

ΗΥ3604 Ενσωματωμένα Συστήματα  
Πραγματικού Χρόνου  
Αριστοτέλειο Πανεπιστήμιο Θεσσαλονίκης  
Εργασία 1<sup>η</sup>

Δημήτριος Αντωνιάδης  
8462  
akdimitri@auth.gr

Μάιος 2019

## Περιεχόμενα

1	Εισαγωγή.	3
2	Φάκελος υποβολής, Compilation και Εκτέλεση.	3
3	Αλγόριθμοι.	4
3.1	Αλγόριθμος simple . . . . .	4
3.2	Αλγόριθμος advanced. . . . .	5
4	Αποτελέσματα.	6
4.1	Αποτελέσματα αλγορίθμου simple. . . . .	6
4.2	Αποτελέσματα αλγορίθμου advanced. . . . .	8
5	Υπολογιστική ισχύς.	9
6	Συμπεράσματα.	11

## 1 Εισαγωγή.

Το παρόν έγγραφο αποτελεί την αναφορά της πρώτης (1<sup>η</sup>) εργασίας που πραγματοποιήθηκε στο πλαίσιο του μαθήματος *Ένσωματωμένα Συστήματα Πραγματικού Χρόνου*. Σκοπός της εργασίας είναι να γίνει τακτική δειγματοληψία με την μικρότερη δυνατή απόκλιση από τον πραγματικό χρόνο. Στο πείραμα αυτό οι τιμές της δειγματοληψίας είναι τα **timestamps** που επιστρέφει η συνάρτηση **gettimeofday()** [1].

Παραπάνω παρουσιάστηκε μία σύντομη εισαγωγή αναφορικά με το θέμα της παρούσας αναφοράς. Στη δεύτερη (2<sup>η</sup>) ενότητα παρουσιάζονται τα αρχεία που περιλαμβάνει η εργασία αυτή και η μέθοδος του *Compilation*. Στην τρίτη (3<sup>η</sup>) ενότητα παρουσιάζονται οι δύο αλγόριθμοι που υλοποιήθηκαν και στην τέταρτη (4<sup>η</sup>) ενότητα τα αποτελέσματα τους. Στην πέμπτη (5<sup>η</sup>) ενότητα παρουσιάζονται τα αποτελέσματα από τις μετρήσεις που έγιναν επάνω στην υπολογιστική ισχύ που καταναλώνει η διεργασία εκτέλεσης. Τέλος, στην έκτη (6<sup>η</sup>) παρουσιάζονται τα συμπεράσματα της εργασίας αυτής.

## 2 Φάκελος υποβολής, Compilation και Εκτέλεση.

Ο φάκελος υποβολής με τον πηγαίο κώδικα βρίσκεται στην παρακάτω διεύθυνση:

- <https://github.com/akdimitri/RTES1>

Στον φάκελο *code* περιλαμβάνονται δύο (2) αρχεία. Τα αρχεία:

- simple.c
- advanced.c

Το πρώτο αρχείο αποτελεί το ζητούμενο πηγαίο κώδικα της εργασίας ο οποίος δεν κάνει χρήση των προηγούμενων *timestamps*. Το δεύτερο αρχείο αποτελεί το ζητούμενο πηγαίο κώδικα της εργασίας ο οποίος κάνει χρήση των προηγούμενων *timestamps* με σκοπό την ακριβή δειγματοληψία πραγματικού χρόνου.

Το *Compilation* των δύο παραπάνω αρχείων πραγματοποιείται με τις εξής εντολές:

- gcc simple.c -o simple -O3
- gcc advanced.c -o advanced -O3

Η εκτέλεση και των δύο εκτελέσιμων αρχείων πραγματοποιείται με τον ίδιο τρόπο. Η εκτέλεση απαιτεί δύο ορίσματα. Το πρώτο όρισμα αντιπροσωπεύει το χρόνο εκτέλεσης του πειράματος σε **ώρες** και το δεύτερο όρισμα το χρονικό διάστημα που μεσολαβεί μεταξύ δύο δειγματοληψιών σε **δευτερόλεπτα**. Για παράδειγμα, με τις παρακάτω εντολές εκτελούνται τα πειράματα για 2 ώρες με χρονικό διάστημα μεταξύ δύο timestamps ίσο με 0.1 δευτερόλεπτα:

- `./simple 2 0.1`
- `./advanced 2 0.1`

Επιπλέον, στο repository περιλαμβάνεται ο φάκελος *Rstudio* ο οποίος περιέχει το αρχείο **script.r** με το οποίο γίνεται η ανάλυση των αποτελεσμάτων και εξάγονται τα διαγράμματα. Ακόμη, περιλαμβάνεται και το αρχείο **regex.R**, το οποίο χρησιμοποιείται για την μελέτη της υπολογιστικής ισχύς του προγράμματος.

