ذخیره و بازیابی اطلاعات

(سیستم و ساختار فایل‌ها)

مؤلف : فرشید شیرافکن

دانشجوی دکترای بیوانفورماتیک دانشگاه تهران

ناشر: سازمان علمی آموزش فرادرس

بزرگترین پلتفرم آموزش آنلاین ایران

وب: www.faradars.org

تقدیم به:

روح پاک پدرم

- فرشید شیرافکن

سخن ناشر

در عین تمام نقدهای وارد شده به کنکور، هنوز راه حلی عملی که در جمیع جوانب، بهتر از سبک کوتاه و چند گزینه‌ای سؤالات باشد؛ ارائه نشده است. همین موضوع، کنکور را به ویژه کنکور کارشناسی ارشد به عنوان یک آزمون متمرکز و سراسری، از اهمیت دوچندانی برخوردار می‌کند.

یکی از آسیب‌های همراه با این آزمون سراسری این است که فضای رقابتی آن با ایجاد مؤسسات گوناگون، به سرعت از فضای یک رقابت علمی تبدیل به فضای رقابت اقتصادی می‌شود؛ به گونه‌ای که هزینه سرسام آور کلاس‌ها، دوره‌ها و منابع مرتبط با آزمون، از عهده بسیاری از دانشجویان خارج می‌شود. دانشجویانی که در عین استعداد تحصیلی بالا، در میدان رقابت مالی تحمیلی، در عین تمام شایستگی‌های خود، قدرت ادامه مسیر را از دست می‌دهند یا به نتیجه‌ای که در فضای مساوی مالی برای همه باید به آن می‌رسیدند، دست نمی‌یابند.

یکی از اهداف و آرمان‌های فرادرس به عنوان بزرگ‌ترین پروژه آموزش دانشگاهی اجرا شده بر بستر وب کشور، ایجاد دسترسی همگانی و یکسان به آموزش و دانش؛ مستقل از جغرافیا، زمان و سطح مالی دانشجویان بوده است. سیاست کاری فرادرس در راستای این آرمان، انتشار آموزش‌های ویدئویی تخصصی و دانشگاهی رایگان و یا بسیار کم هزینه، با تدریس مجرب‌ترین اساتید داخل و خارج کشور بوده است.

ما با انتشار رایگان این کتاب (به همراه نزدیک به ده کتاب رایگان دیگر) یکی از گام های دیگر خود را در راستای آرمان فرادرس برداشتیم. کتاب حاضر که حاصل نزدیک به یک دهه تدریس و پژوهش و تألیف مؤلف و مدرس فرادرس می‌باشد؛ در عین هزینه‌های بالای تألیف و آماده‌سازی، به جای انتشار و فروش؛ با تأمین مالی و سرمایه‌گذاری فرادرس به عنوان ناشر، به صورت کاملاً رایگان منتشر می‌شود. ما در گام‌های بعدی نیز تلاش خواهیم کرد که تا هر جا بتوانیم، حتی شده یک کتاب مرجع دیگر و بیشتر را با پرداخت هزینه، آزادسازی کرده و به صورت رایگان منتشر کنیم.

مؤلفین و ناشرینی که تمایل به واگذاری حق انتشار کتاب خود به فرادرس را دارند، می توانند با ایمیل ebooks@faradars.org مکاتبه نمایند. ما این کتاب‌ها را با پرداخت هزینه تألیف به مدرس و ناشر، به صورت رایگان منتشر خواهیم کرد تا همه دانشجویان مستقل از سطح مالی، به منابع مفید آزمون دسترسی داشته باشند. همچنین اگر ایده و نظری در خصوص کتاب‌های رایگان فرادرس داشته باشید، خوشحال می‌شویم که آن را با ایمیل ebooks@faradars.org مطرح نمایید.

سازمان علمی آموزش فرادرس

بزرگترین پلتفرم آموزش آنلاین ایران

وب: www.faradars.org

منبع مطالعاتی تکمیلی مرتبط با این کتاب

آموزش ذخیره و بازیابی اطلاعات

با افزایش روز افزون اطلاعات، فرآیند ذخیره، بازیابی و استخراج اطلاعات از اهمیت ویژه‌ای برخوردار است. در درس ذخیره و بازیابی اطلاعات معماری روش ذخیره‌سازی، پیکربندی ورودی/خروجی، عملکرد دیسک و سامانه‌های ذخیره‌ساز، پیکربندی دیسک، تکنیک‌های ورودی/ خروجی رسانه‌ها، مفهوم سیستم فایل، شاخص‌بندی و درهم‌سازی مورد بحث قرار می‌گیرد.

آموزش ذخیره و بازیابی اطلاعات، توسط مهندس فرشید شیرافکن، یکی از بهترین مدرسین مسلط به این مباحث، ارائه شده است.

مدرس: مهندس فرشید شیر افکن

مدت زمان: ۲۰ ساعت

جهت مشاهده آموزش ویدئویی این آموزش – کلیک کنید faradars.org /fvsft106

درباره مدرس

مهندس فرشید شیرافکن کارشناس ارشد مهندسی کامپیوتر گرایش نرم‌افزار است و در حال حاضر دانشجوی دکترای بیوانفورماتیک دانشگاه تهران هستند. ایشان از مدرسین نمونه در زمینه ارائه و آموزش دروس دانشگاهی انتخاب شده‌اند.

ایشان مشاور کنکور هستند و بیش از ۳۰ کتاب در زمینه کنکور رشته کامپیوتر تألیف نموده‌اند. ایشان در حال حاضر به عنوان یکی از برترین مدرسین فرادرس از جهت کمیت و کیفیت دروس ارائه شده، نزدیک به ۲۰ عنوان درسی را در قالب آموزش ویدئویی از طریق فرادرس منتشر کرده‌اند. این مجموعه دروس تا کنون مورد استفاده ده‌ها هزار دانشجوی سراسر کشور قرار گرفته اند.

مشاهده همه آموزش های تدریسی و تالیفی توسط مؤلف کتاب - کلیک کنید.

کتب رایگان دیگر از این مجموعه آموزشی

۱. آموزش زبان برنامه سازی C++ – کلیک کنید (+)
۲. آموزش شیءگرایی در سی پلاس پلاس – کلیک کنید (+)
۳. آموزش پایگاه داده‌ها – کلیک کنید (+)
۴. آموزش ساختمان داده‌ها – کلیک کنید (+)
۵. آموزش سیستم عامل – کلیک کنید (+)
۶. آموزش نظریه زبان‌ها و ماشین – کلیک کنید (+)

برای دانلود رایگان این مجموعه کتب، به لینک زیر مراجعه کنید:

http://faradars.org/computer-engineering-exam

دسته‌بندی موضوعی آموزش‌های فرادرس، در ادامه آمده است:

 برنامه‌نویسی برنامه نویسی – کلیک (+)	 آموزش‌های دانشگاهی و تخصصی آموزش های دانشگاهی و تخصصی – کلیک (+)	 هوش مصنوعی و یادگیری ماشین هوش مصنوعی و یادگیری ماشین – کلیک (+)	 مهندسی برق الکترونیک و روباتیک مهندسی برق الکترونیک و روباتیک – کلیک (+)
 دروس دانشگاهی دروس دانشگاهی – کلیک (+)	 مباحث مشترک مباحث مشترک – کلیک (+)	 مهارت‌های دانشگاهی مهارت های دانشگاهی – کلیک (+)	 نرم‌افزارهای تخصصی نرم افزارهای تخصصی – کلیک (+)
 مهندسی نرم‌افزار مهندسی نرم افزار – کلیک (+)	 نرم‌افزارهای عمومی نرم افزارهای عمومی – کلیک (+)	 طراحی و توسعه وب طراحی و توسعه وب – کلیک (+)	 آموزش‌های عمومی آموزش های عمومی – کلیک (+)

فهرست مطالب

فصل ۱ : نوار مغناطیسی- دیسک مغناطیسی

فصل ۲ : سیستم فایل- تکنیک های بلاک بندی

فصل ۳ : فایل در محیط فیزیکی- مدیریت بلاکهای آزاد- پشتیبان گیری- چگالی لود اولیه- لوکالیتی

فصل ۴ : سطوح نشانی دهی - بافرینگ

فصل ۵ : ظرفیت و نرخ انتقال واقعی نوار - ظرفیت و نرخ انتقال واقعی دیسک- تکنیکهای کاهش s,r

بخش دوم: ساختارهای فایل

فصل ۶ : فایل با ساختار پایل

فصل ۷ : فایل با ساختار ترتیبی

فصل ۸ : ساختار ترتیبی شاخص دار- ساختار چند شاخصی

فصل ۹ : ساختار مستقیم

فصل اول :

نوار مغناطیسی – دیسک مغناطیسی

سیستم کامپیوتری

هر سیستم کامپیوتری از یک کامپیوتر و تعدادی تجهیزات جانبی تشکیل شده است. چنین سیستمی دارای دو محیط درون ماشینی و برون ماشینی می باشد. محیط درون ماشینی از کامپیوتر با اجزاء و عناصر داخلی اش و محیط برون ماشینی از دستگاههای جانبی تشکیل شده است.

محیط درون ماشینی شامل دو قسمت است:

1. **MEMORY** (ROM , RAM , Data Instructions)

2. **CPU** (CU , ALU , Local Memory , Registers , Buffers , Flags , Index , Accumulator)

و محیط برون ماشینی شامل ۴ قسمت می باشد:

1. TERMINALS (Keyboard , Video display)

2. Extended Memory (Hard disk , Floppy disk , DVD , CD)

3. Output Device (Printer , Plotter , Card punches)

4. Input Devices (Laser disk drives , Card readers , Magnetic cassette)

حافظه

هر دستگاهی که قادر به نگهداری اطلاعات باشد طوری که کاربر در هر لحظه بتواند به آنها دستیابی داشته باشد را حافظه می نامند.

خصوصیات حافظه عبارتند از :

نوشتن و خواندن	هر حافظه ای این قابلیت را دارد که در آن بتوان نوشت یا از آن خواند.
نشانی پذیری	به اطلاعات مورد نظر در حافظه می توان نشانی دهی کرد.

دستیابی پذیری	هر حافظه ای از طریق مکانیسم نشانی دهی، مورد دستیابی قرار می گیرد.
زمان دستیابی	مدت زمان بین لحظه صدور دستور خواندن یا نوشتن تا آغاز دستیابی به حافظه مورد نظر.
نرخ انتقال	کمیتی از اطلاعات که در واحد زمان از حافظه قابل انتقال است.
ظرفیت	هر حافظه ای دارای ظرفیتی است که به بیت یا بایت یا اضعاف آنها بیان می شود.

واحدهای حافظه عبارتند از:

$1KB = 2^{10} B,\ 1MB = 2^{20} B\ ,1GB = 2^{30} B\ ,1TB = 2^{40}$ B

سیستمهای ذخیره و بازیابی اطلاعات دو هدف عمده را دنبال می کنند:

۱- صرفه جویی در حافظه ۲- افزایش سرعت عملیات.

انواع حافظه ها

حافظه‌ها به دو دسته تقسیم می‌شوند:

۱- حافظه های درون ماشینی (مانند : ثبات، حافظه پنهان (کش)، حافظه اصلی، حافظه فلاش.)

۲- حافظه های برون ماشینی (مانند : دیسک مغناطیسی،دیسک نوری، نوار مغناطیسی.)

مقایسه حافظه های درون ماشینی و برون ماشینی

۱- ظرفیت حافظه‌های برون ماشینی بیشتر است.

۲- سرعت حافظه‌های برون ماشینی کمتر است.

۳- حافظه‌های درون ماشینی نامانا می باشند. (اطلاعات ذخیره شده می‌تواند از بین برود)

۴- حافظه‌های درون ماشینی گران می باشند.

علت استفاده از حافظه‌های برون ماشینی

۱- محدود بودن ظرفیت حافظه‌های درون ماشینی

۲- گران بودن رسانه‌های ذخیره سازی سریع

۳- عدم لزوم ذخیره همه اطلاعات در حافظه‌های درون ماشینی

۴- نامانا (non volatile) بودن حافظه‌های درون ماشینی

۵- نیاز به دسترسی همروند (concurrent) به داده‌ها توسط چند پروسس.

نوار مغناطیسی

رسانه‌ای پلاستیکی با غشاء مغناطیس شونده بر یک رویه و لغزان بر ریلهایی با ابعاد مختلف و برای پردازش پی در پی (sequential) رکوردها. از کاربردهای نوار می توان مواردی چون تولید نسخه‌های پشتیبان (Backup) و ذخیره سازی در حجم بالا (Archive) را نام برد.

انواع نوار مغناطیسی از نظر تکنولوژی:ریل به ریل، کارتریج، کاست و صوتی.

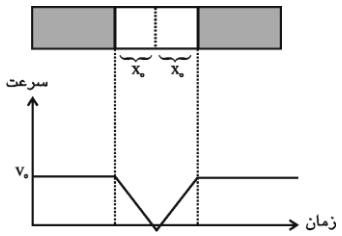
- انواع نوار از نظر تعداد شیار عبارتند از: نوارهای ۷ شیاره و نوارهای ۹ شیاره.
- در هر نوار یکی از شیارها، کنترل کننده پاریتی (Parity) می‌باشند.
- داده ها روی نوار مغناطیسی به صورت رشته های بیتی روی شیارهایی که در سطح نوار وجود دارد، ذخیره می شوند.
- نوار به عنوان رسانه جانبی به کار می رود.

چگالی نوار (Density)

تعداد بیت‌های قابل ضبط در هر اینچ نوار را چگالی نوار می‌گویند. که واحد آن بیت در اینچ(bpi) است، که با توجه به نحوه نشست کاراکترها بر روی شیارها همان بایت در اینچ یا کاراکتر در اینچ است.

گپ (Gap)

فضایی بدون استفاده (هرز) (waste) بین دو رکورد (IRG) یا دو بلاک (IBG) ، که برای متوقف کردن نوار و یا حرکت دوباره آن بکار می‌رود. یعنی برای اینکه نوک خواندن / نوشتن بتواند داده‌های ذخیره شده را حس(sense) کند باید بعد از توقف به سرعتی مطلوب و یکنواخت (سرعت حس) برسد که تکه‌ای از نوار از نوک R/W خواهد گذشت. همچنین در هنگام کاهش سرعت حس تا توقف نیز تکه‌ای از نوار از زیر نوک خواهد گذشت. این دو تکه، گپ را تشکیل می‌دهند.



و می توان نوشت : $t_0 = \frac{2x_0}{V_0} = \frac{IBG}{V_0}$

V_0: سرعت لازم برای حس کردن داده‌ها با واحد اینچ بر ثانیه

IBG : حافظه هرز بین دو بلاک با واحد اینچ یا فوت

t_0 : زمان توقف/ حرکت

به عبارتی : زمان لازم برای صفر شدن سرعت حس یا زمان لازم برای رسیدن به سرعت حس با شروع از وسط گپ .

هر فوت برابر ۱۲ اینچ می باشد.

نحوه ذخیره فایل بر روی نوار

فایل معمولاً به صورت مجموعه‌ای از رکوردها یا بلاکها به طور پی در پی روی نوار ذخیره می‌شود. در یک نوار می توان قسمتی از یک فایل را ذخیره کرد. همچنین می‌توان بیش از یک فایل را ذخیره کرد که هر فایل دارای علامت ابتدا (BOF) و انتها (EOF) می‌باشد. در ذخیره سازی فایل ها روی نوار GAP بین فایل داریم. یک فایل بزرگ می‌تواند روی چند نوار ذخیره شود.

پارامترهای نوار

۱- پارامترهای ظرفیتی (چگالی، طول نوار)

۲- پارامترهای زمانی (نرخ انتقال، زمان حرکت/توقف)

دیسک مغناطیسی

دیسک مغناطیسی رسانه ایست گردان، با امکان دستیابی مستقیم بـه داده هـای ذخیـره شـده کـه بـه آن DASD (Direct Access Device) یعنی رسانه با دسترسی مستقیم می گویند. به عبارتی دیسـک مغناطیسـی صفحه ای است مدور و مغناطیس شونده که حول یک محور عمودی می چرخد . رویـه‌های ایـن صـفحه از غشاء فرو مغناطیسی پوشیده شده که برروی آنها شیارهایی به صورت دایره‌هـای متحـدالمرکز یـا حلزونـی وجود دارد. شیارها از بیرون به درون با شروع از صفر شماره گذاری شده اند. در دیسک سخت، فاصله نـوک R/W با رویه بسیار کم است و هوای تصفیه شده جریان دارد و در دیسک نرم، نوک به رویه می چسبد.

دسته بندی دیسکهای مغناطیسی

دیسکها از نظرات مختلف مانند جنس صفحات، تکنولوژی سـاخت، تعـداد صـفحات، تعـداد لایـه‌ها، تعـداد رویه‌ها دارای تقسیمات مختلفی می باشند. مانند:

۱- دیسک ثابت، دیسک جابجا شدنی

۲- دیسک با نوک ثابت، دیسک با نوک متحرک

۳- دیسک یک رویه ودیسک دو رویه

۴- دیسک تک لایه و دیسک دو لایه

۵- دیسک تک صفحه ای و دیسک چند صفحه ای (Pack)

۶- دیسک مغناطیسی، دیسک نوری و دیسک نوری – مغناطیسی

۷- دیسک سخت (Hard Disk) و دیسک نرم

یک دیسک پک با n صفحه دارای 2n رویه است که 2n-2 رویه آن برای ذخیره بکار می روند.(دو رویـه بالایی و پایینی برای حفاظت می‌باشند و ذخیره‌سازی در آنها انجام نمی‌شود)

نواحی رویه از بیرون به درون عبارتند از : فـرود نـوک (Landing)، شـروع حرکـت (Take off)، احتیـاط بیرونی، شیارهای ضبط داده ، احتیاطی درونی و احتیاطی نهایی.

تقسیمات دیسک

تقسیمات دیسک عبارتند از شیار (Track)، استوانه (Cylinder) و سکتور (Sector).

شیار(track)

محل ضبط بیت های اطلاعات در هر رویه را شیار می گویند. شیارها به صورت دوایر متحد المرکز یا حلزونی می باشند.

- ظرفیت همه شیارها یکسان است ولی چگالی ضبط داده‌ها در شیارهای بیرونی کمتر از چگالی شیارهای درونی است.

استوانه (cylinder)

تمام شیارهای هم شعاع، تشکیل یک استوانه(سیلندر) را می‌دهند. یک دیسک پک به تعداد شیارهای هر رویه، استوانه دارد.

- فایلها با شروع از یک استوانه، استوانه به استوانه روی دیسک ذخیره می‌شوند.
- اهمیت سیلندر در آن است که به همه اطلاعات روی یک سیلندر می‌توان بدون حرکت دادن بازوی نگهدارندهٔ هد خواندن/نوشتن دستیابی داشت.

سکتور (sector)

هر شیار از تعدادی سکتور(قطاع) تشکیل شده است. به عبارتی شیارها به اندازه‌های مساوی به نام سکتور تقسیم می‌شوند. اندازه هر سکتور معمولاً ۵۱۲ بایت است. سکتور بر دو نوع می باشد:

۱- سخت افزاری

سکتور سخت‌افزاری توسط سازنده ایجاد می‌شود.

۲- نرم افزاری

سکتور نرم افزاری از طریق سیستم عامل ایجاد می‌شود.(فرمت نرم افزاری)

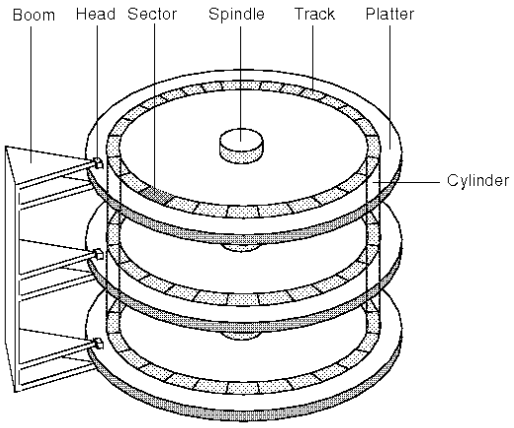
- ابتدای سکتور نرم افزاری باید منطبق بر ابتدای سکتور سخت افزاری باشد.
- اندازه سکتور نرم افزاری بهتر است مضرب صحیحی از اندازه سکتور سخت افزاری باشد.
- اندازه سکتور نرم افزاری می تواند بیشتر یا کمتر از سکتور سخت افزاری باشد.

تعداد سکتور در شیار در دیسکهای سخت جدید، در شیارهای بیرونی بیشتر از شیارهای درونی است که به

این رسانه‌ها IDE می‌گویند. البته تعداد سکتور در همه شیارها از دید سیستم عامل یکسان دیده می‌شود.
(IDE : Integrated Drive Electronics)

سکتور کوچکترین بخش از یک دیسک است که قابل آدرس دهی می‌باشد.

شکل زیر یک دیسک مغناطیسی را نشان می دهد :



کلاستر

کلاستر تعداد ثابتی از سکتورهای پیوسته است و کوچکترین واحدحافظه ای است که می‌توان به یک فایل اختصاص داد. استفاده از کلاسترهای بزرگتر می‌تواند منجر به افزایش کارایی در هنگام پردازش ترتیبی شود. مدیریت فایل تعیین می‌کند که کدام کلاستر از فایل دارای سکتوری است که باید به آن دستیابی شود. جدول تخصیص فایل (FAT)، حاوی لیستی از همه کلاسترهای موجود در یک فایل است که طبق ترتیب منطقی سکتورهای موجود در آن مرتب شده اند.

مولفه های نشانی فیزیکی داده عبارتند از: شماره درایور، شماره استوانه، شماره شیار در استوانه و شماره سکتور.

پارامترهای دیسک

۱- پارامترهای ظرفیتی

اندازه سکتور، تعداد سکتور در شیار، تعداد رویه در استوانه، تعداد شیار در رویه.

۲- پارامترهای زمانی

زمان استوانه جویی، زمان درنگ دوران، سرعت گردش دیسک، نرخ انتقال و زمان استقرار.

زمان استوانه جویی (Seek time)

زمانی که طول می کشد تا نوک خواندن/نوشتن به استوانه ای که داده مورد نظر درآن قرار دارد برسد را زمان استوانه جویی می نامند. متوسط این زمان را با S نمایش می‌دهند و واحد آن میلی ثانیه است. این زمان معمولا بین ۲ تا ۳۰ میلی ثانیه می باشد.

- زمان استوانه جویی(پیگرد)، در دیسک با بازوی ثابت صفر است.

- حرکت بازو (seeking)، کندترین قسمت خواندن اطلاعات از روی دیسک می‌باشد.

- متوسط زمان استوانه جویی (s)،در IBM 3380 برابر 16 ms می باشد.

زمان درنگ دوران (Rotational latency time)

زمانی که طول می کشد تا ابتدای داده مورد نظر در اثر دوران دیسک به زیر نوک R /W برسد. این زمان از حدود ۵ میلی ثانیه تا ۹ میلی ثانیه می باشد.

- متوسط زمان درنگ دورانی را با r (نصف زمان یک دور دیسک)، نمایش می‌دهند. مقدار r بین صفر و زمان چرخش یک دور دیسک قرار دارد $0 \le r \le 2r$)

سرعت گردش دیسک

به کمک سرعت چرخش دیسک می توان زمان یک دور گردش دیسک (2r)، را بدست آورد:

$$2r = \frac{60000}{rpm}$$

واحد سرعت گردش دیسک ، دور در دقیقه (RPM) می باشد.

RPM : Rotation Per Minute

* با فرض اینکه سرعت چرخش یک دیسک ۵۴۰۰ دور در دقیقه باشد، متوسط زمان درنگ دورانی چند میلی ثانیه خواهد بود؟

حل: بین زمان درنگ دورانی(r) و تعداد دور دیسک در دقیقه (rpm) رابطه زیر برقرار است:

$$r = \frac{30000}{rpm} = \frac{30000}{5400} = 5.5$$

■

یک سرعت معمولی برای دوران دیسک ۳۶۰۰ دور در دقیقه است. در این حالت متوسط زمان درنگ دورانی برابر 8.3 میلی ثانیه است. (IBM 3380)

نرخ انتقال

تعداد بایت قابل انتقال در یک ثانیه را نرخ انتقال می گویند که واحدآن بایت در ثانیه می‌باشد.

نرخ انتقال در IBM 3380 برابر $3000\frac{byte}{ms}$ می باشد.

زمان استقرار

وقتی نوک به استوانه ای برده می شود، مدت کوتاهی در حال لرزش است تا استقرار بیابد که به آن زمان استقرار نوک (Setting Time) می گویند. این زمان را به زمان استوانه جویی اضافه می کنند و پارامتر جدا گانه ای محسوب نمی شود.

*با توجه به مشخصات دیسک با بازوی ثابت زیر:

(تعداد سیلندر=۵۰۰، تعداد شیار در سیلندر=۱۸، تعداد بلاک در شیار = ۴۰،ظرفیت بلوک=۲۰۰۰ بایت

، ظرفیت شیار=۸۸۰۰۰ بایت و سرعت چرخش دیسک=۶۰۰۰ دور در دقیقه)

الف- زمان درنگ متوسط چند میلی ثانیه است؟

ب- طول هرگپ چند بایت است؟

ج- ظرفیت دیسک برابر چند مگابایت است؟

د- نرخ انتقال اسمی چند مگابایت در ثانیه است؟

حل:

الف- زمان درنگ دورانی برابر است با :

$$r = \frac{30000}{rpm} \Rightarrow r = \frac{30000}{6000} = 5$$

ب- فضای اشغال شده توسط بلاکها در یک شیار برابر است با:

$$40 \times 2000 = 80000$$

و تعداد بایتهایی که توسط گپ ها در یک شیار اشغال می شود برابر است با :

$$88000 - 80000 = 8000$$

بنابراین فضای اشغال شده توسط یک گپ برابر است با :

$$8000 \div 40 = 200$$

ج- ظرفیت دیسک بدون در نظر گرفتن گپ ها :

$$500 \times 18 \times 40 \times 2000 = 72 \times 10^7 = 720 \times 10^6 \text{ (Byte)}$$

د- نرخ انتقال اسمی توسط سازنده اعلام می شود.

■

زمان دستیابی تصادفی

مدت زمان بین لحظه ای که دستور خواندن یا نوشتن داده می شود و لحظه ای که آغاز داده مورد نظر زیر نوک می رسد، را زمان دستیابی تصادفی (RAT) می گویند. این زمان مجموع دو زمان r و s می باشد و به عبارتی متوسط زمان لازم برای رسیدن به آغاز یک بلاک یا رکورد با مکان مشخص با شروع از مکان نامعین است.

چگالی دیسک

چگالی دیسک نیز از پارامترهای ظرفیتی است ولی در سیستم فایل مستقیما در محاسبه ظرفیت استفاده نمی شود. واحد چگالی دیسک، بیت در اینچ مربع می باشد. افزایش چگالی دیسک از قانون هوگلند تبعیت

می کند. طبق این قانون، چگالی دیسک از رابطه $10^{\frac{SAL-1970}{10}}$ $Mb/inch^2$ (برای تا سال ۲۰۰۰) قابل محاسبه است. مثلا می توان با قرار دادن عدد 1990 به جای متغیر SAL ، چگالی دیسکها را در سال ۱۹۹۰ محاسبه کرد.

دیسکهای نوری

برای افزایش فضای ذخیره‌سازی، سرعت دستیابی و کاهش هزینه از نور به جای مغناطیس استفاده شده است. از انواع دیسکهای نوری می‌توان موارد زیر را نام برد:

1- DVD
2- CD
3- CD-ROM
4- WORM : (Write-Once Read–Many)
5- EOD : (Eraseable Optical Disk)

EOD یا CD-RW، نوعی CD-ROM می باشد که می‌توان چند بار بر روی آن نوشت.

