Cap. 9 - Uniprocessor Scheduling

9.1 - Tipos de Escalonamento

9.2 - Algoritmos de Escalonamento

9.3 - Performance dos Algoritmos

9.4 - Escalonamento no UNIX Tradicional

... Cap. 9 - Uniprocessor Scheduling

- ✯ William STALLINGS; **Operating Systems: Internals and Design Principles**, New Jerssey, Prentice-Hall, 1998, ISBN: 0-13-887407-7

- ✯ Eleri CARDOZO; Maurício MAGALHÃES; Luís F. FAINA; **Sistemas Operacionais**, Dep. de Eng. de Computação e Automação Industrial, Fac. de Engenharia Elétrica e de Computação, UNICAMP, 1996.

... Cap. 9 - Uniprocessor Scheduling

- Nos Sistemas Multiprogramados, múltiplos processos são mantidos em memória principal, cada qual alternando o uso do processador, esperando alguma operação de I/O ser completada ou algum outro evento ocorrer.

- Como fator chave da multiprogramação, 04 tipos de escalonamento são possíveis:

Long–term Scheduling	**The decision to add to the pool of processes to be executed.**
Medium–term Scheduling	**The decision to add to the number of processes that are partially or fully in main memory.**
Short–term Scheduling	**The decision as to which available process will be executed by the processor**
I/O Scheduling	**The decision as to which process's pending I/O request shall be handled by an available I/O device.**

... Cap. 9 - Uniprocessor Scheduling

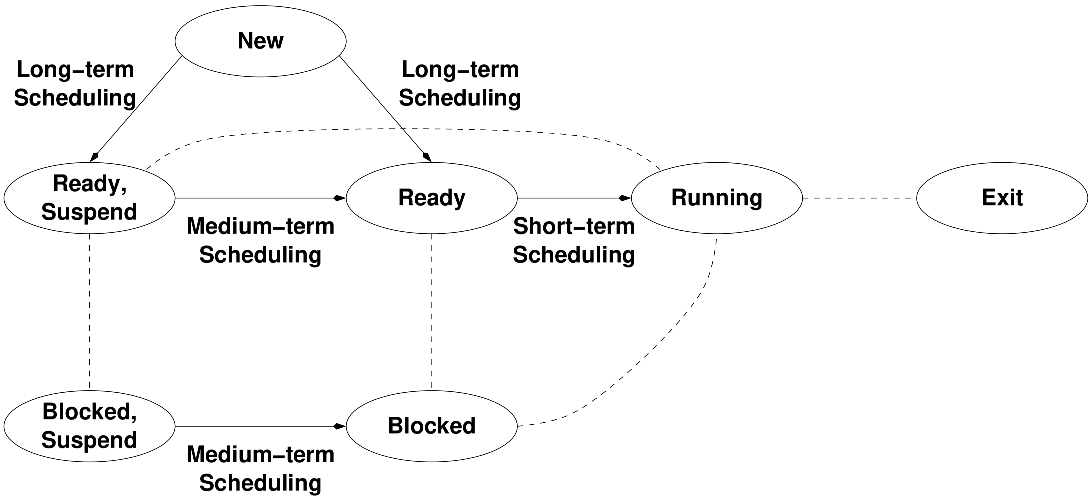
- **Long-term** e **Medium-term Scheduling** estão diretamente relacionados com aspectos de performance, ou seja, grau da multiprogramação.

- ... normalmente realizado quando um processo é criado ou é permutado da memória secundária para a primária, estando bloqueado ou não.

- **Short-term Scheduling** aborda com alto grau de refinamento o escalonamento dos processos que estão prontos em memória à espera do processador.

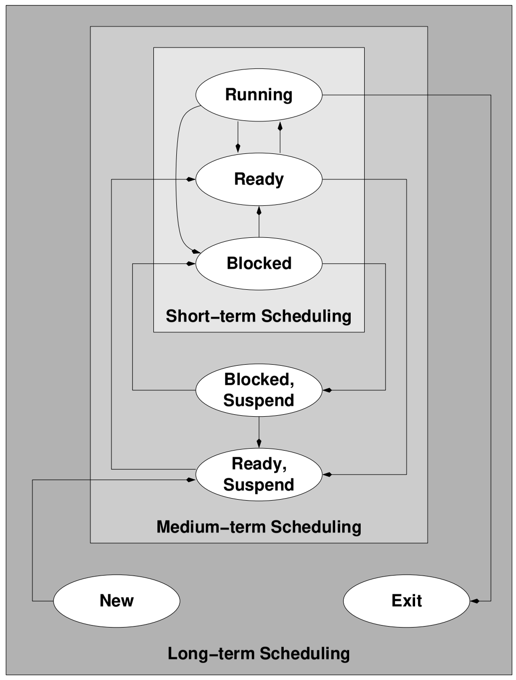
9.1 - Tipos de Escalonamento

✠ O objetivo do escalonador é atribuir processos para serem executados pelo processador(es) de modo a atingir parâmetros de performance do sistema tais como: tempo de resposta, vazão e eficiência do processador.



