse), logo continua a ser possível garantir o respeito das metas temporais.
  - Cálculo do tempo de bloqueio B_i
    - » Se uma tarefa τ_i tem *K* secções críticas que podem levar ao seu bloqueio, o máximo n.º de vezes que a tarefa τ_i pode ser bloqueada é *K*.
    - » O máximo tempo de bloqueio de uma tarefa τ_i é : $B_i = \sum_{k=1}^{K} usage(k,i)C(k)$
  - Máximo Tempo de Resposta: $R_i = C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \times C_j$

$$w_i^{x+1} = C_i + B_{i+} \sum_{j \in hp(i)} \left\lceil \frac{w_i^x}{T_j} \right\rceil \times C_j$$

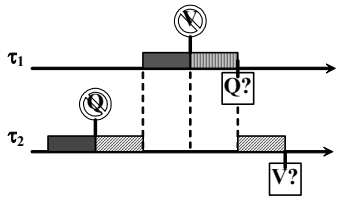
Universidade do Porto
Faculdade de Engenharia

Exclusão Mútua no Acesso a Recursos

- **Herança de prioridades [SRL90]**
  - Comentários
    - » Para implementar o protocolo de herança de prioridades, não é necessário saber qual a tarefa que utiliza qual recurso.
    - » Esta metodologia não elimina a *inversão de prioridades*, limitando-se a impor um limite superior a esta inversão.

Exclusão Mútua no Acesso a Recursos

- **Herança de prioridades [SRL90]**
  - Não resolve o problema de impasse devido a bloqueios cruzados…



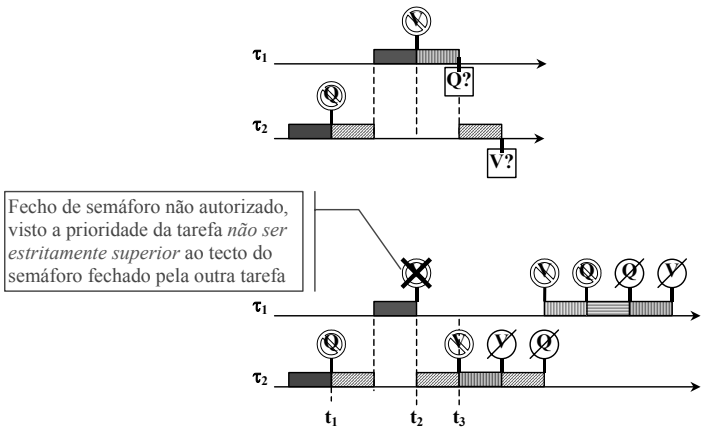
  - Este problema pode ser resolvido aplicando o Protocolo de Tecto de Prioridades.

Exclusão Mútua no Acesso a Recursos

- **Protocolo de tecto de prioridades [SRL90]**
  - Define-se o *tecto de um semáforo*, como sendo o valor máximo de prioridade das tarefas que o podem bloquear.
  - Em tempo de execução, aplicam-se as seguintes regras:
    - » *herança de prioridades*, caso uma tarefa de prioridade inferior bloqueie uma tarefa de prioridade superior;
    - » *tecto de prioridades*, impondo que uma tarefa τ_i só pode fechar um determinado semáforo, caso a sua prioridade no momento seja estritamente superior à de todos os tectos dos semáforos fechados pelas outras tarefas.

Exclusão Mútua no Acesso a Recursos

- **Protocolo de tecto de prioridades [SRL90]**



Exclusão Mútua no Acesso a Recursos

- **Protocolo de tecto de prioridades [SRL90]**
  - *Propriedades*
    - » nenhuma tarefa sofre mais do que *uma situação de inversão de prioridades* durante a sua execução;
    - » a hipótese de impasse é eliminada;
    - » em contrapartida, uma tarefa pode sofrer uma inversão de prioridades durante um intervalo de tempo superior ao estritamente necessário.
  - O *tempo máximo* que uma tarefa pode permanecer bloqueada (B_i) é igual ao tempo de execução da mais longa secção crítica em recursos de prioridade inferior que sejam acedidos por tarefas de prioridade superior.

