سیستم عامل جلسه 3

حالت های فرآیند ها

وقتی فرایندی اجرا می شود، حالت آن تغییر می کند. حالت فرایند تا حدی توسط فعالیت فعلی آن تعریف می شود

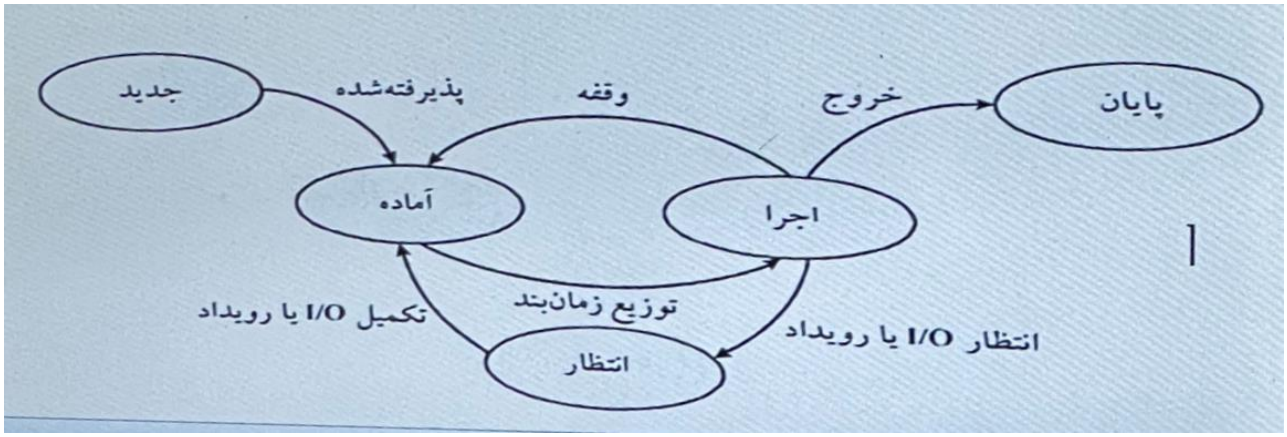
هر فرایند ممکن است در یکی از حالت های زیر باشد

جدید: فرآیند ایجاد می شود

اجرا: دستورات در حال اجرا هستند

انتظار: فرآیند منتظر وقوع رویدادی است (مثل کامل شدن عمل i/o یا دریافت یک سیگنال)

پایان: اجرای فرایند خاتمه یافته است



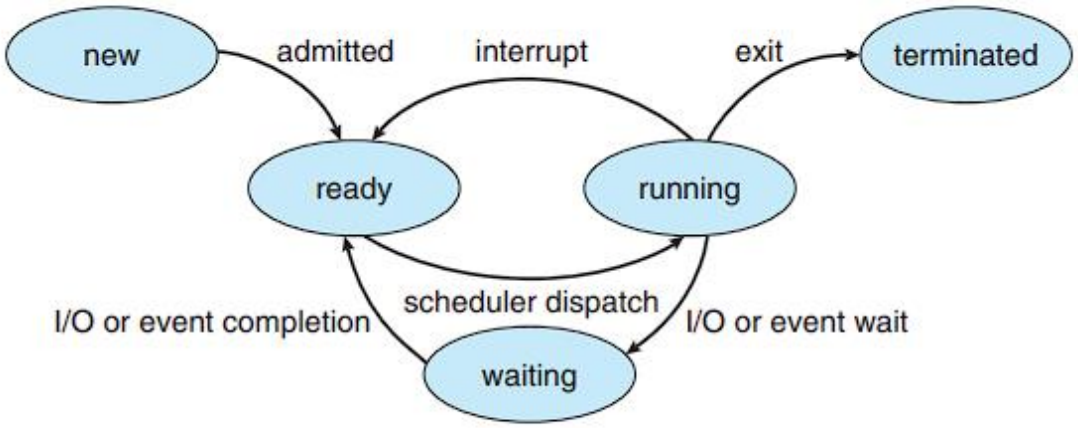


Figure 3.2 Diagram of process state.

