FACIMP
wyden

Análise e Representação Gráfica do BCP e do Scheduler

Discentes: Guilherme Batista Da Silva
Kauã Cantanhede dos Santos
Rennan Almeida Sousa
Reney Lima Chave

Docente: Thalles Canela

2024

Imperatriz-Ma

O **Bloco de Controle de Processos (BCP)** é uma estrutura fundamental usada pelo sistema operacional para gerenciar cada processo em execução no sistema. Ele contém todas as informações necessárias para que o sistema operacional controle e supervisione o ciclo de vida de um processo, desde a criação até a sua finalização.

Informações Armazenadas no BCP:

1. **Identificação do Processo (PID):** Um identificador exclusivo que diferencia cada processo em execução.
2. **Estado do Processo:** Registra o estado atual do processo, como Pronto, Executando, Bloqueado ou Terminado, permitindo que o sistema saiba quando um processo está apto a ser executado ou precisa ser interrompido.
3. **Contador de Programa:** Armazena o endereço da próxima instrução a ser executada, essencial para retomar a execução de um processo após interrupções ou trocas de contexto.
4. **Registradores da CPU:** Contém os valores dos registradores que o processo estava utilizando, garantindo que ele possa ser retomado exatamente do ponto onde foi interrompido.
5. **Informações de Gerenciamento de Memória:** Inclui detalhes sobre a memória virtual e física alocada ao processo, como limites de memória e tabelas de páginas, essenciais para o gerenciamento de memória.
6. **Informações de Entrada/Saída (I/O):** Detalha dispositivos de entrada e saída que o processo está utilizando, incluindo arquivos abertos e buffers, permitindo o gerenciamento das operações de I/O.
7. **Informações de Contabilidade:** Armazena dados como tempo de CPU utilizado e outras métricas de desempenho, que são úteis para monitorar o uso de recursos do sistema.

Papel do BCP no Gerenciamento de Processos:

O BCP desempenha um papel central no gerenciamento de processos em um sistema operacional. Ele permite:

- **Troca de Contexto:** Quando o sistema operacional interrompe um processo para que outro possa ser executado, ele salva o estado do processo atual no BCP e carrega o estado do próximo processo a ser executado. Isso garante que cada processo possa ser retomado de forma precisa e eficiente.
- **Escalonamento de Processos:** As informações contidas no BCP ajudam o sistema operacional a decidir qual processo deve ser executado em seguida, garantindo uma distribuição justa e eficiente do tempo de CPU entre os processos.
- **Gerenciamento de Recursos:** O BCP contém dados cruciais sobre o uso de memória e dispositivos de I/O, permitindo ao sistema operacional alocar e liberar recursos conforme necessário, o que é vital para o desempenho do sistema como um todo.

Os principais campos de um **Bloco de Controle de Processos (BCP)** e sua importância no gerenciamento do sistema operacional são:

Identificação do Processo (PID):

- **Importância:** Fornece um identificador único para cada processo, permitindo ao sistema operacional diferenciá-los e gerenciá-los individualmente. Cada processo em execução em um sistema operacional recebe um PID exclusivo.
- Esse número é essencial para que o sistema possa diferenciar e gerenciar cada processo individualmente.
- Reutilização de PIDs:
- Os PIDs são números finitos e, portanto, são reutilizados ao longo do tempo.
- Quando um processo termina, seu PID é liberado e pode ser atribuído a um novo processo que se inicia posteriormente.
- Gerenciamento de Processos:
- O PID é utilizado em várias chamadas de função do sistema operacional para manipular processos.
- Exemplos incluem ajustar a prioridade de um processo, enviar sinais (como SIGKILL para terminar um processo no Unix/Linux), ou obter informações sobre o estado do processo.
- Ferramentas de Visualização:
- Em sistemas Windows, você pode visualizar os PIDs usando o Gerenciador de Tarefas, onde cada processo listado tem um PID associado.
- Comandos como tasklist no Prompt de Comando ou Get-Process no PowerShell também permitem visualizar e gerenciar processos pelo PID.
- Em sistemas Unix/Linux, comandos como ps, top e htop são usados para listar processos e seus PIDs.

Estado do Processo:

- **Importância:** Indica o estado atual do processo (Pronto, Executando, Bloqueado, etc.), essencial para o gerenciamento do ciclo de vida do processo e para decisões de escalonamento.

