

# Simulador de processos

---

Gabriel Capella - 8962078 e Luís Felipe de Melo - 9297961

September 11, 2016

Centre for Modern Beamer Themes

## Considerações sobre o escalonador

---

- Um processo só pode ser criado a partir de outro.
- O escalonador deve ter um processo em sua fila antes de iniciar.
- Se não há nenhum processo a ser executado, desligamos todas as CPUs.

## Como implementamos

Os três tipos de escalonadores devem apresentar as seguintes funções:

- `P_INIT`: função chamada para ligar o nosso escalonador. Ela verifica quantas CPUs há no computador do usuário e inicia uma thread para cada uma. Finaliza somente quando não há mais processos a serem executados.
- `P_EXEC`: recebe as características de um novo processo (nome, linha do trace, tempo de execução, ponteiro para função e argumentos da função). Adiciona esse processo ao escalonador.
- `P_RUN`: chamada dentro de um processo, verifica se ele deve continuar sua execução ou parar.

## Processo load\_process (-1)

Antes do nosso escalonador iniciar, adicionamos esse processo a ele. Esse processo tem a função criar os nossos outros processos. Ou seja, ele verifica se já está na hora de adicionar outros processos no escalonador baseado no arquivo de trace.

Quando existem processos pedentes a serem criados, ele espera 0,05s e depois adiciona a si mesmo no escalonador.

## Exemplo

Vamos executar esse exemplo com os 4 algoritmos e pintar cada processo com uma cor.

0 1 processo1 20

0 2 processo2 20

0 3 processo3 20

0 4 processo4 20

1 5 processo1 20

1 6 processo2 20

1 7 processo3 20

1 8 processo4 20

1.5 processo9 1 20

1.5 processo10 2 20

1.5 processo11 3 20

1.5 processo12 4 20

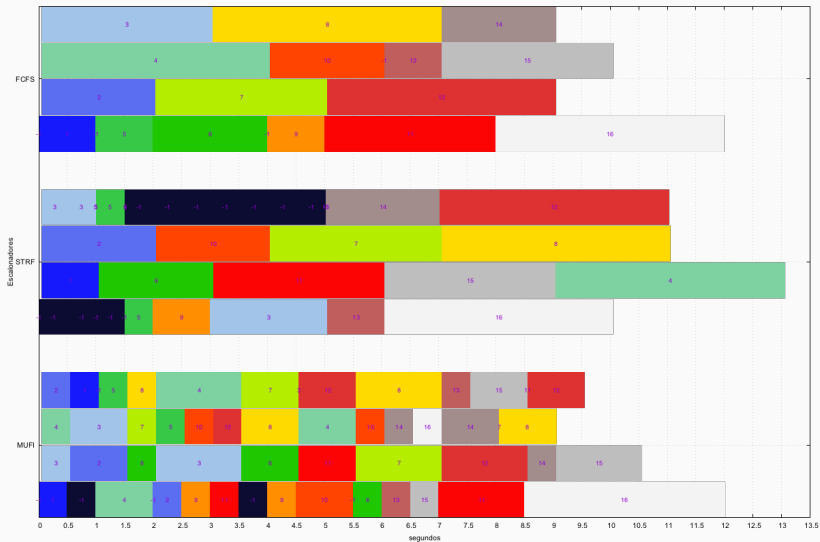
5 13 processo1 20

5 14 processo2 20

5 15 processo3 20

5 16 processo4 20

# Exemplo



\*Nesse exemplo o load\_process está com intervalos de 0,5s.

# Testes

---











Escolhemos um número de processos  $n$  e escolhemos um valor  $x$  entre 0 e 10, uniformemente. Depois escolhemos  $y$  entre  $x$  e 10.  $x$  é o momento em que o processo inicia e  $y - x$  é a duração do processo. Considere as unidades em segundos.

Veja que  $\bar{x} = 5.0$  e  $\bar{y} = 7.5$  e portanto o tempo médio dos nossos processos é de 2.5s.

