سیستم عامل جلسه پنجم

ادامه الگوریتم های زمانبدی

زمانبندی با اولویت (Priority)

الگوریتم SJF در حالت خاصـــی از الگوریتم زمانبندی با اولویت اســـت. به هر فرایند یک اولویت نسبت داده می شود و پردازنده به فرایندی تخصیص می یابد که بالاترین اولویت را دارای باشد

فرایند هایی با اولویت یکسان، به ترتیب FCFS زمانبندی می شوند.

الگوریتم SJF یک الگوریتم با اولویت ساده ا ست که در آن ، اولویت (P) ، معکوس انفجار تخمین زده شـــده ی بعدی پردازنده اســـت. هر چه زمان انفجار پردازنده بیشـــتر باشـــد، اولویت کمتر است و برعکس.

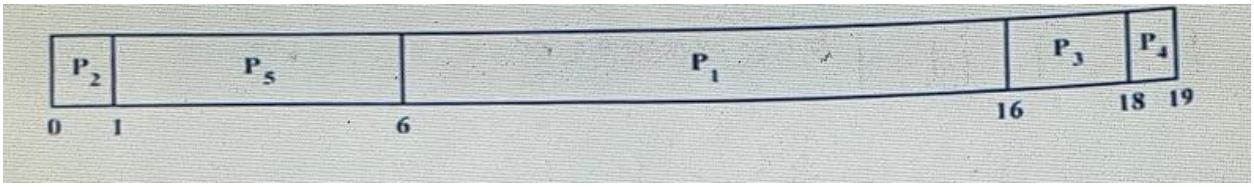
بعضی از افراد برای اولویت های پایین از اعداد کوچک استفاده می کنند ولی بعضی دیگر برای اولویت پایین از اعداد بزرگ استفاده می کنند. این کار موجب سردرگمی است.

فرض می کنیم که اعداد کوچک، اولویت بالا را نشان می دهد

فرایند های زیر را در نظر بگیرید. به طوری که در زمان 0 به ترتیب p5 , ... ,p1,p2,p3 رسیده اند و طول انفجارپردازنده بر حسب میلی ثانیه است:

فرآیند	زمان انفجار (زمان اجرا)	اولویت
P_1	10	3
P_2	1	1
P_3	2	4
P_4	1	5
P_5	5	2

با اســـتفاده از زمانبندی با اولویت، این فرایند ها بر اســـاس نمودار گانت (GANTT) زیر رسم می شوند:



میانگین زمان انتظار در این مثال، 8.2 میلی ثانیه است.

اولویت می تواند به طور داخلی یا خارجی تعریف شـــود. اولویت هایی که به طور داخلی تعریف می شوند، اولویت یک فرایند را با استفاده از کمیت های قابل اندازه گیری تعریف می کنند.

به عنوان مثال حدود زمانی ، نیازمندی های حافظه، تعداد فایل های باز و نسبت میانگین انفجار i/o به میانگین انفجار پردازنده، در محاسبه اولویت های داخلی به کار می آیند.

اولویت های خارجی بر اســـاس معیار هایی تعیین می شـــوند که از نظر ســـیســـتم عامل، خارجی هستند، مثل اهمیت فرایند، نوع و میزان هزینه ای که برای استفاده از کامپیوتر پرداخته شده، میزان پشتیبانی موسسه از کار و سایر عوامل سیاست گذاری.

زمانبندی با اولویت می تواند با قبضه کردن یا بدون قبضه کردن باشد. وقتی فرایندی به صف آماده می رسد، اولویت آن با اولویت فرایند در حال اجرا مقایسه می شود. اگر فرایندی که تازه وارد صف شده بیشتر از اولویت فرایندی با شد که در حال اجرا ا ست، الگوریتم زمانبندی با قبضه کردن (Preemptive)، پردازنده را در اختیار فرایند جدید قرار می دهد. اما در الگوریتم زمانبندی بدون قبضه کردن (Non Preemptive)، فرایند جدید بدون توجه به اولویتش درابتدای صف آماده قرار می گیرد

مساله ی عمده در الگوریتم زمانبندی با اولویت، انسداد (indefinite Blocking) یا گرسنگی (قحطی) (Starvation) است.