زمان استوانه جویی CD-ROM از دیسک مغناطیسی بسیار بالاتر است.

دیسکهای نوری – مغناطیسی

از تلفیق دو تکنولوژی مغناطیس و نور، دیسکهای نوری– مغناطیسی ایجاد شده اند، که دارای خاصیت دیسک نوری یعنی بالا بودن چگالی و خاصیت دیسک مغناطیسی یعنی قابل پاک شدن می‌باشد.

دو نوع دیگر از دیسکها

۱- دیسکهای با تغییر فاز

رویه (سطح) در دیسکهای با تغییر فاز با تابش اشعه لیزر دو حالت کریستال و یا نامشخص را به خود می گیرد. سرعت آنها دو برابر دیسکهای مغناطیسی– نوری می باشد.

۲- دیسکهای دای پولیمر

رویه در دیسکهای دای پولیمر دو لایه بوده که با لیزر گرم شده و بر آمدگی در آن ایجاد می شود و بعد از سرد کردن آن بر آمدگی ثابت می ماند.

طبله (Drum)

طبله رسانه ای است منطقا معادل دیسک با نوک ثابت و تک استوانه ای . به عبارتی استوانه ای است که شیارهایش در سطح خارجی می باشد. موارد استفاده طبله عبارتند از :

۱- استفاده از طبله به عنوان حافظه اصلی قبل از پیدایش حافظه های چنبره ای .

۲- استفاده از طبله به عنوان حافظه پشتیبیان برای ماشین مجازی.

۳- ایجاد فایلهای موقت بسیار فعال مورد استفاده سیستم عامل ، کامپایلرها و سایر نرم افزارها.

۴- ضبط نرم افزارهای ثابت مانند مانیتورها ، برنامه های متصدی اشتباهات خواندن / نوشتن ، ناظر اشتباه یابی برنامه ها ، کامپایلرها ، برنامه های مرتب سازی.

- معمولا برای هر شیار یک نوک خواندن/نوشتن وجود دارد. البته ممکن است تعداد نوک ها از تعداد شیارها کمتر باشد که در این صورت نوک متحرک است.
- زمان استوانه جویی در طبله های با نوک ثابت، صفر است.
- پارامترهای دیسک در این رسانه نیز وجود دارند،غیر از تعداد استوانه که همیشه یک است.
- ظرفیت طبله از نوار و دیسک کمتر است و سرعت طبله از نوار و دیسک بیشتر است.

فصل ۲:

سیستم فایل – تکنیک های بلاک بندی

سیستم فایل

سیستم فایل (file system)، نرم افزاری است که وظیفه آن ایجاد و مدیریت فایلها می باشد. این نرم افزار دارای چندین لایه است که کاربر از جزئیات داخلی آن بی اطلاع است. قبل از بررسی سیستم فایل اصطلاح های فیلد، رکورد و فایل را بررسی می نماییم.

اعمال اساسی در محیط فیزیکی برای یک سیستم فایل عبارتند از :
مکان یابی، خواندن از رسانه و نوشتن بر روی رسانه.

فیلد

مکان ذخیره‌سازی یک فقره اطلاع (یک واحد معنایی داده و نامدار) را فیلد می نامند. طول فیلد، حداقل یک کاراکتر و حداکثر وابسته به زبان برنامه نویسی و سیستم فایل می باشد.

رکورد

مفهوم رکورد را در سه سطح زیر می توان بررسی کرد:

انتزاعی	مجموعه اطلاعاتی است که در مورد هر یک از نمونه های متمایز یک یا بیش از یک نوع موجودیت از یک محیط عملیاتی در اختیار داریم.
برنامه کاربر (رکورد منطقی)	مجموعه‌ای است با ساختار مشخص که از چندین فیلد تشکیل شده است.

محیط ذخیره سازی (رکورد فیزیکی)	رکورد در سطح فیزیکی از دو قسمت داده‌ای و غیر داده‌ای تشکیل شده است.

بخش غیر داده ای رکورد

در بخش غیر داده‌ای، اطلاعات مورد نیاز سیستم فایل در تعدادی فیلد ذخیره می‌شود، از جمله: طول رکورد، نوع رکورد، اشاره گرها، فلاگهای عملیاتی و حفاظتی. به عبارتی شامل اطلاعاتی است که سیستم فایل، برای پردازش رکورد به آنها نیاز دارد. فیلد های بخش غیر داده ای در سیستم ها و ساختارهای گوناگون فایل، متفاوت است. نام‌های دیگر بخش غیر داده ای عبارتند از : پیشوندی ، کنترلی، سیستمی و متابخش.

چند نکته:

۱- کلید رکورد در بخش داده ای رکورد قرار دارد.(کلید رکورد، صفت خاصه ای(فیلد) است که طول آن حتی الامکان کوتاه و دارای یکتایی مقدار در نمونه‌های مختلف رکورد است.)

۲- وقتی رکوردی دارای طول متغیر است، می‌توان طول آن را در بخش غیر داده‌ای ذخیره کرد.

۳- در حذف منطقی، فلاگ حذف در بخش غیر داده ای فعال می‌شود و بعداً رکورد به طور فیزیکی حذف خواهد شد.

قالبهای رکورد منطقی

رکورد منطقی دارای دو قالب است:

ثابت مکان (Fixed Positional)	در طرح رکورد با قالب ثابت مکان، در هر فیلد فقط مقدار صفت خاصه ذخیره می‌شود و اسم صفت خاصه ذخیره نمی‌شود.
غیر ثابت مکان (Non Fixed Positional)	در طرح رکورد با قالب غیر ثابت مکان هم اسم صفت خاصه

	و هم مقدار صفت ذخیره می‌شود و مکان فیلدها از قبل مشخص نمی‌باشد. در این طرح تعداد، طول و مکان فیلدها در نمونه‌های مختلف یک نوع رکورد، یکسان نمی باشد.

رکورد منطقی از نظر طول به دو دسته با طول ثابت و با طول متغیر، تقسیم می شود.

دلایل متغیر شدن طول یک رکورد

۱- متفاوت بودن تعداد صفات خاصه مورد نیاز برای نمونه‌های مختلف یک نوع موجودیت .

۲- متغیر بودن طول فیلدها

۳- وجود صفت خاصه مرکب چند مقداری (پدیده فقره اطلاع تکرار شونده)

مثال برای حالت اول: برای کارمند متاهل به فیلدهایی مانند نام همسر و تعداد فرزندان نیاز است که برای کارمند مجرد نیاز نمی باشد ، برای حالت دوم ، فیلد آدرس دانشجو. و برای حالت سوم ، فیلد شماره درس برای موجودیت دانشجو که در یک نمونه دارای ۲ مقدار و در نمونه ای دیگر دارای ۶ مقدار است. وجود چنین فیلدی ، فایل را نامسطح می کند.

اشاره گر(نشانه رو)

توسط اشاره گرها می‌توان بین رکوردها ارتباط ساختاری برقرار ساخت و نظم منطقی موجود در برنامه را در محیط فیزیکی برقرار کرد. هر اشاره گر یک مبدا و یک مقصد دارد. از این نظر انواع اشاره گرها عبارتند از:

۱- رکورد به رکورد
۲- بلاک به بلاک
۳- رکورد به بلاک
۴- بلاک به رکورد
۵- فایل به فایل
۶- گروهی از بلاکها به گروهی دیگر

انواع اشاره گر از نظر نوع نشانی عبارتند از: نشانی در سطح فیزیکی، نشانی نسبی و نشانی ضمنی.

فایل

فایل مجموعه‌ای است نامدار از نمونه های مختلف یک یا چند نوع رکورد. در حالت اول فایل را تک نوعی و در حالت دوم چند نوعی می گوییم. البته ممکن است فایل دنباله ای از کاراکترها باشد که از نظر سیستم فایل معنای خاصی ندارد.

مفهوم فایل در معنای عام دارای سه ویژگی زیر می باشد:

۱- اندازه بزرگ (طوری که یکباره در حافظه درون ماشینی نگنجد)

۲- پایائی(داده ها از بین نمی روند ، مگر به درخواست کاربر)(Resistance)

۳- اشتراکی بودن بین تعدادی کاربر مجاز(Shared)

ساختار فایل

ساختار فایل می تواند فیزیکی و یا منطقی باشد:

۱- ساختار منطقی

نشان دهنده سازمانی است که بر اساس آن رکوردهای منطقی گرد هم آمده اند.

۲- ساختار فیزیکی

نشان دهنده چگونگی ذخیره بلاکها در دیسک می باشد. (دید برنامه ساز سیستم نسبت به فایل)

لایه های سیستم فایل

سیستم فایل دارای پنج لایه به صورت زیر می باشد:

شیوه دستیابی (AM)	این لایه، بالاترین لایه سیستم فایل است که واسط بین برنامه کاربردی و سیستم فایل منطقی می‌باشد. توسط این لایه روشی برای دسترسی به رکوردها در اختیار برنامه کاربردی قرار داده می‌شود.
سیستم فایل مجازی (VFS)	این لایه همسطح با لایه شیوه دستیابی می باشد. البته ممکن است لایه شیوه دستیابی تحت این لایه عمل کند. به این معنی که کاربر برنامه ساز، شیوه دستیابی مورد نظر خود را طراحی کرده و این واحد نرم افزاری را به واحد سیستم فایل مجازی پیوند بزند و ساختار مورد نظر خود را ایجاد کند. سیستم فایل مجازی، فایل را دنباله ای از کاراکترها می بیند و ساختار خاصی برای فایل قائل نمی باشد. در سیستمهای عامل جدید، کاربران از طریق لایه مجازی با سیستم فایل کار می کنند.
سیستم فایل منطقی (LFS)	این لایه واسط بین سیستم فایل فیزیکی و لایه شیوه دستیابی می‌باشد. امکان دسترسی کاربران به رکوردها توسط این لایه فراهم می‌شود. این لایه با مفهوم بلاک کار نمی کند. برنامه کاربر با یک شیوه دستیابی با این لایه در تماس است و این لایه درخواستهای کاربر را انجام می دهد.
سیستم فایل فیزیکی (PFS)	این لایه مسئول ذخیره سازی بلاکها روی رسانه خارجی و انتقال آنها از رسانه به بافر یا از بافر به رسانه می باشد. این لایه با محتوای بلاکها و یا ساختار فایل کاری ندارد و در بعضی سیستم ها، بخشی از خود سیستم عامل می باشد.
درایور	پایین ترین لایه است و مستقیما با کنترولر در ارتباط است.

سیستم فایل از دید کاربر

سیستم فایل از دید کاربر، نرم افزاری است که کاربر می تواند توسط آن عملیاتی مانند، ایجاد، حذف، باز کردن، بستن و دستیابی را روی فایل خود انجام دهد. کاربر برای انجام این کارها درگیر عملیات درونی سیستم نمی باشد.

بلاک

بلاک قالبی ساختار یافته متشکل از چند رکورد است. بلاک کمترین مقدار داده قابل مبادله بین بیرون و درون ماشین توسط سیستم فایل در یک عملI/O می باشد.

بلاک بندی

قرار دادن چند رکورد در یک بلاک را بلاک بندی می گویند. تعداد رکوردهای درون بلاک را ضریب بلاک بندی (B_f) و تعداد بلاکهای موجود در یک شیار را ضریب شیار بندی (T_f) می‌گویند.

مزایای بلاک بندی

۱- کاهش گپ‌های بین رکوردها

۲- کاهش زمان پردازش فایل

۳- کاهش دفعات ورودی/خروجی

۴- کاهش زمان اجرای برنامه فایل پرداز

معایب بلاک بندی

۱- مصرف بیشتر حافظه اصلی به علت نیاز به بافر بزرگتر

۲- کار نرم افزاری بیشتر برای بلاک بندی و بلاک گشایی

۳- افزایش احتمال اشتباه در تبادل اطلاعات به علت افزایش مقدار داده منتقل شونده.

حالات نشست بلاک روی دیسک

یک بلاک می تواند به یکی از صورتهای زیر بر روی دیسک قرار بگیرد:

۱- یک شیار

۲- یک سکتور سخت افزاری (یا بخشی از یک سکتور)

۳- ترکیبی از چند سکتور سخت افزاری

۴- بخشی از شیار مشخص شده توسط نرم افزار (سکتور نرم افزاری)

بلاک هم مانند رکورد دارای بخشی پیشوندی است. در بلاکهایی که طول متغیر دارند، طول بلاک در این قسمت ذخیره می‌شود.

روشهای تعیین محدوده رکورد با طول متغیر در بلاک

۱- درج طول در بخش پیشوندی رکورد

۲- درج نشانگر پایان رکورد

۳- ایجاد جدول مکان نما (position table)

در جدول مکان نما برای هر رکورد یک مدخل وجود دارد که در آن مدخل، آدرس نسبی رکورد درج می‌شود.

۴- ایجاد جدول طولها (Length table)

توسط استفاده از تکنیک ایجاد جدول طولها، می‌توان محدوده رکورد با طول متغیر را تعیین کرد. در هر مدخل این جدول ، طول رکورد متناظر ذخیره می‌شود. اگر چند رکورد همجوار دارای طول یکسانی باشند، برای هریک مدخلی ایجاد نمی‌شود و فقط یک مدخل ایجاد شده و طول و تعداد آنها در آن ذخیره می‌شود که این عمل باعث صرفه جویی در حافظه می‌شود. همچنین در این تکنیک وقتی طول رکوردها مضربی از یک عدد است، می‌توان به جای درج طول، مقدار مضرب را درج کرد.

* با فرض داشتن یک بلاک شامل سه رکورد با طولهای ۱۰، ۱۵ و ۷ بایت، روشهای تعیین محدوده رکورد را نشان دهید.

۱- درج نشانگر پایان رکورد

R0	R1	R2	

۲- درج طول در بخش پیشوندی رکورد

10	R0	15	R1	7	R2

۳- ایجاد جدول طولها

R0	R1	R2	7	15	10

۴- ایجاد جدول مکان نما

R0	R1	R2	A2	A1	A0

(که A0 آدرس نسبی رکورد R0 در بلاک ، نسبت به آغاز بلاک است)

*اگر از روش قرار دادن طول هر رکورد در ابتدای آن استفاده شود، با توجه به شکل بلاک زیر، طول رکورد R2 کدام است؟

0	1	2	3	4	5	6	7	8	
3	R1			2	R2		1	R3	.....

حل:

بسیار واضح است که طول رکورد R2 برابر ۲ می باشد، چون طول هر رکورد در ابتدای ان ذخیره شده است.

* اگر برای تعیین محدوده طول رکورد، از روش قرار دادن طول هر رکورد به ترتیب در انتهای بلاک استفاده شود، طول رکورد R2 کدام است؟

0	1	2	3	4	5			
R1			R2		R3	.....	1	2	3

حل:

بسیار واضح است که طول R2 برابر ۲ می باشد، چون طول رکوردها در انتهای بلاک ذخیره شده است.

* اگر از روش تعیین محدوده رکورد بر اساس قرار دادن آدرس نسبی ابتدای رکورد در انتهای بلاک استفاده شود، آنگاه مطابق شکل زیر، طول رکورد R1 و R2 کدام است؟

0	1	2	3	4	5			
R1			R2		R3	.....	5	3	0

حل:

طول رکورد R1 برابر است با تفاضل آدرس شروع دو رکورد R2 و R1 : 3-0=3

و طول رکورد R2 برابر است با تفاضل آدرس شروع دو رکورد R3 و R2 : 5-3=2

* اگر از روش قرار دادن طول هر رکورد در ابتدای آن استفاده شود، با توجه به شکل بلاک زیر، آدرس ابتدای رکورد R2 کدام است؟ (برای ذخیره طول به یک بایت نیاز است)

0	1	2	3	4	5	6	7	8	
3	R1			2	R2		1	R3	.....

حل:

1+3+1=5

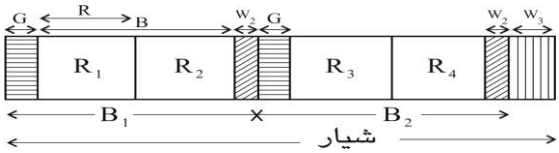
تکنیک‌های بلاک بندی

۱- بلاک بندی رکوردهای با طول ثابت و معمولا یکپاره و گاه دو پاره.

۲ – بلاک بندی رکوردهای با طول متغیر و یکپاره.

۳ – بلاک بندی رکوردهای با طول متغیر و دو پاره.

بلاک بندی رکوردهای با طول ثابت و یکپاره



به طور نمونه چهار رکورد موجود در یک شیار به صورت زیر بلاک بندی می شوند :

در این تکنیک سه نوع حافظه هرز (waste) ایجاد می شود:

W_1 : (گپ بین بلاکها به طول G بایت)

W_2 : (گپ ناشی از نگنجیدن رکوردی دیگر در بلاک) ($0 \le W_2 \le R-1$)

W_3 : (گپ ناشی از نگنجیدن بلاکی دیگر در شیار)

روابطی که در این تکنیک برقرارند:

فاکتور بلاک بندی	$B_f = \left\lfloor \frac{B}{R} \right\rfloor$
حافظه هرز به ازای یک بلاک	$W_B = W_1 + W_2 + \frac{W_3}{T_f}$
حافظه هرز به ازای یک رکورد	$W_R = \frac{W_B}{B_f}$

(B_f : تعداد رکورد در هر بلاک و T_f : تعداد بلاک در هر شیار)

تذکر: اگر متوسط طول W_2 یعنی $\frac{R}{2}$ را در نظر بگیریم، آنگاه داریم:

$$W_B = G + \frac{R}{2} + \frac{W_3}{T_f} \quad , \quad W_R = \frac{1}{B_f}\left(G + \frac{R}{2} + \frac{W3}{T_f}\right)$$

* در یک سیستم فایل، اگر طول هر بلاک برابر ۲۲۰ بایت و طول هر رکورد برابر ۱۰۰ بایت و طول هر شکاف بین بلاکی ۱۶ بایت در نظر گرفته شود، فضای هرز رکوردی کدام است؟

(بلاکها به گونه ای انتخاب شده اند که تمام فضای هر شیار را پر می کنند یعنی $W_3 = 0$.)

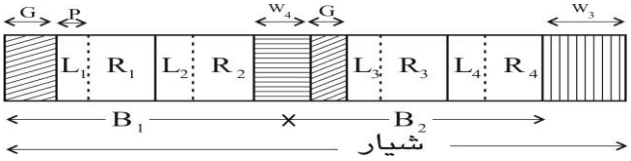
$$B_f = \left\lfloor \frac{B}{R} \right\rfloor = \left\lfloor \frac{220}{100} \right\rfloor = 2$$

$$W_B = 16 + \frac{100}{2} + 0 = 66$$

$$W_R = \frac{66}{2} = 33$$

تعداد $B_f \times T_f$ رکورد در هر شیار وجود دارد.

بلاک بندی رکوردهای با طول متغیر و یکپاره



به طور نمونه چهار رکورد موجود در یک شیار را بلاک بندی می کنیم :

توجه کنید که حافظه هرز W_4 در اثر دو پاره نکردن رکورد R_3 بروز می کند. این نوع حافظه هرز در تکنیک بعدی که R_3 را دو پاره می کنیم، دیگر رخ نخواهد داد.

روابطی که در این تکنیک برقرارند :

فاکتور بلاک بندی	$B_f = \frac{B - W_4}{R + P}$
حافظه هرز به ازای یک بلاک	$W_B = G + B_f.P + W_4 + \frac{W_3}{T_f}$
حافظه هرز به ازای یک رکورد	$W_R = \frac{W_B}{B_f}$

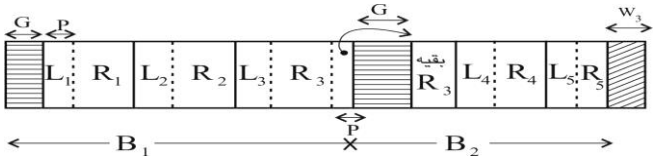
تذکر: اگر متوسط طول W_4 یعنی $\frac{R}{2}$ را در نظر بگیریم ، آنگاه داریم :

$$B_f = \frac{B - R/2}{R + P} \quad , W_R = P + \frac{G + R/2}{B_f} + \frac{W_3}{B_f.T_f}$$

تذکر: منظور از R متوسط طول رکوردها ی موجود در شیار است.

بلاک بندی رکوردها با طول متغیر و دو پاره

در شکل زیر نمونه ای از این بلاک بندی نمایش داده شده است. در این روش R_3 به دو پاره تقسیم شده و در دو بلاک ذخیره می شود که توسط یک فیلد اشاره گر به طول P بایت، پاره اول و دوم را به یکدیگر مرتبط کرده ایم.



روابطی که در این تکنیک برقرارند، به صورت زیر است :

فاکتور بلاک بندی	$B_f = \frac{B-P}{R+P}$
حافظه هرز به ازای یک بلاک	$W_B = G + P + B_f.P + \frac{W_3}{T_f}$
حافظه هرز به ازای یک رکورد	$W_R = \frac{W_B}{B_f} \Rightarrow W_R = P + \frac{G+P}{B_f} + \frac{W_3}{B_f.T_f}$

تذکر: در رابطه B_f، منظور از P صورت کسر، طول فیلد اشاره گر و منظور از P مخرج ، طول فیلد طول است، که برای سادگی این دو برابر در نظر گرفته شده اند.

*در یک فایل با رکوردهایی با طول متغیر که به صورت دو قسمتی در بلاکها قرار گرفته اند، فضای هرز هر بلاک کدام است؟

$(B = 120\ \ byte, G = 15\ \ byte, R = 25\ \ byte, P = 4\ \ byte, W_3 = 30\ \ byte), T_f = 3)$

حل:

$$B_f = \frac{120-4}{25+4} = 4$$

$$W_B = 15 + 4 + 4 \times 4 + \frac{30}{3} = 45$$

■

خواندن رکورد دو پاره در تکنیک سوم به زمان بیشتری نیاز دارد. چون باید دو بلاک خوانده شود. البته اگر فایل باکت بندی شده باشد، می‌توان با یک واکشی، دو بلاک را خواند.

مزیت تکنیک سوم نسبت به تکنیک دوم

۱- حافظه هرز در تکنیک سوم کمتر است.

۲- انعطاف تکنیک سوم بیشتر است.(طول رکورد می‌تواند از طول بلاک بیشتر شود.)

۳- خواندن کل فایل در تکنیک سوم به زمان کمتری نیاز دارد.(حافظه‌های هرز کمتری دارد.)

نرم افزار سیستم فایل تکنیک سوم پیچیده تر است، چون نیاز به مدیریت اشاره گره ها دارد.

پیاده سازی تکنیک اول ساده تر است ولی انعطاف پذیری آن کمتر است.

تذکر: در بررسی تکنیک‌های بلاک بندی، ساده ترین حالات در نظر گرفته شده است، مانند:

الف - قرار گرفتن تمامی R4 در بلاک دوم (در روش دوم) بدون ایجاد حافظه هرز.

ب- قرار گرفتن تمامی R5 در بلاک دوم (در روش سوم) بدون ایجاد حافظه هرز یا دو پاره شدن آن.

چند اصطلاح

۱- باکت : مجموعه‌ای از تعدادی بلاک که طی یک عمل واحد می‌تواند به بافر منتقل شود.

۲- خوشه (cluster) : تعدادی بلاک یا سکتور همجوار.

۳- گسترش : مجموعه ای از شیارهای درون یک استوانه و یا تعدادی استوانه همجوار.

اندازه خوشه برابر است با تعداد بلاکهای خوشه.

باکتهای فایل معمولاًدر محیط فیزیکی ذخیره‌سازی همجوارند.

- گسترش هم در نهایت تعدادی باکت است و با مفهوم باکت بندی مترادف است.
- چهار عمل اصلی که انسان با اطلاع انجام می دهد عبارتند از : ذخیره ، بازیابی، تولید و پردازش.
- فایلها از نظر کاربرد به دو رده عادی و راهنما تقسیم می شوند. فایل عادی حاوی اطلاعات کاربر است و فایل راهنما حاوی اطلاعات سیستم فایل است.

فصل ۳ :

فایل در محیط فیزیکی - مدیریت بلاکهای آزاد
پشتیبان گیری - چگالی لود اولیه - لوکالیتی

فایل در محیط فیزیکی

فایل در دیسک به دو نحوه پیوسته و ناپیوسته می‌تواند ذخیره شود. در نشست پیوسته، فایل در بلاکهای فیزیکی همجوار روی دیسک ذخیره می‌شود و در نشست ناپیوسته، تعدادی بلاک ناهمجوار به فایل تخصیص می یابد.

مزایای طرح پیوسته

۱- سادگی پیاده سازی

با داشتن آدرس بلاک اول روی دیسک، به بلاکهای دیگر می‌توان دسترسی پیدا کرد.

۲- بالا بودن کارائی

کل فایل طی یک عمل واحد از روی دیسک خوانده می شود.

معایب طرح پیوسته

۱- حداکثر اندازه فایل در مرحله ایجاد فایل باید معلوم باشد.

۲- بروز پدیده بندبند شدن فضای دیسک (با تکنیک یکپارچه سازی یا فشرده سازی بر طرف می شود.)

روشهای پیاده سازی طرح نشست ناپیوسته

۱- ایجاد لیست پیوندی

۲- ایجاد لیست پیوندی به همراه جدول راهنما

۳- تکنیک گره I (Index-node)

- دستیابی تصادفی در لیست پیوندی به همراه جدول راهنما سریع تر از روش ایجاد لیست پیوندی می باشد.
- عیب روش ایجاد لیست پیوندی به همراه جدول راهنما در این است که کل جدول باید در حافظه اصلی قرار بگیرد.

ویژگیهای روش لیست پیوندی

۱- سادگی یافتن بلاکهای فایل

۲- کند بودن دستیابی تصادفی به رکوردها

۳- تکه تکه نشدن حافظه

۴- آسان بودن خواندن پی در پی.

تکنیک گره I

برای هر فایل جدول کوچکی به نام گره I ایجاد می‌شود تا بتوان تشخیص داد که کدام بلاک دیسک با کدام فایل مرتبط است. این جدول شامل صفات خاصه فایل و آدرس بلاکهای فیزیکی فایل می‌باشد. برای فایلهای بزرگ در مدخلی از این گره، آدرس بلاکی از دیسک قرار می‌گیرد که حاوی آدرسهای بلاکهای دیگر فایل است. اگر فایل باز هم بزرگتر بود ، مدخل دیگری باید ایجاد گردد و .....

- روش I-node در سیستم عامل Unix به کار می‌رود.
- اگر فایل به طور پیوسته روی واحدهای تخصیص ذخیره شود، آنگاه سیستم فایل برای دسترسی به یک رکورد نیاز به اطلاعات "شروع فایل ، طول رکورد منطقی و طول رکورد فیزیکی" دارد .
- اگر فایل به طور ناپیوسته ذخیره شود، آنگاه سیستم فایل برای دسترسی به یک رکورد دلخواه علاوه بر آدرس شروع فایل به ساختار داده خاصی مثلاً یک فایل شاخص یا یک جدول نیاز دارد.
- از هر دو تخصیص پیوسته و ناپیوسته می‌توان در ذخیره‌سازی فایلهای با ساختار ترتیبی و غیر ترتیبی استفاده کرد.

یک فایل در هنگام ذخیره‌سازی می‌تواند در واحدهای تخصیص مانند شیار، خوشه یا استوانه ذخیره شود که البته هر چه اندازه واحد بزرگتر باشد، حافظه هرز بیشتری روی دیسک ایجاد می‌شود.

