سازمان سما

وابسته به دانشگاه آزاد اسلامی

دانشگاه سما واحد حاجی آباد

سیستم عامل

منبع : سیستم عامل دکتر شیر افکن

حمیدرضا رضاپور

WWW.HREZAPOUR.IR

فصل ۶ (بخش سوم)

مدیریت حافظه

صفحه بندی چند سطحی

سیستم های مدرن از فضای آدرس منطقی گسترده ای پشتیبانی می کنند.

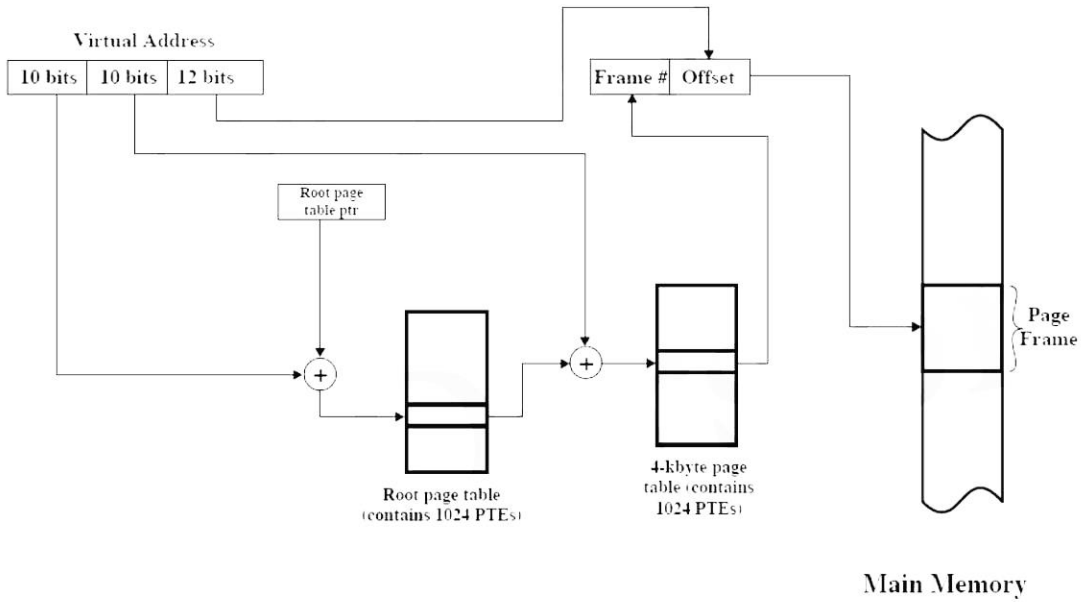
در این محیط ها درایه های جدول صفحه زیاد خواهند شد و در نتیجه اندازه جدول صفحه بزرگ شده و چون نمی خواهیم جدول صفحه را به طور همجوار در حافظه اصلی تخصیص دهیم، باید آن را به فضاهای کوچکتری تقسیم کرد.

برای این کار می توان از الگوی صفحه بندی دو سطحی استفاده کرد که خود جدول صفحه، نیز صفحه بندی می شود.

مثال

سیستمی از یک جدول صفحه دو سطحی و آدرس های مجازی ۳۲ بیتی استفاده می کند. اندازه صفحات 2^{12} بایت است. (۴ کیلو بایت)

ایندکس سطح اول و دوم ۱۰ بیتی است، پس هر جدول صفحه در این دو سطح دارای ۱۰۲۴ درایه است.



یک فرایند ۲۰ کیلو بایتی نیاز به ۵ صفحه دارد. اگر از صفحه بندی تک سطحی استفاده کنیم، جدول صفحه دارای 2^{20} درایه خواهد بود که فقط ۵ درایه آن استفاده خواهد شد .

مثال

سیستمی با مشخصات زیر را در نظر بگیرید. به دلیل آنکه هر جدول باید داخل یک صفحه جای گیرد، یک جدول صفحه چند سطحی استفاده شده است. چند سطح مورد نیاز است؟

فضای آدرس پذیر مجازی= ۱۸ بیتی و ظرفیت صفحات = ۶۴ بایت و اندازه هر مدخل جدول صفحه = ۴ بایت

اندازه هر صفحه ۶۴ بایت : ۶ بیت آفست

12	OFFSET= 6

بنابراین ۱۲ بیت برای شماره صفحه جا داریم. حال باید ببینیم که این ۱۲ بیت را به چند قسمت، تقسیم کنیم.

