EP1 - MAC0422

Nome: Renan Ryu Kajihara
NUSP: 14605762

O código do simulador de processos é determinístico?

Não, apesar de não possuir grandes diferenças de resultado, o código do simulador de processos não é determinístico. Isso acontece porque:

- há a utilização da função “gettimeofday” para a simulação do tempo decorrido no escalonador de processos, que não é precisa de forma absoluta.

- a utilização de múltiplas threads gera diferentes resultados dependendo de qual thread “olha” para a fila de prontos primeiro, especialmente no escalonador com prioridade e no SRTN.

Algoritmo de escalonamento com prioridade

Definiu-se como 0,01 segundo o quantum desse escalonador.

O algoritmo de prioridade define quantos quantums o processo poderá utilizar. Dessa forma, ele funciona da seguinte forma:

1) calcula o tempo restante até a deadline e o tempo que pode ser desperdiçado pelo algoritmo.

```
double tempo_ate_deadline = p.deadline - tempo_atual;
double tempo_pode_desperdicar = tempo_ate_deadline - p.tempo_restante;
```

2) se o tempo que pode ser desperdiçado for menor ou igual a 0, isto é, se o processo não consegue ser concluído a tempo até a sua deadline, será atribuído apenas 1 quantum a ele.

```
if (tempo_pode_desperdicar <= 0) prioridade=1;
```

3) senão, a prioridade (número de quantums que o processo poderá utilizar) é calculado de forma que, se o tempo que o processo pode desperdiçar for menor ou igual a 2 segundos, o processo irá rodar até o seu fim. Caso contrário, o processo irá receber uma quantidade de quantums proporcional ao tempo que ele pode desperdiçar e o tempo restante.

```
double prioridade_real = p.tempo_restante / fmax(0.01, tempo_pode_desperdicar/200);
prioridade = (int)(fmax(1.0, prioridade_real) + 0.5);
```

EXPERIMENTOS

Arquivos de entrada dos experimentos

Para os experimentos, foram utilizados os seguintes arquivos trace:

- entrada-esperado.txt: possui 30 processos. Os 8 primeiros processos que chegam no instante 0 possuem 10 segundos de duração e deadlines muito altas. Os demais processos, que chegam no instante 1, possuem 1 ou 2 segundos de duração e deadlines que variam entre 3 e 60.

- entrada-inesperado.txt: possui 32 processos. Os 8 primeiros processos que chegam no instante 0 possuem 8 segundos de duração e deadlines pouco apertadas. Os demais processos, que chegam no instante 1 ou 2 ou 3, possuem 1 ou 2 ou 3 segundos de duração e deadlines que variam entre 10 e 15 segundos.