مدیریت بلاکهای آزاد

برای مدیریت بلاکهای آزاد ۲ طرح وجود دارد:

۱- ایجاد لیستی از چند بلاک دیسک

۲- استفاده از بیت نقش(Bitmap)

ایجاد لیستی از چند بلاک دیسک

در این روش، در هر بلاک شماره بلاکهای آزاد قرار می گیرد.

*اگر اندازه بلاکها یک کیلو بایت و هر شماره بلاک در 32 بیت نمایش داده شود، در هر بلاک شماره چند بلاک آزاد را می توان ذخیره کرد؟

حل: تعداد شماره هایی که می توان در یک بلاک ذخیره کرد برابر است با:

$$\frac{1KB}{32bit}=\frac{1024\ byte}{4\ byte}=256$$

البته چون یک مدخل بلاک برای ایجاد نشانه رو به بلاک بعدی است، بنابراین در هر بلاک می توان شماره 255 بلاک آزاد را تخصیص داد.

*یک دیسک 200MB که دارای 200K بلاک 1KB است، حداکثر به چند بلاک برای ایجاد لیست نیاز دارد؟

(شماره بلاک در ۴ بایت نمایش داده می شود)

(در انتهای هر بلاک یک مدخل ۴ بایتی برای اشاره به بلاک بعدی در نظر گرفته شده است)

حل:چون هر آدرس ۴ بایتی است، مقدار بایت لازم برای تهیه لیست شماره بلاکهای آزاد برابر است با:

$$200K\times 4=200\times 1024\times 4=819200\ Byte$$

و چون هر بلاک 1KB است، تعداد بلاک لازم برابر است با:

$$\frac{819200}{1024}=800$$

و چون به ازای هر بلاک یک مدخل چهار بایتی برای اشاره به بلاک بعدی داریم، مقدار 3200 بایت (800×4) به فضای مورد نیاز افزوده خواهد شد. بنابراین تعداد بلاکهای اضافی مورد نیاز برای نگهداری این مدخلها برابر است با:

$$\left\lceil \frac{3200}{1024} \right\rceil = 4$$

بنابراین در کل به 800+4 یعنی 804 بلاک نیاز است.

استفاده از بیت نقش(Bitmap)

در این طرح برای دیسکی با n بلاک به n بیت نیاز است. بلاکهای آزاد را با 1 و بلاکهای تخصیص یافته را با 0 نمایش می دهیم.

*در طرح Bitmap، برای دیسکی به ظرفیت 200 مگا بایت با بلاکهای یک کیلو بایتی، به چند بیت حافظه نیاز است؟

حل:

تعداد بلاکها و در نتیجه تعداد بیت ها برابر است با :

$$\frac{200\ \ MB}{1\ \ KB} = \frac{200\times 2^{20}}{1\times 2^{10}} = 200\times 2^{10} = 204800$$

تکنیکهای تهیه پشتیبان

تکنیکهای تولید نسخه پشتیبان از دیسک سخت با ظرفیت زیاد عبارتند از:

۱- آینه سازی (Mirroring)

۲- استفاده از نیمه دو دیسک

۳- تولید دامپهای تدریجی (Incremental dump)

آینه سازی

در این روش از دو یا بیش از دو دیسک استفاده می‌شود. در حالت استفاده از دو دیسک، عمل نوشتن در هر دو دیسک انجام می‌گیرد. نوشتن در دیسک آینه کمی با تأخیر صورت می‌گیرد. ولی عمل خواندن فقط از یک دیسک صورت می‌گیرد. وقتی که یکی از دیسکها خراب شود، از دیسک دیگر می‌توان استفاده کرد.

استفاده از نیمه دو دیسک

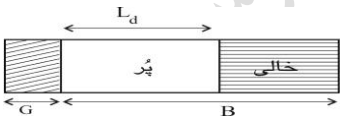
در این روش، هر دیسک به دو نیمه "داده ها و پشتیبان " تقسیم می‌شود. در پایان هر روز کاری نیمه داده ای هر دیسک در قسمت پشتیبان دیسک دیگر کپی می‌شود.

تولید دامپهای تدریجی

در این روش، ضمن ایجاد نسخه پشتیبان بطور هفتگی یا ماهانه، دامپ فایلهایی که از زمان تولید آخرین نسخه پشتیبان تغییر کرده اند، نیز به طور روزانه تولید می شود. این روش به حافظه زیادی نیاز دارد و معمولا از نوار استفاده می شود.

چگالی لود اولیه

می‌توان تمامی فضای بلاک را در لود اولیه پر نکرد و قسمتی از آن را رزرو کرد تا در



عملیات ذخیره‌سازی بعدی از این فضا استفاده کرد. چگالی لود(Loading density) اولیه توسط درصدی از اندازه بلاک مشخص می‌شود که با داشتن نسبت $\frac{L_d}{B} < 1$ بدست می‌آید.

مزایای در نظرگرفتن ناحیه رزرو در هر بلاک

۱- بالا رفتن لوکالیتی رکوردهای فایل

۲- تسهیل انجام بعضی عملیات روی فایل (مثلاً درج رکوردی که در بهنگام سازی طول آن افزایش یافته است.)

معایب در نظر گرفتن ناحیه رزرو در هر بلاک

۱- ناحیه رزرو نوعی حافظه هرز است و اندازه فایل را افزایش می دهد و خواندن کل فایل به زمان بیشتری نیاز پیدا می‌کند.

۲- در صورت عدم توزیع یکنواخت درج رکوردها در بلاکها، در فایل بلاکهای سبکبار(Underload) ایجاد می شود و حافظه هرز در انتهای بعضی بلاکها باقی می ماند.

چگالی لود اولیه را می توان از رابطه $L_d = \frac{n}{b \times B_f} \times 100$ بدست آورد.

*فایلی با ۱۲ رکورد و فاکتور بلاک بندی ۶ و چگالی لود اولیه 50% مفروض است. تعداد بلاکهای اشغال شده توسط فایل کدام است؟

$$b = \left\lceil \frac{n}{l_d \times B_f} \right\rceil = \left\lceil \frac{12}{(\frac{50}{100} \times 6)} \right\rceil = 4$$

به عبارتی اگر از کل فضای هر بلاک استفاده شود، آنگاه به ۲ بلاک نیاز است، ولی چون از نصف فضای هر بلاک استفاده می شود، به ۴ بلاک نیاز است.

*فایلی دارای ۱۰ رکورد است. در صورتیکه این فایل ۴ بلاک را اشغال نماید، چگالی لود اولیه آن کدام است؟ (فاکتور بلاک بندی =۵)

حل:

$$l_d = \frac{n}{b \times B_f} \times 100 = \frac{10}{4 \times 5} \times 100 = \%50$$

لوکالیتی

میزان همسایگی فیزیکی رکوردهای منطقاً همجوار را لوکالیتی رکوردها می‌گویند.

درجات لوکالیتی

موارد زیر لوکالیتی قوی به ضعیف را بیان می‌کند:

۱- رکورد بعدی و رکورد فعلی در یک بلاک هستند و بلاک در بافراست.

۲- رکورد بعدی در بلاک بلافاصله بعدی حاوی رکورد فعلی می باشد و در همان استوانه.

۳- رکورد بعدی در همان استوانه ای قرار دارد که رکورد فعلی قرار دارد.

۴- رکورد بعدی روی استوانه همشماره از دیسک دیگری می باشد.

۵- رکورد بعدی در استوانه همجوار فعلی است.

۶- رکورد بعدی در یک استوانه شناخته شده است و آدرس آن از رکورد فعلی بدست می‌آید.

۷- رکورد بعدی در استوانه ای ناشناخته است و آدرس آن با انجام محاسباتی بدست می‌آید.

۸- رکورد بعدی دراستوانه ای ناشناخته است و آدرس آن با مراجعه به فایل دیگری به دست می‌آید.

۹- رکورد بعدی روی رسانه‌ای است که در حال حاضر روی درایور نیست.

تذکر:در حالت ۴ ، فایل روی چند دیسک توزیع شده است .(تکنولوژی RAID)

- فقط در حالت اول در بازیابی نیاز به عملیات I/O نمی باشد.

- از حالت پنجم به بعد ، معمولا r>0 و همیشه s>0 است.

- در فایل ترتیبی کلیدی (KS) ، اگر فایل در ابتدا روی واحدهای تخصیص همجوار ذخیره شود، همجواری منطقی و فیزیکی در لود اولیه تأمین می‌شود.

- هر چه لوکالیتی رکوردها قویتر باشد، زمان پردازش سریال آنها کمتر خواهد شد، بنابراین در مفهوم لوکالیتی زمان مستتر است.

- لوکالیتی در دوره حیات فایل کاهش می یابد بنابراین فایل باید سازماندهی مجدد شود.

فصل ۴ :

سطوح نشانی دهی - بافرینگ

سطوح نشانی دهی

برنامه فایل پرداز به کمک احکامی از یک زبان برنامه سازی و از طریق سیستم فایل منطقی (یا مجازی نهایتا) به محیط فیزیکی ذخیره‌سازی دستیابی دارد. هر سیستم فایل یک یا چند شیوه دستیابی به داده دارد.

سیستم فایل از دو بخش عمده تشکیل شده است:

۱- سیستم فایل منطقی

این لایه در خواست های برنامه کاربر مانند بازکردن، بستن، خواندن و نوشتن فایل را انجام می دهد.

۲- سیستم فایل فیزیکی

وظیفه این لایه دسترسی فیزیکی به فایلها در محیط فیزیکی می باشد. برای این منظور درخواستهای دریافتی از بخش منطقی را به دستوراتی جهت صدور به کنترولر رسانه تبدیل می کند.

عملیات اساسی که در محیط فیزیکی انجام می شوند عبارتد از : خواندن از رسانه، نوشتن در رسانه و پیگرد (Seek).

تذکر: سطوح برخورد با فایل عبارتند از: سطح برنامه کاربر، سطح سیستم فایل فیزیکی و سطح سیستم فایل منطقی.

سطوح مختلف نشانی دهی

سه سطح نشانی دهی وجود دارد:

۱- در سطح برنامه فایل پرداز

۲- در سطح سیستم فایل منطقی

۳- در سطح سیستم فایل فیزیکی

نشانی دهی در سطح برنامه فایل پرداز

در این سطح به یکی از روشهای زیر به فایل نشانی دهی می شود:

محتوایی (مقداری)	کاربر مقدار یک صفت خاصه (کلید یا غیر کلید)را به عنوان آرگومان جستجو می دهد.
نسبی	کاربر آدرس نسبی رکورد (RRA) را می دهد . کاربر محیط ذخیره سازی را یک ساختار خطی می بیند که در آن هر رکورد شماره ای دارد . شماره رکورد اول یک فرض می شود.
نمادی	کاربر توسط یک نام ، رکورد مورد نظرش را مشخص می‌کند و خود فایل نیز به کمک یک نام در برنامه مشخص می شود.

نشانی دهی در سطح سیستم فایل منطقی

در این سطح از آدرس دهی نسبی، استفاده می‌شود. سیستم فایل منطقی، کل فضای ذخیره‌سازی را به شکل آرایه ای از بلاکها می بیند. آدرس بلاکها از صفر شروع می‌شوند که RBA نام دارد.

سیستم فایل منطقی،آدرس داده شده توسط برنامه را به آدرس نسبی بلاک حاوی رکورد تبدیل می‌کند:

$$RBA_{REC} = RBA_{BOF} + rba_{REC}$$

که آدرس نسبی بلاک حاوی رکورد ، نسبت به ابتدای فایل برابر است با:

$$rba_{REC} = \left\lfloor \frac{(i-1)\times R}{B} \right\rfloor$$

تذکر : مقدار $BYTEOFFSET_{REC}$ برابر $(i-1)\times R$ می باشد.

تذکر: اگر شماره اولین رکورد را به جای یک ، صفر در نظر بگیریم، آنگاه در رابطه های بالا به جای (i-1) باید از i استفاده کرد.

* RBA آدرس نسبی رکورد دهم از کدام بایت شروع می شود و چقدر می باشد؟

$(R = 250, B = 1000, n = 12, RBA_{BOF} = 6)$

حل:

$$RRA = (i-1)\times R = (10-1)\times 250 = 2250$$

$$RBA_{(10)} = 6 + \left\lfloor \frac{(10-1)\times 250}{1000} \right\rfloor = 6 + 2 = 8$$

بنابراین RBA آدرس نسبی رکورد دهم از بایت ۲۲۵۰ ام شروع می شود و برابر ۸ می باشد.

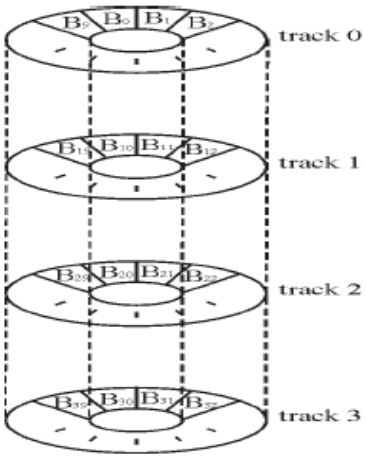
* فایلی با رکوردهای ۲۰ بایتی و بلاکهای ۶۰ بایتی را در دیسکی با ۴ شیار در هر سیلندر و ۱۰ بلاک در هر شیار ذخیره کرده ایم. مقدار RBA را برای هشتمین رکورد بدست آورید؟

حل:

$$RBA = 0 + \left\lfloor \frac{(8-1)\times 20}{60} \right\rfloor = \lfloor 2.3 \rfloor = 2$$

روش دوم:

از آنجا که در هر بلاک، سه رکورد ذخیره می شود، با توجه به شکل سیلندر اول دیسک که در زیر آمده، رکورد ۱و۲و۳ در B_0، رکورد ۴و۵و۶ در B_1 و رکورد ۷و۸و۹ در B_2 ذخیره شده اند. یعنی هشتمین رکورد در B_2 ذخیره شده است.



*در مثال قبل با فرض اینکه $RBA_{BOF} = 5$ فایلی، RBA را برای رکورد هشتم بدست آورید؟

$$RBA = 5 + \left\lfloor \frac{(8-1)\times 20}{60} \right\rfloor = 7$$

به عبارتی چون $RBA_{BOF} = 5$، بنابراین بلاک B_0 را B_5 می نامیم. بنابراین رکورد ۱و۲و۳ در B_5 ، رکورد ۴ و۵ و ۶ در B_6 و رکورد ۷و۸و۹ در B_7 ذخیره شده اند. یعنی هشتمین رکورد در B_7 ذخیره شده است.

نشانی دهی در سطح سیستم فایل فیزیکی

در این سطح، آدرس مکان داده مورد نظر در رسانه مشخص می‌شود. اگر رسانه دیسک باشد، اجزاء آدرس عبارتند از:

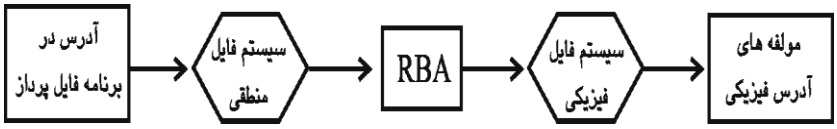
۱- شماره درایور

۲- شماره استوانه

۳- شماره رویه (شیار از استوانه)

۴- شماره سکتور از شیار(یا شماره بلاک)

سیستم فایل فیزیکی، RBA را به آدرس فیزیکی تبدیل می‌کند. برای این تبدیل چنین عمل می کند:



تذکر: تعداد شیار در هر استوانه را با t_i و تعداد بلاک در هر شیار را با b_i نمایش می دهیم . مشخص است که $t_i \times b_i$ نشان دهنده ظرفیت یک سیلندر به بلاک است.

نحوه محاسبه اجزاء آدرس

شماره استوانه	$Cyl\# = RBA \ div \ (t_i \times b_i)$
شماره شیار	$Trk\# = [RBA \mod (t_i \times b_i)] \ div \ b_i$
شماره بلاک	$Blk\# = RBA \mod b_i$

تذکر: در فرمولهای جدول بالا، در حالت کلی باید به جای RBA از $RBA - RBA_{Beginofdevice}$ استفاده شود.

(در روابط بالا فرض کردیم که : $RBA_{BeginOfDevice} = 0$)

تذکر : شماره اولین شیار هر سیلندر صفر می باشد.

* فایلی با رکوردهای ۲۰ بایتی و بلاکهای ۴۰ بایتی را در دیسکی با ۴ شیار در هر سیلندر و ۱۰ بلاک در هر شیار ذخیره کرده ایم. مقادیر CYL#,TRK#,BLK# را برای رکورد ۴۴ ام بدست آورید؟

حل :

ابتدا RBA را بدست می آوریم:

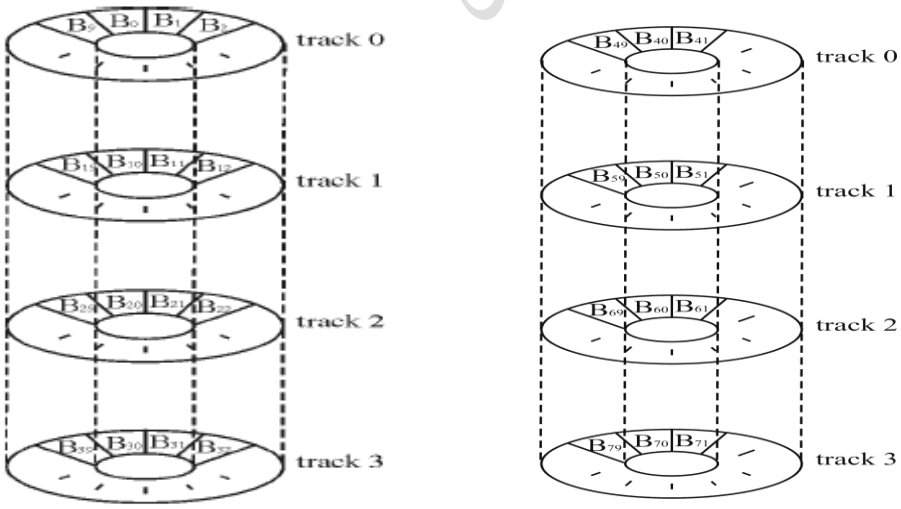
$$RBA_{(44)} = 0 + \left\lfloor \frac{(44-1)\times 20}{40} \right\rfloor = \left\lfloor \frac{43}{2} \right\rfloor = 21$$

$$cyl\# = 21 \ \ div \ \ (10\times 4) = 0$$

$$trk\# = [21 \ \ \mathrm{mod} \ \ (10\times 4)] \ \ div \ \ 10 = 2$$

$$blk\# = 21 \ \ \mathrm{mod} \ \ 10 = 1$$

در شکل زیر دو سیلندر از دیسک نشان داده شده است. اولین سیلندر (سیلندر با شماره صفر) شامل ۴۰ بلاک از شماره ۰ تا ۳۹ و دومین سیلندر (سیلندر با شماره یک) شامل ۴۰ بلاک از شماره های ۴۰ تا ۷۹ می باشد. در شکل مشخص است که رکورد شماره ۴۴ ام با RBA=21 ، در شیار شماره دو مربوط به سیلندر شماره صفر قرار دارد.



همچنین با توجه به شکل به سادگی مشخص است که BLK# این رکورد برابر یک است، چون متناظر با بلاک شماره یک از آن شیار است. به عنوان مثال BLK# برای بلاک با هر یک از شماره های ۱و۱۱و۲۱و۳۱ برابر ۱ می باشد، چون شماره بلاک آنها در شیار مربوط به آنها برابر یک می باشد.■

* محدوده آدرس نسبی بلاک در دیسکی با مشخصات زیر کدام است؟

(تعداد استوانه = ۱۰ ، تعداد شیار در استوانه = ۵ ، تعداد بلاک در شیار = ۴)

حل: ظرفیت دیسک برابر $10\times5\times4=200$ بلاک می باشد، بنابراین داریم :

$0\le RBA\le 199$

■

نحوه محاسبه شماره درایور

شماره درایور در حالتی که دو رسانه با ظرفیت های S_1 و S_2 داشته باشیم، از رابطه زیر محاسبه می شود:

$$DR\#\begin{cases}1 & 0\le RBA\le S_1-1\\ 2 & S_1\le RBA\le S_1+S_2-1\end{cases}$$

*با فرض وجود دو رسانه زیر، طیف مقادیر RBA در کل فضای ذخیره سازی چیست؟

نوع رسانه	تعداد استوانه	تعداد شیار در استوانه	تعداد بلاک در شیار
D_1	c_1	t_1	b_1
D_2	c_2	t_2	b_2

حل: از آنجا که ظرفیت D_1 برابر $c_1\times t_1\times b_1$ و ظرفیت D_2 برابر $c_2\times t_2\times b_2$ می باشد، محدوده RBA برابر است با : $0\le RBA\le(c_1\times t_1\times b_1+c_2\times b_2\times t_2-1)$ ■

* با فرض وجود سه رسانه زیر، طیف مقادیر RBA در کل فضای ذخیره سازی چیست؟

نوع رسانه	تعداد استوانه	تعداد شیار در استوانه	تعداد بلاک در شیار
D_1	1	4	7
D_2	2	5	8
D_3	3	6	9

حل: ظرفیت دیسک اول ۲۸ بلاک، دیسک دوم ۸۰ بلاک و دیسک سوم ۱۶۲ بلاک می باشد. بنابراین داریم:

$$0 \le RBA \le (28+80+162)-1 \quad \Rightarrow \quad 0 \le RBA \le 269 \blacksquare$$

بافر

بافر ناحیه ای واسط در عملیات ورودی و خروجی که برای ایجاد هماهنگی بین عملیات I/O و CPU بکار می‌رود. در بافر حداقل یک رکورد یا یک بلاک (در حالت فایل بلاک بندی شده) قرار داده می‌شود. در سیستم فایل، بافر معمولا از منطقه ای از حافظه اصلی به برنامه فایل پرداز تخصیص داده می شود، که به آن منطقه بافرها (Buffers Pool) می گویند.

- بافرینگ باعث افزایش کارایی سیستم در پردازش فایل است.
- بافرینگ در محیط چند برنامه‌ای، کارایی سیستم و برنامه را افزایش می دهد.

نحوه ایجاد بافرها

بافرها به سه روش ساخته می شوند:

۱- با ایجاد ناحیه ای از حافظه در برنامه و با اجرای یک ماکرو که محتوای بافر را با فایلهای تحت پردازش مرتبط می کند. (در این حالت برنامه ساز خود بافر را ایجاد می کند)

۲- با اجرای یک ماکرو، که از سیستم عامل درخواست ایجاد بافر می کند.

۳- خود سیستم عامل وقتی که فایل باز می شود، اقدام به ایجاد بافر می کند و پس از بسته شدن فایل، بافر را باز پس می گیرد.

انواع بافرینگ از نظر تعداد بافرها

از این لحاظ، انواع بافرینگ به سه دسته ساده، مضاعف و چندگانه تقسیم می شود.

بافرینگ ساده (Single)

از یک بافر استفاده می‌شود. هنگامی که بافر در حال پر شدن است، CPU در حالت عاطل (idle) قرار دارد.

بافرینگ مضاعف (Double)

در بافرینگ مضاعف از دو بافر استفاده می‌شود. در هنگامی که یک بلاک خوانده می‌شود و به بافر منتقل می‌شود، بافر پر دیگر پردازش می‌شود. بدیهی است زمانی را که CPU برای پردازش محتوای یک بافر مصرف می‌کند، باید از زمانی که پرازنده I/O و کنترل کننده دیسک برای انتقال بلاک به یک بافر لازم دارند، کمتر باشد. به عبارتی داشته باشیم : $C_B < b_{tt}$. اگر این شرط برقرار نباشد یعنی $C_B \geq b_{tt}$، بافرینگ مضاعف دیگر کارایی نخواهد داشت و نرخ انتقال واقعی کاهش می یابد.

شرط کارایی بافرینگ مضاعف را می توان به صورتهای زیر نیز نوشت:

1) $C_B < b_{tt}$ 2) $C_B \leq \frac{B+G}{t}$ 3) $C_R \leq \frac{R+W_R}{t}$

$b_{tt} = \frac{B}{t}$: زمان انتقال یک بلاک و t : نرخ انتقال رسانه

C_B : زمان پردازش محتوای بلاک و C_R : زمان پردازش محتوای رکورد

G : طول گپ و B : طول بلاک و R : طول رکورد

در حالت عدم وجود شرط کارایی یعنی $C_B > \frac{B+G}{t}$ ، سیستم ابتدا دو بلاک در دو بافر می خواند و پردازش می کند. وقتی که در همین دور دیسک، آغاز بلاک سوم به زیر نوک خواندن/نوشتن برسد، بافر اول هنوز اشغال است و لذا سیستم نمی تواند بلاک سوم را بخواند. بدین ترتیب برای خواندن دو بلاک بعدی، به اندازه یک دور دیسک باید انتظار کشید. (البته با فرض $C_B < r$)

* مطلوب است حداکثر زمان لازم برای پردازش یک بلاک در بافرینگ مضاف کارا برای رسانه ای با مشخصات داده شده؟

($B=180$ و $G=30$ بایت، نرخ انتقال =۳۰ بایت بر میلی ثانیه)

حل :

$$C_B \le \frac{B+G}{t} \Rightarrow C_B \le \frac{180+30}{30} \Rightarrow C_B \le 7ms$$

بنابراین C_B حداکثر می تواند ۷ میلی ثانیه باشد.

* برای خواندن ۴ بلاک از دیسک و پردازش آنها ، زمان اجرای عملیات فوق در صورت استفاده از بافرینگ مضاعف (یک بافر ورودی و بافر دیگر برای پردازش) چند میلی ثانیه است؟

(زمان خواندن=۸ میلی ثانیه)(زمان پردازش=۲ میلی ثانیه)(ظرفیت بافر= یک بلاک)

حل :

مراحل کار به صورت زیر است:

۱- خواندن B_1 به بافر اول

۲- خواندن B_2 به بافر دوم و پردازش همزمان B_1

۳- خواندن B_3 به بافر اول و پردازش همزمان B_2

۴- خواندن B_4 به بافر دوم و پردازش همزمان B_3

۵- پردازش B_4

بنابراین زمان اجرای عملیات برابر است با:

8+8+8+8+2=34 ms

در مراحل ۲و۳و۴، عملیات خواندن و پردازش به صورت همزمان انجام می گیرد و چون زمان خواندن برابر ۸ میلی ثانیه و زمان پردازش برابر ۲ میلی ثانیه است، در هر کدام از این مراحل بزرگترین زمان یعنی ۸ میلی ثانیه تاثیر گذار است.

بافرینگ چندگانه معمولا به صورت چرخشی پیاده سازی می شود، به همین علت به آن بافرینگ چرخشی (Circular) هم می گویند.

روشهای دسترسی برنامه به محتوای بافر

برنامه به دو روش انتقالی و مکان نمایی می تواند به محتوای بافر دستیابی داشته باشد.

اسلوب انتقالی (Move mode)

رکورد از بافر ورودی به ناحیه کاری برنامه یا از ناحیه کاری به بافر خروجی منتقل می شود. بلاک بندی و بلاک گشایی توسط سیستم انجام می شود. برنامه در این روش به بافر دسترسی ندارد و بافر خاص خود را دارد.

اسلوب مکان نمائی (Locate mode)

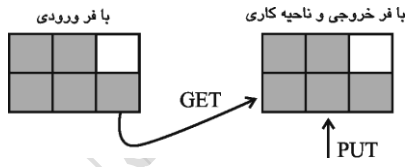
مکان بافر حاوی رکورد به برنامه فایل پرداز داده می‌شود، مثلا با گذاشتن آدرس آن در یک ثبات. در این روش کاربر از همان بافر به عنوان ناحیه کاری استفاده می کند و بلاک بندی و بلاک گشایی توسط خود برنامه انجام می شود. به این روش اسلوب تعویضی نیز می گویند.

