1.

a. I/O bound

CPU bound

**CPU bound** là các Process có tốc độ thực thi phụ thuộc vào tốc độ **CPU**. Một tác vụ thực hiện việc tính toán trên một tập nhỏ các số, ví dụ nhân 2 ma trận, tính số thập phân của số PI.

I/O bound

I/O bound là các Process có tốc độ thực thi phụ thuộc vào tốc độ hệ thống I/O. Ví dụ như tương tác với ổ đĩa (Disk) hay networking. Một chương trình đọc file là 1 I/O bound Process bởi vì nó bị bottleneck khi đọc dữ liệu từ Disk Thế nên người dùng sẽ ngay lập tức được cấp tài nguyên máy tính khi bấm một phím hay khi di chuyển chuột. Trong hệ thống chia sẻ thời gian (time-sharing systems), context switches được thực hiện rất nhanh, thế nên cảm giác như nhiều Process được thực thi một cách đồng thời trên cùng một CPU. Việc thực thi động thời nhiều Process được gọi là Concurrency.

Voluntary context switching

https://www.informit.com/articles/article.aspx?p=366888&seqNum=3

2.

a. FIFO

Với thuật toán FIFO thì cái nào đến trước sẽ được thực hiện trước. Tức là 1 tiến trình sẽ nắm giữ CPU cho đến khi nó thực hiện xong sẽ trả lại CPU. Thuật toán FIFO chiếm khá nhiều thời gian chờ, khi mà các tiến trình vào theo trình tự thời gian có độ chênh lệch khi xử lý lớn. Ngoài ra nó yêu cầu toàn quyền sử dụng CPU, điều này dễ gây nên sự sụp đổ của hệ thống nếu nạp quá nhiều tiến trình mà các tiến trình có thời gian burst lớn trong khi chạy.

b. STCF

Thuật toán STCF khi CPU được cấp thì nó sẽ ưu tiên tiến trình có burst time nhỏ nhất để xử lý. Thuật toán STCF sẽ cho ra thời gian chờ trung bình cho các tiến trình được xử lý một con số tối thiểu nhất. Khi di chuyển tiến trình ngắn hơn lên trước thì nó đã giảm một lượng thời gian chờ đáng kể cho các tiến trình ngắn, bằng cách đó nó giảm thời gian chờ trung bình so với các thuật toán khác. Tuy nhiên thuật toán STCF sẽ gặp khó khăn khi xác định độ dài tiếp theo khi cần đưa vào để CPU xử lý.

c. RR

Thuật toán RR sẽ cấp phát cho tiến trình một đơn vị thời gian chiếm dụng CPU nhất định còn được gọi là quantum. Khi một tiến trình sử dụng hết thời gian quantum mà vẫn chưa xử lý xong thì phần còn lại của nó sẽ được chuyển về phía sau trong danh sách hàng đợi. Dựa theo Ready list thì CPU sẽ lấy tiến trình tiếp theo để xử lý. Vì vậy nó có thể loại bỏ hiện tượng độc quyền CPU nhưng tính hiệu quả tùy thuôc vào lượng thời gian quantum mình chọn. Nếu quantum(q) quá lớn thì nó sẽ tương tự như thuật toán FIFO nếu q quá nhỏ thì chủ yếu để thực hiện context switching. Nó sẽ phù hợp với hệ thống tương tác với người dùng.

d. Lottery