فصل چهارم

بن بست (Dead Lock)

استاد : علی روانگرد

بن بست

- مسدود بودن دائمی محموعه ای از فرایندها که برای منابع سیستم رقابت می کنند یا با یکدیگر در ارتباط هستند.
- این مسدود بودن ادامه می یابد تا سیستم عامل عمل فوق العاده ای انجام دهد، مثلا فرایندی را حذف کند یا مجبور به برگشت به عقب کند.

مثال

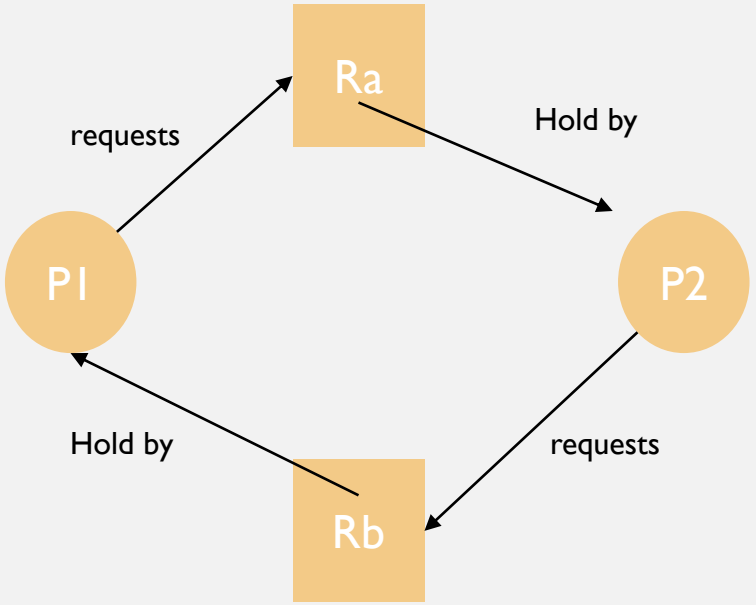
- چهار خودرو که در یک چهارراهی رسیده اند مفروض است. چهار ربع این چهارراه، منابع محسوب می شوند. هر خودرو برای عبور از چهارراه به دو ربع از چهارراه نیاز دارد. اگر هر چهار خودرو وارد وارد تقاطع شوند، هر خودرو یک ربع از چهارراه را در اختیار گرفته ولی نمی تواند پیش برود، چون ربع مورد نیاز دوم در اختیار خودروی دیگر است. درنتیجه بن بست رخ می دهد.

شرایط بن بست

- اگر چهار شرط زیر همزمان در سیستمی وجود داشته باشد، بن بست رخ میدهد.
- برقراری هر چهار شرط، شروط لازم و کافی برای وقوع بن بست می باشند.
- ۱- نگهداری و انتظار (Hold and Wait)
- فرایندی وجود دارد که حداقل یک منبع را در اختیار داشته باشد و منتظر به دست آوردن منبع دیگری باشد که فعلا در اختیار فرایند دیگر است.
- ۲- قبضه نشدنی (Non Preemption)
- هنگامی که فرایندی منبعی را در اختیار دارد و نتوان آن منبع را به زور بازپس بگیرد.
- ۳- انحصار متقابل (Mutual Exclusion)
- در هر زمان تنها یک فرایند از یک منبع استفاده کند و اگر فرایند دیگری درخواست همان منبع را کند، باید منتظر بماند تا منبع آزاد شود.
- ۴- انتظار چرخشی (Circular Wait)
- چرخه ای از انتظار وجود دارد.

مثال انتظار چرخشی:

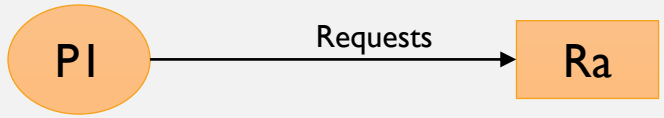
- P1 منتظر منبعی است که در اختیار P2 است.
- P2 منتظر منبعی است که در اختیار P3 است.
- ....
- در نهایت Pn منتظر منبعی است که در اختیار P1 است.