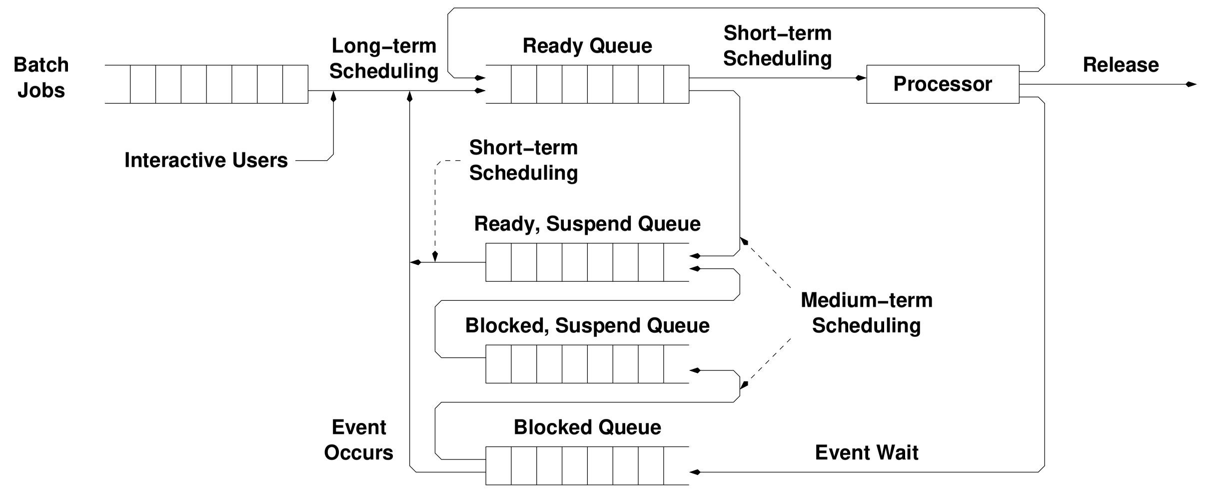
... 9.1 - Tipos de Escalonamento

✠ Reorganização do diagrama anterior com funções de escalonamento aninhadas:



... 9.1 - Tipos de Escalonamento

✡ O escalonamento afeta a performance do sistema pois determina quais processos deverão esperar e quais deverão progredir - gerenciamento de fila.



... 9.1 - Tipos de Escalonamento

✡ **Long-term Scheduling:** determina quais programas serão admitidos pelo sistema para processamento, ou seja, controla o grau da multiprogramação.

- ... em alguns sistemas, um processo que acabou de ser criado inicia-se na memória secundária e, neste caso, será adicionado à fila do escalonador intermediário;

- ... em sistema operacionais de propósito geral e de processamento em lote, processos recém criados são direcionados para o disco e mantidos numa fila de lote.

✡ A decisão de quando criar um novo processo é geralmente tomada como resultado do grau da multiprogramação, haja visto que quanto mais processos menor é o tempo tomado para cada um ser executado, pois mais processos competem pelos recursos;

- ... em termos de freqüência de execução, *long-term scheduler* desempenha o gerenciamento grosseiro e, portanto, é executado com baixa freqüência.

... 9.1 - Tipos de Escalonamento

✰ **Medium-term Scheduling:** responsável pela permuta (*swapping*) entre memória secundária e principal, com a decisão de *swapping-in* baseada no gerenciamento do grau da multiprogramação e nas parâmetros dos processos *swapped-out*;

- ... executado mais freqüentemente que o anterior para o gerenciamento do *swapping*.

... 9.1 - Tipos de Escalonamento

✰ **Short-term Scheduling:** também conhecido como *dispatcher* é executado muito freqüentemente e é assim responsável pelos pormenores do gerenciamento de qual processo deve ser o próximo a ser executado.

✰ Este escalonamento é invocado quando da ocorrência de um evento que conduza o bloqueio do corrente processo, criando-se uma oportunidade de preempção do corrente processo em favor de outro processo;

✰ Alguns desses eventos:

- *clock interrupts*;
- *I/O interrupts*;
- chamadas do sistema operacional;
- sinais

9.2 - Algoritmos de Escalonamento

✰ **Short-term Scheduling** procura alocar o tempo do processador de tal maneira a otimizar um ou mais aspectos do comportamento do sistema;

- ... os critérios freqüentemente utilizados podem ser distinguidos em: critérios orientados ao sistema e critérios orientados aos usuários.

✰ Enquanto os critérios orientados aos usuários são virtualmente importante na quase totalidade dos sistemas, os critérios orientados ao sistema geralmente tem menor importância nos sistema de um único usuário;

- ... em sistemas com um único usuário, muito provavelmente alcançar a máxima utilização do processador ou a máxima vazão não sejam os fatores mais importantes.

... 9.2 - Algoritmos de Escalonamento

✰ **critérios orientados para o usuário:** estão relacionados como os usuários ou os seus processos quantificam o comportamento do sistema.

User Oriented, Performance Related

Response Time	**For an interactive process, this is the time from the submission of a request until the response begins to be received. Often a process can begin producing some output to the user while continuing to process the request. Thus, this is a better measure than turnaround time from the user's pont of view. The scheduling discipline should attempt to achieve low response time and to maximize the number of interactive users receiving acceptable response time.**
Turnarount Time	**This is the interval of time between the submission of a process and its completion. Includes actual execution time plus time spent waiting for resources, including the processor. This is an appropriate measure for a batch job.**
Deadlines	**When process completion deadlines can be specified, the scheduling discipline should subordinate other goals to that of maximizing the percentage of deadlines met.**

User Oriented, Other

Predictability	**A given job should run in about the same amount of time and at about the same cost regardless of the load on the system. A wide variation in response time or turnaround time is distracting to users. It may signal a wide swing in system workloads or the need for system tuning to cure instabilities.**

... 9.2 - Algoritmos de Escalonamento

- **critérios orientados para o sistema:** relaciona-se com métricas que quantificam a utilização efetiva do processador, p.ex.: vazão de processos.

System Oriented, Performance Related

Throughput	**The scheduling policy should attempt to maximize the number of processes completed per unit of time. This is a measure of how much work is being performed. This clearly depends on the average length of a process but is also influenced by the scheduling policy, which may affect utilization.**
Processor Utilization	**This is the percentage of time that the processor is busy. For an expensive shared system, this is a significant criterion. In single–user systems and in some other systems, such as real–time systems, this criterion is less important than some of the others.**

System Oriented, Other

Fairness	**In absence of guidance from the user or other system–supplied guidance, processes should be treated the same, and no process should suffer starvation.**
Enforcing Priorities	**When processes are assigned priorities, the scheduling policy should favor higher priority processes.**
Balancing Resources	**The scheduling policy should keep the resources of the system busy. Processes that will underutilize stressed resources should be favored. This criterion also involves medium–term and long–term scheduling.**

