Πανεπιστήμιο Πατρών
Τμήμα Μηχανικών Η/Υ & Πληροφορικής
Λειτουργικά Συστήματα

Αναφορά Δεύτερης Άσκησης – Ακαδημαϊκό έτος 2012-2013

Αφροδίτη Αλεβιζοπούλου ΑΜ: 3879
Χρήστος Κλαουδιάνης ΑΜ: 3677

Για την ανάπτυξη του κώδικα της 2ης άσκησης βασιστήκαμε στην υλοποίση της 1ης Άσκησης (χρήση sockets και πρωτόκολλο ανταλλαγής πληροφοριών Client-Server). Πιο συγκεκριμένα:

1. Πελάτης:

Στο τμήμα κώδικα του client χρειάστηκε μόνο να αλλάξουμε τους χρόνους (αλλαγή τιμών και μετατροπή από msec σε sec).

2. Εξυπηρετητής:

Το κομμάτι της διαχείρισης των sockets έχει παραμείνει ως επί το πλείστον το ίδιο. Ως την αποδοχή της σύνδεσης δεν έχουμε αλλάξει κάτι.

Για τη διαχείριση των συνδέσεων με νήματα χρειάστηκε να σχεδιαστεί ο κώδικας και οι δομές μας κατάλληλα. Για κάθε σύνδεση δημιουργείται ένα διαφορετικό νήμα που αναλαμβάνει τη διαχείριση της παραγγελίας. Η παραγγελία είναι χωρισμένη στα εξής βήματα (συνάρτηση *process_order)*:

- Προεπεξεργασία παραγγελίας. Μετατροπή των raw bytes σε κατάλληλη δομή (*struct _order*)
- Ψήσιμο παραγγελίας. Για κάθε πίτσα της παραγγελίας ελέγχεται αν υπάρχει διαθέσιμος ψήστης. Αν δεν υπάρχει, το πρόγραμμα αναμένει έως ότου να βρεθεί ο πρώτος διαθέσιμος και μετά δημιουργεί ένα νήμα (συνάρτηση *bake_thread*) για το ψήσιμο. Αυτό κρίθηκε απαραίτητο έτσι ώστε παράλληλα με το ψήσιμο της κάθε πίτσας της παραγγελίας να μπορούμε να αναζητούμε και ψήστες για τις υπόλοιπες.
- Αναζήτηση διαθέσιμου διανομέα. Το νήμα αναμένει έως ότου να υπάρχει διαθέσιμος διανομέας και στη συνέχεια (συνάρτηση *deliver_order*) υπολογίζει το συνολικό χρόνο της παραγγελίας (ψήσιμο + αναμονή για διαθέσιμο διανομέα) και στέλνει τη πίτσα. Αν η παραγγελία έχει καθυστερήσει παραπάνω από T_VERYLONG χρόνο, τότε εμφανίζεται αντίστοιχο μήνυμα και προσθέτουμε στην παραγγελία μια μπύρα δωρεάν. Όπως αναφέραμε, στο συνολικό χρόνο της παραγγελίας δεν συμπεριλαμβάνουμε και το χρόνο παράδοσης της πίτσας, καθώς αυτός είναι σταθερός (3 ή 5). Το μόνο που μας ενδιαφέρει είναι ο χρόνος που χρειάστηκε για να ψηθούν όλες οι πίτσες της παραγγελίας, συμπεριλαμβανομένων των καθυστερήσεων που μπορεί να προέκυψαν, συν το χρόνο που αναμέναμε έως ότου να βρεθεί διαθέσιμος διανομέας. Βάσει αυτού του χρόνου θα προσθέσουμε τη δωρεάν μπύρα στην παραγγελία.

- Αφού έχουν ολοκληρωθεί όλες οι διαδικασίες κλείνουμε το socket και ελευθερώνουμε τους πόρους μας.

Το τελικό βήμα ήταν η προσαρμογή του script *run_tests.sh* της προηγούμενης άσκησης, το οποίο ξεκινά 5 πελάτες μαζί οι οποίοι στέλνουν ο καθένας από 20 τυχαίες παραγγελίες.

Οι πόροι που διαμοιράζοταν ανάμεσα στα νήματα είναι οι μεταβλητές *available_psistes* και *available_deliver_boys.* Για την προστασία τους χρησιμοποιήθηκαν τα mutexes *bake_mutex* και *deliver_mutex* αντίστοιχα, τα οποία χρησιμοποιήθηκαν και για τα condition variables *bake_cond*, *deliver_cond* .

Με το χρόνο T_VERYLONG στην τιμή 20 και με 5 πελάτες (5 παράλληλες συνδέσεις) παρατηρήσαμε ότι όλες οι παραγγελίες πραγματοποιούνται μέσα στο διαθέσιμο χρόνο. Κατεβάζοντας το T_VERYLONG στα δέκα(10) δευτερόλεπτα και αυξάνοντας τις παράλληλες συνδέσεις (πελάτες) σε 20 και πάνω τότε μόνο παρατηρούμε καθυστερήσεις στις παραγγελίες.