تعداد مدخل های هر جدول صفحه که می تواند در یک صفحه جای گیرد: $\frac{64}{4} = 2^4$

پس اندیس های PT_i برابر ۴ بیت است.

آدرس منطقی:

PT1= 4	PT2= 4	PT3= 4	OFFSET= 6

TLB

Translation Lookaside Buffers

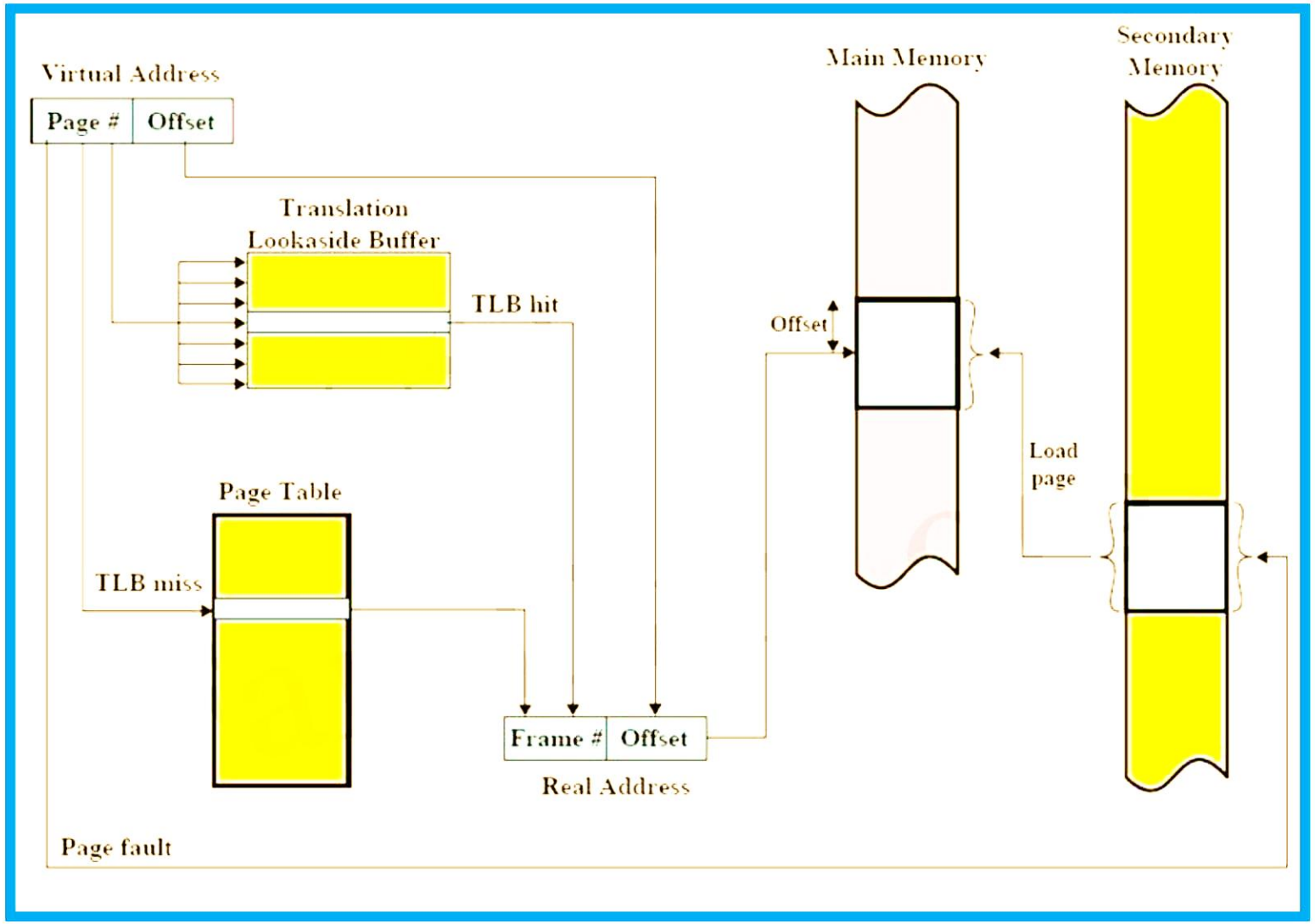
یک سخت افزار جستجوی سریع، کوچک و پنهان

ثباتهای انجمنی TLB را میانگیرهای دم دستی یا بافرهای کناری ترجمه نیز می نامند.

