آزمایشگاه سیستم عامل

دستورکار ۱۲: الگوریتم‌های زمان‌بندی

بخش اول:

در حالت کلی، فرآیند یک برنامه در حال اجرا است که وارد سیستم شده، روی آن پردازش انجام می‌شود و درنهایت از سیستم خارج می‌گردد اما یک فرآیند بلافاصله بعد از ورود نمی‌تواند CPU را در اختیار گرفته و عملیات پردازش روی آن صورت گیرد. الگوریتم‌های زمان‌بندی فرآیندها را بر روی پردازنده به شیوه‌ای کارآمد و مؤثر برنامه‌ریزی می‌کنند. این زمان‌بندی توسط یک زمان‌بند فرآیند انجام می‌شود که با افزایش توان عملیاتی، استفاده از CPU را به حداکثر می‌رساند.

به‌عبارت‌دیگر الگوریتم‌های زمان‌بندی پردازنده، مدیریت فرآیندها در سیستم‌عامل را بر عهده‌دارند و برای توزیع منابع بین طرف‌هایی که به‌طور همزمان و غیر هم‌زمان آنها را درخواست می‌کنند، استفاده می‌شوند.

اهداف الگوریتم‌های زمان‌بندی در سیستم‌عامل

- حداکثر استفاده از CPU:CPU را تا حد امکان مشغول نگه می‌دارند.
- تخصیص منصفانه :CPU اختصاص عادلانه CPU به فرآیندها را انجام می‌دهد.
- حداکثر توان عملیاتی: تعداد فرآیندهایی که اجرای خود را در واحد زمان کامل می‌کنند.
- حداقل زمان بازگشت: زمان صرف شده توسط یک فرآیند برای پایان اجرا می‌باشد.
- حداقل زمان انتظار: زمان انتظار یک فرآیند در صف آماده است.
- حداقل زمان پاسخ: زمانی که یک فرآیند اولین پاسخ را تولید می‌کند.

انواع الگوریتم‌های زمان‌بندی در سیستم‌عامل

الگوریتم‌های زمان‌بندی پردازنده در مدیریت فرآیندهای سیستم‌عامل به دو گروه اصلی تقسیم می‌شوند.

۱. الگوریتم‌های انحصاری
۲. الگوریتم‌های غیر انحصاری

الگوریتم‌های انحصاری(Non Preemptive)

در این الگوریتم‌ها به‌محض اینکه یک فرآیند پردازنده را در اختیار گرفت و شروع به اجراشدن کرد، تا زمانی که فرآیند به‌طور کامل به‌پایان نرسد و یا مسدود نشود پردازنده را در اختیار فرآیند دیگری قرار نمی‌دهد. به‌عبارت‌دیگر در این نوع از الگوریتم‌ها فرآیندها یکجا و در یک‌بار اجراشده و به قسمت‌های کوچک‌تر تقسیم نمی‌شوند. الگوریتم‌های FCFS, SJN از این نوع هستند.

الگوریتم‌های غیر انحصاری(Preemptive)

در این الگوریتم‌ها یک فرآیند که در حال اجراست ممکن است توسط سیستم‌عامل متوقف‌شده و به حالت آماده منتقل شود. این کار برای اختصاص پردازنده به یک فرآیند دیگر و یا انجام عملیات I/O و وقفه انجام می‌شود. به‌عبارت‌دیگر در این نوع از الگوریتم‌ها، فرآیندها ممکن است به چند بخش تقسیم‌شده و در چند مرحله عملیات تخصیص پردازنده انجام‌گرفته و اجرا شود. الگوریتمRR از این نوع است.

بخش دوم: الگوریتم‌های زمان‌بندی مختلف

الگوریتم First-Come, First-Served (FCFS)

این الگوریتم ساده‌ترین و آسان‌ترین نوع الگوریتم‌های زمان‌بندی در سیستم‌عامل است. در این الگوریتم، فرآیندی که ابتدا CPU را درخواست می‌کند،متقابلاً زودتر CPU را برای اجرا در اختیار می‌گیرد. این الگوریتم را می‌توان آن را با استفاده از روش FIFO (First-In, First-Out) Queue)پیاده‌سازی کرد. ویژگی‌های این الگوریتم عبارتند از:

۱- اجرای آن آسان است.
۲- CPUهمیشه بر اساس First-come و First-serve تخصیص داده می‌شود.
۳- اما، این الگوریتم عملکرد ضعیفی دارد زیرا میانگین زمان انتظار آن بسیار بالاست.

الگوریتم Shortest-Job-Next (SJN)

دومین مورد از الگوریتم‌های زمان‌بندی در سیستم عامل SJN الگوریتمی است که در آن فرآیندی که کمترین زمان اجرا را دارد برای اجرای بعدی انتخاب می‌شود. این الگوریتم زمان‌بندی می‌تواند به‌طور قابل‌توجهی میانگین زمان انتظار برای سایر فرآیندها در صف انتظار اجرا را کاهش دهد.

در این الگوریتم هر کار با یک واحد زمان برای تکمیل همراه است و برای پردازش دسته‌ای مفید است، جایی که انتظار برای تکمیل کارها حیاتی نیست. می‌تواند با اطمینان از اینکه کارهای کوتاه‌تر در ابتدا اجرا می‌شوند، عملکرد فرآیند را بهبود ببخشد، بنابراین احتمالاً زمان کوتاه‌تری را صرف می‌کند. با ارائه فرآیندهای کوتاه‌تر، که باید ابتدا اجرا شوند و عمدتاً زمان چرخش کوتاه‌تری دارند، خروجی کار را بهبود می‌بخشد.

