Exercises for Chapter 7 "Programming Shared Address Space Platforms"

March 18, 2013

Exercise 1

For each of the following code segments, use OpenMP pragmas to make the loop parallel, or explain why the code segmenet is not suitable for parallel execution.

a.

```
for (i=0; i < (int) sqrt(x); i++) {
  a[i] = 2.3 * x;
  if (i < 10) b[i] = a[i];
}
```

b.

```
flag = 0;
for (i = 0; (i<n) & (!flag); i++) {
  a[i] = 2.3 * i;
  if (a[i] < b[i]) flag = 1;
}
```

c.

```
for (i = 0; i < n; i++)
  a[i] = foo(i);
```

d.

```
for (i = 0; i < n; i++) {
  a[i] = foo(i);
  if (a[i] < b[i]) a[i] = b[i];
}
```

e.

```
for (i = 0; i < n; i++) {
  a[i] = foo(i);
  if (a[i] < b[i]) break;
}
```

f.

```
dotp = 0;
for (i = 0; i < n; i++)
  dotp += a[i] * b[i];
```

g.

```
for (i = k; i < 2*k; i++)
  a[i] = a[i] + a[i-k];
```

h.

```
for (i = k; i < n; i++)
  a[i] = b * a[i-k];
```

Exercise 2

Suppose OpenMP did not have the `reduction` clause. Show how to implement an efficient parallel reduction by adding a private variable and using the `critical` pragma.

Exercise 3

Write a simple C code to compute the inner-produce of two very long vectors. Use `#pragma omp parallel for` to do the parallelization. Choose different schedulers and chunksizes and observe the time usage.