نسبت اصابت (hit ratio) : درصد تعداد دفعاتی که شماره صفحه در ثباتهای انجمنی پیدا شود.

مثلا نسبت اصابت ٪ 80 ، یعنی که ٪ 80 از تعداد دفعاتی که به ثباتهای انجمنی مراجعه کردیم، شماره صفحه پیدا شده است.

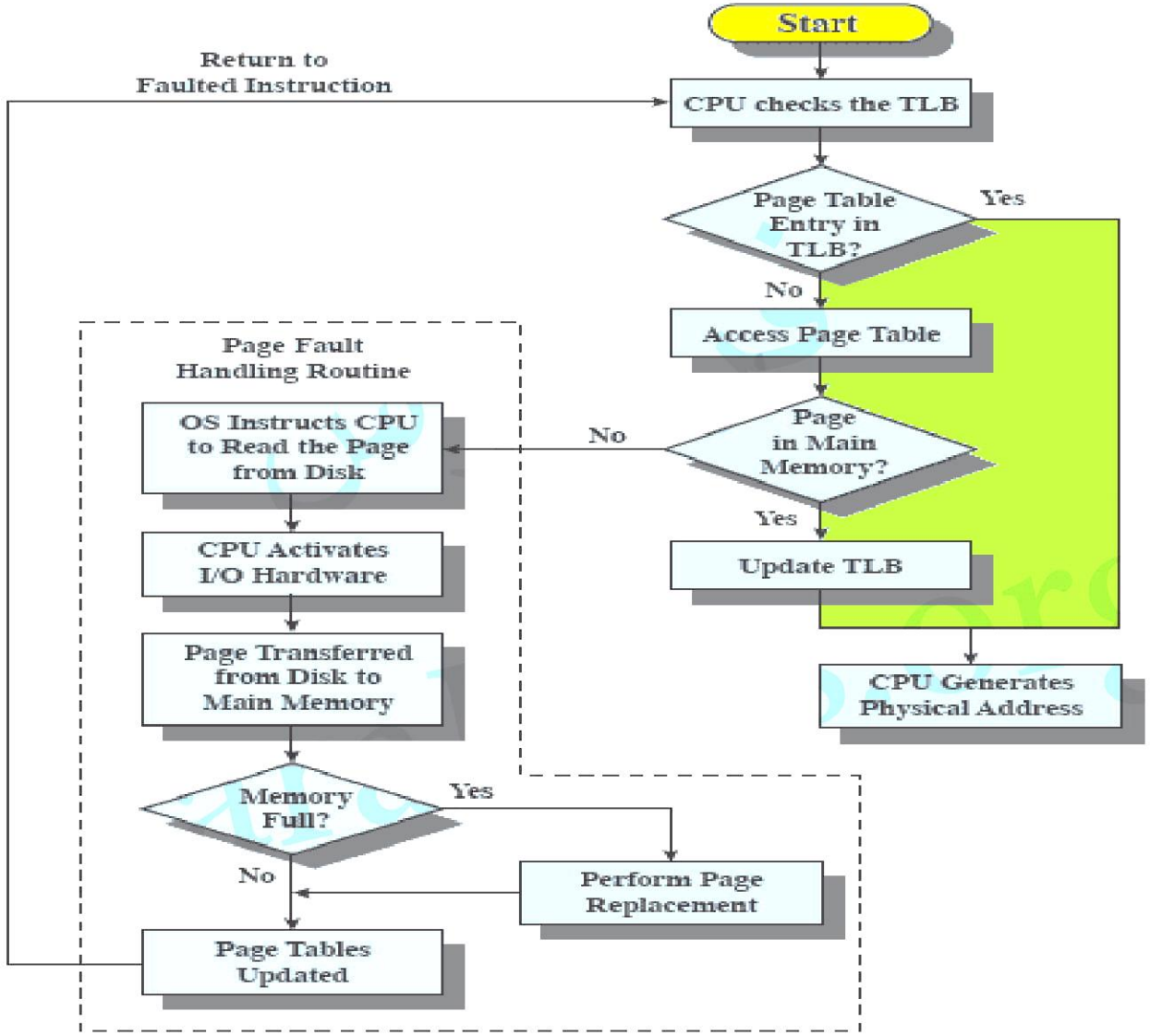
عملیات صفحه بندی و TLB



در صفحه بندی تعداد اندکی از ورودی های جدول صفحه در TLB قرار می گیرد. وقتی آدرسی تولید می شود، اگر شماره صفحه در ثباتهای انجمنی پیدا شود، از شماره قاب آن برای دستیابی به حافظه استفاده می شود. اگر موجود نباشد، ارجاع به جدول صفحه باید انجام گرفته و شماره صفحه و شماره قاب به TLB اضافه می شود تا در مراجعات بعدی به سرعت پیدا شود.

اگر TLB پر باشد، سیستم عامل باید یکی از ورودی ها را حذف کرده و ورودی جدید را به جای آن قرار می دهد.

هر بار که تعویض بستر (انتخاب جدول صفحه جدید) رخ دهد، باید TLB پاک شود تا فرایند بعدی که می خواهد اجرا شود، از اطلاعات ترجمه ای نادرست استفاده نکند.


Start
Return to
Faulted Instruction
CPU checks the TLB
Page Table
Entry in
TLB?
Yes
No
Access Page Table
Page Fault
Handling Routine
OS Instructs CPU
to Read the Page
from Disk
No
Page
in Main
Memory?
Yes
CPU Activates
I/O Hardware
Update TLB
Page Transferred
