# Plánování, rozrhování

# Plánování

- Základní zkoumaná úloha - je dána sekvence úloh, které mají být zpracovány na dostupných strojích.
- V nejjednodušší verzi je úloha dána časem běhu a musí na tento čas být přidělena na nějaký stroj
- V jiných variantách se uvažují další omezení specifikující, který plán zpracování úloh je povolen
- Snažíme se úlohy naplánovat co nejefektivněji, nejčastěji to znamená, aby byly za co nejkratší čas všechny hotové
- Budeme se zabývat tzv. on-line verzí, kdy algoritmus nemá k dispozici najednou celý seznam úloh
- Mnoho plánovacích úloh je NP-úplných
- Po důkazech NP-úplnosti se výzkum zaměřil na aproximační algoritmy

# Definice apod.

- Počet strojů označujeme $m$, počet úloh $n$
- Každá úloha je charakterizovaná dobou běhu $t$ a případně dalšími parametry podle typu úlohy
- Na každý stroj v každém okamžiku může být přidělena pouze jedna úloha
- Všechny plánovací úlohy směřují k minimalizaci nějaké funkce (míra výkonu)
- Výkon on-line algoritmů měříme prostřednictvím poměru kvality nalezeného řešení ku optimálnímu řešení (bereme hodnotu, že pro všechny možné instance je poměr menší nebo roven této hodnotě)
- Minimální celkový čas pro dokončení všech úloh označujeme $T_{opt}$

# Různá paradigmata

- Plánování jeden po druhém (one by one) - úlohy jsou v seznamu, algoritmus je dostává po jedné a musí každou přiřadit některému stroji, aniž by se mohl podívat na následující položky seznamu úloh (ale může si pamatovat předchozí úlohy). Je možné úlohy plánovat na jakýkoliv časový slot, ale po naplánování to již nelze změnit.
- Neznámé časy běhu - než úloha skončí, není známa doba jejího běhu. Často je zde možnost restartu nebo preempce, algoritmus má přístup k již přiřazeným úlohám a může přiřazení měnit.
- Intervalové plánování - každá úloha má přesně daný interval, kdy může být zpracována. Úlohy, které není možné zpracovat v tomto intervalu, je možné odmítnout.

# Kriteriální funkce

- Nejčastěji celková doba zpracování
- Když je možné úlohy odmítat - doba zpracování + pokuta za odmítnutí
- Součet časů, kdy jsou jednotlivé úlohy dokončené
- Součet časů, který stráví jednotlivé úlohy v systému (od jejich příchodu do dokončení)
- Součet časů, které stráví úlohy čekáním na zpracování
- Uvažují se i vážené varianty - hodnoty různých úloh ovlivňují funkci s různou váhou
- Při povolené preempci můžeme uvažovat i počet preempcí úloh

# List scheduling

- Základní algoritmus, studovaný již v roce 1966
- Úlohy jsou uspořádány v seznamu (sekvenci)
- Kdykoliv je nějaký stroj volný, naplánujeme na něj první úlohu ze seznamu, která je k dispozici (tedy není ještě naplánována, aktuální čas je větší než release čas úlohy a všechny předcházející úlohy v precedenčním grafu jsou skončeny)
- Je možné jej použít v různých paradigmatech
- Výkonostní poměr tohoto algoritmu je $2 - 1/m$

# Plánování jeden po druhém

- Algoritmus musí úlohu ze seznamu přiřadit na stroj bez znalosti následujících úloh a přiřazení již pak nemůže změnit
- Obvykle se neuvažují precedenční omezení a release časy
- Většinou se ani neurčuje časový slot, jen stroj
- Pro $m = 2$ a $m = 3$ je prokazatelně nejlepší list scheduling (LS)
- Pro větší počty strojů jsou i lepší algoritmy
- Hlavní problém LS - kdy stroje rovnoměrně zatíží a potom přijde dlouhá úloha (může být až skoro 2x delší plán, než by mohl být)
- Řešení - nějaká mírná disbalance - některé stroje nechává mírně zatížené aby na ně mohl naplánovat dlouhé úlohy
- Pro počet strojů nad 4 získáme poměr lepší než LS, když se úloha přidělí vždy na jeden ze dvou nejméně vytížených strojů. Pro velký počet strojů se ale také poměr blíží 2 (jako u LS)
- Aby se poměr neblížil 2 pro velký počet strojů, je nutné jich nechat nevytížených nějaký konstantní počet - navrženo několik algoritmů

