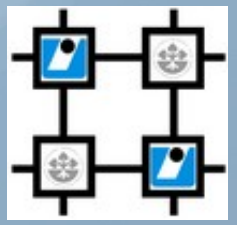
Klastrové a gridové počítanie



Ladislav Hluchý
Ústav informatiky SAV
Bratislava, Dúbravska cesta 9

Paralelizmus – hardvér

- **Moderné architektúry počítačov**

  – Multi-jadrové procesory
    - jadro - paralelismus na úrovni inštrukcií
  – Floatingpoint akcelerátory
  – Grafické procesory (GPU)
  – Komplexné hierarchie pamätí

  – HPC klastre, superpočítače, gridy

Paralelizmus – softvér

- **Architektúra aplikácie**
  - vhodný programovací model schopný využiť čo najviac úrovní hardvérového paralelizmu
    - heterogénne jadrá, akcelerátory procesora
    - klaster
    - grid
- **Nástroje a vývojové prostredia**
  - mali by poskytovať možnosť automatického návrhu paralelnej aplikácie na rôznych úrovniach abstrakcie, uplatnením rôznych stratégií na distribúciu kódu a údajov a použitím paralelných objektov rôznej granularity

Paralelné programovanie

- **Prechod od sekvenčného programovacieho modelu k paralelnému** ⇒ prehodnotenie predstavy toku procesu

  – detekovanie činností, ktoré môžu byť vykonávané paralelne
  – zostavenie aplikácie ako štruktúrovanej množiny úloh spolu s definovaním závislostí medzi úlohami
  – vytvorenie vlákien (thread) v rámci úlohy a definovanie koordinácie medzi nimi a ich operáciami

Dekompozícia programu

- **Základné stratégie**
  - dekompozícia podľa úloh: vlákna vykonávajú rozdielne činnosti
  - dekompozícia podľa dát: vlákna vykonávajú tú istú činnosť nad rozdielnymi dátami
  - dekompozícia podľa toku dát: výstup jedného vlákna slúži ako vstup pre iné vlákno

- **Stratégia dekompozície**
  - voľba závisí od vlastností konkrétnej aplikácie
    - granularita, dominantnosť (výpočty, práca s údajmi, kolaboratívne činnosti)
  - vyžaduje manažment simultánnych procesov a ich interakcií
    - komunikácie, synchronizáciu, vyváženie vyťaženia, škálovateľnosť
  - významnú úlohu hrá empirické testovanie a vyhodnotenie

Kategórie výpočtov (granularita)

- **Jemno-zrnné paralelné výpočty (High performance computing)**
  - pri vykonávaní úlohy každá z jej pod-úloh je silne závislá na výsledku iných pod-úloh
    - klastre, superpočítače, tesne viazané klastre s veľkým počtom procesorov a rýchlou komunikačnou sieťou

- **Hrubo-zrnné paralelné výpočty (High throughput computing)**
  - pri vykonávaní úlohy každá z jej pod-úloh je relatívne nezávislá na výsledku iných pod-úloh (tj. oneskorenie pri obdržaní výsledku z jedného procesora nemá významný vplyv na činnosť iných procesorov)
    - voľne viazané siete výpočtových prostriedkov (napr. grid)

Architektúry počítačov – programovacie modely

- **Základný klasifikačný model** (M.J. Flynn, 1966)
  - SISD (Single Instruction, Single Data)
  - SIMD (Single Instruction, Multiple Data)
  - MISD (Multiple Instruction, Single Data)
  - MIMD (Multiple Instruction, Multiple Data)

- **MIMD architektúra**
  - SPMD (Single Program, Multiple Data) (F. Darema, 1984)
    - najrozšírenejší model paralelného programovania
  - MPMD (Multiple Program, Multiple Data)
    - master-worker, workflow, DAG
  - podľa vlastností fyzickej pamäte (SPMD, MPMD)
    - systémy s distribuovanou pamäťou, zdieľanou pamäťou, distribuovanou a zdieľanou pamäťou