مفاهیم اساسی زمانبندی

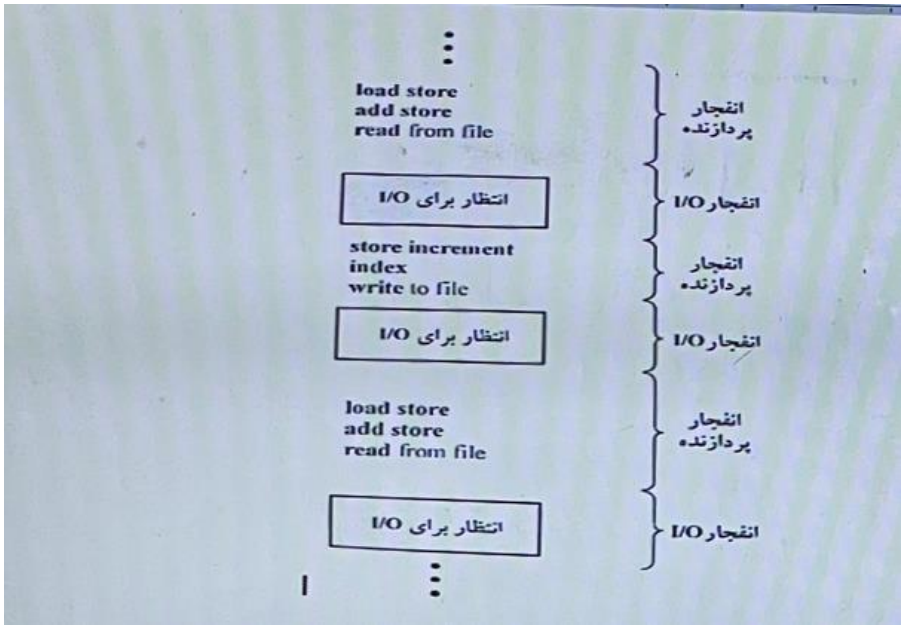
در یک سیستم تک پردازنده ای ، در هر زمان فقط یک فرایند می تواند اجرا شود، بقیه باید منتظر بمانند تا پردازنده آزاد شود تا دوباره بتوانند زمانبندی شوند. هدف چند برنامه ای این است که همیشه چندین فرایند در حال اجرا باشند. تا بهره وری پردازنده به حداکثر برسد. هر فرایند به اجرایش ادامه می دهد تا برای یک عمل I/O در انتظار بماند در یک سیستم کامپیوتری ساده ، در این مدت پردازنده باید بیکار بماند

در چندین برنامه ای، سعی می شود که از این زمان به نحوه احسن استفاده شود، در این سیستم ، چند فرایند به طور همزمان در حافظه نگهداری می شوند، وقتی یک فرایند به حالت انتظار می رود. سیستم عامل، پردازنده را از آن فرایند می گیرد و به فرایند دیگری می دهد. این روند ادامه می یابد ، هر وقت که فرایندی به حالت انتظار رفت، فرایند دیگری از پردازنده استفاده می کند

تقریبا تمام منابع کامپیوتری، قبل از استفاده از زمانبندی می شوند. پردازنده یکی از منابع اصلی کامپیوتر است و در نتیجه زمانبندی آن موضوع اصلی طراحی سیستم عامل است

چرخه انفجار i/o و انفجار پردازنده

موفقیت زمانبندی پردازنده به این خاصیت فرایند ها بستگی دارد ، که اجرای فرایند شامل چرخه ای از اجرای پردازنده و انتظار برای i/o است. فرایند ها بین این دو حالت سوییچ می کنند. اجرای فرایند با انفجار پردازنده شروع می شود. و به دنبال آن یک انفجار i/o قرار دارد که بعد از آن ، انفجار دیگری از پردازنده و سپس انفجار دیگری از i/o قرار دارد و به همین ترتیب ادامه می یابد . سرانجام، آخرین انفجار پردازنده با درخواست سیستم برای خاتمه اجرا، پایان می پذیرد



