DPG – Paralelní zpracování a hardware

Architektury používané při paralelním zpracování

Základním kritériem pro rozdělení výpočetních architektur používaných pro paralelní výpočty je to, jaký typ paměti používají procesy pro vzájemnou komunikaci.

Rozlišují se dva druhy architektur :

A) Paralelní počítače (multiprocesory) se sdílenou pamětí

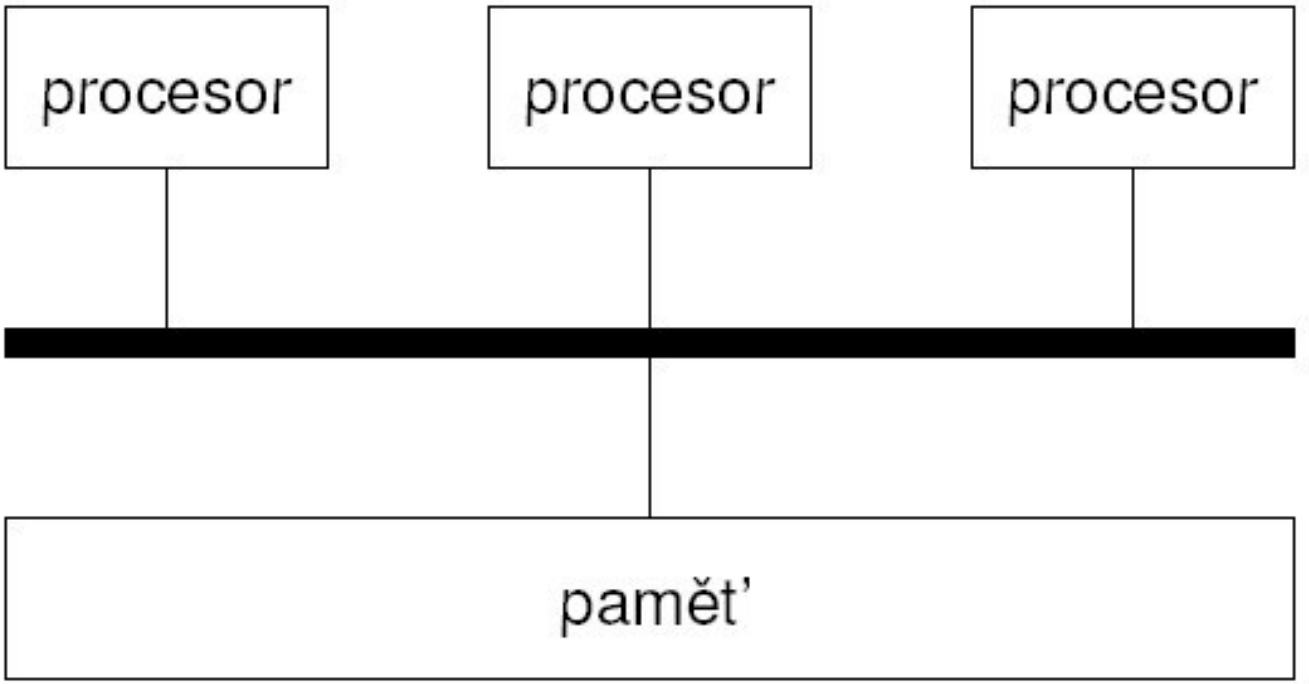
- Procesy mohou sdílet paměť (RAM) jednoho počítače.
- U symetrických multiprocesorů může běžet kterýkoliv proces na kterémkoliv procesoru, procesory jsou univerzální.
- Naproti tomu u asymetrických multiprocesorů je každý procesor specializován na určitý úkol.

B) Paralelní počítače (multiprocesory) s distribuovanou pamětí

- Jedná se o více (většinou velké množství) strojů spojených dohromady pomocí komunikační sítě.
- Procesy spolu většinou komunikují zasíláním zpráv (tzv. message passing).
- Masivně paralelní počítače a clustery

