سیستم عامل جلسه پنجم

ادامه الگوریتم های زمانبدی

زمان بندی 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.

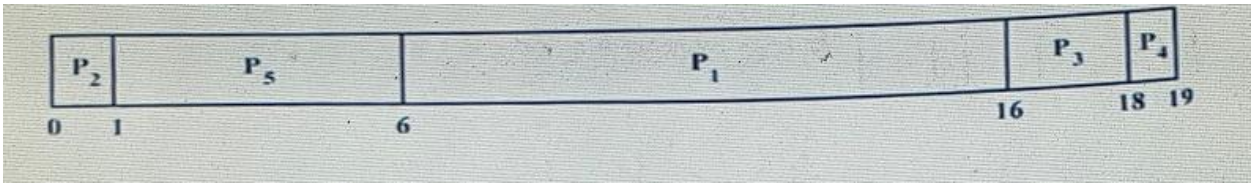
بعضی از افراد برای اولویت های پایین از اعداد کوچک استفاده می کنند ولی بعضی دیگر برای اولویت پایین از اعداد بزرگ استفاده می کنند. این کار موجب سردرگمی است.

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 p1,p2,p3, ... , p5 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 اســاس معیار هایی تعیین می شــوند که از نظر ســیســتم 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 صــف آماده می رســد، اولویت آن با اولویت فرایند در حال اجرا مقایســه می شــود. اگر فرایندی که تازه وارد صف شده بیشتر از اولویت فرایندی با شد که در حال اجرا ا ست، الگوریتم زمانبندی با قبضــه 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

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

فرایندی که آماده ی اجرا اســت ولی منتظر پردازنده باشــد، مســدود در نظر گرفته می شود.

الگوریتم زمانبندیِ با اولویت می تواندو منجر به این شــود 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 شــوند 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 پاســد و همه فرایند های با اولویت پایین که تمام نشــده 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

زمانبندی نوبت گردشی "R R" (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 جابهجا می شودو 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 سیستم عامل می فرستند. 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 سیستم 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 است سیستم، متن 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 شــود. متن، در PCB (Process control Block) مربوط به فرایند ذخیره می شــود. متن شــامل ثبات های پردازنده ، حالت فرایند و اطلاعات مدیریت حافظه است.

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

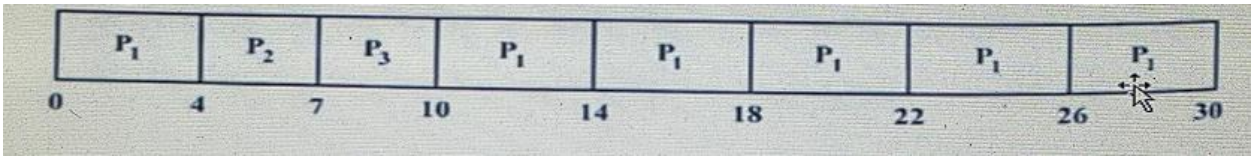
این کار ، تعویض متن یا تعویض بستر یا (Context Switch) نام دارد. وقتی تعویض متن صـــورت می گیرد، هســـته متن فرایند قبلی را در PCB (Process Control Block) آن ذخیره می کند و متن ذخیره شده ی فرایند جدید را که برای اجرا زمانبندی شد، بار می کند. زمان تعویض متن، سرباز محض است، زیرا هنگام تعویض، سیستم، کاری 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 یابدو 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 (10-4) = 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 دراینمثال، میانگین زمان انتظار برابر با

17=(6+7+3)

