**Sistemas Operacionais**

1)Agendamento de processos do Windows

1) O Windows 3.1 xs usava um agendador não preventivo, o que significa que ele não interrompeu os programas. Ele contava com o programa para finalizar ou informar ao sistema operacional que não precisava de processador para que pudesse passar para outro processo. Isso geralmente é chamado de multitarefa cooperativa. O Windows 95 introduziu um agendador preventivo rudimentar; no entanto, para o suporte legado optou por permitir que aplicativos de 16 bits sejam executados sem preempção

2) As versões do Windows baseadas em NT usam um agendador de CPU com base em uma fila de feedback multinível, com 32 níveis de prioridade definidos. Destina-se a atender aos seguintes requisitos de projeto para sistemas multimodo:

Dê preferência a trabalhos curtos.

Dê preferência aos processos vinculados de E / S.

Estabeleça rapidamente a natureza de um processo e programe-o de acordo.

Todos os processos recebem um aumento de prioridade após um evento de espera, mas os processos que tiveram uma espera de E / S do teclado recebem um aumento maior do que aqueles que tiveram uma espera de E / S de disco.

Processos em "primeiro plano" recebem maior prioridade.

3) O Windows XP usa um algoritmo de agendamento prioritário preventivo baseado em quantum. O agendador foi modificado no Windows Vista para usar o registro de contador de ciclos dos processadores modernos para controlar exatamente quantos ciclos de CPU um thread executou, em vez de apenas usar uma rotina de interrupção com intervalo de tempo.

Programação de processos do Linux

Das versões 2.6 a 2.6.23, o kernel usava um agendador O (1). O Completely Fair Scheduler é o nome de um agendador de tarefas que foi mesclado na versão 2.6.23 do kernel do Linux. Ele lida com a alocação de recursos da CPU para a execução de processos e visa maximizar a utilização geral da CPU enquanto maximiza o desempenho interativo. Ele usa que usa árvores vermelho-pretas em vez de filas.

Duas classes de processos:

em tempo real (prazos flexíveis)

algoritmo de compartilhamento de tempo

O agendamento normal de processos usa uma política priorizada, preventiva e baseada em crédito:

O Agendador sempre escolhe o processo com mais créditos para executar.

Em cada interrupção do timer, um crédito é deduzido até que zero seja atingido, momento em que o processo é antecipado.

Se não houver um processo pronto, todos os créditos para um processo serão calculados como créditos = créditos / 2 + prioridade.

Essa abordagem favorece processos vinculados de E / S que não gastam seus créditos quando executados.

Os algoritmos de programação Round Robin e FIFO são usados ​​para alternar entre processos em tempo real

Mac OS e macOS clássicos

O Mac OS 9 usa agendamento cooperativo para encadeamentos, onde um processo controla vários encadeamentos cooperativos e também fornece agendamento preventivo para tarefas de multiprocessamento. O kernel agenda tarefas de multiprocessamento usando um algoritmo de agendamento preventivo. Todos os processos do Process Manager são executados em uma tarefa especial de multiprocessamento, chamada de "tarefa azul". Esses processos são planejados de forma cooperativa, usando um algoritmo de round-robin; um processo gera controle do processador para outro processo chamando explicitamente uma função de bloqueio, como WaitNextEvent. Cada processo possui sua própria cópia do Thread Manager que agenda os threads desse processo de forma cooperativa; um thread gera controle do processador para outro thread chamando YieldToAnyThread ou YieldToThread. [12]

O macOS usa uma fila de feedback multinível, com quatro faixas de prioridade para threads - normal, alta prioridade do sistema, apenas modo de kernel e tempo real. [13] Threads são agendados preemptivamente; O macOS também suporta threads agendados em cooperação na implementação do Thread Manager in Carbon. [12]

AIX

No AIX Versão 4, existem três valores possíveis para a política de planejamento de encadeamento:

Primeiro a entrar, primeiro a sair: uma vez que um encadeamento com essa política é agendado, ele é executado até a conclusão, a menos que seja bloqueado, produz voluntariamente o controle da CPU ou um encadeamento de prioridade mais alta torna-se indisponível. Somente threads de prioridade fixa podem ter uma política de agendamento FIFO.

