Department of
Computer Engineering

Homework 4-Solution

Operating Systems

Fall 2023

Dr. Javadi

پاسخ سوال یک:

دستورات هر پردازه را مطابق زیر نامگذاری می کنیم:

P_1:
```
while(true) {
/*A1*/  wait(s1);
/*A2*/  print("A");
/*A3*/  signal(s2);
/*A4*/  print("B");
}
```

P_2:
```
while(true) {
/*B1*/  wait(s2);
/*B2*/  print("B");
/*B3*/  signal(s3);
/*B4*/   print("C");
}
```

P_3:
```
while(true) {
/*C1*/  signal(s3);
/*C2*/  print("C");
/*C3*/  wait(s1);
/*C4*/   print("A");
}
```

P_4:
```
while(true) {
/*D1*/  wait(s3);
/*D2*/  print("A");
/*D3*/  signal(s1);
/*D4*/  print("B");
/*D5*/  wait(s2);
/*D6*/  print("C");
}
```

الف)ممکن است،یک نمونه توالی اجرا دستورات:

C1⟶D1⟶D2⟶D3⟶A1⟶A2⟶C2⟶B1⟶B2⟶D4

ب)ممکن است،یک نمونه توالی اجرا دستورات:

B1⟶B2⟶B3⟶D1⟶D2⟶D3⟶D4⟶A1⟶A2⟶A3⟶A4

پ)ممکن نیست،یک توالی ممکن برای چاپ سه کاراکتر اول به صورت زیر است:

C1⟶C2⟶D1⟶D2⟶D3⟶D4

پس از آن مقدار s1 برابر یک خواهد بود،برای چاپ دو کاراکتر A لازم است که دو دستور A2 و C4 اجرا شوند قبل از آنکه یک کاراکتر دیگر چاپ شود.قبل از هر دو wait(s1) صدا زده می شود پس تنها یکی از A قابل چاپ است و رشته نهایی خواسته شده غیر قابل چاپ شدن است.

ت)ممکن است،یک نمونه توالی اجرا دستورات:

C1⟶C2⟶B1⟶B2⟶D1⟶D2⟶D3⟶A1⟶A2⟶B3⟶B4

پاسخ سوال دو:

با بررسی کد ها و ترتیب اجرا های مختلف نتیجه می شود می توان رشته های قابل چاپ را به دسته های زیر تقسیم کنیم و هر کدام را جداگانه شمارش نماییم:

i)رشته هایی که 4 کاراکتر اول و 4 کاراکتر دوم آنها، هر کدام از دو تا A و دو تا B تشکیل شده اند و دو کاراکتر آخر محدودیتی ندارند، برای تمام رشته های مذکور ترتیبی برای چاپ آنها موجود است (به این ترتیب که برای 4 کاراکترهای یاد شده باید دستورات درون حلقه هردو پردازه ها یک بار اجرا شوند سپس دستورات یک پردازه برای بار دوم) تعداد آنها:

$$\binom{4}{2} \times \binom{4}{2} \times 2^2 = 144$$

ii)دسته دوم آنها که 4 کاراکتر اول آنها متشکل از دو A و دو B است،پس از آن 4 کاراکتر بعدی شامل سه A و یک B یا برعکس است (که به علت تناظر ما یکی را در شمارش در نظر می گیریم و پاسخ را در دو ضرب می کنیم) در این صورت اگر این 4 کاراکتر به عنوان مثال AABA باشند کاراکتر بعدی باید حتما B باشد و برای کاراکتر آخر محدودیتی وجود ندارد،تعداد رشته های این دسته:

$$2 \times \binom{4}{2} \times \binom{4}{1} \times 2 = 96$$

iii)دسته آخر آنهایی که 4 کاراکتر اول آنها از سه A و یک B تشکیل شده است یا بلعکس،مثلا BAAA که نتیجتا کاراکتر پنجم باید B باشد،حال برای ادامه آن دو حالت زیر را داریم:

الف)چهار کاراکتر بعدی شامل دو تا A و دو تا B باشد و کاراکتر آخر انتخاب آزاد است:

$$2 \times \binom{4}{1} \times \binom{4}{2} \times 2 = 96$$

ب)چهار کارکتر بعدی شامل 3 تا B و یک A باشد،در این حالت نیز کاراکتر آخر آزاد است:

$$2 \times \binom{4}{1} \times \binom{4}{1} \times 2 = 64$$

بنابراین تعداد کل حالات برابر است با 400 = 64+ 96 + 96 + 144

پاسخ سوال سه:

الف)

بن بست نداریم .

منبع پایین را برابر با R0 و منبع بالا سمت چپ را R1 و منبع بالا سمت راست را R2 در نظر میگیریم .

ابتدا میتوان به پردازه P2 اجازه داد تا منبع مورد نیاز خود را از R0 را بگیرد و با اتمام کار خود منابع گرفته شده از R0 و R2 را پس بدهد .