Starvation or indefinite blocking is a phenomenon associated with the Priority scheduling algorithms. A process that is present in the ready state and has low priority keeps waiting for the CPU allocation because some other process with higher priority comes with due respect time.

فرایندی که آماده ی اجرا است ولی منتظر پردازنده باشد، مسدود در نظر گرفته می شود. الگوریتم زمانبندیِ با اولویت می تواند و منجر به این شود که فرایند هایی با اولویت پایین، به مدت نامحدودی منتظر پردازنده با شند. در یک سیستم کامپیوتری با بار زیاد، فرایند هایی با اولویت بالا، مانع از این می شوند که پردازنده به فرایند هایی با اولویت پایین تعلق یابد. معمولا یا سر انجام ، فرایند با اولویت پایین اجرا می شود، یا سیستم کامپیوتری فرو می پاشد و همه فرایند های با اولویت پایین که تمام نشده اند مفقود میشوند

راه حل این م ساله ی از سداد نامحدودِ فرایند های با اولویت پایین، سالمندی (Aging) است.

در این تکنیک اولویت فرایندی که مدت زیادی در سیستم منتظر مانده است، به تدریج افزایش می یابد.

اگر اولویت با مقادیری از 0 (اولویت بالا) تا 127 (اولویت پایین) مشخص شود، می توان هر 15 دقیقه، یک واحد به اولویت یک فرایند اضافه کنیم. سرانجام، حتی فرایندی که اولویت اولیه آن 127 است. اولویت بالایی در سیستم کسب می کند و می تواند اجرا شود. در واقع، برای اینکه فرایندی با اولویت 127 سالمند شود و اولویت 0 را بدست آورد بیش از 32 ساعت طول نمی کشد.

یادداشت:

P = 15 minute - period of each increment

X = 127 process index number

T = 127 total of indexes

((T-1)*15)/60

((127-1)*15)/60

= 31.5 h

زمانبندی نوبت گردشی "RR" (Round Robin)

الگوریتم نوبت گردشی RR مخصوص سیستم های اشتراک زمانی طراحی شده است. این الگوریتم شبیه FCFS ا ست، با این تفاوت که در جابه جایی بین فرایند ها، از زمان بندی با قبضه کردن (Preemptive) استفاده می شود. یک واحد زمانی کوچک، به نام کوانتوم (quantum) زمانی یا برهه ی زمانی (برش زمانی) تعریف می شود. کوانتوم زمانی معمولا 10 – 100 میلی ثانیه ا ست. صف آماده به صورت یک صف چرخ شی در نظر گرفته می شود. زمانبند پردازنده در طول صف آماده جا به جا می شود و پردازنده را حداکثر به مدت یک کوانتوم زمانی به هر فرایند تخصیص می دهد.

برای پ یاده ســـازی ز مانب ندی RR ، صـــف آ ماده را به صـــورت یک صـــف FIFO (First in First Out) از فرایندها در نظر می گیریم، فرایند های جدید به انتهای صـــف آماده اضافه می شود. زمانبندی پردازنده، اولین فرایند را از صف آماده انتخاب می کند و تایمر را طوری تنظیم می کند که پس از یک کوانتوم زمانی وقفه ای صادر شود و فرایند را روی پردازنده توزیع می کند.

دو حالت وجود دارد:

1. پردازنده کمتر از یک کوانتوم زمانی به فرایند اختصـــاص یابد. در این حالت، خود فرایند پردازنده را آزاد می کند و بدین ترتیب، پردازنده به فرایند بعدیِ موجود در صف آماده تخصیص می یابد.
2. اگر پردازنده بخواهد بیش از یک کوانتوم زمانی به فرایند در حال اجرا اختصـــاص یابد. تایمر خاموش می شود و وقفه ای را به سیـ ستم عامل می فر ستند. تعویض متن (**Context switch**) صورت می گیرد و فرایند به انهای صف آماده اضافه می شود سپس زمانبندی پردازنده ، فرایند بعدی را از صف آماده انتخاب می کند

