Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης
Τμήμα Πληροφορικής
2012-2013

Λειτουργικά Συστήματα
Εργασία 2η | Προσομοίωση Δρομολογητή

Όνομα	ΑΕΜ	email
Νεστορίδης Αντώνης	2078	antonest@csd.auth.gr
Οικονομίδης Γιώργος	2080	georgido@csd.auth.gr
Πίσκας Γιώργος	2087	gpiskasv@csd.auth.gr
Σιώζος-Δρόσος Σωκράτης	2134	siozosdr@csd.auth.gr

Σκοπός της εργασίας αυτής είναι η προσομοίωση της λειτουργίας της δρομολόγησης CPU με τους αλγορίθμους SJF και SRTF. Θα παρουσιαστεί η βασική φιλοσοφία της και τα βήματα της προσομοίωσης αυτής:

1) Δημιουργία διεργασιών

Οι διεργασίες δημιουργούνται και διαχειρίζονται από την κλάση ProcessGenerator και βρίσκονται στο αρχείο processes.txt. Μπορούμε είτε να δημιουργήσουμε νέες διεργασίες είτε να διαβάσουμε το υπάρχον αρχείο. Σε περίπτωση που επιλέξουμε δημιουργία νέων διεργασιών, κατασκευάζουμε από 5 έως 100 ψευδοτυχαίες διεργασίες και στην συνέχεια για κάθε μία από αυτές προσδίδουμε και πάλι ψευδοτυχαία τα εξής χαρακτηριστικά:

- PID (αυξάνεται κατα ένα)
- Arrival time (για τις χρονικές στιγμές από 0-14)
- CPU burst time (1-50 μονάδες χρόνου)

Το processes.txt έχει την εξής μορφή. Η πρώτη σειρά περιέχει τους τίτλους της κάθε στήλης. Κάθε επόμενη σειρά αντιπροσωπεύει μία διεργασία και έχει 3 ακέραιους αριθμούς (διαχωρισμένους με ένα κενό) οι οποίοι αντιπροσωπεύουν αντίστοιχα το PID, το arrival time και το cpu burst time της εκάστοτε διεργασίας. Ύστερα γίνεται ανάγνωση του αρχείου και οι νέες διεργασίες τοποθετούνται στη λίστα NewProcessTemporaryList.

2) Εκτέλεση προγράμματος

Για να εκτελεσθεί το πρόγραμμα χρησιμοποιείται το εκτελέσιμο scheduler.jar με δύο ορίσματα μέσω cmd. Κατ' αρχάς κάνουμε "cd path/to/jar" και στη συνέχεια εκτελούμε "java -jar scheduler.jar arg1 arg2".

- Το arg1 μπορεί να είναι "-read" ή "-gen" για διάβασμα ή παραγωγή διεργασιών αντίστοιχα.
- Το arg2 μπορεί να είναι "-sjf" ή "-srtf" για τον αντίστοιχο αλγόριθμο που θα επιλέξουμε.

Σε περίπτωση που δέν υπάρχουν αρκετά ορίσματα ή είναι λανθασμένα, το πρόγραμμα τερματίζεται και ο χρήστης ενημερώνεται σχετικά με τη σωστή χρήση των ορισμάτων.

3) Βασική ροή εκτέλεσης

Αφού γεμίσει η λίστα NewProcessTemporaryList, γεμίζουμε την ReadyProcessesList με τις διεργασίες σύμφωνα με το χρόνο άφιξής τους. Στη συνέχεια γίνεται δρομολόγηση και εκτέλεση. Ενδιάμεσα, ελέγχεται αν υπήρχε και τελείωσε η διεργασία που βρισκόταν στη CPU και εφόσον έχει τελειώσει προστίθεται στην λίστα TerminatedProcessesList. Η εκτέλεση συνεχίζεται μέχρις ότου η TerminatedProcessesList να περιέχει όλες τις διεργασίες του αρχείου εισόδου. Τέλος, γίνεται καταγραφή των στατιστικών. Όλες οι κλάσεις είναι επαρκώς σχολιασμένες για εύκολη κατανόηση της ροής.

Επίσης, το συμπιεσμένο αρχείο περιέχει και το παράδειγμα των διαφανειών το οποίο βρίσκεται στο week4.pdf στη σελίδα 9 και 10.

4) Σχολιασμός αποτελεσμάτων

Έχουμε τρέξει μερικές προσομοιώσεις, οι οποίες βρίσκονται στον φάκελο simulations. Η πρώτη είναι για το παράδειγμα που αναφέραμε παραπάνω, ενώ οι υπόλοιπες τρέχουν για λίστες διεργασιών που παράχθηκαν με ψευδοτυχαίο τρόπο.

Παρατηρήσεις:

- Στις μέσες περιπτώσεις όπου δεν υπάρχουν ακραίες τιμές (μεγάλα burst για ίδιο arrival time κ.τ.λ.) ο μέσος χρόνος αναμονής και για τους 2 αλγορίθμους είναι παρόμοιος αν και ο SRTF τείνει να είναι κατά λίγο μικρότερος.
- Σε πιο ακραίες τιμές ο SRTF ευνοεί τις διεργασίες με μικρό burst time κατα πολύ, κι έτσι αν η πλειοψηφία των διεργασιών είναι τέτοιου είδους, το average wait time είναι πολύ μικρότερο από αυτόν του SJF. Οι υπόλοιποι χρόνοι δεν παρουσιάζουν σημαντική διαφορά.
- Το average wait time για μεγάλο αριθμό διεργασιών τείνει να ειναι κοντά στο 1/3 των συνολικών ticks, που έγιναν για την ολοκλήρωση των διεργασιών, ενώ για μικρό αριθμό διεργασιών είναι αρκετά μικρότερο.
- Ο SRTF δίνει σχεδόν πάντα καλύτερα αποτελέσματα, ως προς οποιοδήποτε χρόνο, σε σχέση με τον SJF.