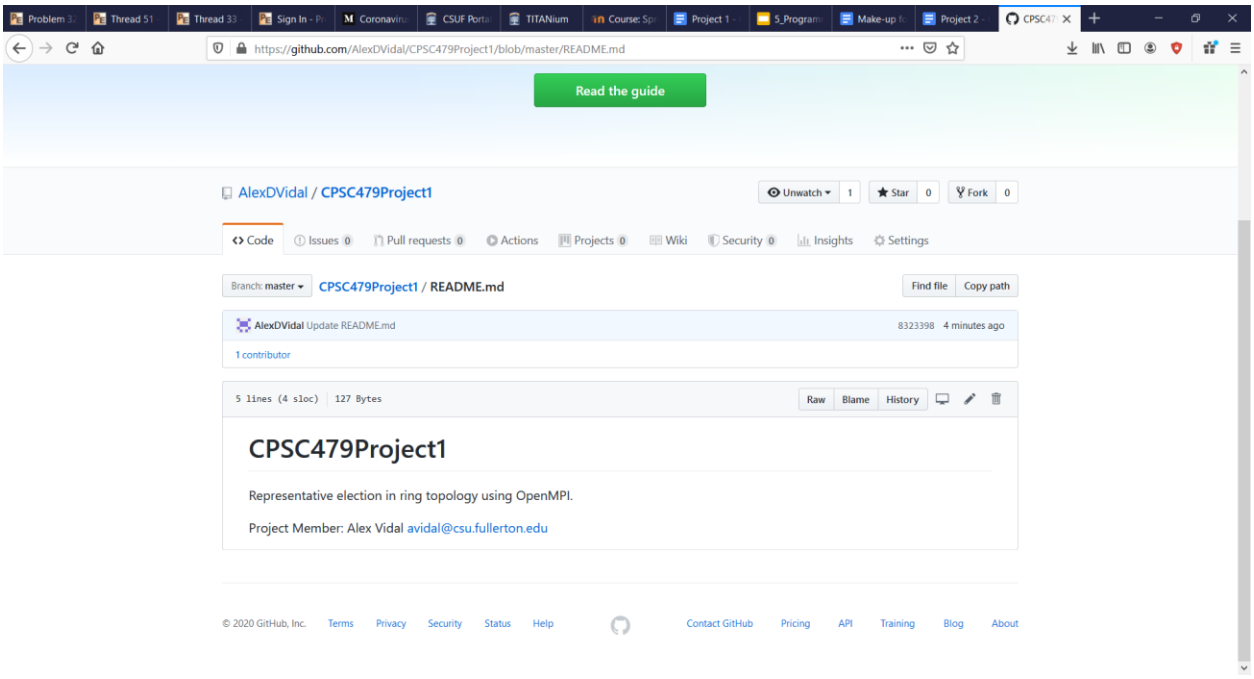
Project 1 Representative Selection in a Ring Topology

Alex Vidal: avidal@csu.fullerton.edu

Screenshot of repository and members



Pseudocode of the algorithm

```
//Find representative by means of passing generated numbers amongst
//neighbors in ring topology. If a process detects it has the smallest value of its
//group it will be considered the representative of the group.
int main(void){
	int rank, size, lowerRank, upperRank, randN, genN, lowerGen, upperGen
	string buffer

	//Start communications and find ranks
	MPI_Init()
	rank = MPI_Comm_Rank()
	size = MPI_Comm_Size()
	upperRank = (rank+1)%size
	if(rank == 0) {
		lowerRank = size-1 }
	else {
		lowerRank = rank – 1 }

	//Generate unique number
	seed(time())
	randN = rand() % 100
	if(randN < 10){ randN += 10 }

	//Generated number is a 1 followed by the random number
	//as 2 digits, and the rank as 2 digits. Rank ensure uniqueness
	sprintf(buffer, “1%02d%02d”, randN, rank)
	sscanf(buffer, “%d”, &genN)

	//Communicate generated numbers with neighbors. All communications are
	//non-blocking to simplify code. Synchronize with a wait
	MPI_Irecv(lowerGen, lowerRank)
	MPI_Irecv(upperGen, upperRank)
	MPI_Isend(genN, lowerRank)
	MPI_Isend(genN, upperRank)
	MPI_Wait()

	//print details and if representative
	printf(“Rank %d, rand %d, gen %d, lower %d, upper %d. ”,
		rank, randN, genN, lowerGen, upperGen)
	if(genN < lowerGen && genN < upperGen){
		printf(“I am representative.”)
	}
	printf(“\n”);

	//close comms
	MPI_Finalize()
}
```