مدت این انفجار های پردازنده از فرایندی به فرایند دیگری و از کامپیوتری به کامپیوتر دیگر متفاوت است. تعداد زیادی از انفجار های کوتاه پردازنده و تعداد اندکی از انفجار های بلند پردازنده وجود دارد. برنامه مقید به i/o (در تنگنای i/o) معمولا دارای چند انفجار کوتاه پردازنده است. برنامه مقید به پردازنده (در تنگنای پردازنده) معمولا چند انفجار بلند پردازنده دارد. این توزیع می تواند در انتخاب الگوریتم زمانبندی پردازنده مهم باشد

انواع زمانبندی ها

کلید چند برنامگی زمانبندی است، زمانبندی بر روی کارایی سیستم اثر می گذارد زیرا مشخص می کند کدام فرایند ها منتظر مانده و کدام فرایند ها به جلو بروند

انواع زمانبندی برای پردازنده عبارت است از:

1. زمانبند بلند مدت (Long Term scheduler)

تصمیم گیری در مورد افزودن به مجموعه ی فرایندها برای اجرا

2. زمانبند میان مدت (Middle Term scheduler)

تصمیم گیری در مورد افزودن به تعداد فرایند هایی که بخش یا تمام آن ها در حافظه اصلی است.

3. زمانبند کوتاه مدت (Short Term Scheduler)

تصمیم گیری در مورد این که کدام یک از فرایند های موجود در حافظه اصلی، برای اجرا توسط پردازنده انتخاب شوند

4. زمانبند ورودی – خروجی (i/o Scheduler)

تصمیم می گیرد که کدام درخواست **i/o** فرایند ها به وسیله یک دستگاه **i/o** موجود انجام گیرد.

وظیفه فعال سازی و تعلیق فرایند ها بر عهده زمانبند میان مدت است

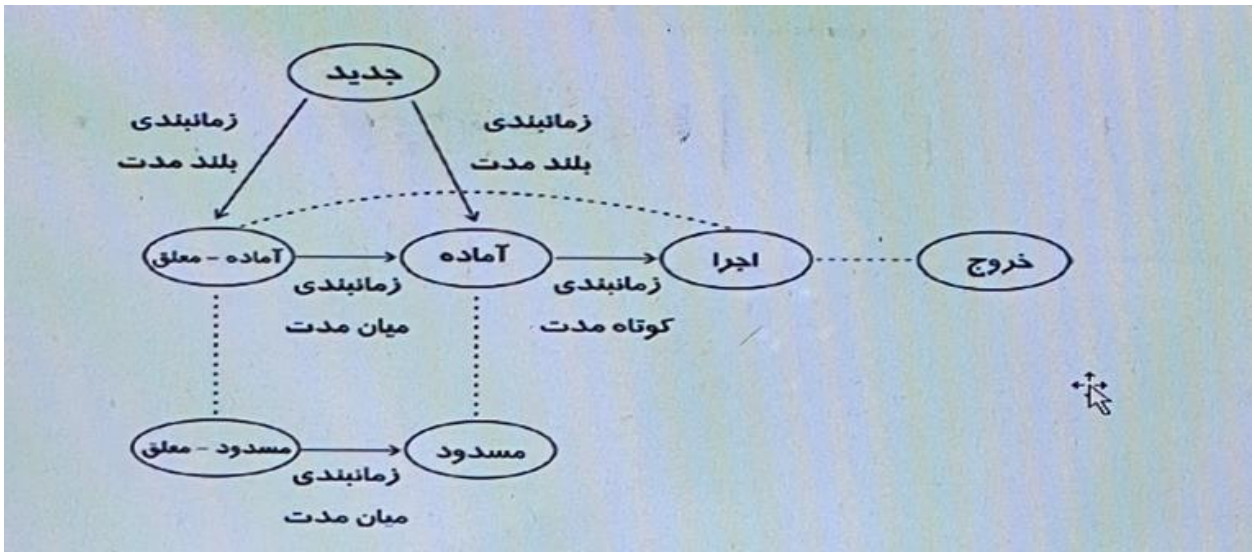
زمانبند میان مدت فرایندی را از حافظه اصلی حذف و به حافظه جانبی می برد. این فرایند بعدا می تواند به حافظه اصلی لود شود. این الگو را مبادله (**Swapping**) می گویند.

ایده اصلی زمانبندی میان مدت این است که می تواند فرایندی را از حافظه حذف کند و درجه چند برنامگی را کاهش دهد.

