سیستم عامل

فرجیان

فصل هشتم

حافظه مجازي

ساختار های سخت افزاری و کنترل:

- مشخصه اصلی صفحه بندی و قطعه بندی کلید پیشرفت مدیریت حافظه اند:
  - کلیه مراجعات یک فرایند به حافظه، آدرسهای منطقی هستند که به صورت پویا در زمان اجرا به آدرس فیزیکی ترجمه میشوند.
    - فرایند ممکن است به داخل و خارج حافظه مبادله شود بنابراین ممکن است بخشهای مختلفی از حافظه را اشغال کند.
  - ممکن است برنامه به بخش های تقسیم شود که نیاز نیست به صورت پیوسته در حافظه قرار گیرند.
  - برای اجرای فرایند نیاز به بار شدن تمام بخشهای فرایند نیست.

اجرای برنامه:

- سیستم عامل بخشی از فرایند را به حافظه اصلی بار میکند.
- مجموعه مقیم: بخشی از فرایند که در حافظه است.
- زمانی که پردازنده به آدرس منطقی نیاز دارد که در حافظه وجود ندارد، وقفه ای به معنای خطای دسترسی به حافظه تولید میکند.
- سیستم عامل فرایند وقفه خورده را مسدود میکند و کنترل را بدست میگیرد.

اجرای برنامه:

- سیستم عامل آن قسمت از فرایند که تولید کننده خطای حافظه است را بداخل بار میکند.
  - سیستم عامل یک درخواست خواندن به دیسک صادر میکند.
  - در حین عمل I/O سیستم عامل فرایند دیگری را اجرا میکند.
  - زمانی که عمل I/O کامل شد، یک وقفه ورودی/خروجی صادر میشود و موجب میشود فرایند مسدود به حالت آماده تغییر حالت دهد.

مزایای تقسیم فرایند:

- فرایندهای بیشتری میتوانند در حافظه نگه داشته شوند.
  - تنها بخشی از فرایند به حافظه بار میشود.
  - با داشتن فرایندهای بیشتر در حافظه احتمال وجود فرایند آماده به اجرا بیشتر میشود و این موجب افزایش کارایی پردازنده میشود.
- فرایند میتواند از حافظه بزرگتر باشد.

حافظه حقيقي و مجازي

حافظه حقيقي : به حافظه اصلي حافظه حقيقي مي گويند.

حافظه مجازي : حافظه بزرگتر و كارآمد تر كه كاربر از آن استفاده مي كند حافظه مجازي مي گويند.

کوبیدگی:

- اگر سیستم عامل تکه ای را دقیقاً قبل از اینکه به کار گرفته شود خارج کند، با فاصله کمی باید آنرا به داخل بیاورد.
- با اجرای زیاد این عمل پردازنده بیشتر وقت خود به جای اجرای برنامه صرف مبادله تکه ها میکند، به این عمل **کوبیدگی** میگویند.

اصل محلي بودن:

- مراجعات به برنامه و داده ها در حافظه ،خوشه اي هستند.
- تعداد محدودي از بخش های فرآیند ها در حافظه اجرا مي شوند و باید حدس زد که کدام بخش برنامه در آینده اجرا مي شود.
- به این ترتیب از کوبیدگي جلو گیري مي کنند .
- تاکید در کارآمدي فرآیندها در یک محیط حافظه مجازي است.

براي كارآمدي حافظه مجازي:

- بايد حمايت سخت افزاري وجود داشته باشد.

- سيستم عامل بايد داراي نرم افزار هايي جهت تبادل قطعه ها از حافظه اصلي به حافظه مجازي باشد.

مشخصات صفحه بندي و قطعه بندي:

IASBS 1992-2012

صفحه بندي ساده	قطعه بندي ساده	حافظه مجازي با صفحه بندي	حافظه مجازي با قطعه بندي
حافظه به تكه هاي مسا وي تقسيم	حافظه اصلي تقسيم نشده	حافظه اصلي به تكه يكسان به نام قاب تقسيم شده	حافظه اصلي تقسيم نشده
برنامه به صفحه تقسيم شده	برنامه قطعه بندي شده	برنامه به صفحه تقسيم شده	برنامه قطعه بندي شده
تكه تكه شدن داخلي در قاب	بدون تكه تكه شدن داخلي	تكه تكه شدن داخلي در قاب	بدون تكه تكه شدنداخلي
بدون تكه تكه شدن خارجي	تكه تكه شدن خارجي	بدون تكه تكه شدن خارجي	تكه تكه شدن خارجي
تهيه جدول صفحه	تهيه جدول قطعه	تهيه جدول صفحه	تهيه جدول قطعه
ليست قا ب هاي آزاد	ليست حفره هاي آزاد	ليست قا ب هاي آزاد	ليست حفره هاي آزاد
بدست آوردن آدرس منطقي	بدست آوردن آدرس منطقي	بدست آوردن آدرس منطقي	بدست آوردن آدرس منطقي
براي اجرا بايد در حافظه اصلي باشند	براي اجرا بايد در حافظه اصلي باشند	براي اجرا بايد در حافظه اصلي باشند	براي اجرا بايد در حافظه اصلي باشند
		خواندن به حافظه اصلي نيازمند ديسک است	خواندن به حافظه اصلي نيازمند ديسک است

ساختار جدول:

- راهکار اصلی برای خواندن یک کلمه از حافظه ،ترجمه یک آدرس مجازی یا منطقی به یک آدرس فیزیکی ،از طریق به کارگیری جدول صفحه است.

- در حافظه اصلی قرار می گیرد.

- آدرس شروع در ثبات قرار می گیرد.

ساختار جدول:

- از شماره صفحه آدرس مجازی ،به عنوان شاخص برای شناسایی شماره قاب استفاده می کند.

- برای به کارگیری
  - جداول صفحه یک سطحی یا دو سطحی
  - استفاده از ساختار جدول صفحه معکوس است.