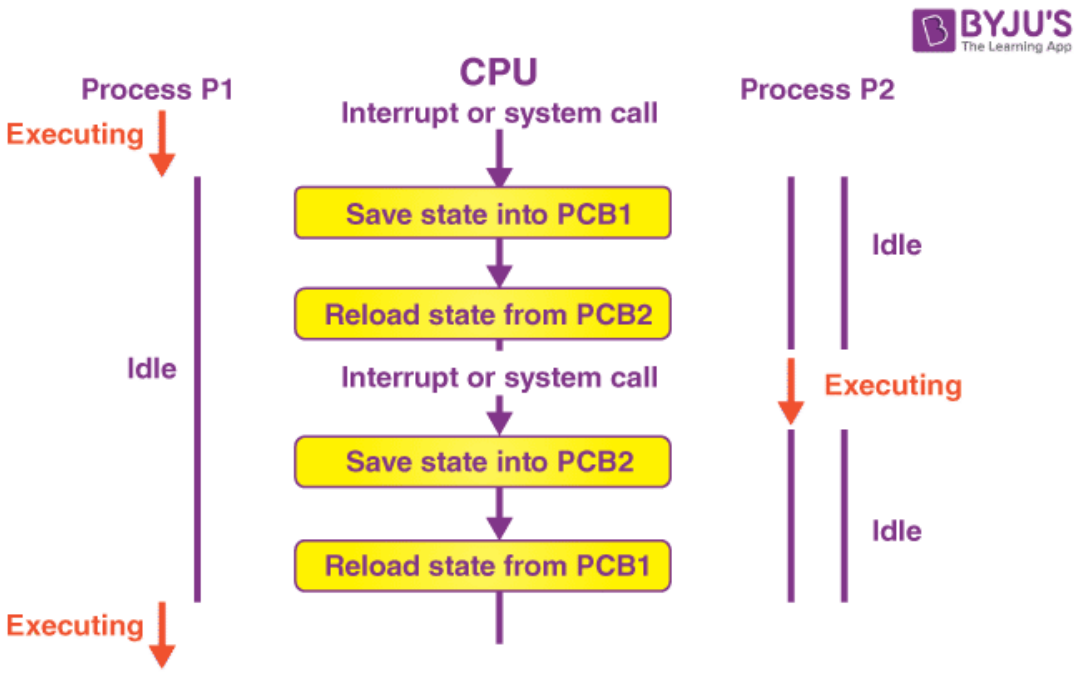
تعویض متن (تعویض بستر یا تعویض زمینه)(Context Switch):

وقفه (**interrupt**) موجب می شود سیـ ستم عامل، پردازنده را از اجرای وظیفه ی فعلی به اجرای روال هســته ببرد. چنین عملیاتی غالبا در ســیســتم های همه منظوره رخ می دهد. وقتی وقفه ای رخ میدهد. لازم اســت ســیســتم، متن (وضــعیت – **status**) فعلی فرایند را که در پردازنده در حال اجرا است، ذخیره کند، به طوری که پس از پردازش، آن متن را بازیابی می کندکه موجب به تعویق افتادن فرایند و ســپس از ســرگیری آن می شود. متن، در **PCB**(**Process control Block**) مربوط به فرایند ذخیره می شود. متن شامل ثبات های پردازنده ، حالت فرایند و اطلاعات مدیریت حافظه است.

PCB شامل و ضعیت کامل هر فرایند می با شد از ریجـ ستر ها تا اطلاعات پایه هر فرایند در واحد کنترل وظیفه (**PCB**) آن نوشته می شود

تعویض پردازنده به فرایند دیگر، نیازمند اجرای ذخیره ی حالت فعلی و بازیابی حالت فرایند دیگر است.

