# پاسخ‌نامه تمرین نهم درس سیستم‌های عامل

دکتر زرندی

پاییز ۱۴۰۰

۱- در خصوص مانیتور به سوالات زیر پاسخ دهید.

الف) در یک مانیتور ممکن است چند فرایند توسط یک متغیر *condition* دچار انتظار شوند و پس از آنکه *signal* روی آن *condition* فراخوانی می‌شود، تنها یک فرایند باید از انتظار خارج شود. یکی از راه‌های انتخاب یک فرایند از میان فرایندهای منتظر استفاده از انتظار مشروط (*conditional waiting*) است. در این روش همراه با فراخوانی *wait* بر روی یک *condition*، یک ورودی از جنس *integer* نیز به عنوان ورودی داده می‌شود. درباره ورودی *c* و تاثیر آن بر انتخاب فرایند منتظر توضیح دهید.

ب) سمافور باینری را با استفاده از مانیتور پیاده‌سازی کنید.

```
Condition x;
x.wait(c);
```

الف) مقدار c که شماره اولویت (priority number) نامیده می‌شود، همراه با نام فرآیندی که متوقف شده است ذخیره می‌شود. هنگامی که تابع signal فراخوانی شود فرآیندی که کمترین شماره اولویت را دارد به اجرای خود ادامه خواهد داد.

ب)

```
monitor SemaphoreSimulation {
    boolean busy = FALSE;
    condition notbusy;
    public:
    wait () {
        if (busy) notbusy.wait ();
        busy = TRUE;
    }
    signal () {
        busy = FALSE;
        notbusy.signal ();
    }
}
```

۲- فرض کنید در هر ۱۰ ثانیه یک فرآیند با مدت اجرای ۲ ثانیه به صف اجرا اضافه می‌شود. همچنین فرآیندهایی با مدت اجرای ۰.۵ ثانیه نیز در هر ۲ ثانیه ساخته می‌شوند.

لف) نمودار گانت اجرای فرآیندها با زمان‌بندی *FCFS* را برای مدت ۲۰ ثانیه رسم کرده و میانگین زمان انتظار فرآیندها را به دست آورید.

ب) آیا در اینجا اثر کاروان (*convoy effect*) رخ داده است؟ توضیح دهید.

ج) در صورتی که زمان اضافه شدن فرآیندها تغییر نکند، چگونه می‌تواند میانگین زمان انتظار را بهبود داد؟

الف)


Average waiting time = (0 + 4 + 2.5 + 1 + 0 + 0 + 0 + 4 + 2.5 + 1 + 0 + 0) / 12 = 1.25

ب) بله رخ داده است. زیرا فرآیندهای کوتاه‌تر P2، P3 و P4 منتظر فرآیند طولانی‌تر P1 و فرآیندهای کوتاه‌تر P8، P9 و P10 منتظر فرآیند طولانی‌تر P7 می‌باشند.

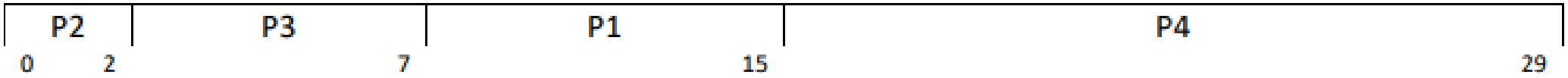
ج) اگر هنگامی که فرآیندهای با مدت اجرای کمتر (مانند فرآیندهای I/O-bound) به صف اجرا اضافه می‌شوند، در اجرای فرآیندهای طولانی (CPU-bound) وقفه ایجاد کنیم و به فرآیندهای دیگر اجازه اجرا دهیم احتمالا میانگین انتظار کاهش خواهد یافت.

۳- با توجه به جدول زیر به سوالات پاسخ دهید.

الف) با استفاده از روش اول کوتاه‌ترین کار (*shortest job first*) نمودار گانت برنامه‌ریزی پردازنده برای انجام فرایندها را رسم کرده و میانگین زمان تاخیر را بدست اورید.

| فرایند | زمان اجرا (ms) |
|---|---|
| P1 | 8 |
| P2 | 2 |
| P3 | 5 |
| P4 | 14 |

الف)




میانگین زمان تاخیر:

$$(2 + 7 + 15 + 0)/4 \ = \ 6ms$$

ب) فرض کنید پیش‌بینی زمان اجرای فرایند آخر توسط پردازنده با پارامتر $\alpha = 0.5$ برابر $3ms$ بدست آمده باشد. در این صورت زمان اجرای فرایند بعدی را پیش‌بینی کرده و در مورد مناسب بودن مقدار پارامتر $\alpha$ اظهار نظر کنید.

ب)

$$\tau_{n+1} = \alpha\, t_n + (1-\alpha)\, \tau_n$$

$$\tau_n = 3ms\ , \qquad t_n = 14ms\ , \qquad \alpha = 0.5$$

پس مقدار $\tau_{n+1}$ برابر 8.5ms بدست می آید.

با توجه به اینکه 3 از مقدار 14 دور است و پیش بینی دقیقی نداشته‌ایم، بهتر است مقدار $\alpha$ را به 1 نزدیک کنیم تا $t_n$ سهم بیشتری در پیش بینی ما داشته باشند و پیش بینی دقیق تری داشته باشیم.

(امتیازی) ۴- در مورد معماری حافظه ی تراکنشی (*transactional memory*) تحقیق کرده و به سوالات زیر پاسخ دهید.
الف) در مورد ساختار این معماری توضیح دهید.
ب) چرا به این نوع معماری *Lock-free* گفته میشود؟
ج)این نوع معماری کدامیک از مشکلات تکنیک های قفل دیگر را حل کرده است؟

الف) حافظه تراکنشی یک انتزاع برنامه‌نویسی سطح بالا (high-level programming abstraction) فراهم می‌کند که اجازه اجرای تراکنش‌ها را می‌دهد. تراکنش به یک دنباله متناهی از دستورات می‌گویند که توسط یک فرآیند اجرا شده و خصوصیات زیر را داشته باشد:

- **توالی‌پذیری (serializability)**: تراکنش‌ها باید به صورت متوالی اجرا شوند. یعنی دستورات یک تراکنش نباید با دستورات تراکنشی دیگر تداخل پیدا کند. همچنین پردازنده‌های مختلف نباید ترتیب اجرای متفاوتی از تراکنش‌ها را ببینند.

- **تجزیه‌ناپذیری (atomicity)**: هر تراکنش یک دنباله از تغییرات غیرقطعی را در حافظه مشترک اعمال می‌کند. هنگامی که تراکنش تکمیل شد، یا این تغییرات تثبیت (commit) شده و برای پردازنده‌های دیگر قابل مشاهده می‌گردند یا اینکه تراکنش بی‌نتیجه مانده (abort) و تغییرات در نظر گرفته نمی‌شوند.

ب) زیرا برای انجام عملیات در آن نیازی به ممانعت متقابل (mutual exclusion) نیست. به عبارت دیگر اگر یک فرآیند در میانه اجرا دچار وقفه شود، فرآیندهای دیگر از انجام عملیات بر روی آن شیء منع نمی‌شوند بلکه تنها هنگامی که تداخلی (conflict) به وجود آید، تراکنش بازگشت (revert) می‌کند.

ج) وارونگی اولویت (priority inversion)، کاروان قفل(lock convoy) و بن‌بست (deadlock)