- حافظه مجازی از یک حافظه پنهان مخصوص برای مدخل های جدول صفحه استفاده می شود که به آن میانگیر دم دستی ترجمه می گویند.

- یکی از راهکار های بهبود آن به کارگیری میانگیر بزرگتر با مدخلهای بیشتر است.

- یکی دیگر از راه حل ها به کار گیری اندازه صفحه بزرگ تر.

- هنگامی که صفحه در حافظه اصلی قرار نداشته باشد یک خطای دسترسی به نام خطای فقدان صفحه صادر می شود.
- یکی از تصمیمات مهم در طراحی سخت افزار تعیین اندازه صفحه است.
- اندازه صفحه به اندازه فیزیکی صفحه بستگی دارد.

طراحی بخش مدیریت حافظه:

بستگی به سه انتخاب زیر دارد:

- اینکه از روش های حافظه مجازی استفاده شود یا خیر.
- استفاده از صفحه بندی یا قطعه بندی یا هر دو.
- الگوریتم های مورد استفاده برای جنبه های مختلف مدیریت حافظه.

سیاستهای سیستم عامل برای حافظه مجازی:

- سیاست واکشی
- سیاست جاگذاری
- سیاست جایگزینی
- مدیریت مجموعه کاری
- سیاست پاکسازی
- کنترل بار

سیاست واکشي:

- زمان آوردن صفحه به حافظه اصلی را مشخص میکند.
- درخواستی.
  - فقط زمانی یک صفحه را به حافظه اصلی می آورد که مراجعه ای به مکانی از آن صفحه انجام گیرد.
- پیش صفحه بندی .
  - در پیش صفحه بندی، صفحه هایی غیر از آنچه به وسیله خطای صفحه درخواست شده نیز به داخل آورده میشوند.
  - اکثرسیستم عامل ها این سیاست را دنبال می کنند.

سیاست جاگذاري:

- محل قرار گرفتن فرایند در حافظه اصلی را تعیین میکند.
- در یک سیستم قطعه بندی ساده سیاست جایگذاری بسیار مهم است.(اولین برازش ،بهترین برازش و...)
- سیاست جایگذاری در سیستم صفحه بندی یا ترکیب صفحه بندی و قطعه بندی مهم نیست. زیرا سخت افزار ترجمه آدرس را با هر ترکیبی از قابهای صفحه با کارایی یکسان انجام میدهد.

سياست جايگزيني:

- زماني كه حافظه پر باشد، بايد تصميم گيري شود كه كدام صفحه يا صفحاتي جايگزين شود.
- سياست جايگزيني شامل موارد زير است:
  - كدام صفحه جايگزين شود؟
  - صفحه حذف شده بايد كمترين مراجعه در آينده نزديک را داشته باشد.
  - اكثر سياست ها رفتار آينده را بر اساس رفتار گذشته پيش بيني ميكنند.
  - سياست جايگزيني پيچيده تر سربار سخت افزاري و نرم افزاري پيچيده تري براي پياده سازي لازم دارد.

سیاست جایگزینی :

- قفل کردن قاب:
  - صفحه ای که در یک قاب قفل شده باشد نمیتواند جایگزین شود.
  - هسته و ساختارهای کنترلی اصلی سیستم عامل در قابهای قفل شده اند.
  - یک بیت قفل به هر قاب پیوند میخورد.

الگوریتم های اصلی جایگزینی :

- **سیاست بهینه:**
- صفحه ای را برای جایگزینی انتخاب میکند که زمان لازم تا مراجعه بعدی به آن طولانی ترین باشد.
- منجر به کمترین تعداد خطای صفحه میشود.
- امکان اجرای این الگوریتم وجود ندارد زیرا نیازمند دانش کافی سیستم عامل از وقایع آینده است.

OptimalP.swf

الگوریتم های اصلی جایگزینی :

- **حداقل استفاده در گذشته نزدیک(LRU)**
  - صفحه ای را جایگزین میکند که برای مدت طولانی دسترسی به آن نشده است.
  - بر اساس اصل محلی بودن این صفحه باید صفحه ای باشد که کمترین احتمال مراجعه در آینده نزدیک را داشته باشد.
  - هر صفحه را می توان با زمان آخرین مراجعه به آن برچسب گذاری کرد. سربار زیادی برای این روش خواهد بود.

LRUPageR.swf

الگوریتم های اصلی جایگزینی :

- **خروج به ترتیب ورود:**
  - با قابهای تخصیص یافته به فرایند ها مانند یک میانگیر مدور برخورد میکند.
  - صفحات به سبک نوبتی گردشی خارج میشوند.
  - تنها به یک اشاره گر نیاز دارد، که بطور چرخشی به قابهای فرایند اشاره کند، بنابراین ساده ترین پیاده سازی را دارد.
  - صفحه ای که بیشترین مدت در حافظه بوده جایگزین میشود.
  - این صفحات ممکن است مجدداً و بزودی نیاز باشند.