Exclusão Mútua no Acesso a Recursos

- **Protocolo de tecto de prioridades [SRL90]**
  - *Propriedades*
    - » Um *teste suficiente* de escalonabilidade para um conjunto de tarefas com prioridades atribuídas pelo algoritmo RM será:

$$\forall i, 1 \le i \le n : \frac{B_i}{T_i} + \sum_{i=1}^{n} \frac{C_i}{T_i} \le i\left(\sqrt[i]{2} - 1\right)$$

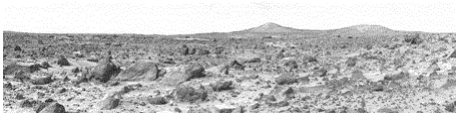
    - » O *tempo de resposta* R_i de uma tarefa poderá ser calculado através da formulação tradicional, à qual deve ser adicionado um termo B_i representando o bloqueio máximo que a tarefa τ_i pode sofrer:

$$R_i = C_i + B_i + I_i$$

$$I_i = \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \times C_j$$

Exclusão Mútua no Acesso a Recursos

- **Exemplo: "*Mars PathFinder Mission*"**
  - Missão não tripulada a Marte (chegada a Marte em 1997/07/04) para recolha de dados meteorológicos, imagens fotográficas, etc.;
  - Incluiu a colocação na superfície de um robot móvel autónomo ("*Sojourner Rover*"), através de uma "aterragem" não convencional.

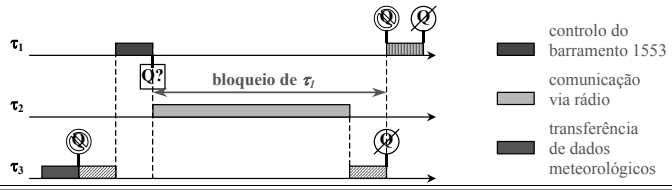


Exclusão Mútua no Acesso a Recursos

- **O problema**
  - Após alguns dias de missão, o *sistema computacional começou a ser reinicializado frequentemente*, ficando incapaz de enviar para a Terra os dados meteorológicos entretanto adquiridos;
- **Descrição sumária do sistema computacional**
  - Sistema mono-processador com sistema operativo VxWorks (escal. preemptivo por prioridades), sobre barramento VME para ligação aos sistemas de rádio e de visão e a um barramento 1553.
  - O segundo barramento (1553) efectua a ligação aos andares de "cruzeiro" (que inclui um sensor solar e um "*scanner*" de estrelas) e de "aterragem" (que inclui acelerómetros, um altímetro e um instrumento de aquisição de dados meteorológicos:ASI/MET) da sonda espacial.

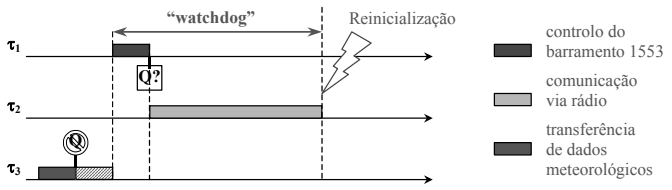
Exclusão Mútua no Acesso a Recursos

- **Análise do problema**
  - De entre diversas tarefas, salientam-se as 3 seguintes:
    - » *tarefa periódica* de controlo do barramento 1553 (*alta prioridade*);
    - » *tarefa de comunicação* via rádio (prioridade intermédia);
    - » *tarefa de transferência de dados* meteorológicos através dos barramentos 1553 / VME (*baixa prioridade*)
  - O barramento 1553 é um *recurso partilhado* protegido por um semáforo sem mecanismos de herança de prioridades. Em consequência, são induzidas situações de bloqueio prolongado.