... 9.2 - Algoritmos de Escalonamento

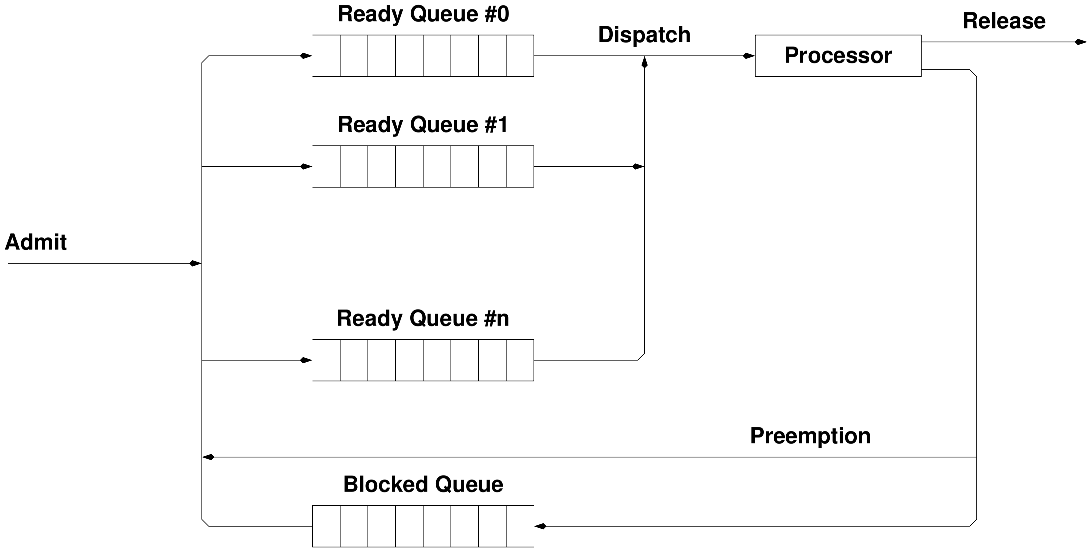
- Como pode ser observado, os critérios são independentes e, por isso, é impossível otimizá-los em conjunto e simultaneamente;

- ... p.ex.: prover bom tempo de resposta pode exigir um algoritmo de escalonamento que freqüentemente troque o contexto entre processos, o que aumente o *overhead* do sistema, ou seja, contribui para reduzir a vazão dos processos.

- Na maior parte dos sistemas operacionais interativos, ou seja, simples usuários e compartilhamento do tempo, o tempo de resposta é o critério chave.

9.2.1 - Algoritmos de Escalonamento com Prioridade

✰ Em muitos sistemas, a cada processo é atribuído uma prioridade e o escalonador irá sempre escolher o processo com mais alta prioridade sobre o de menor prioridade.



... 9.2.1 - Algoritmos de Escalonamento com Prioridade

✰ Um dos problemas do escalonamento com prioridades é que processos com baixa prioridade sofrem *starvation* em um sistema que disponibilize processos prontos para serem executados prioridade maior que a de seus pares;

- ... para resolver este problema, a prioridade do processo pode ser alterada com um função do tempo e história de sua execução, permitindo assim que outros tenham alguma chance não nula de executarem.

9.2.2 - Algoritmos de Escalonamento Alternativos

✰ Informações resumidas de várias políticas de escalonamento:

	FCFS	Round–Robin	SPN	SRT	HRRN	Feedback
Selection Function	max(w)	constant	min(s)	min(s – e)	max((w+s) / s)	(see text)
Decision Mode	Nonpreemptive	Preemptive (at time quantum)	Nonpreemptive	Preemptive (at arrival)	Nonpreemptive.	Preemptive (at quantum time)
Throughput	Not emphasized	May be low if quantum is too small.	High	High	High.	Not emphasized
Response Time	May be high, especially if there is a large variance in process execution times.	Provides good response time for short processes.	Provides good response time for short processes.	Provides good response time.	Provides good response time.	Not emphasized
Overhead	Minimum	Low	Can be high.	Can be high.	Can be high.	Can be high.
Effect on Processes	Penalizes short processes; Penalizes I/O bound processes	Fair treatment	Penalizes long processes.	Penalizes long processes.	Good balance.	May favor I/O bound processes
Starvation	No	No	Possible	Possible.	No	Possible.

... 9.2.2 - Algoritmos de Escalonamento Alternativos

✰ **selection function:** determina qual processo, dentre os processos prontos para serem executados que será o próximo a ser selecionado.

✰ Esta função pode ser baseada em alguns parâmetros do processo, como por exemplo:

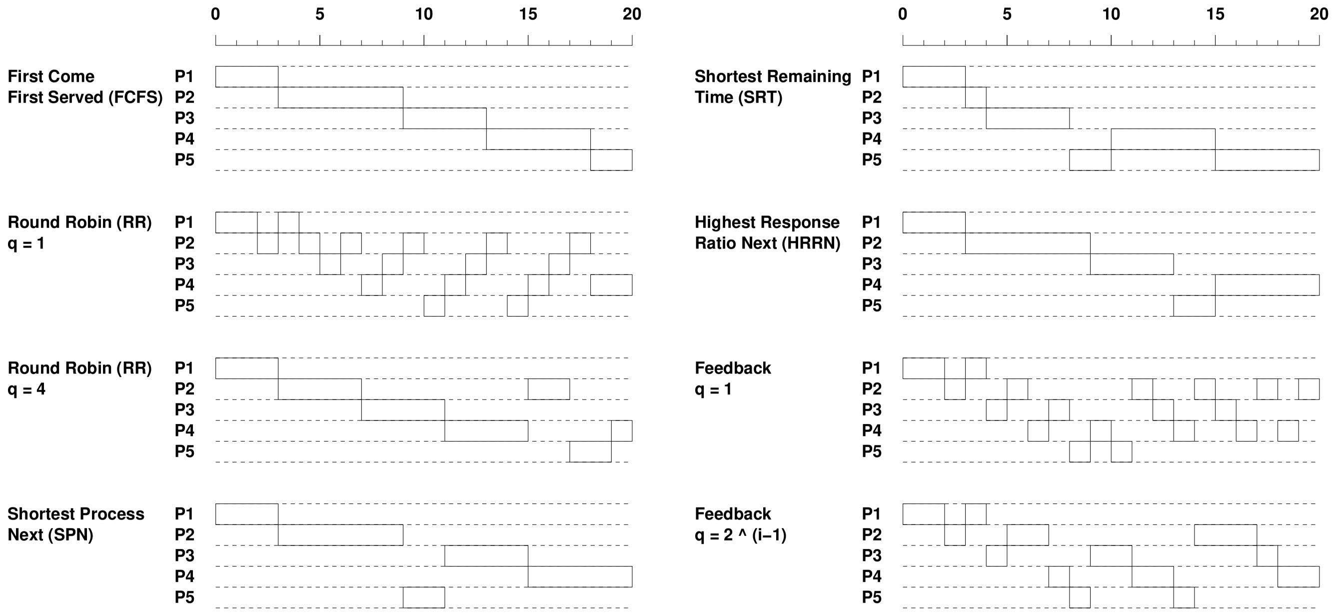
- w - tempo gasto no sistema esperando ou executando;
- e - tempo gasto em execução no sistema;
- s - tempo exigido pelo processo para começar e finalizar sua execução.

