OpenMP

Programowanie współbieżne... (7)

Andrzej Baran 2010/11

LINK: http://kft.umcs.lublin.pl/baran/prir/index.html

OpenMP

OpenMP

http://openmp.org/wp/

OpenMP™

THE OPENMP API SPECIFICATION FOR PARALLEL PROGRAMMING

Subscribe to the News Feed

»» OpenMP Specifications
»About OpenMP
»Compilers
»Resources
»Discussion Forum

Events
»IWOMP 2011 Call For Papers (pdf) - 7th International Workshop on OpenMP, June 13 - 15, 2011, Chicago USA

OpenMP News

»IWOMP 2011 - Call For Papers

Call for Papers 7th International Workshop on OpenMP IWOMP 2011
June 13 – 15, 2011 Chigaco, USA
http://www.iwomp.org/

The **2011 International Workshop on OpenMP (IWOMP 2011)** will be held in Chicago, IL. It is the premier forum to present and discuss issues, trends, recent research ideas and results related to parallel pro- gramming with OpenMP. The international workshop affords an opportunity for OpenMP users as well as developers to come together for discussions and sharing new ideas and information on this topic. IWOMP 2011 will be a three-day event. The first day will consist of tutorials focusing on topics of interest to current and prospective OpenMP developers, suitable for both beginners as well as those interested in learning of recent developments in the evolving OpenMP standard. The second and third days will consist of technical papers and panel session(s) during which research ideas and results will be presented and discussed. **(more...)**

Posted on September 2, 2010

The
suppor
memor
in C/C+
is a po
with a s
interfac
applica
the des
superc
»Read

Get
»Open

Use

Literatura (podstawa opracowania):

Hermanns, M. (2002). Parallel Programming in Fortran 95 using OpenMP. [pdf] http://link.aip.org/link/?CPHYE2/8/117/1

Treść

- **Wstęp**
- **Dyrektywy**
  - **Klauzule, warunki**
- **Funkcje i procedury czasu wykonania**
- **Zmienne środowiskowe**

OpenMP

OpenMP

- Pojęcie dyrektywy OpenMP
- Obszar równoległy
- Warunki wyboru (clause)
- Przykłady programów
- Omówienie podstawowych dyrektyw

Dyrektywy - Fortran

- Dyrektywy OpenMP rozpoczynają się w Fortranie 95 od !$OMP; ! oznacza komentarz i dyrektywa jest pomijana w przypadku gdy program chcemy wykonać sekwencyjnie.
- W starszym wydaniu fortranu mamy: C$OMP lub *$OMP
- Podobnie !$ ma specjalne znaczenie w kompilacji równoległej programu - oznacza kompilację warunkową
- Struktura dyrektywy:

  !$OMP dyrektywa warunki

  !$OMP end dyrektywa warunki

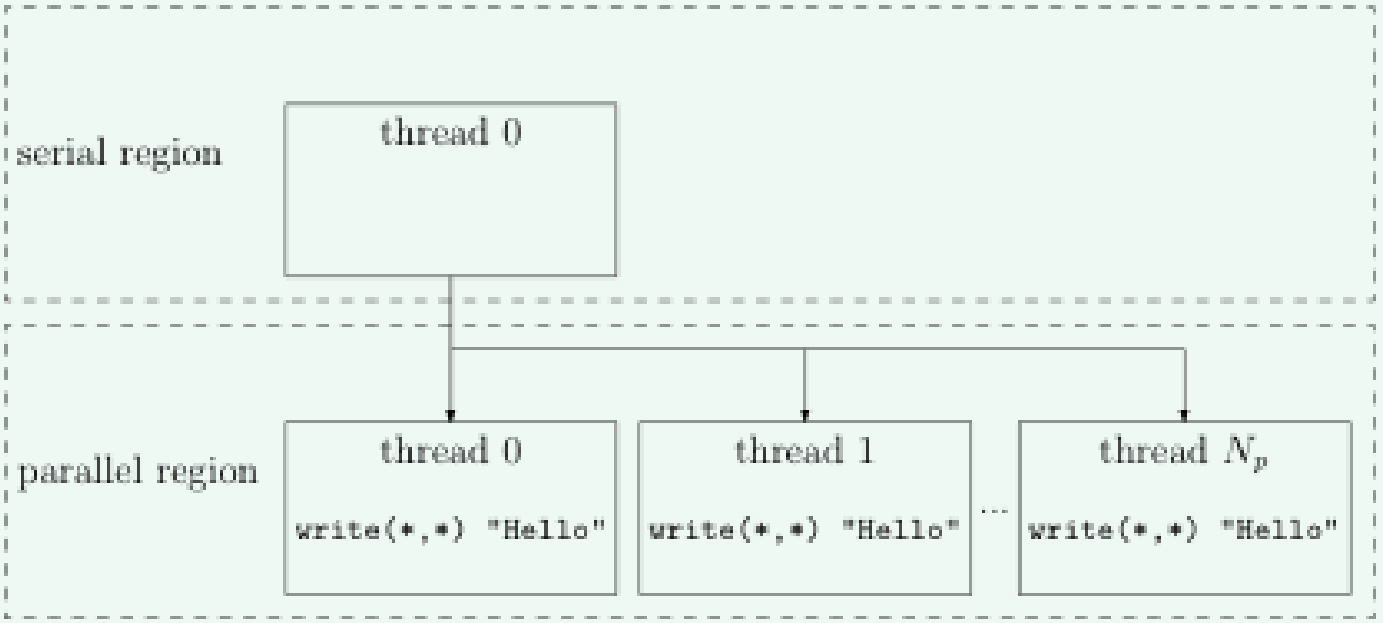
!$omp parallel

- Przyklad prostego konstruktora obszaru równoległego wykonania

```
!$omp parallel
        write(*,*) ‘Hello’
!$omp end parallel
```

Jeśli liczba procesów wynosi np. 4 to słowo „Hello” zostanie wydrukowane 4 razy.

Obszar równoległy



Kompilacja warunkowa

- Przykład kompilacji warunkowej

```
!$ interval = L*OMP_get_thread_num()/ &
!$            (OMP_get_num_threads()-1)
```

Znak &(ampersand) jest znakiem kontynuacji.

Dyrektywy zagnieżdżone

- Przykład

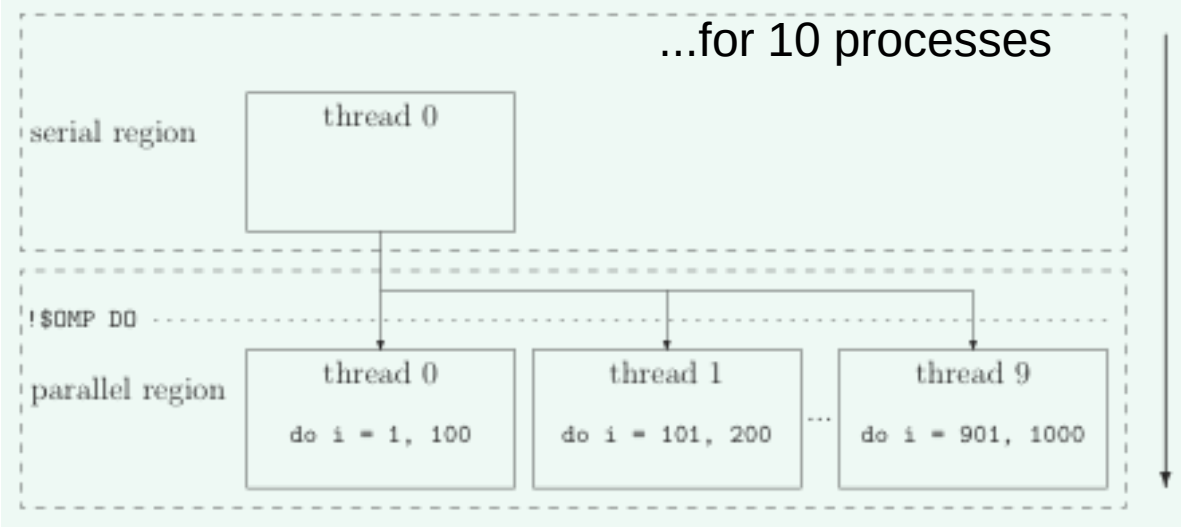
```
!$omp parallel
  write(*,*) ‘Hello’
  !$omp parallel
    write(*,*) ‘Hi!’
  !$omp end parallel
!$omp end parallel
```

Jeśli liczba procesów wynosi N to pojawi się $N^2 + N$ napisów. Dlaczego?