# Tempo Gasto

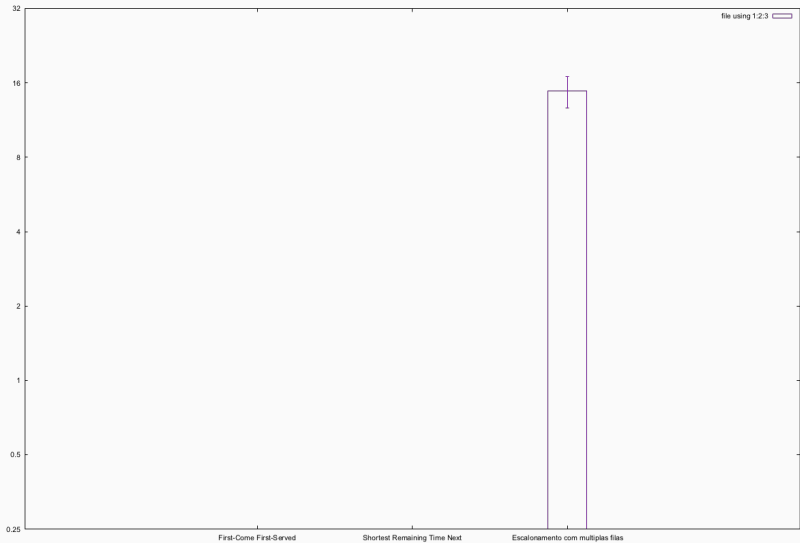
Utilizamos  $n = 10, 100, 500$  e realizamos 30 testes para cada tipo de escalonador para 4 e 8 processadores. Sabendo que o tempo médio esperado era 2.5s, o tempo total gasto previsto foi de  $610 \times 30 \times 3 \times 2.5s \times 2 = 274500s$ , que equivale a 76.25h.

Para agilizar o processo utilizamos um serviço de cloud com a seguinte configuração descrita ao lado.

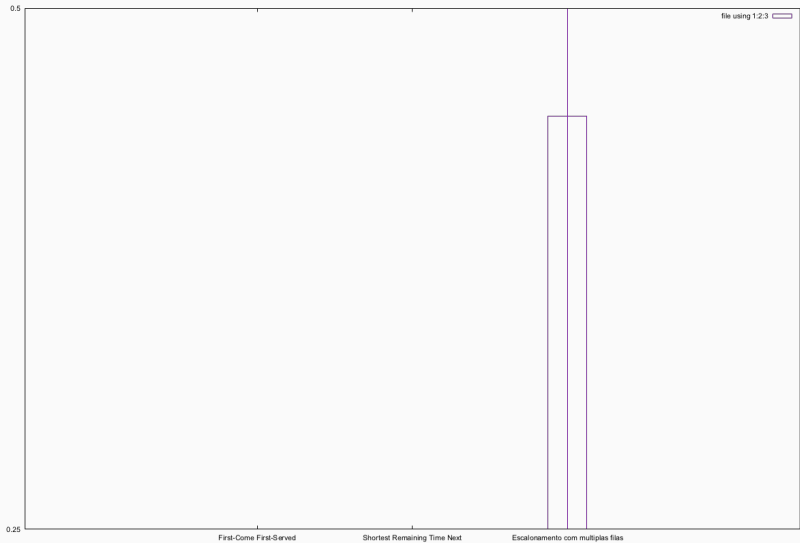
<input type="checkbox"/> Nome ^	Zona	Tipo de máquina
<input type="checkbox"/>  cpu4-1	us-west1-b	4 vCPUs, 3,6 GB
<input type="checkbox"/>  cpu4-2	us-west1-b	4 vCPUs, 3,6 GB
<input type="checkbox"/>  cpu4-3	europa-west1-b	4 vCPUs, 3,6 GB
<input type="checkbox"/>  cpu4-4	europa-west1-b	4 vCPUs, 3,6 GB
<input type="checkbox"/>  cpu4-5	asia-east1-b	4 vCPUs, 3,6 GB
<input type="checkbox"/>  cpu4-6	asia-east1-b	4 vCPUs, 3,6 GB
<input type="checkbox"/>  cpu8-1	us-central1-b	8 vCPUs, 7,2 GB
<input type="checkbox"/>  cpu8-2	us-east1-b	8 vCPUs, 7,2 GB

\*Finalizamos tudo em menos de 3h! 38.125h de processamento em duas máquinas de 8 núcleos, resulta em aproximadamente 2.4h.