Round Robin: É semelhante ao esquema de round-robin do planejador do AIX Versão 3 com base em intervalos de tempo de 10 ms. Quando um encadeamento RR tem controle no final do intervalo de tempo, ele é movido para o final da fila de encadeamentos despacháveis ​​de sua prioridade. Somente threads de prioridade fixa podem ter uma política de agendamento Round Robin.

OUTROS: Esta política é definida por POSIX1003.4a como definida pela implementação. No AIX Versão 4, esta política é definida como equivalente ao RR, exceto que se aplica a encadeamentos com prioridade não fixa. O recálculo do valor de prioridade do encadeamento em execução a cada interrupção do relógio significa que um encadeamento pode perder o controle porque seu valor de prioridade subiu acima do de outro encadeamento despachável. Este é o comportamento do AIX Versão 3.

Os encadeamentos são principalmente de interesse para aplicativos que atualmente consistem em vários processos assíncronos. Esses aplicativos podem impor uma carga mais leve ao sistema se convertidos em uma estrutura multithread.

O AIX 5 implementa as seguintes políticas de planejamento: FIFO, round robin e um round round robin. A política FIFO tem três implementações diferentes: FIFO, FIFO2 e FIFO3. A política de rodízio é denominada SCHED\_RR no AIX, e o rodízio justo é chamado SCHED\_OTHER. [14]

2573/5000

Agendamento no Android

O Android é baseado no Linux e usa os mecanismos de agendamento do kernel do Linux para determinar políticas de agendamento. Isso também se aplica ao código e aos encadeamentos Java.

A política de agendamento de tempo do Linux combina prioridades estáticas e dinâmicas. Os processos podem receber uma prioridade inicial de 19 a -20 (prioridade muito baixa a muito alta). Essa prioridade garantirá que os processos de maior prioridade recebam mais tempo de CPU quando necessário. No entanto, esses níveis são dinâmicos, as tarefas de prioridade de baixo nível que não consomem seu tempo de CPU irão diminuir sua prioridade dinâmica aumentada. Esse comportamento dinâmico resulta em uma melhor capacidade de resposta geral.

Em termos de prioridades dinâmicas, é garantido que os processos de menor prioridade sempre terão uma prioridade dinâmica menor do que os processos com prioridades em tempo real.

O Android usa dois mecanismos diferentes ao agendar o kernel do Linux para executar o agendamento no nível do processo

agendamento em tempo real

O kernel Linux padrão fornece duas políticas de agendamento em tempo real, SCHED\_FIFO e SCHED\_RR. A principal política em tempo real é SCHED\_FIFO. Ele implementa um algoritmo de programação primeiro a entrar, primeiro a sair. Quando uma tarefa SCHED\_FIFO começa a ser executada, ela continua sendo executada até que voluntariamente produza o processador, bloqueie ou seja impedida por uma tarefa em tempo real de prioridade mais alta. Não possui timeslices. Todas as outras tarefas de menor prioridade não serão agendadas até que a CPU seja abandonada. Duas tarefas SCHED\_FIFO de prioridade igual não se antecipam. SCHED\_RR é semelhante a SCHED\_FIFO, exceto que essas tarefas são divididas em horários com base em sua prioridade e executadas até que esgotem sua divisão. Tarefas não em tempo real usam a política de agendamento SCHED\_NORMAL (os kernels antigos tinham uma política chamada SCHED\_OTHER).

O kernel padrão do Android está configurado para permitir o agendamento em grupo de processos em tempo real e o sistema de arquivos para controlar isso é montado em / dev / cpuctl.

O Android usa duas classes de agendamento diferentes (usando grupos de linux) bg\_non\_interactive e padrão (primeiro plano). A configuração é que bg\_non\_interactive é de baixa prioridade e pode no máximo

utilize ~ 5% da CPU (incluindo todas as tarefas em segundo plano) e primeiro plano ~ 95%. Forground significa uma Atividade ou um serviço iniciado primeiro.

Na inicialização, os serviços são executados em bg\_non\_interactive, a menos que tenham sido elevados ao grupo de agendamento em primeiro plano usando startForeground (HMI

aplicativos são sempre definidos como primeiro plano).

Fichário e prioridades

O mecanismo do fichário também propaga prioridades. Esse é o processo de ligação chamado será executado com a mesma prioridade que o chamador.

