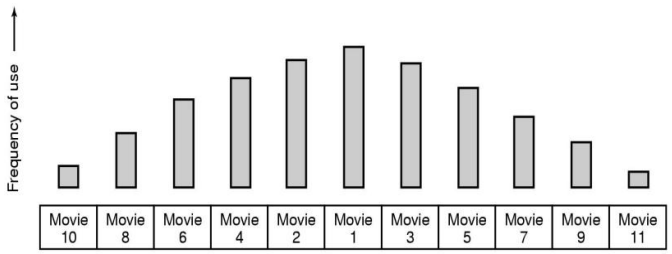
قرار دادن فایل‌ها در یک دیسک

در این حالت فرض می‌شود که تعداد فیلم‌های متعددی وجود دارد که قرار است بر روی یک سرویس دهنده ویدئو قرار گیرد و در این هنگام ممکن است کاربران مختلف برای مشاهده فیلم‌های مختلف وارد سیستم شوند، بنابراین اگر فرض کنیم که سرویس دهنده ویدئو دارای تنها یک دیسک است آن گاه برای آنکه بتواند اطلاعات مربوط به فیلم های مختلف را از فایل‌های مختلف بخواند زمان زیادی برای آنکه head بتواند بین فایل‌های مختلف حرکت کند، اتلاف خواهد شد. اگر این فایل‌ها به صورت تصادفی بر روی دیسک قرار گرفته باشند یعنی مکان قرار گرفتن آنها بر روی دیسک از قاعده خاصی پیروی نکند. بهترین پیشنهادی که برای حل این مشکل ارائه می شود استفاده از عمومیت یا میزان محبوبیت فیلم هاست. به عبارت دیگر بدانیم یک فیلم چقدر مورد علاقه افراد مختلف است؟ با توجه به اینکه فیلم‌های مختلف میزان استقبال مختلفی دارند، لذا در این راستا قانون ZIF بیان می کند که احتمال اینکه مشتری بعدی یا کاربر بعدی آیتمی را انتخاب کند که در رده محبوبیت K قرار دارد برابر با $\frac{c}{k}$ است (c یک ضریب نرمالسازی است) بنابراین اگر ما دارای n فایل (n movie) باشیم، آنگاه رابطه زیر در مورد آنها برقرار است:

$$C/1+C/2+C/3+\ldots\ldots\ldots+C/N=1$$

با توجه به این رابطه یکی از تکنیک‌هایی که می‌توان برای قرار دادن فایل های مختلف بر روی یک دیسک استفاده کرد، تکنیک سازماندهی لوله‌ای(organ pipe) است.در تکنیک organ pipe محبوب ترین فیلم در وسط دیسک قرار داده می شود.



با توجه به شکل، فیلم ۱ یک فیلم محبوب است و وسط دیسک قرار گرفته است. میله های عمودی که در این شکل نشان داده شده است میزان محبوبیت و توجه به هر فیلم را نشان می دهد. در تکنیک organ pipe اتفاقی که خواهد افتاد این است که پس از اینکه محبوب ترین فیلم در وسط قرار گرفت، سایر فیلم‌ها که محبوبیت کمتری دارند در دو طرف این فیلم قرار می‌گیرند. به عبارت دیگر چیدمان فایل‌های مربوط به فیلم ها به گونه ای خواهد بود که یک مثلث را تشکیل می‌دهند، به صورتی که وسط قاعده مثلث محبوب ترین فیلم قرار گرفته است. در این تکنیک سعی خواهد شد که head دیسک همواره در وسط دیسک قرار بگیرد برای اینکه دسترسی به فایل‌هایی که محبوبیت بالاتری دارند سریع تر انجام شود.