سپس پردازه P0 منابعی که اکنون از R0 و R2 آزاد شده را در اختیار میگیرد و پس از اتمام کار خود، منابع را پس میدهد . با آزاد شدن منبع R1 ، اکنون پردازه P1 میتواند منبع R1 را در اختیار بگیرد و منابع R1 و R2 را بعد از اتمام کار آزاد کند.

حال پردازه P3 میتواند منابع مورد نیاز خود ازR2 را گرفته و کار خود را تمام کند.

دنباله: P2->P0->P1->P3

ب)

بن بست داریم .

دلیل آن هم این است که نمیتوانیم تحت هیچ شرایطی تقاضاهای هیچ کدام از پردازه‌ها را برطرف کنیم که پردازه بعد از اتمام کار خود منابع خود را آزاد کنند .

ج)

بن بست نداریم .

زیرا پردازه P5 بعد از اتمام کار خود منبعی که از R4 در اختیار داشت را آزاد میکند و سپس p4 که درخواست آن منبع را داشت آنرا در اختیار میگیرد و بعد از اتمام کار خود منبعی که از R4 , R2 داشت را آزاد میکند سپس p2 میتواند یک منبع R2 را که درخواست داشت در اختیار بگیرد و پس از اتمام کار خود منابعی که از R1,R2 داشت را آزاد میکند و سپس p1 میتواند منبعی که از R1 درخواست داشت را بگیرد و پس از اتمام کار منابع R1,R3 را آزاد کند و در آخر p3 منبعی که از R3 میخواست را در اختیار میگیرد و سپس بعد از اتمام کار خود همه منابع در اختیارش را ازاد میکند.

دنباله: P5->P4->P2->P1->P3

پاسخ سوال چهار:

الف)

	Need A B C D
P0	0 0 0 0
P1	0 7 5 0
P2	1 0 0 2
P3	0 0 2 0
P4	0 6 4 2

ب)

بله. هر یک از پردازه های P0 و P3 می توانند اجرا شوند. در صورت اجرا شدن پردازه P3 و آزادسازی منابع، دیگر پردازه ها می توانند اجرا شوند.

ج)

بله می توان. در نتیجه پاسخ به این درخواست، مقدار ماتریس available برابر خواهد با:

(0، 0، 1، 1)

دنباله پردازه ها نیز به صورت:

(P0, P2, P3, P1, P4)

خواهد بود.

پاسخ سوال پنج:

الف:

در الگوریتم first-fit چون حافظه های داده شده به ترتیب پر می شوند در نتیجه هر پردازه نسبت به مقدار حافظه داده شده مقایسه شده و اگر مقدار حافظه ای که پردازه در یک پارتیشن قرار میگرد از حافظه پارتیشن بیشتر باشد، پردازه در آن پارتیشن قرار میگیرد.

به این ترتیب نحوه قرار گیری پردازه ها در پارتیشن های داده شده به این شکل خواهد بود:

112K -> 500K partition

417K -> 600K partition

212K -> 388K partition (new partition : 388K = 500KB - 112KB)

426K -> MUST WAIT

بدیهی است که پردازه های 112KBو 417KB در پارتیشن های 500K و 600K قرار میگیرند. اما برای پردازه 212KB از انجایی که مقدار ان از 100 بیشتر است در پارتیشن 100K قرار نمیگیرد. در نتیجه به سراغ پارتیشن بعدی می رود که 112KB ان اشغال شده است و از مقدار باقی مانده ی ان یک پارتیشن جدید ایجاد میشود و این پردازه در ان قرار میگیرد. در اخر نیز چون اکثر پارتیشن ها نیمه بیشتری از انها اشغال شده یا حافظه کمی دارند پردازه 426K باید صبر کند تا یکی از پردازه ها کارش به اتمام برسد و از یکی از پارتیشن های 500KBیا 600KBخارج شود.

در الگوریتم best-fit پارتیشنی انتخاب میشود که بیشترین حافظه ممکن را اشغال کند به این ترتیب نحوه قرار گیری پردازه ها به شکل زیر خواهد شد:

112KB -> 200KB

417KB -> 500KB

212KB -> 300KB

426KB -> 600KB

در الگوریتم worst-fit پارتیشنی انتخاب میشود که کمترین حافظه ممکن را اشغال کند. در نتیجه نحوه قرار گیری پردازه ها به این شکل خواهد شد:

112KB -> 600KB

417KB -> 500KB

212KB -> 488KB (new partition : 600KB -112KB = 488KB)

426KB -> MUST WAIT

الگوریتم best-fit از سایر الگوریتم ها بهینه تر از حافظه استفاده می‌کند.

ب:

0,430 : 219 + 430 = 649

Segment : 0 => Base : 219

1,10 : 2300 + 10 = 2310

Segment : 1 => Base : 2300

2,2500 : illegal reference, trap to operating system

Segment : 2 => Base : 90

3,400 : 1327 + 400 = 1727

Segment : 3 => Base : 1327

4,112 : illegal reference, trap to operating system

Segment : 4 => Base : 1952