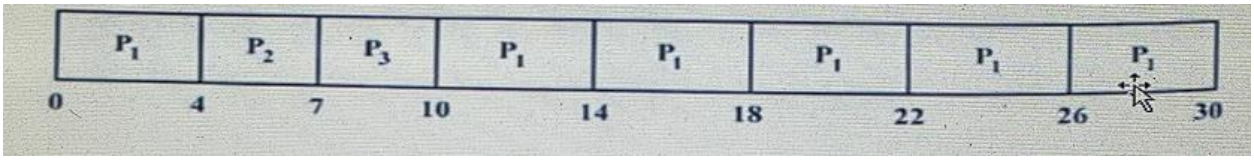
این کار ، تعویض متن یا تعویض بستر یا (Context Switch) نام دارد. وقتی تعویض متن صـــورت می گیرد، هســـته متن فرایند قبلی را در PCB (Process Control Block) آن ذخیره می کند و متن ذخیره شده ی فرایند جدید را که برای اجرا زمانبندی شد، بار می کند. زمان تعویض متن، سرباز محض است، زیرا هنگام تعویض، سیستم، کاری انجام نمی دهد. سرعت آن از ما شینی به ما شین دیگر فرق می کند که به سرعت حافظه، تعداد ثبات هایی که باید کپی شوند و وجود د ستور العمل خاص (مثل تنها یک د ستورالعمل برای باز کردن با ذخیره ی تمام ثبات ها) بســـتگی دارد. ســـرعت عای متدوال، چندین میلی ثانیه هستند.



میانگین زمان انتظار در الگوریتم RR ، اغلب زیاد است. فرایند های زیر را در نظر بگیرید که به ترتیب P1,P2,P3 در زمان O می رســـند و زمان انفجار پردازنده بر حســـب ثانیه است

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

اگر از کوانتوم زمانی 4 میلی ثانیه ا ستفاده کنیم، آن گاه فرایند P1، 4 میلی ثانیه ی اول را می گیرد، چون به 20 میلی ثانیه ی دیگر نیاز دارد، پس از اولین کوانتومِ زمانی قبضـــه می شود و پردازنده به فرایند بعدی موجود در صف، یعنی p2 اختصاص می یابد. فرایند p2 به 4 میلی ثانیه نیاز ندارد. به همین دلیل قبل از انقضـــای کوانتوم زمانی آن، خاتمه می یابد و سپس پردازنده به فرایند بعدی ، یعنی p3 تخصیص می یابد. پس از این که هر فرایند، پردازنده را ، به اندازه ی یک کوانتوم زمانی در اختیار گرفت. پردازنده به فرایند p1بر می گردد تا کوانتوم زمانی دیگری در اختیارش با شد. زمانبندی RR مربوط به این مثال در نمودار GANTT گانت زیر آمده است:



میانگین زمان انتظار را براین این زمانبندی محاسبه می کنیم

P1 به میزان (4-(4+3+3)) = 6 میلی ثانیه منتظر می ماند.

