الگوریتم بانکدار

تهیه و تنظیم: مبین خیری

شماره دانشجویی: 994421017

استاد راهنما: دکتر لیلا شریفی

چکیده:

الگوریتم بانکدار یک الگوریتم تخصیص منابع و اجتناب از بن‌بست است که توسط ادسخر دیسترا توسعه یافته که امنیت آن به وسیله شبیه‌سازی تخصیص بیشترین مقدار ممکن از تمام منابع آزمایش شده به طوری که یک s-state ایجاد می‌کند تا برای همه فرایندهای در حال انتظار تمام شرایط بن‌بست را قبل از تصمیم‌گیری و اجازه تخصیص منبع بررسی کند .

این الگوریتم در طراحی فرایند در سیستم عامل THE توسعه یافته و در اصل (به هلندی) در EWD108 شرح داده شده.

الگوریتم

الگوریتم بانکدار هر زمان که یک فرایند درخواست منبع کند توسط سیستم عامل اجرا می‌شود. الگوریتم به وسیله انکار یا تعویق درخواست از بن‌بست جلوگیری می‌کند. این در صورتی است که اگر تعیین شود که پذیرش درخواست می‌تواند سیستم را به حالت نا امن ببرد. (حالتی که بن‌بست می‌تواند رخ دهد).
هنگامی که یک فرایند جدید وارد یک سیستم می‌شود باید حداکثر تعداد درخواستی از هر یک از منابع را اعلام کند که «نباید از تعداد کل منابع در سیستم تجاوز کند». همچنین هنگامی که یک فرایند همه منابع درخواستی را تحویل می‌گیرد باید آن‌ها را پس از اتمام عملیاتش، بازگرداند .

منابع

الگوریتم بانکدار برای انجام کار نیاز به دانستن سه چیز دارد :

- هر فرایند چه مقدار از هر نوع منبع را درخواست کرده‌است.
- هر فرایند چه مقدار از هر نوع منبع را در اختیار دارد.
- چه تعدادی از هر منبع موجود است.

منابع تنها در صورتی ممکن است اختصاص یابند که شرایط زیر وجود داشته باشد :

1. Resource: ≤ Claimتعداد درخواست‌های یک فرایند از یک منبع، کوچکترمساوی تعداد کل آن منبع باشد.
2. available: ≤ requestتعداد نیازهای یک فرایند به یک منبع، کوچکترمساوی تعداد موجود از آن منبع باشد. (نیاز =درخواست‌ها - تخصیص یافته‌ها(

برخی از منابعی که در سیستم‌های واقعی دنبال می‌شوند حافظه، سمافورها و رابط دسترسی هستند .نام *الگوریتم بانکدار* برگرفته از این حقیقت است که این الگوریتم می‌تواند در سیستم بانکی مورد استفاده قرار گیرد تا تضمین کند که بانک همه منابع خورد را از دست نمی‌دهد. چون بانک هیچ‌گاه پول را به نحوی تخصیص نمی‌دهد که نتواند نیازهای بقیه مشتری‌ها را برطرف نکند. با الگوریتم بانکدار بانک تضمین می‌کند که زمانی که مشتری‌ها پول درخواست می‌کنند بانک هیچ‌گاه به حالت ناامن نمی‌رود. اگر درخواست مشتری باعث نشود که بانک حالت امن را ترک کند، مبلغ اختصاص میابد در غیر اینصورت مشتری باید صبر کند تا زمانی که مشتری‌های دیگر سپرده کافی قرار دهند .

ساختمان داده‌های اولیه برای پیاده‌سازی الگوریتم بانکدار :

nرا شماره فرایند و m را شماره منابع موجود قرار دهید. حال به این ساختمان داده‌ها نیاز است :

- Resources(بردار منابع کل): یک بردار به طول m که نشان دهنده تعداد کل منابع موجود از هر نوع منبع است.
- Available(بردار موجودی): یک بردار به طول m که نشان دهنده تعداد منابع موجود از هر نوع منبع است. اگر Available[j] = k باشد یعنی k تا از منبع نوع Rj موجود است.
- Claim(کل درخواست‌ها): یک ماتریس n×m که حداکثر نیاز هر فرایند را به انواع منابع نشان می‌دهد. اگر Max[i,j] = k باشد آنگاه فرایند Pi حداکثر به kتا از منبع نوع Rj نیاز دارد.
- Allocation(تخصیص یافته): یک ماتریس n×m که تعداد منابع از هر نوع را که به هر فرایند اختصاص یافته‌است نشان می‌دهد. اگر Allocation[i,j] = k یعنی فرایند Pi در حال حاضر k تا از هر منبع نوع Rj را در اختیار دارد.
- Need(نیاز): یک ماتریس n×m که نیاز هر فرایند به هر منبع را نشان می‌دهد. اگر Need[i,j] k =یعنی Pi به k تا از منبع نوع Rj نیاز دارد.

توجه Need = Claim- Allocation :

مثال

با فرض اینکه سیستم دارای چهار نوع منبعA,B ، C,Dباشد. این مثال نشان می‌دهد منابع چگونه اختصاص می‌یابند. توجه که این مثال مشان می‌دهد سیستم یک لحظه قبل از درخواست جدید در چه حالتی است .