الگوریتم الویت‌دار Priority Scheduling

لگوریتم زمان‌بندی اولویت، یک الگوریتم غیر حریصانه و یکی از رایج‌ترین الگوریتم‌های زمان‌بندی در سیستم‌های دسته‌ای است. به هر فرآیند در لحظه ورود اولویتی اختصاص داده می‌شود (زمان رسیدن زودتر برابر با اولویت بالاتر است.) اگر دو فرآیند زمان رسیدن

یکسانی داشته باشند، با اولویت‌ها مقایسه می‌شوند (ابتدا فرآیندی با اولویت بالاتر). همچنین، اگر دو فرآیند دارای اولویت یکسانی هستند، آن را با شماره پردازش مقایسه می‌کنند. (اول تعداد فرآیند کمتر). این فرآیند درحالی‌که تمام فرآیندها اجرا می‌شوند تکرار می‌گردد.

الگوریتم Round Robin(RR)

زمان‌بند یک واحد زمان ثابت را به هر فرآیند اختصاص می‌دهد و آن‌ها را طی می‌کند. اگر فرآیند در آن بازه زمانی کامل شود، خاتمه می‌یابد، در غیر این صورت، پس از دادن فرصت به سایر فرآیندها، مجدداً زمان‌بندی می‌شود. برنامه‌ریزی RR شامل سربار گسترده‌ای است، به‌خصوص اگر یک واحد زمانی کوچک در نظر گرفته شود. به دلیل زمان انتظار بالا، ضرب‌الاجل‌ها به‌ندرت در یک سیستم RRخالص رعایت می‌شود.

گرسنگی هرگز نمی‌تواند رخ دهد، زیرا هیچ اولویتی داده نمی‌شود. ترتیب تخصیص واحد زمان‌بر اساس زمان رسیدن فرآیند، مشابه FIFOاست. اگر Time-Slice بزرگ باشد به FCFS /FIFO یا اگر کوتاه باشد تبدیل به SJF می‌شود.

تمرین

- برنامه‌ای به زبان C بنویسید که الگوریتم FCFS را پیاده‌سازی کند. برای این کار مراحل زیر را طی کنید:
  ۱- تعداد فرآیندها را از کاربر دریافت کنید.
  ۲- زمان سرویس‌دهی هر فرآیند را از کاربر دریافت کنید.
  ۳- زمان انتظار برای فرآیند اول را با صفر مقداردهی کنید.
  ۴- زمان انتظار سایر فرآیندها را مشخص کنید. (دقت کنید که زمان انتظار یک فرآیند برابر با زمان اجرای فرآیند قبل است. زمان اجرای یک فرآیند برابر است با مجموع زمان انتظار و زمان سرویس‌دهی).
  ۵- زمان اجرای فرآیندها را حساب کنید.
  ۶- متوسط زمان انتظار و زمان اجرا برای هر فرآیند را حساب کنید و نمایش دهید.

راهنمایی:

برای فرآیند می‌توانید از ساختاری مشابه ساختار زیر استفاده کنید:

```
struct process
{
int pid;
int st, wt, tt;    //service time, waiting time, total time
}p[10];
```

- برنامه‌ای به زبان C بنویسید که الگوریتم SJN را پیاده‌سازی کند. برای این کار مراحل زیر را طی کنید:
  ۱- تعداد فرآیندها را از کاربر دریافت کنید.
  ۲- زمان سرویس‌دهی هر فرآیند را از کاربر دریافت کنید.
  ۳- زمان انتظار برای فرآیند اول را با صفر مقداردهی کنید.
  ۴- زمان انتظار سایر فرآیندها را مشخص کنید. (دقت کنید که زمان انتظار یک فرآیند برابر با زمان اجرای فرآیند قبل است. زمان اجرای یک فرآیند برابر است با مجموع زمان انتظار و زمان سرویس‌دهی).

۵- زمان اجرای فرآیندها را حساب کنید.
۶- متوسط زمان انتظار و زمان اجرا برای هر فرآیند را حساب کنید و نمایش دهید.

- برنامه‌ای به زبان C بنویسید که الگوریتم زمان‌بندی الویت‌دار را پیاده‌سازی کند. برای این کار مراحل زیر را طی کنید:
۱- تعداد فرآیندها را از کاربر دریافت کنید.
۲- زمان سرویس‌دهی و درجه اهمیت هر فرآیند را از کاربر دریافت کنید.
۳- زمان انتظار برای فرآیند اول را با صفر مقداردهی کنید.
۴- فرآیندها را بر اساس درجه اهمیت مرتب کنید.
۵- زمان انتظار سایر فرآیندها را مشخص کنید. (دقت کنید که زمان انتظار یک فرآیند برابر با زمان اجرای فرآیند قبل است. زمان اجرای یک فرآیند برابر است با مجموع زمان انتظار و زمان سرویس‌دهی).
۶- زمان اجرای فرآیندها را حساب کنید.
۷- متوسط زمان انتظار و زمان اجرا برای هر فرآیند را حساب کنید و نمایش دهید.

- برنامه‌ای به زبان C بنویسید که الگوریتم RR را پیاده‌سازی کند. برای این کار مراحل زیر را طی کنید:
۱- تعداد فرآیندها را از کاربر دریافت کنید.
۲- زمان سرویس‌دهی هر فرآیند را از کاربر دریافت کنید.
۳- کوانتوم زمانی را از کاربر دریافت کنید.
۴- زمان انتظار برای فرآیند اول را با صفر مقداردهی کنید.
۵- ترتیب انجام فرآیندها را نشان دهید.
۶- متوسط زمان انتظار هر فرآیند را حساب کنید و نمایش دهید.