استفاده از روشهای انتقالی و مکان نمائی در هر یک از دو عمل ورودی یا خروجی یا هر دو امکان پذیر است.

در هنگامی که در بافر ورودی و خروجی از اسلوب انتقالی استفاده می‌شود، ناحیه کاری در خارج بافرها قرار دارد.

انتقال رکوردها به ناحیه کاری کاربر در روش انتقالی زمانگیر است که در اسلوب مکان نمایی به ویژه در حالت استفاده از بافرینگ مضاعف این اتلاف زمانی وجود ندارد.

*شکل زیر معرف استفاده از اسلوب انتقالی در ورودی و اسلوب مکان نمایی در خروجی می باشد:



ارتباط عملیات خواندن/نوشتن با نحوه بافرینگ

با توجه به تعداد بافر و اینکه سیستم با چه اسلوبی عمل می کند، پنج حالت به شرح زیر ممکن است وجود داشته باشد:

بافر نداریم و فقط ناحیه کاری داریم. (روش مبنایی)	الف- نمی توان بلاک بندی انجام داد. ب- رکوردها به طور مجزا به ناحیه کاری کاربر در حافظه منتقل می

	شوند. ج- عمل خواندن، رکورد به رکورد انجام می شود.
یک بافر داریم و ناحیه کاری نداریم.	الف- می توان بلاک به بلاک خواند یا نوشت. ب- عملیات خواندن یا نوشتن پیشرس توسط کاربر، امکان پذیر نیست.
یک بافر داریم ویک ناحیه کاری داریم.	الف- می توان بلاک به بلاک خواند. ب- بلاک بعدی را می توان خواند یا نوشت (در اثناء پردازش آخرین رکورد بلاک)
دو بافر داریم و ناحیه کاری نداریم.	پردازش محتوای یک بافر در اثناء پر یا خالی شدن بافر دیگر انجام می گیرد.
دو بافر داریم و ناحیه کاری هم داریم.	کامل بودن همروندی بین پردازش رکوردها و عملیات ورودی/خروجی.

در حالتهایی که ناحیه کاری نداریم، از اسلوب مکان نمایی و در حالتهایی که ناحیه کاری داریم، از اسلوب انتقالی استفاده می شود.

حالات ممکن در عمل خواندن از نوار به روش انتقالی با یک بافر

۱) کار پردازنده ورودی/خروجی درست در همان زمانی که توسط پردازنده اصلی وارسی می شود، تمام شود.

۲) کار پردازنده ورودی/خروجی تمام شده باشد و در حال انتظار باشد.($C_B > b_{tt}$)

۳) کار پردازنده ورودی/خروجی تمام نشده باشد و پردازنده مرکزی در حال انتظار باشد. ($C_B < b_{tt}$)

انواع بافر از نظر محل ایجاد

بافر از نظر محل ایجاد به دو نوع سخت افزاری و نرم افزاری تقسیم می شود.

بافر سخت افزاری

بافر موجود در دستگاههایی مانند کارت خوان یا چاپگر می باشد که تعداد کمی کاراکتر در آن قرار می گیرد و بعد از پر شدن به آن کانال تخصیص داده می شود تا داده از محیط برون ماشینی به محیط درون ماشینی منتقل شود.

بافر نرم افزاری

ناحیه ای در حافظه اصلی یا حافظه پنهان ، که توسط سیستم عامل در اختیار برنامه های فایل پرداز گذاشته می‌شود.

تذکر: زمان پردازش انبوه فایل به عوامل " نوع بافرینگ ، نحوه دستیابی برنامه به رکوردها و زمان پردازش محتوای بلاک" بستگی دارد.

منتخبی از عناوین آموزشی منتشر شده بر روی فرادرس

برنامه‌نویسی	
عنوان آموزش	مدت زمان تقریبی
مبانی برنامه نویسی - کلیک کنید (+)	۳ ساعت
برنامه نویسی C - کلیک کنید (+)	۱۳ ساعت
آموزش برنامه نویسیC++	۲۰ ساعت
برنامه نویسی کاربردی سی شارپ - کلیک کنید (+)	۱۴ ساعت
آموزش جامع شی گرایی در سی شارپ - کلیک کنید (+)	۱۴ ساعت
برنامه نویسی جاوا - کلیک کنید (+)	۲۳ ساعت
آموزش برنامه نویسی PHP - کلیک کنید (+)	۲۸ ساعت
آموزش فریمورک PHP کدایگنایتر - (CodeIgniter) کلیک کنید (+)	۷ ساعت
آموزش اسکریپت برنامه نویسی jQuery کلیک کنید (+)	۷ ساعت
آموزش ویژوال بیسیک دات نت - (Visual Basic.NET) کلیک کنید (+)	۱۳ ساعت
آموزش تکمیلی ویژوال بیسیک دات نت - (Visual Basic.NET) کلیک کنید (+)	۱۶ ساعت
آموزش برنامه نویسی با روش سه لایه به زبان VB.Net - کلیک کنید (+)	۴ ساعت
برنامه نویسی اسمال بیسیک یا Small Basic - کلیک کنید (+)	۱۶ ساعت
آموزش ساخت بازی ساده در ویژوال بیسیک - کلیک کنید (+)	۲ ساعت
آموزش کاربردی SQL Server - کلیک کنید (+)	۱۱ ساعت
آموزش آشنایی با LINQ to SQL در C# - کلیک کنید (+)	۲ ساعت
آموزش برنامه نویسی با روش سه لایه به زبان سی شارپ - کلیک کنید (+)	۴ ساعت
آموزش برنامه نویسی تحت شبکه با سی شارپ در قالب پروژه - کلیک کنید (+)	۱ ساعت
آموزش Cryptography در دات نت - کلیک کنید (+)	۳ ساعت

برنامه‌نویسی (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش قفل نرم افزاری در سی شارپ از طریق رجیستری	۴ ساعت
آموزش ساخت اپلیکیشن کتاب و کار با داده‌ها در اندروید - کلیک کنید (+)	۱۳ ساعت
آموزش ارتباط با دیتابیس سمت سرور در اندروید - کلیک کنید (+)	۱۴ ساعت
آموزش ساخت روبات و کنترل آن با اندروید - کلیک کنید (+)	۱۶ ساعت
آموزش ساخت اپلیکیشن دیکشنری صوتی دو زبانه با قابلیت تشخیص صوت کاربر - کلیک کنید (+)	۷ ساعت
آموزش مدیریت بانک اطلاعاتی اوراکل - کلیک کنید (+)	۹ ساعت
آموزش مدیریت بانک اطلاعاتی اوراکل پیشرفته - کلیک کنید (+)	۷ ساعت
آموزش راه اندازی اوراکل ۱۲ cدر لینوکس	۱ ساعت
آموزش دیتاگارد در اوراکل - کلیک کنید (+)	۳ ساعت
برنامه نویسی متلب - کلیک کنید (+)	۹ ساعت
متلب برای علوم و مهندسی - کلیک کنید (+)	۱۴ ساعت
برنامه نویسی متلب پیشرفته - کلیک کنید (+)	۷ ساعت
طراحی رابط های گرافیکی (GUI) در متلب - کلیک کنید (+)	۸ ساعت
آموزش برنامه نویسی R و نرم‌افزار - RStudioکلیک کنید (+)	۷ ساعت
آموزش تکمیلی برنامه نویسی R و نرم‌افزار - RStudio کلیک کنید(+)	۵ ساعت
آموزش برنامه نویسی پایتون ۱ - کلیک کنید (+)	۲۰ ساعت
آموزش برنامه نویسی پایتون ۲ - کلیک کنید (+)	۵ ساعت
آموزش گرافیک کامپیوتری با - OpenGL کلیک کنید(+)	۱۶ ساعت

راه اندازی و مدیریت وبسایت‌ها و سرورها	
عنوان فرادرس	**مدت زمان تقریبی**
آموزش برنامه نویسی - PHP کلیک کنید (+)	۲۸ ساعت
آموزش فریمورک PHP کدایگنایتر - (CodeIgniter) کلیک کنید (+)	۷ ساعت
آموزش طراحی وب با - HTML کلیک کنید (+)	۳ ساعت
آموزش طراحی وب با - CSS کلیک کنید (+)	۵ ساعت
آموزش پروژه محور HTML و - CSS کلیک کنید - کلیک کنید (+)	۴ ساعت
آموزش جاوا اسکریپت - (JavaScript) کلیک کنید (+)	۹ ساعت
آموزش کار با - cPanel کلیک کنید (+)	۱ ساعت
آموزش مدیریت هاست با - DirectAdmin کلیک کنید - کلیک کنید (+)	۱ ساعت
راه اندازی سایت و کار با وردپرس - کلیک کنید (+)	۷ ساعت
راه اندازی فروشگاه دیجیتال با وردپرس و - Easy Digital Downloads کلیک کنید (+)	۱ ساعت
آموزش راه اندازی سایت شخصی با وردپرس - کلیک کنید (+)	۱ ساعت
آموزش ترجمه قالب وردپرس - کلیک کنید (+)	۲ ساعت
آموزش راه اندازی سایت خبری با وردپرس - کلیک کنید (+)	۲ ساعت

علوم کامپیوتر	
عنوان آموزش	مدت زمان تقریبی
ساختمان داده‌ها -کلیک کنید (+)	۱۰ ساعت
آموزش ساختمان داده‌ها (مرور – تست کنکور ارشد) - کلیک کنید (+)	۲۰ ساعت
آموزش نظریه زبان‌ها و ماشین‌ها -کلیک کنید (+)	۹ ساعت
آموزش نظریه زبان‌ها و ماشین (مرور – تست کنکور ارشد) - کلیک کنید (+)	۸ ساعت
آموزش سیستم های عامل - کلیک کنید (+)	۱۱ ساعت
آموزش سیستم عامل (مرور اجمالی و تست کنکور) - کلیک کنید (+)	۱۲ ساعت
آموزش پایگاه داده‌ها -کلیک کنید (+)	۸ ساعت
آموزش پایگاه داده‌ها (مرور – تست کنکور ارشد) - کلیک کنید (+)	۵ ساعت
آموزش طراحی و پیاده سازی زبان های برنامه سازی - کلیک کنید (+)	۱۰ ساعت
آموزش طراحی و پیاده سازی زبان های برنامه سازی (مرور – تست کنکور ارشد) - کلیک کنید (+)	۱۲ ساعت
آموزش روش های حل روابط بازگشتی - کلیک کنید (+)	۴ ساعت
آموزش روش تقسیم و حل در طراحی الگوریتم - کلیک کنید (+)	۲ ساعت
آموزش ذخیره و بازیابی اطلاعات - کلیک کنید (+)	۸ ساعت
آموزش ساختمان گسسته با رویکرد حل مساله - کلیک کنید (+)	۱۶ ساعت
آموزش جامع مدارهای منطقی - کلیک کنید (+)	۱۰ ساعت
آموزش معماری کامپیوتر با رویکرد حل مسأله - کلیک کنید (+)	۲۰ ساعت
آموزش ساختمان گسسته (مرور و حل تست های کنکور کارشناسی ارشد) - کلیک کنید (+)	۱۲ ساعت
آموزش طراحی الگوریتم - کلیک کنید (+)	۸ ساعت
آموزش شبکه های کامپیوتری ۱ - کلیک کنید (+)	۱۹ ساعت

علوم کامپیوتر (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش نظریه گراف و کاربردها - کلیک کنید (+)	۱۴ ساعت
آموزش نتورک پلاس - (Network+) کلیک کنید (+)	۱۰ ساعت
آموزش مدل سازی UML با نرم‌افزار - Rational Roseکلیک کنید (+)	۳ ساعت
آموزش پردازش ویدئو - کلیک کنید (+)	۳ ساعت
پردازش تصویر در متلب - کلیک کنید (+)	۱۶ ساعت
آموزش پردازش تصویر با - OpenCV کلیک کنید (+)	۱۰ ساعت

هوش مصنوعی	
عنوان آموزش	مدت زمان تقریبی
الگوریتم ژنتیک در متلب - کلیک کنید (+)	۱۴ ساعت
الگوریتم PSO در متلب - کلیک کنید (+)	۱۰ ساعت
الگوریتم ازدحام ذرات (PSO) گسسته باینری - کلیک کنید (+)	۲ ساعت
ترکیب الگوریتم ژنتیک و PSO در متلب - کلیک کنید (+)	۱ ساعت
حل مسأله فروشنده دوره گرد با استفاده از الگوریتم ژنتیک - کلیک کنید (+)	۲ ساعت
الگوریتم مورچگان در متلب - کلیک کنید (+)	۶ ساعت
الگوریتم رقابت استعماری در متلب - کلیک کنید (+)	۱۳ ساعت
طراحی سیستم های فازی عصبی یا ANFIS با استفاده از الگوریتم های فرا ابتکاری و تکاملی - کلیک کنید (+)	۲ ساعت
الگوریتم فرهنگی یا Cultural Algorithm در متلب - کلیک کنید (+)	۲ ساعت

هوش مصنوعی (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
شبیه سازی تبرید یا Simulated Annealing در متلب - کلیک کنید (+)	۴ ساعت
جستجوی ممنوع یا Tabu Search در متلب - کلیک کنید (+)	۲ ساعت
الگوریتم کرم شب تاب یا Firefly Algorithm در متلب - کلیک کنید (+)	۱ ساعت
بهینه سازی مبتنی بر جغرافیای زیستی یا BBO در متلب - کلیک کنید (+)	۲ ساعت
جستجوی هارمونی یا Harmony Search در متلب - کلیک کنید (+)	۲ ساعت
کلونی زنبور مصنوعی یا Artificial Bee Colony در متلب - کلیک کنید (+)	۳ ساعت
الگوریتم زنبورها یا Bees Algorithm در متلب - کلیک کنید (+)	۲ ساعت
الگوریتم تکامل تفاضلی - کلیک کنید (+)	۱ ساعت
الگوریتم بهینه سازی علف هرز مهاجم یا IWO در متلب - کلیک کنید (+)	۲ ساعت
الگوریتم بهینه سازی مبتنی بر و یادگیری یا - TLBO کلیک کنید (+)	۱ ساعت
الگوریتم بهینه سازی جهش قورباغه یا SFLA در متلب - کلیک کنید (+)	۴ ساعت
بهینه سازی چند هدفه در متلب - کلیک کنید (+)	۱۹ ساعت
بهینه سازی مقید در متلب - کلیک کنید (+)	۹ ساعت
شبکه های عصبی مصنوعی در متلب - کلیک کنید (+)	۲۸ ساعت
آموزش کاربردی شبکه های عصبی مصنوعی - کلیک کنید (+)	۹ ساعت
آموزش استفاده از شبکه عصبی مصنوعی با نروسولوشن - کلیک کنید (+)	۳ ساعت
شبکه عصبی GMDH در متلب - کلیک کنید (+)	۴ ساعت
شبکه های عصبی گازی به همراه پیاده سازی عملی در متلب - کلیک کنید (+)	۳ ساعت
طبقه بندی و بازشناسی الگو با شبکه های عصبی LVQ در متلب - کلیک کنید (+)	۳ ساعت
آموزش پیاده سازی الگوریتم های تکاملی و فراابتکاری در سی شارپ - کلیک کنید (+)	۸ ساعت

آمار و داده کاوی	
عنوان آموزش	مدت زمان تقریبی
گنجینه فرادرس های یادگیری ماشین و داده کاوی - کلیک کنید (+)	۸۸ ساعت
گنجینه فرادرس های محاسبات هوشمند - کلیک کنید (+)	۷۱ ساعت
آموزش یادگیری ماشین - کلیک کنید (+)	۲۴ ساعت
داده کاوی یا Data Mining در متلب - کلیک کنید (+)	۲۴ ساعت
آموزش داده کاوی در - RapidMiner کلیک کنید (+)	۲ ساعت
آموزش وب کاوی - کلیک کنید (+)	۱۷ ساعت
شبکه های عصبی مصنوعی در متلب - کلیک کنید (+)	۲۸ ساعت
آموزش کاربردی شبکه های عصبی مصنوعی - کلیک کنید (+)	۹ ساعت
شبکه عصبی GMDH در متلب - کلیک کنید (+)	۴ ساعت
شبکه های عصبی گازی به همراه پیاده سازی عملی در متلب - کلیک کنید (+)	۳ ساعت
طبقه بندی و بازشناسی الگو با شبکه های عصبی LVQ در متلب - کلیک کنید (+)	۳ ساعت
خوشه بندی با استفاده از الگوریتم های تکاملی و فراابتکاری - کلیک کنید (+)	۳ ساعت
تخمین خطای کلاسیفایر یا - Classifier کلیک کنید (+)	۲ ساعت
انتخاب ویژگی یا - Feature Selection کلیک کنید (+)	۲ ساعت
انتخاب ویژگی با استفاده از الگوریتم های فرا ابتکاری و تکاملی - کلیک کنید (+)	۴ ساعت
کاهش تعداد رنگ تصاویر با استفاده از روش های خوشه بندی هوشمند - کلیک کنید (+)	۱ ساعت
آموزش پردازش سیگنال های واقعی در متلب - کلیک کنید (+)	۴ ساعت
مبانی و کاربردهای راهبرد تلفیق داده یا - Data Fusion کلیک کنید (+)	۹ ساعت
آمار و احتمال مهندسی - کلیک کنید (+)	۱۳ ساعت

آزمون‌های فرض مربوط به میانگین جامعه نرمال در - SPSS کلیک کنید (+)	۳ ساعت
آمار و داده کاوی (ادامه از صفحه قبل)	
عنوان آموزش	**مدت زمان تقریبی**
آموزش محاسبات آماری در اکسل - کلیک کنید (+)	۲ ساعت
آموزش کنترل کیفیت آماری - کلیک کنید (+)	۵ ساعت
آموزش کنترل کیفیت آماری با - SPSS کلیک کنید (+)	۲ ساعت
آموزش مدل سازی معادلات ساختاری با - Amos کلیک کنید (+)	۷ ساعت
تجزیه و تحلیل اطلاعات با نرم‌افزار - SASکلیک کنید (+)	۴ ساعت

مهندسی برق	
عنوان آموزش	**مدت زمان تقریبی**
طراحی دیجیتال با استفاده از وریلوگ یا - Verilog کلیک کنید (+)	۷ ساعت
آموزش جامع مدارهای منطقی - کلیک کنید (+)	۱۰ ساعت
آموزش مروری طراحی و پیاده سازی مدارات منطقی - کلیک کنید (+)	۴ ساعت
آموزش میکروکنترلر AVR و نرم‌افزار - CodevisionAVRکلیک کنید (+)	۴ ساعت
آموزش تکمیلی میکروکنترلر AVR و نرم‌افزار - CodevisionAVRکلیک کنید (+)	۴ ساعت
آشنایی باPLC های ساخت شرکت های Omron و - Keyence کلیک کنید (+)	۶ ساعت
میکروکنترلر PIC با کامپایلر - CCS کلیک کنید (+)	۹ ساعت
آموزش تحلیل و طراحی مدارات الکترونیکی با - Proteus کلیک کنید (+)	۳ ساعت
آموزش شبیه سازی و تحلیل مدارهای الکتریکی و الکترونیکی با پی اسپایس (PSpice) -کلیک کنید (+)	۳ ساعت
آموزش مقدماتی - ADS کلیک کنید (+)	۳ ساعت
آموزش تکمیلی آنالیز مدار با نرم‌افزار - ADSکلیک کنید (+)	۲ ساعت

مهندسی برق (ادامه از صفحه قبل)	
عنوان آموزش	مدت زمان تقریبی
آموزش تحلیل ریاضی مدارات الکتریکی با - OrCAD کلیک کنید (+)	۲ ساعت
آموزش شبیه سازی مدارات الکترونیکی با - Orcad Capture کلیک کنید (+)	۲ ساعت
آموزش برنامه نویسی آردوینو (- (Arduinoکلیک کنید (+)	۸ ساعت
آموزش تکمیلی برنامه نویسی آردوینو - (Arduino) کلیک کنید (+)	۷ ساعت
آموزش طراحی برد مدار چاپی به کمک نرم‌افزار - Altium Designer کلیک کنید (+)	۷ ساعت
آموزش مبانی ربات های برنامه پذیر - کلیک کنید (+)	۵ ساعت
آموزش ساخت روبات و کنترل آن با اندروید - کلیک کنید (+)	۱۶ ساعت
آموزش مدارهای الکتریکی ۱ - کلیک کنید (+)	۹ ساعت
آموزش مدارهای الکتریکی ۲ - کلیک کنید (+)	۱۱ ساعت
آموزش سیستم های کنترل خطی - کلیک کنید (+)	۱۰ ساعت
آموزش مکاترونیک کاربردی ۱ - کلیک کنید (+)	۱۳ ساعت
آموزش کامسول (مباحث منتخب) - کلیک کنید (+)	۳ ساعت
آموزش سینماتیک مستقیم و معکوس ربات‌ها -کلیک کنید (+)	۳ ساعت
آموزش تجزیه و تحلیل سیگنال‌ها و سیستم ها - کلیک کنید (+)	۲۷ ساعت
آموزش متلب با نگرش تحلیل آماری، تحلیل سری های زمانی و داده های مکانی - کلیک کنید (+)	۸ ساعت
پردازش سیگنال های دیجیتال با استفاده از نرم‌افزار متلب - کلیک کنید (+)	۴ ساعت
شبیه سازی سیستم با سیمیولینک - کلیک کنید (+)	۴ ساعت
آموزش سیستم های قدرت در سیمولینک و متلب - کلیک کنید (+)	۱۱ ساعت
آنالیز پایداری و کنترل سیستم های قدرت با استفاده از جعبه ابزارهای نرم‌افزار متلب - کلیک کنید (+)	۲ ساعت
آشنایی با SimPowerSystems در شبیه سازی سیستم های قدرت - کلیک کنید (+)	۳ ساعت

مهندسی برق (ادامه از صفحه قبل)	
عنوان آموزش	**مدت زمان تقریبی**
شبیه سازی ماشین های الکتریکی در تولباکس های Simulink و SimPowerSystem در نرم‌افزار متلب - کلیک کنید (+)	۴ ساعت
آموزش الکترونیک قدرت – شبیه سازی در متلب و سیمولینک - کلیک کنید (+)	۸ ساعت
آموزش شبیه سازی عملکرد انواع ماشین های الکتریکی در سیمولینک متلب - کلیک کنید (+)	۱۰ ساعت
برنامه های پاسخگویی بار - کلیک کنید (+)	۴ ساعت
آموزش نرم‌افزار ETAP برای تحلیل سیستم های قدرت - کلیک کنید (+)	۲۱ ساعت
آموزش مقدماتی نرم‌افزار GAMSبرای حل مسائل بازار برق - کلیک کنید (+)	۵ ساعت
آموزش پخش بار اقتصادی (دیسپاچینگ اقتصادی) در GAMS - کلیک کنید (+)	۲ ساعت
کاربرد فازی در سیستم های قدرت - کلیک کنید (+)	۲ ساعت
آموزش نرم‌افزار HFSS - کلیک کنید (+)	۵ ساعت
طراحی آنتن مایکرواستریپ به کمک نرم‌افزار HFSS - کلیک کنید (+)	۱ ساعت
آموزش طراحی و شبیه سازی آنتن های SIW با HFSS - کلیک کنید (+)	۱ ساعت
آموزش بررسی کامل آنتن های مایکرواستریپ و طراحی آن توسط CST - کلیک کنید (+)	۳ ساعت
آموزش تجزیه سیگنال به مولفه های مود ذاتی یا Empirical Mode Decomposition -کلیک کنید (+)	۴۰ دقیقه
نمونه برداری و بازسازی اطلاعات در سیستم های کنترل دیجیتال - کلیک کنید (+)	۳ ساعت
بررسی پاسخ ورودی پله در شناسایی فرآیندهای صنعتی - کلیک کنید (+)	۱ ساعت
مدل سازی و شناسایی سیستم های دینامیکی با استفاده از مدل ARX و شبکه فازی عصبی - ANFIS کلیک کنید (+)	۱ ساعت
طراحی و تنظیم ضرایب کنترل کننده PID با منطق فازی - کلیک کنید (+)	۲ ساعت
آموزش کنترل سیستم چهار تانک - کلیک کنید (+)	۲ ساعت

فصل ۵

ظرفیت واقعی نوار- نرخ انتقال واقعی نوار

ظرفیت واقعی دیسک - نرخ انتقال واقعی دیسک

ظرفیت واقعی نوار

ظرفیت واقعی با داشتن ظرفیت اسمی نوار به صورت زیر بدست می‌آید :

$$S_E = \frac{B}{B+G}.S_N$$

(S_N : ظرفیت اسمی نوار که برابر $L \times D$ می باشد. که L طول نوار و D چگالی نوار می‌باشد.)

تذکر: می توان در محاسبه دقیقتر عمل کرد و به جای B از $B - W_B^{'}$ استفاده کرد. ($W_B^{'}$: میزان حافظه هرز به ازاء یک بلاک، غیر از گپ است.

*در یک نوار به طول 2400 فوت و چگالی1600 bpi ، ظرفیت اسمی و تعداد بلاکهای ذخیره شده در نوار کدام است؟(هر فوت برابر ۱۲ اینچ است)

(B=2000 byte , R=200 byte , IBG=0.6 inch)

حل:

$$S_N = L.D = 2400 \times 12 \times 1600 = 4608 \times 10^4 byte$$

و تعداد بلاکهای ذخیره شده در نوار برابر است با :

$$b = \left\lfloor \frac{L}{B+G} \right\rfloor = \left\lfloor \frac{2400 \times 12}{\frac{2000}{1600} + 0.6} \right\rfloor = 15567$$

تذکر: در رابطه بالا مقدار B را به واحد اینچ تبدیل کرده ایم تا صورت و مخرج کسر هم واحد باشد.

(برای تبدیل اندازه بلاک از بایت به اینچ کافی است به چگالی تقسیم شود.)

*در یک نوار مغناطیسی، ظرفیت واقعی ۰/۸ ظرفیت اسمی آن است. از طرفی طول هر بلاک داده ۱۲۰۰ بایت و چگالی ۳۰۰ بایت در اینچ فرض شده است. طول هر گپ در این نوار چند اینچ است؟

حل :

$$S_E = \frac{B}{B+G}.S_N$$

$$\frac{B}{B+G} = \frac{8}{10} \Rightarrow \frac{1200}{1200+G} = \frac{8}{10} \Rightarrow G = 300byte \Rightarrow G = \frac{300}{300} = 1inch$$

درصد استفاده واقعی از نوار برابر $\frac{B}{B+G} \times 100$ می باشد. (فضای مفید به فضای کل)

*نواری با مشخصات زیر مفروض است. درصد استفاده واقعی از نوار وقتی که فایلی با ۱۰۰۰۰ رکورد ۸۰ بایتی را در آن ذخیره کرده ایم، کدام است؟

$$B_f = 70, IBG = 0.5inch, D = 1600dpi$$

حل :

$$B = B_f \times R = 70 \times 80 = 5600 \text{ (BYTE)}$$

$$G = 0.5 \times 1600 = 800 \text{ (BYTE)}$$

درصد استفاده واقعی= $\frac{5600}{5600+800} \times 100 = \%87.5$

گپ عامل اصلی کاهنده ظرفیت واقعی نسبت به ظرفیت اسمی می‌باشند.

اندازه بلاک با توجه به نوع دستگاه ذخیره‌سازی، نوع سیستم عامل و امکانات درایور تعیین می‌شود. اگر طول بلاک خیلی کوچک باشد ، تعدادIBG ها زیاد شده و در نتیجه حافظه هرز روی نوار زیاد خواهد شد.

