**Bài tập chương 2 : Quản lý tiến trình**

**Bài tập về các thuật toán điều độ**

**I.Lý thuyết**

1. First come first served (FCFS)

* Là phương pháp đơn giản nhất
* Tiến trình ở trạng thái sẵn sàng được xếp vào hàng đợi FIFO
* Nhược điểm : thời gian chờ đợi trung bình lâu
* FCFS là thuật toán điều độ không phân phối lại (tức là khi đã được cấp CPU thì tiến trình sẽ sử dụng CPU đến khi kết thúc tiến trình hoặc phải dừng lại chờ kết quả đầu ra)

1. Thời gian lượng tử (round robin)

* Được cải tiến từ FCFS để dùng cho hệ thống chia sẻ thời gian
* Có cơ chế phân phối lại CPU dựa vào ngắt của đồng hồ
* Hệ thống định nghĩa những khoảng thời gian nhỏ gọi là lượng tử thời gian hoặc lát cắt thời gian t
* Tiến trình sẽ được cấp CPU trong những khoảng thời gian t như vậy trước khi bị ngắt và CPU được hệ điều hành thu hồi và cấp cho tiến trình khác
* Tuy có cải tiến nhưng thời gian chờ đợi trung bình vẫn còn dài
* Là thuật toán được sử dụng trong hệ điều hành Linux

1. Shortest Job/Process First (SJF/SPF)

* Ưu tiên tiến trình có thời gian chạy ngắn nhất
* Ưu điểm : thời gian chờ đợi trung bình ngắn
* Nhược điểm : khó triển khai trong thực tế do đòi hỏi phải biết trước độ dài tiến trình
* Cách giải quyết cho nhược điểm

+ Cách 1 : Áp dụng cho hệ thống xử lí theo mẻ, lập trình viên đăng kí thời gian chạy tối đa của tiến trình để hệ điều hành biết và lựa chọn tiến trình ngắn nhất

+ Cách 2 : Dự đoán độ dài chu kỳ sử dụng CPU tiếp theo dựa vào độ dài trung bình sử dụng CPU trước đó và ra quyết định cấp CPU

* Là tiến trình điều độ không phân phối lại
* Khi được cấp CPU, tiến trình sẽ sử dụng CPU cho đến khi hoàn thành tiến trình cho dù tiến trình mới có thời gian thực hiện ngắn hơn thời gian còn lại của tiến trình đang sở hữu CPU

1. Điều độ ưu tiên thời gian còn lại ngắn nhất (SRTF)

* Là phiên bản cải tiến của SPF và có thêm cơ chế phân phối lại
* Khi 1 tiến trình mới xuất hiện, hệ điều hành sẽ so sánh thời gian còn lại của tiến trình đang sở hữu CPU và thời gian thực hiện của tiến trình mới, từ đó lựa chọn xem cái nào ngắn hơn và ra quyết định cấp CPU
* So với RR thì việc chuyển đổi tiến trình diễn ra ít hơn => thời gian chuyển đổi ngữ cảnh ít hơn

1. Điều độ có mức ưu tiên

* Mỗi tiến trình có một mức ưu tiên riêng
* Tiến trình được ưu tiên hơn sẽ được cấp CPU trước
* Nếu có cùng mức ưu tiên thì áp dụng theo FCFS
* Một số hệ điều hành coi 0 là mức ưu tiên cao nhất, một số khác 0 lại là thấp nhất
* Là thuật toán không phân phối lại
* Tuy nhiên có thêm cơ chế phân phối lại cho phương pháp này. Nếu tiến trình mới xuất hiện có mức ưu tiên cao hơn tiến trình đang chạy thì CPU sẽ được thu hồi để cấp cho tiến trình mới
* Không phân phối lại :
* FCFS
* SPF/SJF
* Điều độ mức ưu tiên
* Phân phối lại :
* Round robin
* SRTF

**II.Bài tập**

Bài 1 : Cho danh sách các tiến trình mà HĐH phải lập lịch hoạt động với thông tin như sau

| Tên tiến trình | Thời gian đến (Arrival time) | Thời gian chạy (Burst Time) |
| --- | --- | --- |
| P1 | 2 | 12 |
| P2 | 0 | 11 |
| P3 | 2 | 7 |
| P4 | 3 | 15 |
| P5 | 4 | 8 |

Tiến hành lập lịch hoạt động của các tiến trình trên với các thuật toán sau:

1. FCFS – Đến trước phục vụ trước
2. SPF – Ưu tiên tiến trình ngắn nhất
3. SRTF – Ưu tiên tiến trình có thời gian còn lại ngắn nhất
4. Round robin (t =3) – Xoay vòng