Τέλος, περιλαμβάνεται στο ρεποζιτορφ και ένας φάκελος με ενδεικτικές μετρήσεις.

## 3 Αλγόριθμοι.

Στην παρούσα ενότητα παρουσιάζονται οι δύο αλγόριθμοι που πραγματοποιήθηκαν σε μορφή ψευδογλώσσας ώστε να είναι εύκολα κατανοητοί.

### 3.1 Αλγόριθμος simple

Στην υποενότητα αυτή παρουσιάζεται ο αλγόριθμος που υλοποιείται από τον πηγαίο κώδικα *simple.c*. Το πρόγραμμα δέχεται ως όρισμα το συνολικό χρόνο εκτέλεσης του προγράμματος (*executionTime*) και το διάστημα (*interval*) που μεσολαβεί μεταξύ δύο δειγματοληψιών χρόνου. Συνεπώς, είναι δυνατό διαιρώντας το συνολικό χρόνο εκτέλεσης με το χρονικό διάστημα μεσολάβησης να βρεθεί ο συνολικός αριθμός χρονικών διαστημάτων μεσολάβησης μεταξύ των δειγματοληψιών. Για να πραγματοποιηθούν τόσα διαστήματα μεσολάβησης απαιτείται να ληφθούν ισόποσα στιγμιότυπα χρόνου (*timestamps*) συν ένα (1) επιπλέον. Η μεταβλητή που αναπαριστά τον αριθμό των συνολικών στιγμιότυπων που θα ληφθούν ονομάζεται *iterations*.

Η δειγματοληψία των χρονικών στιγμών πραγματοποιείται με τη χρήση της συνάρτησης

- `gettimeofday(struct timeval *tv, struct timezone *tz) [1]`

η οποία επιστρέφει σε κατάλληλη δομή (struct) το χρόνο που έχει παρέλθει από τη χρονική στιγμή *Epoch*, δηλαδή τη χρονική στιγμή 1970-01-01 00:00:00 +0000 (UTC).

Το διάστημα που μεσολαβεί μεταξύ της δειγματοληψίας δύο χρονικών στιγμών πραγματοποιείται με τη χρήση της συνάρτησης

- *usleep( useconds\_ t usec)* [1]

Η συνάρτηση αυτή αναστέλλει την εκτέλεση του thread που εκτελείται για χρονικό διάστημα *usec*.

---

**Algorithm 1:** simple.c

---

**input** : *executionTime(HOURS), interval(SECONDS)*  
**output:** *timestamps MATRIX[iterations, 1]*  
1 *iterations*  $\leftarrow$  (*executionTime*) \* 3600/*interval* + 1  
2 *timestamps*[1]  $\leftarrow$  *gettimeofday(...)*  
3 **for** *i*  $\leftarrow$  2 : *iterations* **do**  
4     *usleep(interval \* 1000000)*  
5     *timestamps[i]*  $\leftarrow$  *gettimeofday(...)*  
6 **end**  
7 export *timestamps MATRIX* to a text file

---

### 3.2 Αλγόριθμος advanced.

Ο αλγόριθμος αυτός δειγματοληπτεί τόσες χρονικές στιγμές όσες και ο παραπάνω. Η μόνη διαφορά είναι ότι ο παραπάνω αλγόριθμος αποτυγχάνει τα δειγματοληπτήσει με την ακρίβεια που δειγματοληπτεί ο advanced. Ο αλγόριθμος που παρατίθεται παρακάτω κάνει χρήση των προηγούμενων χρονικών στιγμών που έχουν ληφθεί κατά τη δειγματοληψία ώστε να βελτιώσει την απόδοση του. Συγκεκριμένα, εφόσον είναι γνωστή η πρώτη τιμή δειγματοληψίας και το χρονικό διάστημα που μεσολαβεί μεταξύ δύο δειγματοληψιών είναι δυνατό να γνωρίζουμε και την ακριβή στιγμή που πρέπει να πραγματοποιηθεί η δειγματοληψία της τιμής *i*. Δηλαδή,

$$timestamps[i] \leftarrow timestamps[1] + i * interval$$

Επομένως, για κάθε τιμή δειγματοληψίας είναι γνωστή και η αναμενόμενη τιμή της. Έτσι, για κάθε τιμή δειγματοληψίας μπορεί να υπολογιστεί και η χρονική απόκλιση και να αφαιρεθεί από το επόμενο χρονικό διάστημα μεσολάβησης.