تعداد کل منابع در سیستم :(Resource)

R_1 R_2 R_3

۹ ۳ ۶

بردار موجودی :(Available)

تعداد به کاررفتن هر منبع در کل فرایندها - تعداد کل آن منبع = موجودی آن منبع

R_1 R_2 R_3

۰ ۱ ۱

ماتریس منابع تخصیص یافته : (Allocation)

R_1 R_2 R_3

P_1 1 0 0

P_2 6 1 2

P_3 2 1 1

P_4 0 0 2

ماتریس کل درخواست ها : (Claim)

R_1 R_2 R_3

P_1 3 2 2

P_2 6 1 3

P_3 3 1 4

P_4 4 2 2

ماتریس منابع مورد نیاز فرایندها برای اتمام :(Need)

کل درخواست - تخصیص یافته =نیاز

R_1 R_2 R_3

P_1 2 2 2

P_2 0 0 1

P_3 1 0 3

P_4 4 2 0

حالات امن و ناامن

1. زمانی یک حالت را امن گوییم که حداقل یک ترتیب از فرایندها وجود دارد که تمام فرایندها می‌توانند تا کامل شدن اجرا شوند. از آنجایی که سیستم نمی‌تواند بداند یک فرایند کی به پایان می‌رسد یا چه تعداد منابع درخواست خواهد شد، پس سیستم فرض می‌کند همه فرایندها تلاش می‌کنند که حداکثر منابع را در اختیار داشته باشند تا زودتر به پایان برسند. از آنجایی که سیستم به‌طور ویژه اهمیت نمی‌دهد که اجرای هر فرایند چقدر طول می‌کشد پس این یک فرض معقول در اکثر موارد است. همچنین اگر یک فرایند بدون دستیابی به حداکثر منابع پایان یابد فقط اجرا را روی سیستم ساده‌تر کرده‌است .حالت امن در نظر گرفته شده که تصمیم بگیرد فرایند به صف آماده برود .حالت امن ایمنی را تضمین می‌کند.

طبیعی است که حالتی که امن نباشد، حالت ناامن است .

مثال حالت امن

می‌توانیم با نشان دادن اینکه هر فرایند حداکثر منابع مورد نیاز را دریافت و به پایان می‌رسد حالت امن را در مثال قبل نشان دهیم .

- فرایند p1 برای کامل شدن به <۲ ۲ ۲ >واحد به ترتیب از R1,R2,R3نیاز دارد. اما این تعداد از منابع از تعداد بردار موجودی بیشتر است. پس درخواست رد می‌شود.
- فرایند p2 برای کامل شدن به <۱ ۰ ۰ >نیاز دارد. منابع موجود است. پس اختصاص داده می‌شود و بردار موجودی به حالت زیر در می‌آید :
  - [*<available : <0 1 1> - <0 0 1> = <۰ ۱ ۰>*]

- فرایند p2 تمام منابع درخواستی اش را در اختیار دارد پس تا تمام شدن اجرا می‌شود و بعد از آن منابعش را آزاد می‌کند و منابع آزاد شده به بردار موجودی اضافه می‌شود .
  - [<*available: <0 1 0>+<6 1 3>= <۶ ۲ ۳*>]
  - ماتریس درخواست‌های کل (Claim) به صورت زیر تغییر می‌کند:

R_1 R_2 R_3

P_1 3 2 2

P_2 0 0 0

P_3 3 1 4

P_4 4 2 2

➢پس از اتمام هر فرایند، باید دوباره از اولین ردیف، نیازمندی‌ها بررسی شود

- فرایند p1برای کامل شدن به >۲ ۲ ۲ <نیاز دارد. منابع موجود است. پس اختصاص داده می‌شود فرایند کامل شده و منابعش را آزاد می‌کند :
  - [<*available: <6 2 3>-<2 2 2> + <3 2 2>= <۷۲۳*>]
  - ماتریس درخواست‌های کل:

R_1 R_2 R_3

P_1 0 0 0

P_2 0 0 0

P_3 3 1 4

P_4 4 2 2

- فرایند p3 برای کامل شدن به >۳ ۰ ۱ <نیاز دارد. منابع موجود است. پس اختصاص داده می‌شود فرایند کامل شده و منابعش را آزاد می‌کند:
- [<available: <7 2 3>-<1 0 3> + <3 1 4>= <۹ ۳ ۴]

- ماتریس درخواست‌های کل :

R_1 R_2 R_3

P_1 0 0 0

P_2 0 0 0

P_3 0 0 0

P_4 4 2 2

- فرایند p4 برای کامل شدن به <۰ ۲ ۴ >نیاز دارد. منابع موجود است. پس اختصاص داده می‌شود فرایند کامل شده و منابعش را آزاد می‌کند:
- [*<available : <9 3 4> - <4 2 0> + <4 2 2> = <۹ ۳ ۶>*]

R_1 R_2 R_3

P_1 0 0 0

P_2 0 0 0

P_3 0 0 0

P_4 0 0 0

➢بدیهی است که پس از پایان فرایندها، آخرین مقدار بردار موجودی باید با مقدار کل منابع برابر باشد

➢بنابراین ما «ترتیبی» از اجرای فرایندها یافتیم، که از آن، حالت امن نتیجه گرفته می‌شود

مثال برای حالت ناامن

در مثال قبل فرض کنید مقادیر تخصیص یافته فرایند p2 از <۲ ۱ ۶ >به <۱ ۱ ۵ >و فرایند p1 از <۰ ۰ ۱ >به <۱ ۰ ۲ >تغییر حالت دهند. با این تغییر ماتریس نیازها هم تغییر خواهد کرد. یعنی :

ماتریس تخصیص یافته‌ها :

R_1 R_2 R_3

P_1 2 0 1

P_2 5 1 1

P_3 2 1 1

P_4 0 0 2

ماتریس نیازها :

R_1 R_2 R_3

P_1 1 2 1

P_2 1 0 1

P_3 1 0 3

P_4 4 2 0

بردار موجودی :

R_1 R_2 R_3

۰ ۱ ۱

- همانگونه که مشاهده می‌شود تمامی فرایندها به حداقل یک واحد از منبع R_1 نیازمندند؛ اما بردار موجودی R_1 *صفر* است. پس این حالت از تخصیص، حالت *نا امن* است.

پایان.