✰ **modo de decisão:** determina os instantes de tempo em que a função de seleção é exercida, ou seja, 02 categorias são importantes:

- **não preemptivo:** o processo não poderá ser interrompido por um terceiro enquanto não terminar ou executar um operação que o leve ao estado de bloqueado;
- **preemptivo:** processo pode ser interrompido e conduzido ao estado de pronto para ser executado pelo sistema operacional.

... 9.2.2 - Algoritmos de Escalonamento Alternativos

✰ Padrões de execução de algumas políticas de escalonamento para os processos P1, P2, P3, P4 e P5 com $t_{arrival}$ de 0, 2, 4, 6 e 8 e $t_{service}$ de 3, 6, 4, 5 e 2:



... 9.2.2 - Algoritmos de Escalonamento Alternativos

	Process	**1**	**2**	**3**	**4**	**5**	
	Arrival Time	**0**	**2**	**4**	**6**	**8**	**Mean**
	Service Time (Ts)	**3**	**6**	**4**	**5**	**2**	
FCFS	**Finish Time**	**3**	**9**	**13**	**18**	**20**	
	Turnaround Time (Tq)	**3**	**7**	**9**	**12**	**12**	**8.60**
	Tq / Ts	**1.00**	**1.17**	**2.25**	**2.40**	**6.00**	**2.56**
RR q=1	**Finish Time**	**4**	**18**	**17**	**20**	**15**	
	Turnaround Time (Tq)	**4**	**16**	**13**	**14**	**7**	**10.80**
	Tq / Ts	**1.33**	**2.67**	**3.25**	**2.80**	**3.50**	**2.71**
RR q=4	**Finish Time**	**3**	**17**	**11**	**20**	**19**	
	Turnaround Time (Tq)	**3**	**15**	**7**	**14**	**11**	**10.00**
	Tq / Ts	**1.00**	**2.5**	**1.75**	**2.80**	**5.50**	**2.71**
SPN	**Finish Time**	**3**	**9**	**15**	**20**	**11**	
	Turnaround Time (Tq)	**3**	**7**	**11**	**14**	**3**	**7.60**
	Tq / Ts	**1.00**	**1.17**	**2.75**	**2.80**	**1.50**	**1.84**
SRT	**Finish Time**	**3**	**15**	**8**	**20**	**10**	
	Turnaround Time (Tq)	**3**	**13**	**4**	**14**	**2**	**7.20**
	Tq / Ts	**1.00**	**2.17**	**1.00**	**2.80**	**1.00**	**1.59**
HRRN	**Finish Time**	**3**	**9**	**13**	**20**	**15**	
	Turnaround Time (Tq)	**3**	**7**	**9**	**14**	**7**	**8.00**
	Tq / Ts	**1.00**	**1.17**	**2.25**	**2.80**	**3.5**	**2.14**
FB q=1	**Finish Time**	**4**	**20**	**16**	**19**	**11**	
	Turnaround Time (Tq)	**4**	**18**	**12**	**13**	**3**	**10.00**
	Tq / Ts	**1.33**	**3.00**	**3.00**	**2.60**	**1.50**	**2.29**
FB q=2^(i−1)	**Finish Time**	**4**	**17**	**18**	**20**	**14**	
	Turnaround Time (Tq)	**4**	**15**	**14**	**14**	**6**	**10.60**
	Tq / Ts	**1.33**	**2.50**	**3.5**	**2.80**	**3.00**	**2.63**

9.2.3 - First Come First Served

✡ A medida que cada processo se torna pronto para executar, o mesmo é inserido na fila e quando o processo corrente finaliza sua execução, o processo mais velho na fila de pronto para executar é escalonado para utilizar o processador.

- Como pode ser observado, FCFS funciona melhor para processos longos quando comparado a processos pequenos, por exemplo:

Process	Arrival Time	Service Time (Ts)	Start Time	Finish Time	Turnaround Time (Tq)	Tq / Ts
A	0	1	0	1	1	1
B	1	100	1	101	100	1
C	2	1	101	102	100	100
D	3	100	102	202	199	1.99
Mean					100	26

... 9.2.3 - First Come First Served

✡ Outra dificuldade do FCFS é a tendência em favorece processos com altas exigências computacionais e uso ocasional de entrada/saída (**bound-processor**);

- ... considere o cenário onde um processo exige muito processador (**bound-processor**) e os demais exigem muita entrada/saída (**I/O bound**) !?

- ... FCFS pode resultar no uso ineficiente de ambos os processadores e dispositivos de entrada/saída quandos ambos conjuntos de processos estão bloqueados.

✡ FCFS não é atrativo para Sistema Monoprocessados, não obstante, pode ser combinado com Algoritmo de Prioridades tornando-se um escalonador efetivo.

9.2.4 - Algoritmo Round Robin

✰ Uma alternativa direta para reduzir a penalidade que processos pequenos sofrem no FCFS é a utilização da preempção baseada no relógio (*clock*) – **Round Robin**.

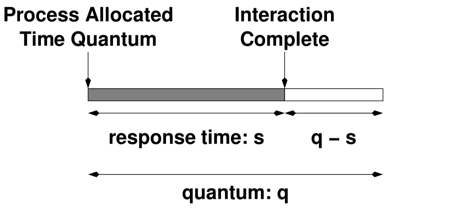
- ... quando a interrupção ocorre, o processo correntemente sendo executado é colocado na fila de pronto para executado e seleciona-se o processo mais velho da mesma fila, ou seja, seleção segundo a política FCFS;
- ... esta técnica é conhecida como *time slicing*, pois cada processo recebe uma fatia de tempo antes de ser preemptado pelo escalonador.