قرار دادن فایل ها در چند دیسک

در این مبحث قرار گرفتن فایل های متعدد بر روی دیسک های مختلف بررسی خواهد شد. یکی از تکنیک‌هایی که در حضور بیش از یک دیسک مورد بحث قرار می گیرد، تکنیک RAID است. تکنیک RAID دارای سطوح مختلفی است، در واقع شش سطح RAID برای یک سیستم قابل تعریف است. به طور کلی می توان عنوان کرد که RAID سعی می کند، قابلیت اطمینان مربوط به داده ها را افزایش دهد، گرچه ممکن است این افزایش قابلیت اطمینان با پرداخت هزینه بر روی افزایش کارایی باشد در تکنیک‌های RAID حتی امکان بازیابی خطا نیز وجود دارد. گرچه در سیستم های مالتی مدیا به بحث تصحیح خطا معمولاً پرداخته نمی‌شود. در تکنیک‌هایRAID این امکان وجود دارد که حتی با از دست رفتن یک یا چند دیسک باز هم بخشی از داده ها قابل دسترسی و قابل بازیابی باشد. علاوه بر اینکه در برخی از سطوح RAID ، علاوه بر امکان تشخیص خطا با توجه به نوع کدگذاری امکان تصحیح خطا هم وجود دارد.

به طور کلی مسألهای که باید به آن اشاره کرد این است که معمولاً برای طراحی سیستم های RAID و کنترل ارسال داده ها و دریافت آنها به یک سیستم RAID از RAID controller ها استفاده می‌شود، که می‌توانند به صورت نرم افزاری و یا سخت افزاری باشند. معمولاً در سیستم‌های حرفه‌ای از RAID controllerهای سخت افزاری استفاده می‌شود. نکته‌ی حائز اهمیت این است که خود این controllerهای RAID معمولاً bottleneck سیستم‌ها هستند، یعنی برای ارسال تمامی درخواست‌ها به RAID controller، این قسمت از سیستم خود یک bottleneck خواهد بود، یعنی حجم زیادی از

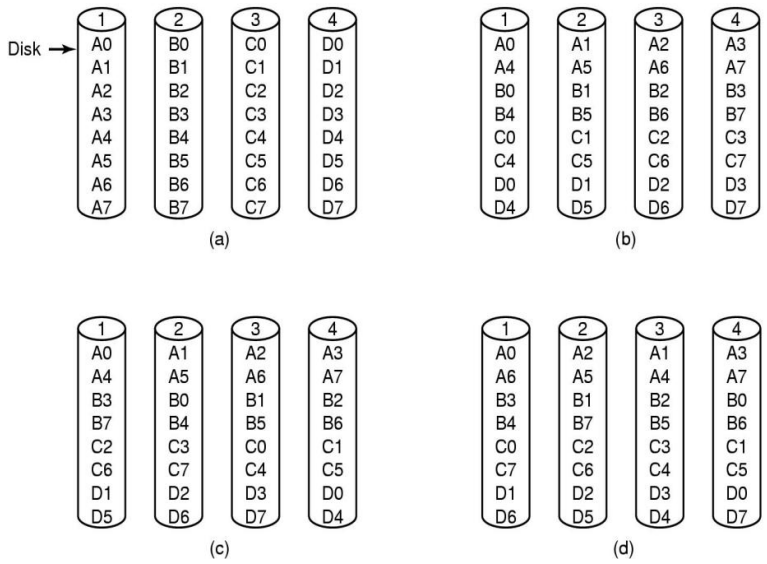
درخواست ها در یک صف مقابل controller قرار می گیرند تا پردازش شوند. لذا در مورد سیستم‌های RAID باید چاره ای اندیشید.

جنگل‌های دیسکی

نکته دیگر در نحوه قرار دادن فایل‌ها بر دیسک‌های مختلف، استفاده از مفهوم جنگل‌های دیسکی یا Disk farms هاست. جنگل‌های دیسکی به ٤ گروه عمده تقسیم می شوند:

No striping, same striping, staggered striping, random striping

در اینجا وقتی که مفهوم stripe مطرح می‌شود، به این معنی است که ما داده های مختلف یک فایل را بر روی دیسک‌های مختلف قرار دهیم. به عبارت دیگر در تکنیک striping یک فایل بر روی دیسک های متعددی کشیده شده و یا قرار می گیرد.



No striping

با توجه به شکل، در حالت No striping یعنی شکل(a)، امکان striping وجود ندارد، یعنی داده‌های یک فایل بر روی دیسک‌های متعددی نیست و داده های هر فایل فقط بر روی یک دیسک قرار دارد.

