C2115
Praktický úvod do superpočítání

XIII. lekce

Petr Kulhánek, Tomáš Bouchal

kulhanek@chemi.muni.cz

Národní centrum pro výzkum biomolekul, Přírodovědecká fakulta,
Masarykova univerzita, Kotlářská 2, CZ-61137 Brno

Obsah

- **Infinity**
  úloha, přehled příkazů, aliasy
- **Spouštíme aplikace**
  sander, pmemd, gaussian, paralelní spouštění
- **Cvičení**
  efektivita paralelního spouštění aplikaci sander, pmemd, gaussian

Infinity

https://lcc.ncbr.muni.cz/whitezone/development/infinity/

Přehled příkazů

Správa software:

- site — aktivace logických výpočetních zdrojů
- module — aktivace/deaktivace software

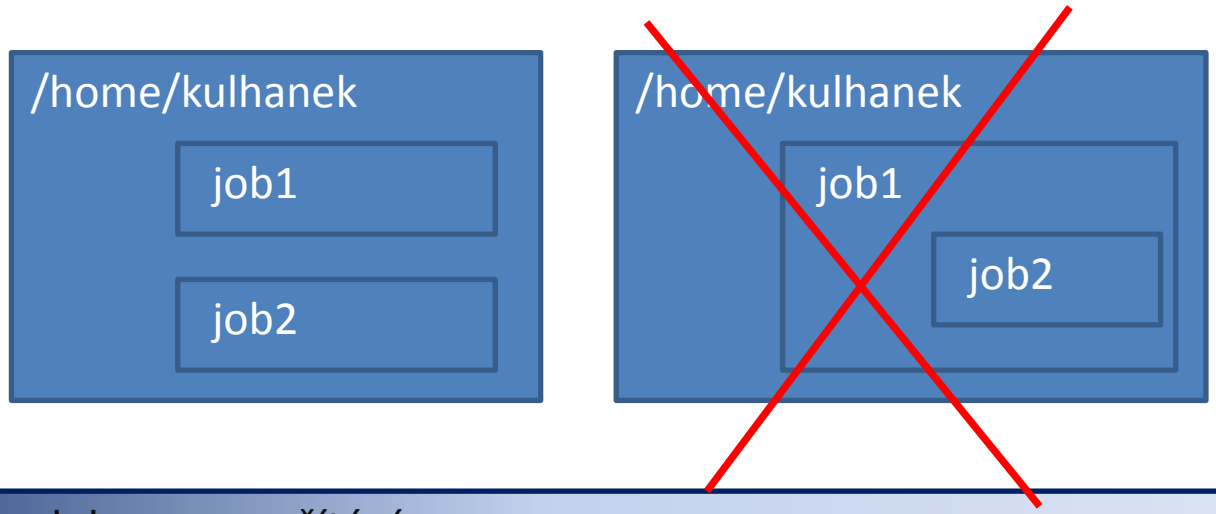
Správa úloh:

- pqueues — přehled front z dávkového systému dostupných uživateli
- pnodes — přehled výpočetních uzlů dostupných uživateli
- pqstat — přehled všech úloh zadaných do dávkového systému
- pjobs — přehled úloh uživatele zadaných do dávkového systému
- psubmit — zadání úlohy do dávkového systému
- pinfo — informace o úloze
- pgo — přihlásí uživatele na výpočetní uzel, kde se úloha vykonává
- paliases — definovaní aliasů

Úloha

Úloha musí splňovat následující podmínky:

- každá úloha se spouští v samostatném adresáři
- všechny vstupní data úlohy musí být v adresáři úlohy
- adresáře úloh nesmí být do sebe zanořené
- průběh úlohy je řízen skriptem nebo vstupním souborem (u automaticky detekovaných úloh)
- skript úlohy musí být v bashi
- ve skriptu úlohy se nesmí používat absolutní cesty, všechny cesty musí být uvedeny relativně k adresáři úlohy



