**Blue Waters Petascale Semester Curriculum v1.0**

**Unit 4: OpenMP  
Lesson 10: Ensemble Based Simulated Annealing in OpenMP  
Sample Assessment***Developed by David A. Joiner for the Shodor Education Foundation, Inc.*



*Except where otherwise noted, this work by The Shodor Education Foundation, Inc. is licensed under CC BY-SA 4.0. To view a copy of this license, visit*[*https://creativecommons.org/licenses/by-sa/4.0*](https://creativecommons.org/licenses/by-sa/4.0)

*Browse and search the full curriculum at*[*http://shodor.org/petascale/materials/semester-curriculum*](http://shodor.org/petascale/materials/semester-curriculum)

*We welcome your improvements! You can submit your proposed changes to this material and the rest of the curriculum in our GitHub repository at*[*https://github.com/shodor-education/petascale-semester-curriculum*](https://github.com/shodor-education/petascale-semester-curriculum)

*We want to hear from you! Please let us know your experiences using this material by sending email to* [*petascale@shodor.org*](mailto:petascale@shodor.org)

For the following code, identify and describe the race condition. How would you protect this race condition if parallelizing the loop in OpenMP

int main(int argc, char \*\* argv) {

int i;

int sum = 0;

for(i=0;i<100;i++) {

sum += i;

}

printf("sum = %d\n",sum);

}

For the following code, identify and describe the loop carried dependency.

#define N 10

int main(int argc, char \*\* argv) {

int i;

int fib[N];

fib[0] = 1;

fib[1] = 2;

for(i=2;i<N;i++) {

fib[i]=fib[i-1]+fib[i-2];

}

printf("fib[N-1] = %d\n",fib[N-1]);

}

The Ensemble Based Simulated Annealing (EBSA) Algorithm is modified to avoid a loop carried dependency by replacing one annealer taking many steps with many annealers taking fewer steps. Describe how this makes EBSA a different algorithm from SA.