2)

O Linux usa LRU (Least Recently Used,menos usado recentemente). Veja isso. O Windows NT usa o algoritmo de relógio (LRU) para sistemas de processador único e um algoritmo de substituição de página aleatória em sistemas com vários processadores. Veja isso.

Geralmente, o FIFO ou LIFO é usado no Windows

3)

Microsoft Windows

Windows 3.xe Windows 9x

A paginação é um recurso do Microsoft Windows desde o Windows 3.0 em 1990. O Windows 3.x cria um arquivo oculto chamado 386SPART.PAR ou WIN386.SWP para uso como arquivo de troca. Geralmente é encontrado no diretório raiz, mas pode aparecer em outro lugar (normalmente no diretório WINDOWS). Seu tamanho depende da quantidade de espaço de troca do sistema (uma configuração selecionada pelo usuário em Painel de controle → Aprimorado em "Memória virtual"). Se o usuário mover ou excluir este arquivo, uma tela azul aparecerá na próxima vez que o Windows for iniciado, com a mensagem de erro "O arquivo de troca permanente está corrompido". O usuário será solicitado a escolher se deseja excluir ou não o arquivo (se ele existe ou não).

O Windows 95, Windows 98 e Windows Me usam um arquivo semelhante e as configurações estão localizadas em Painel de controle → Sistema → guia Desempenho → Memória virtual. O Windows define automaticamente o tamanho do arquivo de paginação para iniciar em 1,5 × o tamanho da memória física e expandir até 3 × memória física, se necessário. Se um usuário executar aplicativos com uso intenso de memória em um sistema com pouca memória física, é preferível definir manualmente esses tamanhos para um valor maior que o padrão.

Windows NT

O arquivo usado para paginação na família Windows NT é pagefile.sys. O local padrão do arquivo de paginação está no diretório raiz da partição em que o Windows está instalado. O Windows pode ser configurado para usar espaço livre em qualquer unidade disponível para arquivos de paginação. No entanto, é necessário que a partição de inicialização (ou seja, a unidade que contém o diretório do Windows) tenha um arquivo de paginação, se o sistema estiver configurado para gravar despejos de kernel ou de memória completa após uma Tela Azul da Morte. O Windows usa o arquivo de paginação como armazenamento temporário para o despejo de memória. Quando o sistema é reiniciado, o Windows copia o despejo de memória do arquivo de paginação em um arquivo separado e libera o espaço que foi usado no arquivo de paginação. [12]

Fragmentação

Ambox current red.svg

Esta seção precisa ser atualizada. Atualize este artigo para refletir eventos recentes ou informações disponíveis recentemente. (Julho de 2014)

Na configuração padrão do Windows, o arquivo de paginação pode expandir além de sua alocação inicial, quando necessário. Se isso acontecer gradualmente, pode se tornar bastante fragmentado, o que pode causar problemas de desempenho. [13] O conselho comum dado para evitar isso é definir um único tamanho de arquivo de página "bloqueado" para que o Windows não o expanda. No entanto, o arquivo de paginação somente se expande quando foi preenchido, o que, em sua configuração padrão, é 150% a quantidade total de memória física. [Citação necessário] Assim, a demanda total por memória virtual suportada por arquivo de paginação deve exceder 250% da memória do computador. memória física antes que o arquivo de paginação seja expandido.

A fragmentação do arquivo de paginação que ocorre quando se expande é temporária. Assim que as regiões expandidas não estiverem mais em uso (na próxima reinicialização, se não antes), as alocações adicionais de espaço em disco serão liberadas e o arquivo de paginação retornará ao seu estado original.

O bloqueio de um tamanho de arquivo de paginação pode ser problemático se um aplicativo do Windows solicitar mais memória que o tamanho total da memória física e do arquivo de paginação, levando a solicitações com falha para alocar memória que pode causar falha nos aplicativos e processos do sistema. Além disso, o arquivo de paginação raramente é lido ou gravado em ordem sequencial, portanto, a vantagem de desempenho de ter um arquivo de paginação completamente seqüencial é mínima. No entanto, um arquivo de paginação grande geralmente permite o uso de aplicativos com muita memória, sem penalidades, além de usar mais espaço em disco. Embora um arquivo de paginação fragmentado possa não ser um problema por si só, a fragmentação de um arquivo de paginação de tamanho variável criará, com o tempo, vários blocos fragmentados na unidade, causando a fragmentação de outros arquivos. Por esse motivo, um arquivo de página contíguo de tamanho fixo é melhor, desde que o tamanho alocado seja grande o suficiente para acomodar as necessidades de todos os aplicativos.