همانطور که در این شکل می بینید داده های مربوط به فایل A همگی بر روی دیسک ۱ قرار گرفته اند، داده های مربوط به فایل B همگی بر روی دیسک ۲ قرار گرفته‌اند و به همین ترتیب. بنابراین فایل‌های مختلف بر روی دیسک های مختلف قرار می‌گیرند. این به این معنی نیست که بر روی یک دیسک تنها یک فایل داریم. در این شکل تنها برای سادگی به ازاء هر دیسک تنها یک فایل نشان داده شده است.

Same striping

در حالت Same striping یعنی شکل(b)، عمل striping یعنی قرار گرفتن داده ها، بر روی دیسک های مختلف انجام می شود. اما همانطور که از اسم این روش برمی آید، روش برای همه فایل ها مشابه است. یعنی تکنیک چیدمان برای تمام فایل ها یکی است. با توجه به شکل، بر روی دیسک ۱ بیت A_0 یا تکه صفر از فایل A ، بر روی دیسک۲ تکه ۱ از فایل A و به همین ترتیب قرار گرفته اند. همین الگو برای فایل B تکرار شده است، یعنی B_0 بر روی دیسک۱ وB_1 بر روی دیسک ۲ و B_2 بر روی دیسک ۳ قرار دارد. این روند برای C و D نیز تکرار شده است. بنابراین این تکنیک برای همه فایل ها از الگوی مشخص در توزیع داده ها استفاده می کند.

Staggered striping

در این تکنیک یعنی شکل(c)، سعی می شود داده ها در عین حال که با الگوی مشخصی بر روی دیسک‌ها قرار می گیرند، این الگو از یک فایل به فایل دیگر تغییر نماید. به عنوان مثال با توجه به آنچه در شکل (c) می بینید، A_0 بر روی دیسک ۱ و A_1 بر روی دیسک ۲ و به همین ترتیب قرار دارد. اما برای فایل B از تکنیک دیگری استفاده شده است، یعنی قسمت مربوط به B_0 به دیسک ۲ منتقل شده است .به عبارت دیگر نسبت به فایل A یک شیفت به جلو داشته ایم. و همین بحث منجر به این شده است که C_0 در دیسک ۳ قرار گیرد و D_0 در دیسک ٤ قرار گیرد. در واقع تکنیک مشابه عملیات Rotation در سطح یک بایت است.

Random Striping

در این تکنیک، الگوی مشخصی برای قرار گرفتن بلاک ها وجود ندارد و هر بلاک مربوط به هر فایل به صورت تصادفی بر روی دیسک قرار می گیرد. لذا در شکل (d) نیز می بینید که نظم مشخصی برای داده‌هایی که بر روی دیسک ۱ قرار دارند وجود ندارد. حال به بررسی مزایا و معایب این روش ها در Disk Frames یا جنگل های دیسکی می پردازیم.

در تکنیک No striping می دانیم که اگر یکی از دیسک ها خراب شود در واقع تمام فیلم هایی که بر روی آن دیسک قرار گرفته اند کاملاً از بین خواهد رفت. از طرف دیگر این تکنیک باعث خواهد شد که میزان بارکاری متعادل نباشد، یعنی چنانچه یک فیلم دارای محبوبیت یا چند فیلم دارای محبوبیت بر روی یک دیسک قرار بگیرند میزان دسترسی به آن دیسک از دسترسی به سایر دیسک ها به مراتب بیشتر خواهد بود.