Contador de Programa (Program Counter):

- **Importância:** Armazena o endereço da próxima instrução a ser executada, crucial para retomar a execução do processo após interrupções. O contador de programa (PC, do inglês Program Counter) é um registrador crucial na Unidade Central de Processamento (CPU). Ele mantém o controle da posição atual na sequência de execução de um programa, armazenando o endereço da próxima instrução a ser executada.
- Função do Contador de Programa
- Sequenciamento de Instruções: O contador de programa garante que as instruções sejam executadas na ordem correta. Ele é automaticamente incrementado após cada ciclo de instrução, apontando para a próxima instrução na memória.
- Execução Sequencial: Normalmente, as instruções são executadas de forma sequencial. O PC é incrementado em 1 (ou pelo tamanho da instrução) após

cada ciclo de instrução, garantindo que a CPU execute as instruções na ordem correta.

- Instruções de Salto: Em casos de instruções de salto (como loops ou chamadas de função), o valor do contador de programa é alterado para o endereço da instrução de destino, permitindo a execução não sequencial.
- Ciclo de Instrução
- O ciclo de instrução da CPU pode ser dividido em três etapas principais:
- Busca (Fetch): A CPU usa o valor do contador de programa para buscar a próxima instrução da memória.
- Decodificação (Decode): A instrução buscada é decodificada para determinar quais ações a CPU deve realizar.
- Execução (Execute): A CPU executa a instrução decodificada.
- Importância do Contador de Programa
- Controle de Fluxo: O contador de programa é essencial para o controle de fluxo de um programa, garantindo que as instruções sejam executadas na ordem correta.
- Eficiência: Ao manter a sequência lógica das instruções, o PC ajuda a minimizar atrasos e maximizar a eficiência da execução do programa.
- Flexibilidade: Permite a execução de instruções de salto, chamadas de função e outras operações que alteram o fluxo normal de execução.
- Inicialização
- No início da execução de um programa, o contador de programa é geralmente inicializado com o endereço da primeira instrução do programa. Isso garante que a CPU comece a execução do ponto correto.
- Resumo
- O contador de programa é um componente fundamental da CPU, responsável por manter a sequência de execução das instruções de um programa. Ele garante que as instruções sejam executadas na ordem correta e permite a execução eficiente e controlada de programas.

Registradores da CPU:

- **Importância:** Contém os valores dos registradores utilizados pelo processo, permitindo a restauração do estado exato do processo durante a troca de contexto. A unidade central de processamento ou CPU (Central Processing Unit), também conhecido como processador, é a parte de um sistema computacional, que realiza as instruções de um programa de computador, para executar a aritmética básica, lógica, e a entrada e saída de dados. A CPU tem papel parecido ao cérebro no computador. O termo vem sendo usado desde o início de 1960. A forma, desenho e implementação mudaram drasticamente desde os primeiros exemplos, porém o seu funcionamento fundamental permanece o mesmo.
- Unidade de Controle (UC)
- A Unidade de Controle (UC), em inglês: control unit (CU), é responsável por gerar todos os sinais que controlam as operações no exterior do CPU, e ainda por dar todas as instruções para o correto funcionamento interno do CPU; a apoiá-la/o terá a colaboração de uma outra estrutura/actor (o decodificador de instruções). É a Unidade de Controle, sendo esta uma UTM,

a responsável pela abstração das diversas máquinas virtualizadas dentro do sistema computacional, ou seja, ela é a responsável pela exata “expressão de software” que rodam internamente ao PC em determinado momento. Sem ela o computador seria uma máquina de função única, não sendo possível o processador funcionar com propósito geral.

- Unidade lógica e aritmética (UC) ou Arithmetic Logic Unit (ALU)
- A unidade lógica e aritmética (ULA) ou em inglês Arithmetic Logic Unit (ALU) é um circuito digital que realiza operações lógicas e aritméticas. A ULA é uma peça fundamental da unidade central de processamento (CPU), e até dos mais simples microprocessadores. É na verdade, uma “grande calculadora eletrônica” do tipo desenvolvido durante a II Guerra Mundial, e sua tecnologia já estava disponível quando os primeiros computadores modernos foram construídos.