DPG – Paralelní zpracování a hardware

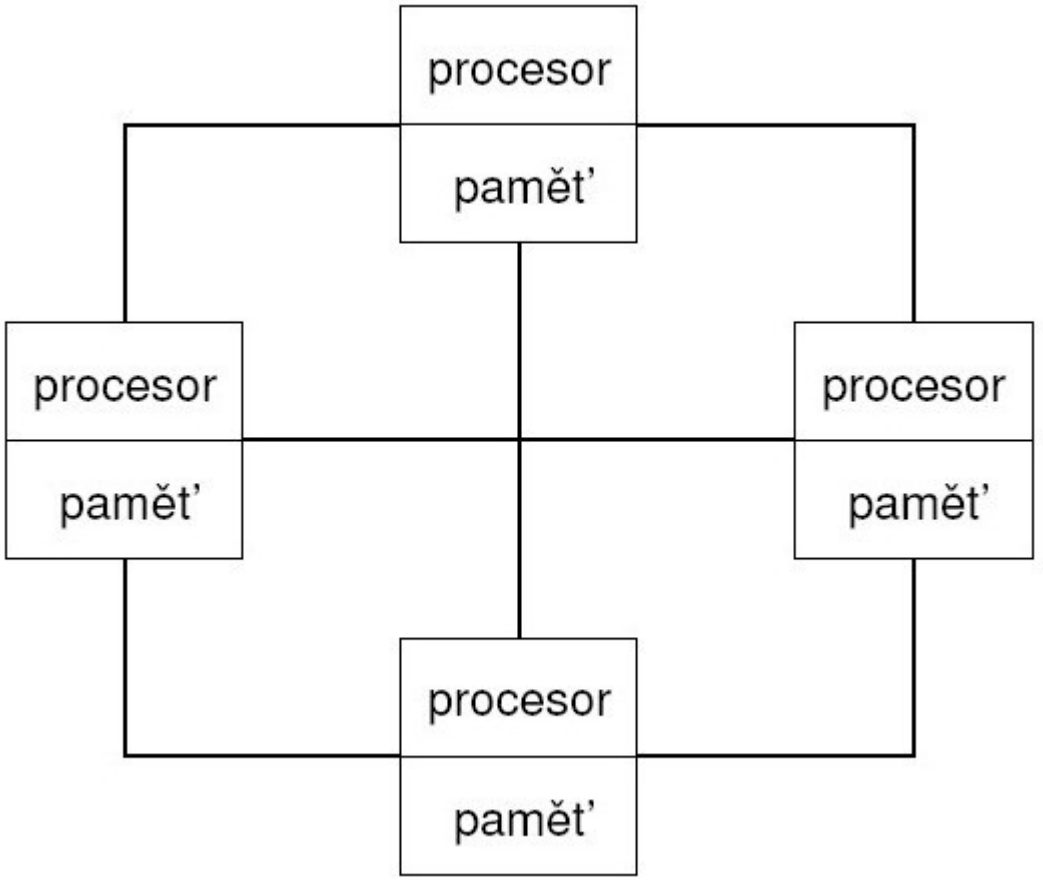
Systémy se sdílenou pamětí



Procesory komunikují prostřednictvím sdíleného paměťového prostoru. Mohou také tak synchronizovat své aktivity, je ovšem nutno řešit problém exkluzivního přístupu do paměti.

DPG – Paralelní zpracování a hardware

Systémy s distribuovanou pamětí



Není problém s exkluzivitou přístupu do paměti, naopak je nutné řešit komunikační problém přímými komunikačním kanálem mezi procesory.

DPG – Paralelní zpracování a programovací jazyky

Druhy paralelního zpracování

Paralelismus, tedy paralelní zpracování nějaké úlohy, můžeme rozdělit do tří základních skupin:

A) Funkční paralelismus

- Složitá činnost (popsaná nějakou funkcí) se rozdělí na více jednodušších, dílčích činností.
- Každou z těchto dílčích činností vykonává jedno vlákno (proces).
- Funkční paralelismus se typicky používá pro složité úlohy nad jednoduchými daty.
- Analogie se životem: při stavbě rodinného domku se zároveň zavádí elektřina i plyn.
- Tento paralelismus má smysl i na jednoprocesorových počítačích.

B) Datový paralelismus

- Používá se v případě relativně jednoduché činnosti nad rozsáhlými daty.
- Tato data se rozdělí „na kousky“ a nad každým kouskem dat provede jedno vlákno tutéž činnost.
- Analogie se životem: dlouhý výkop hloubí parta kopáčů, každý je zodpovědný za jeden úsek výkopu.
- Tento paralelismus prakticky nemá smysl provozovat na jednoprocesorovém stroji

C) Zřetězené zpracování dat

- Vlákna (procesy) si „předávají“ nějaký datový záznam
- Každé s ním udělá nějaký „kus práce“ a předá jej dál.
- V tom okamžiku je již připraveno opět přijmout další záznam.