O espaço em disco necessário pode ser facilmente alocado em sistemas com especificações mais recentes (por exemplo, um sistema com 3 GB de memória com um arquivo de paginação de tamanho fixo de 6 GB em uma unidade de disco de 750 GB ou um sistema com 6 GB de memória e 16 GB arquivo de paginação de tamanho fixo e 2 TB de espaço em disco). Nos dois exemplos, o sistema está usando cerca de 0,8% do espaço em disco com o arquivo de paginação pré-estendido ao máximo.

Às vezes, também é recomendável desfragmentar o arquivo de paginação para melhorar o desempenho quando um sistema Windows está usando cronicamente muito mais memória do que sua memória física total. [Citação necessário] Esta exibição ignora o fato de que, além dos resultados temporários da expansão, o arquivo de paginação não tornar-se fragmentado ao longo do tempo. Em geral, as preocupações de desempenho relacionadas ao acesso ao arquivo de paginação são tratadas com muito mais eficiência adicionando mais memória física.

Sistemas Unix e Unix-like

Os sistemas Unix e outros sistemas operacionais semelhantes ao Unix usam o termo "swap" para descrever o ato de mover as páginas de memória entre a RAM e o disco, [citação necessário] e a região do disco em que as páginas estão armazenadas. Em alguns desses sistemas, é co

paginacao de memória **Linux em 32 bits**

### **Windows em 32 bits**

[AmigaOS](https://pt.wikipedia.org/wiki/AmigaOS)

* [Android](https://pt.wikipedia.org/wiki/Android)
* [BeOS](https://pt.wikipedia.org/wiki/BeOS)
* [BSD](https://pt.wikipedia.org/wiki/Berkeley_Software_Distribution)
* [Chrome OS](https://pt.wikipedia.org/wiki/Chrome_OS)
* [CP/M](https://pt.wikipedia.org/wiki/CP/M)
* [DOS](https://pt.wikipedia.org/wiki/DOS)
* [GNU](https://pt.wikipedia.org/wiki/GNU)
* [Haiku](https://pt.wikipedia.org/wiki/Haiku_(sistema_operacional))
* [iOS](https://pt.wikipedia.org/wiki/IOS)
* [Linux](https://pt.wikipedia.org/wiki/Linux)
* Macintosh
  + [Mac OS Classic](https://pt.wikipedia.org/wiki/Mac_OS_Classic)
  + [macOS](https://pt.wikipedia.org/wiki/MacOS)
* [MINIX](https://pt.wikipedia.org/wiki/MINIX)
* [MorphOS](https://pt.wikipedia.org/w/index.php?title=MorphOS&action=edit&redlink=1)
* [NEXTSTEP](https://pt.wikipedia.org/wiki/NEXTSTEP)
* [OpenVMS](https://pt.wikipedia.org/wiki/OpenVMS)
* [OS/2](https://pt.wikipedia.org/wiki/OS/2)
* [QNX](https://pt.wikipedia.org/wiki/QNX)
* [ReactOS](https://pt.wikipedia.org/wiki/ReactOS)
* [Solaris](https://pt.wikipedia.org/wiki/Solaris_(sistema_operacional))
* [Unix](https://pt.wikipedia.org/wiki/Unix)
* [webOS](https://pt.wikipedia.org/wiki/LG_webOS)
* [Windows](https://pt.wikipedia.org/wiki/Microsoft_Windows)
* [Xinu](https://pt.wikipedia.org/wiki/Xinu)
* [z/OS](https://pt.wikipedia.org/wiki/Z/OS)

4)

FIRST FIT

• 1o segmento é usado;

• Rápido, mas pode desperdiçar memória por fragmentação;

– NEXT FIT

• 1o segmento é usado;

• Mas na próxima alocação inicia busca do ponto que parou

anteriormente;

• SIMULAÇÕES: desempenho ligeiramente inferior;

43Gerenciamento de Memória

– BEST FIT

Procura

na lista toda e aloca o espaço que mais convém;

Menor fragmentação;

Mais lento;

– WORST FIT

Aloca

o maior espaço disponível;