زمانبند بلند مدت ترکیب خوبی از فرایند های **i/o limited** و **CPU limited** ، انتخاب می کند. نام دیگر زمانبند بلند مدت ،زمانبند کار است. نام دیگر زمانبند کوتاه مدت، زمانبند پردازنده است.

زمانبند بلند مدت نسبتا دفعات کمی اجرا می شود. زمانبند میان مدت نسبتا تعداد دفعات بیشتری به اجرا در می آید و زمانبند کوتاه مدت بیشترین دفعات اجرا را دارد.

تذکر : اکثر سیستم های **اشتراک زمانی** فاقد زمابند بلند مدت می باشند.

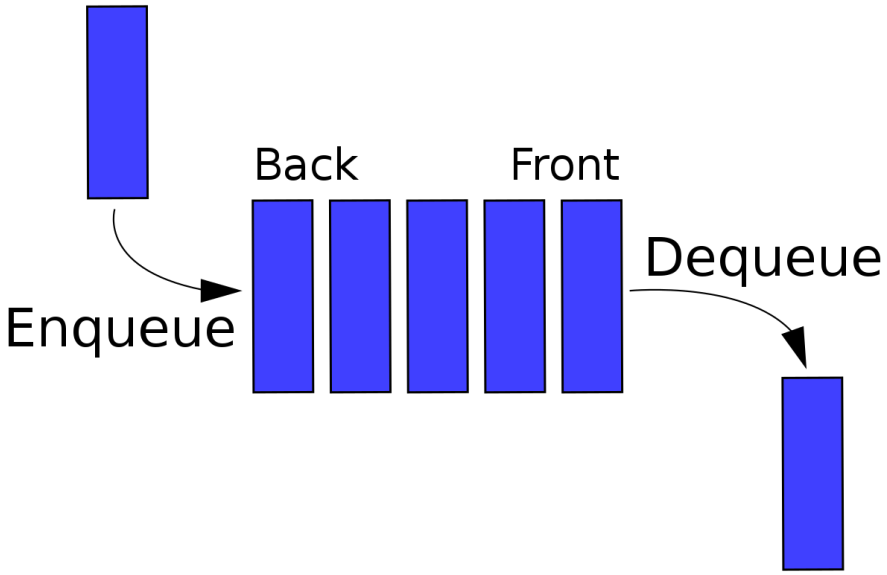


زمانبندی پردازنده

هر وقت پردازنده بیکار می شود، سیستم عامل باید یکی از فرایند های موجود در صف آماده را برای اجرا انتخاب کند. این انتخاب توسط زمانبند کوتاه مدت (زمان بند پردازنده) انجام می گیرد.

زمان بند ،فرایندی را از بین فرایند های موجود در حافظه که آمادگی اجرا دارند انتخاب می کند و پردازنده را به آن تخصیص می دهد.

صف الزاما یک صف با ویژگی "خروج به ترتیب" (FIFO) (First in First Out) نیست. همان طور که هنگام بحث در مورد الگوریتم های زمانبندی خواهیم دید، صف آماده می تواند به صورت صف (FIFO) ، صف اولویت ، درخت (tree) یا یک لیست پیوندی (Linked list) نامرتب پیاده سازی شود. از نظر مفهومی تمام فرایند های موجود در صف آماده ،منتظر به دست آوردن پردازنده هستند تا اجرا شوند



زمان بندی با قبضه کردن (Preemptive) یا غیر انحصاری یا انقطاع پذیر

فرایند درحال اجرا می تواند به وسیله سیستم عامل متوقف شود و به حالت آماده منتقل شود.در این الگوریتم ها در پایان برش زمانی و یا تغییر شرایط سیستم مدیر زمانبندی می تواند کنترل پردازنده را از یک "پردازه ی(فرایند)" در حال اجرا گرفته و به " پردازه ی (فرایند)" دیگری بدهد

تصمیمات زمانبندی پردازنده ممکن است تحت چهار شرط ریز اتخاذ شود:

1. وقتی که فرایندی از حالت اجرا به حالت انتظار می رود (مثل در خواست i/o ، یا فراخوانی wait() برای خاتمه ی یکی از فرایند های فرزند)
2. وقتی فرایندی از حالت اجرا به حالت آماده می رود (مثل وقتی که رویدادی رخ می دهد)
3. وقتی که فرایندی از حالت انتظار به حالت آماده می رود (مثل تکمیل شدن i/o)
4. وقتی فرایندی خاتمه می یابد.