P2 به میزان 4 میلی ثانیه منتظر می ماند

P3 به میزان 7 میلی ثانیه منتظر می ماند

بنابراین دراین مثال، میانگین زمان انتظار برابر با

6 = <waiting time for p1>

4 = <waiting time for p2>

7 = <waiting time for p3>

6 + 4 + 7 = 17

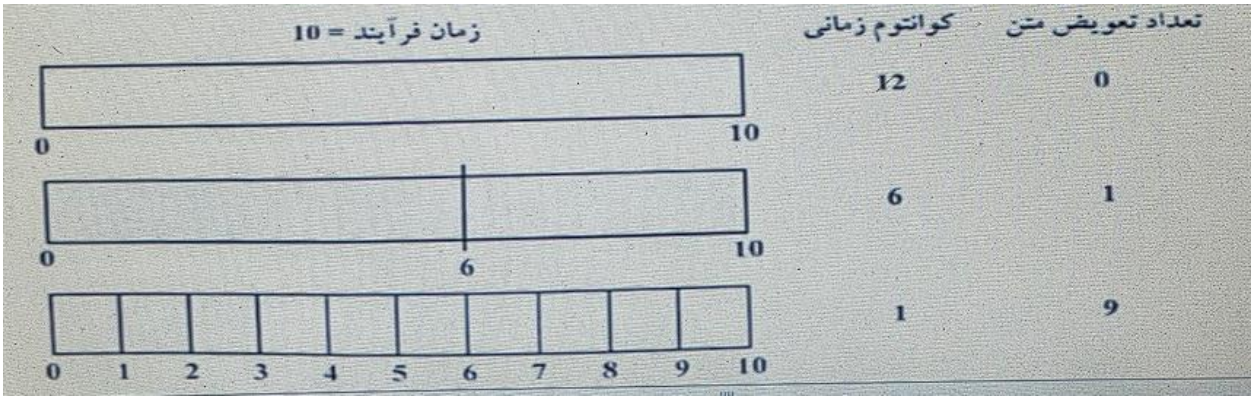
17 / 3= 5.66 = average waiting time

در الگوریتم زمانبندی نوبت گرد شی، پردازنده به هیچ فرایندی به طور متوالی بین از یک کوانتوم زمانی تخصیص نمی یابد (مگر این که تنها فرایند قابل اجرا باشد).

اگر زمان انفجار پردازنده ی فرایندی بیش از یک کوانتوم زمانی شود، ان فرایند قبضه می شود و در انتهای صف آماده قرار می گیرد.

الگوریتم RR، با قبضه کردن (Preemptive) همراه است

اگر n فرایند در صف آماده وجود داشته باشند و کوانتوم زمانی برابر با Q باشد. هر فرایند 1/n زمان پردازنده را حداکثر در واحد زمانی و در اختیار می گیرد. هر فرایند برای به دســـت گرفتن پردازنده در یک کوانتوم زمانی دیگر، نباید بیش ازq * (n-1) واحد زمانی منتظر بماند. به عنوان مثال، اگر 5 فرایند وجود داشته باشد و کوانتوم زمانی برابر با 20 میلی ثانیه باشد، هر فرایند حداکثر در هر 100 میلی ثانیه حداکثر 20 میلی ثانیه پردازنده را در اختیار می گیرد.



کارایی الگوریتم RR شـــدیدا به اندازه کوانتوم زمانی بســـتگی دارد. از یک طرف اگر کوانتوم زمانی بسـیار بزرگ باشـد، سـیاسـت RR مثل FCFS خواهد بود. برعکس ، اگر کوانتوم کوچک باشد (مثلا 1 میلی ثانیه)، روش RR می تواند تعداد زیادی تعویض متن (Context Switch) ایجاد کند (سـر بار زیاد). فرض می کنیم فقط یک فرایند با 10 واحد زمانی داریم. اگر کوانتوم زمانی برابر با 12 واحد زمانی با شد ،فرایند در کمتر از 1 کوانتوم زمانی خاتمه می یابدوســـربازی ندارد، اگر کوانتوم زمانی برابر با 6 واحد زمانی باشـــد فرایند به 2 کوانتوم زمانی نیاز دارد و منجر به یک تعویض متن می شـــود. اگر کوانتوم زمانی برابر با 1 واحد زمان با شد ، نیاز به 9 تعویض متن ا ست. و بدین ترتیب ، اجرای فرایند کند می شود. بنابراین برای مقابله با تعداد تعویض متن، علاقه مند هستیم که کوانتوم زمانی بزرگ با شد، اگر زمان تعویض متن تقریبا 10 (1/10 - N/1) در صد کوانتوم زمانی با شد، انگاه 10 در صد از زمان پردازنده صرف تعویض متن می شود. در عمل اغلب سیستم های مدرن دارای زمان کوانتوم بین 10 تا 100 میلی ثانیه هستند. زمان مورد نیاز برای تعویض متن معمولا کمتر از 10 میلی ثانیه اســـت بنابراین، زمان تعویض متن، کسر کوچکی از کوانتوم زمانی است.

زمان برگشت (Turn Around Time) (زمان کل) نیز به اندازه کوانتوم بستگی دارد. میانگین زمان برگشـــتِ مجموعه ای از فرایندها، با افزایش اندازه ی کوانتوم زمانی، الزاماً

بهبود نمی یابد. به طور کلی میانگین زمان برگشت در صورتی می تواند بهبود یابد که اغلب فرایند ها انفجار بعدی پردازنده ی خود شان را فقط در یک کوانتوم زمانی به اتمام برسانند. به عنوان مثال :