Στη συνάρτηση *process_order* χρησιμοποιούμε ένα array *bake_threads* στο οποίο κρατάμε τα νήματα που φροντίζουν για το ψήσιμο της κάθε πίτσας (δλδ. που περιμένουν όσο χρόνο χρειάζεται το ψήσιμο). Αυτό το χρειαζόμαστε για να μπορούμε να κάνουμε pthread_join έτσι ώστε το νήμα της παραγγελίας να περιμένει ώσπου να ψηθούν όλες οι πίτσες.

Δομές:

(Οι δομές που χρησιμοποιήθηκαν είναι οι ίδιες με αυτές της προηγούμενης άσκησης).

Η βασική δομή που χρησιμοποιήσαμε είναι η δομή order (ορισμένη στο order.h).
Η δομή χρησιμοποιείται τόσο για την ανταλλαγή μηνυμάτων ανάμεσα στον εξυπηρετητή και τον πελάτη αλλά και εσωτερικά στον κώδικα του εξυπηρετητή ανάμεσα στις συναρτήσεις που διαχειρίζονται την παραγγελία. Η δομή αποτελείται από 4 πεδία. Τις πίτσες που θέλει ο πελάτης, την απόστασή του (πληροφορίες που στέλνει ο πελάτης) και κάποια πεδία που χρησιμοποιεί μόνο ο εξυπηρετητής: το χρόνο που λάβαμε την παραγγελία και το αν ο πελάτης δικαιούται δωρεάν μπύρα επειδή η παραγγελία ξεπέρασε το χρόνο T_VERYLONG.

Εξωτερικές δομές που χρησιμοποιήσαμε εκτενώς είναι η *struct sockaddr_un* και η *struct timeval*:

- *struct sockaddr_un* : Χρησιμοποιείται για να παραμετροποιηθεί το socket κατάλληλα. Τόσο από το server για το που θα ακούει (listen) και θα δέχεται αιτήσεις (accept) όσο και από τον πελάτη για να δηλώσει που θέλει να συνδεθεί (connect).

- *struct timeval*. Χρησιμοποιήθηκε εκτενώς για να παρακολοθούμε την εξέλιξη της παραγγελίας. Αν και είχε αναφερθεί ότι μπορούμε να χρησιμοποιούμε alarm για τον αν έχει αργήσει η παραγγελία θεωρήσαμε πιο αποτελεσματικό να ελέγχουμε στο τέλος της παραγγελίας αν έχει αργήσει και κατάλληλα να προσθέτουμε μια μπύρα.

Οδηγίες:

Η εντολή make δημιουργεί δύο εκτελέσιμα αρχεία, το pizzaria (=server) και το client. Αν τρέξουμε τον client χωρίς ορίσματα στέλνει μια default παραγγελία. Αν θελήσουμε να δώσουμε custom παραγγελία τρέχουμε τον client και του δίνουμε 4 ορίσματα με τη σειρά που αναφέρονται:

<αριθμός μαργαρίτα> <αριθμός πεπερόνι> <αριθμός σπέσιαλ> <απόσταση από πιτσαρία>

Ο πελάτης θα στείλει την παραγγελία μόνο αν ο συνολικός αριθμός των πιτσών είναι το πολύ τρεις.

Το script *run_tests.sh* τρέχει τον server στο background και εκτελεί 5 παράλληλους πελάτες που ο καθένας στέλνει 20 τυχαίες παραγγελίες. Επίσης, δίνοντας ένα όρισμα στο script μπορούμε να καθορίσουμε τον αριθμό των πελατών που θα τρέξουν παράλληλα (π.χ ./run_tests.sh 30)

Τα μηνύματα που βλέπουμε στο τερματικό του client τρέχοντας το script είναι τα μηνύματα του server προς τους clients σχετικά με την παράδοση της παραγγελίας τους και είναι τα εξής:

- *"PIZZA READY. Hope to see you again!"* , αν η παραγγελία τους ετοιμάστηκε εγκαίρως.
- *"PIZZA READY. Delivery was late. Sorry. Here get a free beer!"*, αν η παραγγελία τους ξεπέρασε το χρόνο T_VERYLONG.

Στο τερματικό του server τυπώνονται μηνύματα με τις πληροφορίες της παραγγελίας του πελάτη, δηλαδή τα 4 ορίσματα που έδωσε ο πελάτης ως είσοδο και το συνολικό χρόνο ετοιμασίας της παραγγελίας.

Επίσης, οι πληροφορίες σχετικά με την παραγγελία που έστειλε ο κάθε πελάτης και το συνολικό χρόνο ετοιμασίας της φαίνονται και στο αρχείο pizzaria.log που δημιουργείται τρέχοντας το script.