Thuật toán lập lịch cho hệ thống thời gian thực

Đặng Quang Trung

Hệ Điều Hành

Ngày 30 tháng 7 năm 2017

Nội dung

Hệ thống thời gian thực

- Trong thế giới vật lý, mục đích của một hệ thống thời gian thực là có một thực hiện vật lý trong một khung thời gian đã chọn.
- Thông thường, một hệ thống thời gian thực bao gồm một hệ thống điều khiển(máy tính) và một hệ thống bị điều khiển (môi trường).
- Mỗi công việc xảy ra trong một hệ thống thời gian thực có một số thuộc tính thời gian. Các thuộc tính thời gian này cần được xem xét khi lập kế hoạch các nhiệm vụ trên một hệ thống thời gian thực.

  Release time (or ready time), Deadline, Minimum delay, Maximum delay, Worst case execution time, Run time, Weight (or priority).

Định nghĩa

- Một hệ thống chứa một tập các tác vụ:

$$T = \{\tau_1, \tau_2, \ldots, \tau_n\}$$

- Thời gian thực hiện của mỗi tác vụ là C_i với $\tau_i \in T$
- Hệ thống được cho là thời gia thực nếu có tồn tại ít nhất tác vụ $\tau_i \in T$, tác vụ rơi vào tình trạng:

Tình trạng tác vụ

1, Tác vụ τ_i là tác vụ hard real-time. Thời gian thực hiện tác vụ τ_i phải được hoàn thành bởi thời gian hết hạn $D_i (C_i \leq D_i)$.

2, Tác vụ τ_i là tác vụ **soft real-time**. Tác vụ cuối cùng τ_i kết thúc tính toán của nó sau gian gian hết hạn D_i, sẽ bị phạt nặng hơn. Hàm phạt $P(\tau_i)$ được định nghĩa cho tác vụ. Nếu $C_i \leq D_i$ hàm phạt $P(\tau_i)$ bằng 0 trái lại $P(\tau_i) > 0$ giá trị này tăng theo $C_i - D_i$.

3, Tác vụ là **firm real-time**. Tác vụ kết thúc công việc tính toán của nó sớm hơn thời hạn hết hạn D_i, sẽ nhận được thưởng. Hàm thưởng $R(\tau_i)$ được định nghĩa cho tác vụ. Nếu $C_i \geq D_i$, hàm $R(\tau_i)$ bằng 0, trái lại $R(\tau_i) > 0$ giá trị này tăng theo $D_i - C_i$.

Hàm thưởng và phạt của hệ thống

- Một tập các tác vụ thời gian thực $T = \{\tau_1, \tau_2, \ldots, \tau_3\}$ có thể là hỗn hợp của các tác vụ hard, soft, firm thời gian thực.
- T_s là tập tất cả các tác vụ **soft real-time** trong T, ví dụ $T_s = \{\tau_{s,1}, \tau_{s,2}, \ldots, \tau_{s,l}\}$ với $\tau_{s,i} \in T$. Hàm lỗi của hệ thống sẽ là $P(T)$

$$P(T) = \sum_{i-1}^{l} P(\tau_{s,i})$$

- T_f là tập tất cả các tác vụ **firm real-time** trong T, ví dụ $T_f = \{\tau_{f,1}, \tau_{f,2}, \ldots, \tau_{f,k}\}$ với $\tau_{f,i} \in T$. Hàm thưởng của hệ thống sẽ là $R(T)$

$$R(T) = \sum_{i-1}^{k} R(\tau_{f,i})$$

Giới thiệu

- Mục đích của lập lịch
  - Tối ưu hóa hiệu năng của hệ thống, thông lượng.
  - Làm thế nào? sắp xếp các tiến trình.
  - Thuật toán lập lịch Cố gắng để đạt được kết quả tốt nhất có thể (trung bình hoặc cho một bộ quy trình nhất định)
- Tiêu chí tối ưu
  - Thời gian đáp ứng, thời gian chờ đợi, thời gian đáp ứng của quá trình.
  - Sử dụng bộ vi xử lý.
  - Thông lượng (quy trình hoàn thành cho mỗi đơn vị thời gian).
  - Đáp ứng các thời hạn: tính khả thi, khả năng lập lịch

