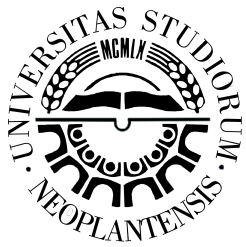
Хибридно програмирање

Рачунарски системи високих перформанси

Горана Гојић Вељко Петровић

Факултет техничких наука
Универзитет у Новом Саду

Рачунарске вежбе, Зимски семестар 2020/2021.

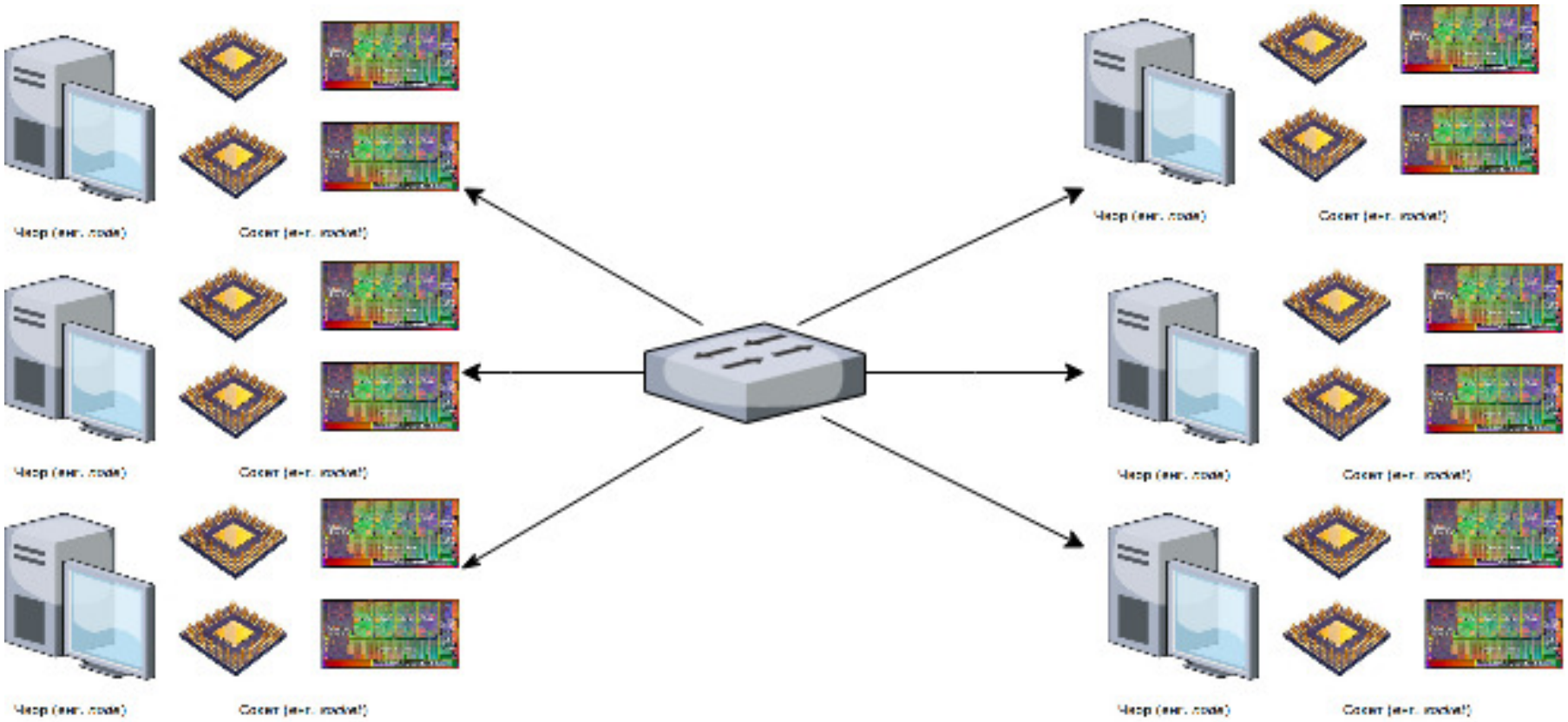


Хибридно програмирање

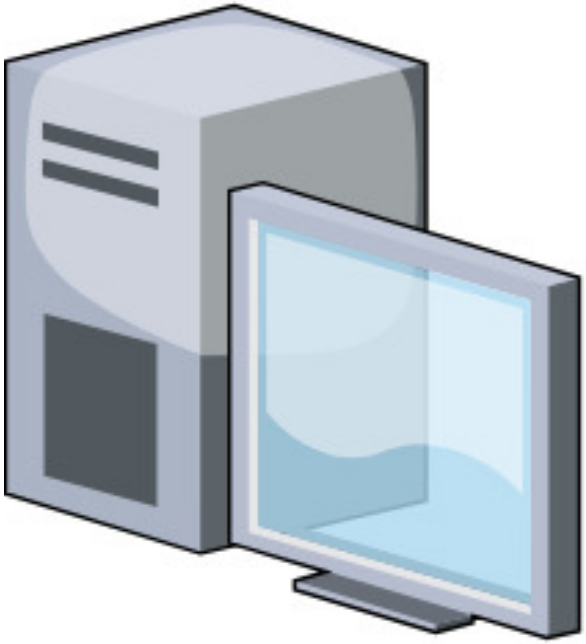
Подразумева комбиновано коришћење више различитих програмских модела.

У овом случају то су OpenMP и OpenMPI како би се искористила два различита нивоа паралелизма.

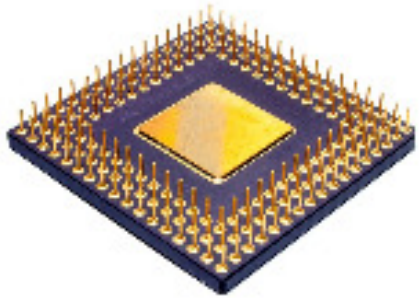
Циљна архитектура

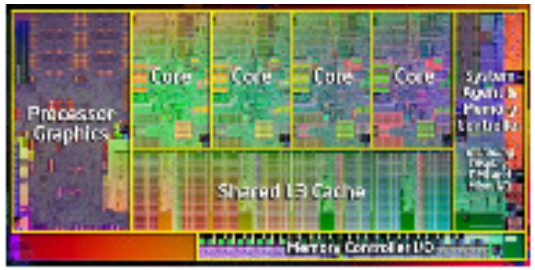


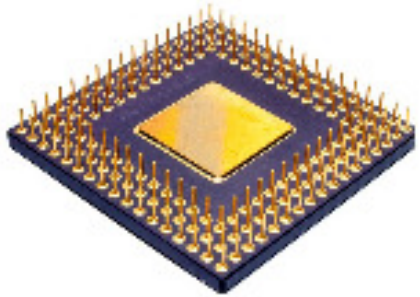
Циљна архитектура

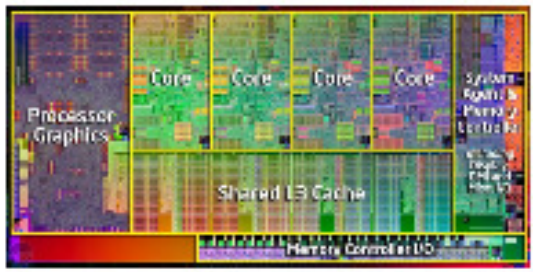


Чвор (енг. *node*)









Сокет (енг. *socket*)

Компајлирање хибридних OpenMP-OpenMPI програма

Позиционирати се у директоријум у којем се налази изворни код хибридног програма и покренути:

```
mpicc <izvorna_datoteka> -fopenmp
```

Покретање:

```
OMP_NUM_THREADS=<Nmp> \
mpiexec [-np <Nmpi>] <izvrsna_datoteka>
```

`-np <Nmpi>` - опција за задавање броја процеса OpenMPI процеса.
`OMP_NUM_THREADS` = <Nmp> - максималан број OpenMP нити по MPI процесу

Пример 1: Hello World!

```
int main(int argc, char *argv[]) {
    int rank;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MPI_COMM_WORLD, &rank);

    #pragma omp parallel
    {
        printf("Hello World iz procesa %d i niti %d.\n",
                rank, omp_get_thread_num());
    }

    MPI_Finalize();

    return 0;
}
```

MPI таксонимија интероперабилности нити

Нивои интероперабилности по стандарду MPI 3.1:

- `MPI_THREAD_SINGLE` - Само једна нит у MPI процесу.
- `MPI_THREAD_FUNNELED` - MPI процес може имати више нити, али MPI позиве може извршавати само главна нит.
- `MPI_THREAD_SERIALIZED` - MPI процес може имати више нити и све нити могу извршавати MPI позиве, али позиви не могу бити извршавани конкурентно из две различите нити, него се серијализују.
- `MPI_THREAD_MULTIPLE` - MPI процес може имати више нити, нити могу извршавати MPI позиве без ограничења.

