ASSIGNMENT(GOL PSEUDO CODE)

ESE 3025: EmbeddedReal Time Operating Systems

Lambton College in Toronto

Instructor: Takis Zourntos

STUDENT NAME & ID:

VISHAL HASRAJANI(C0761544)

Parth Patel(C0764929)

Goutham Reddy Alugubelly(C0747981)

Ratnajahnavi rebbapragada(C0762196)

INTRODUCTION:

Conway's game of life has a 2D grid of square cells which can be either live or dead.Every cell interacts with its 8 neighbours.

DESCRIPTION:

MAIN THREAD EXPLANATION:

First we have to make a 2D array of rows and columns in order to store the data.

```
cell_t env[config_NE][config_ME];
```

update_env is used to store the data after applying the rules

```
cell_t update_env[config_NE][config_ME];
```

`initEnvironment()` function is used to get the data from seed_input.txt file and then store it in an 2D array.

Then we created threads corresponding to communities.

Further, if the reproduction_flag==true,we can allow new generations to check in and when reproduction_flag==false,we can update the display.

CELL COMMUNITY UPDATE THREADS PSEUDO CODE:

Rules of Game of life :

- If a cell is dead but surrounded by exactly three live neighbours, it sprouts to life (birth)
- If a cell is live but has more than 3 live neighbours, it dies (overpopulation)
- If a cell is live but has fewer than 2 live neighbours, it dies (underpopulation)
- All other dead or live cells remain the same to the next generation (i.e., a live cell must have exactly three neighbours to survive)

Main Rules Pseudo code :

```
void updateCell(size_t r, size_t c)
{
```

```
if(state_cell==0 && live_neighbours==3)
{
    update_cell[r][c]= live;

 //cell is dead and having 3 live neighbours,becomes a live
cell in next generation

}
else if(state_cell==1 &&( live_neighbours<2 ||
live_neighbours>3))
{
     update_cell[r][c] = dead;

//cell is live but has more than 3 live neighbours or less
than 2 live neighbours,then it dies in the next generation.
}
else
{
     update_cell[r][c] =state_cell;

//All others remain the same.

}
```

```
}
```

So here we have implemented the rules in the update cell just using simple if else condition.

void* updateCommFunc(void *param) :

This function updates all the cells for a thread (corresponding to one community)

```
void* updateCommFunc(void *param)
{
// If the reproduction flag is true means we can allow new
generations to check in ..

     if(reproduction_flag==true)

     {
          // *testing is a pointer pointing to the same
location as param
          threadID_t  *testing = param;

          //getting the block pair corresponding to a
thread.
          size_t i_t = testing->row;
          size_t j_t = testing->col;

//multiplying it with config_NC and config_MC to get exact
position of a row and column in a particular community
          size_t i_0 = i_t * config_NC;
          size_t j_0 = j_t * config_MC;

//Using FOR loop for updating all the cells corresponding
to a particular community
               for (size_t i = 0; i != config_NC; ++i)
               {
                    for (size_t j = 0; j != config_MC; ++j)
                    {
```

```
                    updateCell(i+i_0,j+j_0);
                    }
               }
        }



}
```

CONCLUSION :

Finally, we tried to implement the pseudo code which will be helpful in our main code implementation.