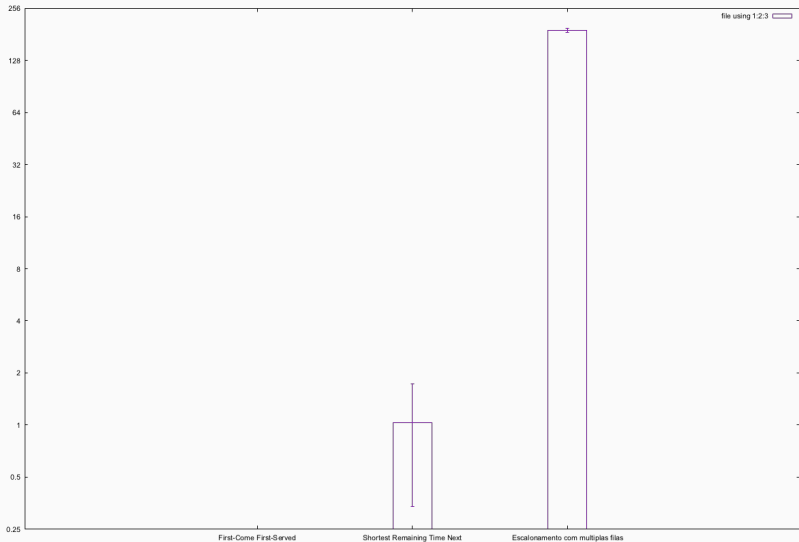
# Trocas de Contexto, $n = 10$ e 4 CPUs



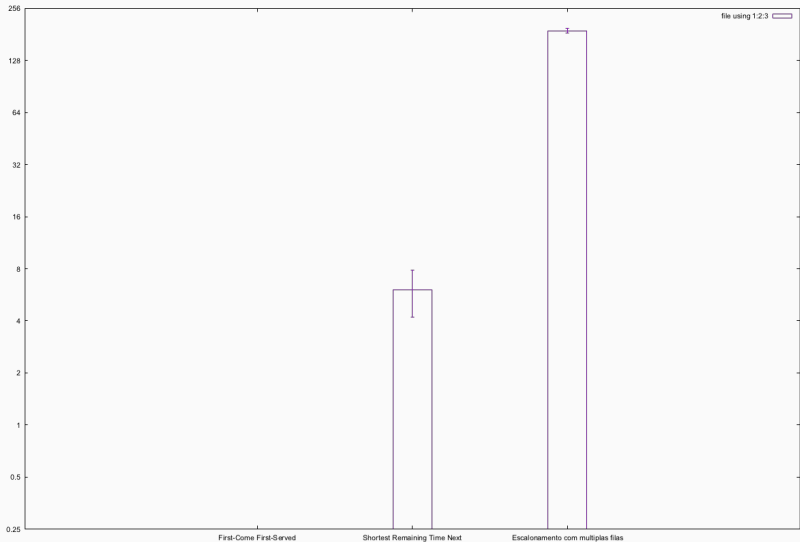
# Trocas de Contexto, $n = 10$ e 8 CPUs



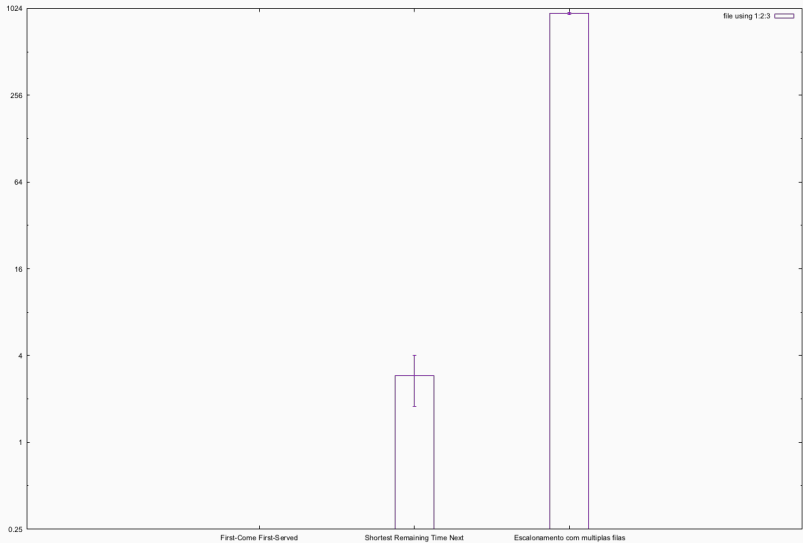
# Trocas de Contexto, $n = 100$ e 4 CPUs