Skript úlohy

Skript úlohy může být uvozen standardním interpretrem pro **bash** nebo speciálním interpretrem **infinity-env**, který nedovolí spuštění úlohy mimo výpočetní uzel. Druhý přístup zabraňuje případnému poškození/přepsání/smazání již vypočtených dat nechtěným opětovným spuštěním skriptu.

```
#!/bin/bash

# vlastni skript
```

```
#!/usr/bin/env infinity-env

# vlastni skript
```

Spuštění úlohy

Úlohu spouštíme **v adresáři úlohy** příkazem **psubmit**.

```
psubmit destination job [resources] [syncmode]
```

destination (kam) je buď:

- název_fronty
- název_uzlu@název_fronty

job je buď:

- název skriptu úlohy
- název vstupního souboru pro automaticky rozpoznávané úlohy

resources jsou požadované zdroje pro úlohu, pokud není uvedeno, požaduje se běh na 1 CPU

syncmode určuje způsob kopírování dat mezi adresářem úlohy a výpočetním uzlem, výchozím módem je "sync"

Monitorování běhu úlohy

K monitorování průběhu úlohy lze použít příkaz **pinfo**, který se spouští buď v adresáři úlohy nebo v pracovním adresáři na výpočetním uzlu. Dalšími možnostmi jsou příkazy **pjobs** a **pqstat**.

Pokud je úloha spuštěna na výpočetním uzlu, je možné použít příkaz **pgo**, který se naloguje na výpočetní uzel a změní aktuální adresář do pracovního adresáře úlohy.