from Disk to
Main Memory
CPU Generates
Physical Address
Memory
Full?
Yes
No
Perform Page
Replacement
Page Tables
Updated

زمان موثر دسترسی

مجموع زمان ترجمه آدرس و زمان دسترسی به کلمه مورد نظر در حافظه اصلی پس از محاسبه آدرس فیزیکی .

زمان موثر دسترسی با فرض استفاده از TLB و جدول صفحه ساده

$$T_{Access} = T_{Translation} + T_{Mem}$$

$$T_{Translation} = H \times T_{TLB} + (1-H) \times (T_{TLB} + T_{Mem}) = T_{TLB} + (1-H) \times T_{Mem}$$

مثال

با فرض اینکه جدول صفحه در حافظه ذخیره شده باشد و ٪ 85 از ارجاعات به حافظه از طریق TLB انجام شود و هزینه هر ارجاع حافظه ۲۵۰ نانو ثانیه و ارجاع به TLB با هزینه ۵ نانو ثانیه انجام شود، با فرض عدم رخداد نقصان صفحه و عدم توازی عملیات در معماری سیستم مذکور، هر ارجاع به حافظه به طور متوسط چقدر طول می کشد؟

$$T_{Translation} = T_{TLB} + (1 - H) \times T_{Mem}$$
$$= 5 + (0.15 \times 250) = 42.5 \quad ns$$

$$T_{Access} = T_{Translation} + T_{Mem} = 42.5 + 250 = 292.5ns$$

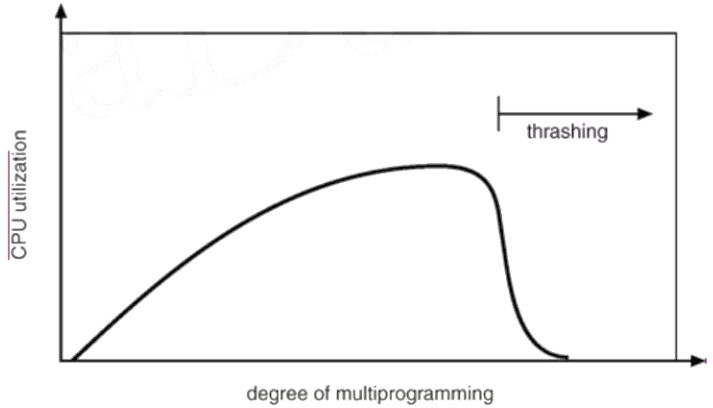
کوبیدگی (Thrashing)

تعداد کل قاب صفحاتی که می توان در یک سیستم به مجموعه فرایندها داد، مقداری ثابت و متناسب با اندازه حافظه اصلی است.