... 9.2.4 - Algoritmo Round Robin

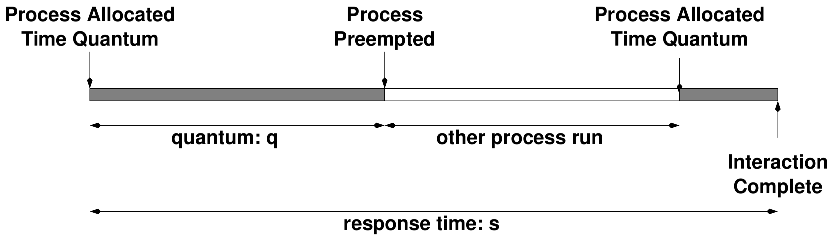
✰ Principal aspecto de projeto do Round Robin é o tamanho desta fatia de tempo (*slice* ou *time quantum*), posto que, se muito pequeno então processos pequenos irão passar pelo sistema rapidamente as custas de um *overhead* grande;

- ... mas se muito grande, podemos cair na política FCFS pura e simplesmente.

✰ Efeito no *quantum* de preempção no tempo de resposta:



Time Quantum Greater than Typical Interaction



Time Quantum Less than Typical Interaction

... 9.2.4 - Algoritmo Round Robin

✰ Round Robin é particularmente efetivo nos Sistemas *Time-Sharing* de propósito geral ou em sistemas de processamento de transações, entretanto tem com desvantagens:

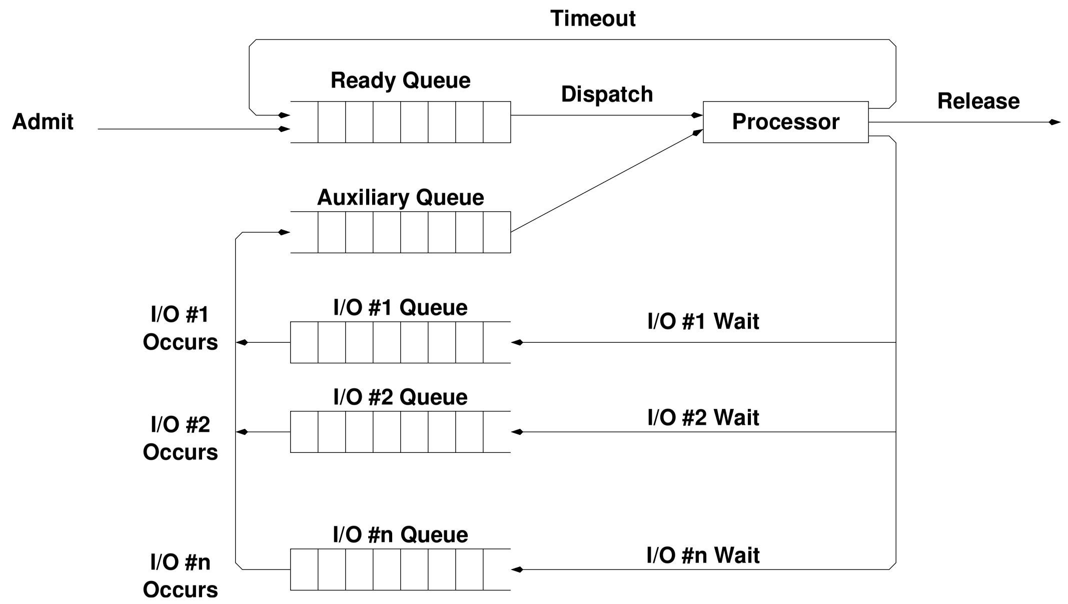
- ... tratamento relativo para processos limitados pelo processador e limitados por operações de entrada/saída, posto que os últimos apresentam tempos de execução (*processor burst*) menores que os primeiros;

- ... processos limitados pelo processador tendem a receber menos tempo para executarem que os processos limitados por operações de entrada/saída resultando:

- ... baixa performance dos processos limitados por operações de entrada/saída, uso ineficiente dos dispositivos e variação no tempo de resposta.

... 9.2.4 - Algoritmo Round Robin

✰ Diagrama de Fila para o Escalonador Round Robin Virtual que tem por objetivo evitar o tratamento desigual dado as duas classes de processos.



9.2.5 - Algoritmo Shortest Process Next

✰ Outra abordagem é reduzir a diferença em favor dos processos longos herdada na Política FCFS — Política **Shortest Process First** ou **SPN**;

- ... política não preemptiva na qual o processo com o menor tempo de execução é o próximo a ser selecionado, ou seja, vão para a cabeça da lista.

✰ Uma das dificuldades do SPN é a necessidade de conhecer ou ao menos estimar o tempo de execução, o que pode exigir trabalho extra do sistema:

$$S_{n+1} = \frac{1}{n}\sum_{i=1}^{n} T_i$$

- onde T_i é o tempo de execução para a i-ésima instância deste processo; S_i é o valor predito para a i-ésima instância e S_1 é o valor predito para a primeira instância.

... 9.2.5 - Algoritmo Shortest Process Next

✰ Para evitar recálculos, a somatório pode ser reescrita como:

$$S_{n+1} = \frac{1}{n}T_n + \frac{n-1}{n}S_n$$

✰ Para ponderar instâncias recentes em detrimento das antigas, uma técnica comum para predição do valor futuro é a da média exponencial:

$$S_{n+1} = \alpha T_n + (1-\alpha)S_n$$

- com $0 < \alpha < 1$ determinando o peso relativo das amostras.

