TRƯỜNG ĐẠI HỌC PHENIKAA

KHOA CÔNG NGHỆ THÔNG TIN

BỘ MÔN HỆ ĐIỀU HÀNH

--❧•❧--Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**Đề Tài:**

**Cpu Scheduling,The Multi-Level Feedback Queue, Proportional Share**

**GIẢNG VIÊN: TS. Trần Đăng Hoan**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | **Tên nhóm:** | **Nhóm 2** | | **Lớp học phần:** | **Hệ điều hành** | |  |
| |  | | --- | | **Link video thuyết trình**:  <https://www.youtube.com/watch?v=JOmmXyGZPlU> | |  | |  |
|  |  |
| *Năm học: 2021 – 2022* | |

**THÔNG TIN CỤ THỂ TỪNG THÀNH VIÊN CỦA NHÓM:**

|  |  |
| --- | --- |
| Họ và Tên | MSV |
| Nguyễn Văn Thuyên(nhóm trưởng) | 20010928 |
| Tăng Thế Mạnh | 20010910 |
| Phạm Tiến Thành Công | 20010886 |
| Phạm Đỗ Việt Dũng | 20010891 |
| Đinh Văn Thức | 20010927 |

MỤC LỤC

[MỞ ĐẦU 4](#_Toc89685005)

[NỘI DUNG 5](#_Toc89685006)

[PHẦN I: CPU Scheduling 5](#_Toc89685007)

[i. CPU Scheduling in Operating System 6](#_Toc89685008)

[ii. Tiêu Chí Lập Lịch 6](#_Toc89685013)

[iii. Lập lịch CPU 6](#_Toc89685014)

[PHẦN II:Multilevel Feedback Queue Scheduling (MLFQ) CPU Scheduling.5](#_Toc89685021)

[i. Giới thiệu 6](#_Toc89685022)

[ii. Định Nghĩa 6](#_Toc89685023)

[iii. Sự cần thiết của Lập lịch hàng đợi Phản hồi Đa cấp (MFQS) 6](#_Toc89685024)

[iv. Ưu điểm của MFQS 6](#_Toc89685025)

[v. Nhược điểm của MFQS 6](#_Toc89685026)

[PHẦN III: Lottery Scheduling 5](#_Toc89685027)

[i. Công cụ lập kế hoạch chia sẻ tỷ lệ 5](#_Toc89685028)

[ii. Khái niệm cơ bản 6](#_Toc89685029)

[iii. Lập kế hoạch xổ số 6](#_Toc89685030)

[iv. Vẻ đẹp của sự ngẫu nhiên (trong lập kế hoạch) 6](#_Toc89685031)

[v.Cơ chế đặt vé 7](#_Toc89685032)

[vi. Thực hiện 8](#_Toc89685033)

[vi.Nghiên cứu công bằng xổ số 9](#_Toc89685034)

[vii. Lập lịch Stride (công cụ lập lịch Fair-share xác định) 9](#_Toc89685035)

[viii. Bộ lập lịch hoàn toàn công bằng cho Linux (CFS) 11](#_Toc89685036)

[ix. Độ đẹp và trọng lượng 11](#_Toc89685037)

[x. Bộ lập lịch hoàn toàn công bằng cho Linux (CFS) 12](#_Toc89685038)

# MỞ ĐẦU

Hệ điều hành là một hệ thống các chương trình hoạt động giữa người sử dụng (user) và phần cứng của máy tính. Mục tiêu của hệ điều hành là cung cấp một môi trường để người sử dụng có thể thi hành các chương trình. Nó làm cho máy tính dễ sử dụng hơn, thuận lợi hơn và hiệu quả hơn. Hệ điều hành là một phần quan trọng của hầu hết các hệ thống máy tính. Một hệ thống máy tính thường được chia làm bốn phần chính : phần cứng, hệ điều hành, các chương trình ứng dụng và người sử dụng. Phần cứng bao gồm CPU, bộ nhớ, các thiết bị nhập xuất, đây là những tài nguyên của máy tính. Chương trình ứng dụng như các chương trình dịch, hệ thống cơ sở dữ liệu, các trò chơi, và các chương trình thương mại. Các chương trình này sử dụng tài nguyên của máy tính để giải quyết các yêu cầu của người sử dụng. Hệ điều hành điều khiển và phối hợp việc sử dụng phần cứng cho những ứng dụng khác nhau của nhiều người sử dụng khác nhau. Hệ điều hành cung cấp một môi trường mà các chương trình có thể làm việc hữu hiệu trên đó.

# NỘI DUNG

# PHẦN I: CPU Scheduling

## CPU Scheduling in Operating System

Lập lịch CPU là một quá trình cho phép một quá trình sử dụng CPU, trong khi việc thực thi một quá trình khác bị tạm dừng (ở trạng thái chờ) do không có sẵn bất kỳ tài nguyên nào như I / O, do đó tận dụng hết CPU. Mục đích của việc lập lịch trình CPU là làm cho hệ thống hoạt động hiệu quả, nhanh chóng và công bằng. Bất cứ khi nào CPU không hoạt động, hệ điều hành phải chọn một trong các tiến trình trong hàng đợi sẵn sàng để thực thi. Quá trình lựa chọn được thực hiện bởi bộ lập lịch ngắn hạn (hoặc bộ lập lịch CPU). Bộ lập lịch trình chọn từ các quá trình trong bộ nhớ sẽ được thực thi và gán CPU cho một trong số chúng.

