# سیستم‌های عامل - دکتر ابراهیمی‌مقدم

امیرحسین منصوری - ۹۹۲۴۳۰۶۹

تمرین سری پنجم

## سوال ۱

### الف)

**نادرست؛** در یک سیستم با $n$ پروسه، بدترین زمان پاسخگویی در RR برای هر پروسه، برابر $(n-1)q$ است. در یک سناریوی فرضی، اگر $n = 3$ پروسه داشته باشیم که هر کدام بخواهند ۳ ثانیه اجرا شوند، در SJF زمان پاسخگویی برابر $3s = \frac{0+3s+6s}{3}$ خواهد بود. در حالی که اگر از RR با $q = 10ms$ استفاده کنیم، بیشترین میانگین زمان ممکن برابر $(n-1)q = 20ms$ خواهد بود که بسیار کمتر از معادل آن در SJF است.

### ب)

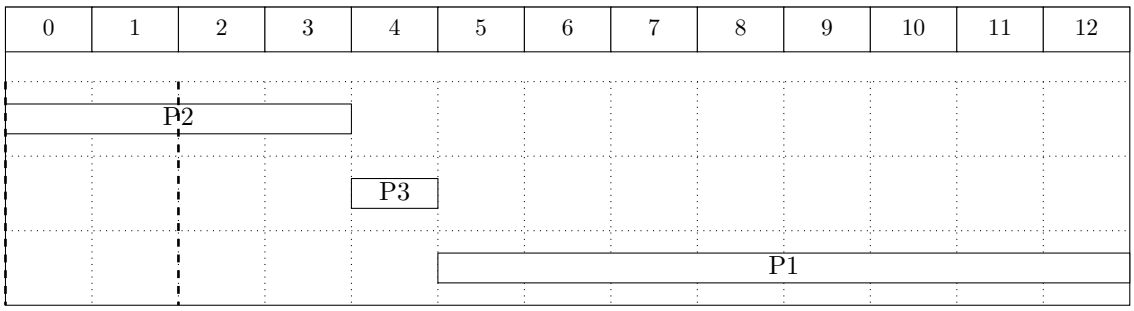
**درست...؛** از نظر متوسط زمان پاسخگویی پروسه‌ها، این جمله درست است. اما معمولا FIFO سربار کمتری نسبت به SJF دارد و از این لحاظ کمی سریع‌تر است.

### ج)

**نادرست؛** در این حالت، زمان‌بند برخی پروسه‌ها را از حافظه اصلی خارج و در Virtual memory ذخیره می‌کند (اصطلاحاً Swap out می‌کند) تا جا برای پروسه‌های دیگر در حافظه اصلی باز شود.

## سوال ۲

### SJF

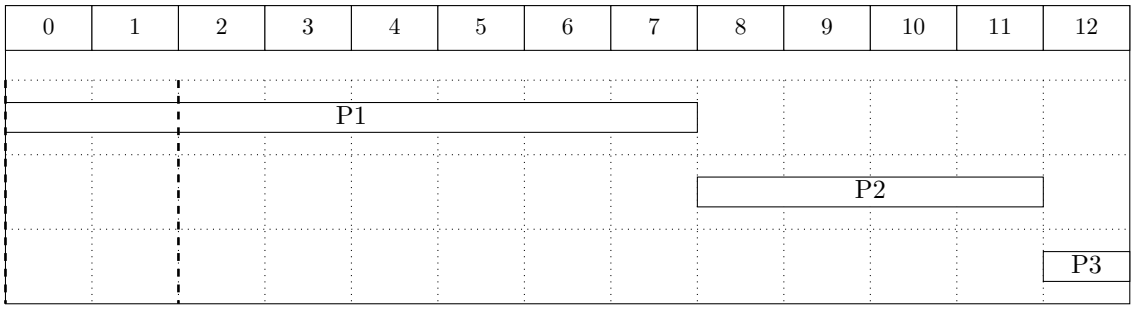



در نمودار بالا، ابتدا P۱ و P۲ با هم وارد می‌شوند. چون Burst time پروسهٔ P۲ کمتر است، این پروسه برای اجرا انتخاب می‌شود. پس از اتمام اجرا، P۳ و P۱ در سیستم هستند و چون P۳ زمان Burst time کمتری دارد، برای اجرا انتخاب می‌شود. در نهایت P۱ نیز اجرا می‌شود.