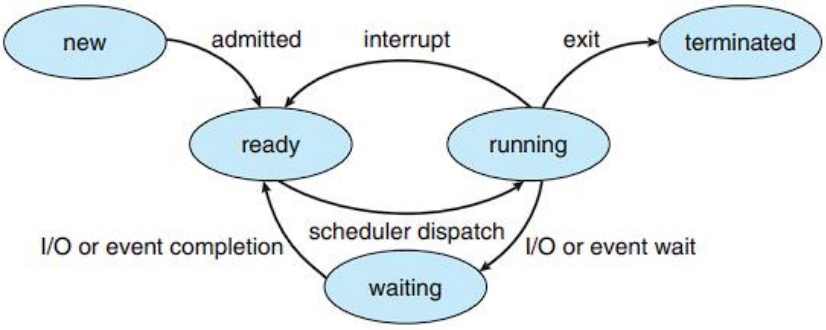
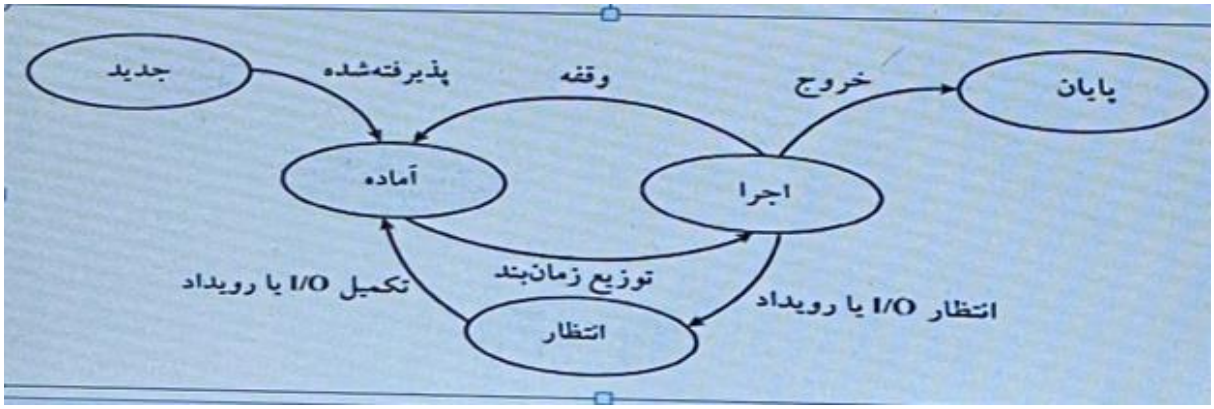


Figure 3.2 Diagram of process state.



برای شرط 1 و 4 انتخابی بر حسب زمانبندی وجود ندارد. در صورت وجود فرایندی در صف آماده ، یک فرایند باید برای اجرا انتخاب شود . اما برای شرایط 2 و 3 امکان انتخاب وجود دارد.

وقتی زمانبندی فقط تحت شرایط 1 و 4 انجام می گیرد الگوی زمانبندی را **بدون قبضه کردن** (NonPreemptive) می نامیم و در غیر این صورت آن را **با قبضه کردن** (Preemptive) **یا غیر انحصاری** می نامیم.

انحصاری / بدون قبضه کردن

در این الگوریتم ها تا زمان خاتمه پردازنده و یا زمان نیاز به عمل I/O نمی توان CPU را از پردازه ی (Process) در حال اجرا گرفت و پردازه ی دیگری داد

همین که یک فرایند در حالت اجرا قرار گرفت آن قدر به اجرا ادامه می دهد تا خاتمه یابد یا با این که خودش (داوطلبانه) برای انتظار i/o مسدود شود

یادداشت:

preemptive در کامپیوتر یعنی خالی کردن

مثلا در سی پی یو اگر زمانبندی preemptive باشد یعنی ما این اجازه را داریم که سی پی یو را از یک برنامه ای که سی پی یو الان دستشه و وقتش هم تمام نشده ازش بگیریم

