Κωδικός Εργασίας (2.3)

**Ομάδα** [13]

**Μέλος A**: [Γεώργιος, Σιδέρης, 1622, sideris@uth.gr]

**Μέλος B**: [Φώτιος, Τσώκος, 1679, tsokos@inf.uth.gr]

1. Περιγραφή σημείων συγχρονισμού με ψευτοκώδικα

**1.1) Η τεχνική επίλυσης (Train.c)**

Έχουμε το τρένο με μια χωρητικότητα Capacity, και Ν επιβάτες. Θέλουμε το τρένο να γεμίζει με επιβάτες να αδειάζει, να επιστρέφει, να παίρνει τους επόμενους κ.ο.κ. μέχρι το τελευταίο δρομολόγιο όπου μπορεί να ξεκινήσει χωρίς να είναι γεμάτο αν δεν υπάρχουν αρκετοί εναπομείναντες επιβάτες για να το γεμίσουν.

Η λογική που χρησιμοποιήσαμε είναι απλή:

Αφού μάθουμε τον αριθμό των συνολικών επιβατών, δημιουργούμε ένα νήμα για τον κάθε επιβάτη, και ένα νήμα για το τρένο. Οι επιβάτες περιμένουν πριν επιβιβαστούν να βεβαιωθούν ότι υπάρχει ακόμα χώρος στο τρένο (1ο σημείο συγχρονισμού),δίνοντας ο ένας επιβάτης την σειρά στον επόμενο. Αν δεν χωράνε άλλοι επιβάτες, το τρένο ξεκινά (2ο σημείο συγχρονισμού), αλλιώς περιμένουμε να γεμίσει. Αφού γεμίσει και κάνει την βόλτα, οι επιβάτες αποβιβάζονται ένας ένας (κάθε επιβάτης δίνει σειρά στον επόμενο να κατέβει) και αν κατέβουν όλοι το τρένο επιστρέφει πίσω (3ο σημείο συγχρονισμού). Το πρόγραμμα μέσω της main περιμένει σήμα από το νήμα του τρένου ότι τελειώσαν οι διαδρομές και οι επιβάτες για να τερματίσει την λειτουργία του(4ο σημείο συγχρονισμού).

**1.2) Ψευδοκώδικας**

Ας υποθέσουμε ότι τα mutex train, end είναι αρχικοποιημένα ως locked και ότι το enter είναι αρχικοποιημένο ως unlocked.

**main** {

onTrain= trainCapacity;

create threads passengers;

create thread train;

lock (&end); **//**wait for train to finish

}

**train** {

while (1){

lock (&train);

printf (MSG);

if (NoMoreTrips) {

unlock (&end);

**//**if there's nothing else to do end program

}else{

onTrain=trainCapacity;

unlock (&enter);}

//else continue by accepting the next passengers

}

}

**passenger** {

lock (&enter); **//1o σημειο συγχρονισμου**

onTrain--;

if (onTrain=0) {

unlock (&train);

}else{

unlock (&enter);}

}

**1.3) Αμοιβαίος αποκλεισμός και λιμοκτονία**

Tο πρόγραμμα μας εγγυάται αμοιβαίο αποκλεισμό (κάθε επιβάτης περιμένει πρώτα να δει αν χωράει πριν μπεi στο τρένο) και μέσω των mutexes δεν υπάρχει περίπτωση να μπούνε δύο μαζί.

Επίσης, δεν υπάρχει περίπτωση ένα νήμα να μείνει κολλημένο για πάντα γιατί η λύση μας προϋποθέτει κάθε νήμα να “ξυπνήσει” από το προηγούμενο του.

**1.4) Αδυναμίες**

Η λύση μας λειτουργεί για πεπερασμένο αριθμό επιβατών. Ωστόσο, για τους ίδιους ακριβώς λόγους με το 2.3, η λύση αυτή μπορεί να εφαρμοστεί και σε αρκετά μεγαλύτερη είσοδο επιβατών(Και εδώ με ένα δεύτερο loop που επαναλαμβάνεται όσο υπάρχουν ενδιαφερόμενοι).

**1.5) Η λύση του passenger.c**

Σε αυτή την εκδοχή του προγράμματος, το τρένο δεν έχει ένα ξεχωριστό νήμα, αλλα η λειτουργία του υλοποιείται μέσα σε κάθε νήμα επιβάτη.

Ολα τα σημεία συγχρονισμού λειτουργούν όπως και στην παραπάνω εκδοχή.

1. Σχολιασμός φαινομένων / αποτελεσμάτων

Το πρόγραμμα δουλεύει σωστά. Όλοι οι επιβάτες μπαίνουν σε τρένο, όλα τα τρένα είναι γεμάτα εκτός από το τελευταίο το οποίο σωστά περιμένει όσους επιβάτες έχουν απομείνει (Passengers%trips).