IB109 Návrh a implementace paralelních systémů

Organizace kurzu a úvod

Jiří Barnat

Organizace kurzu

Organizace kurzu

Místo a čas

- Středa 16:00-17:40, A217

Ukončení předmětu

- Závěrečný písemný test na odpřednášený obsah
- Možno získat několik bodů za nepovinné domácí úlohy
- Požadavky na úspěšné ukončení předmětu

  Z,K: bodové hodnocení testu nad 50%

  ZK: bodové hodnocení testu nad
  60%(E), 67%(D), 75%(C), 82%(B), 90%(A)

Cíl kurzu IB109

Cílem předmětu je seznámit studenty s

- Problematikou programování paralelních aplikací.
- Programátorskými prostředky pro vývoj paralelních aplikací.
- Možnostmi studia tématu na FI.

Úspěšný absolvent kurzu

- Umí identifikovat paralelně proveditelné úlohy.
- Má základní přehled o problémech souvisejících s paralelizací.
- Nebojí se implementovat vlastní vícevláknové nebo jinak paralelní aplikace či systémy.
- Má představu o tom, co se děje v zákulisí použitých knihoven pro podporu programování paralelních aplikací.
- Umí tyto knihovny správně použít.

Organizace kurzu

Osnova:

1. 20/02 — Úvod, organizace kurzu, motivace
2. 27/02 — Práce s daty ve sdílené paměti
3. 06/03 — POSIX Threads 1
4. 13/03 — **Zrušeno**
5. 20/03 — POSIX Threads 2, Vlákna v MS Windows
6. 27/03 — Lock-Free programování
7. 06/04 — OpenMP, TBB, C++11, Parallel_For
8. 10/04 — Principy návrhu paralelních algoritmů
9. 17/04 — Kolektivní komunikace
10. 24/04 — Předávání zpráv v distribuované paměti, MPI
11. 01/05 — **Zrušeno**
12. 08/05 — **Zrušeno**
13. 15/05 — Složitostní analýza paralelních programů

Předpoklady a kontext na FI

IB109

- Úvodní kurz určený pro bakalářské studium.
- Povinný v rámci oboru *Paralelní a distribuované systémy*

Předpoklady

- Základní znalosti o fungování výpočetních prostředků a operačních systémů.
- Základní zkušenost s imperativním programováním sekvenčních algoritmů.

Kontext v rámci FI

PV192 – Paralelní algoritmy

- Paralelní zpracování, Klasifikace paralelních systémů, Úrovně paralelismu, Paralelní počítače, Systémy s distribuovanou pamětí, MPI

PV197 – GPU Programming

- Paralelní výpočty na grafických kartách s technologií CUDA.

PA150 – Principy operačních systémů

- Vlákna, procesy, monitory, semafory, synchronizace.
- Hierarchie pamětí.

IA039 – Architektura superpočítačů a intenzivní výpočty

- Procesory. Paralelní počítače. Překladače. MPI. PVM a koordinační jazyky. Profilování a měření výkonu.

Kontext v rámci FI – pokr.

IV100 – Paralelní a distribuované výpočty

- Distribuované systémy a algoritmy. Komunikační protokoly. Směrovací algoritmy a tabulky. Distribuované algoritmy pro detekci ukončení, volbu vůdce, vzájemné vyloučení, hledání nejkratší cesty. Byzantská shoda.

IV010 – Komunikace a paralelismus

- Teoretický model paralelních procesů a komunikace. CCS. Synchronizace, vnitřní akce. Ekvivalence systémů pomocí slabé/silné bisimulace a relace kongruence.

IV113/IA169 – Úvod do validace a verifikace

- Model checking, verifikace paralelních programů.

Kontext v rámci FI

Laboratoře na FI

- SITOLA
- PARADISE
- SYBILA

Projekty

- IV112 – Projekt z programování paralelních aplikací
- PV197 – GPU Programming

Studijní materiály

Knihy

- Maurice Herlihy, Nir Shavit: *The Art of Multiprocessor Programming*
- A. Grama, A. Gupta, G. Karypis, V. Kumar: *Introduction to Parallel Computing*
- I. Foster: *Designing and Building Parallel Programs*
- W. Group, E. Lusk, A. Skjellum: *Using MPI*
- ...