FIFOPage.swf

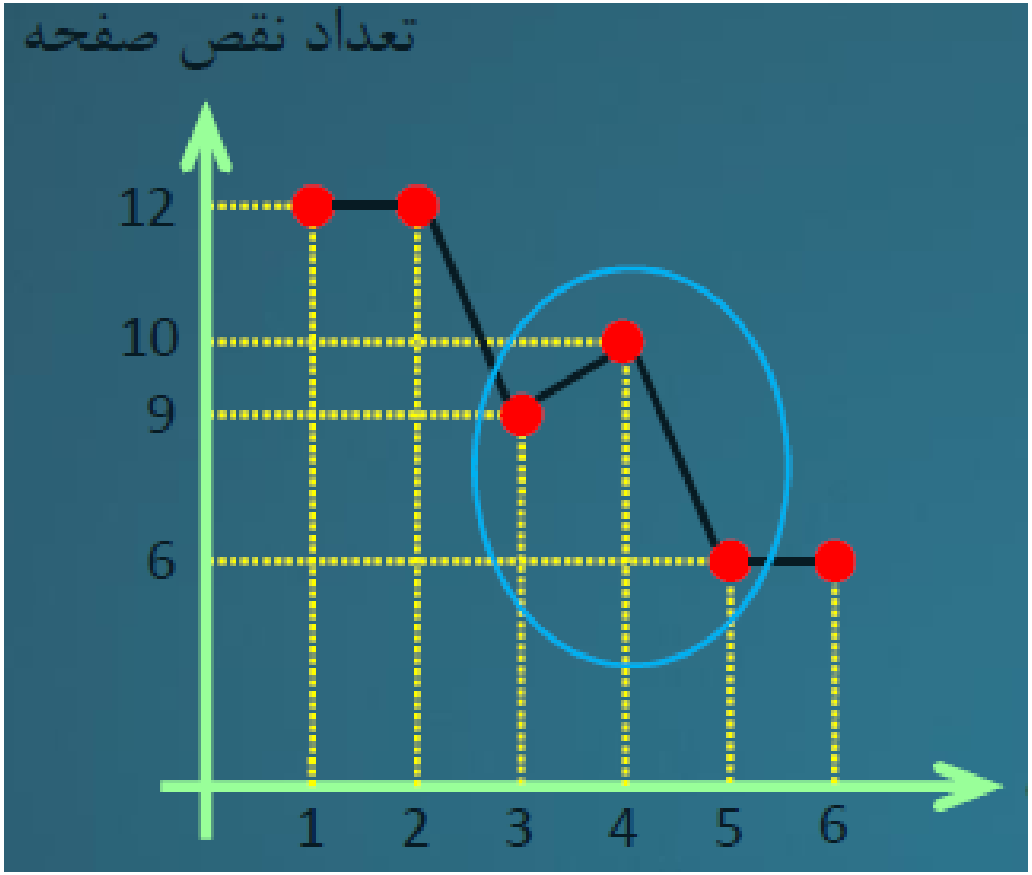
الگوریتم FIFO

- الگوریتم FIFO داراي یک مشکل اساسی است که این مشکل با عنوان ناهنجاري بِلیدي شناخته می شود. یعنی زمانیکه تعداد قاب هاي خالی حافظه افزایش می یابد، بر خلاف انتظار، تعداد خطاهاي نقص صفحه افزایش می یابد.
- تمرین :

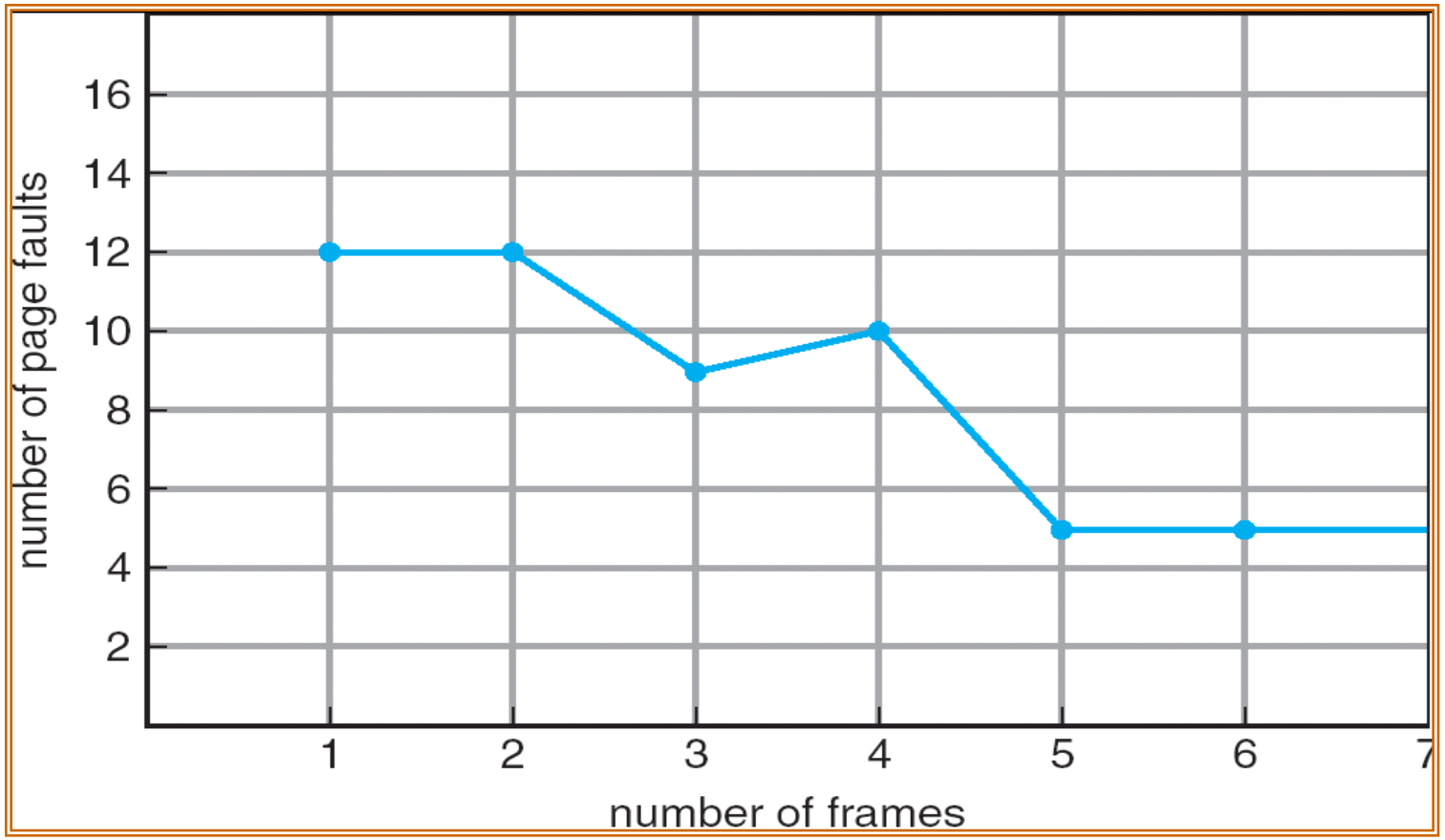
برنامه اي براي اجرا، صفحات زیر را (از چپ به راست) درخواست کند، تعداد نقص هاي صفحه را با الگوریتم FIFO براي تعداد 1 و 3 و 4 و 5 قاب حافظه بررسی کنید.