Gerenciamento de Memória:

- **Importância:** Inclui informações sobre a memória alocada ao processo, fundamental para a alocação, rastreamento e liberação de memória. Cada local na memória tem um endereço único, semelhante a um endereço residencial, que permite ao processador localizar e acessar dados específicos1.
- Endereçamento por Byte:
- Em computadores modernos, cada endereço representa um byte distinto de armazenamento. Dados maiores que um byte ocupam uma sequência de bytes consecutivos1.
- Modos de Endereçamento:
- Existem vários modos de endereçamento, como direto, indireto, indexado, entre outros. Cada modo determina como o endereço efetivo do operando é calculado e acessado1.
- Memória Virtual e Física:
- A memória virtual permite que programas usem mais memória do que a fisicamente disponível, dividindo a memória em páginas que são movidas entre o disco e a RAM conforme necessário1.
- Capacidade de Endereçamento:
- A capacidade de endereçamento depende da arquitetura do sistema. Por exemplo, sistemas de 32 bits podem endereçar até 4 GB de memória, enquanto sistemas de 64 bits podem endereçar até 18 exabytes1.

Um endereço de memória identifica uma locação física na memória de um computador de forma similar ao de um endereço residencial em uma cidade. O endereço aponta para o local onde os dados estão armazenados, da mesma forma como o seu endereço indica onde você reside. Na analogia do endereço residencial, o espaço de endereçamento seria uma área de moradias, tais como um bairro, vila, cidade ou país. Dois endereços podem ser numericamente os mesmos, mas se referirem a locais diferentes se pertencem a espaços de endereçamento diferentes. É o mesmo que você morar na "Rua Central, 32", enquanto outra pessoa reside na "Rua Central, 32" numa outra cidade qualquer.

Informações de I/O:

- **Importância:** Detalha os dispositivos de entrada e saída utilizados pelo processo, garantindo o gerenciamento correto das operações de I/O. Chamamos de dispositivos de entrada e saída os responsáveis por incorporar e extrair informação de um sistema de computador. Eles se adequam dentro da denominada Arquitetura de Von Neumann, que nos informa as principais partes de um computador. Estes dispositivos evoluíram bastante com o tempo, existindo na atualidade muitas variantes que no inicio da informática pareciam impossíveis.
- Todos os periféricos trabalham através de interrupções que fazem com que os processos executados sejam suspendidos temporariamente. Assim, os dispositivos de entrada e saída enviam interrupções à CPU através de um controlador que pode estar conjunto ao próprio processador, além de habilitar ou desabilitar estes pedidos.

Escalonamento de Processos

O escalonamento de processos ou agendador de tarefas é uma atividade organizacional feita pelo escalonador (scheduler) da CPU ou de um sistema distribuído, possibilitando executar os processos mais viáveis e concorrentes, priorizando determinados tipos de processos, como os de I/O Bound e os CPU Bound.

Escalonador de Processo: É um componente do sistema operacional responsável por decidir qual processo será executado pela CPU em um dado momento. Ele é crucial durante a mudança de contexto, que é quando o sistema troca a execução de um processo por outro.

Algoritmos de Escalonamento: São métodos utilizados pelo escalonador para determinar a ordem e o tempo de execução dos processos.

Escalonador de Processos de 2 Níveis: Este tipo de escalonador utiliza dois níveis de decisão:

Primeiro Nível: Escolhe o processo com maior prioridade e menor tempo de execução para ser colocado na memória principal (RAM).

Segundo Nível: Os processos que não são escolhidos ficam alocados no disco (memória secundária), aguardando sua vez de serem executados.

Objetivo: O objetivo principal é maximizar a utilização da CPU, evitando que ela fique ociosa. Ao sempre ter um processo pronto para ser executado na memória principal, o sistema garante que a CPU esteja constantemente em uso.

Scheduler

Faz parte do gerenciamento de processos de um sistema operacional que é responsável por gerenciar as execução dos processos na ***CPU(Central Process Unit),*** onde vai determinar quais/qual processos devem ser executados em determinado momento, garantindo a eficiência do uso da CPU durante o uso do usuário e a realização de multitarefas.

