Projeto: ProbSched: Um Simulador de Algoritmos de Escalonamento Probabilistico

Disciplina: Sistemas Operativos (2024/2025)

Realizado por: Ari Jesus-49903/ Filipa Marques-50457/ Tomás Silva-51967

GitHub: https://github.com/4riJeSuS/soos25

1.Introdução

Este trabalho pretende simular o funcionamento de vários algoritmos de escalonamento de CPU, usando modelos probabílisticos para criar os processos. O simulador permite comparar o desempenho dos algoritmos em especificações tais como tempo de espera, turnaround, utilização de CPU, throuput e verificar se a deadline foi devidamente cumprida

2.Modelos Probabílistico Utilizados

- Para os tempos de chegada foi utilizada a distribuição de Poisson o qual simula chegadas aleatórias e independentes.

- Os tempos de burst (execução) foram gerados com a distribuição Exponencial que reflete tempos curtos e ocasionais bursts longos.

- As prioridades foram atribuídas de forma unforme, sem trocas entre diferentes prioridades.

3.Algoritmos de Escalonamento Implementados

- FCFS (First Come First Served): os processos são executados pela ordem de chegada e é um algoritmo não preemptivo(ocupa a CPU até terminar. Pode causar stravtion para processos que cheguem mais tarde;

- SJF (Shorstes Job First): o processo com menor Burst (tempo de execução) é o selecionado emprimeiro lugar;

- Priority (Non-Preemptive): cada processo tem associado a si uma prioridade e é escohido o de maior prioridade é o escolhido. Depois de iniciado ele executa até terminar;

- Priority (Preemptive): É semelhante ao anterior mas permite que haja interrupções, ou seja, o processo atual é interrompido e o novo é executado;

- RR (Round Robin): Cada processo recebe um quantum fixo de tempo de CPU e se o rpocesso não terminar dentro do seu quantum este é então colocado no fim da fila;

- RM (Rate Monotonic): É um algoritmo de tempo real para processos periódicos ou seja os que têm maior prioridade são os que têm periodos menores;

- EDF (Earliest Deadline First): É um algoritmo de tempo real que seleciona o processo cuja deadline está mais próxima;

4. Execução do Programa

- Compilar:

```
make clean
make
```

-Executar:

```
./probsched config.txt
```

-Modos:

-Random: processos gerados automaticamente.

-Static: Processos lidos de um ficheiro .txt

-Config.txt usado no trabalho

```
mode=static
input_file=processos.txt
algorithm=ALL
n=2
quantum=2
lambda_arrival=1.0
lambda_burst=0.5
```

5. Exemplos de entradas e saídas:

Exemplos do ficheiro processos.txt:

5.1) a.

```
0 0 3 2
1 1 5 1
2 2 2 4
```

Output:

```
P0 [arrival=0, burst=3, priority=2]
P1 [arrival=1, burst=5, priority=1]
P2 [arrival=2, burst=2, priority=4]

--- Algoritmo: FCFS ---
Processo P0 executado de 0 a 3
Processo P1 executado de 3 a 8
Processo P2 executado de 8 a 10

Estatísticas:
Tempo médio de espera: 2.67
Tempo médio de retorno: 6.00
Utilização da CPU: 100.00%
Throughput: 0.30 processos/unidade de tempo
Deadlines perdidas: 0
```

```
--- Algoritmo: SJF ---
Processo P0 executado de 0 a 3
Processo P2 executado de 3 a 5
Processo P1 executado de 5 a 10

Estatísticas:
Tempo médio de espera: 1.67
Tempo médio de retorno: 5.00
Utilização da CPU: 100.00%
Throughput: 0.30 processos/unidade de tempo
Deadlines perdidas: 0
```

```
--- Algoritmo: PRIORITY_NP ---
Processo P0 executado de 0 a 3
Processo P2 executado de 3 a 5
Processo P1 executado de 5 a 10

Estatísticas:
Tempo médio de espera: 1.67
Tempo médio de retorno: 5.00
Utilização da CPU: 100.00%
Throughput: 0.30 processos/unidade de tempo
Deadlines perdidas: 0
--------------------------------------
```

```
--- Algoritmo: PRIORITY_P ---
Tempo 0: Processo P0 executa (resta 2)
Tempo 1: Processo P0 executa (resta 1)
Tempo 2: Processo P2 executa (resta 1)
Tempo 3: Processo P2 executa (resta 0)
Tempo 4: Processo P0 executa (resta 0)
Tempo 5: Processo P1 executa (resta 4)
Tempo 6: Processo P1 executa (resta 3)
Tempo 7: Processo P1 executa (resta 2)
Tempo 8: Processo P1 executa (resta 1)
Tempo 9: Processo P1 executa (resta 0)

Estatísticas:
Tempo médio de espera: 1.33
Tempo médio de retorno: 5.33
Utilização da CPU: 100.00%
Throughput: 0.30 processos/unidade de tempo
Deadlines perdidas: 0
```

```
--- Algoritmo: RR ---
Processo P0 executado de 0 a 2 (restante=1)
Processo P1 executado de 2 a 4 (restante=3)
Processo P2 executado de 4 a 6 (restante=0)
Processo P0 executado de 6 a 7 (restante=0)
Processo P1 executado de 7 a 9 (restante=1)
Processo P1 executado de 9 a 10 (restante=0)

Estatísticas:
Tempo médio de espera: 1.00
Tempo médio de retorno: 6.67
Utilização da CPU: 100.00%
Throughput: 0.30 processos/unidade de tempo
Deadlines perdidas: 0
```

```
--- Algoritmo: RM ---
Iniciando a execução do algoritmo Rate Monotonic...
Processo P1 executado de 0 a 5
Processo P1 executado de 12 a 17
Processo P0 executado de 19 a 22
```
(...)

```
Estatísticas:
Tempo médio de espera: 19.67
Tempo médio de retorno: 512.67
Utilização da CPU: 1.88%
Throughput: 0.01 processos/unidade de tempo
Deadlines perdidas: 3
----------------------------------------

--- Algoritmo: EDF ---
Processo P1 executado de 0 a 3
Processo P0 executado de 3 a 5
Processo P2 executado de 5 a 10

Estatísticas:
Tempo médio de espera: 1.67
Tempo médio de retorno: 5.00
Utilização da CPU: 100.00%
Throughput: 0.30 processos/unidade de tempo
Deadlines perdidas: 0
----------------------------------------
```

5.2) Outros exemplos de Input:

b.

```
0 0 3 2
1 2 5 1
2 4 2 4
3 6 1 3
```

c.

```
0 0 1 3
1 1 2 2
2 2 1 1
3 3 2 4
4 4 1 5
```

d.

```
0 0 5 5
1 1 3 2
2 2 4 4
3 3 2 1
4 5 1 3
```

6. Resultados de Comparação de Algoritmos (exemplo 5.1) a.):

Algoritmo	Tempo Medio de espera	Tempo medio de retorno	CPU Utilização(%)	Thoughput
FCFS	2.67	6.00	100.00%	0.30
SJF	1.67	5.00	100.00%	0.30
PRIORITY_NP	1.67	5.00	100.00%	0.30
PRIORITY_P	1.33	5.33	100.00%	0.30
RR	1.00	6.67	100.00%	0.30
RM	14.33	81.67	10.42%	0.03
EDF	1.67	5.00	100.00%	0.30

7.Conclusão

O trabalho ProbSched permitiu observar e analisar diferentes algoritmos de escalonamento em diferentes contextos. O smiludaor é então uma ferramenta versátil para a análise do escalonamento.