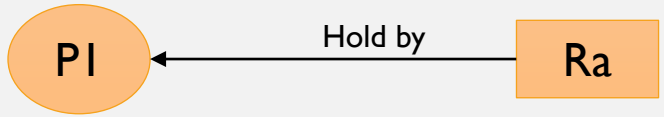
- نکته : شرط انتظار چرخشی انتظار چرخشی منجر به شرط نگهداری و انتظار می گردد.

گراف تخصیص منابع (Resource Allocation Graph)

- روشی برای تشریح دقیقتر مسئله بن بست است. فرایند ها با دایره و منابع با مربع نشان داده می شوند.
- دو نوع یال وجود دارد:
- ۱- یال درخواست: Pi → Ra



- ۲- یال تخصیص: Ra → Pi



مشخص کردن بن بست

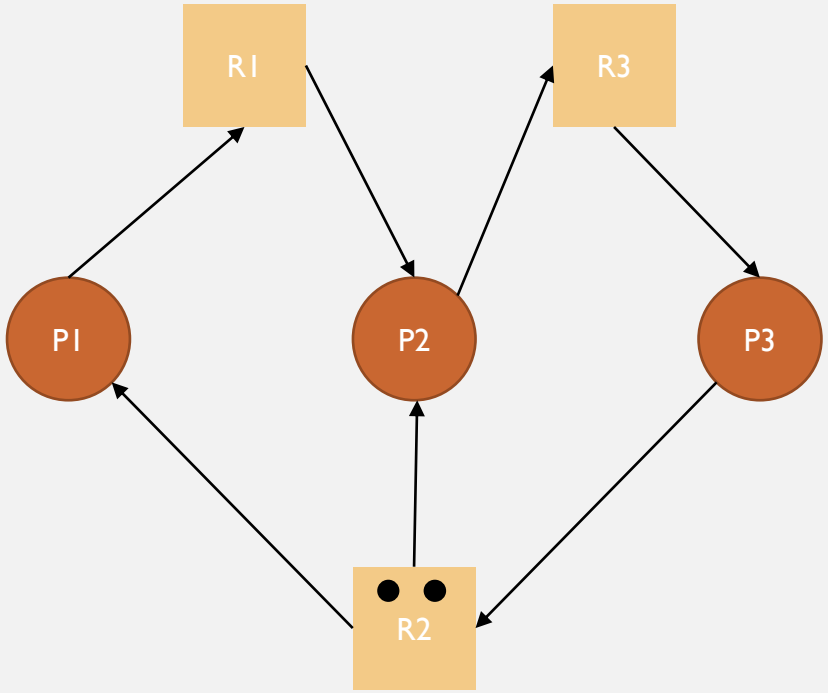
- نحوه تعیین بن بست با توجه به گراف تخصیص منبع:
- ۱- اگر در گراف چرخه (سیکل) نباشد، بن بست نداریم.

- ۲- اگر چرخه ای در گراف باشد:
  - الف: اگر از هر منبع یک نمونه داشته باشیم، بن بست رخ میدهد.
  - ب: اگر هر منبع دارای چند نمونه باشد، احتمال بن بست است. (وجود چرخه شرط لازم برای وجود بن‌بست است ولی شرط کافی نیست.)