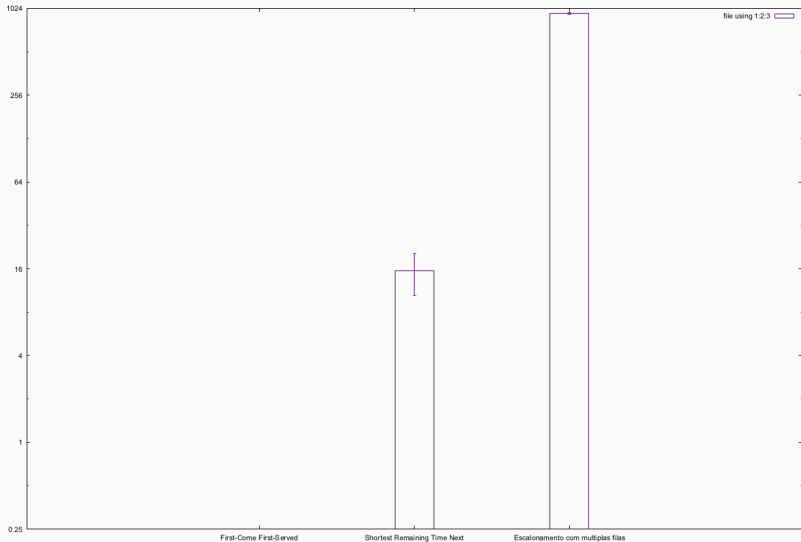
# Trocas de Contexto, $n = 100$ e 8 CPUs