Exclusão Mútua no Acesso a Recursos

- **Análise do problema**
  - A execução da tarefa de prioridade mais elevada (controlo do barramento 1553) estava *"protegida" por um watchdog*, que efectuava a *reinicialização total do sistema*, caso a tarefa não fosse escalonada num intervalo de tempo pré-determinado;
  - Concluindo, devido ao facto de os mecanismos de *herança de prioridades* não estarem activados, o envio de grandes quantidades de dados tinha como consequência a reinicialização do sistema, resultando na perda de dados adquiridos.



Exclusão Mútua no Acesso a Recursos

- **Resolução do problema**
  - Foi utilizado um mecanismo de "*trace/log*" disponível no sistema operativo VxWorks para recolher informação sobre o sistema nos instantes adequados, tendo sido identificada a fonte da avaria;
  - A resolução do problema passou pela *activação dos mecanismos de herança de prioridades* (modificação de parâmetros do semáforo). Para tal foi utilizado um interpretador de linguagem C existente no sistema operativo, normalmente utilizado para efectuar a depuração de programas em modo de funcionamento.

Fontes:

Mike Jones, "*What really happened on Mars?*", Risks Forum, RISKS-19.49, Dec 1997.

"*Authoritative Account about What really happened on Mars*", mensagem de esclarecimento de Glenn Reeves, Responsável pelo "*Mars Pathfinder Flight Software*" também disponível em:
http://research.microsoft.com/~mbj/Mars_Pathfinder/

Universidade do Porto
Faculdade de Engenharia

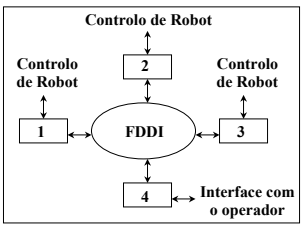
Plano das Aulas

- **Escalonamento de Tempo-Real**
  - Conceitos e Definições
  - Classificação de Algoritmos de Escalonamento
  - Algoritmos Clássicos de Escalonamento
  - Considerações suplementares
  - Exclusão Mútua no Acesso a Recursos Partilhados
  - Exemplo de Escalonamento em Computação Industrial

Universidade do Porto
Faculdade de Engenharia

Exemplo de Escalonamento em Computação Industrial

- **Sistema de Tempo-Real para aplicação na área da Robótica[1]**
  - 4 nós ligados em rede (FDDI): nós 1-3 estão dedicados a aplicações robóticas (medição da forma de tubos, utilizando sensores de distância); nó 4 está dedicado à interface com o operador



- **Requisitos temporais:**
  - A execução local das diferentes tarefas deve ser efectuada de acordo com as metas temporais locais (*"deadlines"*);
  - A execução de tarefas sobre múltiplos recursos computacionais (rede, processadores, etc.) está sujeita a metas temporais de extremo-a-extremo ("*end-to-end deadline*").

1. *M. Klein, J. Lehoczky, R. Rajkumar, "Rate-Monotonic Analysis for Real-Time Industrial Computing", IEEE Computer, January 1994.*

Exemplo de Escalonamento em Computação Industrial

- **Sistema Computacional (nós 1, 2, 3)**
  - Nós 1, 2 e 3 do sistema são similares;
  - Tarefas:

    τ1 (periódica): Controlo de posição;

    τ2 (periódica): Sistema de medição, process. e envio de dados para nó 4;

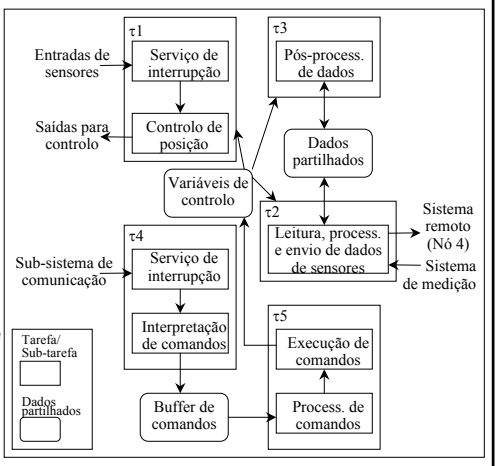
    τ3 (periódica c/ *atraso*): Sistema de pós-process. de dados para uso local;

