OpenMP 1

• Introduction to OpenMP
• Compute Pi example
• OpenMP directives
Introduction to OpenMP

- A set of compiler directives and library routines to support parallel programming in a shared memory architecture
- Can be used with C, C++ and Fortran
- A program created with OpenMP starts with a master thread
- Various directives cause the program to create and execute multiple threads
Introduction to OpenMP

Master Thread

Additional Threads
Execute

New Threads created with 
#pragma omp parallel

Threads joined
Only master thread remains
Introduction to OpenMp

• Directives have the following form
  – #pragma omp construct [clauses]

• Directives are applied to a structured block of code
  – A structured block has one entry point (at the top) and one exit point (at the bottom)
Introduction to OpenMp

• Supports explicit parallelism
• Programmer needs to specify where parallel threads should be used
• Programmer needs to think about synchronization issues
Simple Example

• Sequential algorithm to calculate pi
  – Find the area under the curve $4/(1+x^2)$ between 0 and 1.
  – Example of numerical integration

• Parallelization of the algorithm using OpenMP
Compute Pi
int main(int argc, char *argv[]) {
    double midpoint, width, pi, sum;
    int numIntervals, i;
    numIntervals = atoi(argv[1]);
    width = 1.0 / numIntervals;
    pi = 0.0;
    for(i = 0; i < numIntervals; i++) {
        midpoint = (i + 0.5) * width;
        sum = sum + 4.0 / (1.0 + midpoint * midpoint);
    }
    pi = width * sum;
    printf("%1.20f \n", pi);
}
Parallelized Algorithm to Compute Pi

```c
int main(int argc, char *argv[]) {
    double midpoint, width, pi, sum;
    int numIntervals, I;
    numIntervals = atoi(argv[1]);
    width = 1.0 / numIntervals;
    pi = 0.0;
    #pragma omp parallel for default(shared) private(i, midpoint) reduction(+:sum)
    for(i = 0; i < numIntervals; i++) {
        midpoint = (i + 0.5) * width;
        sum = sum + 4.0 / (1.0 + midpoint * midpoint);
    }
    pi = width * sum;
    printf("%1.20f
", pi);
}
```
OpenMP Directives

- #pragma omp parallel
- #pragma omp for
- #pragma omp sections
### #pragma omp parallel

#### Example

```c
void printId(int myId) {
    printf("%d\n", myId);
}

int main(int argc, char *argv[]) {

    #pragma omp parallel
    {
        printId(omp_get_thread_num());
    }
}
```
#pragma omp for

int main(int argc, char *argv[]) {
    int i;
    #pragma omp parallel for num_threads(4)
    for (i = 0; i < 10; i++) {
        printf("%d %d\n",omp_get_thread_num(),i);
    }
}

#pragma omp sections

int main(int argc, char *argv[]) {
    #pragma omp parallel num_threads(4)
    {
        #pragma omp sections
        {
            #pragma omp section
            {printf("%d\n",omp_get_thread_num());}
            #pragma omp section
            {printf("%d\n",omp_get_thread_num());}
            #pragma omp section
            {printf("%d\n",omp_get_thread_num());}
            #pragma omp section
            {printf("%d\n",omp_get_thread_num());}
        }
    }
}
}