Ciclo de Vida de um Processo

Significa que um processo no sistema operacional passa por diferentes estados durante seu ciclo de vida, sendo eles:

- **Novo**: O processo foi criado e está esperando para ser admitido pelo sistema operacional.
- **Pronto**: O processo está na fila de prontos e está aguardando a sua vez de ser executado na CPU.
- **Executando**: O processo está em execução na CPU.
- **Bloqueado**: O processo está aguardando por um evento externo, como a finalização de uma operação de I/O.
- **Finalizado**: O processo terminou sua execução e será removido da memória.

Com isso, podemos dizer que o escalonador(***scheduler***) tem como sua responsabilidade mover os processos entre os estados de **pronto** e **executando**, e decidir qual processo deve deixar a CPU (não tendo interrupção) e qual deve começar a executar.

Algoritmos de Escalonamento

Quanto a sua forma de escalonamento, vai utilizar diferentes algoritmos para decidir qual processo deve ser executado em cada momento, com base na sua importância. Entre os principais algoritmos, estão:

FIFO (First In, First Out): É um algoritmo de interação direta que trabalha com filas de prontos, onde após a conclusão do processo ele é adicionado na fila. Não tendo prioridades, ou seja, um processo executa até que termine ou seja bloqueado, só após isso que um novo processo é chamado. Esse método implantou que o primeiro que entrou é o primeiro a sair, apesar de levar a ineficiências, como o problema de *convoy effect* (um processo longo bloqueia a execução dos outros processos).

Round Robin (RR): Um dos algoritmos mais comuns em sistemas de tempo compartilhado. Cada processo na fila de prontos recebe um *quantum* de tempo (um tempo limite de execução). Onde um processo pode não ser concluído em um tempo específico ele é finalizado e jogado para a fila de prontos e com isso é permitido a execução de outros processos. Isso garante uma divisão de tempo mais justa entre

os processos, mas pode não ser ideal para processos que necessitam de mais tempo para completar tarefas.

Prioridade: Nesse esquema, cada processo é atribuído a uma prioridade, e o escalonador sempre escolhe para execução o processo com a maior prioridade. As prioridades podem ser estáticas (definidas no início e não mudam) ou dinâmicas (podem mudar durante a execução, por exemplo, para evitar a inanição de processos com baixa prioridade). Processos de baixa prioridade podem ser interrompidos por processos de alta prioridade.

Tomada de Decisão do Escalonador

A decisão de qual processo deve ser executado é baseada no algoritmo de escalonamento implementado. O escalonador avalia a fila de processos prontos, aplica o algoritmo escolhido e decide qual processo deve ser colocado na CPU.

Fatores como o tempo de chegada, tempo de execução restante, prioridade e histórico de execução podem influenciar essa decisão.

Interação com o BCP (Bloco de Controle de Processo)

O **Bloco de Controle de Processo** (BCP) é uma estrutura de dados que armazena todas as informações necessárias para a gestão de um processo, como o estado atual do processo, valores dos registradores da CPU, ponteiros de pilha, limites de memória, entre outros.

Quando ocorre uma troca de contexto, o escalonador salva o estado do processo atual em seu BCP, e então carrega o estado do próximo processo a ser executado. Essa operação é crucial para garantir que, quando o processo retomar a execução, ele continue exatamente do ponto onde parou, com todos os seus dados e contexto intactos.

Troca de Processos na CPU

A **troca de processos** na CPU, ou troca de contexto, ocorre quando o escalonador decide que um processo deve ser interrompido para dar lugar a outro. Essa troca envolve salvar o estado do processo que está saindo da CPU e carregar o estado do próximo processo a ser executado, garantindo a continuidade correta da execução de ambos os processos.

Esses mecanismos são fundamentais para o funcionamento de sistemas multitarefa, permitindo que vários processos pareçam estar executando simultaneamente em um sistema de tempo compartilhado.

Diferença entre Sistemas Operacionais:

- **Unix/Linux** prioriza equidade e escalabilidade, adequado para uma ampla variedade de aplicações e ambientes, desde desktops até servidores de alta performance.
- **Windows** busca equilibrar responsividade e desempenho, suportando uma diversidade de aplicações e cenários de uso, com mecanismos dinâmicos para ajustar prioridades conforme necessário.
- **iOS** foca na responsividade e eficiência energética, crítico para dispositivos móveis, utilizando tecnologias como QoS e GCD para otimizar o desempenho e a experiência do usuário enquanto preserva a bateria.