    τ2 e τ3 estão sincronizadas;

    τ4 (espor.): Recepção e interpretação de comandos provenientes do nó 4;

    τ5 (espor.): Processamento e execução de comandos;

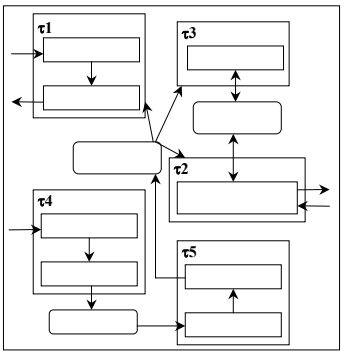
    τ4 e τ5 estão sincronizadas;



Exemplo de Escalonamento em Computação Industrial

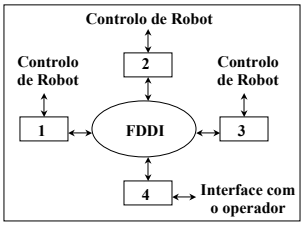
- **Requisitos temporais (nós 1, 2, 3):**
  - As sub-tarefas executam com prioridades idênticas (em [1] são atribuídas de uma forma arbitrária diferentes prioridades às diferentes sub-tarefas);
  - A tarefa τ2 do nó 1 está sujeita a uma meta temporal de extremo-a-extremo ("*end-to-end deadline*") de 500ms;
  - As secções críticas das diferentes tarefas (rotinas de interrupção; escrita em dados partilhados) estão indicadas a negrito.

	T_i	C_{i1}	C_{i2}	C_{i3}	C_{i4}	D_i	U_i
τ1	40	**1**	5			40	0,150
τ2	50	7	11	**2**		50	0,400
τ3	100	10	**5**	5		100	0,200
τ4	200	**8**	18	**3**	2	200	0,155
τ5	400	2	12	**10**		400	0,060
U							0,965



Exemplo de Escalonamento em Computação Industrial

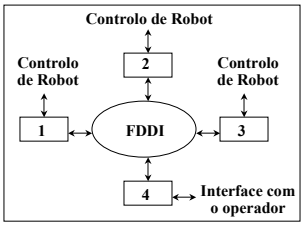
- **Sistema Computacional (nó 4)**
  - Tarefas $\tau 1$ e $\tau 3$ lêem e visualizam dados provenientes dos nós 2 e 3;
  - Tarefas $\tau 1$ e $\tau 3$ acedem em exclusão mútua a um recurso partilhado;
  - Tarefa $\tau 2$ lê e visualiza dados provenientes do nó 1. Esta tarefa tem uma meta temporal de 200ms (parte da meta temporal de extremo-a-extremo de 500ms relativa aos dados provenientes do nó 1).



	T_i (ms)	C_i (ms)	D_i (ms)	Secção crítica	U_i
$\tau 1$	80	20	80	4	0,25
$\tau 2$	100	61	**200**		0,61
$\tau 3$	300	30	300	5	0,10
U					0,96

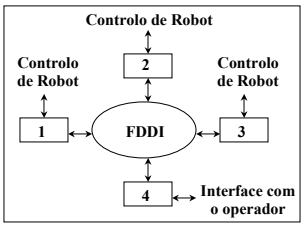
Exemplo de Escalonamento em Computação Industrial

- **Sistema Computacional (rede)**
  - FDDI: Rede em anel, com débito de 100Mbit/s;
  - Acesso à rede controlado por passagem de "*token*", estando cada nó autorizado a transmitir unicamente quando estiver na posse do "*token*";
  - O tempo de ciclo do "*token*" foi fixado em 8ms, sendo o tempo de transmissão disponível para o conjunto dos nós de 7ms (os mecanismos de passagem de "*token*" consomem 1ms por ciclo);
  - Em cada nó foram fixados os seguintes tempos de permanência do "*token*"
    - 2,1ms para cada um dos nós 1, 2 e 3;
    - 0,7ms para o nó 4.
  - A atribuição efectuada aos nós 1, 2 e 3 permite que as respectivas tarefas $\tau 2$ transfiram 1Mbit de dados para o nó 4 todos os 50ms.



Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: fundamentos da análise efectuada**
  - decomposição do problema de escalonamento global de recursos em múltiplos problemas de escalonamento local (nós e rede);
  - O envio de dados provenientes de sensores do nó 1 deve ser visualizado no nó 4 com uma meta temporal de extremo-a-extremo de 500ms.



    - » Estes dados deverão ser escalonados no nó 1, rede e nó 4, pelo que a soma das metas temporais locais deverá ser inferior à meta temporal de extremo-a-extremo (500ms);
    - » A atribuição das metas temporais locais faz-se de acordo com o período das tarefas correspondentes: 50ms, 50ms e 200ms (e não 100ms, devido ao excesso de carga colocado pela tarefa $\tau 2$ do nó 4: 61%);

Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: nó 4 (ignorando secções críticas)**
  - Teste RM (utilização): $U = \sum_{i=1}^{n} \frac{C_i}{T_i} \le n\left(\sqrt[n]{2}-1\right)$ $\quad \sum_{i=1}^{3} \frac{C_i}{T_i} = 0{,}96 \ge 0{,}779 = 3\left(\sqrt[3]{2}-1\right)$
  - Análise do máximo tempo de resposta: $R_i = C_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \times C_j$

	T_i (ms)	C_i (ms)	D_i (ms)	R_i(ms)
$\tau 1$	80	20	80	20
$\tau 2$	100	61	**200**	101
$\tau 3$	300	30	300	293

$\boxed{R_1 = 20ms}$

$w_2^0 = 61 + 1(20) = 81ms$
$w_2^1 = 61 + 2(20) = 101ms$
$w_2^2 = 61 + 2(20) = 101ms$
$\boxed{R_2 = 101ms}$

$w_3^0 = 30 + 1(20) + 1(61) = 111ms$
$w_3^1 = 30 + 2(20) + 2(61) = 192ms$
$w_3^2 = 30 + 3(20) + 2(61) = 212ms$
$w_3^3 = 30 + 3(20) + 3(61) = 273ms$
$w_3^4 = 30 + 4(20) + 3(61) = 293ms$
$w_3^5 = 30 + 4(20) + 3(61) = 293ms$
$\boxed{R_3 = 293ms}$

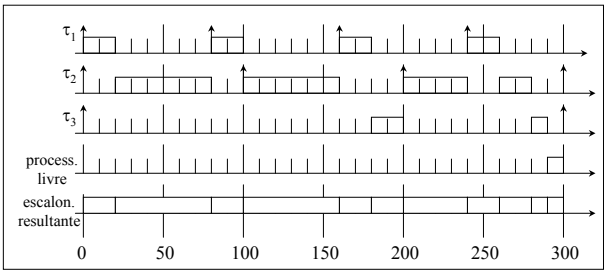
  - » O teste suficiente de utilização não é respeitado;
  - » Os tempos de resposta respeitam as metas temporais. No entanto, devido ao facto de $D_2 > T_2$, a regra do algoritmo RM/DM ($D_i \le T_i$) não é respeitada, logo não é suficiente verificar a escalonabilidade após o instante crítico.

Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: nó 4**

  (ignorando secções críticas)

	T_i (ms)	C_i (ms)	D_i (ms)	R_i(ms)
τ1	80	20	80	20
τ2	100	61	**200**	101
τ3	300	30	300	293



$$w_2^2 = 61 + 2(20) = 101ms$$

$$w_3^5 = 30 + 4(20) + 3(61) = 293ms$$

Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: nó 4 (ignorando secções críticas)**
  - Generalização do teste RM (utilização) para o caso $D_i = \Delta . T_i$, com Δ=2,3,...

$$\sum_{i=1}^{n} \frac{C_i}{T_i} \leq \Delta(n-1)\left(\sqrt[n-1]{\frac{\Delta+1}{\Delta}} - 1\right)$$

  - » Utilizando o teste generalizado para n=2 e Δ=2: $0{,}86 \leq 2(1{,}5-1) = 1{,}0$
  - » Em consequência, todas as invocações de t2 no nó 4 são terminadas antes da meta temporal de 200ms.