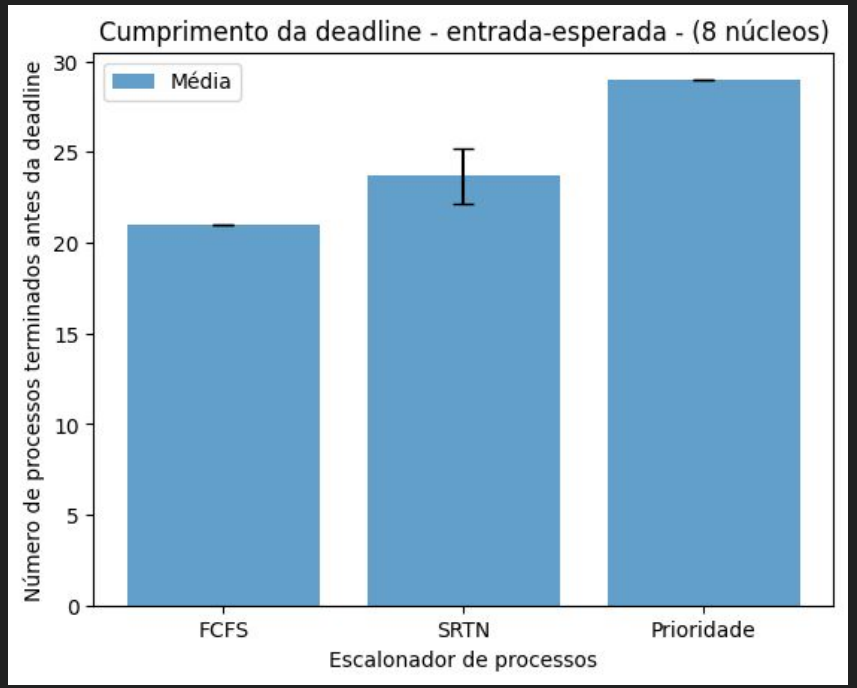
Máquinas de teste

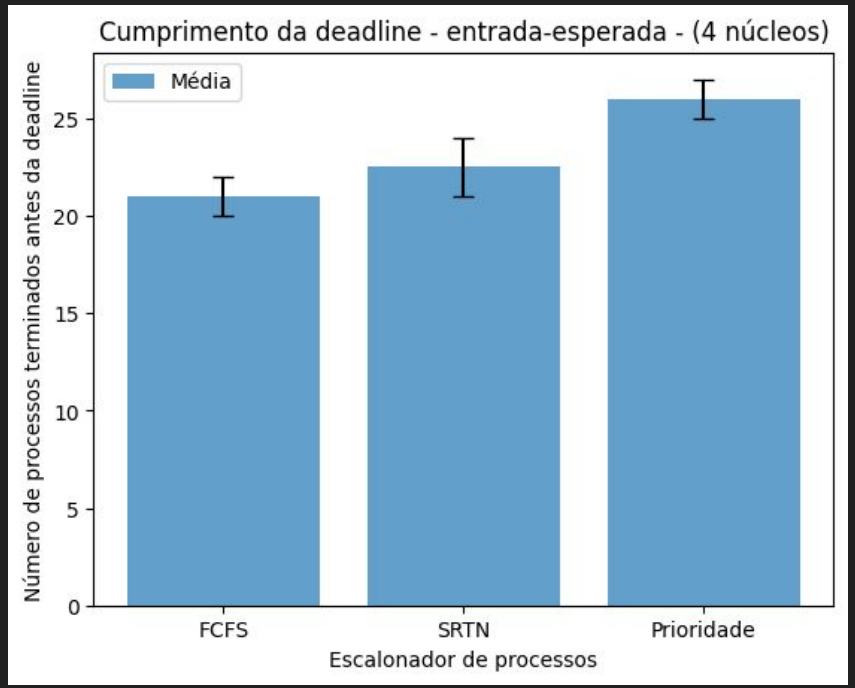
Os escalonadores de processos foram testadas em duas máquinas diferentes, com as seguintes especificações:

- Máquina A: 8 núcleos de processamento, processador Intel Core i7, com Sistema Operacional Ubuntu na versão 22.04.

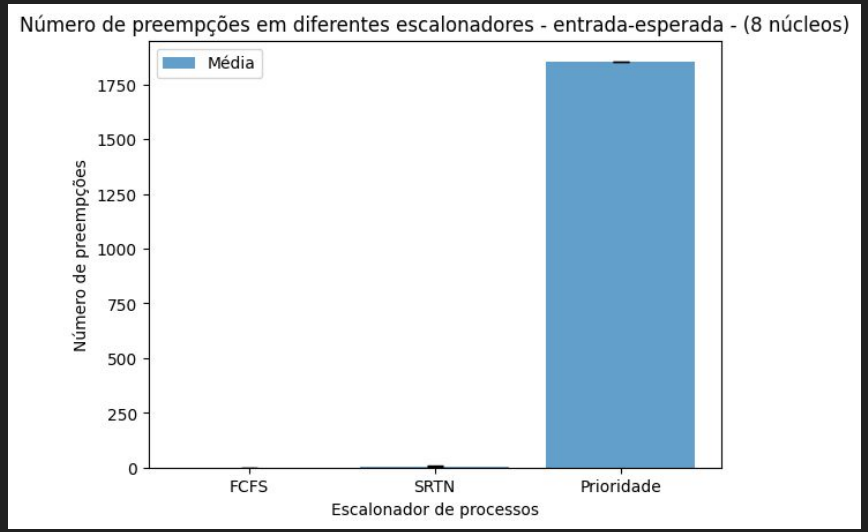
- Máquina B: 4 núcleos de processamento, processador Intel Core i5, com Sistema Operacional Ubuntu na versão 20.04.4 LTS

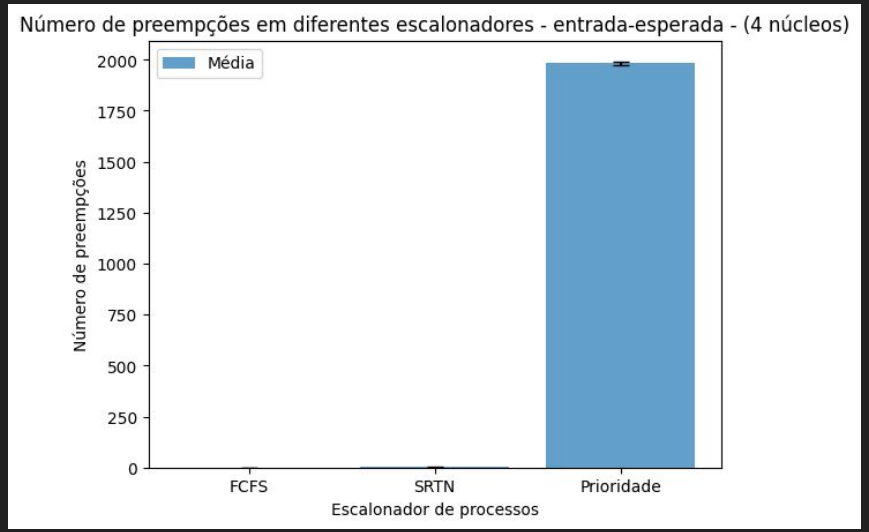
Cumprimento da deadline (entrada-esperada)



