Instituto Federal do Triângulo Mineiro
Análise e Desenvolvimento de Sistemas

Sistemas Operacionais

Shortest Remaining Time(SRT)

Patrocínio - MG
2024

ALEXSSANDER JOSÉ DE OLIVEIRA DE CASTRO

LUCAS AMARAL LUCIANO

PABLO VINÍCIUS LIMA SOUZA

Sistemas Operacionais

Shortest Remaining Time(SRT)

Envio de documento em PDF sobre a simulação do modelo de escalonador SRT da disciplina de Sistemas Operacionais no curso de Análise e Desenvolvimento de Sistemas

Prof.: Gilberto Oliveira Viana

Patrocínio - MG
2024

Simulação do Modelo de Escalonador SRT (Shortest Remaining Time)

1-Introdução

Esse relatório tem o objetivo de detalhar o desenvolvimento do algoritmo, foi escolhido como base para seu desenvolvimento o modelo de escalonamento Shortest Remaining Time(SRT), no qual tem como objetivo o escalonamento de processos de um sistema com base no tempo de processamento restante, que por sua vez se coloca em prioridade o processo com o menor tempo de processamento restante dentro dos processos em estado de pronto.

2-Linguagem de programação escolhida

A implementação foi feita em **C++**. A escolha dessa linguagem deve-se à familiaridade da equipe com C++ e às suas capacidades de manipulação de estruturas de dados de forma eficiente.

3-Desenvolvimento do algoritmo

Ao escolher o modelo de escalonamento SRT, prezamos pela capacidade e facilidade de manipulação dos processos usando uma lista simplesmente encadeada, já que com ela se tem mais flexibilidade de colocar e excluir elementos da lista o que não acontece no caso do uso de um vetor convencional já que ele não e um tipo dinâmico de estrutura de dados, a lista por sua vez tinha como atributos os endereços do primeiro e ultimo elemento da lista para que seja possível percorre-la, além da quantidade de processos registrados, e para representar cada processo foram criados “nós” que por sua vez guardam o endereço do próximo elemento da lista, o nome de cada processo que é representado por um caractere, o tempo de chegada do processo e seu tempo de processamento.

4-Apresentação do algoritmo trabalhado

Entrada do sistema parte 1:

O começo da execução tem a leitura da variável ‘n’ onde é escolhida a quantidade de processos que serão lidos, logo após isso se usa uma estrutura de repetição ‘for’ para ler cada processo com suas informações(letra ,chegada e processamento) e então usando a função pushBack da struct Lista se criam os novos ‘nós’ que representam os processos;

```
198  int main(){
199
200      setlocale(LC_ALL, "Portuguese");
201
202      List SRT;
203
204      char letra;
205      int chegada;
206      int processamento;
207      int n;
208
209      printf("Digite quantos processos serão lidos: ");
210      scanf("%d%*c",&n);
211
212
213      for(int i = 0 ; i < n ; i++){
214
215          scanf("%c %d %d%*c",&letra,&chegada,&processamento);
216
217          SRT.pushBack(letra,chegada,processamento);
218
219      }
```

Entrada do algoritmo:

```
Digite quantos processos serão lidos: 5
A 0 3
B 0 5
C 1 2
D 2 6
E 3 2
```

pushBack: no pushBack se registra o novo processo seguindo a ideia de uma lista normal , onde se a lista está vazia tanto o primeiro quanto ultimo elementos são o que foi registrado, além disso se tem uma variável na Lista que guarda a soma de processamento total dos processos armazenados.

```
46  void pushBack(char letra,int chegada,int processamento){
47
48      Node* n = new Node(letra,chegada,processamento);
49      c++;
50      sum = sum + processamento;
51
52      if(empty()){
53          first = n;
54          last = n;
55          return;
56      }
57
58      last->next=n;
59      last=n;
60
61  }
62
63  int soma(){
64      return sum;
65  }
```

Execução do algoritmo:

Usando a função que resgata a soma se cria um ‘for’ que chama a função verificar para execução final do processo de acordo com o tempo total de processamento.

```
223  for(int i = 0 ; i < SRT.soma() ; i++){
224
225      SRT.verificar(i);
226
227  }
```

```
116  void verificar(int tempo){
117
118      if(tempo>=10){
119          printf("%d| Processos em estado de pronto: ",tempo);
120
121      }else{
122          printf("%d | Processos em estado de pronto: ",tempo);
123      }
124
125      int t = 999999;
126
127      Node* aux = first;
128      Node* aux2 = NULL;
129
130      while(aux!=NULL){
131
132          if(aux->chegada <= tempo){
133             printf("%c - Processamento restante: %d ( Tempo de chegada: %d )\n",aux->letra,aux->proc,aux->chegada);
134             printf("  |                                  ");
135             if(aux->proc<t){
136               t = aux->proc;
137               aux2 = aux;
138             }
139          }
140
141          aux=aux->next;
142
143      }
```

Verificação da variável na CPU 1: primeiramente é printado o tempo atual da execução seguido da mensagem “Processos em estado de pronto:”, logo após isso são printados todos os processos que já chegaram naquele momento junto ao tempo de execução e o tempo de chegada deles, isso e feito

usando um ‘while’ em que também se guarda o processo com menor tempo de execução que já chegou.