### 1.CPU Scheduling: Dispatcher

Một thành phần khác tham gia vào chức năng lập lịch của CPU là Bộ điều phối. Bộ lập lịch là một mô-đun chuyển quyền điều khiển CPU đến quá trình được bộ lập lịch ngắn hạn chọn. Chức năng này bao gồm: • Chuyển đổi ngữ cảnh • Chuyển sang chế độ người dùng • Chuyển đến vị trí thích hợp trong chương trình người dùng để khởi động lại chương trình từ vị trí đã dừng lần trước. Bộ lập lịch phải càng nhanh càng tốt vì nó sẽ được gọi trong mỗi lần chuyển đổi quy trình. Thời gian cần thiết để bộ lập lịch dừng một quá trình và bắt đầu một quá trình khác được gọi là độ trễ lập lịch

### 2.Types of CPU Scheduling

Quyết định lập lịch trình CPU có thể xảy ra trong bốn trường hợp sau:

1. Khi một tiến trình chuyển từ trạng thái đang chạy sang trạng thái chờ

2. Khi một tiến trình chuyển từ trạng thái đang chạy sang trạng thái sẵn sàng

3. Khi một tiến trình chuyển từ trạng thái chờ sang trạng thái sẵn sàng 4. Khi một tiến trình kết thúc.

1 và 4 là lập lịch không ưu tiên

2 và 3 là ưu tiên. Trong trường hợp 1 và 4, không có sự lựa chọn nào về lịch trình. Một quy trình mới (nếu một quy trình tồn tại trong hàng đợi sẵn sàng) phải được chọn để thực thi

### 3.Non-Preemptive Scheduling

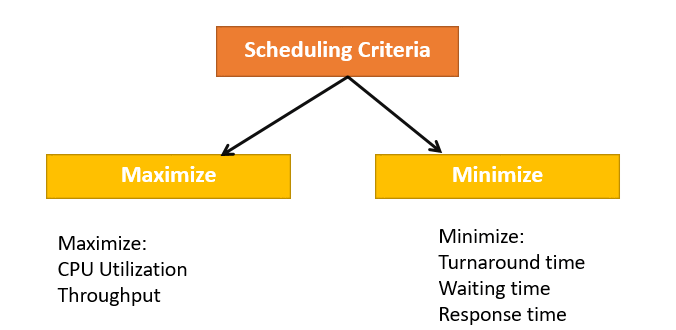
Đây là phương pháp duy nhất có thể được sử dụng trên các nền tảng phần cứng nhất định vì nó không yêu cầu phần cứng đặc biệt (chẳng hạn như bộ hẹn giờ) cần thiết cho việc lập lịch trước. Trong lập lịch không ưu tiên, nó không làm gián đoạn quá trình chạy CPU trong quá trình thực thi. Thay vào đó, nó đợi quá trình hoàn thành thời gian bùng nổ CPU và sau đó nó có thể phân bổ CPU cho bất kỳ quá trình nào khác. Một số thuật toán dựa trên lập lịch không ưu tiên là: Lập lịch công việc ngắn hạn (SJF về cơ bản là không ưu tiên) và lập lịch ưu tiên (không ưu tiên).

### 4.Preemptive Scheduling

Trong kiểu lập kế hoạch này, các nhiệm vụ thường được chỉ định mức độ ưu tiên. Đôi khi cần phải chạy một tác vụ có mức độ ưu tiên cao hơn trước một tác vụ khác, mặc dù nó đang chạy. Do đó, tác vụ đang chạy sẽ bị gián đoạn trong một khoảng thời gian và sẽ tiếp tục sau khi tác vụ ưu tiên được thực thi. Loại lập lịch này chủ yếu được sử dụng khi tiến trình chuyển từ trạng thái đang chạy sang trạng thái sẵn sàng hoặc từ trạng thái chờ sang trạng thái sẵn sàng. Tài nguyên (tức là chu kỳ CPU) chủ yếu được phân bổ cho tiến trình trong một khoảng thời gian giới hạn và sau đó được lấy đi. Quá trình sẽ vẫn ở trong hàng đợi sẵn sàng cho đến khi nó có cơ hội thực thi tiếp theo. Một số thuật toán dựa trên lập lịch trước bao gồm lập lịch theo vòng (RR), thời gian còn lại ngắn nhất trước (SRTF), lập lịch ưu tiên (phủ đầu), v.v.

