Naman Gangwani
nkg160030@utdallas.edu

CS 4348.002
March 31, 2018

Project 2 Design

Semaphores

Semaphore *receptionistAvailable*: initial value of 1; blocks the patient when he/she enters the waiting room if the receptionist is already busy with another patient or allows the patient to see the receptionist if he/she is not with another patient. The patient that was recently with the receptionist can then signal this semaphore to unblock the next patient in line to see the receptionist when he/she is done.

Semaphore *receptionistTakesPatient*: initial value of 0; blocks the receptionist when he/she is waiting on a patient to serve. A ready patient may signal this semaphore to unblock the receptionist which triggers the receptionist to begin serving him/her.

Semaphore *goToWaitingRoom*: initial value of 0; blocks the patient when he/she is talking to the receptionist. When the receptionist has finished serving the patient, he/she may signal this semaphore to unblock the patient and allow him/her to go to the waiting room.

Semaphore *waitingInWaitingRoom*: initial value of 0; blocks the patient when he/she waiting in the waiting room and awaiting an available nurse to take him/her to the doctor's office. An available nurse can signal this semaphore to unblock his/her next patient to take him/her to the doctor's office.

Semaphore *nurses[]*: initial value of 0 for all the semaphores in the array; blocks a nurse in the array when that nurse does not have any patient to work with. A ready patient may signal an available nurse in this array of semaphores to notify that he/she is ready to be taken to the doctor's office and unblock the same nurse once again after he/she has finished his/her doctor's appointment. An array allows for a "mailbox" of semaphores between an arbitrary number of patients and nurses.

Semaphore *doctors[]*: initial value of 0 for all the semaphores in the array; blocks a doctor in the array when that doctor does not have any patient to work with. A ready patient may signal his/her assigned doctor (corresponding with the nurse) when he/she has entered his/her doctor's office. An array allows for a "mailbox" of semaphores between an arbitrary number of patients and doctors.

Semaphore *patientDoctorInteraction[]*: initial value of 0 for all the semaphores in the array; blocks a patient waiting on a semaphore in the array when his/her corresponding doctor is listening to his/her symptoms. The doctor can signal his/her blocked patient when he/she has finished listening, triggering the patient to listen to the doctor's advice. Once the patient has listened to the doctor's advice, he/she is free to leave.

Naman Gangwani
nkg160030@utdallas.edu

CS 4348.002
March 31, 2018

Pseudocode

semaphore receptionistAvailable = 1;

semaphore receptionistTakesPatient = 0, goToWaitingRoom = 0, waitingInWaitingRoom = 0;

semaphore[] nurses = {0}, doctors = {0}, patientDoctorInteraction = {0}

void patient()

{

/ patient is assigned a unique patientId */*

semWait(receptionistAvailable);

print("Patient [patientId] enters the waiting room, waits for receptionist.");

semSignal(receptionistTakesPatient);

semWait(goToWaitingRoom);

print("Patient [patientId] leaves receptionist and sits in waiting room");

semSignal(receptionistAvailable);

semWait(waitingInWaitingRoom);

/ randomly select a doctorId of a nurse/doctor that is available */*

semSignal(nurses[doctorId])

semWait(patientDoctorInteraction[patientId]);

print("Patient [patientId] enters doctor [doctorId]'s office");

semSignal(doctors[doctorId]);

semWait(patientDoctorInteraction[patientId]);

print("Patient [patientId] receives advice from doctor [doctorId]");

semSignal(nurses[doctorId]);

print("Patient [patientId] leaves");

}

Naman Gangwani
nkg160030@utdallas.edu


```
void receptionist()
{
	while (true)
	{
		semWait(receptionistTakesPatient);
		print("Receptionist registers patient [currentPatientId]");
		semSignal(goToWaitingRoom);
	}
}
```

```
void nurse()
{
	/* nurse is assigned a unique nurseId (that corresponds with a doctor's doctorId) */
	while (true)
	{
		semSignal(waitingInWaitingRoom);
		semWait(nurses[nurseId]);
		print("Nurse [nurseId] takes patient [currentPatientId] to doctor's office");
		semSignal(patientDoctorInteraction[currentPatientId]);
		semWait(nurses[nurseId]);
	}
}
```

Naman Gangwani
nkg160030@utdallas.edu


```
void doctor()
{
        /* doctor is assigned a unique doctorId */
        while (true)
        {
                semWait(doctors[doctorId]);
                print("Doctor [doctorId] listens to symptoms from patient [currentPatientId]");
                semSignal(patientDoctorInteraction[currentPatientId]);
        }
}

void main()
{
        /* assuming N patients and M nurses/doctors */
        parbegin (      patient, . . . N times, . . . patient,
                        receptionist,
                        nurse, . . . M times, . . . nurse,
                        doctor, . . . M times, . . . doctor          );
}
```