

سیستم عامل جلسه چهارم

الگوریتم های زمانبندی

زمانبندی پردازنده با این مسأله سر و کار دارد که پردازنده باید به کدام فرایند موجود در صف آماده تخصیص یابد.

زمانبندی ارایه خدمت به ترتیب ورود (First Come First Served)

ساده ترین الگوریتم زمانبندی پردازنده، الگوریتم خدمت به ترتیب ورود (FCFS) نام دارد. در این الگوریتم فرایندی که زودتر پردازنده را درخواست کرده، زودتر آن را در اختیار می گیرد. پیاده سازی سیاست

FCFS با یک صف (FIFO - First in First Out) انجام می شود.

وقتی فرایندی وارد صف آماده می شود، (PCB (Process control block آن در انتهای صف قرار می گیرد. وقتی پردازنده آزاد شد؛ به فرایند موجود در ابتدای صف تخصیص می یابد. سپس PCB فرایند در حال اجرا، از صف حذف می شود. نوشتن و درک برنامه ی زمانبندی FCFS ساده است

FCFS (Example)

Process	Duration	Oder	Arrival Time
P1	24	1	0
P2	3	2	0
P3	4	3	0

Gantt Chart :



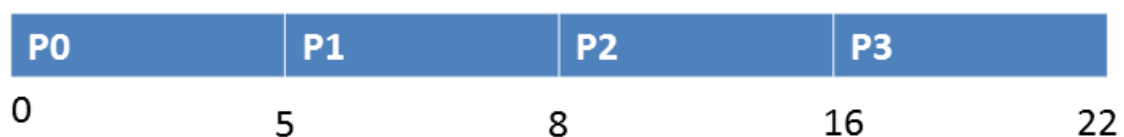
P1 waiting time : 0
P2 waiting time : 24
P3 waiting time : 27

The Average waiting time :
 $(0+24+27)/3 = 17$

At what time the process is completed *Completion time-Arrival time*

Process	Arrival time	Burst time	Comple tion time	Turn around time
P0	0	5	5	5
P1	1	3	8	7
P2	2	8	16	14
P3	3	9	22	19

Criteria: Burst time
Mode: Non preemption



* هر فرایند در سیستم عامل به وسیله ی بلوک کنترل فرایند (PCB) نمایش داده می شود. نام دیگر PCB بلوک کنترل وظیفه است

میانگین زمان انتظار تحت سیاست FCFS بسیار زیاد است. مجموعه فرایندهای زیر را به همراه انفجار پردازنده بر حسب میلی ثانیه، در نظر بگیرید

فرآیند	زمان انفجار (زمان اجرا)
P_1	24
P_2	3
P_3	3

اگر هر فرایند به ترتیب p_1, p_2, p_3 بیایند، و به ترتیب FCFS خدمات بگیرند. نتیجه در نمودار گانت (GANTT) زیر مشخص شده است. نمودار گانت، یک نمودار میله ای است که زمان بندی خاص را نشان می دهد. از جمله زمان های شروع و پایان هر فرایند



زمان انتظار برای P_1 برابر با صفر میلی ثانیه، برای P_2 برابر با 24 میلی ثانیه و برای P_3 برابر با 27 میلی ثانیه است

بنابراین میانگین زمان انتظار برابر با $(27+24+0)/3 = 17$ میلی ثانیه است.