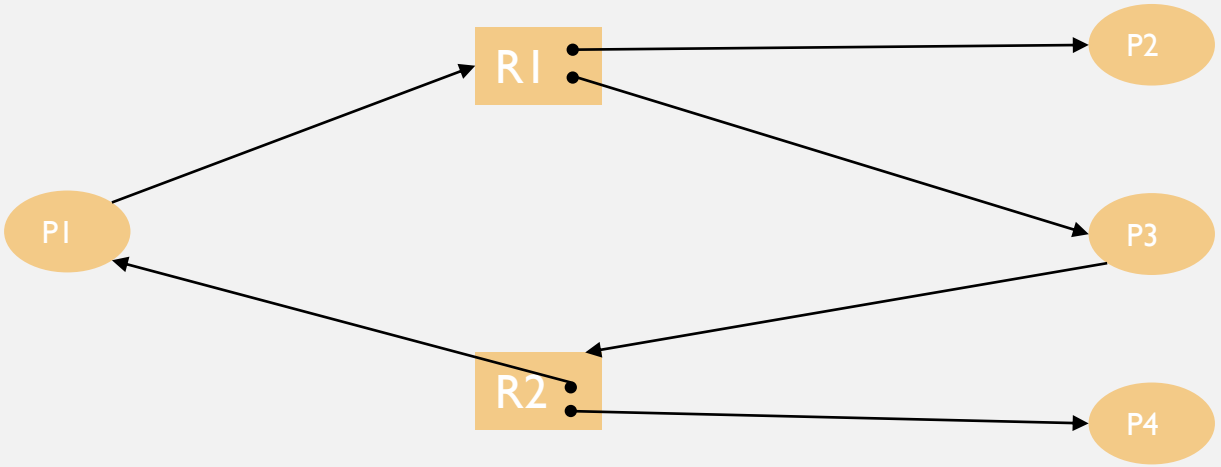
مثال:

- 1) P1 → R1 → P2 → R3 → P3 → R2 → P1
- 2) P2 → R3 → P3 → R2 → P2

- بنابراین بن بست رخ میدهد.



مثال



- باوجود چرخه بن بست رخ نمیدهد. چون P4 می تواند R2 را آزاد کند و آنگاه R2 به P3 داده می شود و چرخه از بین برود.

شرط رخ ندادن بن بست:

- اگر در سیستمی n فرایند و m منبع (از یک نوع) موجود باشد، در صورت برقرار بودن شرط زیر بن بست رخ نخواهد داد.

$$\sum_{i=1}^{n} request(i) < m + n$$

- مثال
- کامپیوتری دارای m واحد از یک منبع است و n پردازه برای در خاتیار گرفتن این منابع با هم رقابت می کنند.
- هرپردازه حداکثر به دو واحد از این منبع نیاز داشته و در هر زمان می تواند تنها یک واحد منبع را آزاد یا درخواست نماید.
- حداکثر مقدار n که به ازای آن سیستم دچار بن بست نمی شود، را مشخص کنید.

$$\sum_{i=1}^{n} request(i) < m + n \Rightarrow 2n < m + n \Rightarrow n < m$$

روشهای رفع بن بست را میتوان به سه گروه عمده تقسیم کرد:

- ۱- پیشگیری (جلوگیری) از بن بست (Prevention):
- سیستم طوری طراحی شود که از قبل امکان بن بست از بین برود، یعنی تضمین کنیم که حداقل یکی از چهار شرط بن بست رخ ندهد.
- ۲- اجتناب از بن بست (Avoidance)
- در مورد درخواست های منبع طوری رفتار شود که حداقل یک از چهار شرط بن بست رخ ندهد.
- ۳- کشف بن بست (Detection)
- هرجا که ممکن باشد، منابع به فرایندها داده می شود. سیستم عامل به طور متناوب الگوریتمی را دنبال می کند تا وجود شرط انتظار چرخشی را کشف کند.

روش های پیشگیری:

- نگهداری و انتظار
- باید فرایند را مجبور کرد تا همه منابع مورد نیاز را یکباره درخواست کند و تا زمانی که همه منابع به او داده نشود، فرایند را مسدود کرد.
- قبضه نکردن
- اگر فرایندی منابعی را دراختیار دارد، درخواست جدیدش موافقت نشود. در این حالت باید منابع قبلی خود را آزاد کند و در صورت نیاز، مجددا با منابع اضافی درخواست نماید.