!$omp do

- Przykład

```
!$omp do
  do i=1, 1000
    ...
  end do
!$omp end do
```



!$omp do

- Niebezpieczeństwa

```
dimension a(100), b(100)
do i=1, 100
    b(i) = 11 * a(i)
    a(i) = a(i) + b(i)
end do
```

Tutaj b(i) nie jest określone aż do momentu wykonania !$omp end do

!$omp do

- Niebezpieczeństwa

```
dimension a(100), b(100)
do i=1, 99
 a(i) = a(i+1)
end do
```

Wyniku działania nie można tutaj przewidzieć. W chwili wykonywania operacji o numerze i element a(i+1) nie musi być w stanie pierwotnym – mógł być już zmieniony (*racing*)

rozwiązanie

```
! MOŻLIWE ROZWIĄZANIE PROBLEMU…
real(8) :: A(1000), dummy(2:1000:2)
!Saves the even indices
!$OMP DO
do i = 2, 1000, 2
        dummy(i) = A(i)
enddo
!$OMP END DO
!Updates even indices from odds
!$OMP DO
do i = 0, 998, 2
        A(i) = A(i+1)
enddo
!$OMP END DO
```

```
!Updates odd indices with evens
!$OMP DO
     do i = 1, 999, 2
        A(i) = dummy(i+1)
     end do
!$OMP END DO
```

!$omp do (czy optymalnie...?)

```
do i = 1, 10
  do j = 1, 10
    !$OMP DO
      do k = 1, 10
        A(i,j,k) = i * j * k
      enddo
    !$OMP END DO
  enddo
enddo
```

```
!$OMP DO
  do i = 1, 10
    do j = 1, 10
      do k = 1, 10
        A(i,j,k) = i * j * k
      enddo
    enddo
  enddo
!$OMP END DO
```

Najlepszym jest rozwiązanie pokazane z prawej strony. Dlaczego?

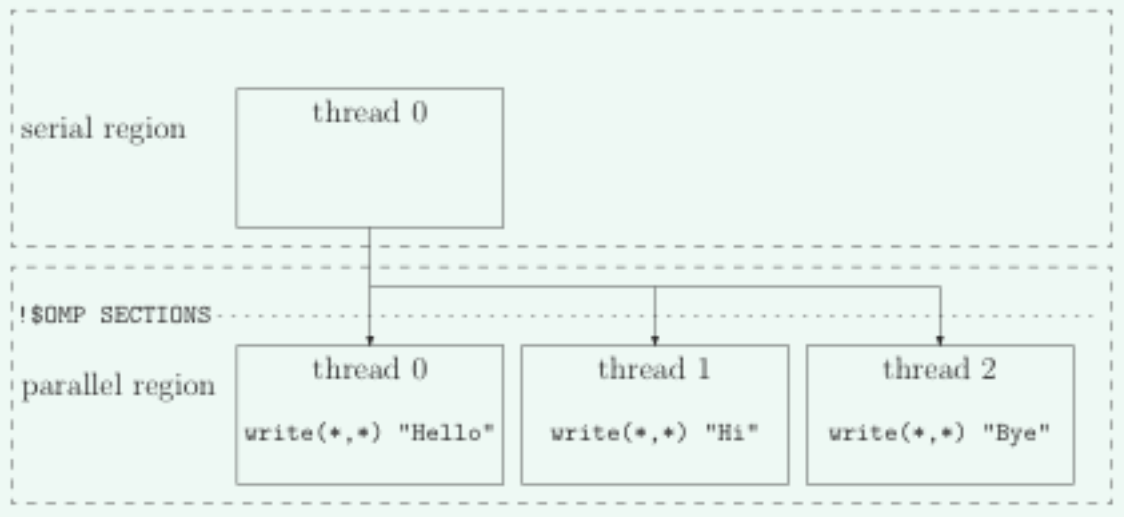
Tablice w fortranie są umieszczane w pamieci komputera „KOLUMNAMI”.

```
!$OMP DO
  do k = 1, 10
    do j = 1, 10
      do i = 1, 10
        A(i,j,k) = i * j * k
      enddo
    enddo
  enddo
!$OMP END DO
```

!$omp sections / ...end...

- ‘sections’ używa się tam gdzie mamy niezależne częsci programu
- Przykład

```
!$OMP SECTIONS
!$OMP SECTION
  write(*,*) "Hello"
!$OMP SECTION
  write(*,*) "Hi"
!$OMP SECTION
  write(*,*) "Bye"
!$OMP END SECTIONS
```



!$omp single / ...end...