برعکس این مدل nonpreemptive است و به این معناست که ما حق گرفتن سی پی یو از برنامه ای را نداریم و باید منتظر بمانیم تا سی پی یو خودش برنامه را رها کند

در زمانبندی بدون قبضه کردن،وقتی پردازنده به فرایندی تخصیص می یافت، آن قدر پردازنده را نگه می دارد تا این که خاتمه یابد یا به حالت انتظار برود که در این صورت پردازنده را آزاد می کند. این روش زمانبندی توسط ویندوز 3.xمورد استفاده قرار می گرفت. .ویندوز 95 ، زمابندی با قبضه کردن را معرفی کرد وتمام نسخه های بعدی سیستم عامل ویندوز از زمانبندی با قبضه کردن استفاده نمودند. سیستم عامل Mac OS X برای مکینتاش نیز از زمانبندی با قبضه کردن استفاده میکند، نسخه قبلی سیستم عامل مکینتاش مبتنی

بر زمانبندی همکاری (Cooperative Scheduling) بود. زمانبندی همکاری تنها روشی است که می تواند بر روی سکوهای سخت افزار اجرا شود، زیرا به سخت افزار خاصی (مثلا تایمر) که برای زمانبندی با با قبضه کردن نیاز است نیاز ندارد.

زمانبندی با قبضه کردن هزینه دارد . حالتی را در نظر بگیرید که دو فرایند داده های مشترکی دارند. یکی از آن ها ممکن است در زمان به هنگام ذخیره سازی داده ها قبضه شود و پردازنده ی دوم در حال اجرا باشد. فرایند دوم ممکن است تلاش کند داده هایی را بخواند که در حالت ناسازگار قرار دارد. بنابراین نیاز به راهکار جدیدی است تا دستیابی به داده های مشترک را هماهنگ کند

زمانبندی با قبضه کردن در طراحی هسته سیستم عامل موثر است. هنگام پردازش فراخوان سیستم ،هسته ممکن است مشغول انجام فعالیتی در رابطه با یک فرایند باشد.این فعالیت ها ممکن است داده های مهم هسته را تغییر دهند (مثل صف i/o)

اگر فرایندی که در حال انجام این تغییرات است ، قبضه شود و هسته (یا گرداننده دستگاه) نیاز به خوانده یا تغییر همان ساختار داشته باشد ، چه اتفاقی می افتد؟

به نظر می رسد که هرج و مرج پیش می اید. بعضی از سیستم های عامل، از جمله اغلب نسخه های یونیکس، این مسئله را به این ترتیب حل می کنند. که قبل از تعویض متن ، منتظر می مانند تا فراخوان سیستم کامل شود یا یک عمل i/o رخ دهد و این الگو تضمین می کند که ساختار هسته ساده باشد ،زیرا در حالی که ساختمان داده های هسته در وضعیت ناسازگاری وجود دارند ؛ هسته فرایندی را قبضه نمی کند.

متاسفانه این مدل اجرای هسته برای پشتیبانی بی درنگ و پردازشی مناسب نیست.

چون وقفه ها درهر زمانی می توانند رخ دهند و چون هسته همیشه نمی تواند آن ها را نادیده بگیرد، بخش هایی از کد تحت تاثیر وقفه ها قرار دارند ، باید از استفاده ی همزمان در امان باشند و لازم است سیستم عامل وقفه ها را در تمام اوقات بپذیرد، وگرنه ممکن است ورودی مفقود شود و خروجی از بین برود. برای این که چندین فرایند به طور همزمان به این بخش های کد دسترسی نداشته باشند، هنگام ورود به آن ها وقفه را غیرفعال و هنگام خروج، وقفه را فعال می سازند. توجه به این نکته مهم است که بخش هایی از کد که وقفه ها را غیرفعال می کند زیاد نیستند و دستور العمل های زیادی ندارند.