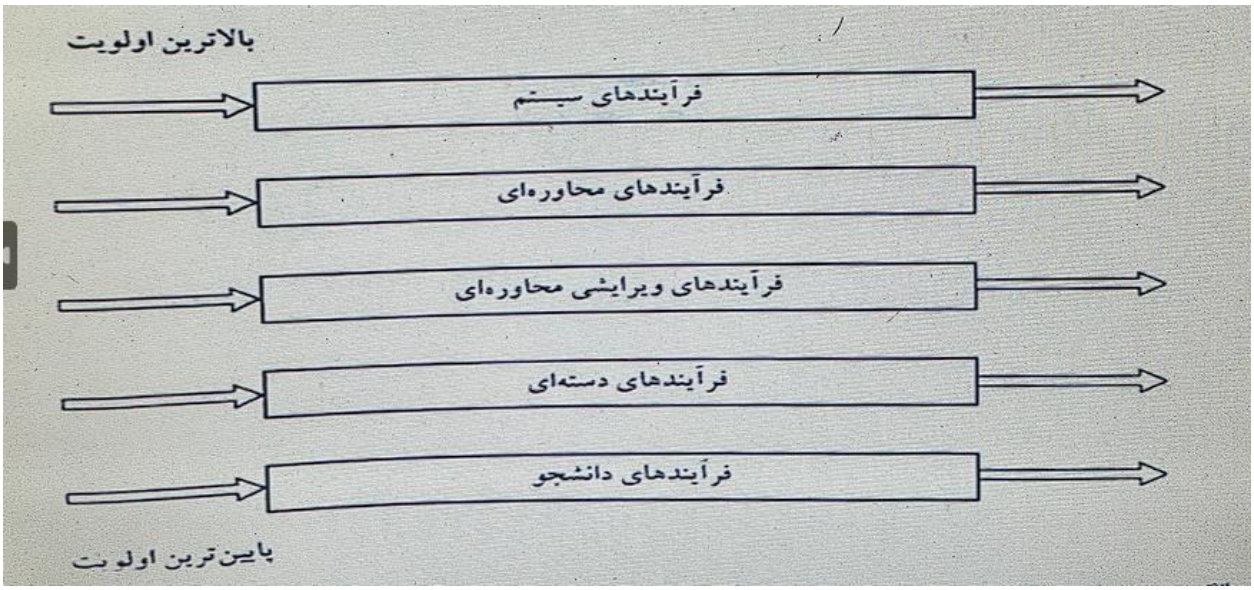
با سه فرایند که هرکدام به 10 واحد زمانی نیاز دا شته با شد و کوانتوم زمانی برابر با 1 واحد زمانی با شد، میانگین زمان برگشت 29 است. اگر کوانتوم زمانی برابر با 10 با شد. میانگین زمان برکشت به 20 کاهش می یابد. با در نظر گرفتن زمان تعویض متن، هر چه کوانتوم زمانی کوچک تر باشد. میانگینِ زمانِ برگشت افزایش می یابد زیرا نیاز به تعویض متن بیشتری دارد.

گرچه کوانتوم زمانی باید در مقایسه با زمان تعویض متن بزرگ تر با شد. اما نیاز نیست بسیار بزرگ با شد، اگر کوانتوم زمانی بسیار بزرگ با شد زمانبندی RR به FCFS تبدیل می شود یک حساب سرانگشتی نشان می دهد که 80 دردصد انفجار های پردازنده باید کوتاه تر از زمان کوانتوم زمانی باشد

زمانبندی صف چند سطحیMultilevel queue

دسته ی دیگری از الگوریتم های زمانبندی برای وضعیت هایی ایجاد شدند که در آن ها، فرایند ها می توانند به دو گروه تقسیم شوند. به عنوان مثال، یک تقسیم بندی متداول

این است که فرایند ها دو دسته اند: فرایند های پیش زمینه (محاوره ای) و پس زمینه (دسته ای). این دو نوع فرایند، زمان پاسخ زمان پاسخ متفاوتی دارند و در نتیجه باید زمانبندی های متفاوتی داشته باشند. علاوه بر این ممکن است فرایند های پیش زمینه اولویت بیشتری نسبت به فرایند های پس زمینه داشته باشند



الگوریتم زمانبندی صف چند سطحی، صف آماده را به چند بخش مجزا تقسیم می کند. هر فرایند بر اساس صفاتی که دارد در صفی قرار می گیرد. این صفات عبارت اند از:

اندازه حافظه، اولویت فرایند، نوع فرایند.