Klastrové počítanie

- **Moderný klaster s viac-jadrovými CPU**
  počítačový systém s distribuovanou a zdieľanou pamäťou
  - množina nezávislých viac-jadrových procesorov, navzájom spojených prostredníctvom prepojovacej siete, ktorá umožňuje medzi-uzlovú komunikáciu
  - každý procesor má svoju vlastnú pamäť (distributed memory - DM)
  - všetky jadrá v rámci uzla zdieľajú jednu pamäť (shared memory - SM)

- **Klastrové počítanie** ⇒ hybridný programovací model
  - uplatnenie technológií pre DM a technológií pre SM (prípadne aj technológií pre GPU)

Klastrové počítanie

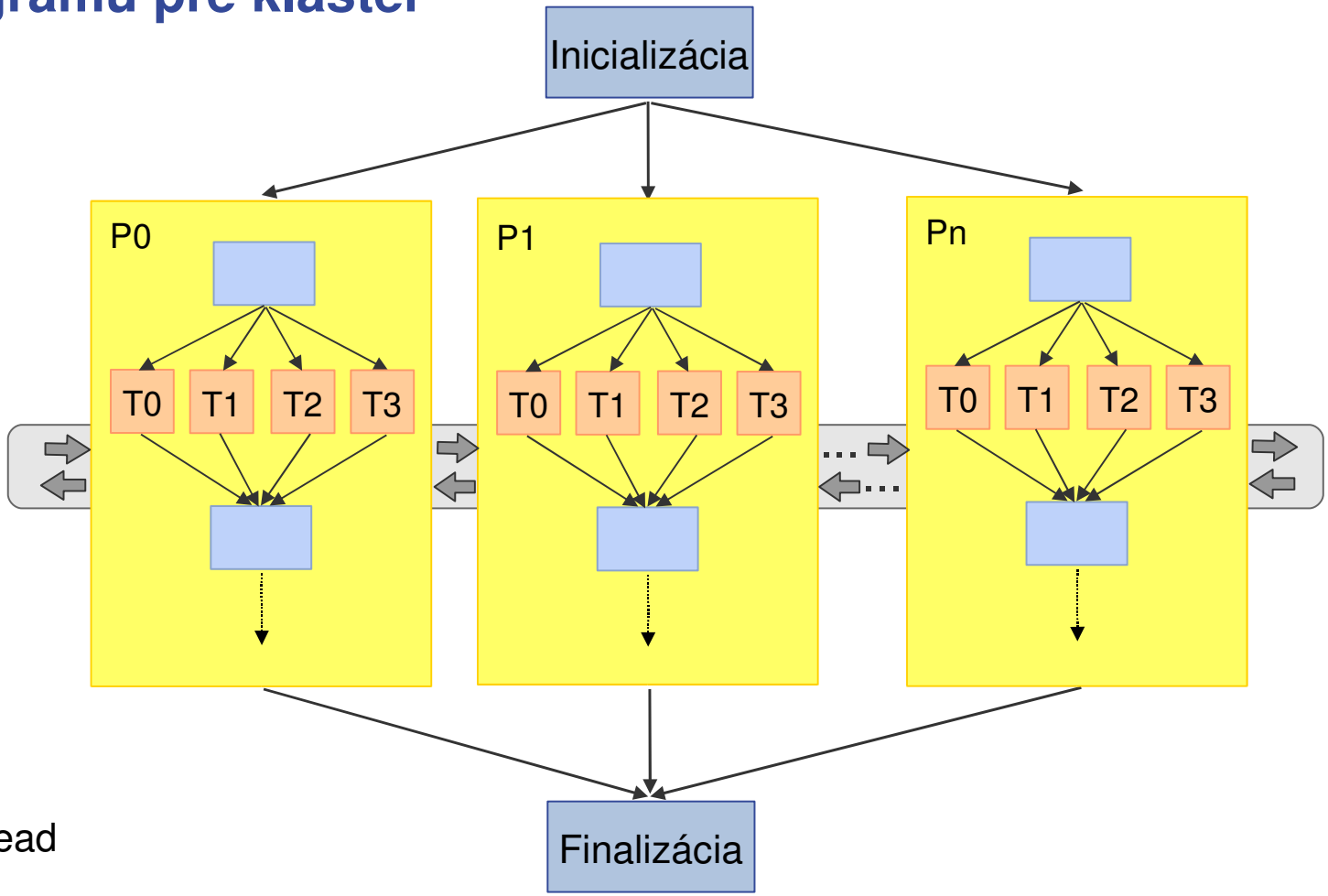
- **Systém s distribuovanou pamäťou**
  - každý procesor má svoj vlastný adresný priestor: výpočtové úlohy môžu operovať iba nad svojimi lokálnymi dátami
    - vzdialené dáta je nutné prenášať prostredníctvom komunikácií s inými procesormi (“message passing”) cez prepojovaciu sieť
    - kľúčovou otázkou je: ako distribuovať dáta aby sa predišlo častým komunikáciám medzi procesormi
  - MPI (Message Passing Interface) – špecifikácie MPI-1 a MPI-2 sa stali štandardom pre implementáciu komunikácií a rôznych kolektívnych operácií nad procesormi
    - existuje veľa implementácií (pre Fortran a C) pre rôzne platformy
    - väčšina implementácií alokuje pri inicializácii programu fixný počet MPI procesov (jeden proces na CPU/jadro)