بر تکنیک Same striping یک ایراد می توان وارد کرد و آن این است که نقطه صفر تمام فایل ها بر روی دیسک اول است و این مسأله باعث عدم توازن خواهد شد. در حالت Staggered ، بار مربوط به فایل ها تقریباً متعادل شده است و یکی از تکنیک های مناسب است. اما مسأله ای که در اینجا باید به آن پرداخت این است که وقتی ما می گوییم Striping و داده ها Stripe می شوند منظور از Stripe چیست؟

وقتی در شکل‌های قبلی بحث A_0، A_1 یا A_5 را داشتیم، A_0 و A_1 چه نوع داده ای هستند؟ یا چه حجمی از داده ها را شامل می شوند؟ برای این مفهوم دو ایده وجود دارد و آن این است که بخش های داده‌ای، فریم ها هستند یا بلاک ها. حال مزایا و معایبی این حالت را بررسی می‌کنیم. اگر فرض کنیم که هر یک از داده ها یک فریم هستند اتفاقی که خواهد افتاد این است که Load یا میزان بار فریم ها متوازن خواهد شد. اما تقریباً می توان گفت که بهینه سازی در سرعت، اتفاق نمی افتد. اما در مورد Block Striping ، یعنی اگر فرض کنیم که هر یک از A_0، A_1 تا A_n ها خود یک بلاک از فریم ها هستند آنگاه به این نتیجه خواهیم رسید که می توان با استفاده از بلاک Striping مجموعه درخواست های مطرح شده برای قسمت‌های

مختلف یک فیلم را که در موقعیت نزدیک به هم قرار دارند را به یکباره پاسخ دهیم. در عین حال وقتی که مجموعه درخواست ها کامل شود آنگاه می توانیم مدعی شویم که یک قسمت مناسب و یک حجم خوبی از فایل آماده پردازش است. از سوی دیگر امکان پردازش درخواست‌ها به صورت موازی افزایش می‌یابد.

نکته بعدی که در جنگهای دیسکی مورد توجه قرار می گیرد، این است که در صورت استفاده از روش striping چند دیسک را در یک stripe استفاده کنیم؟

از دو روش می توان استفاده نمود که عبارتند از:

۱- Wide striping

۲- Narrow striping

در wide striping هر فیلم روی تمام دیسک ها stripe می‌شود و احتمالاً در این حالت فیلم ها از دیسک مشابهی در روند ذخیره سازی خود استفاده نمی‌کنند، به خصوص اگر تعداد دیسکها زیاد باشد. ایرادی که بر این روش وارد است این است که اگر یک دیسک دچار مشکل شود تمامی فیلم ها دچار نقصان می شوند. زیرا تمامی فیلم ها بر روی تمام دیسک‌ها قرار می‌گیرند.

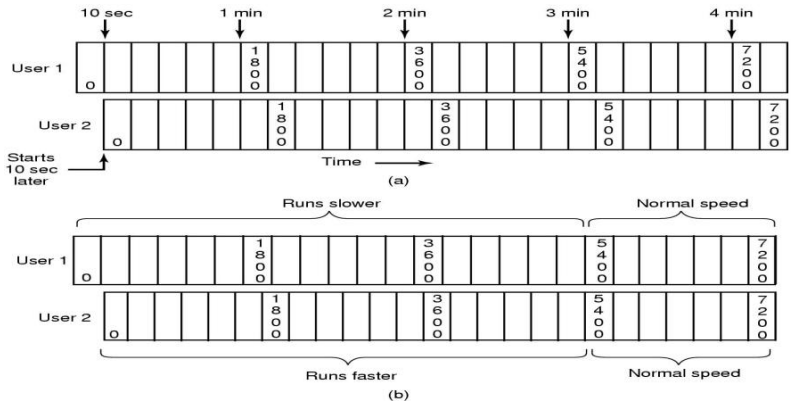
حالت دیگر striping narrow یا باریک است. دراین حالت دیسک‌ها به گروه‌های کوچکی تقسیم می‌شوند و هر فیلم بر روی یک گروه stripe می‌شود. بنابراین یک فیلم بر روی تمام دیسک‌ها قرار ندارد. نکته ای که در اینجا حائز اهمیت است، بحث hot spot است. یعنی ممکن است یک پارتیشن یا یک گروه، گروهی باشد که بر حسب تصادف تعداد فیلم هایی که در آن قرارگرفته‌اند، همه از فیلم‌های دارای محبوبیت (popular) باشند. بنابراین این پارتیشن تبدیل به یک bottleneck خواهد شد، در صورتی که پارتیشن های دیگر حجم کمی از درخواست‌ها را مدیریت می‌کنند.