## Tiêu Chí Lập Lịch

* Trong Tiêu Chí lập lịch CPU sẽ có 2 tiêu chí: tối đa hóa và giảm thiểu



* **Tối đa hóa gồm những gì ???**
* Sử dụng CPU: là nhiệm vụ chính mà hệ điều hành cần có, để đảm bảo rằng CPU hoạt động tối đa công suất có thể. Nó có thể nằm trong khoảng từ 0% đến 100%. Tuy nhiên, đối với **Hệ điều hành thời gian thực**, nó có thể dao động từ 40% đối với hệ thống cấp thấp và 90% đối với hệ thống cao cấp.
* Thông lượng(**Throughput**): Nó là tổng số quá trình được hoàn thành trên một đơn vị thời gian hay nói đúng hơn là tổng khối lượng công việc được thực hiện trong một đơn vị thời gian, và khi CPU đang bận thực hiện tiến trình, tại thời điểm đó, công việc đang được thực hiện và công việc được hoàn thành trên một đơn vị thời gian được gọi là Thông lượng.
* **Giảm Thiểu gồm những gì ???**
* Thời gian chờ đợi: là khoảng thời gian cụ thể cần phải có sẵn trong hàng đợi, để sẵn sàng được kiểm soát và sử dụng trên CPU
* Thời gian đáp ứng(thời gian phản hồi): là khoảng thời gian từ lúc yêu cầu gửi đi, cho đến khi phản hồi được đáp ứng(tạo ra), chú ý: đây là khoảng thời gian đến khi bắt đầu tiến chình chứ không phải là thời gian hoàn thành tiến trình
* Thời gian quay vòng: là khoảng thời gian để nó thực hiện 1 tiến trình cụ thể, tức là Khoảng thời gian từ khi gửi quy trình đến thời điểm hoàn thành quy trình
* Nhìn chung, việc sử dụng CPU và Thông lượng được tối đa hóa và các yếu tố khác được giảm bớt để tối ưu hóa phù hợp.
* **Interval Timer(bộ hẹn giờ khoảng):**
* Ngắt hẹn giờ là một phương pháp có liên quan mật thiết đến quyền ưu tiên.
* Khi một quy trình bất kì được phân bổ bởi CPU, khi đó một bộ đếm thời gian có thể được đặt thành một khoảng thời gian nhất định.
* Cả bộ hẹn giờ và chế độ ưu tiên sẽ buộc một quá trình trả lại CPU khi tiến trình xảy ra lỗi
* Hầu hết hệ điều hành nó sẽ sử dụng một số dạng đếm thời gian để ngăn một số tiến trình hoạt động mãi mãi
* **Dispatcher(điều phối viên):**
* Điều phối viên là gì: nó là một mô-đun cung cấp quyền kiểm soát CPU đối với các tiến trình
* Điều phối viên thì nó phải nhanh, để nó có thể chạy tên mọi ngữ cảnh, và độ trễ của nó là lượng thời gian cần thiết để bộ lập lịch CPU dừng lại một tiến trình này và bắt đầu một tiến trình khác
* Các Chức năng của Dispatcher thực hiện:

+ Chuyển đổi ngữ cảnh

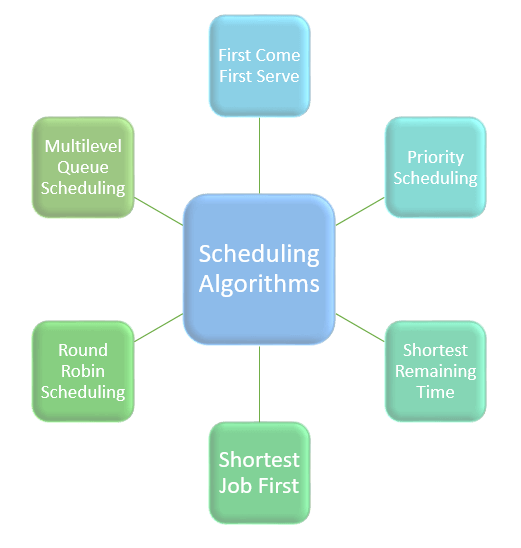
+ Chuyển đổi sang chế độ người dùng

+ Di chuyển đến vị trí chính xác trong chương trình mới được tải.

## Lập lịch CPU

* **Mục đích:**
* Cải thiện hiệu quả của CPU
* Giúp nó phân bổ tài nguyên giữa các tiến trình cạnh tranh nhau
* Việc sử dụng CPU tối đa có thể đạt được với đa lập trình
* Các quy trình thực thi sẽ được nằm sẵn trong hàng đợi.
* **Các thuật toán lập lịch CPU**
* Có 6 loại thuật toán lập lịch phổ biến:

1. First Come First Serve (FCFS)
2. Shortest-Job-First (SJF) Scheduling
3. Shortest Remaining Time
4. Priority Scheduling
5. Round Robin Scheduling
6. Multilevel Queue Scheduling



### 1.First Come First Serve (FCFS)

#### 1.1Khái Niêm

* Thuật toán lập lịch “First come first serve”. Giống như tên thì thuật toán này yêu cầu: quá trình nào đến trước thì sẽ thực thi trước quá trình nào đến sau thì sẽ thực thi sau.
* First come first serve thì nó cũng giống như cấu trúc dữ liệu: Hàng đợi.
* Ví dự thực tế của thuật toán First come first serve đó là mua vé tại quầy bán vé.