با توجه به شکل:

$$\text{Response Time for P۱} = 5$$
$$\text{Response Time for P۲} = 0$$
$$\text{Response Time for P۳} = 4$$
$$\Rightarrow \text{Average Response Time} = \frac{5+0+4}{3} = 3$$

**FIFO (یا FCFS)**

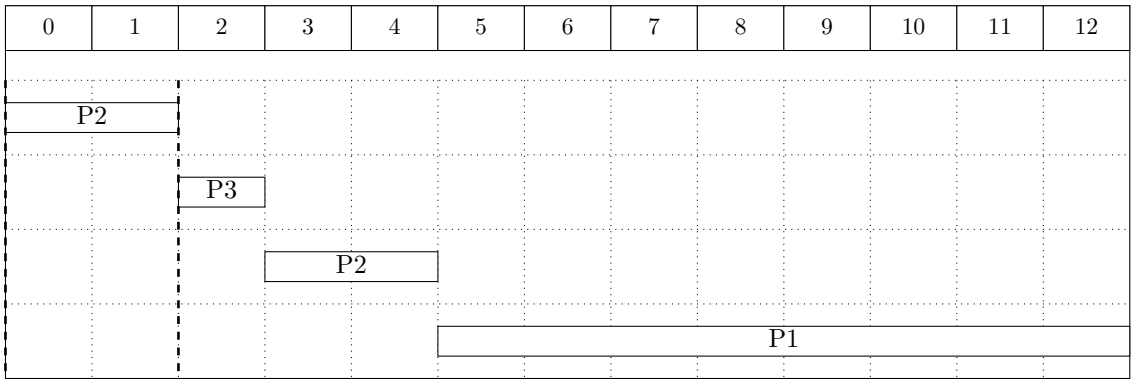



P1 and P2 Enter P3 Enters

بین پروسه‌های P۱ و P۲ که همزمان وارد می‌شوند، ابتدا P۱ که اولویت بیشتری دارد اجرا می‌شود و سپس P۲ اجرا می‌شود. در نهایت P۳ که از همه دیرتر وارد شده، اجرا می‌شود. داریم:

$$\text{Response Time for P۱} = 0$$
$$\text{Response Time for P۲} = 8$$
$$\text{Response Time for P۳} = 12$$
$$\Rightarrow \text{Average Response Time} = \frac{0+8+12}{3} = \frac{20}{3} \approx 6.67$$

**SRTF**




P1 and P2 Enter P3 Enters

از بین پروسه‌های P۱ و P۲، پروسهٔ P۲ که زمان باقی‌ماندهٔ کوتاه‌تری دارد (برابر ۴)، وارد می‌شود. در زمان $t+2$، پروسهٔ P۳ وارد می‌شود و چون کوتاه‌ترین زمان باقی‌مانده را دارد (برابر ۱)، جایگزین پروسهٔ قبلی می‌شود و اجرا می‌شود. بعد از پایان این پروسه، پروسهٔ P۲ که ۲ واحد زمانی نیاز دارد تا به اتمام برسد (در مقابل P۱ که ۸ واحد زمانی نیاز دارد)، به اجرا شدن ادامه می‌دهد. بعد از اتمام این پروسه، P۱ در نهایت اجرا می‌شود. داریم:

$$\text{Response Time for P۱} = 5$$
$$\text{Response Time for P۲} = 0$$
$$\text{Response Time for P۳} = 2$$
$$\Rightarrow \text{Average Response Time} = \frac{5+0+2}{3} = \frac{7}{3} \approx 2.33$$

## سوال ۳

### الف)

همهٔ پروسه‌های وارد شده در سیستم، فارغ از وضعیتشان، اولویت خود را به مرور از دست می‌دهند. اما چون $b < a$، پروسه‌های در وضعیت Running اولویت خود را سریع‌تر از پروسه‌های Ready از دست می‌دهند. این باعث می‌شود که یک پروسهٔ Running، بعد از مدتی جای خود را با یک پروسهٔ Ready عوض کند. همچنین چون اولویت یک پروسهٔ جدید برابر صفر است، از همه پروسه‌های موجود اولویت بیشتری خواهد داشت و بنابراین به هنگام ورود، بلافاصله اجرا می‌شود. در دو حالت می‌توان الگوریتم حاصل را بررسی کرد:

- **همه پروسه‌ها با هم وارد شوند:** در این حالت، الگوریتم دقیقا مثل RR عمل خواهد کرد که Time quantum کوچکی دارد.
- **پروسه‌ها در زمان‌های مختلف وارد شوند:** در این حالت، اولویت با پروسه‌ای است که کمترین CPU Time تا به حال به آن اختصاص داده شده؛ مثلا پروسهٔ جدیدی که تا به حال اجرا نشده، بیشترین اولویت را پیدا می‌کند و بلافاصله شروع به اجرا می‌کند. اگر برای مدت طولانی پروسهٔ جدیدی وارد نشود، بعد از مدتی اولویت پروسه‌ها نزدیک به هم می‌شود و الگوریتم دوباره مثل RR عمل می‌کند.