وقفه (interrupt)

وقفه رخدادی در سیستم عامل است. که ترتیب اجرای دستورالعمل ها را توسط پردازشگر (CPU) تغییر می دهد. وقفه را سخت افزار کامپیوتر تولید می کند اما علت آن می تواند نرم افزاری باشد

یادداشت: در لینوکس وقفه های 1-18 وقفه های Nonmaskable هستند یعنی غیرقابل چشم پوشی و از 19 تا 31 برای intel رزرو شده است و از 32 تا 255 آزاد هستند یا Maskable

انواع وقفه ها

داخلی (Trap): بر اثر اجرای دستورات خود برنامه به صورت داخلی در CPU رخ می دهد.(از جنس نرم افزاری است و تولید کننده آن نرم افزار است)

خارجی:از دستگاه های خارجی مثل دستگاه های ورودی یا خروجی ، تایمر ، صفحه کلید و خطاهای سخت افزاری ناشی می شود.

نرم افزار (SVC): بر اثر فراخوانی توابع سیستمی توسط برنامه رخ می دهد

توزیع کننده

مولفه ی دیگری که در زمانبندی پردازنده دخالت دارد، توزیع کننده (Dispatcher) است ؛ توزیع کننده ، پیمانه ای (module) است که کنترل را به پردازنده ای می دهد که زمانبند کوتاه مدت ، آن را انتخاب کرده است

این عمل شامل موارد زیر است:

- **تعویض متن (تعویض بستر)**
- **تغییر به حالت کاربر**
- **پرش به محل مناسبی در برنامه ی کاربر و آغاز مجدد آن برنامه**

توزیع کننده باید سرعت بالایی داشته باشد، زیرا هنگام تعویض هر فرایند فراخوانی می شود. مدت زمانی که توزیع کننده صرف می کند تا یک فرایند را متوقف و فرایند دیگری را شروع کند **تاخیر توزیع** (Dispatch Latency) نام دارد

معیار های زمانبندی

- **بهره وری پردازنده :** میخواهیم تا آن جایی که ممکن است پردازنده مشغول باشد. درصد بهره وری پردازنده می تواند از **0** تا **100** باشد ، در یک سیستم واقعی ، باید از **40** (برای سیستم هایی با بار کم) تا **90** درصد (سیستم هایی با بار زیاد) باشد

- **توان عملیاتی (Throughput)(گذردهی):**اگر پردازنده مشغول اجرای فرآیند ها باشد، کاری که در حال انجام است. یک معیار کار، تعداد فرایند هایی است که در واحد زمان کامل می شوند و توان عملیاتی نام دارد. برای فرایند

های طولانی ،ممکن است بر حسب ساعت سنجیده شود. برای تراکنش های کوتاه توان عملیاتی ممکن است 10 فرایند در ثانیه باشد

- **زمان برگشت (Turnaround time) (یا زمان کل):**از نقطه نظر یک فرایند خاص، مهم ترین معیار، زمان اجرای یک فرایند است. فاصله ی زمانی از زمان تحویل فرایند ، تا زمان کامل شدن اجرای آن، زمان برگشت نام دارد. زمان بازگشت مجموع زمان انتظار برای قرار گرفتن در حافظه، مان انتظار برای قرار گرفتن در صف آماده ، اجرا در پردازنده و عمل i/o است.

- **زمان انتظار:**الگوریتم زمانبندی پردازنده، بر مدت زمانی که در اثنای آن فرایندی که اجرا می شود یا عمل i/o انجام می دهد، تاثیر ندارد.فقط بر مدت زمانی که فرایند در صف آماده منتظر می ماند موثر است، زمان انتظار برابر با مجموع مدت های انتظار یک فرایند در صف آماده است.

- **زمان پاسخ:**در یک سیستم عامل محاوره ای ، زمان بازگشت ممکن است معیار خوبی نباشد. اغلب یک فرایند ممکن است بعضی از نتایج را زود تولید کند و هنگامی که این نتایج به خروجی می روند، محاسبات دیگری صورت می گیرد و نتایج دیگری به خروجی برود. لذا معیار دیگر، زمان تحویل یک درخواست تا دریافت اولین پاسخ است. این معیار که زمان پاسخ نام دارد ، مدت زمانی که طول می کشد پاسخ آماده

شود ، نه مدت زمانی که طول می کشد تا آن پاسخ چاپ شود . زمان برگشت ، به سرعت دستگاه خروجی بستگی دارد.