Caching

مبحث دیگری که در مورد سیستم‌های چندرسانه ای بررسی خواهد شد، مبحث caching است. در سیستم های سنتی و در file caching های مربوط به سیستم های سنتی یکی از روش هایی که مورد

استفاده قرار می‌گرفت روش LRU بود. به این صورت که در یک بافر به عنوان cache داشتیم که معمولاً cache یک حافظه با سرعت دسترسی بسیار بالاست و درخواست ها به جای اینکه مستقیماً از روی دیسک پاسخ داده شوند احیاناً از روی cache مربوط به دیسک پاسخ داده می شد، به شرطی که داده های مربوطه در cache قرار داشته باشند. با آمدن یک درخواست، بررسی می‌شد آیا miss رخ داده است یا hit) hit به معنای این است که داده ی مربوط در cache قرار دارد.) . پس از این بررسی در صورت وجود hit درخواست مورد نظر پاسخ داده می شد و دیسک هیچ عملیات خاصی انجام نمی‌داد. در صورت بروز miss داده ها از دیسک منتقل می‌شد. در اینجا علاوه بر اینکه داده مورد نظر منتقل می‌شود همسایه‌های مربوط به این داده نیز منتقل خواهند شد، به این امید که بر طبق اصل locality درآینده، این همسایه ها مورد دستیابی قرار گیرند. بدیهی است که برای انتقال این داده ها باید قسمتی از داده های قبلی که در cache قرار گرفته اند حذف شوند. تکنیک LRU برای حذف این داده‌ها مورد استفاده قرار می‌گرفت. اما اگر بخواهیم از فایل‌های چند رسانه ای یا multimedia استفاده کنیم تکنیک caching متفاوت خواهد بود. ابتدا مبحث block caching را مورد بررسی قرار می‌دهیم. یعنی با فرض اینکه واحد انتقال داده‌های ما بلاک است. برای block caching اولین وظیفه‌ای که بر عهده سیستم قرار می‌گیرد این است که سیستم باید بررسی کند که چه فیلمی یک بیننده دارد و یا بیش از یک بیننده دارد. آنگاه استراتژی مورد بحث این است که همه block های فیلمی که بیش از یک بیننده دارد بعد از اولین استفاده cache می شوند، تا سایر بینندگان نیز بتوانند آن را استفاده کنند و block های دیگری را در اینجا cache نخواهیم کرد.

اما نکته ای که مهم است این است که معمولاً بینندگان یک فیلم لزوماً همگی در یک لحظه شروع به دیدن فیلم نمی کنند. یعنی ممکن است ما برای یک فیلم استریم های متعددی داشته باشیم که دارای اختلاف زمانی هستند. لذا برای اینکه سودمندی های caching را بتوانیم برای این حالت بدست آوریم باید سعی کنیم که استریم های مختلف را که با یکدیگر دارای اختلاف زمانی هستند ادغام کنیم. لذا لازم است که frame rate این دو را با هم تغییر دهیم تا یکسان شود.



با توجه به شکل، user1 و user2 هر دو در حال مشاهده یک فیلم هستند. اما به دلیل اینکه زمان آغاز user2 ده ثانیه بعد از زمان آغاز user1 بوده است اختلاف زمانی در نمایش فریم‌ها وجود دارد و شاید تکنیک caching در اینجا جواب مناسب تولید نکند. برای اینکه این دو استریم را باهم سازگار کنیم، ساده ترین کار این است که یکی از آنها را اندکی سریعتر اجرا کرده و یکی را اندکی کندتر اجرا کنیم، تا این دو بهم برسند و سپس پس از اینکه این دو به هم رسیدند از سرعت نرمال استفاده کنیم.