کاهش تعداد قاب ها باعث می شود که یک فرایند نتواند مجموعه صفحاتی که در طی یک زمان از آنها استفاده می کند را در حافظه اصلی بارگذاری کند. در نتیجه تعداد نقص صفحه ها زیاد می شود.

در این وضعیت، زمان CPU به جای اجرای فرایندها، صرف مبادله صفحه ها می شود و کارایی سیستم کاهش می یابد.

به این پدیده، کوبیدگی (Thrashing) می گویند.



مدل مجموعه کاری (Working Sets)

در هر لحظه زمانی (t)، مجموعه ای از صفحات وجود دارد که در k مراجعه اخیر به حافظه، مورد استفاده واقع شده اند. به این مجموعه، مجموعه کاری می گویند .

سیستم عامل با نظارت به مجموعه کاری هر فرایند، به آن قاب کافی اختصاص می دهد.

اگر مجموع اندازه های مجموعه کاری فرایندها، زیادتر از تعداد کل قاب ها شود، پردازش معلق شده و از کوبیدگی پیشگیری می شود.

مثال

ارجاعات یک برنامه به صفحات حافظه :

0 , 1 , 4 , 2 , 0 , 2 , 6 , 5 , 1 , 2 , 3 , 2 , 1 , 2 , 6 , 2 , 1 , 3 , 6 , 2

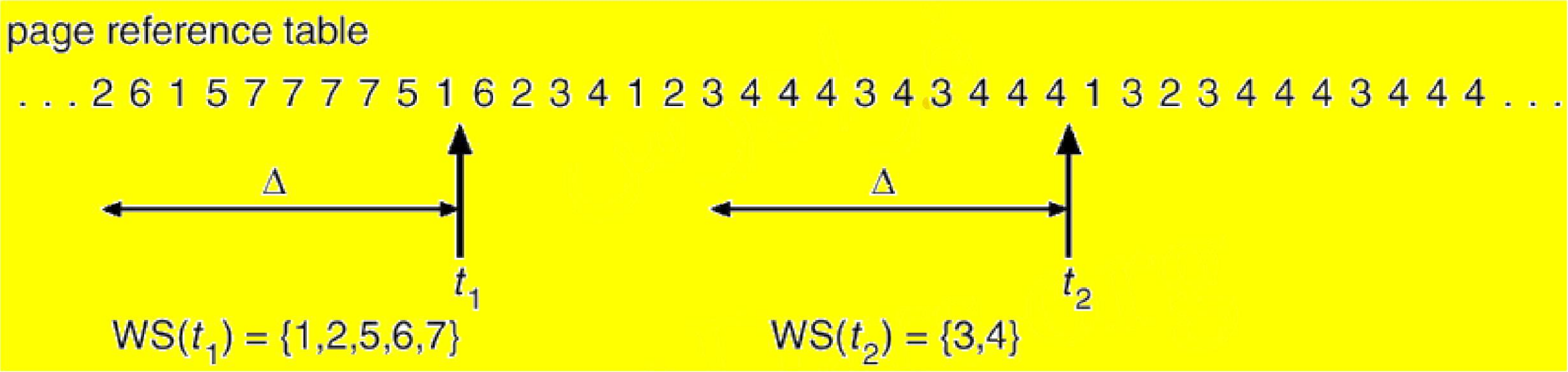
T : زمان بین سیزدهمین و چهاردهمین ارجاع

Δ : پنج ارجاع

0 , 1 , 4 , 2 , 0 , 2 , 6 , 5 , 1 , 2 , 3 , 2 , 1 , 2 , 6 , 2 , 1 , 3 , 6 , 2

مجموعه کاری : $\{1,2,3\}$

مثال


page reference table
. . . 2 6 1 5 7 7 7 7 5 1 6 2 3 4 1 2 3 4 4 4 3 4 3 4 4 4 1 3 2 3 4 4 4 3 4 4 4 . . .
Δ
Δ
t_1
t_2
$WS(t_1) = \{1,2,5,6,7\}$
$WS(t_2) = \{3,4\}$