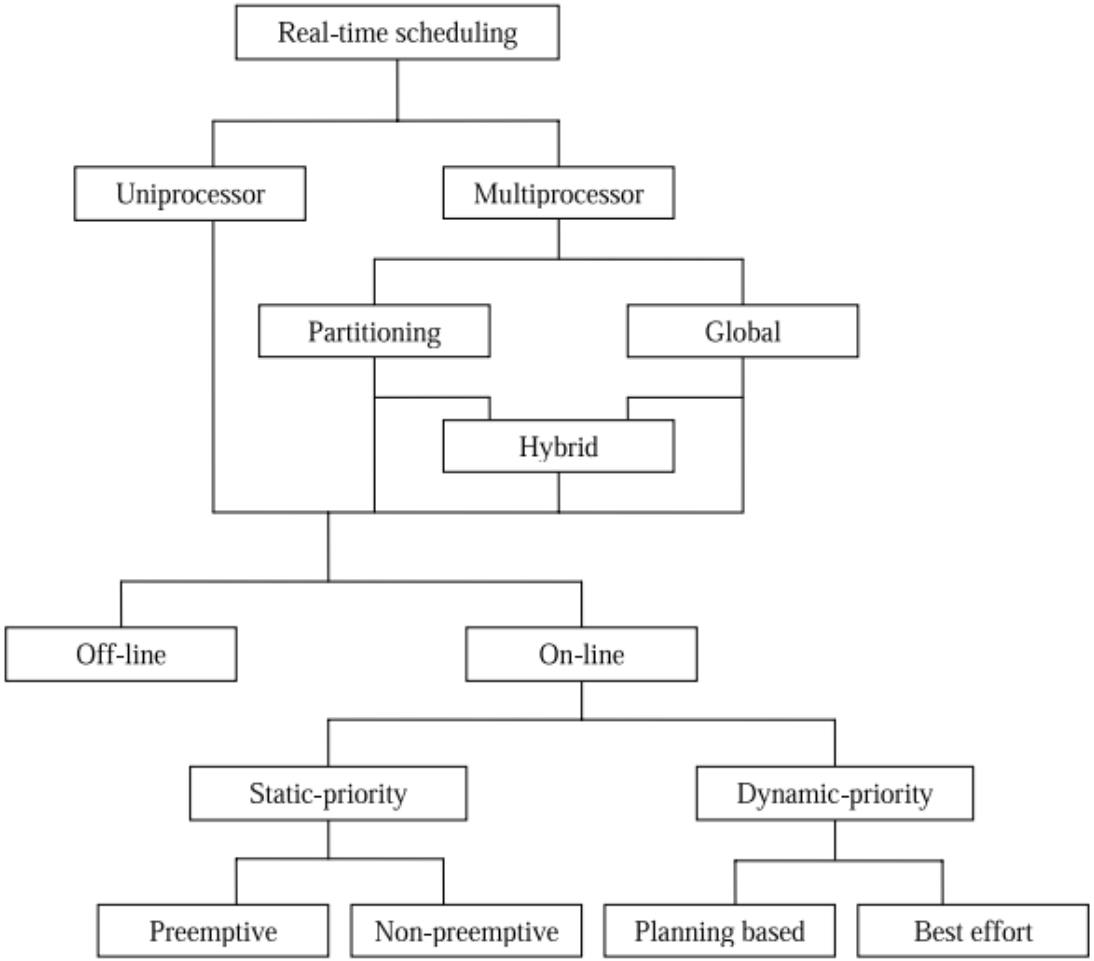
Phân lớp thuật toán lập lịch

- Static
  - Tất cả các thông số về độ ưu tiên được biết trước.
  - Lập lịch có thể được hoàn thành trong thời gian thiết kế, do đó hiệu quả cao trong thời gian chạy.
  - Áp dụng cho các hệ thống nhúng đóng.
  -
- Dynamic
  -
  - Không phải tất cả các tham số đều được biết trước
  - Lập lịch trực tuyến.
  - Mục tiêu tối ưu thường có thể đạt được chỉ khoảng.
  - Lập lịch yêu cầu hiệu năng tính toán.
  - Hoạt động hệ điều hành thông thường cho hệ thống tương tác

Phân lớp thuật toán lập lịch (2)

- Lập lịch không trưng dụng.
  - Hệ điều hành không thu hồi bộ xử lý từ tiến trình đang chạy, nó chạy cho đến khi nó kết thúc hoặc trả lại bộ xử lý.
  - Ưu điểm: đơn giản thực hiện thay đổi quy trình.
  - Nhược điểm: không hữu ích cho các hệ thống thời gian thực, nơi mà phản ứng nhanh với các sự kiện bên ngoài là cần thiết
- Lập lịch trưng dụng
  - Hệ điều hành sẽ thu hồi bộ xử lý của tiến trình đang chạy bất cứ khi nào có nhu cầu lập lịch lại.