نوار را به دو روش بلاکی و جریانی می توان خواند. در روش بلاکی (Block mode) ، نوار بعد از خوانده شدن یک بلاک می ایستد و در روش جریانی (Stream mode)، نوار بعد از خوانده شدن N بلاک می ایستد.

محاسبه نرخ انتقال واقعی نوار

پارامترهای طول گپ(G)، طول بلاک(B)، نرخ انتقال اسمی(t) و زمان حرکت- توقف (τ)، در ارزیابی نرخ انتقال واقعی نوار دخالت دارند. نرخ انتقال واقعی نوار در حالت اسلوب بلاکی از رابطه زیر محاسبه می شود:

$$t' = \frac{B}{\frac{B}{t} + \frac{G}{t} + \frac{\tau}{1000}}$$

و در اسلوب جریانی، از رابطه محاسبه می شود:

$$t' = \frac{NB}{\frac{NB}{t} + \frac{NG}{t} + \frac{\tau}{1000}}$$

نرخ انتقال واقعی، یک تابع هموگرافیک از B است که به N و G هم بستگی دارد.

زمان خواندن یک بلاک برابر $\frac{B}{t}$ و زمان واقعا سپری شده برابر $\frac{B}{t} + \frac{G}{t} + \frac{\tau}{1000}$ می باشد.

*زمان واقعی خواندن هر بلاک در نواری با مشخصات زیر چند ثانیه است؟

$$(\ B = 250(byte), G = 60(byte), \tau = 25(ms), \frac{B}{t} = 8(s) \ )$$

حل:

$$\frac{B}{t} = 8 \Rightarrow t = \frac{B}{8} = \frac{250}{8} = 31.25 \ \ (\frac{byte}{s})$$

$$\frac{B}{t} + \frac{G}{t} + \frac{\tau}{1000} = 8 + \frac{60}{31.25} + \frac{25}{1000} = 9.945 \ \ s$$

ارزیابی دیسک

برای دیسک نیز دو پارامتر بررسی شده برای نوار یعنی ظرفیت واقعی و نرخ انتقال واقعی را بررسی می کنیم.

ظرفیت واقعی دیسک

نحوه فرمت بندی شیار در ارزیابی ظرفیت واقعی دیسک مؤثر است. دیدیم که هر بلاک با طول B، دارای یک بخش پیشوندی به طول C می باشد. اگر ظرفیت اسمی شیار را C_{NT} بنامیم، در صد استفاده واقعی از فضای شیار در یک دیسک برابر است با :

$$C_{ET} = \frac{B.T_f}{C_{NT}} \times 100$$

که فاکتور شیار بندی از رابطه $T_f = \left[\frac{C_{NT}}{C+B}\right]$ بدست می آید. البته اگر بخواهیم دقیقتر عمل کنیم، به جای B از $B - W_B^{'}$ استفاده می کنیم.

* می‌خواهیم فایلی با بلاکهای ۱۲۰۰ بایتی با فاکتور بلاک بندی ۱۲ را روی دیسکی با طول شیار ۱۹۰۰۰ بایت ذخیره کنیم. با فرض اینکه بخش کنترلی هر بلاک ۱۷۰ بایت باشد، درصد استفاده واقعی از فضای شیار کدام است؟

حل:

$$T_f = \left[\frac{C_{NT}}{C+B}\right] = \left\lfloor\frac{19000}{170+1200}\right\rfloor = 13$$

$$C_{ET} = \frac{B.T_f}{C_{NT}} \times 100 = \frac{1200 \times 13}{19000} \times 100 = \%82$$

میزان واقعی استفاده از شیار به نحوه بلاک بندی و فرمت کردن شیار بستگی دارد.

وقتی اندازه بلاک کوچک است، برای افزایش میزان استفاده واقعی از شیار باید B_f را زیاد کرد.

میزان استفاده واقعی از حافظه در دیسکهای سکتوربندی شده

میزان استفاده واقعی از حافظه در دیسکهای سکتوربندی شده برابر است با :

$$E = \frac{R \times B_f}{N \times L_s} \times 100$$

(N : تعداد سکتور مورد نیاز برای ذخیره یک بلاک ، Ls : طول سکتور)

تذکر: در صورتی که مقدار N داده نشده باشد، می توان از رابطه $N = \left\lceil \frac{B}{L_s} \right\rceil$ آن را محاسبه کرد.

* فایلی با طول رکوردهای ۱۶۰ بایت و طول سکتور ۲۵۶ بایت مفروض است. با فرض $B_f = 4$ ، میزان استفاده واقعی از دیسک چند درصد است؟

حل: شکل زیر نشان می دهد که برای ذخیره یک بلاک ۶۴۰ بایتی (شامل ۴ رکورد ۱۶۰ بایتی)، نیاز به ۳ سکتور ۲۵۶ بایتی می باشد:

رکورد ۱	رکورد ۲	رکورد ۳	رکورد ۴
سکتور ۱		سکتور ۲	سکتور ۳

بنابراین میزان استفاده واقعی از دیسک برابر است با :

$$E = \frac{R \times B_f}{N \times L_s} \times 100 = \frac{160 \times 4}{3 \times 256} \times 100 = \%83$$

تذکر: کاهش ۱۷٪ از میزان استفاده واقعی به علت خالی ماندن ۱۲۸ بایت در انتهای سکتور سوم است:

$(3 \times 256 - 640 = 128)$

انتخاب اندازه بلاک و فاکتور بلاک بندی و تکنیک بلاک بندی، در میزان حافظه هرز تاثیر می گذارد.

در دیسک با سکتوربندی سخت افزاری، باید تعداد درستی از رکوردها در تعداد درستی از سکتورها قرار گیرد تا حافظه هرز انتهای شیار زیاد نشود.

نرخ انتقال واقعی دیسک

نرخ انتقال واقعی دیسک را در دو حالت دسترسی مستقیم و دسترسی ترتیبی بررسی می کنیم.

الف - در حالت دستیابی مستقیم به بلاک

$$t_{(D)} = \frac{B}{s + r + b_{tt}}$$

($s + r + b_{tt}$:زمان خواندن مستقیم یک بلاک)

البته برای ارزیابی دقیق تر نرخ انتقال واقعی می توان به جای B از $B - W_B$ استفاده کرد.

ب- در حالت دستیابی ترتیبی به بلاکها در پردازش انبوه

در دستیابی ترتیبی با شروع از نقطه ای از فایل (مثلا BOF)، بلاکها به ترتیبی که ذخیره شده انـد، خوانـده می شوند. در پردازش انبوه (پردازش تعدادی بلاک)، نوع بافرینگ و زمان پردازش بلاک در نرخ انتقال تاثیر گذارند و چهار حالت زیر را ممکن می سازند:

۱- بافرینگ ساده و مرتب خوانی

سیستم یک بلاک را در بافر می خواند و با شروع پردازش محتوای بافر، آغـاز بـلاک بعـدی در اثـر دوران دیسک از زیر نوک خواندن/نوشتن رد می شود و برای خوانـدن آن، پردازنـده I/O بایـد یـک دور دیسـک منتظر بماند. $t' = \frac{B}{2r + b_{tt}}$

۲- بافرینگ ساده و درهم خوانی

در این حالت، محدودیت خواندن بلاکها به ترتیب نشست آنها روی شیار را در نظر نمی گیریم. مثلا مـی خواهیم میانگین مقادیر یک فیلد از تعدادی رکورد را بدست آوریم. اگر $C_B \le b_{tt}$ باشد، بلاکهای شـیار در دو دور خوانده می شوند و داریم: $t' = \frac{B.T_f}{4r}$

۳- بافرینگ مضاعف و شرط کارایی

در این حالت، تمام بلاکهای شیار در یک دور دیسک خوانده می شوند و داریم:

$$t' = \frac{B.T_f}{2r}$$

۴- بافرینگ مضاعف و عدم شرط کارایی

در این حالت، سیستم ابتدا دو بلاک را در دو بافر خوانده و پردازش می کند. وقتی که آغاز بلاک سوم به زیر نوک برسد، بافر اول هنوز مشغول است و به یک دور دیسک انتظار نیاز است تا دو بلاک بعدی خوانده شوند و داریم:

$$t' = \frac{2B}{2r + 2b_{tt}} = \frac{B}{r + b_{tt}}$$

نرخ انتقال واقعی در حالت بافرینگ ساده با نماد $t'_{(1)}$ و در حالت بافرینگ مضاعف با نماد $t'_{(2)}$ نمایش داده می شود.

* نرخ انتقال واقعی دیسکی با مشخصات زیر، چند بایت بر ثانیه است؟

(زمان خواندن مستقیم یک بلاک=۲ ثانیه) (طول بلاک=۶۵۰ بایت) (IBG=۵۰ بایت)

$$t_{(D)} = \frac{B - W_B}{s + r + b_{tt}} = \frac{650 - 50}{2} = 300 \ (\frac{byte}{s})$$

زمان خواندن ترتیبی b بلاک برابر $s + r + b \times ebt$ و زمان خواندن تصادفی b بلاک برابر $b(s + r + btt)$ می باشد. $(ebt = \frac{B}{t'}, btt = \frac{B}{t})$

محاسبه زمان کل پردازش فایل

برای محاسبه زمان کل پردازش فایل، دو حالت پردازش رکوردی و بلاکی به طور تصادفی را بررسی می کنیم. در پردازش رکوردی، هر بار که یک بلاک خوانده می‌شود، فقط یک رکورد آن بلاک پردازش می‌شود در حالی که در پردازش بلاکی، هر بار که یک بلاک خوانده می‌شود، همه رکوردهای آن پردازش می‌شود.

زمان پردازش کل فایل در حالت پردازش رکوردی برابر است با:

$$T_{pfile(R)} = n(s + r + b_{tt}) + n.c_R$$

و زمان پردازش کل فایل در حالت پردازش بلاکی برابر است با:

$$T_{pfile(B)} = b(s + r + b_{tt}) + b.C_B$$

(c_R:زمان پردازش یک رکورد، c_B: زمان پردازش یک بلاک)

(n : تعداد رکوردهای فایل، b : تعداد بلاکهای فایل)

*در یک فایل با 18 بلاک، زمان خواندن کل فایل به صورت تصادفی و به فرم بلاکی برابر 25 میلی ثانیه است. اگر زمان پردازش هر بلاک 3.5 میلی ثانیه باشد، زمان پردازش کل فایل به صورت بلاکی چند میلی ثانیه خواهد بود؟

حل:

$$T_{pfile}(B) = b.(s + r + b_{tt}) + b.C_B = 25 + 18 \times 3.5 = 88$$

■

حداکثر نرخ انتقال

عواملی مانند زمان درنگ دورانی، زمان پردازش بلاک و حالات بافرینگ، تاثیر زیادی در نرخ انتقال واقعی دارد. بهترین حالت این است که سیستم بتواند در یک دور دیسک، تمام بلاکهای شیار را بخواند. بنابراین مفهوم حداکثر نرخ انتقال مطرح می شود که از رابطه زیر محاسبه می شود:

$$Max(t') = \frac{Tracksize}{2r}$$

* در دیسکی با اندازه شیار 19254 بایت و زمان چرخش یک دور کامل 16.7 میلی ثانیه، مقدار حداکثر نرخ انتقال چند مگا بیت بر ثانیه می باشد؟

حل:

$$Max(t') = \frac{19254(byte)}{16.7ms} = \frac{19254(byte)}{0.0167(s)} = 1.1 \ (\frac{Mbyte}{s})$$

ارزیابی دقیقتر زمان درنگ دوران

می دانیم که مقدار متوسط زمان درنگ دورانی برابر است با نصف زمان یک دور دیسک:

$$r = \frac{1}{2} \times \frac{60 \times 1000}{rpm}$$

ولی از آنجا که در ابتدای هر شیار، فیلدی حاوی اطلاعات کنترلی شیار و شناسایی آغاز آن وجود دارد و در هنگام واکشی یک رکورد (که بلاک حاوی آن رکورد نیز خوانده می شود)، سیستم باید شناسه آغاز شیار (Track Identifier) مورد نظر را شناسایی کند و پس از رسیدن نوک خواندن/نوشتن به آغاز شیار در زمان r و رد شدن آن از زیر نوک، به آغاز بلاک مورد نظر برسد.

بنابراین داریم:

$$r' = r + \frac{1}{2}.\frac{T_f - 1}{T_f}.2r = r(2 - \frac{1}{T_f})$$

($\frac{1}{2}.\frac{T_f - 1}{T_f}$: متوسط فاصله بین آغاز شیار تا آغاز بلاک مورد نظر)

اگر در هر شیار فقط یک بلاک داشته باشیم ($T_f = 1$)، با رسیدن به ابتدای شیار، به بلاک مورد نظر رسیده ایم و تاخیر برابر با r است.

تکنیکهای بهبود کارایی سیستم فایل

بعضی مواقع نیاز است تا فایل به طور نوبتی و به صورت پی در پی خوانده شود. داشتن بافرینگ کارا و امکانات تخصیص بافر، نقش مهمی در بهبود کارایی پردازش تمام فایل ایفا می کند. همچنین تکنیکهای کاهش زمان درنگ دورانی و زمان استوانه جویی موجب تسریع عملیات روی فایل می شوند.

تکنیکهای کاهش زمان درنگ دوران

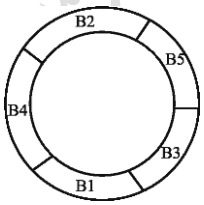
۱- تداخل بلاکها (Interleaving)

۲- تغییرمکان نقطه آغاز شیارها (track staggering)

۳- پراکنده خوانی

تداخل بلاکها (interleaving)

در تکنیک تداخل بلاکها (درهم چینی بلاکها)، بلاکها به صورت n در میان روی شیار چیده می‌شوند. در شکل زیر پنج بلاک به صورت یک در میان روی شیار چیده شده اند:



کاربرد تکنیک تداخل بلاکها :

۱- وقتی که نتوان دو بافر به فایل تخصیص داد و یک بافر داشته باشیم.

۲- وقتی که $c_B > b_{tt}$ باشد و بلاکها را بخواهیم مرتب بخوانیم.

در این تکنیک خواندن n بلاک از یک شیار به $\frac{n}{2}$ دور دیسک نیاز دارد که این کاهش در نرخ انتقال واقعی تأثیر می گذارد و زمان لازم برای انتقال کل فایل نسبت به حالتی که بلاکها به ترتیب شماره سریال روی شیار چیده شده است، به نصف کاهش می‌یابد.

مقدار n+1 را ضریب تداخل می گویند که با i_f نمایش داده می شود. وقتی می گوئیم ضریب تداخل ۲ است ، یعنی بلاکها یک در میان چیده شده اند.

نرخ انتقال انبوه در تکنیک interleaving برابر $t'_{interleaving} = \frac{t'}{i_f}$ می باشد.

تغییرمکان نقطه آغاز شیارها (Track Staggering)

در حالت معمولی مکان هندسی نقطه آغاز شیارها، یک شعاع صفحه است. اما در این روش، نقطه آغاز هر شیار نسبت به شیار قبلی زاویه θ می سازد. این تکنیک سخت افزاری است و وقتی کارایی دارد که زمان رفتن به استوانه همجوار بعدی خیلی کمتر از r باشد $(S_1 << r)$.

$$\theta = \frac{360}{60 \times 1000}.\text{rpm}.\text{Max}(S_1)$$

*با فرض اینکه سرعت چرخش دیسکی ۳۶۰۰ دور در دقیقه و حداکثر زمان لازم برای رفتن به استوانه همجوار بعدی ۲ میلی ثانیه باشد، مقدار θ در تکنیک track staggering چند درجه باید باشد، تا زمان انتظار دوران کاهش یابد؟

$$\theta = \frac{360}{60 \times 1000} \times \text{rpm} \times \text{Max}(S_1)$$
$$= \frac{6}{1000} \times 3600 \times 2 = 43.2$$

■

پراکنده خوانی

در مواقعی که پردازش می‌تواند فاقد نظم باشد، مانند خواندن کل فایل بطور پی در پی (Sequential) برای محاسبه حاصل جمع مقادیر یکی از صفت خاصه همه رکوردها، می‌توان بلاکها را پراکنده خواند. البته باید به تعداد کافی بافر موجود باشد. دراین حالت r برای رسیدن به ابتدای یک بلاک نصف زمان انتقال بلاک است:

$$r = \frac{1}{2} \times \left(\frac{B+G}{t} \right)$$

تکنیکهای کاهش زمان استوانه جویی

۱- استفاده از دیسکهای با بازوی ثابت

۲- توزیع فایل روی چند دیسک

۳- اعمال ملاحظاتی خاص در جایدهی رکوردها

۴- استفاده از الگوریتمهای کنترل حرکت بازو

استفاده از دیسکهای با بازوی ثابت

در این دیسکها، به ازاء هر شیار از رویه ، یک نوک خواندن/نوشتن به بازو متصل است و بازو حرکتی ندارد و بدین ترتیب زمان S صفر است .این تکنیک سخت افزاری است و هزینه آن بالا است.

توزیع فایل روی چند دیسک

در این تکنیک نرم افزاری، فایل بر روی استوانه های هم شماره از چند دیسک pack قرار داده می‌شود. بنابراین چون نوک R/W حرکت نمی‌کند، زمان استوانه جویی صفر است.

تکنیک RAID

در این تکنیک، به جای استفاده از یک واحد دیسک با ظرفیت بالا، از چند واحد دیسک کوچکتر به صورت یک آرایه استفاده می‌شود. این مجموعه دیسکها از نظر سیستم عامل به صورت یک واحد دیسک منطقی دیده می‌شود. در این تکنیک، داده‌ها روی دیسکهای مختلف توزیع می‌شوند و داده‌ها با افزونگی ذخیره می‌شوند تا در صورت خرابی، ترمیم (Recovery) داده‌ها، به راحتی انجام شود. با استفاده از این تکنیک، امکان همزمانی دستیابی به داده ها میسر می شود و کارایی سیستم فایل در عملیات I/O افزایش می یابد.

تکنیک RAID در چند سطح پیاده سازی می شود از جمله :RAID-0 تا RAID-7.

اعمال ملاحظاتی خاص در جایدهی رکوردها

قرار دادن رکوردها با تعداد دسترسی زیاد در استوانه های میانی دیسک و رکوردها با تعداد دسترسی کم در استوانه های بیرونی تر، موجب کاهش متوسط زمان استوانه جویی می شود.

استفاده از الگوریتمهای کنترل حرکت بازو

الگوریتم های کنترل حرکت بازو عبارتند از :

۱- **FCFS : (First Come First Serviced)**

در خواستها به ترتیب ورود، اجرا می‌شوند.

۲- **SSTF : (Shortest Seek Time First)**

بازو به سمت رکوردی حرکت می‌کند که به کمترین زمان برای حرکت بازو نیاز دارد.

۳- **SCAN**

بازوی دیسک مرتباً رویه را مرور (SCAN) کرده و در مسیر به درخواستها پاسخ می دهد. در ابتدا بازو به جهتی حرکت می‌کند که کوتاهترین زمان استوانه جویی را برای دستیابی نیاز دارد. اگر در جهت انتخاب شده به همه درخواستها پاسخ داده شد، جهت حرکت عوض می‌شود.

*صف درخواستهای سیلندر به صورت ۱۴، ۲۰، ۹، ۵، ۱۲ می‌باشد و هد بر روی سیلندر ۱۰ قرار دارد. در صورت استفاده از هر یک از الگوریتم ها ی کنترل حرکت بازو، ترتیب حرکت هد را مشخص کنید؟

حل:

FCFS : 10 , 12 , 5 , 9 , 20 , 14

SSTF : 10 , 9 , 12 , 14 , 20 , 5

SCAN : 10 , 9 , 5 , 12 , 14 , 20

■

*در یک دیسک سخت، نوک I/O HEAD روی سیلندر ۲۰ قرار دارد. اگر تقاضا برای خواندن سیلندرهای بـه ترتیب ۱۰، ۲۲، ۲۰، ۲، ۴۰، ۶ و ۳۸ به Driver آن وارد شود و چنانچه حرکت هد I/O بین دو سیلندر مجاور ۶ میلی ثانیه طول بکشد، در صورت استفاده از الگوریتم SSTF برای خواندن سیلندرهای، کل seek time مورد نیاز چقدر خواهد بود؟

حل : در SSTF همواره به سمت نزدیکترین سیلندر حرکت می شود:

$$20 \rightarrow 22 \rightarrow 10 \rightarrow 6 \rightarrow 2 \rightarrow 38 \rightarrow 40$$

مجموعه فاصله‌های برابر است با:

$$2 + 12 + 4 + 4 + 36 + 2 = 60$$

و چون هر حرکت ۶ میلی ثانیه طول می‌کشد، پس در کل 6×60 میلی ثانیه طول خواهد کشید.

■

فصل ۶

ساختار پایل

در هر سیستم فایل دو هدف سرعت عملیاتی و صرفه جویی در حافظه مورد نظر است که برای رسیدن به آنها باید جنبه های زیر را در طراحی سیستمهای ذخیره و بازیابی در نظر گرفت :

۱- حداقل بودن افزونگی

۲- دستیابی سریع

۳- سهولت در عملیات بهنگام سازی

۴- سهولت نگاهداری سیستم

۵- قابلیت اطمینان بالا

رسیدن به این اهداف در یک سیستم فایل به ساختاری که آن سیستم ایجاد می‌کند بستگی دارد. بنابراین شناخت ساختارهای فایل اهمیت خاص خود را دارد.

انواع ساختار های فایل عبارتند از :

۱- پایل (بی نظم)

۲- ترتیبی

۳- ترتیبی شاخص دار

۴- چند شاخصی

۵- مستقیم

۶- چند حلقه ای

۷- درختی

برای ارزیابی ساختارهای مختلف فایل، ضوابط زیر بررسی می‌شوند:

۱- اندازه رکورد (R)

۲- زمان واکشی یک رکورد از فایل (T_F)

۳- زمان بازیابی رکورد بعدی (T_N)

۴- زمان بهنگام سازی از طریق درج یک رکورد (T_I)

۵- زمان بهنگام سازی از طریق ایجاد تغییر در یک رکورد (T_U)

۶- زمان خواندن تمام فایل (T_X)

۷- زمان سازماندهی مجدد فایل (T_Y)

عملیات تغییر دهنده محیط فیزیکی عبارتند از : درج، بهنگام سازی، حذف و سازماندهی مجدد.

عمل حذف حالت خاصی از بهنگام سازی است.

انجام عملیات ششگانه (واکشی، بازیابی رکورد بعدی، درج رکورد جدید، بهنگام سازی، خواندن تمام فایل و سازماندهی مجدد)، به سه عمل مکان یابی، خواندن فیزیکی و نوشتن فیزیکی در محیط فیزیکی منجر می‌شود.

انواع فایل

۱- فایل متراکم : فایلی که تمام مقادیر همه صفات خاصه تمام رکوردهایش مشخص باشند.

۲- فایل غیر متراکم: فایلی که برخی از مقادیر بعضی از صفات خاصه در برخی از رکوردها موجود نباشد.

هنگامی فایل غیرمتراکم می‌شود که رکوردها طول ثابت و قالب ثابت مکانی داشته باشند و در نتیجه حافظه هرز ایجاد می‌شود.

وقتی فایلی را با رکوردهایی با قالب غیر ثابت مکان طراحی می کنیم، حالت غیر متراکم پدید نمی آید.

در فایل غیرمتراکم، یک یا بیش از یک فقره اطلاع در مورد بعضی نمونه های یک موجودیت وجود ندارد، یعنی اطلاع نهست (Missing information) داریم.

فایل با افزونگی

فایلی که مقادیر بعضی از صفات خاصه اش بیش از یکبار در محیط فیزیکی ذخیره سازی ذخیره شده باشند را فایل با افزونگی (Redundancy) می گویند.

انواع افزونگی

افزونگی بر دو نوع است :

۱- افزونگی تکنیکی (Technical Redundancy)

تکرار بعضی(یا تمام) از مقادیر یک (یا چند) صفت خاصه در محیط ذخیره سازی جهت ایجاد یک شیوه دستیابی کاراتر برای فایل را افزونگی تکنیکی می گویند. مثلا وقتی که روی صفت خاصه ای از یک فایل، شاخص ایجاد می‌کنیم، مقادیر آن صفت در فایل شاخص تکرار خواهند شد.

۲- افزونگی طبیعی (Natural Redundancy)

در افزونگی طبیعی ، یک مقدار مشخص از صفت خاصه در تعدادی از نمونه رکوردها وجود دارد. مثلا در فایل ثبت نام دانشجویان وجود نام یک درس در رکورد تمامی دانشجویان که آن درس را گرفته اند ، یک نوع افزونگی طبیعی است.

افزونگی طبیعی را باید کاهش داد. مثلاً از ساختار کارائی مانند چندحلقه ای استفاده کرد یا از تکنیکهای فشرده سازی استفاده کرد.

تکنیک فشرده سازی ماتریس بیتی

این تکنیک هنگامی کاربرد دارد که فقره اطلاع تکرار شونده (صفت چند مقداری) داشته باشیم و همچنین مقادیر صفت خاصه از مجموعه ای محدود انتخاب شده باشند. در این روش، برای ذخیره سازی تمام این

صفات به n بیت حافظه نیاز است. که n تعداد عناصر مجموعه‌ای است که مقادیر صفت خاصه مـورد نظـر از آن گرفته شده است.در شرایطی که هم طول رکوردها متغیر و هم افزونگی طبیعی تشدید می‌شود، تکنیک ماتریس بیتی یکی از روشهای کاهش این افزونگی است.

*ذخیره فایل اطلاعات دانشجو- درس شامل ۵ دانشجو و ۱۳ درس:

الف- بدون استفاده از ماتریس بیتی

شماره دانشجو	شماره درس
54381	177 , 179 , 183 , 184
54407	177 , 178 , 181 , 183
54408	176 , 177 , 184
54503	178 , 181
54504	178 , 183

ب- با استفاده از ماتریس بیتی

	176	177	178	179	180	181	182	183	184
54381	0	1	0	1	0	0	0	1	1
54407	0	1	1	0	0	1	0	1	0
54408	1	1	0	0	0	0	0	0	1
54503	0	0	1	0	0	1	0	0	0
54504	0	0	1	0	0	0	0	1	0

شرح اصول عملیات ششگانه

۱- واکشی رکورد دلخواه

۲- بازیابی رکورد بعدی

۳- بهنگام سازی از طریق درج

۴- بهنگام سازی از طریق تغییر محتوای رکورد

۵- خواندن تمام فایل

۶- سازماندهی مجدد

واکشی رکورد دلخواه

واکشی یک رکورد دلخواه یک عمل محتوایی است. یعنی مقدار یکی از صفات خاصه رکورد به عنوان آرگومان جستجو داده می‌شود. لازمه این عمل، جستجو کردن در فایل، دستیابی به بلاک حاوی رکورد مورد نظر و خواندن آن است. برای این دسترسی از روش ترتیبی یا مستقیم می‌توان استفاده کرد. در روش دسترسی ترتیبی، تعدادی بلاک به طور پی درپی خوانده می‌شود تا به بلاک مورد نظر برسد و در روش مستقیم، آدرس بلاک بدست آمده و بلاک مستقیماً خوانده می‌شود.

زمان خواندن یک بلاک با روش دسترسی ترتیبی برابر $\frac{B}{t'}$ و با روش دسترسی مستقیم برابر $s+r+b_{tt}$ می باشد.

روشهای تنظیم درخواست واکشی

۱- ساده (Single request)

درخواست واکشی رکورد با شماره دانشجویی ۱۵۰ .