Conclusão Final

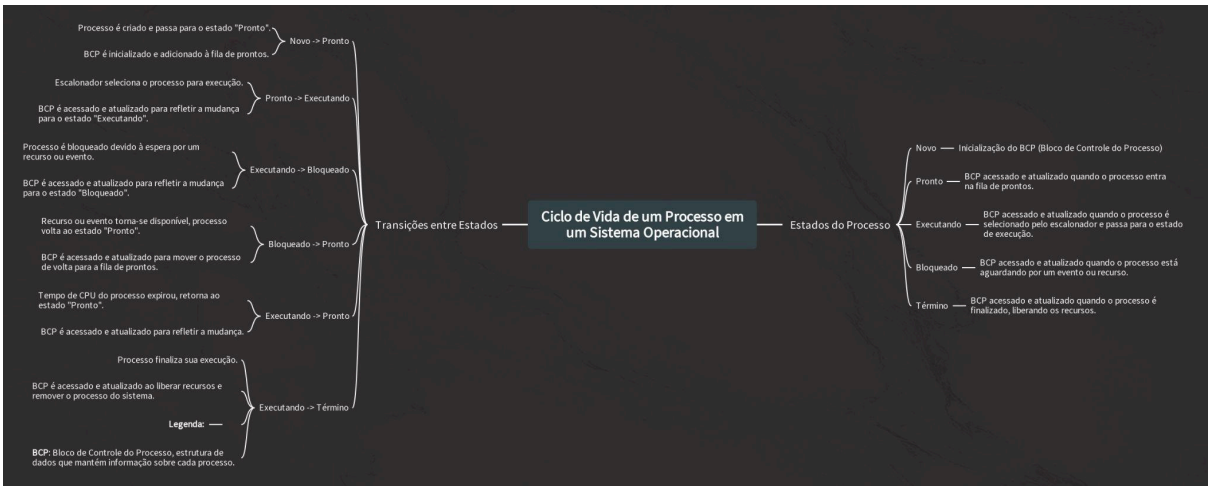
O Bloco de Controle de Processos (BCP) é uma estrutura fundamental no gerenciamento de processos em um sistema operacional. Ele armazena informações cruciais sobre o estado, identificador, registradores da CPU, e recursos alocados a cada processo, permitindo que o sistema operacional controle e organize a execução dos processos de forma eficiente e segura.

O *scheduler* é o componente responsável por decidir qual processo será executado pela CPU em um dado momento. Utilizando algoritmos como FIFO, Round Robin e Prioridade, ele garante uma utilização eficiente da CPU, maximizando o desempenho do sistema e mantendo a responsividade.

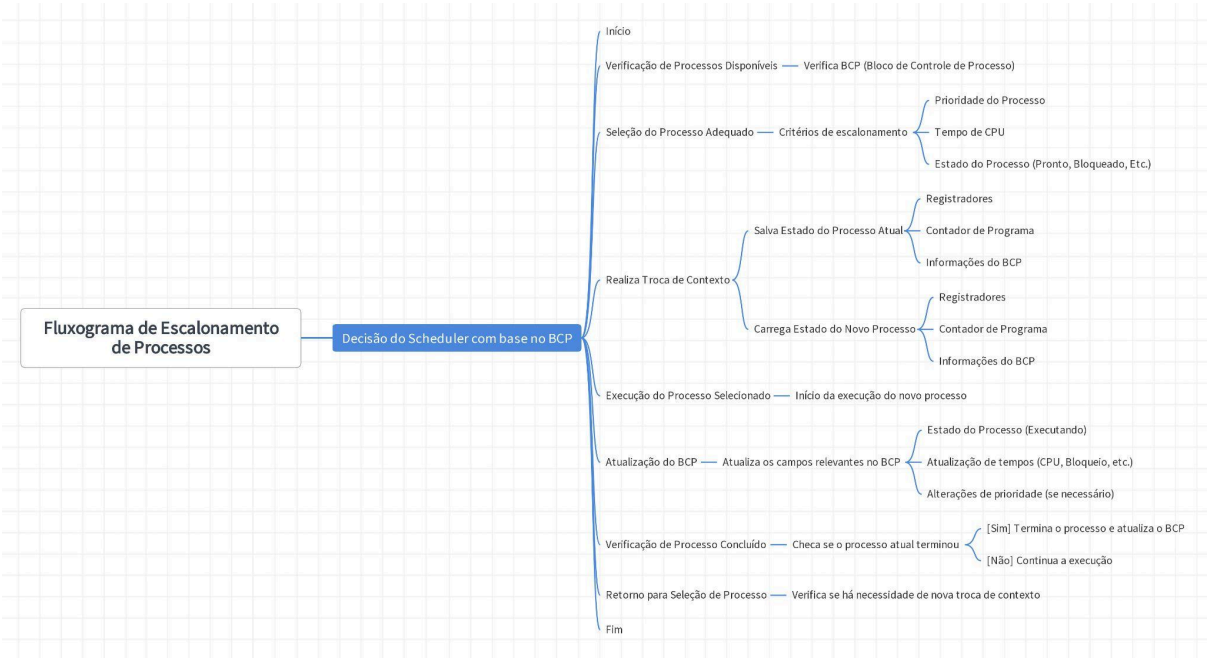
O BCP e o *scheduler* trabalham em conjunto, onde o *scheduler* utiliza as informações do BCP durante a troca de contexto para assegurar que os processos sejam interrompidos e retomados corretamente. Embora Unix/Linux, Windows e iOS implementem esses mecanismos de maneira distinta para atender às suas necessidades específicas, todos dependem da interação entre o BCP e o *scheduler* para garantir um gerenciamento de processos eficaz e otimizado.

