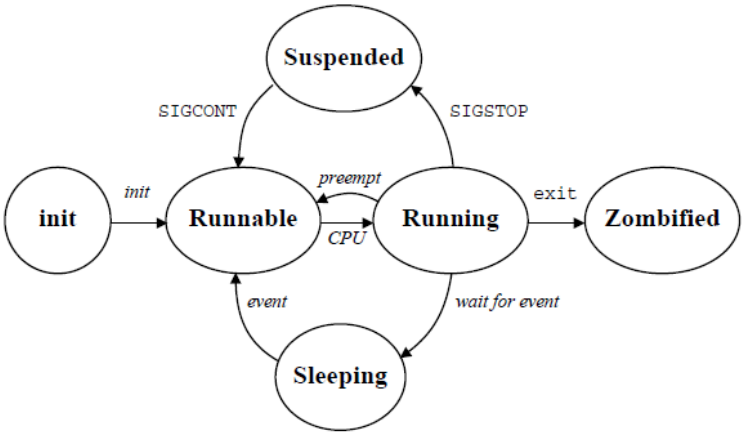
# 35. Plánování a synchronizace procesů

## Proces

Proces je spuštěný počítačový program, uložený v operační paměti počítače jako sled strojových instrukcí vykonávaných procesorem. Proces obsahuje současně kód programu a dynamicky se měnící data zpracovávaná procesorem.
Proces je identifikován PID, stavem plánování, řídícím programem, obsahem registrů, daty a zásobníkem, využitím dalších zdrojů.

Stavy procesu:

| stav | význam |
|---|---|
| Vytvořený | ještě neinicializovaný |
| Připravený | mohl by běžet, ale nemá CPU |
| Běžící | používá CPU |
| Mátoha | po `exit`, rodič ještě nepřevzal exit-code |
| Čekající | čeká na událost (např. dokončení `read`) |
| Odložený | „zmrazený“ signálem `SIGSTOP` |




Části procesu v paměti v Unixu:

- Uživatelský adresový prostor
- Uživatelská oblast
- Záznam v tabulce procesů
- tabulka paměťových regionů procesi

# Multitaskíng

Pokud to umožňuje operační systém, může se proces skládat z několika vláken provádějících instrukce paralelně (tzv. Multitasking). Pokud běží více procesů, než je v sytému procesorů, je nutné procesy plánovat a přidělovat jim procesorový čas. Dnešní OS jsou optimalizovány pro multitasking a plánování procesů.

Existují 3 druhy plánování procesů. Liší se v délce časového intervalu mezi plánováním.

## Dlouhodobé plánování ( plánování úloh - job scheduling )

Výběr úlohy, která se bude provádět. Má smysl zejména u dávkového zpracování, kvůli plánování vstup/výstupních operací. V normálních procesorech se nevyužívá, je používán zejména u real-time operačních systémů.

## Střednědobé plánování

Je využíván u systému s virtuální pamětí a vybírá data procesu, které mohou být odsunuta z operační paměti na pevný disk. Tohle téma tzn výpadky stránek, tabulky tabulek tabulek stránek a podobný shit podrobněji rozeberu v otázce 34 - správa paměti.

## Krátkodobé plánování

Tohle už je plánování procesoru - výběr procesu, který poběží.

## Plánování procesů

Plánování procesů má na starosti jádro operačního systému. Vybírá, kterému procesu bude přidělen procesorový čas, podle různých faktorů (např priorita procesu, nebo pořadí ve frontě). Při plánování procesů je nutné dodržovat určitou míru spravedlnosti. Pokud není plánovač spravedlivý může dojít k tzv. vyhladovění procesu (starvation). To znamená, že procesu není přidělen procesorový čas, protože má např malou prioritu.

Plánování probíhá několikrát za sekundu a je spuštěno při následujících událostech:

1. Pokud některý běžící proces přejde do stavu blokovaný
2. Pokud některý proces skončí
3. Pokud je běžící proces převeden do stavu připravený
4. Pokud je některý proces převeden ze stavu blokovaný do stavu připravený - používá se pouze u RTOS systémů.

## Preemptivnost

Preemptivnost je schopnost plánovače a procesoru pozastavit běh procesu bez toho, aby byla vyžadována jeho spolupráce. Po přerušení běhu procesu je přepnut kontext. Pokud může proces zabránit přerušení, jedná se o nepreemptivní plánování (nazývané taky kooperativní multitasking).
U nepreemptivního plánovače dochází k plánování pouze u bodů 1 a 2 a plánovač čeká na to, až se proces ukončí, nebo využije všechen přidělený CPU čas. Plánovač v tomto případě tedy nemá absolutní kontrolu nad počítačem.
U preemptivního plánovače dochází k plánování v bodech 1 - 3. Plánovač v tomto případě drží plnou kontrolu nad počítačem.

## Tabulka popisu procesů (PCB)

PCB je datová struktura v jádře OS, jsou v ní uchovávána data potřebná k běhu procesu. Každý proces má svoji PCB, která je využívána při změně kontextu k uložení stavu procesu.
Obsah PCB:

- PID
- stav plánování procesu
- obsah registrů
- plánovací informace
- informace o správě paměti
- informace spojené s účtováním (kolik procesorového času proces dostal, atd)
- využití IO zdrojů

## Změna kontextu

Aby měl uživatel dojem, že všechny aplikace probíhají paralelně, dochází k přepínání kontextu cca 100-1000x za sekundu.
Při změně kontextu je nutné, aby proces po opětovném spuštění pokračoval ze stejného místa ve kterém byl přerušen.
Probíhají vždy 2 operace : uložení kontextu jednoho procesu a obnovení kontextu dalšího procesu.