Usage

To build the project type the following command, it will create an executable named “project1”:

```
~$ mpicc project.c -o project1
```

Once the project builds you can run it with the following command:

```
~$ mpirun -n <X> ./project1
```

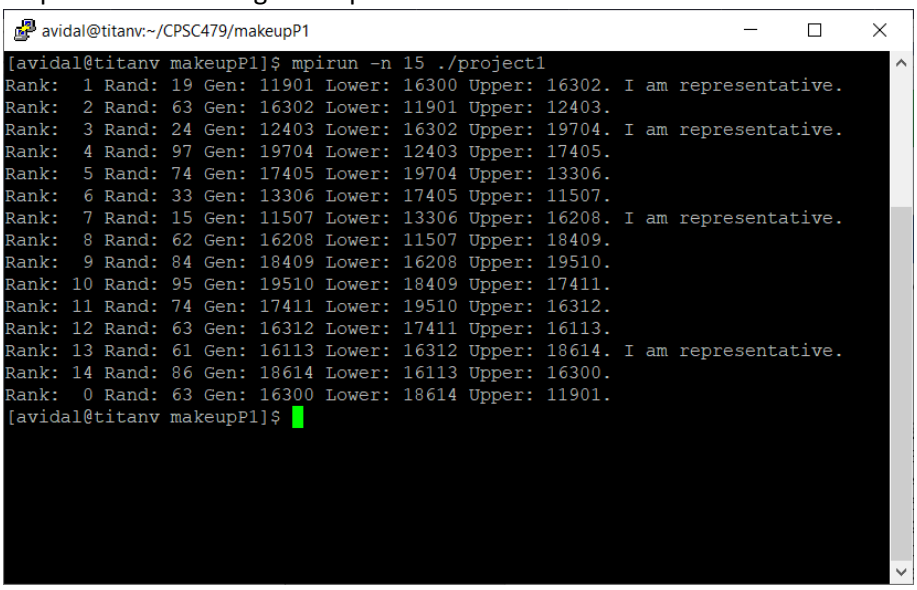
Where *<X>* can be any integer less than 100, but for the purposes of this project is between 6 and 20. Here’s some sample output.

```
[avidal@titanv makeupP1]$ mpirun -n 7 ./project1
Rank:  2 Rand: 62 Gen: 16202 Lower: 19501 Upper: 13003.
Rank:  3 Rand: 30 Gen: 13003 Lower: 16202 Upper: 17004. I am representative.
Rank:  4 Rand: 70 Gen: 17004 Lower: 13003 Upper: 11605.
Rank:  5 Rand: 16 Gen: 11605 Lower: 17004 Upper: 11506.
Rank:  0 Rand: 12 Gen: 11200 Lower: 11506 Upper: 19501. I am representative.
Rank:  1 Rand: 95 Gen: 19501 Lower: 11200 Upper: 16202.
Rank:  6 Rand: 15 Gen: 11506 Lower: 11605 Upper: 11200.
[avidal@titanv makeupP1]$
```

Each row of output is a single process printing its rank and the generated numbers of itself and its neighbors. If it has the lowest number of the group it will additionally print that it is a representative.

Output screenshots

Here are screenshots of two different runs:
15 processes resulting in 4 representatives



20 processes resulting in 7 representatives