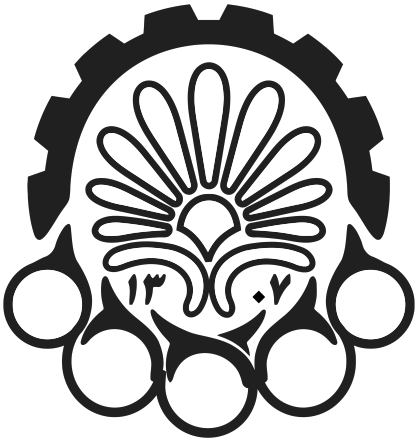
سیستم‌های عامل

دکتر زرندی



دانشگاه صنعتی امیرکبیر

(پلی تکنیک تهران)

دانشکده مهندسی کامپیوتر

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

تمرین سری نهم

۸ دی ۱۴۰۳

سیستم‌های عامل

تمرین سری نهم

رضا آدینه پور ۴۰۲۱۳۱۰۵۵

دانشکده مهندسی کامپیوتر

سوال اول

با فرض وجود سه قاب (frame) از الگوریتم‌های FIFO، LRU و بهینه (optimal) برای رشته‌های رجوع به صفحه (page) زیر با ذکر مراحل استفاده کرده (از چپ به راست) و در نهایت تعداد نقص صفحه (page fault) را به ازای هر الگوریتم به دست آورید.

- 3, 7, 3, 7, 6, 5, 6, 3, 3, 8, 7, 7, 9, 5, 6, 0, 2, 4, 3, 5
- 7, 6, 7, 5, 2, 3, 5, 7, 6, 6, 4, 3, 3, 2, 0, 8, 2, 7, 8, 7
- 5, 4, 6, 8, 3, 5, 2, 7, 1, 7, 8, 1, 7, 1, 2, 3, 6, 2, 8, 5

پاسخ

رشته اول: 3, 7, 3, 7, 6, 5, 6, 3, 3, 8, 7, 7, 9, 5, 6, 0, 2, 4, 3, 5

- FIFO: تعداد نقص صفحه: 10
- LRU: تعداد نقص صفحه: 9
- Optimal: تعداد نقص صفحه: 8

رشته دوم: 7, 6, 7, 5, 2, 3, 5, 7, 6, 6, 4, 3, 3, 2, 0, 8, 2, 7, 8, 7

- FIFO: تعداد نقص صفحه: 11
- LRU: تعداد نقص صفحه: 9
- Optimal: تعداد نقص صفحه: 7

رشته سوم: 5, 4, 6, 8, 3, 5, 2, 7, 1, 7, 8, 1, 7, 1, 2, 3, 6, 2, 8, 5

- FIFO: تعداد نقص صفحه: 12
- LRU: تعداد نقص صفحه: 10
- Optimal: تعداد نقص صفحه: 8

باتوجه به محاسبات انجام شده، الگوریتم Optimal کمترین تعداد نقص صفحه را دارد، زیرا آینده را پیش‌بینی می‌کند و صفحه‌ای را که دیرتر مورد نیاز است جایگزین می‌کند. الگوریتم LRU نیز عملکرد بهتری نسبت به FIFO دارد زیرا صفحات کمتر استفاده شده اخیر را جایگزین می‌کند.

سوال دوم

فرض کنید از صفحه‌آوری مبتنی بر درخواست (demand paging) استفاده می‌کنیم. جدول صفحات در حافظه اصلی نگهداری می‌شود که زمان دسترسی به آن ۱۱۰ نانوثانیه است. بنابراین ویژگی‌های حافظه ثانویه در این سیستم سرویس‌دهی به نقص صفحه در ۶۵ درصد مواقع ۴ میلی‌ثانیه و باقی مواقع ۲۱۰ میلی‌ثانیه طول می‌کشد. با این مفروضات بیشترین نرخ نقص صفحه چقدر می‌تواند باشد تا زمان مؤثر دسترسی بیشتر از ۲۰۰ نانوثانیه نشود؟