هر صف، الگوریتم زمانبندی خاص خودش را دارد. به عنوان مثال، ممکن است برای فرایند های پیش زمینه و پس زمینه از صف های جداگانه ای استفاده شود و صف پیش زمینه بر اساس الگوریتم RR و صف پس زمینه بر اساس الگوریتم FCFS زمانبندی شود.

علاوه بر این، بین صف ها نیز باز زمانبندی وجود داشته باشد که بر اساس زمانبندی همراه با قبضه کردن (Preemptive) و با اولویت ثابت ، پیاده سازی می شود.

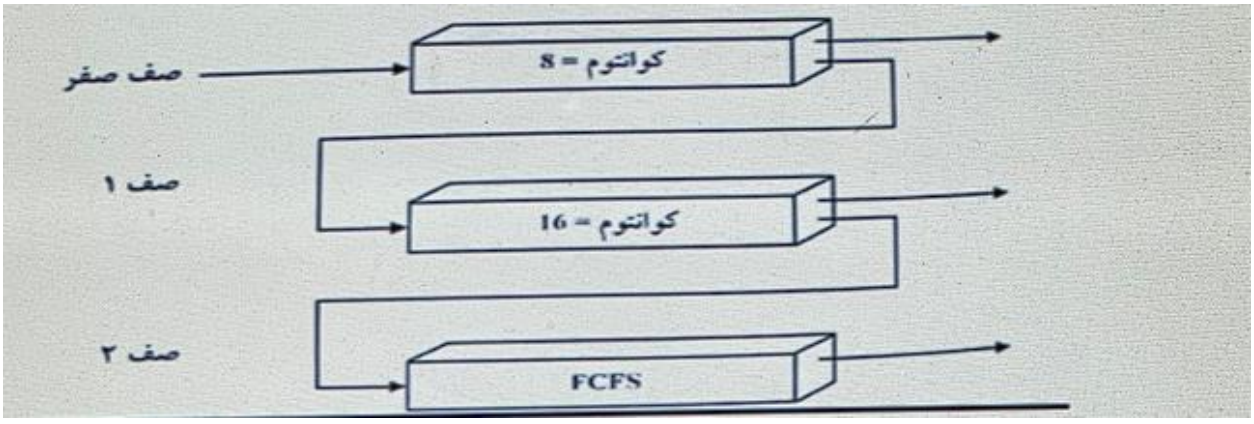
به عنوان مثال: صف پیش زمینه ممکن است نسبت به صف پس زمینه اولویت مطلق(بیشتری) داشته باشد

هر صف نسبت به صف های با اولویت پایین تر، اولویت مطلقی دارد.

امکان دیگر، استفاده از برهه ی زمانی در بین صف هاست. هر صف بخشی از از زمان پردازنده را به خود اختصاص می دهد و می تواند بین فرایند های مختلف خود زمانبندی کند. به عنوان نمونه در مثالِ مربوط به فرایند های پیش زمینه و پس زمینه، صف پیش زمینه می تواند 80 درصد پردازنده را در اختیار گیرد و آن را به روش RR بین فرایند هایش زمانبندی کند، در حالی که صف پس زمینه 20 دردصد وقت پردازنده را در اختیار می گیرد و آن را به روش FCFS بین فرایند هایش زمانبندی می کند.

زمانبندی صف چند سطحی بازخوردی (فیدبک) (feedback)

Multilevel feedback queue



معمولا در الگوریتم زمانبندی صف چند سطحی، فرایند ها هنگام ورود به سیستم در صفی قرار می گیرند. به طوری که از صفی به صف دیگر نمی رود. به عنوان مثال» اگر صف های جداگانه ای برای فرایند های پیش زمینه و پس زمینه وجود داشته باشد، فرایند ها از صفی به صف دیگر منتقل نمی شوند، زیرا ماهیت پیش زمینه ای و پس زمینه ای آن ها تغییر نمی کند. این کار موجب کاهش سربار زمانبندی می شود ولی قابلیت انعطاف کم می شود