Studijní materiály

E-zdroje:

- http://www.wikipedia.org
- Kurzy a jejich studijní materiály na různých univerzitách
  - http://www.cs.arizona.edu/people/greg/mpdbook (Foundations of Multithreded, Parallel, and Distributed Programming)
  - http://renoir.csc.ncsu.edu/CSC495A

- http://www.hlrs.de/organization/par/par_prog_ws Parallel Programing Workshop (MPI, OpenMP)
- Domovské stránky projektů MPI, TBB, OpenMP, ...
- Online tutoriály
- ...

Motivace

Souběžnost a Výpočetní paralelismus

Souběžnost

- Existence dvou a více procesů (v obecném smyslu slova) v jeden časový okamžik.

Výpočetní paralelismus

- Napříč všemi úrovněmi, od implementace registru až po koexistenci rozsáhlých výpočetně distribuovaných aplikací.
- Chtěný pro zvýšení výkonu.
- Nutný kvůli prostorové distribuci.
- Vykoupený složitostí návrhu a pořizovací cenou.

Souběžnost v Computer Science

- Specifikace, implementace a analýza paralelních a distribuovaných systémů.
- Inherentně sekvenční algoritmy, složitostní třída *NC*.

Důvody vývoje paralelních aplikací

Výkonnost

- Umožní efektivně využít aggregované výpočetní prostředky k zrychlení výpočtu.

Proveditelnost

- Agregace výpočetní síly není volba, ale nutnost pro dokončení úlohy (velký objem výpočtů, nepřijatelná latence).

Bezpečnost

- Duplikace klíčových částí systému pro případ havárie, či ohrožení důvěry jedné části.

Cena

- Oddělení nesouvisejících částí aplikace, levnější údržba.
- Agregovaná výpočetní síla je levnější.

Výpočetních platformy – abstraktně

Abstraktní model výpočetního systému

Procesor - Datová cesta - Datové úložiště

- Všechny části systému mohou být úzkým místem vůči výkonnosti aplikace jako celku.
- Paralelismus je přirozený způsob překonání úzkých míst.

Procesory

Procesory

- Neustálá potřeba zvyšovat výkon.
- Výkon procesoru spojován s Moorovým zákonem.