Verificação da variável na CPU 2: Após descobrir o processo com menor processamento ao percorrer a lista, se printa a mensagem “Processo dentro da CPU: ”, em seguida se printa o nome do processo que foi encontrado e diminui-se seu tempo de processamento total, e no caso desse tempo chegar a zero o processo é removido da lista, e no caso de não se ter nenhum processo naquele tempo a soma e incrementada para aumentar o tempo total de execução.

```
144         printf("Processo dentro da CPU: ");
145         if(aux2!=NULL){
146
147             printf("%c",aux2->letra);
148             aux2->proc--;
149
150             if(aux2->proc==0){
151                 remove(aux2);
152             }
153
154         }else{
155             sum++;
156         }
157
158         printf("\n  |\n--------------------------------
159         if(tempo==sum-1){
160            printf("\n\n");
161            return;
162         }
163         printf("\n  |\n");
164
165     }
```

Exemplos de caso de teste:

Teste 1:

```
Digite quantos processos serão lidos: 3
A 0 2
B 1 3
C 2 2

0 | Processos em estado de pronto: A - Pocessamento restante: 2 ( Tempo de chegada: 0 )
  |                                Processo dentro da CPU: A
  |
--------------------------------------------------------------------------------
  |
1 | Processos em estado de pronto: A - Pocessamento restante: 1 ( Tempo de chegada: 0 )
  |                                B - Pocessamento restante: 3 ( Tempo de chegada: 1 )
  |                                Processo dentro da CPU: A
  |
--------------------------------------------------------------------------------
  |
2 | Processos em estado de pronto: B - Pocessamento restante: 3 ( Tempo de chegada: 1 )
  |                                C - Pocessamento restante: 2 ( Tempo de chegada: 2 )
  |                                Processo dentro da CPU: C
  |
--------------------------------------------------------------------------------
  |
3 | Processos em estado de pronto: B - Pocessamento restante: 3 ( Tempo de chegada: 1 )
  |                                C - Pocessamento restante: 1 ( Tempo de chegada: 2 )
  |                                Processo dentro da CPU: C
  |
--------------------------------------------------------------------------------
  |
4 | Processos em estado de pronto: B - Pocessamento restante: 3 ( Tempo de chegada: 1 )
  |                                Processo dentro da CPU: B
  |
--------------------------------------------------------------------------------
  |
5 | Processos em estado de pronto: B - Pocessamento restante: 2 ( Tempo de chegada: 1 )
  |                                Processo dentro da CPU: B
  |
--------------------------------------------------------------------------------
  |
6 | Processos em estado de pronto: B - Pocessamento restante: 1 ( Tempo de chegada: 1 )
  |                                Processo dentro da CPU: B
  |
--------------------------------------------------------------------------------
```

Teste 2:

```
Digite quantos processos serão lidos: 4
A 0 2
B 4 3
C 4 2
D 5 3

0 | Processos em estado de pronto: A - Pocessamento restante: 2 ( Tempo de chegada: 0 )
  |                                Processo dentro da CPU: A
  |
-------------------------------------------------------------------------------------
  |
1 | Processos em estado de pronto: A - Pocessamento restante: 1 ( Tempo de chegada: 0 )
  |                                Processo dentro da CPU: A
  |
-------------------------------------------------------------------------------------
  |
2 | Processos em estado de pronto: Processo dentro da CPU:
  |
-------------------------------------------------------------------------------------
  |
3 | Processos em estado de pronto: Processo dentro da CPU:
  |
-------------------------------------------------------------------------------------
  |
4 | Processos em estado de pronto: B - Pocessamento restante: 3 ( Tempo de chegada: 4 )
  |                                C - Pocessamento restante: 2 ( Tempo de chegada: 4 )
  |                                Processo dentro da CPU: C
  |
-------------------------------------------------------------------------------------
  |
5 | Processos em estado de pronto: B - Pocessamento restante: 3 ( Tempo de chegada: 4 )
  |                                C - Pocessamento restante: 1 ( Tempo de chegada: 4 )
  |                                D - Pocessamento restante: 3 ( Tempo de chegada: 5 )
  |                                Processo dentro da CPU: C
  |
-------------------------------------------------------------------------------------
  |
6 | Processos em estado de pronto: B - Pocessamento restante: 3 ( Tempo de chegada: 4 )
  |                                D - Pocessamento restante: 3 ( Tempo de chegada: 5 )
  |                                Processo dentro da CPU: B
  |
-------------------------------------------------------------------------------------
  |
7 | Processos em estado de pronto: B - Pocessamento restante: 2 ( Tempo de chegada: 4 )
  |                                D - Pocessamento restante: 3 ( Tempo de chegada: 5 )
  |                                Processo dentro da CPU: B
  |
-------------------------------------------------------------------------------------
  |
8 | Processos em estado de pronto: B - Pocessamento restante: 1 ( Tempo de chegada: 4 )
  |                                D - Pocessamento restante: 3 ( Tempo de chegada: 5 )
  |                                Processo dentro da CPU: B
  |
-------------------------------------------------------------------------------------
  |
9 | Processos em estado de pronto: D - Pocessamento restante: 3 ( Tempo de chegada: 5 )
  |                                Processo dentro da CPU: D
  |
-------------------------------------------------------------------------------------
  |
10| Processos em estado de pronto: D - Pocessamento restante: 2 ( Tempo de chegada: 5 )
  |                                Processo dentro da CPU: D
  |
-------------------------------------------------------------------------------------
  |
11| Processos em estado de pronto: D - Pocessamento restante: 1 ( Tempo de chegada: 5 )
  |                                Processo dentro da CPU: D
  |
```