پاسخ

$$\text{EAT} = (1 - \text{p}) \times \text{memory access time} + \text{p} \times \text{page fault service time}$$

که در آن:

- p: نرخ نقص صفحه (page fault rate)
- memory access time = 110 نانوثانیه
- page fault service time: زمان متوسط سرویس‌دهی به نقص صفحه

محاسبه زمان متوسط سرویس‌دهی به نقص صفحه
زمان سرویس‌دهی به نقص صفحه به دو حالت بستگی دارد:

- در 65 درصد مواقع: 4 میلی‌ثانیه (4000000 نانوثانیه)
- در 35 درصد مواقع: 210 میلی‌ثانیه (210000000 نانوثانیه)

میانگین زمان سرویس‌دهی (average page fault service time) برابر است با:

$$\text{average service time} = 0.65 \times 4000000 + 0.35 \times 210000000$$

$$\text{average service time} = 2600000 + 73500000 = 76100000 \text{ nanoseconds}$$

جایگذاری در فرمول EAT
فرمول EAT را داریم:

$$200 = (1 - p) \times 110 + p \times 76100000$$

حل برای p:

$$200 = 110 - 110p + 76100000p$$

$$200 - 110 = 76100000p - 110p$$

$$90 = 76099990p$$

پاسخ

$$p = \frac{90}{76099990}$$

$$p \approx 0.00000118$$

بنابراین:
بیشترین نرخ نقص صفحه (p) که زمان مؤثر دسترسی (EAT) از 200 نانوثانیه بیشتر نشود، برابر است با:

$$p \approx 0.000118 \quad 0.0118\%.$$

سوال سوم

یک حافظه فیزیکی با ۱۰۲۴ قاب (frame) تحت نگاشت یک فضای آدرس‌دهی منطقی شامل ۲۰۴۸ صفحه که اندازه هر صفحه آن ۴ کیلوبایت می‌باشد، قرار گرفته است. برای آدرس‌دهی منطقی و آدرس‌دهی فیزیکی این فضا به چه تعداد بیت نیاز داریم؟

پاسخ

آدرس‌دهی منطقی (Logical Addressing)
فضای آدرس‌دهی منطقی شامل 2048 صفحه است و اندازه هر صفحه 4 KB (2^{12} بایت) است. بنابراین فضای آدرس‌دهی منطقی برابر است با:

$$2048 \times 2^{12} \text{ bytes} = 2^{11} \times 2^{12} = 2^{23} \text{ bytes}$$

برای آدرس‌دهی 2^{23} بایت، به 23 بیت نیاز داریم.

آدرس‌دهی فیزیکی (Physical Addressing)
حافظه فیزیکی شامل 1024 قاب (frame) است و اندازه هر قاب برابر با اندازه یک صفحه (4 KB) است. بنابراین فضای آدرس‌دهی فیزیکی برابر است با:

$$1024 \times 2^{12} \text{ bytes} = 2^{10} \times 2^{12} = 2^{22} \text{ bytes}$$

برای آدرس‌دهی 2^{22} بایت، به 22 بیت نیاز داریم.

سوال چهارم

با توجه به لیست درخواست شده (از چپ به راست) ترتیب دسترسی به فضاهای خواسته شده را با استفاده از الگوریتم‌های SCAN، C-SCAN، LOOK، C-LOOK و SSTF (Shortest Seek Time First) را بنویسید و همچنین مقادیر Head movement را به ازای هر الگوریتم نیز به دست آورید.
- مقدار اولیه سر (head) بر روی 50 است و بازه دیسک از 0 تا 199 است.