۲- طیفی(Range request)

درخواست واکشی مشخصات دانشجویان از شماره دانشجویی ۱۰۰ تا ۲۰۰. در این درخواست نشانوند جستجو هر صفتی می تواند باشد و لزوما کلید نیست.

۳- محاسباتی (Functional request)

مثل در خواست بازیابی معدل، وقتی که خود معدل در فایل ذخیره نشده است و برای محاسبه آن از اطلاعات ذخیره شده در فایل استفاده می کنیم.

۴- بولی (Boolean request)

درخواستی که پاسخ به آن توسط عملگرهای AND,OR,XOR به دست می آید.

۵- مرکب (Composite request)

حالت خاصی از درخواست بولی که در آن مقدار چند صفت خاصه داده می شود.

بازیابی رکورد بعدی

رکورد بعدی منطقی، رکوردی است که بر اساس یک نظم خاص مورد نظر پردازشگر فایل، بعد از رکورد فعلی باید بازیابی شود. مشخص است که کاربر برای بازیابی رکورد بعدی، نشانوند جستجو را نمی دهد. موقعیت رکورد بعدی نسبت به رکورد فعلی به یکی از سه صورت زیر است:

۱- همجوار فیزیکی باشند.

۲- از رکورد فعلی به بعدی اشاره گر وجود داشته باشد.

۳- هیچ ارتباطی بین آنها نباشد.(بازیابی رکورد بعدی ممکن نیست)

بهنگام سازی از طریق درج

درج رکورد، صرفاً یک عمل نوشتن نمی‌باشد و به عملیاتی نیاز دارد که حجم آن به ساختار فایل بستگی دارد. این عملیات عبارتند از :

۱- یافتن و خواندن بلاکی که رکورد باید در آن درج شود.

۲- جا دادن رکورد در بلاک

۳- بازنویسی بلاک

۴– عملیات پس از درج در بعضی ساختارها (مانند تنظیم اشاره گر ها)

بهنگام سازی از طریق تغییر محتوای رکورد

این عمل یعنی تغییر مقدار یک یا بیش از یک صفت خاصه در یک رکورد مشخص ، کـه دارای اصـول زیـر است:

۱- واکشی رکورد بهنگام در آمدنی
۲- ایجاد نسخه جدید در بافر
۳- بازنویسی نسخه جدید در جای قبلی در بهنگام سازی درجا
۴- تنظیم ارتباط ساختاری بین رکورد با رکوردهای دیگر فایل

انواع بهنگام سازی

۱- درجا (inplace)

رکورد بهنگام درآمده، در محل قبلی اش نوشته می‌شود. لازم به ذکر است کـه همـواره بهنگـام سـازی درجا ممکن نمی‌باشد ، چون ممکن است طول رکورد بعد از بهنگام سازی تغییر کند.

۲- برون از جا (outplace)

رکورد بهنگام درآمده در جایی دیگر نوشته می‌شود و نسخه قدیم با نشانگر « حـذف شـده » در محـل قبلی درج می‌شود.

اگر فایل دارای افزونگی باشد، بهنگام سازی باید منتشر شـونده (propagating update) باشـد، چـون در غیر اینصورت پدیده ناسازگاری (inconsistent) بوجود می‌آید.

خواندن تمام فایل

در صورت درخواست کاربر، سازماندهی مجدد، کپی گرفتن و یا در ایجاد یک استراژی دسـتیابی بـه فایـل، باید فایل را خواند. نحوه خواندن به صورت پی در پی (sequential) و یا سریال (serial) می باشد. در حالت پی درپی، بلاکها به ترتیب از ابتدا تا انتهای فایل پشت سرهم خوانده می‌شوند و در حالت سریال، بر اساس نظم صعودی یکی از صفات خاصه (معمولاً کلید)، عمل خواندن انجام می‌شود.

اگر دسترسی به رکورد بعدی ممکن نباشد، نمی‌توان فایل را سریال خواند.

اگر رکوردها در محیط فیزیکی به طور سریال ذخیره شده باشند، آنگاه نتیجه خواندن پی در پی و سریال یکی خواهد بود.

سازماندهی مجدد

بعد از مدتی که از لود اولیه فایل می گذرد، به علت انجام عملیات ذخیره‌سازی، در فایل تغییراتی ایجاد می‌شود که باعث کاهش کارایی اولیه آن می‌شود. دلایل سازماندهی مجدد عبارتند از :

۱- احیاء نظم ساختاری آغازین

۲- خارج کردن حافظه‌های هرز

۳- اصلاح استراتژی دستیابی

همه دلایل سازماندهی مجدد، لزوماً در همه ساختار ها مطرح نمی‌باشند.

برای سازماندهی مجدد ، عملیات زیر انجام می گیرد :

۱- خواندن تمام فایل

۲- بلاک بندی مجدد رکوردها ضمن خارج کردن رکوردهای حذف شدنی

۳- بازسازی ساختار مربوط به استراتژی دستیابی (در صورت وجود)

می‌توان بلاک را در همان دور جاری دیسک بازنویسی کرد. در این حالت زمان بازنویسی برابر است با : $T_{RW} = 2r$ و اگر عملیات در بافر به موقع انجام نگیرد سیستم یک دور را از دست می دهد و در این حالت $T_{RW} = 4r$ خواهد شد.

فایل با ساختار پایل (بی نظم)

رکوردهای این فایل براساس مقادیر هیچ صفتی مرتب نمی‌باشند و برای ایجاد فایل، رکوردها بخش بندی نمی‌شوند و در این ساختار استراتژی دستیابی وجود ندارد. قالب رکوردها غیرثابت مکان و طول متغیر دارند و مکان فیلدهای صفات در رکوردهای مختلف، متفاوت است. در هر رکورد اسم و مقدار صفت خاصه برای همه صفات خاصه باید ذخیره شود که باعث افزونگی خواهد شد. همچنین برای اطلاع نهست فیلدی وجود ندارد و باعث صرفه جویی در حافظه می‌شود.

*رکورد یک دانشجو در فایل دانشجویان می‌تواند به صورت زیر باشد:

$$ID = 123, Name = Ali, age = 23$$

ID اسم صفت خاصه و 123 مقدار صفت خاصه می باشد.

موارد استفاده ساختار پایل

۱- در محیطهایی که داده‌ها نظم پذیر نمی‌باشند و پیش پردازشی روی داده ها انجام نشده است و اساسا فایل برای بایگانی ایجاد شود.

۲- در محیطهایی که امنیت داده‌ها مورد نظر باشد.(بی نظمی امنیت را بالا می برد)

۳- مبنایی برای درک و طراحی ساختارهای بهتر.

ارزیابی کارایی ساختار پایل

متوسط اندازه رکورد

متوسط اندازه رکورد در ساختار پایل برابر است با :

$$R = a'(A+V+2)$$

a` : متوسط تعداد صفات خاصه یک رکورد

A : متوسط حافظه لازم برای ذخیره اسم صفت خاصه

V : متوسط حافظه لازم برای ذخیره مقدار صفت خاصه

تذکر: در محاسبه R ، یک بایت برای علامت انتساب (=) و یک بایت برای علامت جداساز (کاما) در نظر گرفته می‌شود.

عوامل دخیل در ارزیابی اندازه رکورد عبارتند از:

بخش داده ای رکورد، بخش غیر داده ای رکورد، W_R با در نظر داشتن همه هرزها، متراکم یا غیر متراکم بودن فایل، پدیده افزونگی و استفاده یا عدم استفاده از تکنیک های فشرده سازی.

*در یک فایل پایل متوسط حافظهٔ لازم برای ذخیره سازی اسم صفت خاصه ۸ بایت و متوسط تعداد صفات خاصه برای رکورد ۳ می باشد.در صورتی که تعداد رکوردها ۴ و فضای مقداری هر یک از رکوردها به شکل زیر باشد، طول متوسط رکورد کدام است؟

$V(R_1) = 30 , V(R_2) = 15 , V(R_3) = 10 , V(R_4) = 5$

حل: ابتدا متوسط مقدار صفت خاصه را محاسبه می کنیم:

$$V = \frac{30+15+10+5}{4} = 15$$

و سپس R را محاسبه می کنیم :

$$R = a'(A+V+2) \quad , \quad R = 3\times(8+15+2) = 75$$

واکشی رکورد

برای دسترسی به یک رکورد به طور متوسط باید نصف رکوردها یا نصف بلاکها را بررسی کرد:

$$T_F = \frac{n}{2}\cdot\frac{R}{t'} \quad , \quad T_F = \frac{b}{2}\cdot\frac{B}{t'}$$

(n : تعداد رکوردهای فایل - b : تعداد بلاکهای فایل)

*در فایل پایلی با مشخصات زیر، برای پاسخ به درخواست واکشی ۵۰ رکورد به یکباره، زمان متوسط واکشی یک رکورد چند میلی ثانیه است؟

$[n=10000, R=150(B), t' = 3000(\frac{B}{ms})]$

حل:

$$T_F = \frac{n}{2} \times \frac{R}{t'} = \frac{10000}{2} \times \frac{150}{3000} = 250ms$$

$$T_F(1) = \frac{2}{L} \times T_F = \frac{2}{50} \times 250 = 10ms$$

■

بازیابی رکورد بعدی

در ساختار پایل T_N برابر T_F است ، چون ارتباط ساختاری بین رکورد فعلی و بعدی وجود ندارد.

$$T_N = T_F$$

در ساختار پایل، سیستم در عمل بازیابی رکورد بعدی بسیار ناکارا می باشد.

درج

چون فایل بدون نظم است، بنابراین درج در انتها صورت می‌گیرد. بنابراین بلاک آخر در زمان $s+r+b_{tt}$ ، باید خوانده شود و بعد از انتقال رکورد جدید از ناحیه کاری به بلاک موجود در بافر، بلاک را در زمان T_{RW} بازنویسی می‌کنیم:

$$T_I = s + r + b_{tt} + T_{RW} \Rightarrow T_I = S + 3r + b_{tt}$$

در ساختار پایل، سیستم در عمل درج کارا می باشد.

بهنگام سازی

بهنگام سازی در این ساختار در حالت کلی به صورت برون ازجا می‌باشد. برای انجام بهنگام سازی ، رکورد مورد نظر واکشی شده و نشانگر حذف شده در قسمت پیشوندی آن قرار گرفته و بعد از ایجاد نسخه جدید، نسخه قدیم بازنویسی شده و نسخه جدید به انتهای فایل درج می شود: $T_U = T_F + T_{RW} + T_I$

تذکر: عملیاتی که در بافر انجام می‌شود، در ارزیابی دخالت نمی دهیم چون زمان آنها بسیار کم است.

حذف، حالت خاصی از بهنگام سازی است. کافی است که نسخه جدید را در انتهای فایل درج نکنیم

$$T_{U_{delete}} = T_F + T_{RW}$$:

*در یک فایل پایل اگر $T_N = 3ms$ و تعداد دور دیسک در دقیقه ۲۰۰۰ باشد، زمان حذف چند میلی ثانیه خواهد بود ؟

$$2r = \frac{60000}{rpm} = \frac{60000}{2000} = 30 \ \ ms \quad T_d = T_N + 2r \Rightarrow T_d = 3 + 30 = 33ms$$ ■

خواندن تمام فایل

زمان خواندن پی درپی فایل برابر است با :

$$T_{X_{SEQ}} = 2.T_F$$

فایل پایل را نمی‌توان به صورت سریال خواند چون بازیابی رکورد بعدی عملی نمی‌باشد. البته می‌توان فایل را بر روی صفت خاصی مرتب کرد، که در این حالت فایل پایل نخواهد بود و ترتیبی کامل نیز نخواهد بود چون ممکن است صفت خاصه در همه رکوردها موجود نباشد. بنابراین تنها بخشی از فایل مرتب خواهد بود. در این صورت داریم : $T_{X_{ser}} = T_{sort}(n) + Tx_{seq}$

*زمان خواندن کل فایل پایلی به صورت ترتیبی با مشخصات زیر چند ثانیه خواهد بود؟(تعداد بلاکها =۱۰۰ و اندازه هر بلاک ۲۰۰۰ بایت ، نرخ انتقال =۴۰۰۰ بایت در ثانیه)

حل :

$$T_{Xseq} = 2T_F = 2 \times \frac{b}{2} \times \frac{B}{t'} = b \times \frac{B}{t'} = 100 \times \frac{2000}{4000} = 50$$

سازماندهی مجدد

برای خارج کردن حافظه‌های هرز ناشی از عمل حذف، باید فایل را بطور متناوب سازماندهی مجدد کرد. ابتدا باید کل فایل را خوانده و سپس فایل را با حذف رکوردهای حذف شدنی بازنویسی کرد.

$$T_Y = (n+o)\frac{R}{t'} + (n+o-d)\frac{R}{t'}$$

(o ، تعداد رکوردهای درج شده از لود اولیه تا لحظه سازماندهی مجدد و d تعداد رکوردهایی است که نشانگر حذف شده خورده اند)

اگر رکورد حذف شده نداشته باشیم (d=0)، آنگاه زمان خواندن کل فایل با زمان بازنویسی برابر خواهد بود.

تذکر : ساختار پایل از نظر احیاء وضع ساختاری اولیه، نیازی به سازماندهی مجدد ندارد.

پارامترهای زمانی ساختار پایل

واکشی رکورد	$T_F = \frac{n}{2} \cdot \frac{R}{t'}$
واکشی رکورد بعدی	$T_N = T_F$
درج رکورد در فایل	$T_I = S + 3r + b_{tt}$
خواندن ترتیبی کل فایل	$T_{X_{seq}} = n \times \frac{R}{t'} = 2T_F$
سازماندهی مجدد	$T_Y = (n+o)\frac{R}{t'} + (n+o-d)\frac{R}{t'}$

فصل ۷

ساختار ترتیبی

معرفی ساختار ترتیبی

فایل ترتیبی بر دو نوع ترتیبی زمانی و ترتیبی کلیدی می باشد. در فایل ترتیبی زمانی، رکوردها به ترتیب ورود به سیستم ذخیره می‌شوند و حالت خاصی از فایل پایل است که در آن رکوردها دارای قالب ثابت مکان می باشند. در فایل ترتیبی کلیدی، رکوردها به ترتیب کلید ذخیره می شوند. این فایل نسبت به فایل پایل دو بهبود ساختاری دارد :

۱- در لود اولیه، همه رکوردها براساس مقادیر کلید مرتب هستند و این نظم با همجواری فیزیکی رکوردها پیاده سازی می شود.

۲- قالب رکوردها ثابت مکان است و نیازی به ذخیره‌سازی اسم صفت خاصه نمی‌باشد و فقط مقدار صفت خاصه ذخیره می‌شود. قالب رکورد آن به صورت زیر است:

V1	V2	…	Vn

این دو بهبود ساختاری نسبت به فایل پایل دارای مزایا و معایبی زیر می باشد :

الف- مزایا

۱- صرفه جویی در حافظه به علت ذخیره نکردن اسم صفت خاصه

۲- سادگی قالب رکورد

۳- سادگی نرم افزار پردازش فایل

۴- وجود یک استراتژی دستیابی

۵- سریع شدن پردازش سریال فایل

ب- معایب

۱- کاهش انعطاف پذیری ساختار .

۲- مصرف حافظه بیشتر به علت در نظر گرفتن فیلد برای اطلاع نهست

۳- عدم تقارن(Asymmetry) : چون استراتژی دستیابی فقط متکی به کلید است و صفات دیگر نقش ندارند.

در ساختار ترتیبی نمی‌توان طول رکورد را تغییر داد.

فایل T.L.F

فایل ثبت تراکنش ها (Transaction Log File) در عملیات درج استفاده می شود. در عمل درج، رکورد درج شدنی باید در محل منطقی اش درج شود و اگر فایل بزرگ باشد، این عمل به زمان زیادی نیاز دارد. به همین علت فایل ترتیبی را فقط خواندنی ایجاد می‌کنند و تمام عملیات تغییر دهنده در فایلی به نام T.L.F انجام می‌شود. ظرفیت فایل T.L.F با توجه به حجم عملیات تغییر دهنده و نیاز کاربر تعیین می شود.

در ساختار ترتیبی دو نوع فایل وجود دارد، یکی فایل اصلی که ترتیبی کلیدی است و دیگری فایل T.L.F (ترتیبی زمانی) که در فایل اصلی ادغام می‌شود و یک فایل ترتیبی جدید تولید می‌شود.

موارد استفاده ساختار ترتیبی

۱- کاربردهای تجاری (وقتی که رکوردها را با سیستم Batch پردازش می کنیم، پردازش سریال آنها به طور پریودیک لازم باشد.)

۲- نیازی به تغییر طول رکورد نباشد.

۳- واکشی سریع تک رکوردها مورد نظر نباشد.

۴- کاربرد سیستمی (برای خواندن فایل پایل به صورت سریال، بهتر است عملا فایل ترتیبی ایجاد شود.)

۵- در ایجاد بعضی از ساختارها لازم است که ابتدا فایل به صورت ترتیبی ایجاد گردد.

ارزیابی کارایی ساختار ترتیبی

متوسط اندازه رکورد

از آنجا که رکورد دارای قالب ثابت مکان است، می توان نوشت :

$$R = a.V$$

(a : تعداد صفات در یک نمونه رکورد) و (V : متوسط حافظه لازم برای ذخیره مقدار صفت)

ظرفیت فایل در این حالت از رابطه $S=(n+o).R$ محاسبه می شود. (تعداد رکوردهای فایل تراکنش برابر o و تعداد رکوردهای فایل اصلی برابر n است.)

واکشی رکورد

برای واکشی یک رکورد در ساختار ترتیبی دو حالت ممکن است :

۱- آرگومان جستجو کلید نباشد.

اگر آرگومان جستجو کلید نباشد، فایل حالتی از پایل خواهد بود و داریم :

$$T_F = \frac{(n+o')}{2} \times \frac{R}{t'}$$

(o': تعداد رکوردهای T.L.F در لحظه واکشی)

۲- آرگومان جستجو کلید باشد.

اگر آرگومان جستجو کلید باشد، از یک الگوریتم جستجو استفاده می‌شود. الگوریتم های جستجو در این حالت عبارتند از : جستجوی دودویی، جستجو با پرش بلاکی وجستجو با تخمین و کاوش.

جستجوی دودویی

برای اینکه بتوان الگوریتم جستجوی دودویی را پیاده سازی کرد، فایل باید در یک فضای پیوسته ذخیره شده باشد، چون اگر فایل ناپیوسته باشد، آنگاه نمی‌توان هر بار آدرس بلاک میانی را بدست آورد. در جستجوی دودویی، یک جستجو انجام می‌شود تا بلاک حاوی رکورد مورد نظر، پیدا شود. سپس برای هر بلاک که به بافر آورده می‌شود، یک جستجو دودویی درون بلاکی انجام می‌شود تا رکورد مورد نظرپیدا شود. ممکن است رکورد در فایل T.L.F باشد، که باید فایل تراکنش را نیز خواند.

$$T_F = [(s+r+b_{tt}+C_B) \times \log_2^{(\frac{nR}{B})}] + (\frac{o'}{2}.\frac{R}{t'})$$

$s+r+b_{tt}$: زمان خواندن مستقیم یک بلاک (بلاک میانی)

C_B : زمان بررسی محتوای بلاک (آیا رکورد در بلاک هست یا نه)

$\log_2^{(\frac{nR}{B})}$ یا $\log_2^b$: تعداد مراجعات به فایل(واحد جستجو در سطح خارجی بلاک است)

$\frac{O'}{2}.\frac{R}{t'}$: زمان واکشی رکورد از T.L.F

جستجو با پرش بلاکی (Skipped block search)

این جستجو مانند جستجوی خطی است، با این تفاوت که در جستجوی داخل بلاک، کلید رکورد مورد نظر با کلید رکورد آخر بلاک مقایسه می‌شود و در صورت بزرگتر بودن با پرش از بلاک، بلاک بعدی خوانده شده و در صورت کوچکتر بودن محتوای بلاک خوانده می شود. در این حالت بطور متوسط نصف بلاکها باید خوانده شود.

- در جستجوی پرش بلاکی ، اگر B_f برابر $\sqrt{n}$ باشد ، آنگاه تعداد رکوردهایی که باید بررسی شود به حداقل می رسد. به عبارتی B_f بهینه برابر $\sqrt{n}$ است.

- حداقل تعداد مقایسه ها برای یافتن رکورد به روش جستجو پرش بلاکی برابر $\sqrt{n}$ می باشد.

جستجو با تخمین و کاوش (Probing)

در این جستجو آدرس تقریبی رکورد تخمین زده می‌شود و از این آدرس جستجوی خطی انجام می‌گیرد تا رکورد مورد نظر پیدا شود. اگر رکورد در کاوش اول پیدا نشود، دوباره آدرس رکورد تخمین زده می‌شود و کاوش انجام می‌گیرد. اگر تعداد بلاکهایی که باید خوانده شوند تا رکورد پیدا شود را k در نظر بگیریم، آنگاه:

$$T_F = k(\frac{B}{t'}) + (s + r + b_{tt})$$

تذکر: کارایی این روش بستگی به چگونگی توزیع رکوردها دارد.

بازیابی رکورد بعدی

اگر رکورد اول تا رکورد یکی به آخر بلاکی را رکورد فعلی در نظر بگیریم، رکورد بعدی در همان بلاک خواهد بود و برای بازیابی آن نیاز به خواندن بلاک بعدی نمی باشد. اما اگر آخرین رکورد بلاک را رکورد فعلی در نظر بگیریم ، آنگاه رکورد بعدی در بلاک بعدی خواهد بود و باید برای بازیابی آن بلاک بعدی خوانده شود. احتمال این حالت $\frac{1}{B_f}$ می باشد. $T_N = \frac{1}{B_f}\frac{B}{t'} = \frac{R}{t'}$

اگر رکورد بعدی در T.L.F باشد ، چون هیچ ارتباط ساختاری مثلا از طریق اشاره گر بین رکورد فعلی از فایل اصلی و رکورد بعدی وجود ندارد ، بازیابی رکورد بعدی به یک جستجو در T.L.F منجر می شود که عملی محتوایی است و امکان پذیر نیست، مگر اینکه مقدار کلید رکورد بعدی را داشته باشیم و عمل بازیابی رکورد بعدی تبدیل به واکشی یک رکورد از T.L.F می شود.

درج

برای درج دو حالت را در نظر می گیریم:

۱- درج در فایلهای کوچک

در فایلهای کوچک، رکورد را در محل واقعی اش درج می‌کنیم. برای این کار محل درج را پیدا کرده و رکوردهای دیگر را به سمت انتهای فایل شیفت می دهیم و رکورد را در بلاک مورد نظر درج می‌کنیم. در این حالت به طور متوسط نصف بلاکها شیفت داده می‌شوند:

$$T_I = T_F + \frac{b}{2}(\frac{B}{t'} + T_{RW})$$

(T_F : زمان پیدا کردن محل درج و $\frac{B}{t'} + T_{RW}$: زمان شیفت یک بلاک)

۲- درج در حالت کلی

رکورد در آخرین بلاک T.L.F درج می‌شود تا در سازماندهی مجدد، بـازآرایی (Reordering) شـود و در محـل واقعی اش در فایل اصلی قرار بگیرد.

$$T_I = (s + 3r + b_{tt}) + \frac{T_y}{o}$$

$s + 3r + b_{tt}$: زمان درج در T.L.F (برابر با زمان درج در فایل پایل)

$\frac{T_y}{o}$: سرشکن کردن زمان سازماندهی مجدد بر روی o رکورد T.L.F

بهنگام سازی

رکورد مورد نظر را واکشی کرده و بعد از انجام عملیات بهنگام سازی در بافر، رکورد بهنگام درآمده را همراه با یک رکورد کوچک پیوست شده به آن (شامل تاریخ بهنگام سازی و نشانگر حذف شده) ، در T.L.F درج می کنیم.

$$T_U \approx T_F + T_I$$

تذکر: منظور از نشانگر حذف شده، این است که نسخه قدیم در فایل اصلی، در سازماندهی مجدد، حذف شدنی است.

خواندن تمام فایل

اگر فایل به طور پی در پی خوانده شود:

$$T_{Xseq} = (n + o`)\frac{R}{t`}$$

و برای خواندن سریال، T.L.F باید مرتب شود :

$$T_{xser} = T_{sort}(o') + (n + o')\frac{R}{t'}$$

تذکر: البته گاه در عمل، اگر پریود خواندن سریال طولانی باشد، فایل اصلی را با T.L.F ادغام می کنند. (سازماندهی مجدد) و سپس فایل اصلی جدید که کاملا ترتیبی است، خوانده می شود.

سازماندهی مجدد

برای سازماندهی مجدد فایل ترتیبی مراحل زیر انجام می شود:

۱- مرتب کردن فایل تراکنش (همتوالی کردن با فایل اصلی)

۲- خواندن فایل اصلی

۳- خواندن فایل تراکنش

۴- بلاک بندی مجدد رکوردها (ادغام آنها طبق نظم) ضمن خارج کردن رکوردهای حذف شدنی

۵- بازنویسی کل فایل.

$$Ty = T_{sort}(o) + n\frac{R}{t'} + o.\frac{R}{t'} + (n+o-d)\frac{R}{t'}$$

پارامترهای زمانی ساختار ترتیبی

واکشی (جستجوی خطی)	$T_F = \frac{(n+o')}{2} \times \frac{R}{t'}$
واکشی (جستجوی دودویی)	$T_F = [(s+r+b_{tt}+C_B) \times \log_2^{(\frac{nR}{B})}] + (\frac{o'}{2}.\frac{R}{t'})$
واکشی بعدی	$T_N = \frac{R}{t'}$
درج در یک فایل بزرگ	$T_I = (s+3r+b_{tt}) + \frac{T_y}{o}$
درج در یک فایل کوچک	$T_I = T_F + \frac{b}{2}(\frac{B}{t'} + T_{RW})$
خواندن ترتیبی	$T_{Xseq} = (n+o`)\frac{R}{t`}$
خواندن سریال	$T_{Xseq} = T_{sort}(o`) + (n+o`)\frac{R}{t`}$
سازماندهی مجدد	$Ty = T_{sort}(o) + n\frac{R}{t'} + o.\frac{R}{t'} + (n+o-d)\frac{R}{t'}$

فصل ۸

ساختار ترتیبی شاخص دار – ساختار چند شاخصی

معرفی ساختار های شاخص دار

در ساختار های شاخص دار (indexed structures)، به کمک شاخص می‌توان به رکوردها دسترسی پیدا کرد. شیوه بکار رفته در این ساختار در دسته روشهای دسترس تصادفی است. در ساختار شاخص دار، دو مجموعه رکورد وجود دارد:

۱- رکوردهای داده ای (فایل شاخص بندی شده)

۲- رکوردهای شاخص (فایل شاخص)

تذکر: گاه به فایل شاخص بندی شده، مجموعه داده ها و به فایل شاخص، مجموعه شاخص می گویند

فایل شاخص

مجموعه‌ای از تعدادی مدخل، که هر مدخل یک رکورد است که از دو قسمت مقدار و آدرس با طول V+P بایت تشکیل شده است. در فیلد مقدار، مقدار یک صفت ساده یا مرکب می‌تواند قرار بگیرد که در حالت خاص شامل مقدار کلید اصلی است . همچنین فیلد آدرس به یک رکورد یا گروهی از رکوردها در فایل داده‌ای اشاره می‌کند.

انواع شاخص

شاخص را می توان به دو نوع تقسیم کرد:

۱- شاخص اصلی (Primary index) : صفت خاصه شاخص، کلید اصلی باشد.

۲- شاخص ثانویه (Secondary index) : صفت خاصه شاخص، کلید ثانویه است.