Klastrové počítanie

- **Systém so zdieľanou pamäťou**
  - viac-jadrový procesor umožňuje zdieľanie jedného spoločného adresného priestoru pre všetky jadrá
    - nie je potrebné explicitné špecifikovanie dátových komunikácií
    - rôzne jadrá sa môžu navzájom rušiť pri zápise na rovnaké miesto pamäti – je nutné použiť synchronizačné mechanismy (zámky, semafory, bariéry)
    - pochopenie a manažment lokálnosti je zložitejší problém
  - OpenMP (Open Multiprocessing) – dominantný programovací model na implementáciu "multi-threading" konceptu pre SM
    - API pre paralelné programovanie v jazykoch C/C++ a Fortran
    - explicitný "Fork-Join" model pre vykonávanie paralelného programu
    - hlavné komponenty: compiler directives, library routines, environment variables

Klastrové počítanie

- **Model programu pre klaster**



Pi – MPI proces
Ti – OpenMP thread
→ tok riadenia
⇨ dátové komunikácie

Klastrové počítanie

- **Vytvorenie úlohy**
  - vývoj kódu, kompilácia (Linux)
  - vývoj skriptov (pre spustenie úlohy lokálne na klastri, automatizovanie procesu vykonávania úlohy)

- **Vykonanie úlohy**
  - prostredníctvom systému PBS (Portable Batch System)
  - najpoužívanejšie príkazy:
    - **qsub** – predloží vstupný skript batch-serveru na vykonanie
    - **qstat** – požiada batch-server o výpis stavu úlohy
    - **qdel** – požiada batch-server o zrušenie úlohy

- **Popis úlohy:** PBS shell-skript (vstup pre “qsub”)

Klastrové počítanie

- **PBS shell-skript** (príklad)

```
#!/bin/sh
#PBS -l nodes=4:ppn=4
#PBS -N testing
#PBS -o std.out
#PBS -e std.err
#PBS -q local
#PBS -v OMP_NUM_THREADS=4
cd $PBS_O_WORKDIR
# . . . shell commands . . .
echo "Testing start: "`date`
${MPI_PATH}/mpiexec  --bynode -np 4 testing.exe input.dat
echo "Testing end: "`date`
exit
```