5.66=3/17

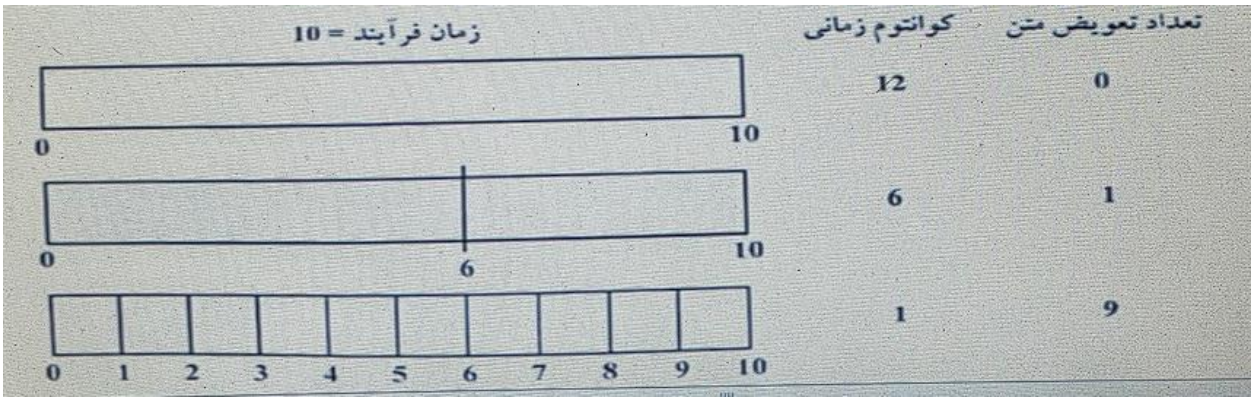
در الگوریتم زمانبندی نوبت گردشی، پردازنده به هیچ فرایندی به طور متوالی بین از یک کوانتوم زمانی تخصیص نمی یابد (مگر این که تنها فرایند قابل اجرا باشد).

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 د ست گرفتن پردازنده در یک کوانتوم زمانی دیگر، نباید بیش از q * (n-1) واحد زمانی منتظر بماند. به عنوان مثال، اگر 5 فرایند وجود دا شته با شد و کوانتوم زمانی برابر

با 20 میلی ثانیه باشـــد، هر فرایند حداکثر در هر 100 میلی ثانیه حداکثر 20 میلی ثانیه پردازنده را در اختیار می گیرد.



کارایی الگوریتم RR شدیدا به اندازه کوانتوم زمانی بستگی دارد. از یک طرف اگر کوانتوم زمانی بســیاز بزرگ باشــد، ســیاســت RR مثل FCFS خواهد بود. برعکس ، اگر کوانتوم کوچک باشـــد (مثلا 1 میلی ثانیه)، روش RR می تواند تعداد زیادی تعویض متن ایجاد کند. فرض می کنیم فقط یک فرایند با 10 واحد زمانی داریم. اگر کوانتوم زمانی برابر با 12 واحد زمانی باشد ، فرایند در کمتر از 1 کوانتوم زمانی خاتمه می یابد و سربازی ندارد، اگر کوانتوم زمانی برابر با 6 واحد زمانی با شد فرایند به 2 کوانتوم زمانی نیاز دارد و منجر به یک تعویض متن می شـــود. اگر کوانتوم زمانی برابر با 1 واحد زمان باشـــد ، نیاز به 9 تعویض متن اســـت. و بدین ترتیب ، اجرای فرایند کند می شـــود. بنابراین برای مقابله با تعداد تعویض متن، علاقع مند هســـتیم که کوانتوم زمانی بزرگ باشـــد، اگر زمان تعویض متن تقریبا 10 (1/10 - N/1) درصـــد کوانتوم زمانی باشـــد، انگاه 10 درصـــد از زمان پردازنده صـــرف تعویض متن می شـــود. در عمل اغلب ســـیســـتم های مدن دارای زمان کوانتوم بین 10 تا 100 میلی ثانیه هستند. زمان مورد نیاز برای تعویض متن معمولا کمتر از 10 میلی ثانیه است بنابراین، زمان تعویض متن، کسر کوچکی از کوانتوم زمانی است.

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

بهبود نمی یابد. به طور کلی میانگین زمان برگشـــت در صـــورتی می تواند بهبود یابد که اغلب فرایند ها انفجار بعدی پردازنده ی خودشـــان 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 برگ شت 29 ا ست. اگر کوانتوم زمانی برابر با 10 با شد. میانگین زمان برکشـــت 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 مقای سه با زمان تعویض متن بزرگ تر با شد. اما نیاز ب سیاز بزرگ باشد، اگر کوانتوم زمانی بسیار بزرگ باشد زمانبندی RR به FCFS تبدلی می شود یک حساب سرانگشتی نشان می دهد که 80 درصد انفجار های پردازنده باید کوتاه تر از زمان کوانتوم زمانی باشد