در شکل این قضیه را می بینید، در واقع user1 اندکی کندتر اجرا خواهد شد و فریم ها با نرخ کمتری برای وی ارسال می شود، چون که این کاربر زودتر درخواست داده است و نرخ ارسال فریم ها به user2 سریعتر خواهد بود و این مسأله باعث می شود که پس از مدتی این دو کاربر با یکدیگر همزمان شوند و از اینجا به بعد بحث caching برای هر دو آنها ممکن خواهد بود.

file caching

در این مبحث نکته ای که باید به آن پرداخت این است که بیشتر فیلم ها بر روی DVDها یا نوارها قرار می‌گیرند. بنابراین در مفهوم file caching اینگونه بحث می شود که هنگامی که یک فیلم مورد نیاز است محتویات آن را بر روی دیسک کپی کنیم. می دانیم که سرعت دسترسی به هارد دیسک سریعتر از سایر نوع ها است. اگرچه کپی کردن بر روی دیسک زمان زیادی را در زمان start up خواهد داشت، اما این مسأله باعث خواهد شد که در لحظات بعدی زمان انتظار کاهش یابد. استفاده از این تکنیک باعث می‌شود که

بیشتر سرعت را افزایش دهیم. در ادامه می توان عنوان کرد که فیلم های بسیار محبوب و بسیار پر استفاده را دیگر بر روی DVD منتقل نمی کنیم و همواره آنها را بر روی دیسک قرار می دهیم.

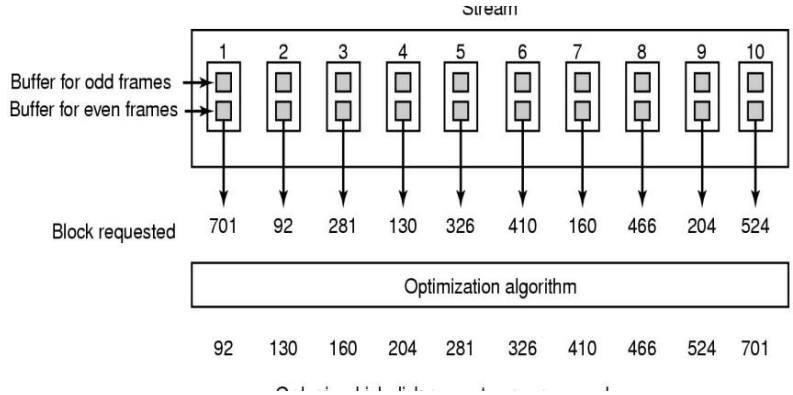
Disk Scheduling (زمان بندی دیسک)

درخواستهای متعددی برای انجام شدن توسط دیسک به یک کنترلر دیسک می رسد بحثی که در اینجا مطرح می‌شود این است که این درخواست‌ها چگونه مدیریت شوند؟ و ابتدا به کدامیک اولویت بیشتری داده شود تا زمان اجراء مناسب ایجاد شود؟ در سیستم های چند رسانه ای که بحث dead line نیز مطرح است، ضمن اینکه باید کارایی آن افزایش پیدا کند تعداد فرایندهایی که dead line آنها miss می شود یا از دست می رود نیز باید کاهش پیدا کند. در سیستم های سنتی، الگوریتم‌های مختلفی به عنوان الگوریتم‌های scheduling بررسی شده است، مانند الگوریتم scan ، الگوریتم C-Scan والگوریتم look و موارد مشابه آن.

static scheduling

با تکیه بر سیستم های چندرسانه ای مباحثی ساده در مورد disk scheduling بررسی خواهد شد. اولین تکنیک disk scheduling تکنیک static scheduling است. در این تکنیک مفهومی به نام round یا دور وجود دارد. در round یا دور که معمولاً برابر با زمان یک فریم است به این شکل عمل می شود که در ابتدای هر round همه ی فرآیندهایی که در سیستم وجود دارد باید درخواست های مربوط به دیسک را به دیسک ارسال کنند. قبل از اینکه round آغاز شود درخواست ها پذیرفته می شود و پس از آن دیگر درخواستی پذیرفته نمی شود، مگر اینکه یک round به پایان برسد. درخواست های ارائه شده در static scheduling در قالب ترتیب بهینه مرتب می‌شوند. یکی از بهترین معیارهای مرتب شدن شماره ی سیلندر درخواست ها می باشد. باید دقت کرد که در اینجا منظور از درخواست ها، درخواست های مربوط به دیسک است که یک درخواست مربوط به دیسک معمولاً شامل شماره‌ی سیلندر، شماره سکتور، میزان داده ی مورد نظر و در سیستم های چند رسانه ای dead line مربوطه است.