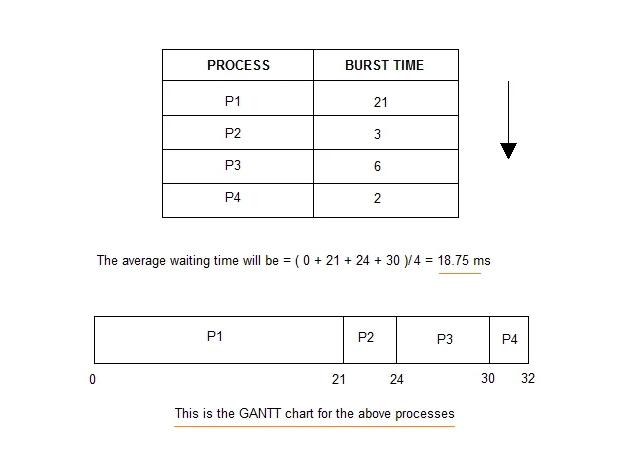
#### 1.2Các vấn đề về lập lịch First come first serve

* Thuật toán Non Pre-emptive hay thuật toán về mức độ ưu tiên sẽ không thể sử dụng, mức độ ưu tiên của quy trình không còn quan trọng.
* Chính vì thế hệ thống sẽ dễ bị sập, chỉ vì lập kế hoạch quy trình không đúng.
* Thời gian trung bình không tối ưu
* Việc sử dụng tài nguyên song song là không thể. Vì thế sẽ dẫn đến hiệu ứng “**Convoy**”. Do đó việc sử dụng tài nguyên (CPU, I/O,v.v) kém.

#### 1.3Hiệu ứng Convoy là gì ???

* Hiệu ứng Convoy xảy ra khi có nhiều tiến trình diễn ra, và nó cần sử dụng tài nguyên trong một thời gian ngắn nhưng lại bị chặn bởi một tiến trình đang giữ tài nguyên đó trong một thời gian dài !
* Điều này dẫn đến việc sử dụng nhiều tài nguyên và hiệu suất kém

#### 1.4Sơ đồ minh họa hoạt động:

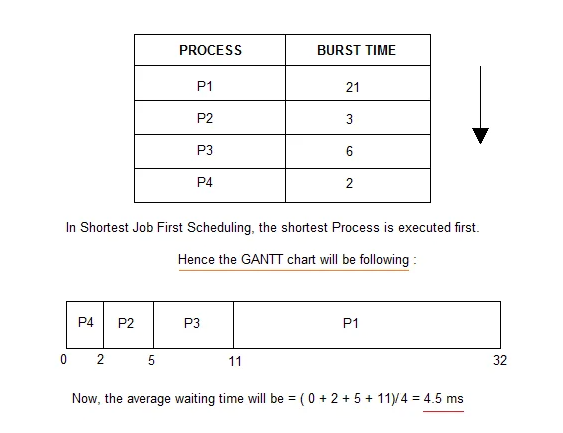


### 2.Shortest-Job-First (SJF) Scheduling

#### 2.1Đặc điểm

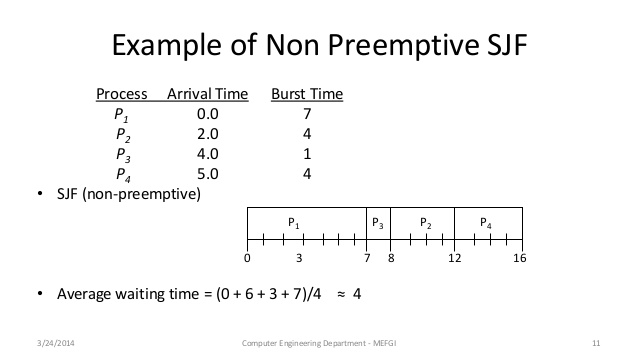
* Nó hoạt đọng theo nguyên lý với thời gian hoặc khoảng thời gian ngắn nhất sẽ chạy trước tiên.
* Sẽ giảm thiểu tối đa thời gian chờ đợi.
* Sử dụng trong hệ thống hàn loạt
* Để thực hiện thành công thì buộc bộ sử lý phải biết trước được thời gian thực thi của các tiến trình. Chính vì thế điều này thực tế thì khá thiếu khả thi.
* Thuật toán lập lịch này là tối ưu nếu tất cả các tiến trình sẵn sàng thực thi cùng một lúc.

#### 2.2Nguyên lý hoạt động



2.3Vấn đề của Shortest-Job-First với Non Pre-emptive

* Khi thời gian sẵn sàng hoạt động của các tiến trình là khác nhau, hay các tiến trình nó không có sẵn tại hàng đợi tại thời điểm 0. Trong những tình huống như vậy, đôi khi các tiến trình mới có thời gian ngắn phải đợi quá trình thực thi kết thúc thì mới thực hiện được. Bởi vì trong SJF không ưu tiên khi xuất hiện một tiến trinhg mới,
* Chính vì vậy nó dẫn đến việc thiếu tài nguyên, và một tiến trình ngắn hơn phải đợi một tiến trình dài hơn thực hiện



### 3.Shortest Remaining Time

#### 3.1Khái Niệm

* Trong Phương pháp này các tiến trình sẽ được sắp xếp sao cho các tiến trình sắp hoàn thành sẽ được phân chạy trước và các tiến trình chưa chạy sẽ được phân chạy sau. Điều này sẽ ngăn các tiến trình mới thực thi và nó sẽ thực thi các tiến trình cũ hơn

#### 3.2Đặc điểm

* Phương pháp này hầu heeys được áp dụng trong môi trường hoàng loạt, nơi ưu tiên các công việc ngắn hạn
* Đây thì không phải là một phương pháp lý tưởng để thực hiện nó trong một hệ thống chia sẻ, mà thời gian CPU cần thiết không xác định.
* Tuy nhiên thì đây là một thuật toán giúp lập lịch trình với thời gian ngắn nhất có thể.