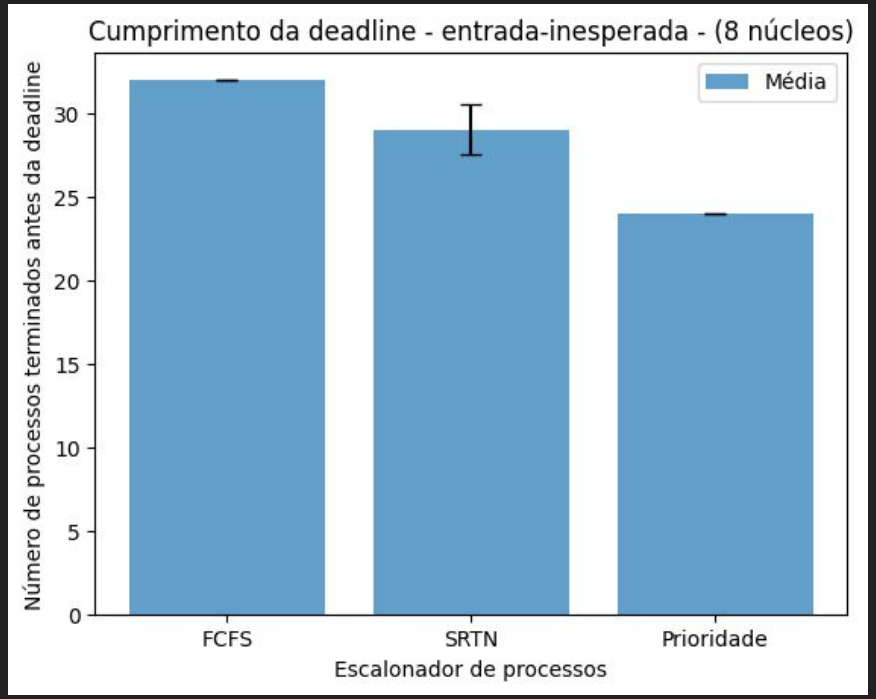


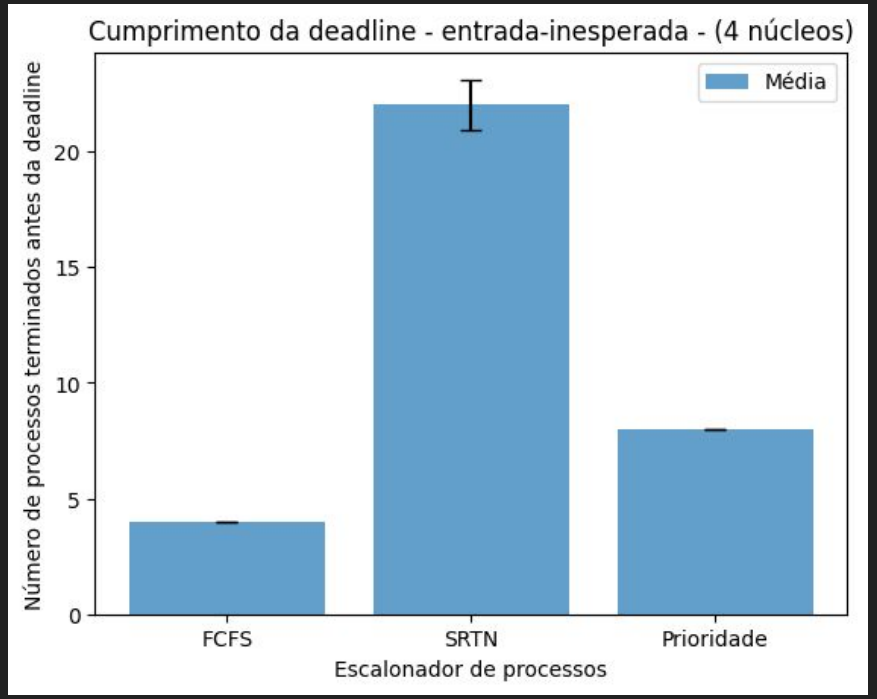
Número de preempções (entrada-esperada)



