**Question 1:**

1. 1 iteration – Single Processor

1000 nesc

1 iteration – Two Processors

500 + M nsec,

M = 100 + 10x

Speedup – S = TS(N)/TP(N) > 1

1000 / (500 + 100 + 10x) > 1

1000 > 500 + 100 + 10x

400 > 10x

X < 40

1. Efficiency – E = S/p > 0.7

[1000 / (500 + 100 + 10x)] / 2 > 0.7

1000 / 2\*(500 + 100 + 10x) > 0.7

1000 > 1.4\*(500 + 100 + 10x)

1000 > 700 + 140 + 14x

160 > 14x

x < 11.4286

1. x = 20

S?

E?

S:

1000 / (500 + 100 + 10\*20)  
 1000 / (500 + 100 + 200)  
 1000 / 800

S = 1.25

E:

1.25 / 2

E = 0.625

1. With Buffer:

Odd iterations – O

Even iterations – E

x = 20

Sbuffer-O:

1000 / 500 = 2

S buffer-O = 2

S buffer-E:

1000 / (500 + 100 + 2\*10x)  
 1000 / (500 + 100 + 2\*200)  
 1000 / (500 + 100 + 400)  
 1000 / 1000  
 1

S buffer-E = 1

Sbuffer-average

1O + 1E / 2  
 SO + SE / 2  
 2 + 1 / 2  
 1.5

Sbuffer-avarage = 1.5

Sbuffer-avarage > Sc (1.5 > 1.25)

**Question 2:**

|  |  |
| --- | --- |
| Thread 1 | Thread 2 |
| if (myrank % 2 == 0) **[TRUE]** |  |
| pthread\_mutex\_lock(&M1); |  |
| insert(L1, myrank); |  |
| *Context Switch out* | *Context Switch in* |
|  | if (myrank % 2 == 0) **[FALSE]** |
|  | pthread\_mutex\_lock(&M2); |
|  | insert(L2, myrank + 1); |
|  | pthread\_mutex\_lock(&M1); **[WAITING]** |
| *Context Switch in* | *Context Switch out* |
| pthread\_mutex\_lock(&M2); **[WAITING]** |  |
| **[DEADLOCK]** | |

1. Yes. Mutexes are similar to busy-waiting in operation. It would occur in a deadlock too
2. One possible solution is to add more one mutex

if (myrank % 2 == 0) {

**pthread\_mutex\_lock(&M3);**   
 pthread\_mutex\_lock(&M1);   
 insert(L1, myrank);   
 pthread\_mutex\_lock(&M2);   
 insert(L2, maxvalue(L1));   
 pthread\_mutex\_unlock(&M2);   
 pthread\_mutex\_unlock(&M1);

**pthread\_mutex\_unlock(&M3);**   
} else {

**pthread\_mutex\_lock(&M3);**  
 pthread\_mutex\_lock(&M2);   
 insert(L2, myrank+1);   
 pthread\_mutex\_lock(&M1);   
 insert(L1, minvalue(L2));   
 pthread\_mutex\_unlock(&M1);   
 pthread\_mutex\_unlock(&M2);

**pthread\_mutex\_unlock(&M3);**  
}

**Question 3:**

**The answer to this question can be found in C code**