# 6550 Parallel Programming HW2

Riley Densley

September 2019

## 1 Time Bomb

The time bomb is created by process 0 every time regardless of how many processes are running. It chooses a random number for the time and passes it to a random process using the function whoToThrowItTo. This is done so the same function can be used for future passes and for output to the console. Receiving and sending the time bomb is then processed in a while loop. All processes wait to receive. Once a process receives the bomb it sends it after it decrements the time. Once the bomb explodes a kill command is sent to all the processes to get them out of the loop and closes the program.

## 2 Code

```
#include <iostream>
#include <mpi.h>
#include <random>
#include <unistd.h>
#define MCW MPI_COMM_WORLD

using namespace std;

int whoToThrowItTo(){
    int rank, size;
    MPI_Comm_rank(MCW, &rank);
    MPI_Comm_size(MCW, &size);
    while(true){
        int reciever = rand() % size;
        if(reciever != rank){
            return reciever;
        }
    }
}
```

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

    int rank, size;
    int data;
    MPI_Init(&argc, &argv);
    MPI_Comm_rank(MCW, &rank);
    MPI_Comm_size(MCW, &size);

    //Process 0 starts the game
    int timer;
    int thrownTo;
    srand (time(NULL));
    if(rank == 0){
        timer = 10 + rand() % 20;
        thrownTo = whoToThrowItTo();
        cout<<rank<<" started the game with the bomb at "<<timer<<" and threw it to "<<throw
        MPI_Send(&timer,1,MPI_INT,thrownTo,0,MCW);

    }

    bool gameOver = false;

    while(!gameOver){

    MPI_Recv(&timer,1,MPI_INT,MPI_ANY_SOURCE,0,MCW,MPI_STATUS_IGNORE);
    sleep(.50);
    if( timer == 0 ){
        cout<<rank<<" EXPLODED!!!\n   Sending end of game command\n";
        timer = -1;
        for(int i = 0; i < size; i++){
            if(i!=rank){
                MPI_Send(&timer,1,MPI_INT, i, 0, MCW);
            }
        }
        gameOver = true;
        break;
    }
    else if (timer < 0){
        gameOver = true;
        cout<<rank<<" was told the game is over\n";
        break;
    }

    thrownTo = whoToThrowItTo();
    cout<<rank<<" recieved the bomb with a timer of "<<timer<<". Sent the bomb to "<<thrownT
    timer--;
    MPI_Send(&timer,1,MPI_INT, thrownTo,0,MCW);

    }

    MPI_Finalize();

    return 0;
}
```

## 3 Output

### 3.1 Commands

```
    mpic++ assign2.cpp
    mpirun -np 8 ./a.out
```

### 3.2 Output

```
0 started the game with the bomb at 19 and threw it to 7
7 received the bomb with a timer of 19. Sent the bomb to 5
7 received the bomb with a timer of 17. Sent the bomb to 6
5 received the bomb with a timer of 18. Sent the bomb to 7
6 received the bomb with a timer of 16. Sent the bomb to 5
5 received the bomb with a timer of 15. Sent the bomb to 6
6 received the bomb with a timer of 14. Sent the bomb to 7
1 received the bomb with a timer of 12. Sent the bomb to 5
1 received the bomb with a timer of 10. Sent the bomb to 7
5 received the bomb with a timer of 11. Sent the bomb to 1
7 received the bomb with a timer of 13. Sent the bomb to 1
7 received the bomb with a timer of 9. Sent the bomb to 0
0 received the bomb with a timer of 8. Sent the bomb to 6
1 received the bomb with a timer of 6. Sent the bomb to 6
6 received the bomb with a timer of 7. Sent the bomb to 1
6 received the bomb with a timer of 5. Sent the bomb to 0
0 received the bomb with a timer of 4. Sent the bomb to 1
1 received the bomb with a timer of 3. Sent the bomb to 0
0 received the bomb with a timer of 2. Sent the bomb to 2
1 was told the game is over
2 received the bomb with a timer of 1. Sent the bomb to 5
5 EXPLODED!!!
  Sending end of game command
6 was told the game is over
7 was told the game is over
```

```
0 was told the game is over
3 was told the game is over
2 was told the game is over
4 was told the game is over
```