### ب)

چون همواره اولویت پروسه‌های موجود در حال کم شدن است، پروسه‌ای که تازه وارد می‌شود، همیشه بیشترین اولویت را خواهد داشت و بلافاصله جایگزین پروسهٔ فعلی در حال اجرا می‌شود. همچنین پروسه‌های در وضعیت Running کندتر از پروسه‌های در وضعیت Ready اولویت خود را از دست می‌دهند. بنابراین یک پروسهٔ در وضعیت Running، همیشه از همهٔ پروسه‌های Ready موجود اولویت بیشتری خواهد داشت و بنابراین تا اتمام پروسه، در وضعیت Running باقی خواهد ماند. همچنین چون پروسه‌ها در ابتدا اولویت صفر دارند، هرچه کمتر در صف Ready بوده باشند، یا به عبارتی *دیرتر* وارد شده باشند، اولویت بیشتری دارند.

در نهایت می‌توان این یک الگوریتم Pre-emptive LIFO است؛ که یعنی بیشترین اولویت را به پروسه‌های جدید می‌دهد (Last In، First Out) و در صورت وارد شدن یک پروسه جدید، بلافاصله پردازنده را به آن اختصاص می‌دهد (Pre-emptive).

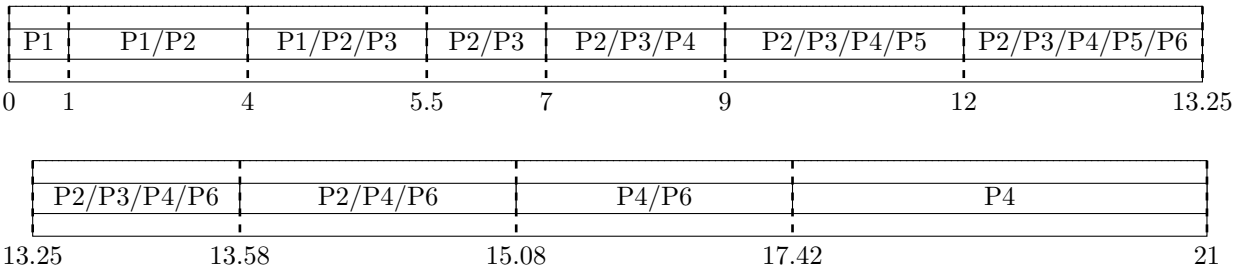
### ج)

چون $b > 0$، اولویت پروسه‌های در وضعیت Running همواره در حال افزایش، و چون $a < 0$، اولویت پروسه‌های در وضعیت Ready همواره در حال کاهش است. بنابراین یک پروسهٔ در وضعیت Running، از هر پروسهٔ دیگر (چه جدید باشد، چه Ready) اولویت بیشتری دارد و بنابراین هیچ‌گاه اجرای آن متوقف نمی‌شود (دچار Pre-emption نمی‌شویم). همچنین مانند قسمت قبل، بین پروسه‌های Ready، آن که کمتر در صف بوده باشد و به عبارتی جدیدتر آمده باشد، اولویت بیشتری دارد.

در نهایت می‌توان گفت این یک الگوریتم Non-preemptive LIFO است؛ چون مانند قسمت قبل، بیشترین اولویت را به پروسه‌های جدیدتر می‌دهد؛ اما در عمل هیچ‌گاه Pre-emption در آن رخ نمی‌دهد و یک پروسهٔ در حال اجرا، تا اتمام کارش پردازنده را در اختیار خواهد داشت.

## سوال ۴

نمودار Gantt chart پروسه‌ها به صورت زیر است. به علت طولانی بودن محاسبات عددی مربوط به آن، از آوردن این محاسبات صرف‌نظر کرده‌ایم!