انتظار چرخشی

- برای پیشگیری از بروز این شرط، یک ترتیب خطی از انواع منابع تعریف میکنیم.
- اگر منابعی از نوع R به یک فرایند تخصیص یابد، در ادامه تنها منابعی را می تواند درخواست کند که نوع آنها بعد از R قرار گرفته باشد.
- مثلا اگر i>j آنگاه Ri قبل از Rj است. اگر فرایند P1 منبع Ri را در اختیار داشته باشد وRj را تقاضا کند و P2 منبع Rj را در اختیار داشته باشد و Ri را تقاضا کند، این حالت غیرممکن است چون مستلزم i>j و j>i می‌باشد.
- پیشگیری از شرط انتظار چرخشی، ممکن است موجب کند کردن فرایندها و رد کردن غیرضروی دسترسی منابع گردد.

روش‌های اجتناب از بن بست

- برای اجتناب از بن بست دو رویکرد وجود دارد:
- ۱- عدم شروع فرایندی که ممکن است درخواست‌هایش منجر به بن‌بست شود.
- ۲- عدم پاسخ به درخواست‌های منبع اضافی از طرف فرایندی که با این تخصیص ممکن است منجر به بن بست شود.(الگوریتم بانکداران)

- در اجتناب از بن‌بست، تصمیم گیری زیر بصورت پویا انجام می‌شود:
- «اگر منبع درخواست شده، تخصیص داده شود، آیا می توان منجر به بن بست شود یا نه»
- یعنی اجتناب از بن بست نیازمند اطلاع از درخواستهای آینده است.

محدودیت های اجتناب از بن بست

- ۱- تعداد منابع تخصیص باید ثابت باشد.
- ۲- فرایندی که منبعی در اختیار دارد نمی تواند خارج شود.
- ۳- حداکثر منابع مورد نیاز هر فرایند باید از قبل معلوم شود.
- ۴- فرایندهای مورد نظر باید مستقل باشند.

- امتیاز اجتناب از بن بست این است که قبضه کردن فرایند، لازم نمی باشد. (بر خلاف کشف)

حالت امن

- حالتی که در آن حداقل یک ترتیب از فرایندها وجود دارد که می تواند اجرا و کامل شوند. بدون اینکه بن بست رخ دهد.

- وضعیت بن بست، یک وضعیت ناامن است.
- اما تمام وضعیت های ناامن، الزاما بن بست نیستند.

الگوریتم بانکداران

- این الگوریتم برای اجتناب از بن بست استفاده می شود.
- از بردار و ماتریس های زیر استفاده می شود:
- ۱- بردار Resource: شامل مقدار کل هر یک از منابع
- ۲- بردار Available: شامل مقدار کل هر یک از منابعی که موجود است (تخصیص داده نشده)
- ۳- ماتریس claim (حداکثر نیاز): شامل تمام درخواست های فرایند ها
- ۴- ماتریس Allocation: شامل منابع تخصیص داده شده به فرایندها

- مثال
- بررسی وجود حالت امن:

(Resource: R1=9, R2 = 3, R3 = 6)

	R1	R2	R3
P1	3	2	2
P2	6	1	3
P3	3	1	4
P4	4	2	2

Claim

	R1	R2	R3
P1	1	0	0
P2	6	1	2
P3	2	1	1
P4	0	0	2

Allocation

	R1	R2	R3
P1	3	2	2
P2	6	1	3
P3	3	1	4
P4	4	2	2

Claim

	R1	R2	R3
P1	1	0	0
P2	6	1	2
P3	2	1	1
P4	0	0	2

Allocation

	R1	R2	R3
P1	2	2	2
P2	0	0	1
P3	1	0	3
P4	4	2	0

Need

(Resource: R1=9, R2 = 3, R3 = 6) → (R1=0, R2=1, R3=1)

بعد از اجرای P2:

3	2	2
0	0	0
3	1	4
4	2	2

Claim

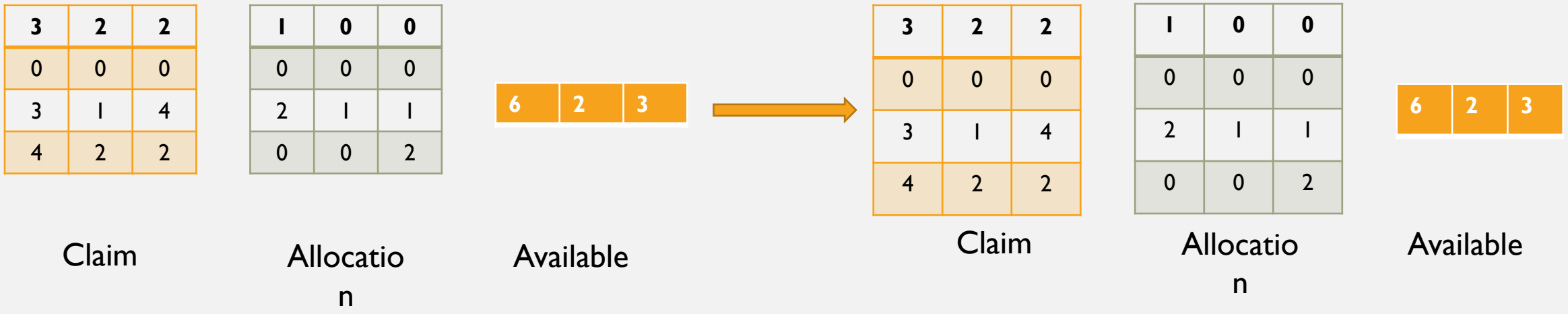
1	0	0
0	0	0
2	1	1
0	0	2

Allocation

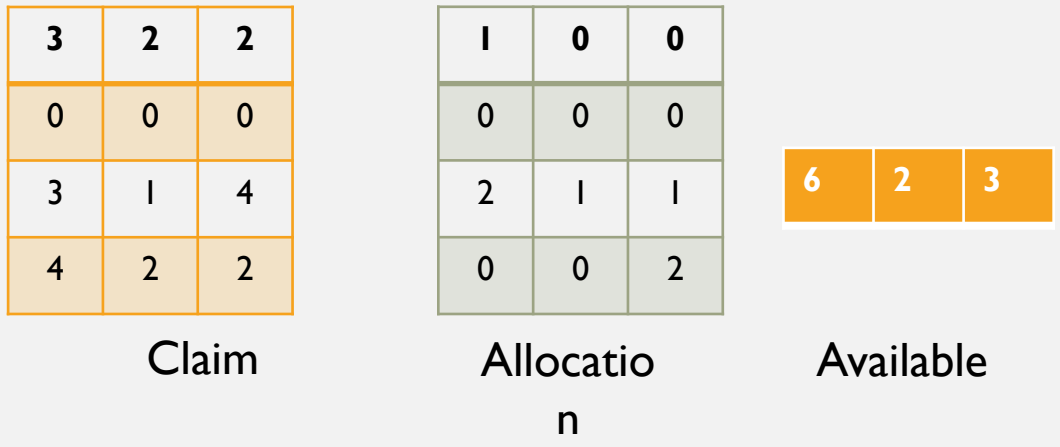
6	2	3

Available

- بعد از اجرای P1:



- در نهایت P4اجرا می شود.



- ترتیب امن: P2→P1→ P3 → P4

- مثال : تحت سیستم عاملی ۴ فرایند و ۲ منبع مدیریت می شود. حالت سیستم چیست؟ منابع در دسترس: (R1=2, R2=1)

	R1	R2
P1	7	2
P2	1	3
P3	1	1
P4	3	0

مقادیر اختصاص داده شده

	R1	R2
P1	9	5
P2	2	6
P3	2	2
P4	5	0

حداکثر نیاز

	R1	R2
P1	9	5
P2	2	6
P3	2	2
P4	5	0

- منابع در دسترس: (R1=2, R1 = 1)
- فرایند p1 و p2 در ابتدا نمی توانند اجرا شوند. بعد از اجرای p3 : (R1=3, R2= 2)
- باز هم p1 و p2 نمی توانند اجرا شوند. بعد ازاجرای p4 : (R1=6, R2=2)
- باز هم p1 و p2 نمی توانند اجرا شوند.