اما، الگوریتم زمانبندی صف چند سطحی بازخوردی به فرایند ها اجازه می دهد از صفی به صف دیگر منتقل شوند. فلسفه ی این کار این است که ویژگی های انفجار های پردازنده ی فرایند ها با یکدیگر متفاوت است. اگر فرایندی پردازنده را مدت زیادی در اختیار گیرد، به صفی با اولویت پایین تر منتقل می شود. بدین ترتیب، فرایند های در تنگنای i/o و محاوره ای، در صف هایی با اولویت بالاتر قرار می گیرند. به طور مشابه ، فرایندی که به مدت زیادی در صفی با اولویت پایین تر منتظر می ماند، ممکن است به صفی با اولویت بالاتر منتقل شود. در این شکل سالمندی (Aging) ، از مشکل گرسنگی (قحطی) (starving) جلوگیری می شود

به عنوان مثال، یک زمانبند صف چند سطحی بازخوردی ، با سه صف را در نظر بگیرید گه از صف تا 2 شماره گذاری شده اند.

زمانبند ابتدا تمام فرایند های موجود در صف را اجرا می کند. وقتی صف صفر خالی باشد، فرایند های صف 1 اجرا می شوند. به همین ترتیب، فرایند های صف 2 وقتی اجرا می شوند که صف های 0 و 1 خالی باشند. فرایندی که برای صف 1 می آید. یک فرایند

از صف 2 را قبضه می کند. به همین ترتیب، هر فرایند موجود در صف 1 ، توسط فرایندی که جدیدا وارد صف صفر می شود ، قبضه (Preemptive) می شود

فرایندی که می خواهد به صف آماده وارد شود در صف صفر قرار می گیرد. به هر فرایند در صف صفر، کوانتوم زمانی 8 میلی ثانیه ای نسبت داده می شود. اگر فرایندی در این مدت زمان به اتمام نرسد، به انتهای صف 1 منتقل می شود. اگر صف صفر خالی باشد، به فرایند موجود در ابتدای صف 1 ، کوانتوم زمانی 16 میلی ثانیه ای تخصیص می یابد.

اگر اجرای ان در این مدت زمان کامل نشد، قبضه می شود و در صف 2 قرار داده می شود. در صورتی که هر یک از دو صف 0 و 1 خالی باشند؛ فرایند های موجود در صف 2 بر اساس الگوریتم FCFS اجرا می شوند.

این الگوریتم زمانبندی ، به فرایندی که انفجار پردازنده ی آن کمتر یا مساوی 8 میلی ثانیه ای باشد، بالاترین اولویت را می دهد. چنین فرایندی، سریعا پردازنده را در اختیار می گیرد، انفجار پردازنده ی خودش را انجام می دهد و به انفجار بعدی i/o خود می رود فرایند هایی که به بیش از (16+8) کوانتوم زمانی نیاز داشته باشند،

گرچه اولویت کمتری نسبت به فرایند های کوتاه تر دارند ، ولی سریعا اجرا می شوند. فرایند های طولانی با زمان بیشتر از 24 کوانتوم زمانی، به طور خودکار به صف 2 می روند و به ترتیب FCFS اجرا می شوند

(با چرخه های پردازنده ی باقی مانده از صف 0 و 1)

زمانبند صف چند سطحی بازخوردی با پارامتر های زیر تعریف می شوند

- تعداد صف ها
- الگوریتم زمانبندی برای هر صف
- روشی که تعیین می کند چه هنگامی یک فرایند به صفی با اولویت بیشتر منتقل شود.
- روشی که تعیین می کند چه هنگامی یک فرایند به صفی با اولویت کمتر منتقل شود.
- روشی که تعیین می کند فرایندی که نیاز به خدمات دارد، به چه صفی وارد شود

زمانبند صف چند سطحی بازخوردی، متداول ترین الگوریتم زمانبندی پردازنده است.

این الگوریتم، پیچیده ترین الگوریتم است، زیرا تعریف بهترین زمان، نیاز به ابزار هایی برای انتخاب مقادیر برای تمام پارامتر ها دارد.