- Para !$omp single and !$omp end single oznacza obszar, w którym może pracować tylko jeden proces, pierwszy;

!$omp workshare / ...end...

- Działanie funkcji wewnętrznych Fortranu, które pracują na tablicach:

  matmul, dot product, sum, product, maxval, minval, count, any, all, spread, pack, unpack, reshape, transpose, eoshift, cshift, minloc and maxloc

  można zrównoleglić używając pary dyrektyw:

  !$omp workshare / !$omp end workshare

!$omp workshare / ...end...

OpenMP

33

The Workshare construct

Fortran has a fourth worksharing construct:

```
!$OMP WORKSHARE

    <array syntax>

!$OMP END WORKSHARE [NOWAIT]
```

Example:

```
!$OMP WORKSHARE
    A(1:M) = A(1:M) + B(1:M)
!$OMP END WORKSHARE NOWAIT
```

!$omp workshare / ...end...

Przykład, w którym dyrektywa “DO” nie może być użyta (Dlaczego?):

```
!$OMP DO
      do i = 1, 1000
        B(i) = 10 * i
        A(i) = A(i) + B(i)
      end do
!$OMP END DO
```

...ŹLE

można teraz zastąpić takim oto rozwiązaniem:

```
!$OMP WORKSHARE
      forall(i=1:1000)
        B(i) = 10 * i
      end forall
      A = A + B
!$OMP END WORKSHARE
```

DOBRZE...

Kombinacje dyrektyw

- !$OMP PARALLEL DO / !$OMP END PARALLEL DO
- !$OMP PARALLEL SECTIONS / !$OMP END PARALLEL SECTIONS
- !$OMP PARALLEL WORKSHARE/!$OMP END PARALLEL WORKSHARE

!$omp master

Para !$omp master / !$omp end master pozwala pracować w danym obszarze tylko procesowi głównemu (master process lub root)

!$omp critical / ...end...

- W danym czasie, w obszarze pary dyrektyw „critical” (!$omp critical/!$omp end critical) może działać tylko jeden proces (który pierwszy się w nim znajdzie). Ma to zastosowanie w przypadku np. czytania/drukowania (input/output).

  !$OMP CRITICAL name

  ...

  !$OMP END CRITICAL name

  Tutaj name jest opcjonalną nazwą dla obszaru krytycznego – sekcje posiadające taką samą nazwą są traktowane jak wspólna sekcja krytyczna.

!$omp barrier

- Dyrektywa „barrier”oznacza miejsce, w którym wymaga się synchronizacji wszystkich procesów (procesy czekają na siebie)
- Należy uważać by nie używać konstrukcji, które prowadzą do śmierci (zakleszczenia; deadlock) programu. Np. konstrukcja

```
!$OMP CRITICAL
        !$OMP BARRIER
!$OMP END CRITICAL
```

prowadzi do sytuacji gdzie jeden wątek czeka na pozostałe, ale te nie mogą wejść do sekcii „critical”

!$omp barrier

- Inne sytuacje DEADLOCK:

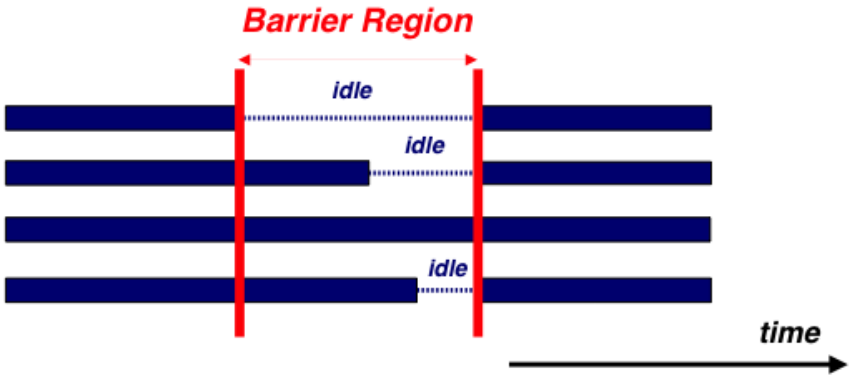
```
!$OMP SINGLE
        !$OMP BARRIER
!$OMP END SINGLE

!$OMP MASTER
        !$OMP BARRIER
!$OMP END MASTER

!$OMP SECTIONS
        !$OMP SECTION
                !$OMP BARRIER
        !$OMP SECTION
                ...
        ...
!$OMP END SECTIONS
```