پس از مرتب شدن داده ها، درخواستها معمولاً بر حسب ترتیب قرار گرفتن آنها، انجام می شوند. بدیهی است که تعداد فریم ها در هر دور با توجه به تعداد درخواست ها متفاوت خواهد بود. در این سیستم معمولاً از تکنیک double buffering نیز استفاده می شود.



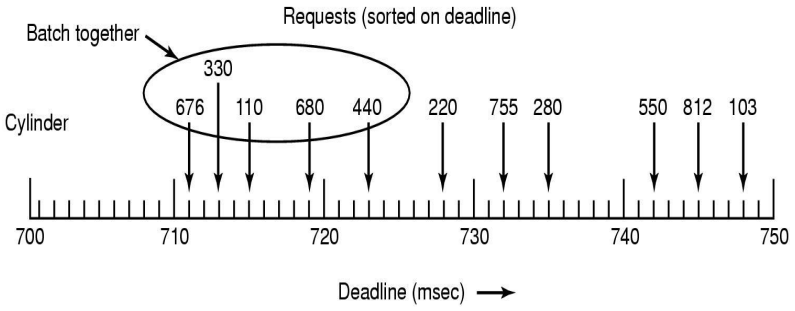
با توجه به شکل یک استریم وجود دارد که درخواست های متعددی برای آن داده شده است. چون تکنیک، تکنیک double Buffering است در داخل هر درخواست از دو بافر استفاده می شود، یکی برای فریم های زوج و دیگری برای فریم های فرد. در ابتدای یک round همه درخواست ها گرفته شده است و سپس الگوریتم های بهینه آنها را مرتب کرده و به ترتیبی که مرتب شده اند آنها را اجرا می کند. در اینجا معیار مرتب شدن شماره ی سیلندر آنها بوده است. بنابراین ابتدا درخواستی که برای سیلندر شماره ی ۹۲ است اجرا می شود و...

زمان بندی پویا (dynamic scheduling)

تکنیک دوم مرتب سازی dynamic scheduling است. در اینجا فرض می کنیم که dead line بسیار حائز اهمیت است. بنابراین هر درخواست خواندن دارای یک dead line است. یکی از ساده ترین الگوریتم ها، Scan-EDF است. در الگوریتم Scan-EDF که به صورت dynamic scheduling عمل می کند، درخواست هایی که dead line شان نزدیک یکدیگر هستند انتخاب شده و باهم یک گروه می

سازند. در داخل این گروه فرایندها و درخواست ها بر حسب شماره ی سیلندرشان انتخاب می شوند. در واقع ترکیبی از الگوریتم Scan و EDF در اینجا پیشنهاد شده است.

در این تکنیک الگوریتم هنگامی درخواست های جدید را می پذیرد که پذیرش این درخواست ها safe (مطمئن) باشد. مطمئن بودن به دو صورت قابل بررسی است. یکی بر حسب نوع درخواست ها و دیگری برحسب وضعیت سیستم و باری که در سیستم وجود دارد.



با توجه به شکل، نمونه ای از اجرای Scan-EDF نشان داده شده است. کسانی که dead line آنها نزدیک به یکدیگر است در یک گروه قرار می‌گیرند، و محور افقی deadline را نشان می دهد. بنابراین تعدادی از درخواست‌ها که شامل ۵ درخواست است در یک گروه قرار گرفته اند. حال این گروه به صورت scan درخواست های خود را انجام می‌دهد. در بین این مجموعه داده ها با توجه به موقعیت جاری دیسک و جهت حرکت آن یکی از درخواست های ٦٧٦، ٣٣٠، ١١٠، ٦٨٠ و یا ٤٤٠ اولین درخواست خواهد بود. بدیهی است که اگر جهت هد به سمت track یا شیار با شماره ی بیشتر باشد و موقعیت جاری آن۲۰۰ باشد، اولین درخواستی که طبق الگوریتم scan انجام می شود، درخواست مربوط به سیلندر شماره ۳۳۰ است.