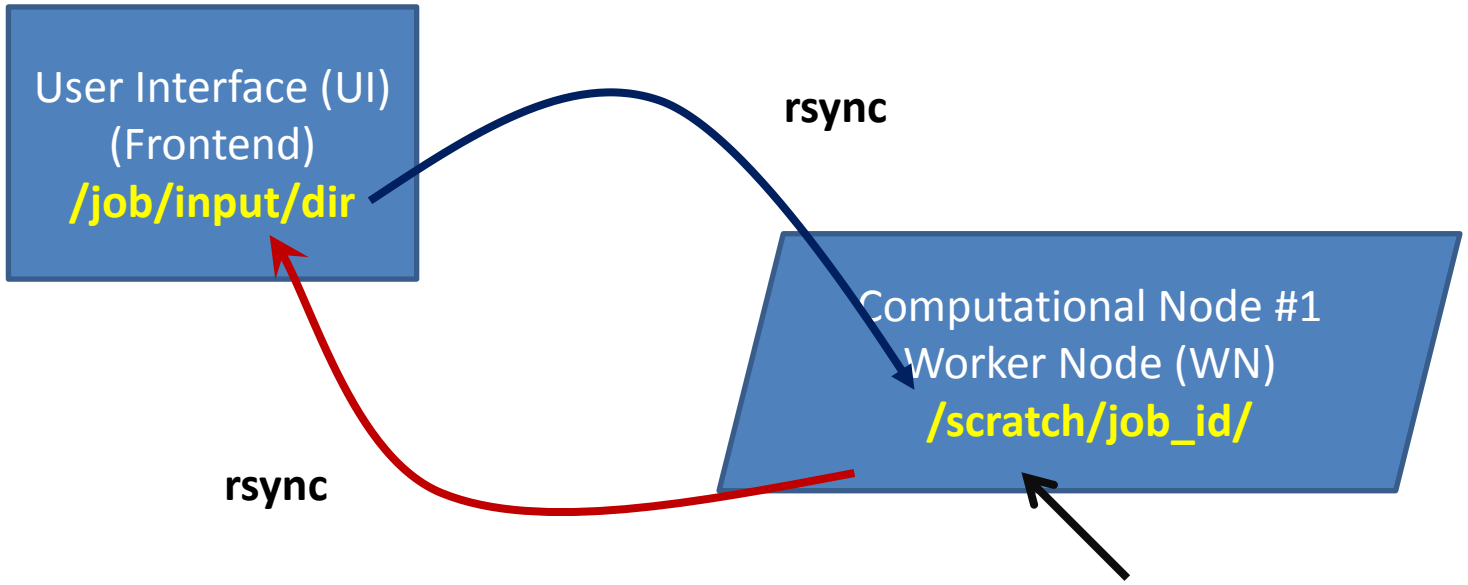
Servisní soubory

V adresáři úlohy vznikají při zadání úlohy do dávkového systému a dále v průběhu života úlohy a po jejím ukončení servisní soubory. Jejich význam je následující:

- *.info kontrolní soubor s informacemi o průběhu úlohy
- *.infex vlastní skript (wrapper), který se spouští dávkovým systémem
- *.infout standardní výstup z běhu *.infex skriptu, **nutno analyzovat při nestandardním ukončení úlohy**
- *.nodes seznam uzlů vyhrazených pro úlohu
- *.gpus seznam GPU karet vyhrazenych pro úlohu
- *.key unikátní identifikátor úlohy
- ***.stdout** **standardní výstup z běhu skriptu úlohy**

Synchronization modes, sync

Mode	Meaning
sync	Data are copied from the job input directory to the working directory on the computational node. The working directory is created on the scratch of the computational node. After the job is finished, all data from the working directory are copied back to the job input directory. Finally, the working directory is removed if the data transfer was successful.



Note: default synchronization mode

determined by **scratch_type** resource token

Spouštíme aplikace

pmemd

pmemd je program určen pro molekulovou dynamiku. Podrobnější informace lze nalézt zde: http://ambermd.org

```
#!/bin/bash

# aktivovat modul amber obsahujici aplikace
# sander a pmemd
module add amber

# spusteni aplikace
pmemd –O –i prod.in –p topology.parm7 \
                      -c input.rst7
```

pmemd

Délka simulace:
Délka simulace (výpočtu) je určena klíčovým slovem (**nstlim**) uvedeným v souboru prod.in, který určuje počet integračních kroků.

Výsledkem simulace jsou soubory:

mdout
mdinfo <-- obsahuje statistické informace, např. kolik ns za den je program schopen nasimulovat
mdcrd
restrt

pmemd – paralelní běh

Při paralelním spouštění se mění jen zadání zdrojů u příkazu psubmit. **Ostatní se nemění!** (zůstávají stejná vstupní data a skript úlohy).

```
$ psubmit short test_sander ncpus=1
```

může se vynechat

***.stdout**

```
.....
Module build: amber:12.0:x86_64:single
.....
```

Výpočetní uzel:

```
S %CPU %MEM     TIME+  COMMAND
R  100  0.6   1:13.37 sander
R    0  0.0   0:00.01 top
```

```
$ psubmit short test_sander ncpus=2
```

***.stdout**

```
.....
Module build: amber:12.0:x86_64:para
.....
```

Výpočetní uzel:

```
%CPU %MEM     TIME+  COMMAND
 100  1.6   0:40.41 sander.MPI
  99  1.7   0:40.60 sander.MPI
```

Cvičení

Cvičení 1

Cílem cvičení je určit jak dobře škáluje aplikace pmemd v rozsahu počtu CPU, které jsou násobky dvou. Určete skutečnou a teoretickou délku výpočtu, reálné urychlení a reálné využití CPU v procentech. Do grafu vyneste reálné urychlení jako funkci počtu CPU. Nalezenou křivku porovnejte s křivkou pro ideální škálování.

Testovací výpočty můžete provádět na klastru WOLF. Finální výpočty pak budete provádět v metacentru na výpočetních uzlech stejného klastru.

Vstupní data jsou na klastru WOLF v adresářích:
/home/kulhanek/Data/C2115/Lesson13/pmemd