اگر فرایندها به ترتیب p_1, p_3, p_2 باشد، نتیجه نمودار گانت به صورت زیر است:



اکنون میانگین زمان انتظار برابر با

$$3 = \frac{0+3+6}{3}$$

میلی ثانیه است، این کاهش زمان قابل توجه است. بنابراین میانگین زمان انتظار تحت سیاست FCFS کمینه (حداقل) نیست و ممکن است با تغییرات زیادی که در انفجار پردازنده‌ی فرایندها ایجاد می‌شود، تغییر کند

فرض می‌کنیم یک فرایند در تنگنای پردازنده و چند فرایند در تنگنای I/O داریم. وقتی فرایندها در صف‌های سیستم جا به جا می‌شوند، این وضعیت ممکن است پیش بیاید:

فرایند در تنگنای پردازنده، پردازنده را در اختیار می‌گیرد و آن را نگه می‌دارد. در این زمان، تمام فرایندهای دیگر، I/O خود را تمام خواهند کرد و به صف آماده می‌روند و منتظر پردازنده می‌مانند. در حالی که فرایندها در صف آماده منتظر هستند. دستگاه‌های I/O بی‌کار می‌مانند. سرانجام فرایند در تنگنای پردازنده، انفجار پردازنده‌ی خود را به اتمام می‌رساند و به دستگاه I/O می‌رود. تمام فرایندهای در تنگنای I/O که در زمان انفجار پردازنده‌ی آن‌ها کم است، سریعاً اجرا می‌شوند، و به صف‌های I/O می‌گردند. در این نقطه، پردازنده بی‌کار می‌ماند. سپس فرایند در تنگنای پردازنده به صف آماده می‌رود و پردازنده به آن تخصیص می‌یابد. دوباره تمام فرایندهای I/O به صف آماده می‌روند تا فرایند در تنگنای پردازنده اجرا شوند. در اینجا یک اثر اسکورت

(Convoy Effect) وجود دارد. به طوری که تمام فرایندها منتظر هستند. تا یک فرایند بزرگ، پردازنده را رها کند. این اثر نسبت به روشی که ابتدا به تمام فرایندهای کوتاه تر سرویس می دهد، منجر به بهره وری اندک پردازنده و دستگاه ها می شود

الگوریتم زمانبندی FCFS بدون قبضه کردن است. وقتی پردازنده به فرایندی تخصیص یافت، آن را در اختیار می گیرد تا آن را رها کند. رها کردن پردازنده ممکن است در اثر خاتمه یافتن فرایند یا درخواست i/o صورت بگیرد. الگوریتم FCFS برای سیستم عامل های اشتراک زمانی (Time Sharing) مشکل زا است. زیرا در این سیستم ها کاربر پردازنده را در فواصل زمانی منظمی به دست می گیرد. این که یک فرایند اجازه داشته باشد پردازنده را به مدت زیادی در اختیار داشته باشد، برای کارایی سیستم خطرناک است.

زمانبندی بر حسب کوتاه ترین کار SJF (Shortest Job First)

الگوریتم SJF به هر فرایند، طول انفجار پردازنده ی بعدی اش را نسبت می دهد، وقتی پردازنده مهیا باشد، به فرایندی نسبت داده می شود. که انفجار پردازنده ی بعدی کوچک تری داشته باشد

اگر طول انفجار پردازنده ی بعدی در فرایند یکسان باشد. برای انتخاب یکی از آن ها، از زمان بعدی FCFS استفاده می شود

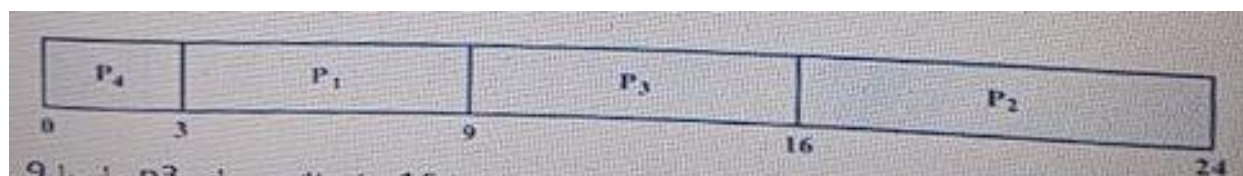
توجه کنید که بهتر است نام این الگوریتم را کوتاه ترین انفجار پردازنده ی بعدی بنامیم. زیرا زمانبندی با بررسی طول انفجار پردازنده ی بعدی فرایند انجام می شود (نه طول کلی آن).