### 4.Priority Scheduling

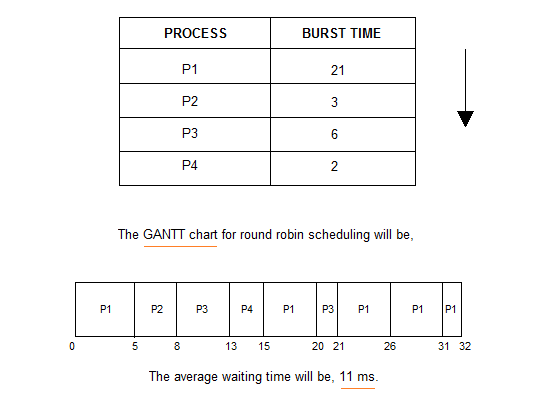
* Đây là phương pháp lập lịch dự trên mức độ ưu tiên. Khi đó bộ lập lịch chọn các tác vụ, tiến trình để làm việc theo mức độ ưu tiên
* Vì vậy các công việc có độ ưu tiên cao hơn sẽ được thực hiện sớm hơn. Và các công việc có mức độ ưu tiên ngang nhau sẽ được thực hiện theo Round-Robin hoặc First Come First Serve.

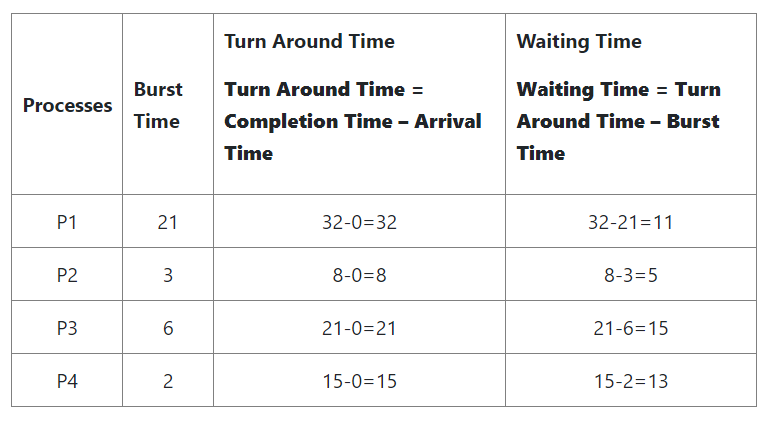
### 5.Round-Robin Scheduling.

#### 5.1Khái Niệm

* Thuật toán lập lịch Round Robin chủ yếu được thiết kế cho các hệ thống chia sẻ thời gian.
* Thuật toán này tương tự FCFS nhưng trong thuật toán lập lịch Round Robin quyền ưu tiên sẽ được thêm vào
* Thuật toán này sẽ khắc phục tình trạng Convoy ở thật toán FCFS. Vì tắt cả các tiến trình thì đều có một phần CPU công bằng.

#### 5.2Nguyên lý hoạt động





#### 5.3Một số ưu, nhực điểm:

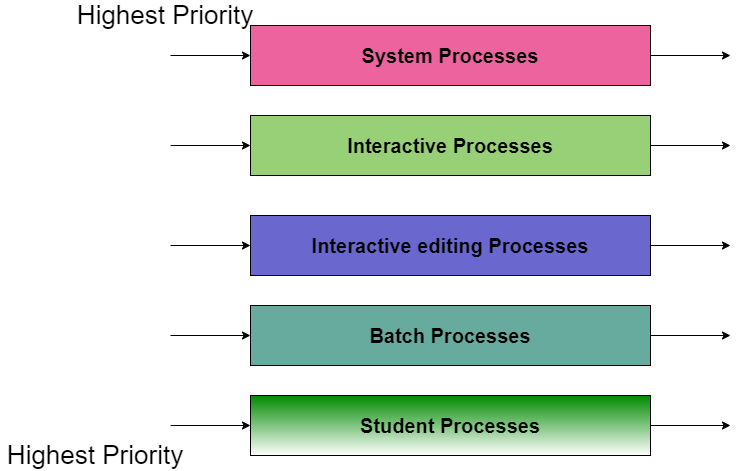
* Ưu điểm:
* Về thời gian phản hồi trung bình bì thuật toán này cho hiệu suất tốt nhất
* Công việc sẽ được CPU phân bổ hợp lý
* Sử lý được tất cả các tiến trình mà không cần bất kì sự ưu tiên nào.
* Mỗi tiến trình sẽ có cơ hội lên lịch lại sau một thời gian cụ thể.
* Nhược điểm:
* Thuật toán dành nhiều thời gian cho việc chuyển đổi giữa các tiến trình và lập kế hoạch
* Cung cấp thời gian chờ và phản hồi lớn