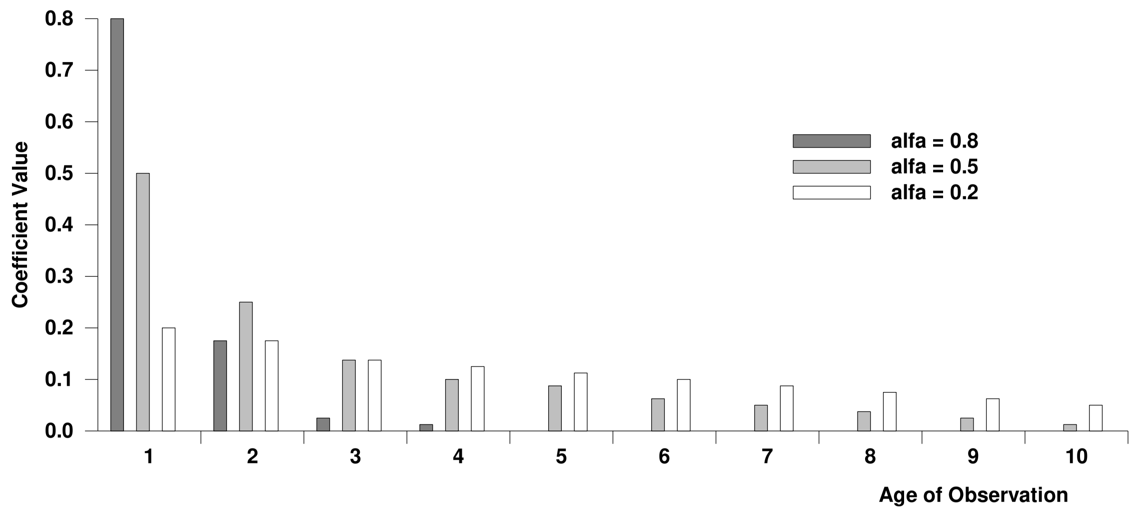
✰ Para ver mais claramente, considere a expansão da equação anterior:

$$S_{n+1} = \alpha T_n + (1-\alpha)\alpha T_{n-1} + ... + (1-\alpha)^i \alpha T_{n-i} + ... + (1-\alpha)^n S_1$$

... 9.2.5 - Algoritmo Shortest Process Next

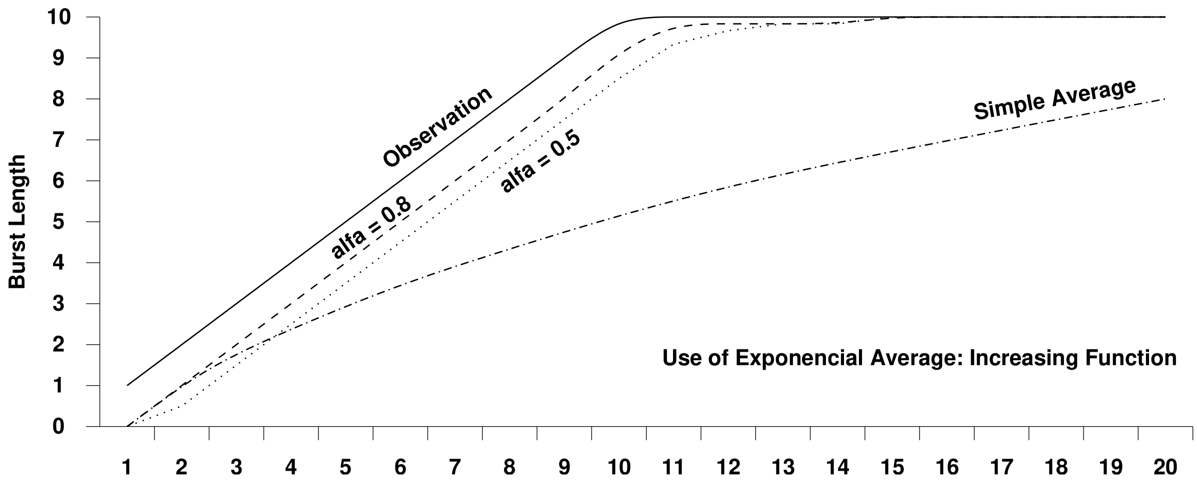
✰ Para $\alpha = 0.8$, obtém-se:

$$S_{n+1} = 0.8T_n + 0.16T_{n-1} + 0.032T_{n-2} + 0.00064T_{n-3} + ...$$



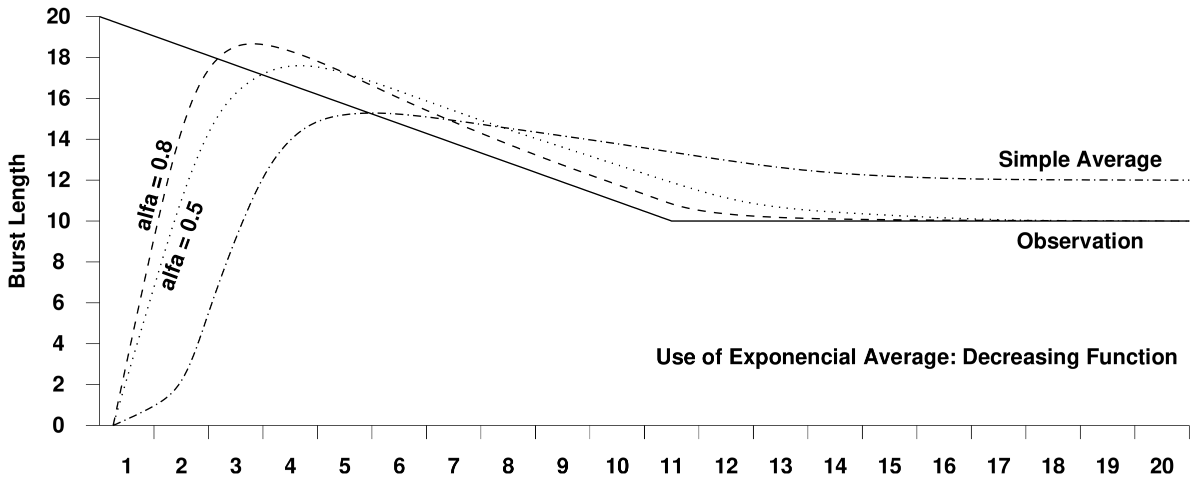
... 9.2.5 - Algoritmo Shortest Process Next

✰ Comparação entre média simples e média exponencial com valores começando em 1, crescendo gradualmente até chegar a 10 onde estacionam.



... 9.2.5 - Algoritmo Shortest Process Next

✰ Comparação entre média simples e média exponencial com valores começando em 20, decrescendo gradualmente até chegar a 10 onde estacionam.