Sở đồ cấu trúc phân lớp tổng quát



Hình: Sơ đồ lập lịch

Lập lịch hệ thống thời gian thực

- Earliest Deadline First(EDF)
- Rate-monnotonic (RM)

Earliest Deadline First(EDF)

- Lập lịch dựa trên thời gian hết hạn
  - Chỉ sử dụng thời gian hết hạn của tiến trình lên kế hoạch, không dungf thời gian tính toán.
  - Bộ xử lý được gán cho tác vụ với thời gian hết hạn gần nhất(thời gian hết hạn sớm nhất, EDF).
  - Không trưng dụng: Lập lịch lại khi tác vụ hiện tại thực hiện xong.
  - Trưng dụng: Lập lịch lại khi có 1 tiến trình mới sẵn sàng.

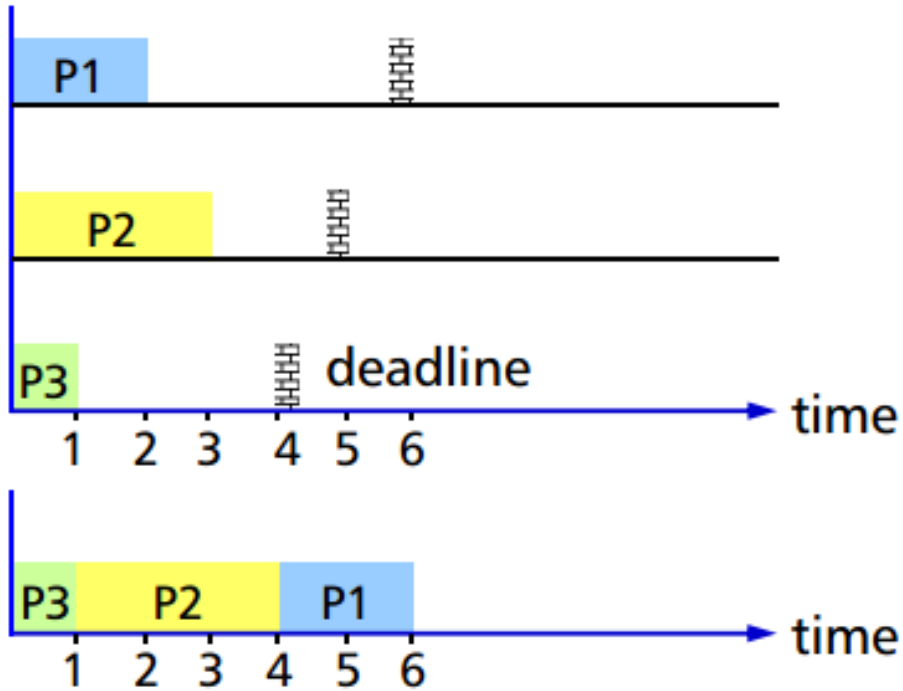
Earliest Deadline First

- Static EDF cùng thời gian sẵn sàng.
  - Sắp xếp các tiến trình với thời gian hết hạn tăng dần.
  - Bất cứ khi nào một tiến trình được lập lịch, kiểm tra xem thời hạn của nó có được đáp ứng hay bị vi phạm.
  - Cho k tiến trình nó thỏa mãn.

$$\sum_{i=1}^{k} C_i \leq D_k \quad k = 1 \ldots n$$

Earliest Deadline First

- Ví dụ:



- EDF là tối ưu thời gian sẵn sàng giống nhau.

Earliest Deadline First

- Chứng minh
  - Bộ xử lý luôn được sử dụng vì $R_i = 0$.
  - Nếu lập lịch đã được tiến hành đúng thời điểm trước khi thêm P_k, sau đó sẽ không thêm được bất kì gì nữa:

$$\sum_{i=1}^{k} C_i > D_k$$

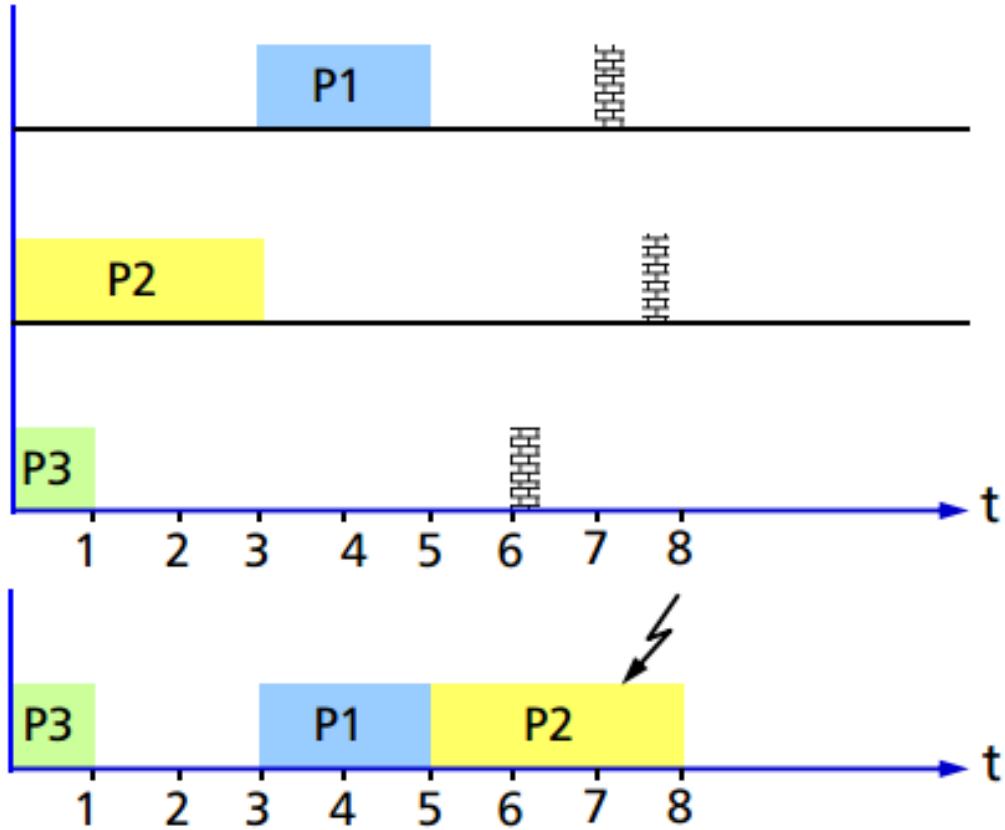
  - Khả năng duy nhất là sau đó trao đổi P_k với một trong các tiến trình đã được lập lịch P_j (j < k).
  - Tổng thời gian thực hiện vẫn không thay đổi.
  - Chúng ta biết rằng $D_j \leq D_k$ và do đó:

$$\sum_{i=1}^{k} C_i > D_j$$

  - Do đó sự trao đổi làm cho mọi thứ tồi tệ hơn.
  - Nếu thuật toán không tìm được giải pháp nào khác.

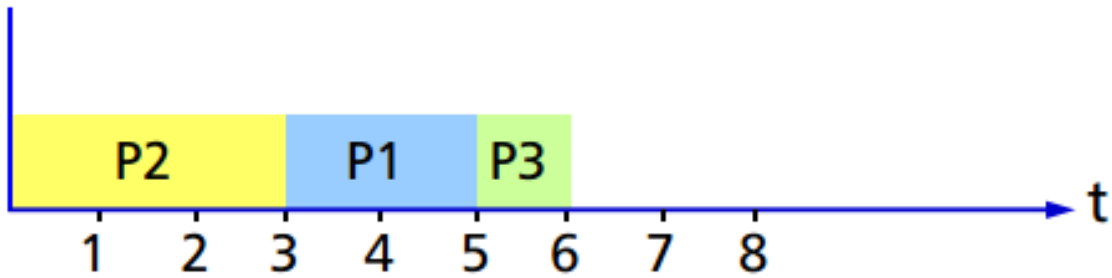
Earliest Deadline First

- Static EDF với thời gian sẵn sàng khác nhau
  - Không trưng dụng là không tối ưu

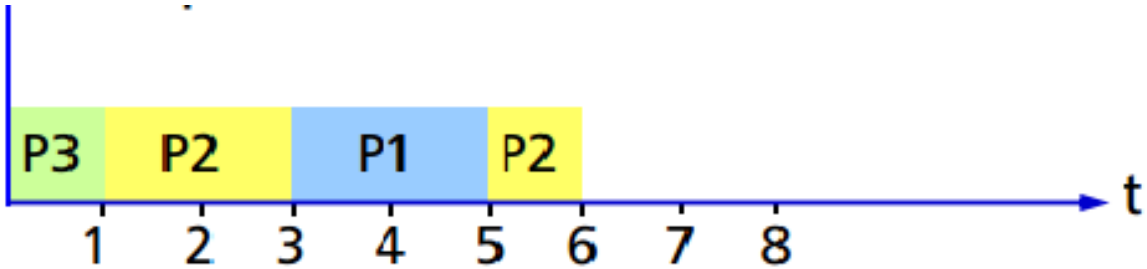


Earliest Deadline First

- Có lập lich tối ưu khác ví du:



- Thuật toán EDF sẽ tối ưu cho trường hợp này nếu có trưng dụng
- Ví dụ:




Thank You!

Tài liệu tham khảo