### 6.Multilevel Queue Scheduling

#### 6.1 Khái niệm

* Thuật toán này, nó sẽ tách hàng đợi sẵn sàng thành nhiều hàng đợi riêng biệt khác nhau. Các quá trình gán cho một hàng đợi dựa trên một thuộc tính cụ thể của tiến trình, như : độ ưu tiên, kích thước bộ nhớ, vv
* Tuy nhiên đây không phải là một thuật toán lập lịch độc lập mà nó cần sử dụng các thuật toán khác để lập lịch công việc.

#### 6.2Ví dụ mô tả:



Nó sẽ chia hàng đợi có sẵn thành các hàng đợi:

* Quy trình hệ thống (FCFS)
* Quy trình tương tác (SJF)
* Quy trình chỉnh sửa tương tác (SJF)
* Quy trình hàng loạt (RR)
* Quy trình học sinh (lên lịch ưu tiên)

#### 6.3 Ưu nhược điểm của tiến trình

* Ưu điểm:
* Ta có thể áp dụng nhiều loại lập lịch khác nhau
* Nhược điểm:
* Nó sẽ làm cho các tiến trình thấp cấp không được thực thi hoặc nếu có thì phải đợi trong một thời gian dài.

# PHẦN II:Multilevel Feedback Queue Scheduling (MLFQ) CPU Scheduling

## Giới thiệu

Trong thuật toán lập lịch hàng đợi đa cấp, các quy trình được gán vĩnh viễn cho một hàng đợi khi vào hệ thống. Các tiến trình không di chuyển giữa các hàng đợi. Thiết lập này có ưu điểm là chi phí lập lịch trình thấp, nhưng nhược điểm là không linh hoạt.

Tuy nhiên, lập lịch hàng đợi phản hồi đa cấp cho phép một quá trình di chuyển giữa các hàng đợi. Ý tưởng là tách các quy trình có các đặc điểm CPU-nổ khác nhau. Nếu một tiến trình sử dụng quá nhiều thời gian của CPU, nó sẽ được chuyển đến hàng đợi có mức độ ưu tiên thấp hơn. Tương tự, một quá trình chờ đợi quá lâu trong hàng đợi có mức độ ưu tiên thấp hơn có thể được chuyển sang hàng đợi có mức độ ưu tiên cao hơn. Hình thức lão hóa này ngăn chặn nạn đói.

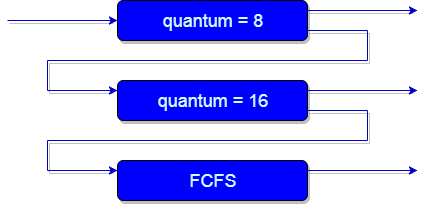
Nói chung, bộ lập lịch hàng đợi phản hồi đa cấp được xác định bởi các tham số sau:

* Số lượng hàng đợi.
* Thuật toán lập lịch cho mỗi hàng đợi.
* Phương pháp được sử dụng để xác định thời điểm nâng cấp quy trình lên hàng đợi có mức độ ưu tiên cao hơn.
* Phương pháp được sử dụng để xác định thời điểm hạ cấp một quy trình xuống hàng đợi có mức độ ưu tiên thấp hơn.
* Phương pháp được sử dụng để xác định hàng đợi mà một quy trình sẽ vào khi quy trình đó cần dịch vụ.

## Định Nghĩa

Định nghĩa của bộ lập lịch hàng đợi phản hồi đa cấp làm cho nó trở thành thuật toán lập lịch CPU chung nhất. Nó có thể được cấu hình để phù hợp với một hệ thống cụ thể đang được thiết kế. Thật không may, nó cũng yêu cầu một số phương tiện chọn giá trị cho tất cả các tham số để xác định bộ lập lịch tốt nhất. Mặc dù hàng đợi phản hồi đa cấp là **sơ đồ chung nhất** , nhưng nó cũng là **sơ đồ phức tạp nhất** .

**Ví dụ về hàng đợi phản hồi đa cấp:**

****

**Giải trình:**

Trước hết, Giả sử rằng hàng đợi 1 và 2 theo vòng tròn với lượng tử thời gian 8 và 16 tương ứng và hàng đợi 3 theo FCFS. Một trong những cách triển khai của Lập lịch hàng đợi Phản hồi Đa cấp như sau:

1. Nếu bất kỳ quá trình nào bắt đầu thực thi thì trước hết nó sẽ vào hàng đợi 1.
2. Trong hàng đợi 1, quy trình thực hiện với 8 đơn vị và nếu nó hoàn thành trong 8 đơn vị này hoặc nó cung cấp cho CPU cho hoạt động I / O trong đơn vị 8 đơn vị này thì mức độ ưu tiên của quá trình này không thay đổi và nếu vì một số lý do, nó lại đến trong hàng đợi sẵn sàng hơn là nó lại bắt đầu thực thi trong Hàng đợi 1.
3. Nếu một quy trình ở hàng đợi 1 không hoàn thành trong 8 đơn vị thì mức độ ưu tiên của nó sẽ bị giảm và nó được chuyển sang hàng đợi 2.
4. Điểm 2 và 3 ở trên cũng đúng với các quá trình ở hàng đợi 2 nhưng lượng tử thời gian là 16 đơn vị. Nói chung, nếu bất kỳ quá trình nào không hoàn thành trong một lượng tử thời gian nhất định thì nó sẽ được chuyển sang hàng đợi ưu tiên thấp hơn.
5. Sau đó trong hàng đợi cuối cùng, tất cả các quá trình được lên lịch theo cách thức FCFS.
6. Điều quan trọng cần lưu ý là một tiến trình nằm trong hàng đợi có mức ưu tiên thấp hơn chỉ có thể thực thi khi hàng đợi có mức ưu tiên cao hơn trống.
7. Bất kỳ quá trình nào đang chạy trong hàng đợi có mức độ ưu tiên thấp hơn đều có thể bị gián đoạn bởi một quá trình đến hàng hàng có mức độ ưu tiên cao hơn.

## Sự cần thiết của Lập lịch hàng đợi Phản hồi Đa cấp (MFQS)

Sau đây là một số điểm để hiểu sự cần thiết của việc lập lịch phức tạp như vậy:

* Lập lịch này linh hoạt hơn lập lịch hàng đợi Đa cấp.
* Thuật toán này giúp giảm thời gian phản hồi.
* Để tối ưu hóa thời gian quay vòng, thuật toán SJF là cần thiết, thuật toán này về cơ bản yêu cầu thời gian chạy của các quy trình để lập lịch trình cho chúng. Như chúng ta biết rằng thời gian chạy của các tiến trình không được biết trước. Ngoài ra, việc lập lịch trình này chủ yếu chạy một quá trình trong một lượng tử thời gian và sau đó, nó có thể thay đổi mức độ ưu tiên của quá trình nếu quá trình kéo dài. Do đó, thuật toán lập lịch này chủ yếu học từ hành vi trong quá khứ của các quy trình và sau đó nó có thể dự đoán hành vi trong tương lai của các quy trình. Bằng cách này, trước tiên, MFQS cố gắng chạy một quy trình ngắn hơn, điều này đổi lại dẫn đến việc tối ưu hóa thời gian quay vòng.

## Ưu điểm của MFQS

* Đây là một thuật toán lập lịch linh hoạt
* Thuật toán lập lịch này cho phép các quy trình khác nhau di chuyển giữa các hàng đợi khác nhau.
* Trong thuật toán này, một quá trình chờ đợi quá lâu trong hàng đợi có mức độ ưu tiên thấp hơn có thể được chuyển sang hàng hàng có mức độ ưu tiên cao hơn, giúp ngăn chặn nạn đói.

## Nhược điểm của MFQS

* Thuật toán này quá phức tạp.
* Khi các quy trình di chuyển xung quanh các hàng đợi khác nhau, dẫn đến việc tạo ra nhiều chi phí CPU hơn.
* Để chọn bộ lập lịch tốt nhất, thuật toán này yêu cầu một số phương tiện khác để chọn các giá trị

# PHẦN III: Lottery Scheduling

## i. Công cụ lập kế hoạch chia sẻ tỷ lệ

Công cụ lập lịch chia sẻ công bằng

Đảm bảo rằng mỗi công việc có được một tỷ lệ thời gian CPU nhất định.

Không được tối ưu hóa cho thời gian quay vòng hoặc phản hồi

## ii. Khái niệm cơ bản

Vé

Trình bày phần chia sẻ tài nguyên mà một quy trình sẽ nhận được

Phần trăm vé thể hiện phần chia sẻ tài nguyên hệ thống của nó trong câu hỏi.

Ví dụ

Có hai quá trình, A và B.

Quy trình A có 75 vé 🡪 nhận 75% CPU

Tiến trình B có 25 vé 🡪 nhận 25% CPU

## iii. Lập kế hoạch xổ số

Người lên lịch chọn một vé trúng thưởng.

Tải trạng thái của quá trình chiến thắng đó và chạy nó.

Example

Graphical user interface, application, Word

Description automatically generated

## iv. Vẻ đẹp của sự ngẫu nhiên (trong lập kế hoạch)

Dễ dàng giao dịch với các tình huống ở góc

Những người khác nên có nhiều "ifs" để ngăn chặn họ

Yêu cầu trạng thái nhỏ

Không cần theo dõi chi tiết của từng quy trình trong quá khứ

Thực sự nhanh

Tốc độ cao hơn à nhiều giả ngẫu nhiên hơn (hoặc hỗ trợ CTNH)

## v.Cơ chế đặt vé

1.Tiền vé

Một người dùng phân bổ vé giữa các công việc của họ bằng bất kỳ loại tiền tệ nào họ muốn.

Hệ thống chuyển đổi tiền tệ thành giá trị toàn cầu chính xác.