1,2,3,4,1,2,5,1,2,3,4,5

الگوریتم FIFO



الگوریتم اولین ورودی، اولین خروجی و بی نظمی Belady



الگوریتمNRU

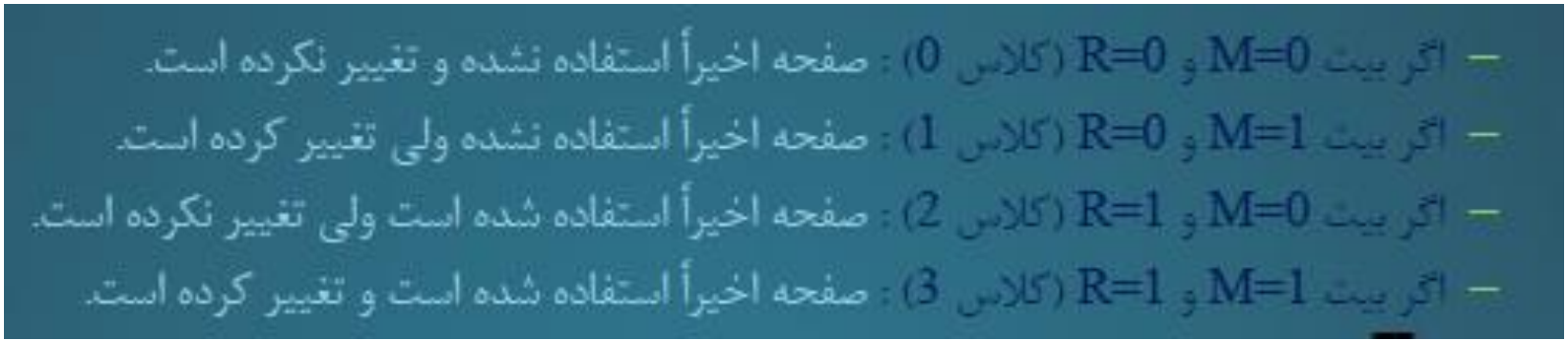
- نام این روش اخیراً استفاده نشده (Not Recently Used) می باشد.
- این روش همان الگوریتمFIFOاست که تغییر کوچکی در آن داده شده است.
- الگوریتمNRUهمانند FIFOدارای مشکل ناهنجاری بِلیدی است.

مبتنی بر کتاب تَننباوم NRUالگوریتم

- در این روش از دو بیت زیر استفاده می شود :
- بیت مراجعهR(refrence):هرگاه برنامه به یک صفحه دسترسی می یابد، این بیت ۱ میشود.
- بیت تغییر M (Moified): هرگاه محتویات یک صفحه توسط برنامه اش تغییر می کند، این بیت ۱ میشود. در برخی از کتب به این بیت، بیت کثیف D (Dirty) گفته میشود.
- نکته : هرگاه این بیت ها ۱ می شود، همانطور باقی می ماند تا سیستم عامل آنها را ۰ کند.

الگوریتمNRU

- در این روش وقتی یک فرآیند شروع می شود، سیستم عامل هر دو بیت Rو M صفر خواهد کرد.
- پس از اجرای برنامه، بیت R به صورت متناوب مثلا هر ۲۰ میلی ۰ میشود تا صفحه هایی که اخیرا مورد استفاده نبودند از بقیه جدا شوند .(در کلاس ها بطور شانسی انتخاب میشوند)