Gridové počítanie

- **Grid**
  paralelný a distribuovaný systém, ktorý umožňuje zdieľanie, výber a zoskupenie geograficky distribuovaných autonómnych prostriedkov (výpočtových, úložných, softvérových, a iných) dynamickým spôsobom, v závislosti od ich dostupnosti, vybavenia, výkonnosti, ceny a používateľských požiadaviek na kvalitu služieb
  - súčasné výpočtové prostriedky: počítačové klastre

- **Gridové počítanie** ⇒ hybridný programovací model
  - uplatnenie technológií pre klastrové počítanie a technológií webových a gridových služieb

Gridové počítanie

- **Vytvorenie úlohy**
  - vývoj kódu, kompilácia (Linux) – ako pre klaster
  - vývoj skriptov (pre spustenie úlohy na gride, automatizovanie procesu vykonávania úlohy)
- **Vykonanie úlohy**
  - prostredníctvom gridového middlewaru gLite (Globus)
    EMI (European Middleware Initiative): gLite+ARC+UNICORE
  - najpoužívanejšie príkazy (gLite WMS):
    - **glite-wms-job-submit** – predloženie úlohy na vykonanie
    - **glite-wms-job-status** – výpis stavu vykonávanej úlohy
    - **glite-wms-job-output** – výber výsledkov úlohy
    - **glite-wms-job-cancel** – predčasné ukončenie úlohy
- **Popis úlohy:** skript v jazyku JDL (Job Description Language)
  (vstup pre “glite-wms-job-submit”)

Gridové počítanie

- **Typ úlohy** (JDL pre glite WMS)
  - Job - jednoduchá úloha
    - **Normal** (štandardné vykonateľné programy a skripty, skripty spúšťajúce MPI programy)
    - **Parametric** (množina identických úloh, ktoré bežia s rôznymi parametrami resp. dátami)
    - **MPICH** (MPICH programy vykonateľné priamo)
    - **Interactive** (štandardný vstup/výstup úlohy je priamo pripojený ku klientovi, ktorý spúšťa úlohu, bez presmerovania z/do súboru)
  - DAG - graf závislých úloh
  - Collection - množina nezávislých úloh

Gridové počítanie

- **Jednoduchá úloha**

```
Type="Job";
Jobtype="Normal";
Executable="testing.exe";
Arguments="input.dat";
StdOutput="std.out";
StdError="std.err";
InputSandbox={ "testing.exe", "input.dat" };
OutputSandbox={ "output.dat" };
RetryCount=3;
ShallowRetryCount=0;
#Requirements=
```

Gridové počítanie

- **MPI úloha**

```
Type="Job";
Jobtype="MPICH";
CpuNumber=4;
Executable="mpi-testing.exe";
Arguments="input.dat";
StdOutput="std.out";
StdError="std.err";
InputSandbox={ "mpi-testing.exe", "input.dat" };
OutputSandbox={ "output.dat" };
RetryCount=0;
ShallowRetryCount=0;
#Requirements=
```

Gridové počítanie

- **MPI úloha** (spustená cez wrapper-skript)

```
Type="Job";
Jobtype="Normal";
CpuNumber=4;
Executable="run_mpi-testing.sh";
Arguments="4 mpi-testing.exe input.dat";
StdOutput="std.out";
StdError="std.err";
InputSandbox={"run_mpi-testing.sh","mpi-testing.exe", "input.dat"};
OutputSandbox={ "output.dat",  "mpistd.out"};
RetryCount=0;
Requirements= Member("OPENMPI",
   other.GlueHostApplicationSoftwareRunTimeEnvironment);
```

Gridové počítanie

- **Parametrická úloha**

```
Type="Job";
Jobtype="Parametric";
Executable="par-testing.exe";
Arguments="input_PARAM_.dat";
Parameters=10;
ParameterStart=0;
ParameterStep=1;
StdOutput="std.out";
StdError="std.err";
InputSandbox={ "par-testing.exe", "input_PARAM_.dat" };
OutputSandbox={ "output_PARAM_.dat" };
```