Ví dụ\

Có 200 vé (đơn vị tiền tệ toàn cầu)

Quy trình A có 100 vé

Quy trình B có 100 vé

Người dùng A 🡪 500 (đơn vị tiền tệ của A) đến A1 à 50 (đơn vị tiền tệ toàn cầu)

🡪500 (đơn vị tiền tệ của A) đến A2 à 50 (đơn vị tiền tệ toàn cầu)

Người dùng B 🡪10 (đơn vị tiền tệ của B) thành B1 🡪 100 (đơn vị tiền tệ toàn cầu)

2. Chuyển vé

Một quy trình có thể tạm thời chuyển giao vé của nó cho một quy trình khác.

3. Lạm phát vé

Một quy trình có thể tạm thời tăng hoặc giảm số lượng vé được sở hữu.

Nếu bất kỳ quá trình nào cần thêm thời gian CPU, nó có thể tăng vé của nó.

A picture containing diagram

Description automatically generated

## vi. Thực hiện

U: chỉ số về sự không công bằng

Thời gian công việc đầu tiên hoàn thành chia cho thời gian công việc thứ hai hoàn thành

Thí dụ:

Có hai công việc, mỗi công việc có thời gian chạy là 10.

Công việc đầu tiên kết thúc lúc 10

Công việc thứ hai kết thúc lúc 20

U=10/20=0.5

U sẽ gần bằng 1 khi cả hai công việc hoàn thành gần như cùng một lúc.

## vi.Nghiên cứu công bằng xổ số

Chart, line chart

Description automatically generated

Khi thời lượng công việc không dài lắm,

sự bất công trung bình có thể khá nghiêm trọng.

## vii. Lập lịch Stride (công cụ lập lịch Fair-share xác định)

Stride của mỗi quy trình

Được xác định là (một số lớn) / (số lượng vé của quá trình)

Ví dụ: một số lớn = 10.000

Quá trình A có 100 vé 🡪 bước tiến của A là 100

Quá trình B có 50 vé 🡪 bước tiến của B là 200

Tiến trình C có 250 vé 🡪 bước tiến của C là 40

Một tiến trình chạy, tăng một bộ đếm (= giá trị vượt qua) cho nó theo bước của nó.

Chọn quá trình để chạy có giá trị vượt qua thấp nhất

Table

Description automatically generated with medium confidence

Stride Scheduling Example

Table

Description automatically generated

Nếu công việc mới nhập với giá trị vượt qua 0,

Nó sẽ độc quyền CPU !:

Trình lập lịch ghi đè yêu cầu trạng thái toàn cục (trong

tương phản với xổ số)

## viii. Bộ lập lịch hoàn toàn công bằng cho Linux (CFS)

5% tổng thời gian CPU trung tâm dữ liệu bị lãng phí trong bộ lập lịch

Tập trung vào sự công bằng và giảm thiểu chi phí lập lịch trình

Theo dõi thời gian chạy (ảo) của mỗi quá trình

Tại thời gian lập lịch, hãy chọn quy trình có thời gian chạy ảo thấp nhất

Phần thời gian có thể thay đổi: theo số lượng các quy trình sẵn sàng chạy (từ Sched\_latency (48 ms) cho 1 quá trình đến min\_granularity (6 ms) cho bất kỳ số lượng quy trình nào

Chart, bar chart

Description automatically generated

## ix. Độ đẹp và trọng lượng

Thời gian và thời gian chạy ảo của quá trình có thể bị ảnh hưởng bởi trọng lượng (độ đẹp)

Text

Description automatically generated with low confidence

Thông thường, từ -20 đến 19

Theo mặc định, 0

A picture containing text, receipt

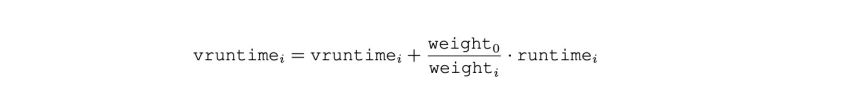
Description automatically generated

Thí dụ

Hai quá trình: A (độ tốt -5) B (độ tốt 0)

trọng lượngA = 3121, trọng lượngB = 1024

time\_sliceA = 3/4 (36 ms), time\_sliceB = 1/4 (12 ms) trong tổng số Sched\_latency (48 mili giây)



Ngoài ra, thời gian chạy ảo thay đổi theo trọng số A tích lũy thời gian chạy ảo ở 1/3 của B

## x. Bộ lập lịch hoàn toàn công bằng cho Linux (CFS)

Chi phí tối thiểu của bộ lập lịch

Cấu trúc dữ liệu phải có thể mở rộng: Không có danh sách

CFS sử dụng cây cân bằng (cây đỏ-đen) của các quy trình sẵn sàng chạy

O (log n) chèn và tìm kiếm

A picture containing different

Description automatically generated

Nhiều chi tiết khác

I / O được xử lý thay đổi thời gian hoạt động của quá trình đánh thức thành giá trị nhỏ nhất trong cây

Heuristics để lập lịch đa CPU (ái lực bộ nhớ cache, tần số, độ phức tạp của lõi…)

Lịch trình hợp tác đa quy trình