---

**Algorithm 2:** advanced.c

---

```
input : executionTime(HOURS), interval(SECONDS)
output: timestamps MATRIX[iterations, 1]
1 iterations  $\leftarrow$  (executionTime) * 3600/interval + 1
2 delay  $\leftarrow$  0

3 timestamps[1]  $\leftarrow$  gettimeofday(...)
4 for i  $\leftarrow$  2 : iterations do
5   | usleep(interval * 1000000 - delay)
6   | timestamps[i]  $\leftarrow$  gettimeofday(...)
   | delay  $\leftarrow$  timestamps[i] - i * interval - timestamps[0]
7 end
8 export timestamps MATRIX to a text file
```

---

## 4 Αποτελέσματα.

Στην ενότητα αυτή παρουσιάζονται τα αποτελέσματα από την εκτέλεση των παραπάνω αλγορίθμων. Τα παρακάτω διαγράμματα προκύπτουν ύστερα από την επεξεργασία των δεδομένων μέσω του προγράμματος **Rstudio**. Το *script.r* περιλαμβάνει όλες τις εντολές για την εξαγωγή των παρακάτω αποτελεσμάτων και διαγραμμάτων.

### 4.1 Αποτελέσματα αλγορίθμου simple.

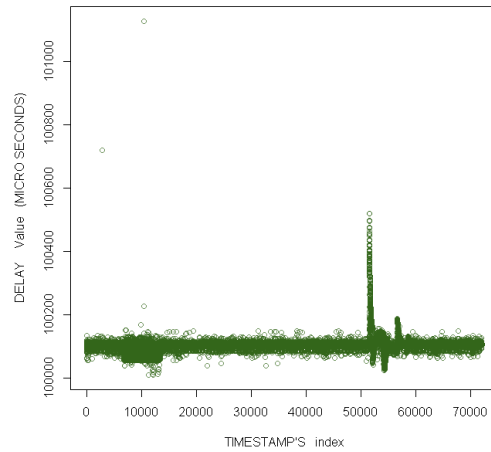
Η εκτέλεση του αλγορίθμου αυτού με ορίσματα:

- *executionTime*  $\leftarrow$  2
- *interval*  $\leftarrow$  0.1

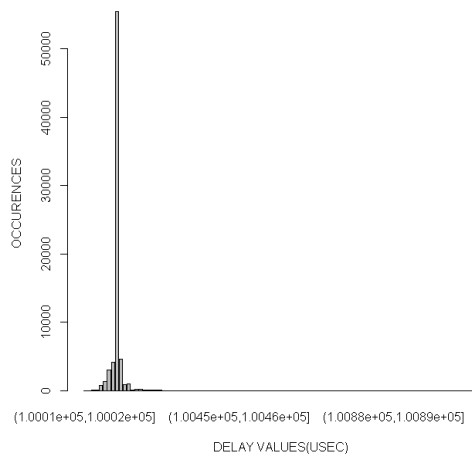
είχε συνολικό χρόνο: 7207.682 *secs* δηλαδή καθυστέρησε για περίπου **7.5** δευτερόλεπτα. Η μέση τιμή ήταν **100106.7 usec**, δηλαδή μεταξύ κάθε κάθε μέτρησης υπήρχε καθυστέρηση περίπου 107 μικρο-δευτερολέπτων. Η τυπική απόκλιση ήταν **18.024**.

Όπως φαίνεται και από τα παρακάτω διαγράμματα στο σύνολό τους οι τιμές ήταν μεγαλύτερες των 100000 usec που ήταν η επιθυμητή τιμή, για το λόγο αυτό παρατηρήθηκε και αυτή η απόκλιση.

Σχήμα 1: Τιμές διαστημάτων μεσολάβησης μεταξύ 2 δειγματοληψιών του αλγορίθμου simple



Σχήμα 2: Κατανομή τιμών των διαστημάτων μεσολάβησης μεταξύ 2 δειγματοληψιών του αλγορίθμου simple



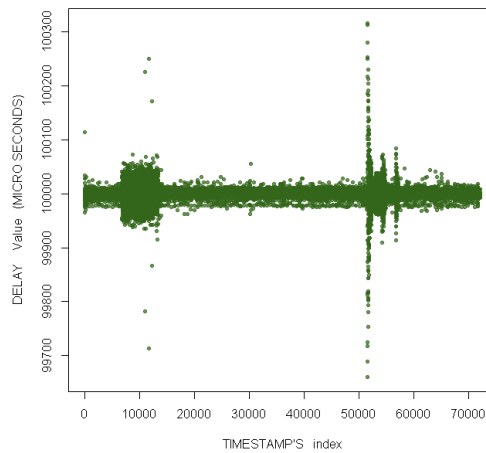
## 4.2 Αποτελέσματα αλγορίθμου advanced.