Директна размена порука подржана само између процеса - не може се експлицитно адресирати нит.

MPI таксонимија интероперабилности нити

Немају све MPI имплементације исти ниво подршке за рад са више нити. `MPI_init_thread` уместо `MPI_Init`.

```
int MPI_Init_thread(int *argc, char ***argv,
                    int required, int *provided)
```

(улазни параметар)

`required` := `MPI_THREAD_SINGLE` | `MPI_THREAD_FUNNELED` | `MPI_THREAD_SERIALIZED` | `MPI_THREAD_MULTIPLE`

(излазни параметар)

`provided` := `MPI_THREAD_SINGLE` | `MPI_THREAD_FUNNELED` | `MPI_THREAD_SERIALIZED` | `MPI_THREAD_MULTIPLE`

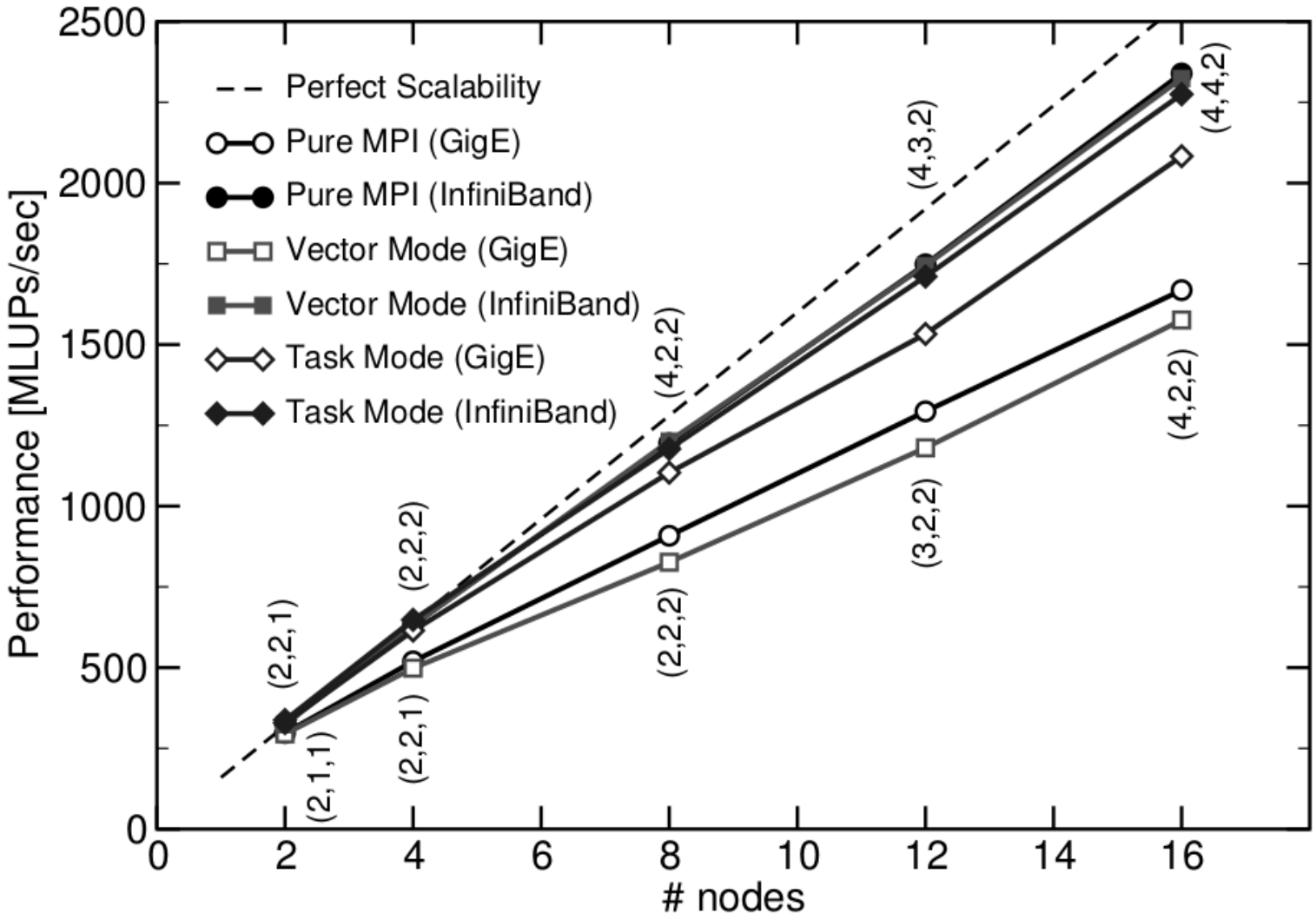
[1] MPI_Init
[2] MPI_Init_thread

Перформансе хибридног решења

"Consider going hybrid only if pure MPI scalability is not satisfactory."