... 9.2.5 - Algoritmo Shortest Process Next

✰ Um risco com SPN é a possibilidade de *starvation* com processos longos, posto que tenhamos uma fonte constante de processos menores no sistema;

- ... embora SPN reduza a diferença em favor dos processos longos, este escalonador não é ainda objeto de busca em sistemas *time-sharing* ou em sistemas de processamento de transações dada a falta de preempção;

- ... veja o pior caso na análise descrita no FCFS para os processos A, B, C e D ainda assim executará na mesma ordem, resultando grande penalidade ao processo C.

9.2.6 - Algoritmo Shortest Remaining Time

✰ **Shortest Remaining Time:** versão preemptiva do SPN na qual o escalonador sempre escolhe o processo que tem o menor tempo remanescente de execução;

- ... quando um novo processo se junta a fila de pronto para executar, ele na verdade tem o menor tempo remanescente de execução que o processo corrente;

- ... assim como no SPN, o escalonador precisa de estimativas quanto ao tempo de execução dos processos para fazer a melhor escolha bem como corre o risco de *starvation* para processos longos.

✰ SRT oferece tempos de *turnaround* melhores que o SPN, dado que pequenas tarefas tem preferência na execução diante de tarefas mais longas.

9.2.7 - Algoritmo Highest Response Ratio Next

✰ Quando um processo completa ou é bloqueado, escolha um processo da lista de prontos para executar com o maior valor de RR (*Response Ratio*):

$$RR = \frac{w + s}{s}$$

- onde w é o tempo gasto esperando pelo processador e s é o tempo de serviço.

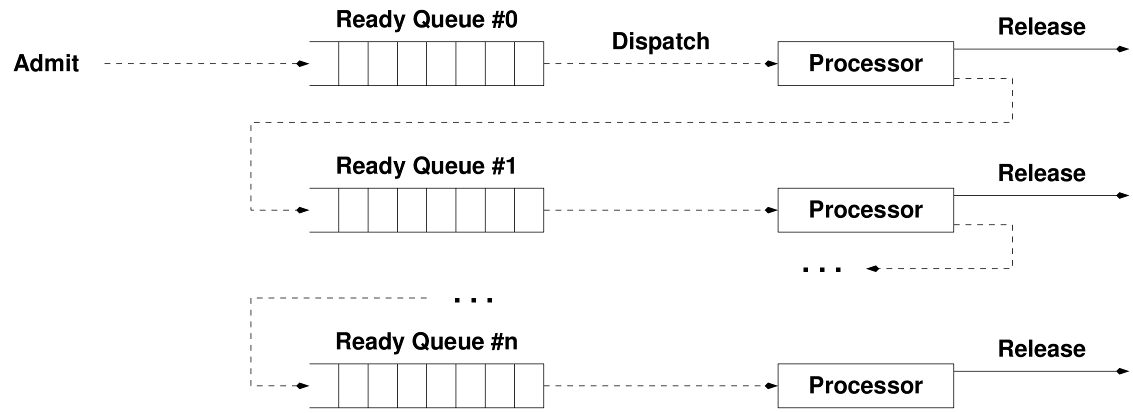
✰ Esta abordagem é atrativa pois procura minimizar o tempo de *turnaround* normalizado de todo o sistema e não somente de um processo em particular;

- ... enquanto processos menores são favorecidos, envelhecer sem ser atendido aumenta esta razão, assim, processos maiores irão eventualmente competir como os menores.

9.2.8 - Algoritmo Feedback

✰ Caso não tenhamos informações acerca do tamanho dos processos, nenhum dos Algoritmos SPN, SRT e HRRN poderão ser utilizados, então uma alternativa é penalizar processos que já executaram por mais tempo;

- … para tanto, o escalonamento é feito tendo por base a preempção e, adicionalmente, com o mecanismo de atribuição de prioridades dinâmicas:



… 9.2.8 - Algoritmo Feedback

✰ Esta abordagem dá margem para um grande número de variações, sendo que na versão mais simples realiza-se a preempção em intervalos periódicos – Round Robin;

- … na simulação realizada, apresentamos este algoritmo com *quantum* de 1 no primeiro case e *quantum* variável (2^{i-1}) no segundo caso.

✰ Para processos longos, o tempo de *turnaround* pode se tornar muito grande, de fato, *starvation* pode acontecer se tivermos entrada regular de processos no sistema.

9.3 - Performance dos Algoritmos

✰ **performance dos algoritmos:** constitui-se um fator crítico na escolha de uma política de escalonamento, entretanto, é difícil estabelecer um modelo de comparação posto que performances diferentes dependem de uma grande variedade de fatores:

- distribuição de probabilidade dos tempos de serviço;
- eficiência do escalonamento e mecanismo de chaveamento de contexto;
- natureza da demanda e performance do sub-sistema de entrada/saída.

✰ Toda disciplina de escalonamento que escolhe o próximo elemento a ser servido independentemente do tempo de serviço segue a seguinte relação:

$$\frac{T_q}{T_s} = \frac{1}{1-\rho}$$

- onde T_q é o tempo de *turnaround*, T_s é o tempo médio gasto pelo processo executando e ρ é a utilização do processador.

... 9.3 - Performance dos Algoritmos

✰ Com base na Teoria de Filas, as seguintes premissas devem ser satisfeitas para o equacionamento de Filas de um Único Servidor e 02 Classes de Prioridades:

- taxa de chegada de processos segundo Poisson;
- elementos com prioridade 1 são atendidos antes daqueles com prioridade 2;
- para elementos com mesma prioridade, política FIFO de atendimento;
- nenhum elemento é interrompido enquanto está sendo atendido;
- nenhum elemento deixa a fila por chamada perdida atrasada.