	T_i (ms)	C_i (ms)	D_i (ms)	R_i(ms)
τ1	80	20	80	20
τ2	100	61	**200**	101
τ3	300	30	300	293

Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: nó 4 (considerando as secções críticas)**
  - efectuada através da análise do tempo de resposta, considerando que o tempo de bloqueio máximo B_i (provocado por uma tarefa de menor prioridade) a que uma tarefa τ_i pode ficar sujeita é: $R_i = C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \times C_j$
  - ou seja, sendo R_i^* o tempo de resposta anteriormente calculado, então:

$$R_i = R_i^* + B_i$$

	T_i (ms)	C_i (ms)	D_i (ms)	R_i^* (ms)	Secção crítica	R_i (ms)
τ1	80	20	80	20	4	25
τ2	100	61	**200**	101		106
τ3	300	30	300	293	5	293

» pelo que o conjunto de tarefas é escalonável.

Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: nós 1, 2 e 3**

	T_i	C_{i1}	C_{i2}	C_{i3}	C_{i4}	D_i	U_i
τ1	40	**1**	5			40	0,150
τ2	50	7	11	**2**		**50**	0,400
τ3	100	10	**5**	5		100	0,200
τ4	200	**8**	18	**3**	2	200	0,155
τ5	400	2	12	**10**		400	0,060
U							0,965

  - Teste RM (utilização):

$$U = \sum_{i=1}^{n} \frac{C_i}{T_i} \le n\left(\sqrt[n]{2} - 1\right)$$

$$\sum_{i=1}^{5} \frac{C_i}{T_i} = 0{,}965 \ge 0{,}7435 = 5\left(\sqrt[5]{2} - 1\right)$$

» Teste suficiente não respeitado, pelo que nada se pode concluir acerca da escalonabilidade das tarefas

  - Análise do máximo tempo de resposta: $R_i = C_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \times C_j$

$$\boxed{R_1 = 6ms}$$

$$w_2^0 = 20 + 1(6) = 26ms$$
$$w_2^1 = 20 + 1(6) = 26ms$$
$$\boxed{R_2 = 26ms}$$

$$w_3^0 = 20 + 1(6) + 1(20) = 46ms$$
$$w_3^1 = 20 + 2(6) + 1(20) = 52ms$$
$$w_3^2 = 20 + 2(6) + 2(20) = 72ms$$
$$w_3^3 = 20 + 2(6) + 2(20) = 72ms$$
$$\boxed{R_3 = 72ms}$$

Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: nós 1, 2 e 3**
  - Análise do máximo tempo de resposta: $R_i = C_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \times C_j$

	T_i	C_{i1}	C_{i2}	C_{i3}	C_{i4}	D_i	U_i
τ1	40	**1**	5			40	0,150
τ2	50	7	11	**2**		**50**	0,400
τ3	100	10	**5**	5		100	0,200
τ4	200	**8**	18	**3**	2	200	0,155
τ5	400	2	12	**10**		400	0,060
U							0,965