# Randomizované algoritmy

- Mnohem méně prozkoumány než deterministické
- Uvažujeme poměr střední doby zpracování úloh ku optimální době
- Pro dva stroje algoritmus s poměrem $4/3$, což je nejlepší možné:
  - Když to jde, naplánuje úlohu náhodně tž. střední doba délky plánu je $2/3$ součtu časů zpracování úloh
  - Když to nejde, naplánuje úlohu na méně zatížený stroj
- I pro $m = 3, 4, 5$ je znám nejlepší možný algoritmus, pro $m = 6, 7$ je lepší než deterministický, ale není dokázáno, že by byl nejlepší možný
- Vždy dává úlohu na jeden ze dvou nejméně vytížených strojů. Pro velké počty strojů se výkonostní poměr ale blíží také 2

# Plánování s preempcí

- Úloha může být přiřazena více strojům, ale časové sloty musí být disjunktní
- Přiřazení musí být určeno hned, když úloha dorazí (stále uvažujeme paradigma jeden po druhém)
- Pro off-line je řešení lehké - optimální celkový čas je maximum z maximální doby běhu úlohy a sumy dob běhů všech úloh dělené $m$
- Deterministické i randomizované verze jsou plně vyřešeny

# Plánování s odmítáním

- Za pokutu je možné úlohu odmítnout
- Každá úloha má kromě dané doby výpočtu určenou i pokutu
- Kriteriální funkce je celková doba na zpracování všech úloh plus součet pokut za odmítnuté úlohy
- Uvažujeme stroje se stejnou rychlostí a nebereme do úvahy žádná další omezení (precedenci apod.)
- Hledá se tedy rovnováha mezi zařazením úlohy a tím prodloužením doby výpočtu a pokutou za odmítnutí úlohy
- Na začátku je často výhodné úlohy s malou pokutou odmítnout, ale později může být výhodné je dodatečně zařadit a zpracovat (vytíží se třeba volný stroj a sníží se pokuta)

# Plánování s odmítáním

- Deterministické algoritmy bez preempce - je dokázáno, že po efektivním rozhodnutí o odmítnutí úlohy již na přidělování úloh na stroje stačí LS alg.
- Optimální řešení je známo pro dva stroje a pro velký počet strojů
- Pro dva stroje:
  - úloha je odmítnuta, když $t \geq mp$ ($p$ je pokuta)
  - úloha je odmítnuta, když $t \geq \phi(P + p)$ ($P$ je celková dosud zaplacená pokuta, $\phi$ je zlatý řez)
  - akceptované úlohy naplánuje prostřednictvím LS
- Randomizované - varianty deterministických, náhodně se volí prahy pro odmítnutí úlohy

# Různé rychlosti strojů

- Uvažují se 3 varianty:
  - Related - stroje mají různé rychlosti, ale stejné pro všechny úlohy
  - Unrelated - vektor rychlostí strojů se liší pro různé úlohy
  - Restricted - úloha má určeno, na kterých strojích může být zpracována
- Doubling strategy - konstantní poměr pro related verzi:
  - Odhadneme celkový čas každou úlohu dáme na nejpomalejší stroj tak, aby nebyl odhad překročen
  - Není-li naplánování možné, zdvojnásobíme odhad
- Poměr jde vylepšit sofistikovanějšími technikami zvětšování odhadu

# Shop scheduling

- Úloha (job) je složena z podúloh (task)
- Uvažují se 3 varianty:
  - Open shop - podúlohy můžou být zpracovány v libovolném pořadí, ale ne dvě současně na různých strojích
  - Flow shop - pořadí podúloh je pevné a stejné pro všechny úlohy
  - Job shop - pořadí podúloh je pevné pro každou úlohy, ale pro jinou úlohu může být jiné
- Poprvé mezi zmiňovanými problémy dává smysl nechávat na strojích volné sloty
- Pro flow a job shop není možné navrhnout algoritmus s poměrem lepším než 2, dosáhnout tento poměr je snadné