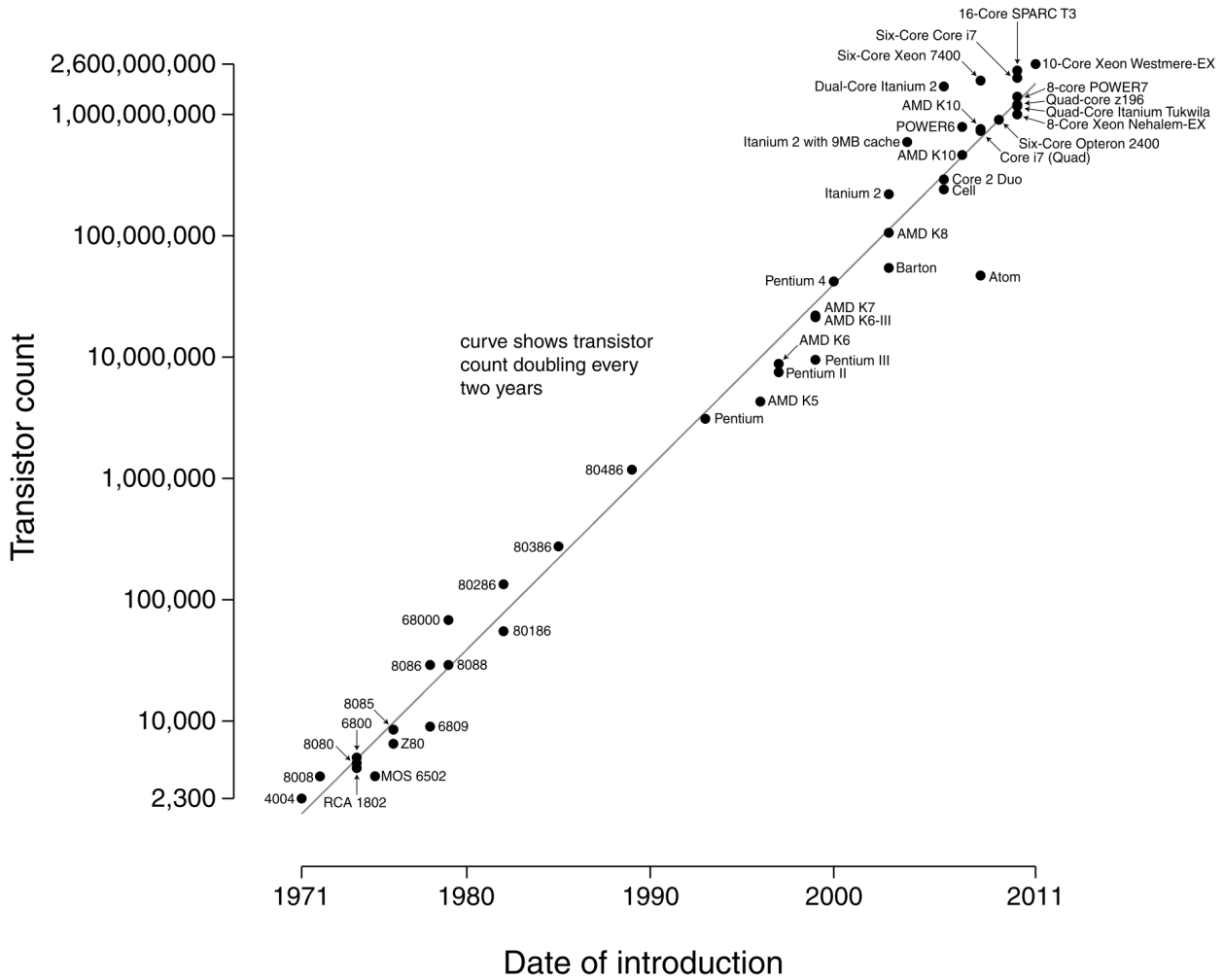
Moorův zákon

- Gordon Moore, spoluzakladatel Intelu
- Počet tranzistorů v procesoru se zdvojnásobí přibližně každých 18 měsíců.

Metody zvyšování výkonu procesorů

- Zvyšování frekvence vnitřních hodin.
- Multiplicita, Paralelismus

Moore's Law


Microprocessor Transistor Counts 1971-2011 & Moore's Law
Transistor count
2,600,000,000
1,000,000,000
100,000,000
10,000,000
1,000,000
100,000
10,000
2,300
curve shows transistor count doubling every two years
16-Core SPARC T3
Six-Core Core i7
Six-Core Xeon 7400
10-Core Xeon Westmere-EX
Dual-Core Itanium 2
8-core POWER7
Quad-core z196
AMD K10
Quad-Core Itanium Tukwila
POWER6
8-Core Xeon Nehalem-EX
Itanium 2 with 9MB cache
Six-Core Opteron 2400
AMD K10
Core i7 (Quad)
Core 2 Duo
Itanium 2
Cell
AMD K8
Barton
Pentium 4
Atom
AMD K7
AMD K6-III
AMD K6
Pentium III
Pentium II
AMD K5
Pentium
80486
80386
80286
68000
80186
8086
8088
8085
6800
6809
8080
Z80
8008
MOS 6502
4004
RCA 1802
1971
1980
1990
2000
2011
Date of introduction

Trend ve vývoji procesorů

Pozorování

- Výrobcům procesorů se nedaří zvyšovat výkon jednoho jádra.
- Fyzikální zákony brání neustálé miniaturizaci – okolo 5nm již nelze udržet elektrony v atomu.
- Současná technologie 14-16nm (dříve 22nm, 28nm, 32nm, 45nm, 65nm, 90nm).

Řešení

- Multi-core a many-core procesory.
- Pravděpodobný způsob zvyšování výkonu i v budoucnosti.
- Částečný odklon od jednotek monolitických jader k desítkám menších specializovaných, či k hybridním řešením.

Důsledek

- Sekvenční algoritmy nemohou nadále těžit z rostoucího výkonu procesorů.
- Paralelizace výpočtů je nevyhnutelný směr vývoje.

Datové cesty

Role paralelismu v komunikaci

- Větší propustnost komunikačních linek s následným efektem snižování latence.
- Robustnost a spolehlivost komunikačních linek.