# Trocas de Contexto, $n = 500$ e 4 CPUs

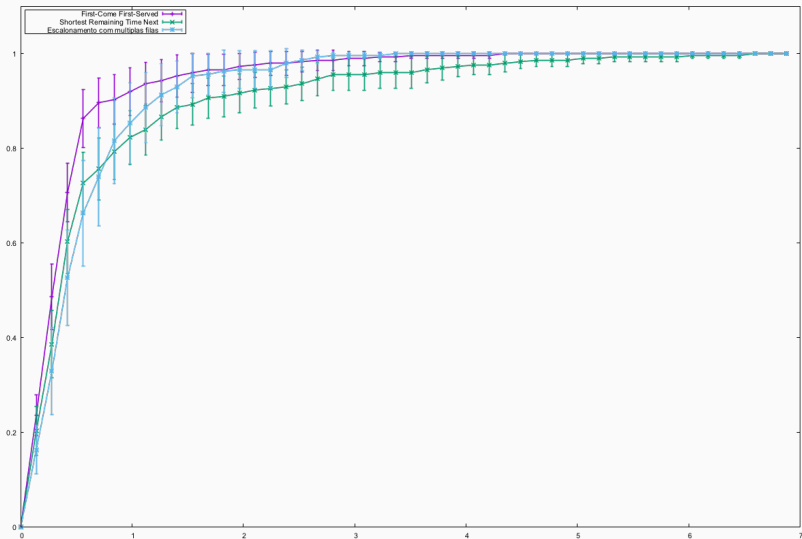


# Trocas de Contexto, $n = 500$ e 8 CPUs



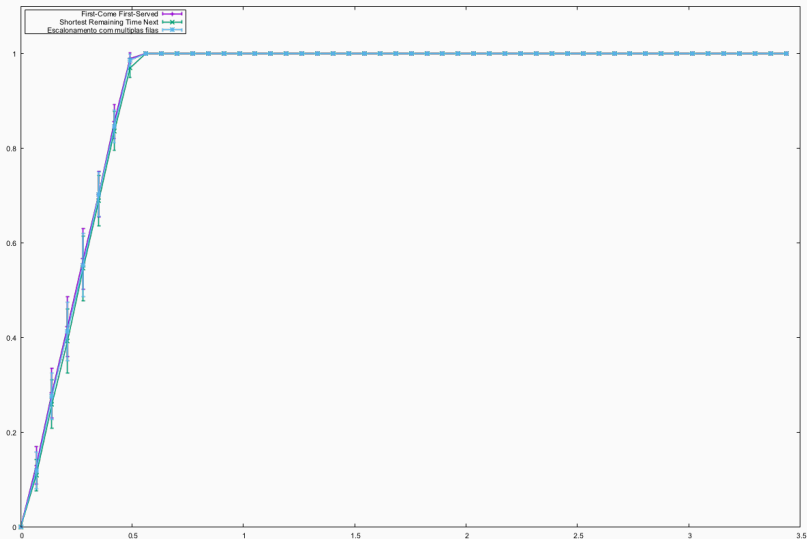


# Porcentagem de Deadlines Cumpridos, $n = 10$ e 4 CPUs



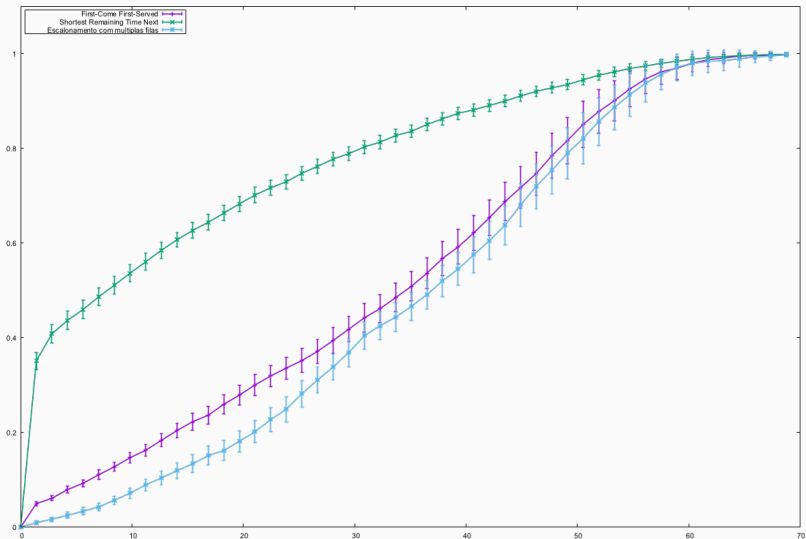
No eixo x temos o tempo em segundos que foi adicionado y do processo para determinar o deadline.

# Porcentagem de Deadlines Cumpridos, $n = 10$ e 8 CPUs



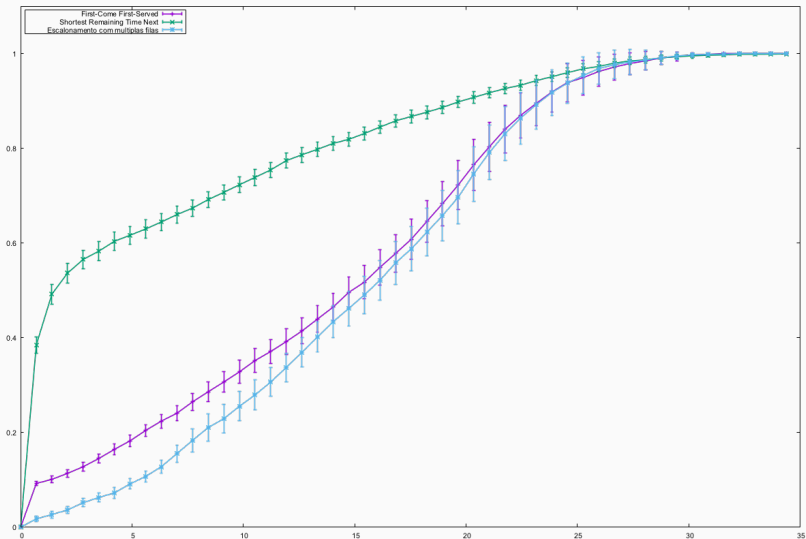
No eixo x temos o tempo em segundos que foi adicionado y do processo para determinar o deadline.