Η εκτέλεση του αλγορίθμου αυτού με ορίσματα:

- $executionTime \leftarrow 2$
- $interval \leftarrow 0.1$

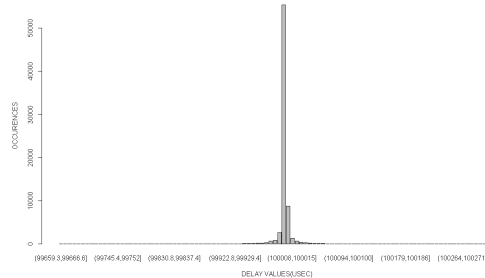
είχε συνολικό χρόνο: 7200.000108 *secs* δηλαδή καθυστέρησε για περίπου **0.000108** δευτερόλεπτα ή 108 μικρο-δευτερόλεπτα. Η μέση τιμή ήταν **100000.0015 u-sec**, δηλαδή μεταξύ κάθε κάθε μέτρησης υπήρχε καθυστέρηση περίπου 0.0015 μικρο-δευτερολέπτων. Η τυπική απόκλιση ήταν **9.14**.

Σχήμα 3: Τιμές διαστημάτων μεσολάβησης μεταξύ 2 δειγματοληψιών του αλγορίθμου advanced





Σχήμα 4: Κατανομή τιμών των διαστημάτων μεσολάβησης μεταξύ 2 δειγματοληψιών του αλγορίθμου *advanced*



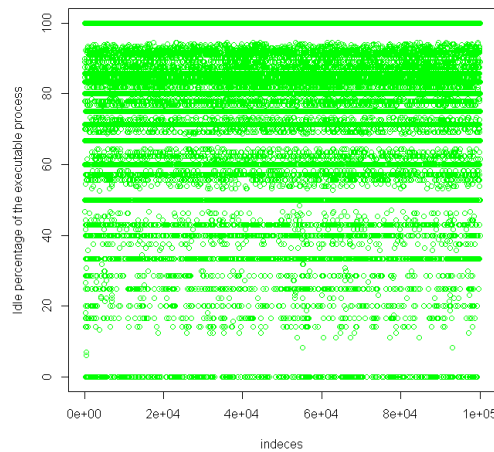
Στα αποτελέσματα που προέκυψαν η διεργασία αναγνωρίστηκε μόλις **706** φορές σε κατάσταση **Running** και **99294** φορές σε λειτουργία **Sleeping**.

Παρατίθενται τα διαγράμματα ποσοστών κατάστασης idle της διεργασίας ανά μέτρηση, όπως αυτά προέκυψαν από τις παραπάνω μετρήσεις και την ανάλυση.

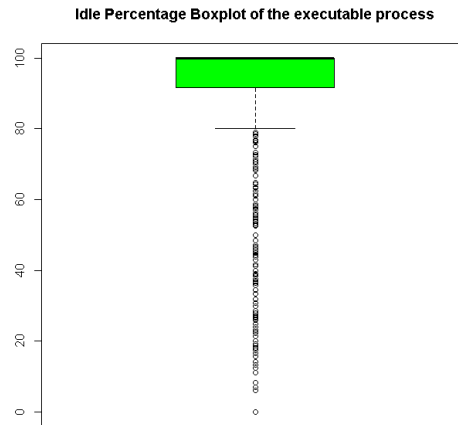
Σημειώνεται για τη μεταβλητή που εξετάζεται ότι αντιπροσωπεύει το ποσοστό χρόνου κατά το οποίο εκτελείται η μέτρηση στο οποίο η διεργασία παραμένει στον kernel idle handler. [1], ή διαφορετικά:

- id, idle : time spent in the kernel idle handler [1]

Σχήμα 5: Τιμές μεταβλητής idle(%)



Σχήμα 6: Boxplot τιμών μεταβλητής idle(%)



## 6 Συμπεράσματα.

Τελικά, η εργασία αυτή καταλήγει στο συμπέρασμα ότι η δειγματοληψία τιμών την οποία εκτελεί ένα ενσωματωμένο σύστημα είναι καλό να ελέγχεται και να γίνεται προσπάθεια να είναι όσο το δυνατό ακριβέστερη. Στην παρούσα υλοποίηση παρατηρήθηκε σφάλμα της τάξης του 0.0000015% στα διαστήματα δειγματοληψίας.

## Αναφορές

- [1] Michael Kerrisk. Linux Programmer's Manual. [http://man7.org/linux/man-pages/dir\\_all\\_alphabetic.html](http://man7.org/linux/man-pages/dir_all_alphabetic.html).