Příklady paralelismu na datové cestě

- Šířka sběrnice 32/64/128 bitů.
- Domácí dual-band WIFI router.

Paměť

Fakta

- Výkon procesorů převyšuje výkon ostatních komponent.
- Cesta **procesor – paměť – disk** je zdlouhavá.
- Doba nutná pro získání jednotky informace z paměti roste se vzdáleností místa uložení od místa zpracování.

Víceúrovňové uložení informací

- Registry procesoru
- L1/L2/L3 cache
- Operační paměť
- Cache I/O zařízení
- Magnetické/optické mechaniky

Cache

Cache paměť obecně:

- Kopie části dat v rychleji dostupném místě.
- Může a nemusí být kontrolovatelná uživatelem nebo OS.

Příklady Cache s různou možností kontroly

- L1/L2 cache v rámci CPU nekontrolovatelná programátorem
- I/O efficient algoritmy
  - Obcházejí virtualizaci paměti kontrolovanou OS, a místo toho realizují vlastní způsob použití operační paměti jako cache pro data na disku.

Použití mnoha paměťových modulů

Multiplicita paměťových modulů

- Větší množství uložitelných/zapamatovatelných informací.
- Větší množství linek do paměti (větší propustnost).
- Větší režie na udržení konzistence.

Příklady

- Disková pole
- Peer-to-Peer sítě
- NUMA architektury
  - Více procesorové počítače s více paměťovými moduly uspořádanými tak, že přístup jednoho procesoru do různých paměťových modulů je různě rychlý.

Paralelní výpočty

Paralelismus z pohledu OS

Multitasking na jednom výpočetním jádru

- Aplikace se v běhu na CPU jádru střídají.
- Na jednom CPU zdánlivě "běží"více aplikací.
- Jednotka plánování OS je proces.

Multitasking na více výpočetních jádrech

- Různé aplikace přiřazeny na různá výpočetní jádra.
- Jinak standardní multitasking na každém jednom jádře.
- Jednotka plánování OS je proces.

Multitasking a multithreading

- Každá aplikace může mít více výpočetních vláken.
- Vlákna se v běhu na jednom CPU jádru střídají.
- Vlákna jedné aplikace mohou běžet na různých jádrech.
- Jednotka plánování OS je vlákno.

Flynnova klasifikace

Single Instruction Single Data

- V daný okamžik je zpracovávána jedna instrukce, která pracuje nad jedním datovým proudem.
- Klasický sekvenční výpočet.

Single Instruction Multiple Data

- Jedna tatáž instrukce je vykonávána nad více datovými proudy.
- Vektorové instrukce CPU, architektura GPU.

Multiple Instruction Multiple Data

- Nezávislý souběh dvou a více SISD, SIMD přístupů.
- Více jádrové procesory.

Multiple Instruction Single Data

- Prakticky se nevyskytuje.

Distribuovaný systém

Distribuovaný/paralelní systém

- Specifikován po částech (procesy).
- Chování systému vzniká interakcí souběžných procesů.
- Emergentní jevy.