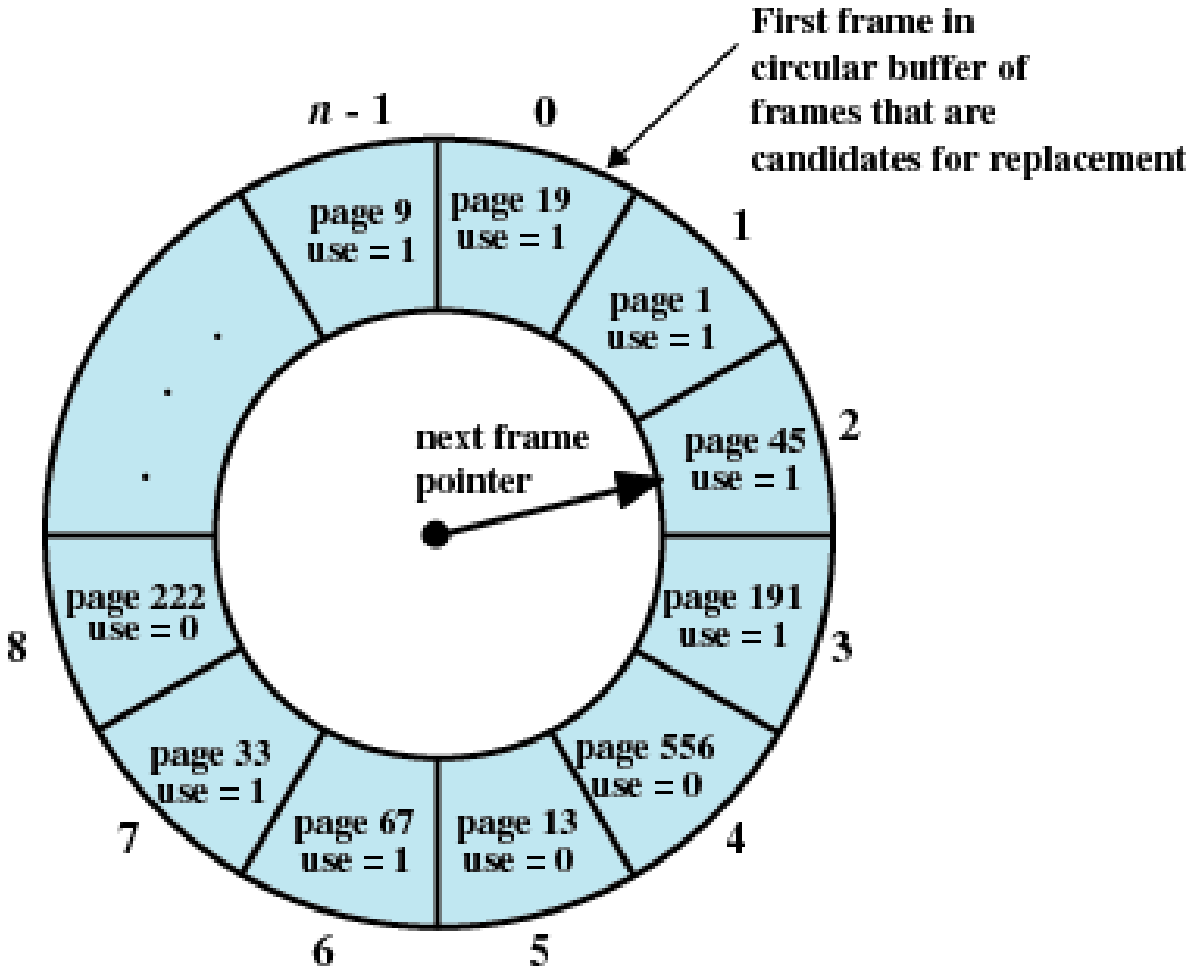
الگوریتم شانس دوباره(second chance)

- این الگوریتم مانند NRU است با این تفاوت که بدون داشتن دوره زمانی ثابت ، یعنی وقفه ساعت
- این الگوریتم مانند FIFO یک لیست پیوندی میسازد در هنگام نقص صفحه ابتدا بیت R قدیمی ترین صفحه در ابتدای لیست بررسی میشود.
  - اگر یک باشد 0 شده و صفحه به انتهای لیست
  - اگر صفر باشد صفحه جایگزین میشود .

الگوریتم های اصلی جابجایی:

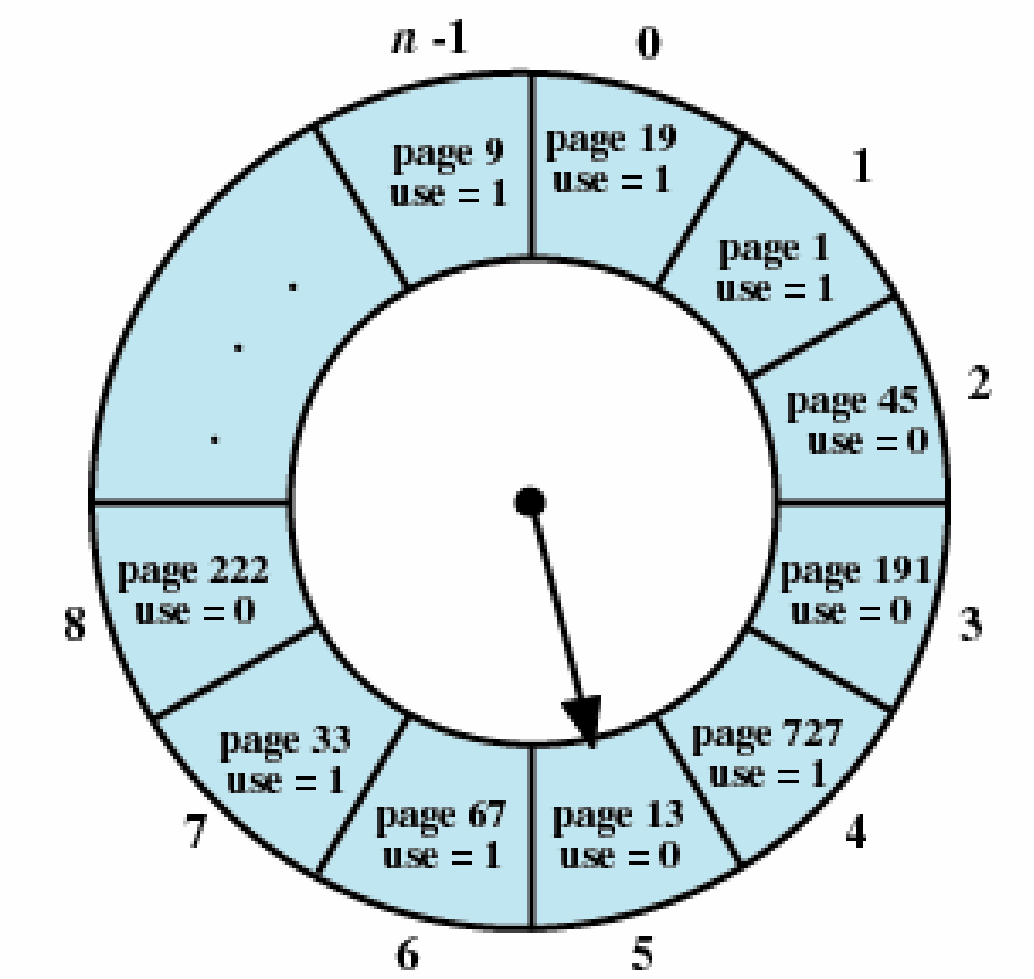
- سیاست ساعت:
  - مانند FIFO است، به جز اینکه در سیاست ساعت از یک بیت اضافی استفاده می شد که به آن بیت استفاده میگویند.
  - برای اولین بار که یک صفحه به داخل قابی در حافضه اصلی بار میشود این بیت ۰ میشود
  - زمانی که به این صفحه مراجعه میشود این بیت ۱ میشود.
  - هنگامیکه زمان جایگزینی صفحه فرا میرسد سیستم عامل میانگیر را مرور کرده تا قابی را بیابد که بیت استفاده از آن صفر باشد.
  - در حین جستجو برای جابجایی هر بیت ۱ به ۰ تغییر میکند.