DPG – Paralelní zpracování a programovací jazyky

Programovací prostředky pro paralelizaci

A) Jazyky s explicitní podporou paralelismu

- Jazyk s explicitní podporou paralelismu (paralelismus je přímou součástí jazyka)
- Podpora paralelismu na úrovních příkazů, procedur a procesů
- Typickým představitelem je ADA a High Performance Fortran

B) Použití speciálních paralelizačních kompilátorů

- Možnost použití univerzálního jazyka (např. C nebo Fortran)
- Kompilátor automaticky detekuje možnou souběžnost kódu v sekvenčním algoritmu
- Umožňuje „bezpracný“ převod na paralelní zpracování - řešení není vždy přímočaré

C) Použití speciálních paralelizačních knihoven

- Možnost použití univerzálního jazyka a běžného kompilátoru (výhoda)
- Paralelizace je realizována pomocí speciálních knihoven externích funkcí a objektů
- Nejčastěji jsou používány knihovny PVM (Parallel Virtual Machine) či MPI (Message Passing Interface)
- Typickými jazyky jsou C, C++, Fortran, Java (balíčky JPVM a mpiJava)

DPG – Paralelní zpracování a programovací jazyky

Realizace paralelních procesů

A) Samostatné úlohy operačního systému

- Typicky se používají u systémů s distribuovanou pamětí.
- Úlohy spolu nějak spolupracují, např. pomocí nástrojů typu PVM nebo MPI.
- Známe jej jako tzv. hrubý paralelismus (coarse grain)

B) Vlákna (threads, lightweight processes)

- Používají se u systémů se sdílenou pamětí.
- Vlákna běží v rámci jedné úlohy, tj. sdílí její kontext.
- Známe jej jako tzv. jemný paralelismus (middle grain).
- Vláknům bude věnována jedna z příštích přednášek

C) Korutiny (pseudoparalelismus, kvaziparalelismus)

- Nejedná se o opravdový paralelismus.
- Jedná se o procedury, které si vzájemně předávají řízení v jednoprocesorovém systému.
- Podrobnosti viz následující slide

DPG – Paralelní zpracování a programovací jazyky

Součásti paralelního procesu

- Program, dle kterého je prováděn
- Lokální data
- Stav -- lokálních data, hodnoty registrů procesoru, zásobník, místo v programu, kde se proces nachází.

Existují tyto základní druhy řízení paralelních procesů:

- Časová kvanta (Time Sharing), každému procesu je přiděleno určité časové kvantum, po jeho vypršení je procesu odebrán procesor a ten je dán jinému procesu.
- Preemtivní multitasking (Preemptive Multitasking), každý proces má přidělenu prioritu, běží proces s největší prioritou. Pokud se objeví proces s vyšší prioritou, než má právě běžící proces, je tento přerušen.
- Řízení korutin, přidělování procesoru je řízeno korutinami samými, tj. právě prováděná korutina rozhoduje, jaká korutina poběží dále a kdy dojde k přepnutí.

DPG – Paralelní zpracování a programovací jazyky

Korutiny

- Korutina je programová jednotka pro modelování pseudoparalelního jednoprocesorového systému.
- Představuje speciální druh podprogramů
- Korutina je aktivována z hlavního programu podobně jako procedura.
- Běh korutiny může být kdykoliv přerušeno a řízení může být předáno do libovolného bodu jiné korutiny.
- Po návratu zpět do korutiny se pokračuje od místa přerušení.