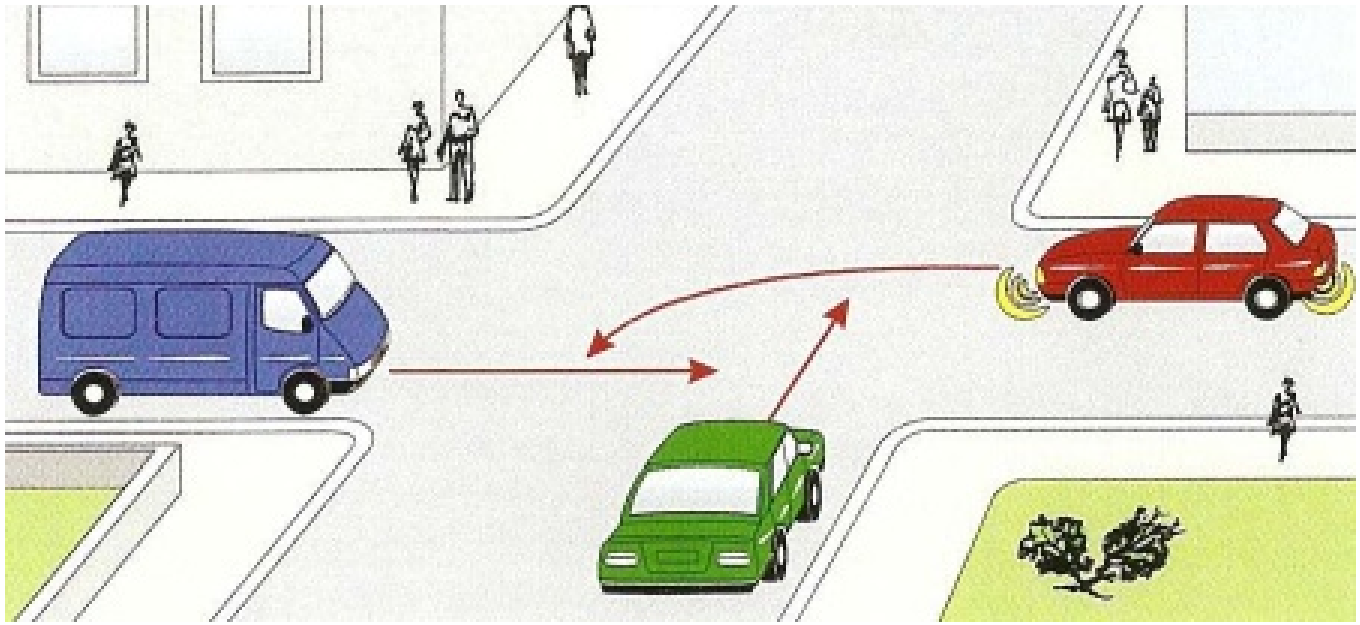
Synchronizace

- Omezení na prokládání a souběh akcí jednotlivých procesů distribuovaného systému.

Komunikace

- Přenos informace z jednoho procesu na druhý.

Příklad distribuovaného systému



Vybrané problémy distribuovaných systémů

Jevy

- Nekonzistentní vize konzistentního světa.
- Vzájemná interference

Rizika

- Race-condition, nedeterministické chování.
- Uváznutí (Deadlock, Livelock)
- Stárnutí, hladovění (Starving)
- Přetečení buferů, problém producent-konzument.
- Zbytečná ztráta výkonu (aktivní čekání).

Vývoj paralelních systémů je náročnější

Důvody:

- Nutnost specifikace souběžných úkolů.
- Nutnost specifikace koordinace úkolů.
- Paralelní algoritmy.
- Nedostačující vývojová prostředí.
- Nedeterminismus při simulaci paralelních aplikací.
- Absence reálného modelu paralelního počítače.
- Rychlý vývoj a zastarávání použitých technologií.
- Výkon aplikace náchylný na změny v konfiguraci systémů.
- ...

Příklad

- V minulých letech byl doporučován pro hry 2-jádrový procesor, proč ne 4-jádrový, když byl zcela určitě výkonnější?

Vývoj paralelních systémů je náročnější

Důvody:

- Nutnost specifikace souběžných úkolů.
- Nutnost specifikace koordinace úkolů.
- Paralelní algoritmy.
- Nedostačující vývojová prostředí.
- Nedeterminismus při simulaci paralelních aplikací.
- Absence reálného modelu paralelního počítače.
- Rychlý vývoj a zastarávání použitých technologií.
- Výkon aplikace náchylný na změny v konfiguraci systémů.
- ...

Příklad

- V minulých letech byl doporučován pro hry 2-jádrový procesor, proč ne 4-jádrový, když byl zcela určitě výkonnější?
- Je obtížné napsat herní engine, který by fungoval dobře na 2-jádrovém stroji a na 4-jádrovém stroji běžel 2x rychleji, je preferována vyšší frekvence 2-jádrového procesoru.

HPC

HPC

HPC (High Performance Computing)

- Oblast Computer Science
- Výpočty na vysoce paralelních platformách

Nejvýkonější počítač světa [Jaro 2018]

- National Supercomputing Center in Wuxi
- TFLOPS: 93 014

Nejvýkonější počítač světa [Jaro 2010]

- Roadrunner
- TFLOPS: 1 105

- Více viz www.top500.org.

Nejvýkonnější počítače