نمایی از سیاست ساعت(قبل جایگزینی صفحه):



(a) State of buffer just prior to a page replacement

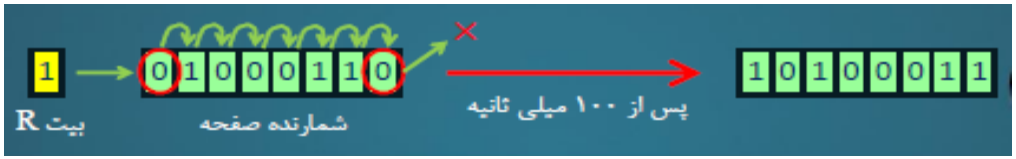
نمایی از سیاست ساعت(بعد جایگزینی صفحه):



(b) State of buffer just after the next page replacement

الگوریتم سالمندی(Aging)

- این الگوریتم در واقع روش LRU را شبیه سازی می کند.
- در این روش یک شمارنده (اغلب ۸ بیتی) برای هر صفحه در جدول صفحه در نظر گرفته می شود. مقدار این شمارنده در ابتدا برابر ۰۰۰۰۰۰۰۰ می باشد.
- در فاصله های مرتب مثلاً هر ۱۰۰ میلی ثانیه، شمارنده هر صفحه به راست شیفت داده شده و از سمت چپ، محتویات بیت R همان صفحه وارد شمارنده مربوطه می شود.
- در هنگام جایگزینی صفحه، سیستم عامل صفحه ای را جایگزین می کند که مقدار عدد شمارنده آن کمتر است.



الگوریتمLFU

- نام این روش (کمترین متناوباً استفاده شده Least Frequently Used) می باشد.
- در این روش یک شمارنده برای هر صفحه در جدول صفحه در نظر گرفته می شود.
- هر چند لحظه بیتRمربوط به هر صفحه با مقدار این شمارنده جمع می شود
- هنگام خطای نقص صفحه، الگوریتم جایگزینی صفحه، صفحه ای را که شمارنده اش دارای مقدار کمتری است، جایگزین می کند.
- این الگوریتم همواره نتیجه مناسب را در بر ندارد.

الگوریتمMFU

- نام این روش (بیشترین متناوباً استفادهMost Frequently Used) میباشد.
- در این روش یک شمارنده برای هر صفحه در جدول صفحه در نظر گرفته می شود.
- هر چند لحظه بیتRمربوط به هر صفحه با مقدار این شمارنده جمع می شود
- هنگام خطای نقص صفحه، الگوریتم جایگزینی صفحه، صفحه ای را که شمارنده اش دارای مقدار بیشتری است، جایگزین می کند.

مقایسه الگوریتمهای جایگزینی صفحه در حالت تخصیص ثابت و دیدگاه محلی:

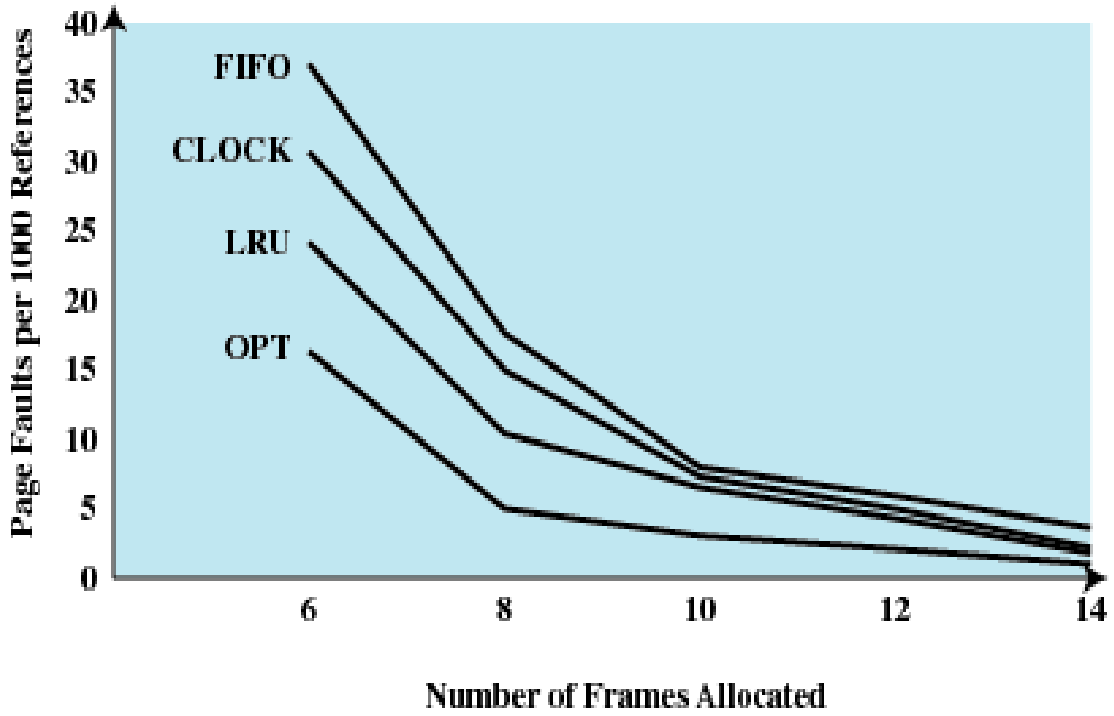


Figure 8.17 Comparison of Fixed-Allocation, Local Page Replacement Algorithms

بحث تخصیص قاب ها

- پس از بررسی الگوریتم های جایگزینی صفحه، نوبت به بحث تخصیص قاب می رسد.
- مشکلی که تخصیص قاب با آن مواجه است بصورت زیر می باشد :
  - " چگونه مقدار مشخصی از حافظه را بین پردازش های تقاضی تقسیم کنیم تا کارآیی سیستم بهتر شود ! "
- یا به عبارت دیگر
  - " چگونه قاب های حافظه را بین پردازش های متقاضی تقسیم کنیم تا کارآیی سیستم بهتر شود ! "

مدیریت مجموعه کاری:

- سیستم عامل باید تصمیم بگیرد چه مقدار از حافظه اصلی را به آن اختصاص دهد واین تخصیص می تواند به صورت پویا و یا به صورت ایستا باشد.

- اندازه مجموعه کاری ثابت یا متغیر.

- قلمروی جایگزینی سراسری یا محلی.

تخصیص قاب مساوي و متناسب

- تخصیص قاب مساوی ساده ترین مدل اختصاص قاب ها است که m قاب را بین n فرآیند به صورت مساوی تقسیم می کند.
  - مثال: اگر ۹۴ قاب آزاد حافظه و ۵ فرآیند داشته باشیم، آنگاه به هر فرآیند ۱۸ قاب حافظه اختصاص خواهد یافت.

$$\begin{matrix} m=94 \\ n=5 \end{matrix} \rightarrow \frac{m}{n}=\frac{94}{5}\approx 18$$

- تخصیص قاب متناسب، قاب های موجود را به نسبت اندازه فرآیندها تقسیم می کند.
  - مثال: اگر دو فرآیند یکی دارای ۱۰ صفحه و دومی با ۴۰ صفحه داشته باشیم و حافظه دارای ۲۰ قاب آزاد باشد، آنگاه به فرآیند اول ۴ قاب حافظه و به فرآیند دوم ۱۶ قاب اختصاص خواهد یافت.

$$\frac{10}{10+40}\times 20=4 \qquad \frac{40}{10+40}\times 20=16$$

تعداد صفحات هر فرآیند

مجموع تعداد صفحات همه فرآیند

تخصیص قاب سراسری و محلی

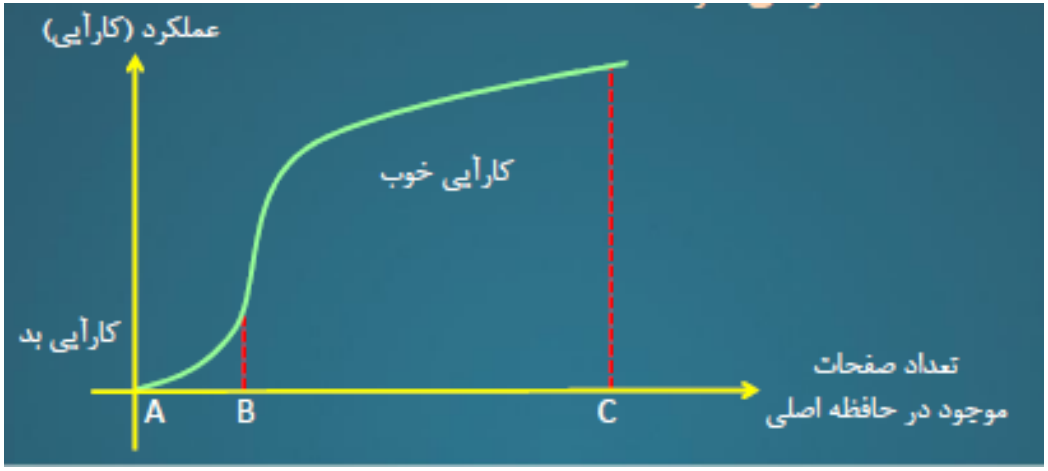
- تخصیص قاب سراسری از بین کل قاب های موجود یک قاب را برای جایگزینی انتخاب می کند، حتی اگر این قاب متعلق به فرآیند دیگری باشد.
  - مشکل: مجموعه قاب های یک فرآیند هم به رفتار صفحه بندی خود فرآیند و هم به رفتار صفحه بندی سایر فرآیندها وابسته است. یعنی مثلاً زمان اجرای فرآیند در یک اجرا ۲ ثانیه و در اجرای دیگر ۲۰ ثانیه باشد.
- تخصیص قاب محلی فقط از بین قاب های مخصوص به آن فرآیند برای جایگزینی صفحات آن فرآیند استفاده می کند.
  - مشکل: ممکن است اجرای یک فرآیند را به تأخیر اندازد، چرا که ممکن است قاب های فرآیندهای دیگر خالی باشند ولی این فرآیند قاب برای اجرا کم بیاورد.

مديريت مجموعه كاري:

	جايگزيني محلي	جايگزيني سراسري
تخصيص ثابت	تعداد قابهاي تخصيص يافته ثابت،صفحه اي كه بايد تخصيص يابد از ميان قابهاي تخصيص يافته انتخاب مي شود.	ممكن نيست.
تخصيص متغيير	تعداد قابهاي تخصيص يافته مي تواند متغيير باشد. صفحه اي كه بايد تخصيص يابد از ميان قابهاي تخصيص يافته انتخاب مي شود.	صفحه اي كه قرار است جايگزين شود از ميان كليه قابهاي حافظه اصلي انتخاب مي شود.