فرایند های زیر را به همراه طول زمان انفجار که بر حسب میلی ثانیه بیان شده است در نظر بگیرید

فرایند	زمان انفجار (زمان اجرا)
P_1	6
P_2	8
P_3	7
P_4	3

با استفاده از زمانبندی SJF این فرایند ها بر اساس نودار زیر زمانبندی می شوند:

P4	P1	P3	P2
----	----	----	----



زمان انتظار برای فرایند p_1 برابر با 3 میلی ثانیه است، برای p_2 برابر با 16 میلی ثانیه و برای p_3 برابر با 9 میلی ثانیه و برای p_4 برابر با 0 میلی ثانیه است. بنابراین میانگین زمان انتظار برابر با

$$(3+9+16+0) / 4 = 7$$

میلی ثانیه است. اگر از الگوی زمانبندی FCFS استفاده می کردیم. میانگین زمان انتظار برابر با 10.25 میلی ثانیه بود

الگوریتم زمانبندی SJF احتمالا بهینه است. زیرا برای این مجموعه از فرایندها میانگین زمان انتظار آن کمینه است. با انتقال فرایند کوتاه به قبل از فرایند بلند زمان انتظار مربوط به فرایند کوتاه، پیش از زمان انتظار مربوط به فرایند بلند، کاهش می یابد. در نتیجه میانگین زمان انتظار کاهش پیدا می کند

مشکل عمده الگوریتم SJF این است که طول درخواست بعدی پردازنده باید

مشخص باشد. برای زمانبند بلند مدت (زمان بند کار) در یک سیستم دسته ای (batch)، می توان حد زمانی را که کاربر هنگام تحویل کار تعیین کرده است به عنوان طول فرایند در نظر گرفت. بنابراین، کاربران سعی می کنند حد زمانی فرایند را دقیقاً برآورد کنند؛ زیرا اگر مقدار حد زمانی کمتر برآورد شود؛ به معنای دریافت پاسخ سریع تر است (اگر خیلی کم باشد موجب بروز خطای حد زمانی می شود که مستلزم تحویل دوباره است)

زمانبندی SJF در زمانبندی بلند مدت کاربر زیادی دارد

گرچه الگوریتم SJF بهینه است اما نمی تواند در سطح زمانبندی کوتاه مدت پردازنده به کار گرفته شود. راهی وجود ندارد که از انفجار بعدی پردازنده آگاهی پیدا کنیم. می توانیم زمانبندی SJF را تخمین بزنیم. ممکن است طول انفجار بعدی پردازنده را ندانیم. اما می توانیم اندازه اش را پیش بینی کنیم. انتظار داریم که طول انفجار بعدی پردازنده، مشابه قبلی باشد.

بنابراین با تخمین طول انفجار بعدی پردازنده می توانیم فرایندی را انتخاب کنیم که طول انفجار بعدی پردازنده ی کوتاه تر است/

الگوریتم SJF ممکن است با قبضه کردن یا بدون قبضه کردن باشد

اگر فرایندی در حال اجرا باشد و فرایند جدیدی به صف آماده وارد شود، یکی از این دو فرایند باید انتخاب شود. اگر زمان انفجار بعدی پردازنده ی فرایند جدید ، کمتر از انفجار باقی مانده ی پردازنده در فرایند فعلی باشد در این صورت الگوریتم "با قبضه کردن Preemptive". از اجرای فرایند در حال اجرا جلوگیری میکند و فرایند جدید اجرا می شود. در حالی که الگوریتم "بدون قبضه کردن Nonpreemptive" اجازه می دهد که فرایند در حال اجرا به اجرای ادامه دهد تا انفجار پردازنده آن به اتمام برسد.

زمانبندی SJF با قبضه کردن را گاهی خدمات به کوتاه ترین زمان باقی مانده

(Shortest-remaining-time-first) می نامند