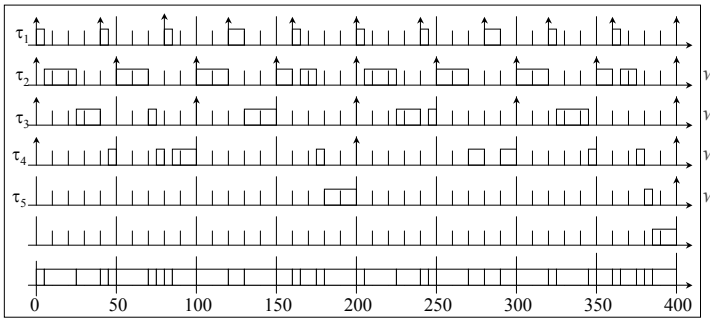
$$w_5^0 = 24 + 1(6) + 1(20) + 1(20) + 1(31) = 101ms$$
$$w_5^1 = 24 + 3(6) + 3(20) + 2(20) + 1(31) = 173ms$$
$$w_5^2 = 24 + 5(6) + 4(20) + 2(20) + 1(31) = 305ms$$
$$w_5^3 = 24 + 6(6) + 5(20) + 3(20) + 2(31) = 282ms$$
$$w_5^4 = 24 + 8(6) + 6(20) + 3(20) + 2(31) = 314ms$$
$$w_5^5 = 24 + 8(6) + 7(20) + 4(20) + 2(31) = 354ms$$
$$w_5^6 = 24 + 9(6) + 8(20) + 4(20) + 2(31) = 380ms$$
$$w_5^7 = 24 + 10(6) + 8(20) + 4(20) + 2(31) = 386ms$$
$$w_5^8 = 24 + 10(6) + 8(20) + 4(20) + 2(31) = 386ms$$
$$\boxed{R_5 = 386ms}$$

$$w_4^0 = 31 + 1(6) + 1(20) + 1(20) = 77ms$$
$$w_4^1 = 31 + 2(6) + 2(20) + 1(20) = 103ms$$
$$w_4^2 = 31 + 3(6) + 3(20) + 2(20) = 149ms$$
$$w_4^3 = 31 + 4(6) + 3(20) + 2(20) = 155ms$$
$$w_4^4 = 31 + 4(6) + 4(20) + 2(20) = 175ms$$
$$w_4^5 = 31 + 5(6) + 4(20) + 2(20) = 181ms$$
$$w_4^6 = 31 + 5(6) + 4(20) + 2(20) = 181ms$$
$$\boxed{R_4 = 181ms}$$

Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: nós 1, 2 e 3**

	T_i	C_{i1}	C_{i2}	C_{i3}	C_{i4}	D_i	R_i
τ1	40	**1**	5			40	6
τ2	50	7	11	**2**		**50**	26
τ3	100	10	**5**	5		100	72
τ4	200	**8**	18	**3**	2	200	181
τ5	400	2	12	**10**		400	386

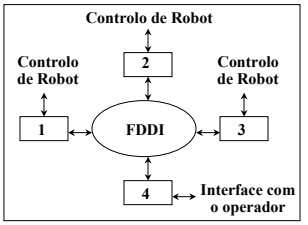


$$w_2^1 = 20 + 1(6)$$
$$w_3^3 = 20 + 2(6) + 2(20)$$
$$w_4^6 = 31 + 5(6) + 4(20) + 2(20)$$
$$w_5^8 = 24 + 10(6) + 8(20) + \\ + 4(20) + 2(31)$$

Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: nós 1, 2 e 3**
  - Os tempos de resposta respeitam as metas temporais, logo o conjunto de tarefas é localmente escalonável (falta considerar as secções críticas).
  - Associado à tarefa $\tau 2$ existe ainda uma meta temporal de extremo-a-extremo, cuja validade deverá ser posteriormente verificada.

	T_i	C_{i1}	C_{i2}	C_{i3}	C_{i4}	D_i	R_i
$\tau 1$	40	**1**	5			40	6
$\tau 2$	50	7	11	**2**		**50**	26
$\tau 3$	100	10	**5**	5		100	72
$\tau 4$	200	**8**	18	**3**	2	200	181
$\tau 5$	400	2	12	**10**		400	386
							(ms)