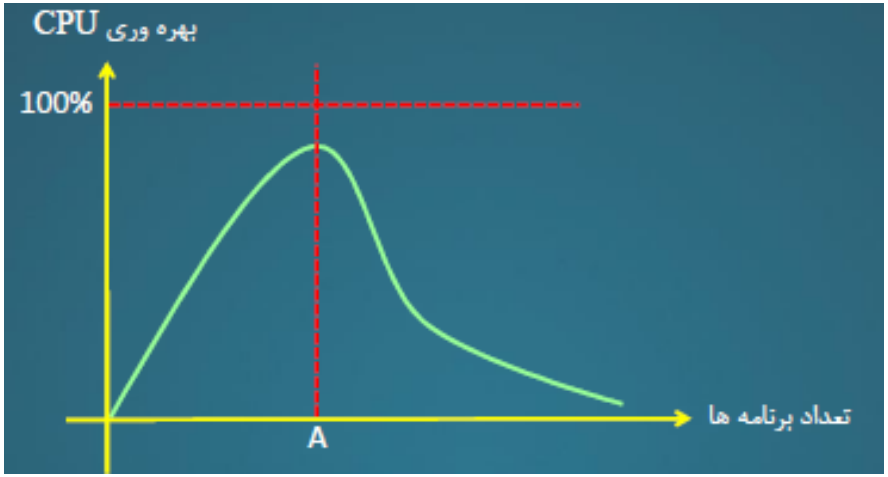
کوبیدگی(Thrashing)

- مجموعه کاری (Work Set) هر برنامه، تعداد صفحاتی از آن برنامه است که اگر درحافظه باشد، کارآیی برنامه و سیستم خیلی بالا می رود.
- اگر مجموعه کاری یک فرآیند در حافظه باشد، فرآیند بدون آنکه تعداد خطای نقص صفحه زیادی داشته باشد، اجرا می شود.



کوبیدگی(Thrashing)

- اگر تمام صفحات یک برنامه در حافظه نباشد، زمان هایی وجود خواهد داشت که صفحات برنامه بین دیسک و حافظه مبادله می شوند. در اي زمان ها می توان برنامه هاي ديگر را اجرا کرد تا CPU همیشه مشغول باشد. نمودار بهره وري CPU بصورت زیر خواهد بود:



کوبیدگی(Thrashing)

- در نمودار صفحه قبل در اثر افزایش سطح چند برنامگی، بهره وري CPU بالا می رود.
- در نقطه A برنامه اول تقاضاي صفحه می کند. یک صفحه از برنامه دوم از حافظه خارج شده و صفحه درخواستی برنامه اول به حافظه می آید.
- وقتی نوبت به اجراي برنامه دوم می رسد، این برنامه نیز درخواست صفحه می کند. یک صفحه از بنامه اول از حافظه خارج شده و صفحه درخواستی برنامه دوم به حافظه می آید.
- این عمل بصورت متناوب تکرار شده و در چنین شرایطی، مجموعه کاري هیچ برنامه اي در حافظه نخواهد بود و کارآیی سیستم به شدت پایین می آید.
- این حالت نا متعادل را کوفتگی یا کوبیدگی می نامند.

نحوه کنترل کوبیدگی

روش اول: با محدود کردن چند برنامگی در یک سطح بی خطر یا کنترل بار

۱) سطح چند برنامگی را در یک حد ثابت و بی خطر نگه داریم.

۲) سطح چند برنامگی بر اساس تخمین مجموعه کار تعیین گردد.

فقط کارهایی با هم اجرا شوند که حاصل جمع مجموعه هاي كاري آنها از اندازه RAM بیشتر نباشد.

۳) به کمک تکنیک فرکانس خطاي صفحهPage Fault Frequency سطح چند برنامگی بصورت خودکار تنظیم شود.

هرگاه تعداد نقص صفحه از یک حد فراتر رود، سطح چند برنامگی کاهش می یابد و برعکس.

نحوه کنترل کوبیدگی

روش دوم: جلوگیری از کوبیدگی توسط کنترل مداخله برنامه ها

– مداخله یعنی اینکه یک برنامه باعث شود صفحاتی از مجموعه کاری برنامه دیگر از حافظه خارج شود.

۱) استفاده از تخصیص قاب محلی

۲) اولویت بندی برنامه ها

- در این حالت برنامه های با اولویت بالا می توانند صفحات برنامه های با اولویت پایین را از حافظه بیرون کنند.

اندازه صفحه

عوامل مؤثر در تعیین اندازه صفحه عبارتند از:

۱) هر چه صفحه کوچکتر باشد، تکه تکه شدن داخلی کمتر می شود.

۲) اگر صفحه بزرگ باشد، ممکن است همه محتویات آن استفاده نشود. ولی اگر کوچک باشد احتمال استفاده همه محتواي آن وجود دارد. لذا صفحه بزرگتر، فضاي حافظه بیشتري را به هدر می دهد.

۳) هر چه صفحات کوچکتر باشند، تعدادشان بیشتر شده و در نتیجه حجم جدول صفحه زیاد خواهد شد.

۴) زمان تبادل صفحه بین حافظه و دیسک به اندازه صفحه بستگی ندارد. پس هر چه صفحه بزرگتر باشد، اطلاعات بیشتري در یک مبادله جابجا می شود.

نکته ! در عمل امروزه اندازه صفحات بزرگتر شده اند چون قدرت و سرعت CPU ها و اندازه حافظه افزایش یافته است.

سیاست پاکسازی:

- بر عکس سیاست واکشی است.

- باید تعیین شود که یک صفحه تغییر یافته چه موقعی باید در حافظه ثانوی نوشته شود.

سیاست پاکسازی:

- **پاکسازی درخواستی:** فقط زمانی یک صفحه در حافظه مجازی نوشته می شود که برای جایگزینی انتخاب شده باشد.

- **پیش پاکسازی:** از ابتدا قاب هایشان را در حافظه مجازی می نویسد.

پایان جلسه