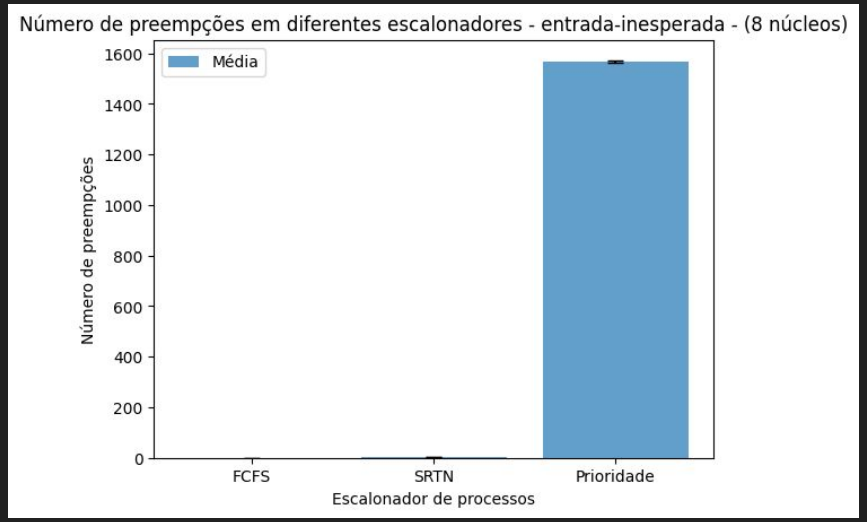


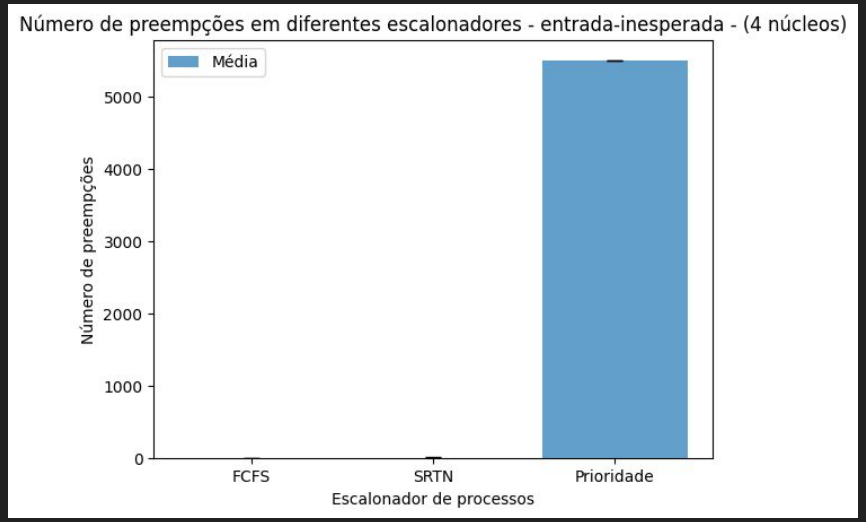
Cumprimento da deadline (entrada-inesperada)





Número de preempções (entrada-inesperada)





Análise dos experimentos

Para o arquivo trace entrada-esperado.txt:

- o FCFS gasta muito tempo com os primeiros processos que chegam que são muito grandes e não consegue cumprir os processos que chegam no instante 1 que são curtos e possuem deadlines apertadas.
- o SRTN faz poucas preempções e não consegue definir qual processo que chega no instante 1 possui maior prioridade, pois não “olha” para a deadline dos processos.
- o escalonador com prioridade funciona perfeitamente, cumprindo todos os processos, dando prioridade aos processos que possuem deadlines apertadas e fazendo um grande número de preempções.

Análise dos experimentos

Para o arquivo trace entrada-inesperado.txt:

- o FCFS gasta tempo suficiente para os processos grandes e pequenos cumprindo todas as deadlines.
- o SRTN faz poucas preempções conseguindo cumprir a deadline dos processos pequenos, mas negligencia os processos grandes que chegam no instante 0, conseguindo cumprir apenas alguns deles.
- o escalonador com prioridade faz muitas preempções atrasando o tempo de execução de alguns processos. Além disso, ele dá pouca prioridade aos processos grandes que chegam no instante 0 e acaba não conseguindo finalizá-los no final antes da deadline.

Observação: foi difícil encontrar arquivos trace que dessem os resultados inesperados de acordo com as especificações do enunciado do EP. Dessa forma, o arquivo trace escolhido funciona perfeitamente para computadores com 8 núcleos de processamento, mas possui resultados diferente quando testadas com computadores com mais ou menos núcleos de processamento. Assim, a análise feita anteriormente diz respeito aos testes no computador com 8 núcleos de processamento.