تذکر: صفت خاصه ای غیر از کلید اصلی که خاصیت کلید بودن را دارد، کلید ثانویه نام دارد.

لنگرگاه (Anchor point)

نقطه ای از فایل داده ای که مدخل شاخص به آن اشاره دارد را لنگرگاه می گویند. که با توجه بـه لنگرگـاه می توان شاخص را به دو دسته تقسیم کرد :

۱- شاخص متراکم (Dense index)

اگر لنگرگاه رکورد باشد.

۲- شاخص غیرمتراکم (Non Dense index)

اگر لنگرگاه، گروهی از رکوردها (بلاک، باکت) باشد.

> در شاخص متراکم لزومی بر مرتب بودن فایل داده ای نمی‌باشد ولی در شاخص غیرمتراکم برای اینکه بتوان رکوردها را گروه بندی کرد، باید فایل داده ای مرتب باشد.

> در هر دو حالت شاخص، فایل داده ای می تواند بلاک بندی شده باشد.

شاخص خوشه ساز (Clustering index)

شاخصی که روی صفت خوشه ساز ایجاد شده باشد، شاخص خوشه ساز نام دارد. این شاخص غیـر متـراکم است.

> صفتی که مقادیرش در فایل تکراری است، صفت خوشه ساز نام دارد.

> صرف تکراری بودن مقادیر صفت خاصه ایجاب نمی کند که شاخص حتما غیر متـراکم و خوشـه سـاز باشد و می توان شاخص متراکم هم ایجاد کرد و برای اجتنـاب از تکـرار مقـدار شـاخص در مـدخلهای شاخص، از یک سطح آدرس دهی غیر مستقیم می توان استفاده کرد.

شاخص سخت افزاری

شاخص سخت افزاری از دو سطح شاخص تشکیل شده است :

الف- شاخص استوانه (Cylinder index)

در هر مدخل شاخص استوانه،موارد زیر وجود دارد :

۱- کوچکترین مقدار رکوردهای آن استوانه

۲- شماره استوانه

۳- آدرس شاخص شیارهای همان استوانه

ب- شاخص شیار (Track index)

در هر مدخل از این شاخص موارد زیر وجود دارد :

۱- شیارهای هر استوانه

۲- کوچکترین مقدار رکوردهای همان شیار

۳- شماره شیار

عیب شاخص سخت افزاری، عدم انعطاف پذیری می باشد.

سیستم فایل ISAM ازIBM مثالی از شاخص بندی سخت افزاری است.

ظرفیت نشانه روی بلاک شاخص

تعداد مدخلهای یک بلاک شاخص را ظرفیت نشانه روی بلاک شاخص می‌گویند که برابر است با:

$$y = \left\lfloor \frac{B}{V+P} \right\rfloor$$

شاخص چند سطحی (Multi Level index)

اگر تعداد مدخلهای شاخص زیاد باشد، آنرا در چند سطح می سازند تا زمان یافتن مدخل شاخص کمتر شود. تعداد سطوح شاخص را عمق شاخص (index deep) می‌گویند و برابر است با :

$$x = \left| \log_y^{e_1} \right|$$

تذکر: می توان به جای e_1 مقدار b را قرار داد.

اگر x=1 باشد، آنگاه شاخص را خطی می‌گویند.

شاخص سطح اول متراکم یا غیرمتراکم است.

- شاخص های سطح دوم به بعد، غیر متراکم هستند.

- بالاترین سطح شاخص (سرشاخص)، به اندازه یک بلاک است

- سر شاخص (Master index) در حافظه اصلی ذخیره می شود.

- بلاکهای سایر سطوح شاخص در شیارهای استوانه آغازین یا شیارهای آغازین هر استوانه ذخیره می شوند. $y^{X-1} \le e_1 \le y^X$

- هر چه x بیشتر باشد، تعداد دفعات دستیابی برای واکشی رکورد بیشتر خواهد بود. بنابراین باید مقدار x را کاهش داد و برای اینکار باید y را افزایش دهیم.

- برای افزایش y می‌توان بلاک شاخص را بزرگتر گرفت، اما اندازه بلاک شاخص برابر اندازه بلاک داده ای می‌باشد و انتخاب آن به امکانات بافرینگ سیستم بستگی دارد. بنابراین مدخل شاخص را فشرده کرده تا y افزایش یابد.

جمع بندی انواع شاخص

۱- روی صفت خاصه غیر کلید ← شاخصهای خوشه ساز(غیر متراکم)

۲- روی صفت خاصه کلید

الف- روی کلید اصلی (معمولا غیر متراکم)

ب- روی کلید ثانوی (معمولا متراکم)

معایب شاخص بندی

معایب ساختارهای شاخص دار به ویژه در حالت چند شاخصی عبارتند از :

۱- مصرف حافظه برای ایجاد شاخص.

۲- فزونکاری (Over head) در عملیات ذخیره‌سازی.

ساختار ترتیبی شاخص دار (indexed sequential)

این ساختار برای تسریع عمل واکشی تک رکورد از فایل ترتیبی طراحی شده و شامل چهار جزء است:

۱- فایل ترتیبی (ناحیه اصلی) ۲- ناحیه سرریزی (overflow area)

۳- اشاره گرها ۴- مجموعه شاخص

ویژگی های ساختار ترتیبی شاخص دار

۱- فایل ترتیبی روی صفت خاصه کلید مرتب است.

۲- شاخص ایستا است.

۳- شاخص را نرم افزاری فرض می کنیم.

۴- شاخص غیر متراکم است.

۵- از شاخص برای تسریع واکشی رکوردها استفاده می شود و در عملیات خواندن فایل کاربرد ندارد.

۶- شاخص ناظر به ناحیه سرریز نمی‌باشد و فقط به ناحیه اصلی ناظر است.

۷- شاخص ایستا است، بنابراین شاخص در سازماندهی مجدد تنظیم می‌شود نه همروند با عملیات ذخیره‌سازی در فایل.

۸- وجود زنجیره ها از ناحیه اصلی به ناحیه سرریز، امکان پردازش سریال را فراهم می کند.

۹- شاخص بندی را آنقدر ادامه داده تا اندازه فایل شاخص کوچکتر یا مساوی اندازه بلاک شود.

نحوه انجام عملیات خواندن در ساختار ترتیبی شاخص دار

۱- خواندن پی در پی

در خواندن پی در پی، ناحیه اصلی و ناحیه سرریز، بلاک به بلاک خوانده می‌شوند و چون رکوردهای ناحیه سرریزی مرتب نمی‌باشند، رکوردها به طور سریال خوانده نمی‌شوند.

۲- خواندن سریال

در خواندن سریال، بلاکهای ناحیه اصلی روی کلید خوانده شده تا به بلاک یا رکوردی برسد که دارای اشاره گری به ناحیه سرریز باشد.در این حالت، به ناحیه سرریز رفته و زنجیره سرریزی طی می‌شود. بعد از پایان زنجیره به ناحیه اصلی برمی گردیم و خواندن را ادامه می دهیم.

* در فایلی با ده میلیون رکورد ۲۰۰ بایتی، چند سطح شاخص نیاز است؟

(V=14 byte , P=6 byte , B=2000 byte)

حل: تعداد بلاکهای فایل برابر است با :

$$b = \frac{n}{B_f} = \frac{10^7}{10} = 10^6$$

ظرفیت نشانه روی بلاک شاخص برابر است با :

$$y = \left\lfloor \frac{B}{V+p} \right\rfloor = \left\lfloor \frac{2000}{20} \right\rfloor = 100$$

تعداد مدخل در سطح اول و دوم و سوم شاخص برابر است با :

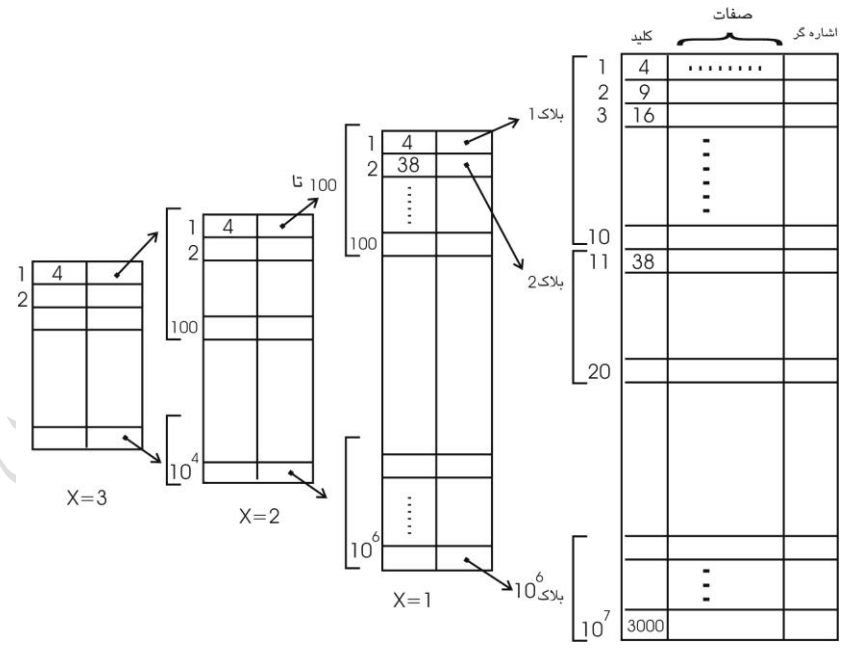
$$e_1 = b = 10^6 \ , \ e_2 = \frac{10^6}{100} = 10^4 \ , \ e_3 = \frac{e_2}{y} = \frac{10^4}{100} = 100$$

عمق شاخص برابر ۳ می باشد، چون حجم فایل شاخص سوم برابر ۲۰۰۰ بایت است (۱۰۰ مدخل ۲۰ بایتی) که برابر اندازه یک بلاک است. به عبارتی، شاخص بندی را آنقدر ادامه می‌دهیم تا اندازه فایل شاخص کوچکتر یا مساوی اندازه بلاک شود.

تذکر: می‌توان با توجه به رابطه زیر، مستقیما تعداد سطوح شاخص را بدست آورد:

$$X = \lceil \mathrm{Log}_y^b \rceil = \left\lceil \mathrm{Log}_{100}^{10^6} \right\rceil = \left\lceil \mathrm{Log}_{10^2}^{10^6} \right\rceil = \left\lceil \frac{6}{2} \right\rceil = 3$$

شکل زیر این شاخص بندی را نشان می دهد:



*فایلی با مشخصات زیر را در نظر بگیرید. تعداد مدخلها در سطح دوم شاخص کدام است؟

(n=1000000 , B=2000 Byte , R=200Byte , V+P=20 Byte)

حل :

$$B_F = \frac{B}{R} = 10 \ , \ y = \frac{B}{V+P} = \frac{2000}{20} = 100$$

$$e_1 = \frac{n}{B_F} = \frac{10^6}{10} = 10^5 \ , \ e_2 = \frac{10^5}{100} = 1000$$

■

*در فایلی با مشخصات زیر تعداد سطوح شاخص کدام است؟

(n=10^7 , $B_f = 10$, V=14 , P=6 , B=2000)

حل :

$$y = \left\lfloor \frac{B}{V+P} \right\rfloor = \left\lfloor \frac{2000}{14+6} \right\rfloor = 100 \ , \ b = \frac{n}{B_f} = \frac{10^7}{10} = 10^6$$

$$x = \left\lceil \log_y^b \right\rceil = \left\lceil \log_{100}^{10^6} \right\rceil = \left\lceil \frac{6}{2} \right\rceil = 3$$

■

* میزان حافظه لازم (دیسک) برای شاخص چند بایت است؟

$$(e_1 = 10^5, e_2 = 1000, e_3 = 10, x = 3, v + p = 20)$$

حل:

$$S_I = \sum_{i=1}^{x-1} S_i = S_1 + S_2 = 10^5 \times 20 + 1000 \times 20 = 2020000 byte$$

(توجه شود که سطح سوم در حافظه اصلی نگهداری می شود نه در دیسک)

■

روش های انتخاب فضای لازم برای درج رکوردهای سرریزی

۱- در نظر گرفتن جا در هر بلاک در لود اولیه

در این روش لوکالیتی رکوردها قوی است ولی امکان کمبود جا در برخی بلاکها و امکان خالی ماندن فضا در بلاکهای دیگر وجود دارد.

۲- ایجاد یک فایل جداگانه

در این روش چون بین فایل اصلی و این فایل باید ارتباط برقرار شود، زمانگیر خواهد بود. همچنین لوکالیتی رکوردهای سرریزی ضعیف است. از این روش در فایل ترتیبی استفاده شد.

۳- در نظر گرفتن ناحیه ای جداگانه در همان فایل داده ای

تذکر : سومین مورد، مناسبترین راه حل است.

طرح های تخصیص فیزیکی

فضای انتخاب شده برای ناحیه سرریز در محیط فیزیکی به فایل به دو صورت ممکن است تخصیص داده شود:

۱- تخصیص استوانه هایی در انتهای فایل، برای ایجاد ناحیه جداگانه

روش مناسبی نیست، چون باعث ضعیف شدن لوکالیتی رکوردهای سرریزی خواهد شد و در نتیجه،S افزایش می یابد.

۲- تخصیص شیارهایی در انتهای استوانه، به عنوان ناحیه سر ریزی استوانه

این راه حل s را کاهش می دهد، چون رکوردهای سرریزی هر استوانه در همان استوانه قرار می گیرند. با پر شدن ناحیه سرریزی یک استوانه، باید ناحیه سرریزی ثانویه ایجاد شود و یا فایل مجددا سازماندهی شود.

منطقا نیازی به یکسان بودن اندازه ناحیه سرریزی در هر استوانه نیست ولی معمولا یکسان گرفته می شوند.

تکنیکهای درج سرریزی

۱- درج در اولین بلاک جادار در ناحیه سرریزی

رکورد جدید وارد اولین بلاک جادار در ناحیه سرریز می‌شود و از رکورد منطقاً قبلی به آن اشاره گر ایجاد می‌شود.برای هر رکورد چه در ناحیه اصلی و چه در ناحیه سرریزی یک فیلد اشاره گر وجود دارد. مبدا زنجیره ها در ناحیه اصلی است و زنجیره سرریزیهای رکورد در این روش وجود دارد.

۲- درج با جابجایی (Push through)

رکورد جدید در بلاک مربوط در ناحیه اصلی، در محلی که منطقاً باید قرار بگیرد، درج می‌شود و رکوردهای بعدی همان بلاک (به غیر از اولین رکورد بلاک)به سمت انتهای بلاک شیفت داده می‌شوند و رکورد آخر بلاک به اولین بلاک جادار در ناحیه سرریزی، منتقل می‌شود.

در روش دوم زنجیره سرریزیهای بلاک(Block Overflow Chain) وجود دارد و برای هر بلاک از ناحیه اصلی یک اشاره گر وجود دارد.

پردازش سریال فایل در روش دوم ساده تر و طول زنجیره بیشتر است.

کاربرد ساختار ترتیبی شاخص دار

۱- وقتی که پردازش سریال برحسب مقادیر کلید مطرح است (نه صفات دیگر آنها)

۲- وقتی که واکشی تک رکوردها از طریق کلید آنها، عملی رایجی می باشد.

۳- سیستم های داده پردازی تجاری ـ مدیریتی.

ارزیابی کارایی ساختار ترتیبی شاخص دار

با فرض مفروضات زیر، کارایی ساختار ترتیبی شاخص دار را بررسی می‌کنیم:

۱- غیرمتراکم بودن سطح اول شاخص

۲- همتوالی بودن فایل داده‌ای و فایل شاخص

۳- قرار داشتن بلاکهای شاخص در یک استوانه و داشتن ناحیه سرریزی استوانه

۴- تکنیک درج Push through

۵- ایستا بودن ساختار شاخص

۶- پر بودن ناحیه اصلی و بلاکهای شاخص و خالی بودن ناحیه سرریز بعد از سازماندهی مجدد

۷- حذف منطقی رکوردها و انجام حذف فیزیکی در سازماندهی مجدد

۸- بلاک بندی شدن فایل داده ای و شاخص با طول هر بلاک B بایت.

متوسط اندازه رکورد

عوامل مؤثر در محاسبه R، عبارتند از:

۱- حافظه لازم برای یک رکورد از ناحیه اصلی

۲- حافظه مصرف شده برای ناحیه سرریزی به ازاء یک رکورد از ناحیه اصلی

۳- حافظه مصرف شده برای شاخص به ازاء یک رکورد از ناحیه اصلی

$$R = R_{data} + R_{over} + R_{index}$$

$$R_{data} = (av + \frac{P}{B_f})$$

$$R_{over} = \frac{o}{n+o}(av + P)$$

$$R_{index} = \frac{S_I}{n+o}$$ (S_I : کل حافظه مصرف شده برای شاخص)

واکشی رکورد

برای واکشی، باید ابتدا سرشاخص که در حافظه اصلی است را بررسی کنیم، سپس درسطح شاخص جستجو کرده تا به مدخل مربوطه در سطح اول برسیم. یعنی در هر سطح شاخص، یک بلاک خوانده شده و بعد از پیدا کردن آدرس مورد نظر در ناحیه اصلی، آنرا می خوانیم. (البته احتمال رفتن به ناحیه سرریز و جستجو در زنجیره سرریزی‌ها وجود دارد.)

$$T_F = C_B + 2S + (X + \frac{1}{2}\frac{O`}{n+O`} + \frac{1}{2}.\frac{O`}{n})(r + b_{tt})$$

هر چه x کمتر باشد، زمان واکشی رکورد از ناحیه اصلی کمتر خواهد بود.

بازیابی رکورد بعدی

با توجه به آخرین رکورد واکشی شده، متوجه می شویم که آیا رکورد بعدی در بلاکی از ناحیه اصلی است یا در بلاکی از ناحیه سرریز. اگر رکورد بعدی در بلاکی از ناحیه اصلی باشد، با زمان $\frac{1}{B_F}b_{tt}$ بدست می آید و اگر در ناحیه سرریز باشد، در زمان $r + b_{tt}$ بدست می آید:

$$T_N = \frac{1}{B_F}b_{tt} + \frac{o`}{n+o`}(r + b_{tt})$$

($\frac{o`}{n+o`}$: احتمال اینکه رکورد در ناحیه سرریز باشد .)

ارزیابی دقیق تر

با توجه به وضعیت رکورد بعدی نسبت به رکورد فعلی، شش حالت ممکن است بوجود آید:

۱- رکورد فعلی در بلاکی از ناحیه اصلی است و رکورد بعدی نیز در همان بلاک است و بلاک در بافر است.

۲- رکورد فعلی آخرین رکورد بلاک است از ناحیه اصلی و رکورد بعدی در بلاک بعدی است از همان استوانه.

۳- رکورد فعلی آخرین رکورد بلاک از آخرین بلاک استوانه است و رکورد بعدی در بلاک بعدی از استوانه دیگر می باشد.

۴- رکورد فعلی آخرین رکورد بلاک است و رکورد بعدی در بلاکی از ناحیه سرریزی است.

۵- رکورد فعلی در بلاکی از ناحیه سرریزی است و رکورد بعدی هم در بلاکی از ناحیه سرریزی و از همان استوانه است.

۶- رکورد فعلی در بلاکی از ناحیه سرریزی است و رکورد بعدی در بلاکی از ناحیه اصلی.

با در نظر گرفتن حالات ششگانه به کمک مقدار pro و ساده کردن خواهیم داشت :

$$T_N = (\frac{1 - pro}{B_F} + pro)(r + b_{tt})$$

که با قرار دادن $pro = \frac{o'}{n+o'}$ خواهیم داشت :

$$T_N = \frac{n + o`B_F}{(n + o`)B_F}(r + b_{tt})$$

درج

مراحل درج یک رکورد به صورت زیر است:

۱- پیدا کردن بلاکی که رکورد باید در آن درج شود.

۲- وارد کردن رکورد در این بلاک و خارج کردن آخرین رکورد بلاک و قرار دادن در بافر کمکی و ساختن فیلد PTR به رکورد جابجا شونده.

۳- بازنویسی این بلاک

۴- خواندن بلاکی از ناحیه سرریز

۵- وارد کردن رکورد خارج شده از بلاک اصلی، در این بلاک

۶- بازنویسی همین بلاک.

$$T_I = T_F + T_{RW} + r + b_{tt} + T_{RW}$$

با در نظر گرفتن $T_{RW} = 2r$ داریم :

$$T_I = T_F + 5r + b_{tt}$$

در عمل درج در ساختار ترتیبی شاخص دار، به دو بازنویسی نیاز می باشد.

بهنگام سازی

اگر مقدار کلید تغییر نکند، می‌توان بهنگام سازی درجا (inplace) انجام داد. برای اینکار رکورد بهنگام در آوردنی را واکشی کرده و نسخه جدید آن را در بافر ایجاد کرده و سپس بازنویسی می‌کنیم:

$$T_{U_{inplace}} = T_F + 2r$$

اما در حالت کلی، نسخه قدیمی رکورد را نشانگر حذف می زنیم و رکورد جدیدی که در بافر ساخته‌ایم را درج می‌کنیم:

$$T_{U_{outplace}} = T_F + T_{RW} + T_I$$

که با قرار دادن مقدار T_I و T_{RW} خواهیم داشت :

$$T_{U_{outplace}} = 2T_F + 7r + b_{tt}$$

خواندن کل فایل

۱- خواندن ترتیبی

$$T_{xseq} = (n + o`)\frac{R}{t'}$$

۲- خواندن سریال

در این حالت اولین رکورد واکشی شده و بقیه رکوردها در یک سلسله عملیات بازیابی رکورد بعدی، خوانده می‌شوند.

$$T_{xser} = T_F + (n + o' - 1)T_N$$

سازماندهی مجدد

هنگامی که ناحیه سرریز پر می‌شود یا طول زنجیره ها طولانی می‌شود، فایل را سازماندهی مجدد می‌کنند. برای اینکار فایل را به طور سریال خوانده و رکوردها را با خارج کردن حذف شدنی‌ها، بلاک بندی کرده و بعد از بازنویسی نسخه جدید ساختار شاخص را بازسازی می‌کنند.

$$T_y = T_{xser} + (n + o - d)\frac{R}{t'} + \frac{S_I}{t'}$$

$\frac{S_I}{t'}$: زمان بازنویسی بلاکهای شاخص

معایب ساختار ترتیبی شاخص دار

۱- عدم تقارن

۲- ایستا بودن شاخص

۳- مسئله درج سرریزی ها (زنجیره های طولانی کارایی سیستم را کم می کند.)

پارامترهای زمانی ساختار ترتیبی شاخص دار

واکشی رکورد	$T_F = T_{F(main)} + T_{F(0ver)}$
واکشی رکورد بعدی	$T_N = \frac{n + o`B_F}{(n + o`)B_F}(r + b_{tt})$
درج یک رکورد	$T_I = T_F + 5r + b_{tt}$
خواندن ترتیبی کل فایل	$T_{xseq} = (n + o`)\frac{R}{t'}$
خواندن سریال کل فایل	$T_{xser} = T_F + (n + o' - 1)T_N$
سازماندهی مجدد	$T_y = T_{xser} + (n + o - d)\frac{R}{t'} + \frac{S_I}{t'}$

ساختار چند شاخصی(فایل شاخص دار)

ساختار ترتیبی شاخص دار دارای معایبی مانند عدم تقارن، ایستا بودن شاخص و مشکل بودن درج رکوردهای جدید می‌باشد که این معایب در ساختار چند شاخصی بر طرف شده است. اجزاء اصلی ساختار چند شاخصی عبارتند از:

۱- فایل داده ای

۲- چندین فایل شاخص

چند نکته :

۱- فایل داده ای در این ساختار از نوع پایل می باشد.

۲- ساختار چند شاخصی دارای تقارن است. (چون روی تعدادی یا حتی تمام صفات خاصه می‌توان شاخص ایجاد کرد.)

۳- وقتی که روی تمام صفات خاصه شاخص ایجاد شود، فایل را وارون می‌گویند.

۴- چون بر روی هر یک از صفات خاصه می‌توان شاخص ایجاد کرد، بنابراین کاربر می‌تواند هر یک از آنها را به عنوان آرگومان جستجو به کار ببرد.

۵- اگر a صفت خاصه در فایل باشد، حداکثر a فایل شاخص می‌توان داشت. هر چه تعداد صفات شاخص بیشتر شود، فایل در بازیابی کاراتر و عدم تقارن آن کمتر می شود.

موراد استفاده از ساختار چند شاخصی

۱- محیط هایی که واکشی سریع تک رکوردها مطرح است.

۲- محیط هایی که فایل حالت نامانا دارد یعنی داده‌ها مرتباً در حال تغییر می‌باشند.

۳- محیط هایی که کاربر بخواهد از طریق صفات خاصه مختلف به رکوردها دستیابی داشته باشد.

به عنوان مثال در سیستم رزرواسیون جا در خطوط هوایی که در آن اطلاعات مربوط به جا مرتبا تغییر می کند، استفاده از ساختار چند شاخصی مناسب است.

ساختار شاخص

تنها استراتژی دستیابی در فایل چند شاخصی، همان شاخص است. ساختار داده ای در ایجاد فایل شاخص، درخت متعادل (B-TREE) است. در این درخت عمق تمام شاخه ها از ریشه تا گره انتهایی یکسان است. این درخت برای نگهداری اطلاعات با تغییر زیاد کاربرد دارد. یک B – TREE از مرتبه m، یک درخت جستجوی m طرفه (m-way search TREE) است که یا تهی است یا دارای خواص زیر است:

۱- گره ریشه دارای حداقل دو فرزند است.

۲- همه گره ها دارای $\left\lceil \frac{m}{2} \right\rceil$ فرزند هستند. (بجز ریشه و گره های خطا)

۳- همه گره های خطا در یک سطح هستند.

۴ - حداقل تعداد داده ها برابر $\left\lceil \frac{m}{2} - 1 \right\rceil$ و حداکثر m-1 می باشد.

۵- حداقل تعداد فرزندان برابر $\left\lceil \frac{m}{2} \right\rceil$ و حداکثر m می باشد.

۶- اگر داده های یک گره بیش از حداکثر شود، نود باید شکسته (Split) شود.

چند نکته :

۱- درخت B قادر به پاسخگویی به تقاضاهای ساده، محدوده ای و منطقی می باشد.

۲- درخت B وقتی در پاسخگویی به تقاضاهای محدوده ای موثر است که تنوع داده ها زیاد باشد.

۳- ساختار B-TREE امکان می دهد تا فایل شاخص رفتاری پویا داشته باشد و همروند با عملیات روی فایل تنظیم شود.

۴- می توان فایل را به کمک شاخص به طور سریال پردازش کرد. مثلا در بازیابی رکورد بعدی یا خواندن تمام فایل به پردازش سریال نیاز است.

۵- هر بلاک شاخص، گرهی از درخت B-TREE می‌باشد که در لود اولیه قسمتی از آن خالی است. یعنی چگالی لود اولیه کمتر از صددرصد می باشد.

۶- ظرفیت اشاره گر اسمی برابر $Y = \left[\frac{B}{V+P}\right]$ می باشد و به تمامی در لود اولیه پـر نمی‌شـود و تعـدادی مدخل به عنوان رزرو در نظر گرفته می‌شوند. ظرفیت واقعی اشاره گر در لود اولیه حداقل $\frac{y}{2}$ و حداکثر y می‌باشد. یعنی در لود اولیه، حداقل نیمی از مدخلهای یک گره درخت پر است.