✰ Fórmulas Gerais para Filas de um Único Servidor e 02 Classes de Prioridades:

$$\lambda = \lambda_1 + \lambda_2; \qquad \rho_1 = \lambda_1 T_{s1}; \qquad \rho_2 = \lambda_2 T_{s2}; \qquad \rho = \rho_1 + \rho_2;$$

$$T_s = \frac{\lambda_1}{\lambda} T_{s1} + \frac{\lambda_2}{\lambda} T_{s2} \qquad T_q = \frac{\lambda_1}{\lambda} T_{q1} + \frac{\lambda_2}{\lambda} T_{q2}$$

... 9.3 - Performance dos Algoritmos

✰ Tempo de Serviço Exponencial sem interrupções:

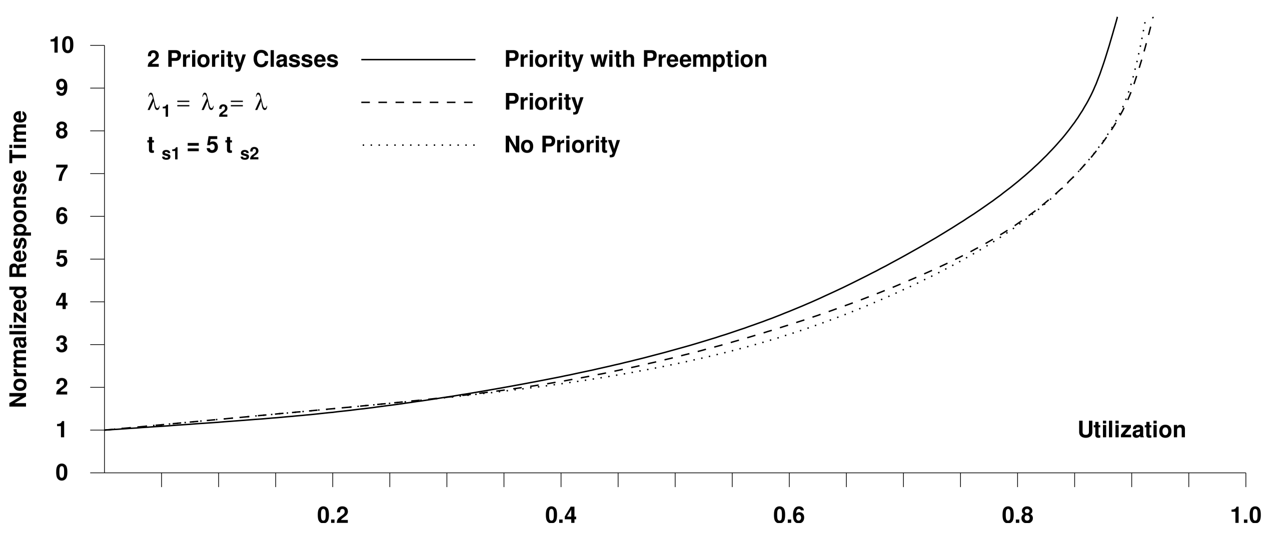
$$T_{q1} = T_{s1} + \frac{\rho_1 T_{s1} + \rho_2 T_{s2}}{1 - \rho_1} \qquad T_{q2} = T_{s2} + \frac{T_{q1} - T_{s1}}{1 - \rho}$$

✰ Tempo de Serviço Exponencial com disciplina de fila *preemptive-resume*:

$$T_{q1} = 1 + \frac{\rho_1 T_{s1}}{1 - \rho_1} \qquad T_{q2} = 1 + \frac{1}{1 - \rho}\left(\rho_1 T_{s1} + \frac{\rho T_s}{1 - \rho}\right)$$

... 9.3 - Performance dos Algoritmos

✰ Considere 02 classes de prioridade com igual nro. de chegada de processos e com tempo médio de serviço da classe mais baixa igual a 5 vezes o tempo médio de serviço da classe mais alta.



9.4 - Escalonamento no UNIX Tradicional

✰ Utilizado no SVR3 e 4.3 BSD UNIX o escalonamento no UNIX Tradicional oferece bom tempo de resposta para usuários interativos, enquanto garante que processos em *background* com baixa prioridade sofram *starvation*;

- ... embora este algoritmo tenha sido substituído nos sistemas modernos, é interessante avaliarmos esta abordagem por ser representativa do ponto de vista prático como algoritmo de escalonamento de sistemas *time-sharing*.

✰ Escalonador UNIX emprega *feedback* multinível utilizando o Round Robin dentro de cada uma das filas de prioridade com *quantum* de 1s de preempção.

... 9.4 - Escalonamento no UNIX Tradicional

✰ Prioridade é baseada no tipo do processo e na sua história de execução:

$$P_j(i) = Base_i + \frac{CPU_i(i-1)}{2} + nice_i$$

$$CPU_j(i) = \frac{U_j(i)}{2} + \frac{CPU_j(i-1)}{2}$$

- onde $P_j(i)$ é a prioridade do processo j no início do intervalo i, sendo que valores menores indicam prioridade maior; $Base_i$ é a prioridade base do processo j;
- $U_j(i)$ representa a utilização do processador pelo processo j no intervalo i;
- $CPU_j(i)$ representa a média exponencial ponderada de utilização do processador pelo processo j no intervalo i; e $nice_j$ é um fator de ajuste controlável pelo usuário.

... 9.4 - Escalonamento no UNIX Tradicional

- Ao subdividir os processos em bandas, otimiza-se o acesso aos dispositivos de bloco permitindo que o sistema operacional responda rapidamente às chamadas de sistema;

- ... em ordem decrescente de prioridade, as possíveis bandas são: *swapper*; controle de dispositivo de bloco de entrada/saída; manipulação de arquivo; controle de dispositivo de caracter de entrada/saída e processos do usuário.

- Na mesma banda de processos do usuário, a utilização da história de execução favorece a penalização de processos com alta carga computacional (*processor bound*) em comparação com os processos que realizam mais entrada/saída (*I/O bound*).

... 9.4 - Escalonamento no UNIX Tradicional

- Processos A, B e C criados no mesmo instante com $Base_i = 60$, freqüência de interrupções de 60 Hz e incrementos de contador por processo que está executando.

	Process A		Process B		Process C	
Time	**Priority**	**CPU Count**	**Priority**	**CPU Count**	**Priority**	**CPU Count**
0	60	0 1 ... 60	60	0	60	0
1	75	30	60	0 1 ... 60	60	0
2	67	15	75	30	75	0 1 ... 60
3	63	7 8 ... 67	67	15	67	30
4	76	33	63	7 8 ... 67	67	15
5	68	16	76	33	63	7