# Porcentagem de Deadlines Cumpridos, $n = 100$ e 4 CPUss



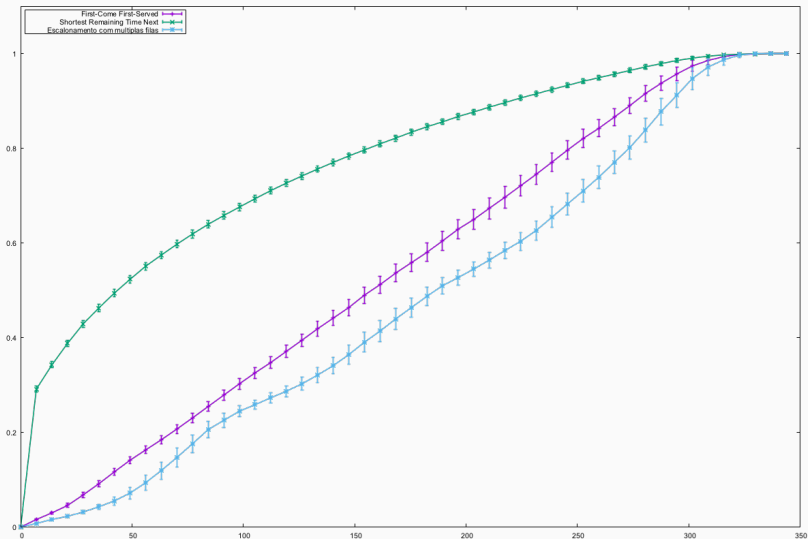
No eixo x temos o tempo em segundos que foi adicionado y do processo para determinar o deadline.

# Porcentagem de Deadlines Cumpridos, $n = 100$ e 8 CPUs



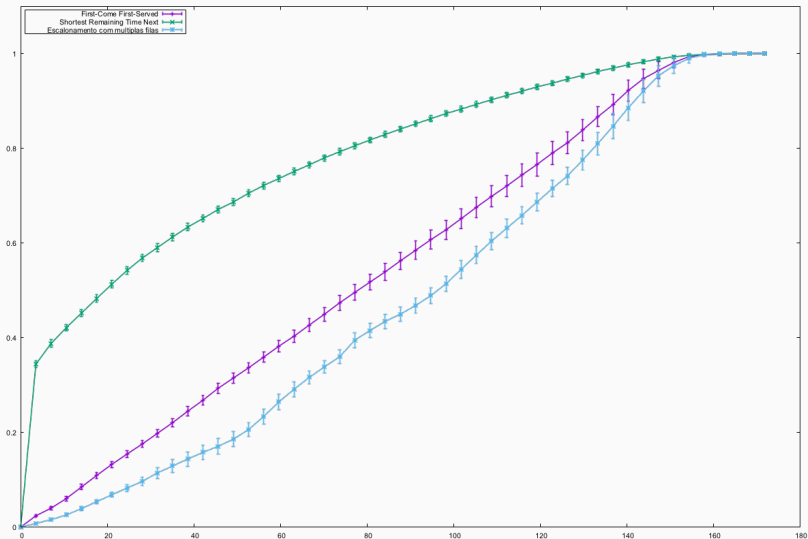
No eixo x temos o tempo em segundos que foi adicionado y do processo para determinar o deadline.

# Porcentagem de Deadlines Cumpridos, $n = 500$ e 4 CPUs



No eixo x temos o tempo em segundos que foi adicionado y do processo para determinar o deadline.

# Porcentagem de Deadlines Cumpridos, $n = 500$ e 8 CPUs



No eixo x temos o tempo em segundos que foi adicionado y do processo para determinar o deadline.