Exemplo de Escalonamento em Computação Industrial

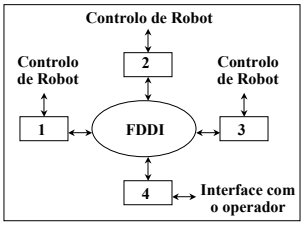
- **Análise temporal: nós 1, 2 e 3 (considerando as secções críticas)**
  - efectuada através da análise do tempo de resposta, considerando que o tempo de bloqueio máximo B_i (provocado por uma tarefa de menor prioridade) a que uma tarefa τ_i pode ficar sujeita é: $R_i = C_i + B_i + \sum_{j \in hp(i)} \left\lceil \frac{R_i}{T_j} \right\rceil \times C_j$
  - ou seja, sendo R_i^* o tempo de resposta anteriormente calculado, então: $R_i = R_i^* + B_i$

	T_i	C_{i1}	C_{i2}	C_{i3}	C_{i4}	D_i	R_i^*	B_i	R_i
$\tau 1$	40	**1**	5			40	6	10	16
$\tau 2$	50	7	11	**2**		50	26	10	36
$\tau 3$	100	10	**5**	5		100	72	10	82
$\tau 4$	200	**8**	18	**3**	2	200	181	10	191
$\tau 5$	400	2	12	**10**		400	386		386
									(ms)

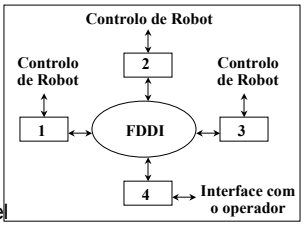
» pelo que o conjunto de tarefas é escalonável.

Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: rede de comunicação**
  - O tempo de ciclo do "*token*" foi fixado em 8ms, sendo o tempo de transmissão disponível para o conjunto dos nós de 7ms por ciclo de *"token";*
  - Em cada nó foram fixados os seguintes tempos de permanência do "*token*":
    - » 2,1ms para cada um dos nós 1, 2 e 3;
    - » 0,7ms para o nó 4.



Exemplo de Escalonamento em Computação Industrial



- **Análise temporal: rede de comunicação**
  - Localmente, a rede de comunicação pode ser encarada como um recurso escasso a escalonar entre o nó local e os outros nós:
    - » Em cada nó (1,2,3), a rede está inacessível durante (2,1+2,1+0,7+1)=5,9ms cada 8ms
    - » Em cada nó (1,2,3), $\tau 2$ gera 1Mbit de dados (a 100 Mbit/s corresponde a 10ms) cada 50ms;
  - Análise do tempo de resposta:

	C_i(ms)	T_i(ms)	U_i	R_i
$\tau 1$	5,9	8	0,737	5,9
$\tau 2$	10	50	0,200	39,5
				(ms)

$$w_2^0 = 10 + 1(5,9) = 15,9ms$$
$$w_2^1 = 10 + 2(5,9) = 21,8ms$$
$$w_2^2 = 10 + 3(5,9) = 27,7ms$$
$$w_2^3 = 10 + 4(5,9) = 33,6ms$$
$$w_2^4 = 10 + 5(5,9) = 39,5ms$$
$$w_2^5 = 10 + 5(5,9) = 39,5ms$$

» pelo que a transferência de dados para o nó 4 é escalonável. $\boxed{R_2 = 39,5ms}$

Exemplo de Escalonamento em Computação Industrial

- **Análise temporal: meta temporal de extremo-a-extremo**
  - Tempos máximos de execução da transacção:
    - » 36ms para a tarefa $\tau 2$ no nó 1 terminar a sua execução . Esta tarefa é activada pela aquisição dos dados.
    - » 39,5ms para a rede transmitir 1Mbit de dados gerados pelo nó 1. No pior caso, haverá um ciclo inicial de "*token*" não aproveitado (já considerado na análise).
    - » 106ms para a tarefa $\tau 2$ do nó 4 processar os dados recebidos.
    - » A tarefa $\tau 2$ do nó 4 não está sincronizada com a chegada do "*token*", pelo que no pior caso haverá um atraso suplementar de 100ms (T_2=100ms).
  - Tempo de resposta extremo-a-extremo: R=(36+39,5+106+100)=281,5ms
    - » pelo que a transacção é também escalonável.