Với mỗi thuật toán, so sánh các thông số sau:

* Thời gian chờ đợi trung bình

**Bài làm:**

1. FCFS

Lịch hoạt động:

A picture containing shoji, crossword puzzle

Description automatically generated

1. SPF

Lịch hoạt động:

A picture containing shoji, crossword puzzle, indoor

Description automatically generated

1. SRTF

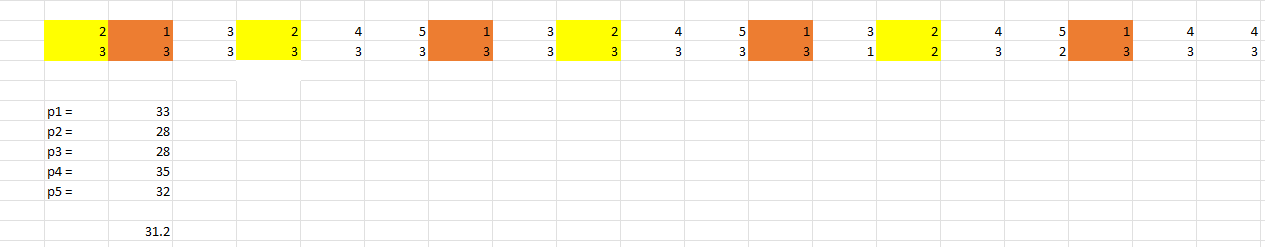
Lịch hoạt động:

A picture containing shoji, indoor, crossword puzzle, building

Description automatically generated

1. Round robin

Lịch hoạt động:



**So sánh thời gian chờ đợi: (đơn vị: chu kỳ CPU)**

**Chú ý : thời gian chờ đợi = tổng thời gian chờ đợi – thời điểm đến**

Thời gian chờ đợi trung bình = tổng thời gian chờ đợi / số tiến trình

|  | P1 | P2 | P3 | P4 | P5 | Trung bình |
| --- | --- | --- | --- | --- | --- | --- |
| FCFS | 9 | 0 | 21 | 27 | 41 | 19.6 |
| SPF | 24 | 0 | 9 | 35 | 14 | 16.4 |
| SRTF | 24 | 15 | 0 | 35 | 5 | 15.8 |
| RR (t = 3) | 33 | 28 | 28 | 35 | 32 | 31.2 |

Vậy thời gian chờ đợi trung bình của SRTF là ngắn nhất, của Round robin (t = 3) là dài nhất

Bài 2 : Cho danh sách các tiến trình mà HĐH phải lập lịch hoạt động với thông tin như sau (số ưu tiên nhỏ ứng với mức ưu tiên lớn)

| Tên tiến trình | Thời gian đến (Arrival time) | Thời gian chạy (Burst Time) | Mức độ ưu tiên |
| --- | --- | --- | --- |
| P1 | 0 | 10 | 1 |
| P2 | 2 | 8 | 3 |
| P3 | 4 | 5 | 2 |
| P4 | 5 | 3 | 2 |

* 1. Điều độ theo mức ưu tiên không có phân phối
  2. Điều độ ưu tiên tiến trình ngắn nhất
  3. Điều độ ưu tiên thời gian còn lại ngắn nhất

1. Điều độ theo mức ưu tiên không có phân phối

| P1 | P3 | P4 | P2 |
| --- | --- | --- | --- |
| 10 | 5 | 3 | 8 |

1. Điều độ ưu tiên tiến trình ngắn nhất

| P1 | P4 | P3 | P2 |
| --- | --- | --- | --- |
| 10 | 3 | 5 | 8 |

1. Điều độ ưu tiên thời gian còn lại ngắn nhất

| P1 | P3 | P4 | P3 | P1 | P2 |
| --- | --- | --- | --- | --- | --- |
| 4 | 1 | 3 | 4 | 6 | 8 |

Thời gian chờ đợi trung bình

|  | P1 | P2 | P3 | P4 | Trung bình |
| --- | --- | --- | --- | --- | --- |
| ĐĐTMUT | 0 | 16 | 6 | 10 | 8 |
| SPF | 0 | 16 | 9 | 5 | 7.5 |
| SRTF | 8 | 16 | 3 | 0 | 6.75 |

Vậy thời gian chờ đợi trung bình của SRTF là ngắn nhất, của điều độ theo mức ưu tiên là dài nhất.