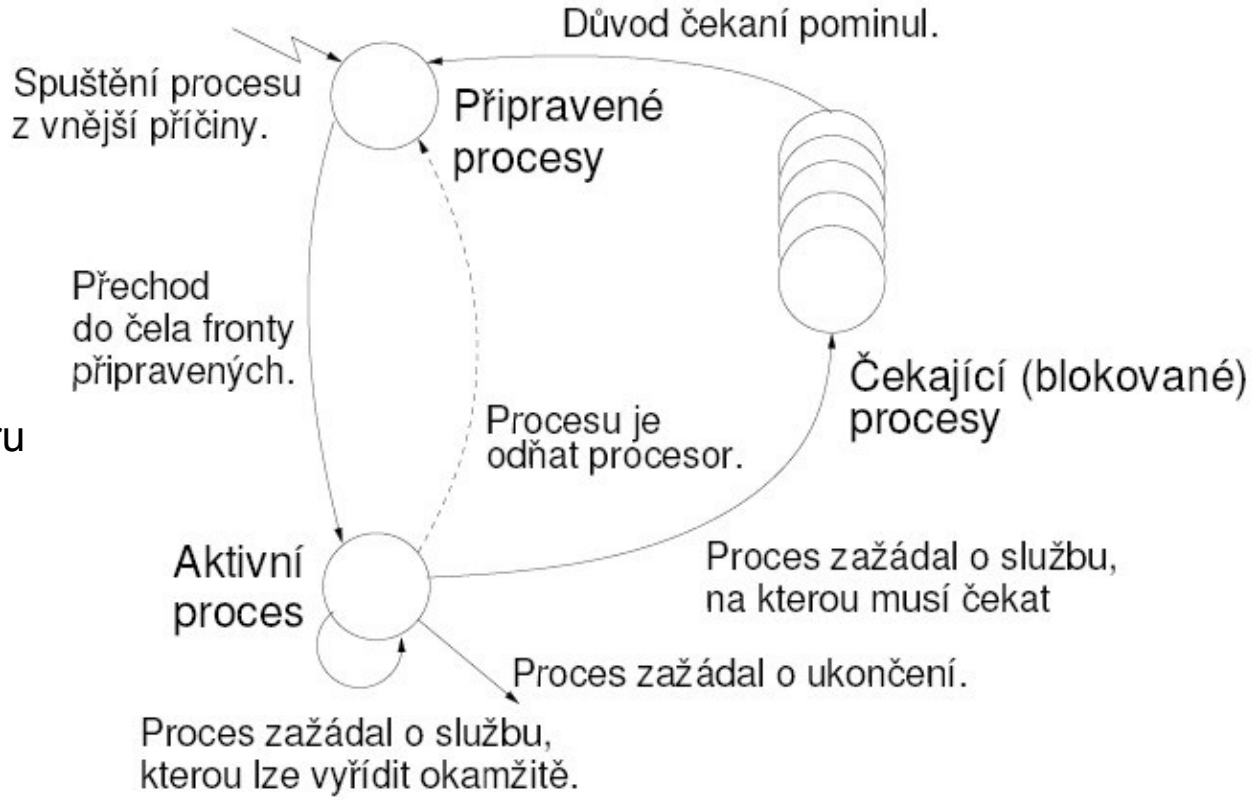
Vlastnosti korutin

- Symetričnost – žádná není „rodič“ ani „potomek“, všechny jsou si rovny
- Sekvenčnost – pouze jediná korutina je aktivní, přerušuje provádění předáním řízení jiné korutině
- Mohou mít více vstupů
- Korutina má své lokální běhové prostředí (runtime environment, RE)
- RE zůstává v paměti i v době, kdy korutina není aktivní

DPG – Paralelní zpracování a programovací jazyky

Proces

Proces je spuštěný program ve vyhrazeném prostoru paměti. Jedná se o entitu operačního systému, která je plánována pro nezávislé provádění. Každý proces je identifikován svým jedinečným PID (Process ID). Plánovač procesů řídí efektivní přidělování procesoru procesům na základě jejich vnitřních stavů a na strategiích plánování



Stavy procesů

- Executing – právě běžící na procesoru
- Blocked – proces čeká na periferie
- Waiting – proces čeká na procesor
- Terminated – proces je ukončen

DPG – Paralelní zpracování a programovací jazyky

Úloha operačního systému

- Operační systém integruje a synchronizuje práci jednotlivých procesorů
- Odděluje uživatele od fyzické architektury

OS poskytuje :

- Prostředky pro tvorbu a rušení procesů
- Prostředky pro správu více procesorů a procesů
- Prostředky pro rozvrhování procesů na procesory
- Systém a mechanismus pro přístup k paměti
- Mechanismy meziprocesní komunikace
- Mechanismy pro synchronizaci procesů

DPG – Paralelní zpracování a programovací jazyky

Paralelizace výpočetních postupů – základní modely dekompozice

Programování paralelní aplikace logicky předchází fáze analýzy s cílem určit přístup pro dekompozici výpočtu na složky (procesy) a dekompozici dat a zvolit odpovídající optimální model.

Model MPMD (Multiple Program Multiple Data)

- Dekompozice výpočtu na samostané paralelní činnosti
- Použitelné pro složitou činnost nad relativně málo objemnými daty
- Důraz je tedy kladen hlavně na dekompozici činnosti
- Data jsou logicky přiřazována jako lokální (odlišná data každého procesu)

Model SPMD (Single Program Multiple Data)

- Dekompozice podle dat
- Použitelné pro jednoduchou činnost nad objemnými daty
- Důraz je tedy kladen hlavně na dekompozici dat
- Stejné procesy pracují nad jednotlivými částmi dat – urychlení výpočtu (násobení matic, objemné iterace)

Model MPSD (Multiple Program Single Data)

- Zřetězené zpracování dat
- Různé procesy pracují nad stejnými daty
- Zpracování rozsáhlého proudu dat, nad jednotlivými datovými prvky jsou prováděny různé operace
- Analogie s výrobní linkou (pásová výroba)