- 57, 140, 23, 98, 7, 102, 48, 52, 17, 12

پاسخ

۱. SCAN:
الگوریتم SCAN به این صورت عمل می‌کند که سر ابتدا در یک جهت حرکت می‌کند تا به انتهای دیسک برسد، سپس جهت خود را تغییر داده و به سمت دیگر حرکت می‌کند.
- ابتدا سر در موقعیت 50 است و به سمت 0 حرکت می‌کند. - در مسیر به سمت 0، درخواست‌ها را به ترتیب بررسی می‌کنیم: 48, 23, 17, 12, 7. - پس از رسیدن به 0، جهت سر تغییر کرده و به سمت 199 حرکت می‌کند. - در مسیر به سمت 199، درخواست‌های باقی‌مانده را به ترتیب بررسی می‌کنیم: 52, 57, 98, 102, 140.
ترتیب دسترسی: 48, 23, 17, 12, 7, 0, 52, 57, 98, 102, 140
مجموع Head movement:

$$|50-48|+|48-23|+|23-17|+|17-12|+|12-7| = 2+25+6+5+5 = 43$$

$$|7-0|+|0-52|+|52-57|+|57-98|+|98-102|+|102-140| = 7+52+5+41+4+38 = 147$$

$$\text{مجموع} = 43+147 = 190$$

۲. C-SCAN:
الگوریتم C-SCAN مشابه SCAN است، اما پس از رسیدن به انتهای دیسک، سر به طور مستقیم به ابتدای دیسک بازمی‌گردد.
- ابتدا سر در موقعیت 50 است و به سمت 0 حرکت می‌کند. - در مسیر به سمت 0، درخواست‌ها را به ترتیب بررسی می‌کنیم: 48, 23, 17, 12, 7. - پس از رسیدن به 0، سر به طور مستقیم به 199 بازمی‌گردد و سپس درخواست‌های باقی‌مانده را به ترتیب بررسی می‌کنیم: 52, 57, 98, 102, 140.
ترتیب دسترسی: 48, 23, 17, 12, 7, 0, 199, 52, 57, 98, 102, 140
مجموع Head movement:

$$|50-48|+|48-23|+|23-17|+|17-12|+|12-7| = 2+25+6+5+5 = 43$$

$$|7-0|+|0-199|+|199-52|+|52-57|+|57-98|+|98-102|+|102-140|$$
$$= 7+199+147+5+41+4+38 = 441$$

$$\text{مجموع} = 43+441 = 484$$

پاسخ

۳. LOOK:
الگوریتم LOOK مشابه SCAN است، با این تفاوت که سر پس از رسیدن به آخرین درخواست، جهت خود را تغییر می‌دهد و نیازی به حرکت به انتهای دیسک ندارد.
- ابتدا سر در موقعیت 50 است و به سمت 0 حرکت می‌کند. - در مسیر به سمت 0، درخواست‌ها را به ترتیب بررسی می‌کنیم: 48, 23, 17, 12, 7. - پس از رسیدن به 7، جهت سر تغییر کرده و به سمت 140 حرکت می‌کند. - در مسیر به سمت 140، درخواست‌های باقی‌مانده را به ترتیب بررسی می‌کنیم: 52, 57, 98, 102, 140.
ترتیب دسترسی: 48, 23, 17, 12, 7, 140, 102, 98, 57, 52
مجموع Head movement:

$$|50-48|+|48-23|+|23-17|+|17-12|+|12-7| = 2+25+6+5+5 = 43$$

$$|7-140|+|140-102|+|102-98|+|98-57|+|57-52| = 133+38+4+41+5 = 221$$

$$\text{مجموع} = 43+221 = 264$$

۴. C-LOOK:
الگوریتم C-LOOK مشابه LOOK است، اما پس از رسیدن به آخرین درخواست، سر به طور مستقیم به کوچکترین درخواست بازمی‌گردد.
- ابتدا سر در موقعیت 50 است و به سمت 0 حرکت می‌کند. - در مسیر به سمت 0، درخواست‌ها را به ترتیب بررسی می‌کنیم: 48, 23, 17, 12, 7. - پس از رسیدن به 7، سر به طور مستقیم به 140 بازمی‌گردد و سپس درخواست‌های باقی‌مانده را به ترتیب بررسی می‌کنیم: 52, 57, 98, 102, 140.
ترتیب دسترسی: 48, 23, 17, 12, 7, 140, 102, 98, 57, 52
مجموع Head movement:

$$|50-48|+|48-23|+|23-17|+|17-12|+|12-7| = 2+25+6+5+5 = 43$$

$$|7-140|+|140-102|+|102-98|+|98-57|+|57-52| = 133+38+4+41+5 = 221$$

$$\text{مجموع} = 43+221 = 264$$

۵. SSTF (Shortest Seek Time First):
الگوریتم SSTF به این صورت عمل می‌کند که سر همیشه به نزدیک‌ترین درخواست حرکت می‌کند.
- ابتدا سر در موقعیت 50 است. - نزدیک‌ترین درخواست به 50، 48 است. - سپس نزدیک‌ترین درخواست به 48، 52 است. - بعد از آن نزدیک‌ترین درخواست به 52، 57 است. - سپس نزدیک‌ترین درخواست به 57، 98 است. - و به همین ترتیب ادامه می‌دهیم.
ترتیب دسترسی: 48, 52, 57, 98, 102, 140, 23, 17, 12, 7
مجموع Head movement:

$$|50-48|+|48-52|+|52-57| = 2+4+5 = 11$$

$$|57-98|+|98-102|+|102-140| = 41+4+38 = 83$$

$$|140-23|+|23-17|+|17-12|+|12-7| = 117+6+5+5 = 133$$

$$\text{مجموع} = 11+83+133 = 227$$

سوال پنجم

در چه حالاتی (ترتیبی از درخواست‌ها) استفاده از الگوریتم C-SCAN بهتر از SCAN می‌باشد؟ با ذکر مثال دلیل آورید. توجه کنید منظور از بهتر بودن لزوماً کمتر بودن Head movement نمی‌باشد.

پاسخ

الگوریتم C-SCAN بهتر از SCAN در شرایطی است که حرکت سر دیسک (Head movement) به دلیل بازگشت به ابتدای دیسک در الگوریتم SCAN بیشتر می‌شود. در الگوریتم C-SCAN پس از رسیدن به انتهای دیسک، سر به طور مستقیم به ابتدای دیسک بازمی‌گردد و سپس به سمت انتهای دیگر حرکت می‌کند، در حالی که در الگوریتم SCAN سر پس از رسیدن به انتهای دیسک جهت خود را تغییر داده و دوباره به سمت دیگر حرکت می‌کند.
به عبارت دیگر، استفاده از C-SCAN زمانی مناسب است که تعداد درخواست‌ها به سمت یکی از انتهای دیسک متمرکز باشد و نیاز به بازگشت به ابتدای دیسک در SCAN موجب افزایش Head movement شود.
مثال:
فرض کنید درخواست‌ها به صورت زیر باشند:

$$7, 140, 23, 98, 17, 102, 48, 52, 57, 12$$

مقدار اولیه سر (head) روی 50 است و بازه دیسک از 0 تا 199 است.
در الگوریتم SCAN:

۱. ابتدا سر در موقعیت 50 است و به سمت 0 حرکت می‌کند.

۲. درخواست‌ها به ترتیب بررسی می‌شوند: 48, 23, 17, 12, 7.

۳. پس از رسیدن به 0، سر جهت خود را تغییر داده و به سمت 199 حرکت می‌کند.

۴. درخواست‌های باقی‌مانده بررسی می‌شوند: 52, 57, 98, 102, 140.

در الگوریتم C-SCAN:

۱. ابتدا سر در موقعیت 50 است و به سمت 0 حرکت می‌کند.

۲. درخواست‌ها به ترتیب بررسی می‌شوند: 48, 23, 17, 12, 7.

۳. پس از رسیدن به 0، سر به طور مستقیم به 199 بازمی‌گردد.

۴. درخواست‌های باقی‌مانده بررسی می‌شوند: 52, 57, 98, 102, 140.

در این مثال، استفاده از C-SCAN باعث می‌شود که سر دیسک پس از رسیدن به 0 به طور مستقیم به 199 بازگردد و نیازی به حرکت مجدد به سمت 0 نخواهد بود، در حالی که در SCAN سر پس از رسیدن به 0 باید دوباره به سمت 199 حرکت کند.
بنابراین، C-SCAN در این حالت می‌تواند سریع‌تر باشد زیرا نیازی به بازگشت به ابتدای دیسک نیست.