**Bài tập thuật toán người cho vay**

**I.Lý thuyết**

- Thuật toán người cho vay là thuật toán dùng để phòng tránh bế tắc của hệ điều hành

- Ngăn ngừa bế tắc sử dụng những quy tắc và ràng buộc khi cấp phát tài nguyên để ngăn xảy ra bế tắc. Việc này gây ra sự kém hiệu quả, giảm hiệu năng của tiến trình.

-> Từ đó ta có giải giáp tốt hơn là phòng tránh bế tắc

- Bản chất của phòng tránh bế tắc là ngăn ngừa, khác ở chỗ nó sẽ cho phép 3/4 điều kiện xảy ra và chỉ đảm bảo trạng thái bế tắc sẽ không đạt đến được (tức là không để cho 4/4 điều kiện cùng xảy ra để dẫn tới bế tắc)

- Mỗi yêu cầu cấp phát tài nguyên sẽ được xem xét tùy vào tình hình tại thời điểm đó chứ không phải theo 1 nguyên tắc như ngăn ngừa.

**Thuật toán người cho vay**

* Khi tiến trình mới khởi tạo, nó sẽ báo dạng tài nguyên và số lượng tối đa cho mỗi loại mà nó cần trong cả tiến trình.
* Nếu số lượng yêu cầu không vượt quá khả năng của hệ điều hành của hệ thống thì tiến trình sẽ được khởi tạo

Trạng thái của tiến trình sẽ được xác định bởi những thông tin sau

* Số lượng tối đa tài nguyên mà tiến trình yêu cầu sẽ được lưu trong một ma trận M, với M[ i ] [ j ] sẽ cho ta biết số lượng tài nguyên j cần cho tiến trình i.
* Số lượng tài nguyên còn lại của hệ thống được cho dưới dạng vector A, với A[ j ] là số lượng còn lại của tài nguyên j.
* Lượng tài nguyên đã cấp cho tiến trình được thể hiện được thể hiện ở ma trận D, với D[ i ] [ j ] là số lượng tài nguyên j đã cấp cho tiến trình i.
* Tương tự lượng tài nguyên j còn cần cấp cho tiến trình i sẽ được lưu trong ma trận C.

Trạng thái an toàn là trạng thái mà từ đó có ít nhất một phương pháp cấp phát sao cho hệ thống không xảy ra bế tắc và ngược lại sẽ là trạng thái không an toàn.

**Thuật toán xác định trạng thái an toàn**

1.Khai báo mảng W kích thước m và F kích thước n

W = A ( A là vector lưu số lượng tài nguyên còn lại của hệ thống )

F[n] là mảng lưu trạng thái hoàn thành hay không hoàn thành của các tiến trình

( khởi tạo F[ i ] = false )

2.Tìm i sao cho

F[ i ] = false và C[ i ][ j ] <= W[ j ]

Nếu không tìm thấy i thỏa mãn thì ta chuyển sang bước 4

3.W = W + D[ i ]

F[ i ] = true

Quay lại bước 2

4.Nếu F[ i ] = true với mọi i ta kết luận an toàn

Else không an toàn

**II.Bài tập**

Xét trạng thái cấp phát tài nguyên của hệ thống như sau:

P2 yêu cầu cấp phát 1 tài nguyên Y, 2 tài nguyên Z. Sử dụng thuật toán người cho vay để xác định xem yêu cầu của P2 có được đáp ứng hay không?

A picture containing text, crossword puzzle

Description automatically generated

7 - 5 - 5

Khai báo mảng W = A = [ 3 ] [ 2 ] [ 2 ] và mảng F[ i ] = false với mọi i

Tìm i sao cho F[i] = false và C[i][j] <= W[j] với mọi j

* Với i = 1 => không thỏa mãn vì C[1][1] = 7 > W[1] = 3
* Với i = 2 => thỏa mãn

W = W + D[2] = [5][3][4] = 3 - 2 - 2 + 2 - 1 - 2

F[2] = true

* Với i = 1 => không thỏa mãn
* Với i = 3 => không thỏa mãn
* Với i = 4 thỏa mãn

W = 5 - 3 - 4 +2 - 1 - 1 = [7][4][5]

F[4] = true

* Với i = 1 => thỏa mãn

W = 7 - 4 - 5 + 0 - 1 - 0 = [7][5][5]

F[1] = true

* Với i = 3 => thỏa mãn

W = 7 - 5 - 5 + 3 - 0 - 2 = [10][5][7]

F[3] = true

=> Như vậy yêu cầu cấp tài nguyên (X, Y, Z) = (0, 1, 2) của P2 được đáp ứng vì sau khi cấp vẫn có thể tìm ra 1 cách cấp phát không dẫn đến bế tắc là P2, P4, P1, P3 (đây là trạng thái an toàn)