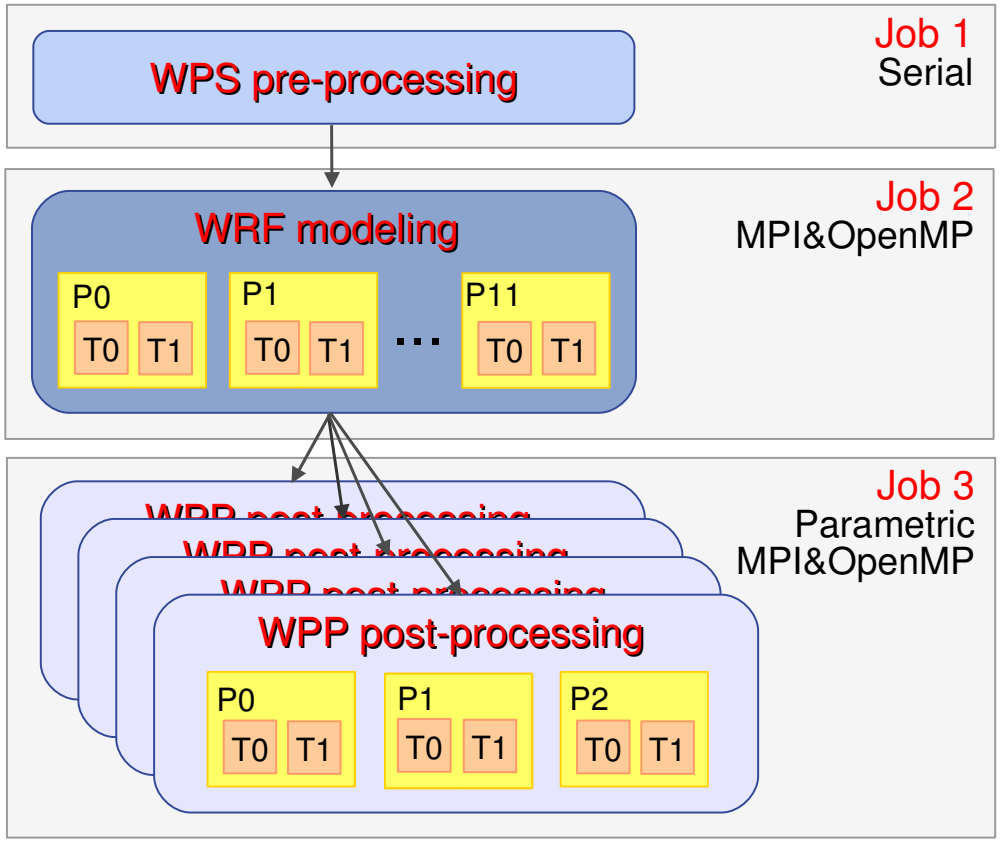
Gridové počítanie

- **DAG úloha**

```
Type="Dag";
DefaultRetryCount=0;
InputSandbox={ "input.dat" };
max_running_nodes=2;
nodes = [
   nodeA = [ description = [
      Jobtype="Normal";
      Executable="testingA.exe";
      InputSandbox={ "testingA.exe", root.InputSandbox[0] };
       . . .
   ]; ];
   nodeB = [ description = [ . . . ]; ];
   dependencies = { nodeA, nodeB };
];
```

Aplikácia

- **Workflow WRF simulácie (Weather Research and Forecast Model)**



Pi – MPI proces, Ti – OpenMP thread, ⟶ tok riadenia (závislosti)

Linky

- EGEE http://www.eu-egee.org
- EGI http://www.egi.eu
- SlovakGrid http://www.slovakgrid.sk
- MPI http://www.mcs.anl.gov/research/projects/mpi
- OpenMP http://openmp.org/wp
- gLite http://glite.web.cern.ch/glite
- PBS http://www.pbsworks.com

Ďakujem za pozornosť!