!$omp barrier

OpenMP

26

Barrier/3



Barrier syntax in OpenMP:

```
#pragma omp barrier
```

```
!$omp barrier
```

!$omp atomic

- Dyrektywa ogranicza dostęp do operacji na obiekcie tylko do jednego wątku w danym czasie. Procesy pojedynczo ją wykonuja. Przykład:

```
!$OMP DO
    do i = 1, 1000
        a = a + I     ! x = x operator expr
    end do
!$OMP END DO
```

Zmienna a może być uzupełniana tylko przez jeden proces w danym momencie… !$omp atomic może być użyta z operatorami +, *, -, /, .and., …, Max, Min, Iand, Ior, Ieor.

!$omp flush

- Używamy w miejscu, w którym wymaga się by wszystkie wątki widziały to samo, te same wartości zmiennych wspólnych (dzielonych przez wątki), w danym momencie.

Zadanie. Opisz dokładnie jak działa i gdzie jest stosowana dyrektywa !$omp flush.

Synchronizacja...

- Zazwyczaj synchronizacja odbywa się automatycznie. Jest tak w przypadku dyrektyw:
  - !$OMP BARRIER
  - !$OMP CRITICAL and !$OMP END CRITICAL
  - !$OMP END DO
  - !$OMP END SECTIONS
  - !$OMP END SINGLE
  - !$OMP END WORKSHARE
  - !$OMP ORDERED and !$OMP END ORDERED
  - !$OMP PARALLEL and !$OMP END PARALLEL
  - !$OMP PARALLEL DO and !$OMP END PARALLEL DO
  - !$OMP PARALLEL SECTIONS and !$OMP END PARALLEL SECTIONS
  - !$OMP PARALLEL WORKSHARE and !$OMP END PARALLEL WORKSHARE

Synchronizacja

- Dyrektywy, które nie wymuszają synchronizacji:
  - !$OMP DO
  - !$OMP MASTER and !$OMP END MASTER
  - !$OMP SECTIONS
  - !$OMP SINGLE
  - !$OMP WORKSHARE

- Jeśli dodano w dyrektywie warunek (clause) NOWAIT wówczas synchronizacja wymuszona zostaje wyłączona!

!$omp ordered/...end...

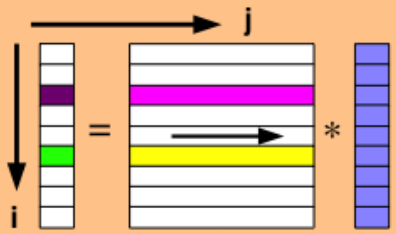
- Zapewnia prawidłowy porządek wykonania...

Klauzule, warunki

- Większość dyretyw OpenMP dopuszcza dodatkowe warunki, tzw. klauzule (clauses)
  - Warunki te pozwalają sprecyzować działanie dyrektywy
- Przykładowo private(a) jest warunkiem dla dyrektywy do
  - !$omp do private(a)
- Konkretne warunki, których można używać zależą od dyrektywy

Example - Matrix times vector


```
#pragma omp parallel for default(none) \
            private(i,j,sum) shared(m,n,a,b,c)
for (i=0; i<m; i++)
{
   sum = 0.0;
   for (j=0; j<n; j++)
     sum += b[i][j]*c[j];
   a[i] = sum;
 }
```



TID = 0

```
for (i=0,1,2,3,4)
 i = 0
   sum =    b[i=0][j]*c[j]
   a[0] = sum
 i = 1
   sum =    b[i=1][j]*c[j]
   a[1] = sum
```

TID = 1

```
for (i=5,6,7,8,9)
 i = 5
   sum =    b[i=5][j]*c[j]
   a[5] = sum
 i = 6
   sum =    b[i=6][j]*c[j]
   a[6] = sum
```

... etc ...

Podsumowanie

- Krótkie podsumowanie OpenMP (pliki pdf)
  - http://www.openmp.org/mp-documents/OpenMP3.0-FortranCard.pdf
  - http://www.openmp.org/mp-documents/OpenMP3.0-SummarySpec.pdf
- Dobry przegląd dyrektyw OpenMP można znaleźć na stronie Sun Microsystems (OpenMP API User's Guide):
  - http://docs.sun.com/source/819-0501/index.html

Problemy...?