[1] Преузето из књиге *"Introduction to High Performance Computing for Scientists and Engineers"*

Поређење перформанси извршавања Јакоби алгоритма



[1] Преузето из књиге *"Introduction to High Performance Computing for Scientists and Engineers"*

Перформансе хибридног решења

... веома зависе од броја процеса, броја нити у оквиру процеса, MPI и MP имплементације, архитектуре на којој се хибридни програм покреће, саме имплементације решења, итд. ...

Предности и мане хибридних решења

- Боље искоришћење кеша.
- Експлоатисање додатних нивоа паралелизма у односу на чисто MPI решење.
- Смањење времена извршавања преклапањем комуникације и рачунања.
- ...

- МНОГО захтевније за писање - више посла око осмишљавања решења, писања кода без штетног преплитања, много више потенцијалних места за прављење неефикасног решења услед коришћења два различита програмска модела.
- Често је немогуће инкрементално направити хибридно решење од нехибридног решења - захтева писање решења од почетка.
- ...

Алтернативне технологије

- MPI за комуникацију између чворова, MPI 3.0 модел за дељену меморију
- MPI за комуникацију између чворова, `pthreads` модел за дељену меморију
- ...

Задатак 1: Рачунање броја π

Имплементирати чисто OpenMPI и хибридно OpenMP-OpenMPI решење за рачунање броја π рачунањем вредности интеграла

$$\int_0^1 \frac{4}{(1+x^2)}$$

Секвенцијална и OpenMP верзија програма су дате у директоријуму `resenja`.

- Поредити време извршавања хибридног решења са временом извршавања OpenMP убрзаног решења.
- Мерити време извршавања хибридног решења за различите комбинације броја OpenMPI процеса и OpenMP нити.

Препорука: имплементирати OpenMPI верзију програма на основу секвенцијалног решења, па је проширити OpenMP директивама.

Задатак 2: Претрага низа бројева

Имплементирати секвенцијално и хибридно OpenMPI-OpenMP решење за претрагу низа целих бројева минималне дужине милион елемената. Елементе низа генерисати насумично из интервала [-100, 100].

- Мерити време извршавања хибридног решења за различите комбинације броја OpenMPI процеса и OpenMP нити.
- Опционо имплементирати чисто OpenMPI и чисто OpenMP решење и поредити перформансе ових решења поредити са перформнсама хибридног решења

Препорука: имплементирати OpenMPI верзију програма на основу секвенцијалног решења, па је проширити OpenMP директивама.

Задатак 3: Множење матрица - **домаћи**

Имплементирати хибридно OpenMP-OpenMPI решење у C програмском језику за множење две квадратне матрице на основу OpenMPI и OpenMP решења задатака са претходних вежби. Претпоставити да један MPI процес дистрибуира делове матрице преосталим процесима. Костур решења који је потребно попунити се налази у директријуму `MatrixMultiplicationHybrid`.

- Резултујућу матрицу сачувати у h5 формату у датотеци под називом `result<nxn>.h5`, где се <nxn> мења димензијама матрице која представља решење.
- Мерити време извршавања хибридног, OpenMP и OpenMPI решења и забележити их у приложену датотеку statistika.csv.

Задатак 3: Множење матрица - **домаћи**

- Мерити време извршавања хибридног решења за различите комбинације броја MPI процеса и OpenMP нити. У случају да се решење испробава на рачунару са сокетом и више језгара, пробати следеће комбинације:
  - По један MPI процес за свако логичко језгро
  - По један MPI процес за свако физичко језгро и по једна нит за свако логичко језгро.
  - Један MPI процес и по једна нит за свако логичко језгро.

  По жељи додати још различитих конфигурација и резултате уписати у `statistika.csv`. Анализирати добијене резултате.

При анализи перформанси решења може помоћи поглавље 11 књиге *"Introduction to high performance computing for Scientists and Engineers."*

Материјали

- Georg Hager, Gerhard Wellein, "Introduction to High Performance Computing for Scientists and Engineers"
- MPI 3.1 стандард, поглавље 12.4
- OpenMP документација
- OpenMP SC13 Tutorial: Hybrid MPI and OpenMP Parallel Programming