## Strategie plánování procesoru

Plánovač procesů se může zaměřovat na různá kritéria, podle kterých přidělí procesorový čas.
**spravedlnost** – každý proces dostane spravedlivý díl času procesoru
**efektivita** – udržovat maximální vytížení procesoru, případně jiné části systému
**čas odezvy** – minimalizovat dobu odezvy pro interaktivní uživatele

**doba obrátky** – minimalizovat dobu zpracování každé dávkové úlohy
**průchodnost** – maximalizovat množství úloh zpracovaných za jednotku času

### Příklady strategií

- FIFO
  - řadí procesy do fronty, ve chvíli, kdy proces skončí, začne se provádět další proces
  - nejjednodušší algoritmus, nízká režie přepínání,
  - funguje bez priorit => nedochází k vyhladovění
  - může docházet k dlouhým prodlevám a dlouhým odezvám, při zpracovávání náročných procesů.
- STR - Shortest Time Remaining
  - plánovač dává přednost procesům s nižší odhadovanou dobou zpracování (je třeba tuto hodnotu nějak zjišťovat)
  - je preemptivní => pokud příjde do plánovače proces s nižší dobou zpracování než má současný proces, je přepnut kontext na nový proces.
  - Dochází často k řazení fronty a je třeba odhadovat čas zpracování, míra režie je tedy celkem vysoká.
  - může dojít k vyhladovění
- Fixed priority pre-emptive scheduling
  - každý proces má neměnnou prioritu
  - plánovač řadí procesy podle priority
  - proces s vyšší prioritou může vyhodit proces s nižší prioritou z obsluhy
  - může dojít k vyhladovění
- Round-robin scheduling
  - Procesy řadí do fronty a postupně každému přiděluje určitý procesorový čas, po vyčerpání času proces vrátí na konec fronty
  - průměrná doba odezvy je vysoká, obzvlášť při velkém množství procesů
  - nemůže dojít k vyhladovění
- LIFO
  - zásobníková fronta, může dojit k vyhladovění
- O(1)

# Synchronizace procesů

Pokud přistupuje ke společným zdrojům současně více procesů, může dojit k nekonzistenci dat a zpráv. Toto je třeba řešit pomocí synchronizace procesů.

## Kritická sekce

Sekce kódu, ve které proces přistupuje ke sdíleným datům a znemožní ostatním procesům k datům přistupovat.
Je třeba řešit dostupnost kritické sekce pro ostatní procesy, aby nedošlo k vyhladovění. Zároveň je třeba zajistit, aby v jednu chvíli mohl s datz manipulovat pouze jeden proces

Pro 2 procesy lze použít tzv petersonův algoritmus. Pro větší množství Bakery algoritmus.

TestAndSet

```
bool TestAndSet(bool &target) {
  bool rv = target;
  target = true;
  return rv;
}
```

```
bool lock = false;

  // ...
  while (TestAndSet(lock)) ;
  // critical section
  lock = false;
  // ...
```

Swap:

```
void Swap(bool &a, bool &b) {
  bool temp = a;
  a = b;
  b = temp;
}
```

```
bool lock = false;

  // ...
  bool key = true;
  while (key == true)
    Swap(lock,key);
  // critical section
  lock = false;
  // ...
```

## Semafory

Semafor je celočíselná hodnota, přístupná pouze dvěmi operacemi - lock a unlock (musejí být atomické operace). Může také obsahovat frontu procesů čekajících na zamčený semafor.
Hodnota čísla: 1 volný, 0,-1,-2,... zamčený
U semaforů není potřeba aktivní čekání. Pokud je semafor zamčený a některý proces se ho pokusí zamknout sám, je zařazen do fronty.

### Monitory

Vysokoúrovňový synchronizační prostředek. Zapouzdřuje data a má definované operace. Dovolí pouze jednomu procesu operovat s daty. Monitory je možné implementovat pomocí semaforů.

### Synchronizační problémy

Producent - konzument
čtenáři - písaři
večeřící filozofové

#### deadlock - uváznutí

Nutné podmínky uvá¡znutí (Coffmanovy podmínky):
1. vzájemné vyloučení při používání prostředků,
2. vlastnictví alespoň jednoho zdroje a čekání na další,
3. prostředky vrací proces pouze po dokončení jejich využití,
4. cyklická závislost na sebe čekajících procesů.

#### hladovění, stárnutí - starvation

#### Live-lock

Zvláštní případ hladovění. Všechny procesy běží, ale vykonávají jen část kódu a čekají na uvolnění zdrojů obsazených jiným procesem.

## Transakce

Logická jednotka provádění programu operujícího s daty v databázi. Je tvořena posloupností databázových operací.

Model transakce ACID:

- Atomičnost - buď se provede všechno nebo se neprovede nic
- Konzistence - po provedení transakce zůstává databáze konzistentní
- Izolace - souběžné provádění transakcí by mělo mít stejný výsledek jako sekvenční
- trvalost - změny v databázi jsou trvalé

## Pojmy

**Multitasking** - více procesů sdílí výpočetní čas na procesoru
**Přepínání kontextu** - přepínání řízení mezi procesy
**Kontext** - stav procesoru ( obsahu registrů ), a stavu ostatních zařízení (obsah cache atd)
**Přerušení** - přepnutí kontextu signálem z jiného zařízení
**Preempce** - v informatice přerušení právě vykonávaného procesu (úlohy) bez toho, aby byla vyžadována jeho spolupráce. (Wiki)