برای به دست آوردن نمودار بالا، کافی است محاسبه کنیم در زمان وارد شدن هر پروسه، چند پروسه همزمان داریم؛ و این پروسه‌های همزمان تا چه زمانی با هم اجرا می‌شوند، یا کدام‌یک از آن‌ها در چه زمانی به اتمام می‌رسند. به طور کلی اگر $n$ پروسه همزمان در سیستم داشته باشیم، و کمترین زمان باقی‌مانده بین این $n$ پروسه برابر $t$ باشد، این پروسه‌ها تا $n \times t$ زمان بعد با هم اجرا می‌شوند؛ و درست بعد از این زمان پروسه با کمترین زمان باقی‌مانده به اتمام می‌رسد و $n-1$ پروسه باقی می‌مانند. همچنین اگر $n$ پروسه با هم در سیستم باشند، و زمان $q$ بگذرد، از زمان باقی‌مانده هر کدام از پروسه‌ها $\frac{q}{n}$ کم می‌شود. با استفاده از این روابط، اعداد نمودار بالا قابل محاسبه است.

زمان انتظار هر پروسه نیز برابر

$$(\text{Finish time} - \text{Arrival time}) - \text{Burst time}$$

خواهد بود. بنابراین:

$$\begin{aligned}
W_{P_1} &= (5.5 - 0) - 3 = 2.5 \\
W_{P_2} &= (15.08 - 1) - 5 = 9.08 \\
W_{P_3} &= (13.58 - 4) - 3 = 6.58 \\
W_{P_4} &= (21 - 7) - 7 = 7 \\
W_{P_5} &= (13.25 - 9) - 1 = 3.25 \\
W_{P_6} &= (17.42 - 12) - 2 = 3.42
\end{aligned}$$

و میانگین این زمان‌ها برابر با ۵/۳۰۵ دقیقه خواهد بود.

## سوال ۵

### الف)

برای این که مدیریت این پروسه‌ها ممکن باشد، باید داشته باشیم $\sum_{i=1}^{m} \frac{c_i}{p_i} \leq 1$. بنابراین

$$\begin{aligned}
\frac{x}{100} + \frac{40}{200} + \frac{100}{500} &\leq 1 \\
\Rightarrow x &\leq 60
\end{aligned}$$

### ب)

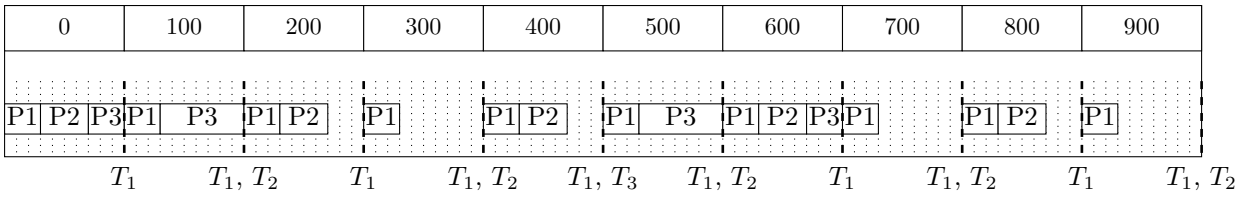
برای این که مدیریت این پروسه‌ها با الگوریتم RMS ممکن باشد، باید داشته باشیم $\sum_{i=1}^{m} \frac{c_i}{p_i} \leq m(2^{\frac{1}{m}} - 1)$. بنابراین

$$\begin{aligned}
\frac{x}{100} + \frac{40}{200} + \frac{100}{500} &\leq 3(2^{\frac{1}{3}} - 1) \approx 0.78 \\
\Rightarrow x &\leq 37.97
\end{aligned}$$

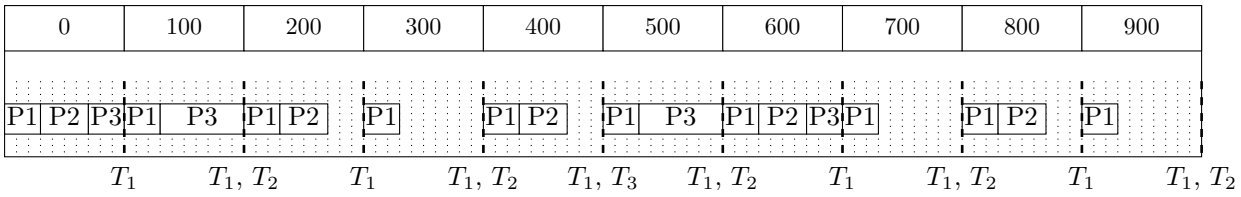
### ج)

در هر دو مثال، فرض می‌کنیم $x = 30$.

**RMS**




**EDF**




نمودار مربوط به الگوریتم EDF، دقیقا مانند RMS به دست می‌آید.