Fluxogramas

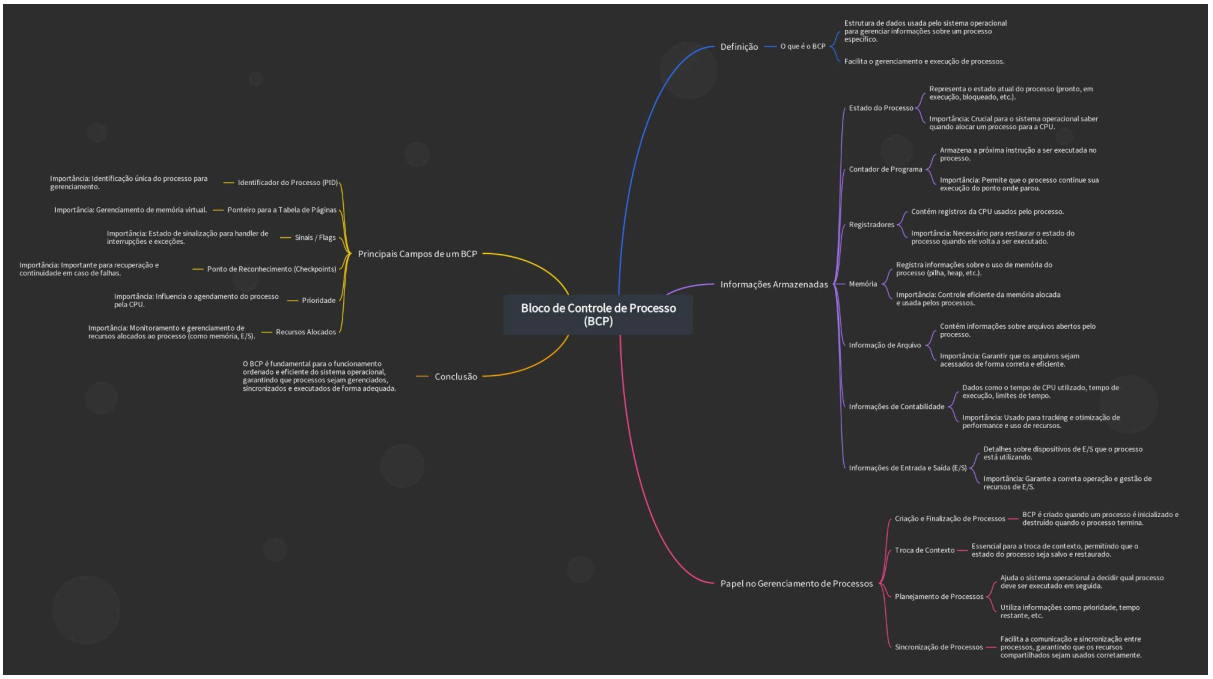
Fluxo do Ciclo de Vida de um Processo



Fluxograma da Interação entre BCP e Scheduler



Fluxograma BCP



Fluxograma Scheduler