پیش صفحه بندی (PrePaging)

سیاست واکشی در مورد تعیین زمانی که یک صفحه باید به داخل حافظه آورده شود، دو رویکرد دارد:

۱- صفحه بندی درخواستی ۲- پیش صفحه بندی

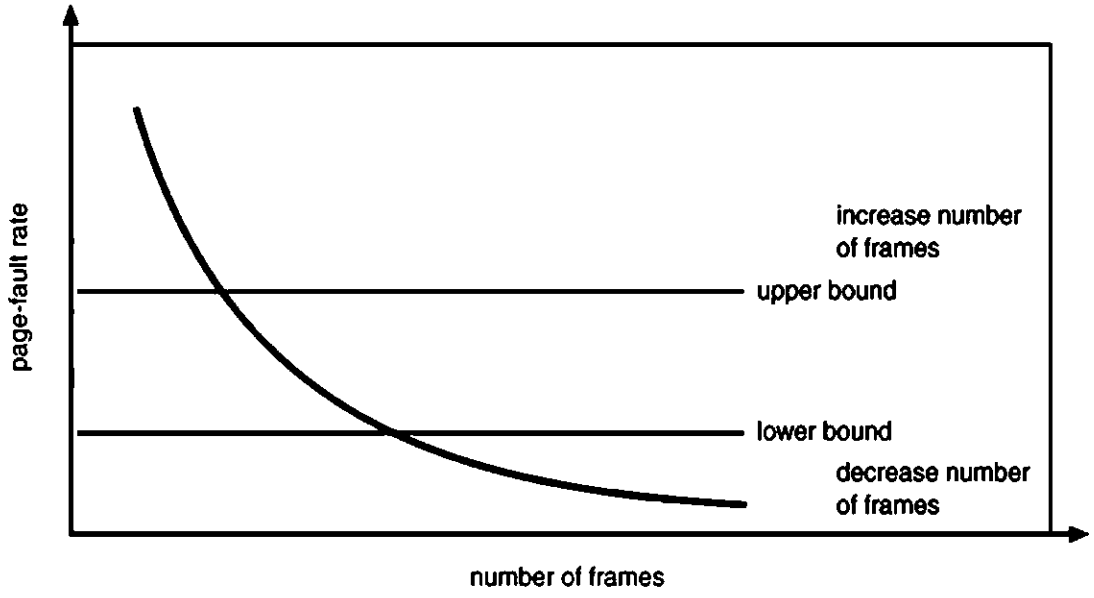
در صفحه بندی درخواستی، فقط زمانی که مراجعه ای به مکانی از یک صفحه شود، آن صفحه به حافظه اصلی آورده می شود. ولی در پیش صفحه بندی، صفحه هایی به غیر از آنچه به وسیله خطای صفحه درخواست شده نیز به داخل آورده می شوند.

یکی از خصوصیات صفحه بندی درخواستی، رخ دادن تعداد زیادی خطای صفحه در شروع یک کار می باشد، که پیش صفحه بندی سعی به جلوگیری از این صفحه بندی زیاد دارد و از ابتدا تمام صفحات مورد نیاز فرایند را به صورت یکجا، به حافظه می آورد.

سیاست پیش صفحه بندی می تواند یا در زمان شروع فرایند به کار گرفته شود یا هر بار که یک خطای صفحه رخ می دهد.

Page Fault Frequency

در این روش کران بالا و پایین برای نرخ خطای صفحه تعیین می شود. اگر نرخ خطای صفحه از کران بالا، شود، قاب دیگری به آن فرایند تخصیص داده می شود. اگر نرخ خطای صفحه از کران پایین شود، قابی از فرایند پس گرفته می شود.



بنابراین PFF سعی می کند که نرخ صفحه بندی را بین دو حد قابل قبول نگه دارد تا از تفریط(کوبیدگی) و افراط(اتلاف حافظه) جلوگیری شود.

پایان