ساختار شاخص در درج

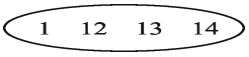
رکورد را در فایل داده ای اضافه کرده و ارتباط ساختاری را برقرار می سازیم. اگر برای ایجاد مدخل در بلاک شاخص مربوطه در سطح اول دیگر جا نباشد ($y = y_{eff}$)، آنگاه بلاک شاخص پر شده را تقسیم (split) کرده و یک بلاک شاخص خالی به نام بلاک همراه (Partner) به فایل شاخص اختصاص داده و نیمی از مدخلهای پر شده، به این بلاک منتقل می شوند. این گره جدید باید با گره ای در سطح بالاتر مرتبط شود، بنابراین در عمل تقسیم بلاک پر شده، اقلا سه بلاک شاخص باید ایجاد و یا بهنگام در آیند.

* با داده های زیر یک B-TREE به صورت ۵ طرفه بسازید:

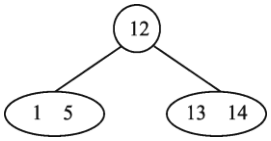
12 , 13 , 14 , 1 , 5 , 9 , 3 , 10

حل:

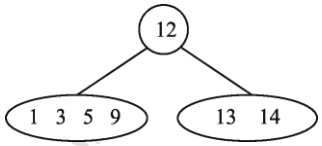
هر گره در درخت ۵ طرفه شامل حداکثر ۴ داده می باشد و بنابراین ۴ داده اول را خوانده و مرتب شده آن را در ریشه قرار می دهیم:



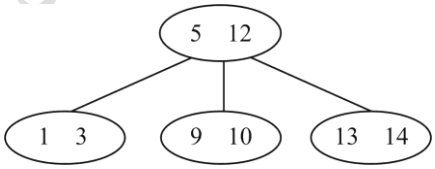
با درج مقدار ۵ درخت زیر حاصل می شود. در واقع گره شکسته می شود و مقدارهای ۱و۵ در چپ ریشه و مقدارهای ۱۳و۱۴ در راست ریشه قرار می گیرند و مقدار ۱۲ در ریشه قرار می گیرد:



و با درج مقدارهای ۹ و ۳ ، درخت زیر حاصل می شود :



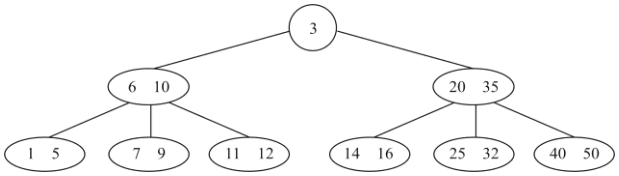
و در نهایت مقدار ۱۰ را درج می کنیم :



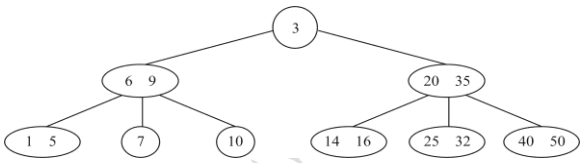
ساختار شاخص در حذف

رکورد حذف شدنی را با ضبط نشانگر "حذف شده" از فایل داده ای حذف کرده و همچنین ارتباط ساختاری آن با درخت شاخص را از بین می بریم. اگر بعد از حذف، تعداد مدخلهای واقعی از $\frac{y}{2}$ کمتر شود و مجموع مدخلهای واقعی این بلاک و بلاک همراه آن از y کمتر شود، دو بلاک با هم ادغام می شوند.

* مطلوب است به ترتیب حذف داده های ۱۱و۱۲ از درخت B سه طرفه زیر.

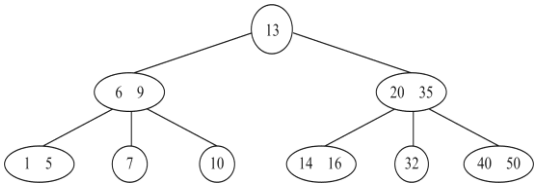


حل : حذف ۱۱ به راحتی انجام می شود ولی برای حذف ۱۲ عمل توزیع انجام می شود:

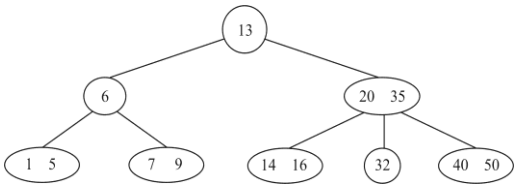


تذکر: حداقل ظرفیت هر گره در درخت ۳ طرفه برابر یک می باشد: $\left(\left\lceil \frac{3}{2}-1 \right\rceil\right)$

* مطلوب است حذف داده ۱۰ از درخت B زیر :



حل : چون گره با مقدار ۱۰ دارای ظرفیت کمتر از ۱ می شود و همزاد آن نیز از حداقل بیشتر ندارد ، عمل توزیع ممکن نیست و عمل ترکیب انجام می گیرد:



واکشی رکورد

واکشی رکوردی که بر روی نشانوند جستجوی آن شاخص وجود نداشته باشد، مانند واکشی در فایل پایل است. در صورت وجود شاخص، ابتدا بلاکهای شاخص را در شاخه مناسب خوانده و سپس بلاک داده ای حاوی رکورد مورد نظر را می خوانیم.

> برای واکشی سریع تک رکوردها، لزومی ندارد که حتماً از کلید اصلی به عنوان آرگومان جستجو استفاده کرد.

بازیابی رکورد بعدی

رکورد بعدی، رکوردی است که مدخل بعدی به آن اشاره می‌کند. با فرض اینکه مدخل بعدی در همان بلاکی باشد که مدخل ناظر به رکورد فعلی در آن قرار دارد، آنگاه آدرس رکورد بعدی در دسترس است و کافی است آن را بخوانیم. اگر بلاکهای شاخص موجود در سطح اول براساس نظم به یکدیگر متصل شده باشند، آنگاه آدرس بلاک بعدی شاخص، از بلاک فعلی به دست می‌آید.

درج

بلاک مربوطه باید خوانده شود و بعد از قرار دادن رکورد در آن، بلاک بازنویسی می‌شود. سپس مدخل شاخص مربوطه در سطح اول ایجاد می‌شود و به احتمالی امکان عمل تقسیم وجود دارد.

بهنگام سازی

چون فایل داده ای پایل است، بنابراین در حالت کلی، بهنگام سازی برون از جا انجام می شود. عملیات لازم عبارتند از:

۱- واکشی رکورد بهنگام در آمدنی

۲ – ساختن نسخه جدید و ضبط نشانگر "حذف شده" در نسخه قدیم

۳- درج نسخه جدید

۴- حذف نسخه قدیم

خواندن تمام فایل

جهت خواندن کل فایل به صورت سریال، رکورد اول واکشی شده و بقیه رکوردها طی یک سلسله عملیات بازیابی رکورد بعدی به دست می آیند.

سازماندهی مجدد

در ساختار چند شاخصی به علت وجود یکی از دلایل زیر می‌توان عمل سازماندهی مجدد را انجام داد:

۱- باز پس گیری حافظه‌های هرز

۲- برگرداندن یکنواختی توزیع گره های درخت شاخصی که در اثر حذف یا درج گره‌های همسایه، از بین رفته است.

برای سازماندهی مجدد، ابتدا فایل را خوانده و سپس با حذف رکوردهای حذف شدنی، فایل را بازنویسی کرده و سپس شاخص ها سازماندهی مجدد می شوند.

مشکل B-TREE این است که برای دستیابی به هر داده ، باید از طریق داده های موجود در پدر آن داده، خود را به داده مورد نظر برسانیم. برای رفع این مشکل از B^+ -TREE استفاده می کنیم.

یک B–TREE از مرتبه m که همه گره های خطا در آن در سطح L + 1 قرار دارند، حداکثر $m^L - 1$ کلید دارد.

فصل ۹

ساختار فایل مستقیم

معرفی ساختار مستقیم

ساختار فایل مستقیم (درهم یا Hashed)، بـه سـاختارهای قبلـی وابسـته نمی‌باشـد و اسـتراتژی دسـتیابی مستقیم به رکوردها از طریق آدرس هر رکورد تامین می‌شود. در هنگام ایجاد فایـل در لـود اولیـه، یکـی از صفات خاصه رکورد، کلید در نظر گرفته می‌شود. بعد از آنکه مقدار کلید به سیستم فایل داده شد، سیسـتم پردازشی را روی آن انجام داده و آدرسی که رکورد باید در آن قرار گیرد را بر می گردانـد. ایـن پـردازش را KAT یعنی تبدیل کلید به آدرس می‌گویند و تابعی که عمل تبدیل کلید به آدرس را انجام می دهد را تـابع درهم ساز می نامند.

فایل در این ساختار دارای یک فضای آدرس با m آدرس از 1 تا m (یا از صـفر تـا m-1) مـی باشـد کـه هـر آدرس به یک حفره (slot) مرتبط است. در ایـن فضـای آدرس بایـد n رکـورد درج شـود (n<m) کـه $\frac{n}{m}$ را فاکتور لود می‌گویند و همواره کوچکتر یا مساوی یک می‌باشد.

یکی از توابع درهم ساز (مبدل) ، تابع تقسیم نام دارد که آدرس ، باقیمانده تقسیم صحیح کلید به عدد اول نزدیک به m می باشد. به طو نمونه آدرس حاصل از کلید 12345678 در فضای آدرسی 1..5000 برابر ۳۰۸۸ می باشد ، چون :

12345678 mod 4997 = 3088

(عدد ۴۹۹۷ ، عدد اول نزدیک به ۵۰۰۰ می باشد.)

تصادف (collision)

تصادف یا برخورد وقتی رخ می دهد که پس از اعمال تابع مبدل کلید به آدرس، به ازای دو کلیـد متفـاوت، آدرس یکسانی تولید شود. $K_i \neq K_j \Rightarrow a_i = a_j$

> هرچه تعداد رکوردهـای تصـادفی کمتـر باشـد، واکشـی رکوردهـا سـریعتر خواهـد بـود. (رکوردهـای غیرتصادفی با یکبار دستیابی بازیابی می‌شوند.)

روش های حل مشکل تصادف و درج سرریزی ها

۱- ایجاد فایل جداگانه

۲- در نظر گرفتن ناحیه ای جداگانه در خود فایل

۳- جستجوی خطی و درج تصادفی در اولین باکت جادار

۴- درهم سازی مجدد (Rehashing)

۵- ایجاد زنجیره بدون جایگزینی

۶- ایجاد زنجیره با جایگزینی

تذکر: روشهای ۱و۲ رایج نمی‌باشند، روشهای ۳و۴ را آدرس دهی باز(open addresing) و روشهای ۵و۶ را chaining می نامند.

ایجاد فایل جداگانه

در این روش یک فایل جداگانه در نظر گرفته شده و رکوردهای تصادفی در آن درج می‌شوند. این روش دارای دو عیب زیر می باشد:

۱- حفره های هرز در فایل اصلی بوجود می آید.

۲- سیستم فایل باید دو فایل را پردازش کند.

در نظر گرفتن ناحیه ای جداگانه در خود فایل

در این روش ناحیه ای جداگانه مثلاً درانتهای هر استوانه برای درج رکوردهای سرریز استفاده می‌شود. عیب این روش در این است که امکان دارد باکت هایی از ناحیه جداگانه پر شود در حالیکه باکت هایی از ناحیه اصلی دارای حفره های خالی باشند.

جستجوی خطی و درج تصادفی در اولین باکت جادار

در این روش با شروع از محل تصادف، جستجوی خطی به سمت انتهای فایل شروع شده و رکورد تصادفی در اولین باکت جادار درج می‌شود. معایب این روش عبارتند از :

۱- طولانی شدن زمان جستجوی خطی برای واکشی رکورد تصادفی وقتی که فاکتور لود افزایش می یابد.(به علت بررسی رکوردهای غیر مرتبط)

۲- رکوردها را نمی‌توان به راحتی حذف کرد، چون احتمال دارد با وجود اینکه رکوردی در فایل وجود داشته باشد، پیام رکورد یافت نشد، صادر شود. زیرا در صورت حذف یک کورد، حفره ای خالی می شود و چون کاوش خطی برای یافتن یک رکورد مورد نظر ، به محض برخورد با یک حفره خالی پایان می پذیرد، سیستم قادر به یافتن رکورد مورد نظر نخواهد بود.

در این روش چون هر باکت پر شده ، از حفره های خالی نزدیکترین باکت استفاده می کنـد، بـه روش " همسایگی بد " معروف است.

درهم سازی مجدد (Rehashing)

در این روش کلید رکورد تصادفی به تابع دیگری داده می شود. اگر اولین تابع به صورت (۱) عمل کند و تابع مشابه دیگری به صورت (۲) عمل کند، در نهایت آدرس جدید برابر A+D خواهد بود.

A=key mod m1 (1)
D=key mod m2+1 (2)

(m1 عدد اول نزدیک به m و m2 عدد اول بلافاصله کوچکتر از m1 می باشد.)

۱- اگر A+D از m بزرگتر باشد ، عمل جمع به پیمانه m باید انجام شود.

۲- چون معمولاً مقادیر D بزرگتر از BKf می‌باشد، باکت بندی فایل لزومی ندارد.

۳- تعداد حفره های بررسی شونده در این روش، کمتر از روش کاوش خطی است.

روش ایجاد زنجیره بدون جایگزینی

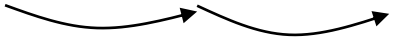
در روش جستجوی خطی، رکوردهای غیرمرتبط بررسی می‌شوند. برای جلوگیری از بررسی این رکوردها، رکوردهای تصادفی به یکدیگر زنجیر می‌شوند تا از رکوردهای غیر دخیل پرش شود.

* تابع درهم ساز را روی کلید چند رکورد اعمال کرده ایم و فایل بعد از درج این رکوردها به صورت زیر در آمده است. رکورد R8 با آدرس ۱۵ و رکورد R9 با آدرس ۱۸ و رکورد R10 با آدرس ۱۵ را درج کنید.

…..	R1	R2	R3		R4	R5		R6	R7		….
	15	16	17	18	19	20	21	22	23	24	25

حل: رکورد R8 که آدرس تولید شده برای آن 15 است، در اولین آدرس خالی یعنی 18 درج می‌شود و از R1 به R8 اشاره گری ایجاد می‌شود. سپس برای درج رکورد R9 که آدرس تولید شده برای آن 18 است، از خانه با آدرس 21 استفاده می‌شود و از حفره 18 به21 اشاره گری ایجاد می‌شود. در نهایت، رکورد R10 که آدرس تولید شده برای آن 15 است را در حفره آدرس 24 درج می‌کنیم و زنجیره ای که از آدرس 15 شروع و از 18 و 21 گذشته بود به 24 ختم می‌شود.

…..	R1	R2	R3	R8	R4	R5	R9	R6	R7	R10	….
	15	16	17	18	19	20	21	22	23	24	25



در واقع دو زنجیره یکی به مبدأ 15 و یکی به مبدأ 18 وجود دارد، که به یک زنجیره تبدیل شده‌اند. این مشکل را ائتلاف زنجیره‌ها (Coalesce) می‌گویند.

- پدیده ائتلاف باعث می‌شود که با افزایش فاکتور لود، طول زنجیره ها طولانی شده و کارایی فایل در درج و حذف کاهش می یابد.
- مشکل اساسی روش درج بدون جایگزین وقتی است که رکورد سر زنجیره حذف شود. در این حالت در هنگام بازیابی یک رکورد از زنجیره، پیام رکورد وجود ندارد اعلام خواهد شد. بنابراین سر زنجیره بهتر است حذف منطقی شود، یعنی نشانگر حذف شده برای آن درج شود و به عنوان حفره خالی اعلام نشود.

- زمان جستجو در روش ایجاد زنجیره بدون جایگزینی از روشهای قبلی کمتر است.

روش ایجاد زنجیره با جایگزینی

روش ایجاد زنجیره با جایگزینی، علاوه بر رفع مشکل حذف، زمان جستجو را باز هم کوتاهتر می کند. در این روش هنگام درج رکورد جدید، چنانچه آدرس طبیعی آن اشغال باشد، رکورد موجود در حفره مربوط به رکورد درج شدنی از حفره برداشته شده و رکورد جدید در حفره طبیعی خود جای می گیرد.

* با توجه به مثال قبل ، در صورت استفاده از روش ایجاد زنجیره با جایگزینی بعد از درج سه رکورد فایل به صورت زیر در خواهد آمد:

…..	R1	R2	R3	R9	R4	R5	R8	R6	R7	R10	….
	15	16	17	18	19	20	21	22	23	24	

ابتدا R8 در حفره 18 قرار می‌گیرد و پس از آن چون آدرس طبیعی R9 برابر 18 است ، R8 برداشته می‌شود و در اولین حفره خالی یعنی 21 قرار گرفته و R9 به جای آن درج می‌شود. در این حالت دو زنجیره داریم و پدیده ائتلاف زنجیره ها رخ نمی دهد و طول زنجیره ها کاهش می یابد .همچنین احتمال اینکه یک رکورد در آدرس طبیعی اش قرار بگیرد، بیشتر است.

موارد استفاده فایل مستقیم

۱- در محیط هایی که ماهیت پردازش ترتیبی نمی‌باشد و دستیابی سریع به رکوردها مورد نظر باشد.

۲- در محیط هایی که نرخ عملیات درج پایین باشد.

۳- در محیط هایی که طول رکوردها ثابت و کوچک باشند.

۴- در ایجاد ساختارهای ترکیبی(دسترسی به سرآیندها در فایل چند حلقه ای یا دسترسی به مدخلهای شاخص در سطح اول شاخص)

معایب فایل مستقیم مبنایی

۱- بروز حافظه هرز در فایل و توزیع نایکنواخت رکوردها در فضای آدرسی

۲- عدم تقارن (فقط یک صفت خاصه به عنوان کلید معرفی می‌شود.)

۳- محدودیت ثابت بودن طول رکوردها

۴- پدیده تصادف

۵- عدم امکان بازیابی رکورد بعدی به علت پایل بودن فایل

۶- عدم امکان پردازش سریال رکوردها

واکشی رکورد

عملیات لازم برای واکشی رکورد:

۱- اعمال تابع و یافتن آدرس (c_h)

۲- خواندن باکتی که آدرس آن بدست آمده است.

۳- بررسی محتوای باکت

۴- اگر رکورد در باکت نباشد، تصادفی است و باید آن را در رکوردهای تصادفی جستجو کرد.

بازیابی رکورد بعدی

چون فایل از نوع پایل است، رکورد بعدی هیچ مفهومی ندارد.بنابراین بازیابی رکورد بعدی مانند واکشی یک رکورد جدید است.(مانند ساختار پایل $T_N = T_F$)

بهنگام سازی

اگر مقدار کلید عوض نشود، بهنگام سازی به صورت درجا انجام می‌شود . برای اینکار رکورد واکشی شده و بعد از ایجاد نسخه جدید در بافر، رکورد بازنویسی می‌شود. البته اگر کلید رکورد عوض شود، نسخه قدیم با نشانگر حذف شده بازنویسی می‌شود و نسخه جدید درج می‌شود.

خواندن کل فایل

چون این فایل دارای حفره های خالی می‌باشد، در هنگام خواندن ترتیبی، باید همه آنها خوانده شوند.

در فایل با ساختار مستقیم زمان لازم برای بازیابی تمام رکوردها با استفاده از کلید nT_F می باشد.

سازماندهی مجدد

اگر زنجیره تصادفی طولانی شود یا ناحیه جداگانه سرریزی پر شود، فایل باید سازماندهی مجدد گردد. برای اینکار ابتدا کل فایل را باید خواند و سپس فایل را با رکوردهای فعال لود کرد. که برای لود کردن فایل مستقیم، رکوردها را یکی یکی در فضای آدرس می نویسیم :

فایل مستقیم از نظر احیاء نظم آغازین نیازی به سازماندهی مجدد ندارد.

روش های لود کردن فایل مستقیم

فایل مستقیم را به دو روش مستقیم و ترتیبی می توان لود کرد:

لود مستقیم

در این روش ، رکوردها از فایل ورودی خوانده شده و بعد از اعمال تابع مبدل روی مقدار کلید رکورد، رکورد در آدرس تولید شده درج می‌شود و در صورت پر بودن آدرس در فایل سرریز درج خواهد شد. سپس رکوردهای فایل سرریز در فایل مستقیم درج خواهند شد.

لود ترتیبی

در این روش ، رکوردهای فایل ورودی خوانده شده و آدرس تولید شده بعد از اعمال تابع مبدل روی کلید اصلی، در فیلد جدیدی از رکورد درج می‌شود و رکورد در یک فایل ترتیبی زمانی درج می‌شود. سپس رکوردها براساس آدرس ها مرتب شده و در یک فایل مستقیم دیگر نوشته می‌شوند و رکوردهای سرریزی نیز در یک فایل سرریزی وارد شده و در نهایت این رکوردهای سرریزی بر اساس یکی از تکنیکهای سرریزی در فایل مستقیم لود می شوند.

انتخاب تابع درهم ساز مناسب

طراح سیستم فایل باید از ضوابط انتخاب تابع توسط پردازشگر فایل مطلع باشد، تا بتواند بهترین تابع را انتخاب کند. این ضوابط عبارتند از :

۱- بتوان تابع را روی تمام ارقام کلید اعمال کرد.

۲- رکوردها به طور یکنواخت تری توزیع شوند.

۳- کمتر بودن تعداد تصادفی های لازم برای واکشی یک رکورد.

۴- کمتر بودن متوسط تعداد عملیات I /O لازم برای واکشی یک رکورد.

* کدام یک از دو تابع h_1, h_2 که بر روی ۷ رکورد موجود در یک فایل اعمال شده اند و آدرسهای زیر را تولید کرده اند، مناسب ترند؟ (تابع بر روی تمام ارقام کلید اعمال شده است)

h_1:1,2,3,4,5,1,1 , h_2:1,2,3,4,5,1,2

حل: تعداد تصادف ها در هر دو تابع یکسان است:

در تابع h_1، رکورد ششم با رکورد یکم و رکورد هفتم نیز با رکورد یکم تصادف کرده است و درتابع h_2 ، رکورد ششم با رکورد یکم و رکورد هفتم با رکورد دوم تصادف کرده است. به عبارتی تعداد تصادف ها در هر دو تابع برابر است و برای انتخاب تابع بهتر باید متوسط تعداد عملیات I/O لازم برای واکشی یک رکورد را محاسبه کرد.

$$h_1 : \frac{1+1+1+1+1+2+3}{7} = \frac{10}{7} = 1.4 \quad , \quad h_2 : \frac{1+1+1+1+1+2+2}{7} = \frac{9}{7} = 1.2$$

بنابراین تابع h_2 از h_1 مناسبتر است.■

باکت بندی

اگر فایل مستقیم با m حفره را باکت بندی کنیم، به جای آدرس حفره، آدرس باکت (از صفر تا M-1) حواهیم داشت. با فرض اینکه در هر باکت BK_f حفره وجود داشته باشد، داریم: $M = \frac{m}{BK_f}$

که برای ساختن M آدرس حداقل به $\log_2^M$ بیت نیاز است.

مزایای باکت بندی

۱- تسهیل در حل مشکل تصادف ۲- کوتاهتر شدن طول آدرسها

۳- امکان ایجاد فایل مستقیم با رکوردهای با طول متغیر

* اگر در حل مساله تصادف از روش باکت بندی استفاده نمائیم و رکوردهای تصادفی در یک آدرس باکت قرار گیرند و فرض کنیم تعداد حفره ها (m) برابر ۵۱۲ و تعداد حفره ها در باکت (B_{kf}) برابر ۳۲ باشد، در این صورت تعداد بیت های لازم برای آدرس دهی را بدست آورید؟ $M = \frac{512}{32} = 16$

بنابراین برای ساختن ۱۶ حفره به ۴ بیت نیاز داریم: $\log_2^{16} = 4$ ■

* استفاده از باکت بندی در مثال قبل، چند بیت طول آدرسها را کوتاهتر کرد؟

حل: اگر از باکت بندی استفاده نمی کردیم، تعداد بیت های مورد نیاز برابر بود با:

$\log_2^m = \log_2^{512} = 9$

و دیدیم که در صورت استفاده از باکت بندی، به ۴ بیت برای آدرس دهی نیاز داریم. بنابراین میزان ۵ بیت طول آدرسها کاهش یافت. ■

کاربردهای ساختارهای مطالعه شده

ساختار	موارد استفاده
پایل	۱- در محیطهایی که داده‌ها نظم پذیر نمی‌باشند و پیش پردازشی روی داده ها انجام نشده است و اساسا فایل برای بایگانی ایجاد شود. ۲- در محیطهایی که امنیت داده‌ها مورد نظر باشد. ۳- مبنایی برای درک و طراحی ساختارهای بهتر.
ترتیبی	۱- کاربردهای تجاری ۲- نیازی به تغییر طول رکورد نباشد. ۳- واکشی سریع تک رکوردها مورد نظر نباشد. ۴- کاربرد سیستمی ۵- در ایجاد بعضی از ساختارها
ترتیبی شاخص دار	۱- وقتی که پردازش سریال برحسب مقادیر کلید مطرح است(نه صفات دیگر آنها) ۲- وقتی که واکشی تک رکوردها از طریق کلید آنها، عملی رایجی می باشد. ۳- سیستم های داده پردازی تجاری ـ مدیریتی.
چند شاخصی	۱- محیط هایی که واکشی سریع تک رکوردها مطرح است. ۲- محیط هایی که فایل حالت نامانا دارد یعنی داده‌ها مرتباً در حال تغییر می‌باشند. ۳- محیط هایی که کاربر بخواهد از طریق صفات خاصه مختلف به رکوردها دستیابی داشته باشد.

مستقیم	۱- در محیط هایی که ماهیت پردازش ترتیبی نمی‌باشد و دستیابی سریع به رکوردها مورد نظر باشد. ۲- در محیط هایی که نرخ عملیات درج پایین باشد. ۳- در محیط هایی که طول رکوردها ثابت و کوچک باشند. ۴- در ایجاد ساختارهای ترکیبی

دسته‌بندی موضوعی آموزش‌های فرادرس، در ادامه آمده است:

			
برنامه نویسی – کلیک (+)	آموزش های دانشگاهی و تخصصی – کلیک (+)	هوش مصنوعی و یادگیری ماشین – کلیک (+)	مهندسی برق الکترونیک و روباتیک – کلیک (+)
			
دروس دانشگاهی – کلیک (+)	مباحث مشترک – کلیک (+)	مهارت های دانشگاهی – کلیک (+)	نرم افزارهای تخصصی – کلیک (+)
			
مهندسی نرم افزار – کلیک (+)	نرم افزارهای عمومی – کلیک (+)	طراحی و توسعه وب – کلیک (+)	آموزش های عمومی – کلیک (+)

منبع مطالعاتی تکمیلی مرتبط با این کتاب

آموزش ذخیره و بازیابی اطلاعات

با افزایش روز افزون اطلاعات، فرآیند ذخیره، بازیابی و استخراج اطلاعات از اهمیت ویژه‌ای برخوردار است. در درس ذخیره و بازیابی اطلاعات معماری روش ذخیره‌سازی، پیکربندی ورودی/خروجی، عملکرد دیسک و سامانه‌های ذخیره‌ساز، پیکربندی دیسک، تکنیک‌های ورودی/ خروجی رسانه‌ها، مفهوم سیستم فایل، شاخص‌بندی و درهم‌سازی مورد بحث قرار می‌گیرد.

آموزش ذخیره و بازیابی اطلاعات، توسط مهندس فرشید شیرافکن، یکی از بهترین مدرسین مسلط به این مباحث، ارائه شده است.

مدرس: مهندس فرشید شیر افکن

مدت زمان: ۲۰ ساعت

جهت مشاهده آموزش ویدئویی این آموزش – کلیک کنید **faradars.org /fvsft106**