REPORT: OS pthread project

Οι εργασία που ολοκληρώσαμε πραγματεύεται τον πολυνηματικό προγραμματισμό και η συγκεκριμένη προσομοιώνει την εξυπηρέτηση παράλληλων πελατών προσομοιώνοντας την λειτουργία μίας υποτιθέμενης πιτσαρίας.

Η ροή της εκτέλεσης ξεκινάει από την main() η οποία δημιουργεί 100 παράλληλους πελάτες, το οποίο σημαίνει ότι δεν επιρεάζει ο ένας άμεσα τις ενέργειες του άλλου βάζοντας τους σε μία συνάρτηση όπου περιμένουν για διάστημα δοσμένο στην εκφώνηση. Αφού η παραγγελεία του πελάτη νήματος παραδοθεί ή ματαιωθεί λόγω αδυναμίας πληρωμής, η main κάνει Join αυτό το thread με βάση το επιστρεφόμενο id από την συνάρτηση παραγγελειοληψίας και τυπώνει μετά την εξυπηρέτηση όλων των νημάτων μερικά στατιστικά με βάση τις ενέργειες που πραγματοποιήθηκαν συνολικά.Η αρχικοποίση έγινε επίσης σε αυτό το τμήμα κώδικα(mutexes και condition variables)

Το ενδιαφέρον όμως τμήμα του κώδικα είναι η ίδια η συνάρτηση order_func(void* iod).

--Παραγγελειοληψία

Ας δούμε αναλυτικά τον τρόπο λειτουργίας της. Αρχικά η είσοδος ενός πελάτης κρατίεται σε ένα ρολόι με χρόνο 0 την αρχή της ημέρας και συχνότητα πελατών σύμφωνη με την εκφώνηση αυτός ο χρόνος φυλλάσετε μέσα σε mutex.Ένα συχνό pattern είναι η "ατομικότητα" των ενεργειών σε κάθε παραγγελεία οτι οι ενέργειες γίνονται μέσα σε κλειδώματα.Με το ίδιο κλείδωμα ελέγχουμε και την πληρωμή.Το αποτέλεσμα της πληρωμής , ή της απόρριψης της, επιρεάζει τον σύνολο revenue που κλειδώνεται με το revenue_mutex.

Κάθε νήμα ως πελάτης έχει ένα κόστος παραγγελείας αρχικοποιημένο με 0 έως ότου να καθοριστεί ψευδοτυχαία η επιλογές που θα έκανε και μετά με πιθανότητα 90% θα μπορέσει να πληρώσει άρα και να προστεθέι το προσωπικό του ποσό στα έσοδα το οποίο προστατεύεται γύρω από κλειδώματα revenue mutex για να διασφαλιστεί η σωστή τιμή των εσόδων, επίσης κρατάμε και την συγκεκριμένη παραγγελία ανά πελάτη μέσα στο ίδιο lock διότι πολλοί παράγοντες προετιμασίας την χειρίζοται(διασφάλιση ακεραιότητας της τιμής).

--Παρασκευή παραγγελίας

Αρχικά πρέπει ο cook να κάνει την προετιμασία τον δεσμεύω για διάστημα sleep(Tprep*number_of_pizzas), προσπαθώ τουλάχιστον(while(Ncooks=0)), η περιμένω να μου σηματοδοτηθεί η συνθήκη &no_cooks όταν ένας απο αυτούς απελευθερωθεί , δηλαδη Ncook++ , στην οποία στιγμή έχει δεσμεύσει κατάλληλο πλήθος φούρνων με όμοιο τρόπο το σήμα διαθεσιμότητας φούρνων δίνεται από την δέσμευση πακεταριστων που τις αφαιρουν από εκεί(συμμετρικά η εναπόθεσει σε φούρνους σημαίνει ελεύθερος cook). Κρατάμε από τον φούρνο timestamp για τον

χρόνο κρυώματος και στο πακετάρισμα για χρόνο ετοιμασίας από την άφιξη. int xronos_apo_efmanish_pelath_mexri_telos_paketarismatos = (telosPaketarismatos.tv_sec-arxhT.tv_sec);

--Παράδοση

Όμοιως τρόπος δέσμευσης με μάγειρες με την διαφορά ότι δεν αλληλεπιδρούν άμμεσα με κάποιον άλλο πόρο και η αποδέσμευση τους παρεμβάλεται απο χρόνο παράδοσης , αλλαγή μεταβλητών χρόνου εντός κλειδώματος(total_xronos_krywmatos για το M.O,megistos_xronos_krywmatos μέγιστο) και την επιστροφή στο κατάστημα όπου στέλνεται σήμα στα άλλη threads που τον περίμεναν.

Note: Αποδεσμεύω όσους φουρνους δέσμευσα (Noven+= num_of_pizzas;) σηματοδοτόντας την ύπαρξη τους pthread_cond_signal(&ovens);

dev/testing notes: το .sh file τρέχει με εντολή ./deploy.sh