Σειρά Προβλημάτων 4

Ημερομηνία Παράδοσης: 17/11/17

Σε αυτή την άσκηση καλείστε να μοντελοποιήσετε και να αναλύσετε ένα πρωτόκολλο εκλογής αρχηγού. Θεωρήστε ένα δίκτυο το οποίο αποτελείται από ένα σύνολο διασυνδεδεμένων κόμβων. Κάθε ένας από τους κόμβους αυτούς χαρακτηρίζεται από μια διεύθυνση. Στόχος του πρωτοκόλλου είναι η αναγνώριση του κόμβου με την πιο μικρή διεύθυνση στο δίκτυο και η εκλογή αυτού του κόμβου ως αρχηγού.

Ως μέρους του πρωτοκόλλου, κάθε κόμβος *i* του δικτύου διατηρεί δύο πληροφορίες:

1. l_i: ο κόμβος που ο *i* θεωρεί ως αρχηγό, και
2. h_i: η απόσταση του κόμβου *i* μέχρι τον αρχηγό, όπου ως απόσταση εννοούμε των αριθμό των ακμών (hops) της συντομότερης διαδρομής από τον *i* μέχρι τον l_i.

Σε περίπτωση που κάποιος κόμβος *j* θεωρεί τον αριθμό του ως αρχηγό, τότε $l_j = j$ και $h_j = 0$.

Οι κόμβοι του δικτύου ανταλλάσουν μεταξύ τους μηνύματα της μορφής

$$m = (\textit{πηγή}, \textit{προορισμός}, \textit{αρχηγός}, \textit{απόσταση})$$

όπου

- το πεδίο «*πηγή*» καθορίζει τη διεύθυνση του αποστολέα του μηνύματος,
- το πεδίο «*προορισμός*» καθορίζει τη διεύθυνση του παραλήπτη του μηνύματος,
- το πεδίο «*αρχηγός*» καθορίζει τον κόμβο που η πηγή θεωρεί ως αρχηγό και
- το πεδίο «*απόσταση*» καθορίζει τον αριθμό των συνδέσμων από την πηγή μέχρι τον αρχηγό.

Κάθε φορά που ένας κόμβος *i* παραλαμβάνει ένα μήνυμα *m*, τότε ο *i* συγκρίνει τα πεδία *m.αρχηγός* και *m.απόσταση* με τις τοπικές πληροφορίες που κατέχει (l_i και h_i). Σε περίπτωση που ($m.\textit{αρχηγός} > l_i$) ή ($m.\textit{αρχηγός} = l_i$ και $m.\textit{απόσταση} > h_i$) ο κόμβος *i* αγνοεί/απορρίπτει το μήνυμα. Διαφορετικά, οι τοπικές πληροφορίες αναθεωρούνται και θέτουμε ($l_i = m.\textit{αρχηγός}$ και $h_i = m.\textit{απόσταση} + 1$).

Κάθε φορά που οι τοπικές πληροφορίες ενός κόμβου αναθεωρούνται, τότε αυτός στέλνει ένα μήνυμα σε κάθε ένα από τους γειτονικούς του κόμβους με εξαίρεση τον κόμβο ο οποίος του έστειλε το μήνυμα στον οποίο οφείλεται η αλλαγή. Υπάρχει ένα άνω όριο στο χρόνο μετάδοσης ενός μηνύματος το οποίο ονομάζεται MSG_DELAY. Σημειώστε ότι δεν υπάρχει κατώτατο όριο στο χρόνο παράδοσης ενός μηνύματος και γι' αυτό το λόγο μπορεί να υπάρξει αναδιάταξη της σειράς των μηνυμάτων.

Χρονικές Παράμετροι

- Η παράδοση οποιουδήποτε μηνύματος απαιτεί κάποιο χρόνο *t* που κυμαίνεται ως $0 \leq t \leq$ MSG_DELAY.
- Το πρωτόκολλο διαθέτει ένα μηχανισμό χρονικού ορίου εντολής: Αν ένας κόμβος δεν παραλάβει κάποιο «χρήσιμο» μήνυμα (εξαιρούνται αυτά που έχουν απορριφθεί) για *περισσότερο* από κάποια χρονική περίοδο *timeout*, τότε θα εμφανίσει λήξη του χρονικού του ορίου (time out) το πολλή σε *TIMEOUT_DELAY* χρόνο. Δηλαδή, υπάρχει ένα διάστημα [*timeout, timeout+TIMEOUT_DELAY*] μετά από την παραλαβή ενός «χρήσιμου» μηνύματος μέσα στο οποίο το όριο αναμονής θα λήξει (και ο κόμβος θα εμφανίσει timeout).

Με την έναρξη της εκτέλεσης του πρωτοκόλλου, η τιμή του *timeout* αρχικοποιείται μέσω μιας σταθεράς, έστω *INIT_TIMEOUT.* Με την παραλαβή κάθε «χρήσιμου» μηνύματος από κάποιο κόμβο *i* η τιμή του *timeout* μεταβάλλεται ως ακολούθως:

$$timeout = INIT\_TIMEOUT + TIMEOUT\_DELAY + h_i \times MSG\_DELAY$$

Το πρωτόκολλο που θα αναλύσουμε ορίζεται ως εξής:

1. *Κάθε κόμβος i αρχικοποιεί* $l_i = i$ *και* $h_i = 0$.
2. *Μετά την αρχικοποίηση ο κάθε κόμβος στέλνει ένα μήνυμα σε κάθε ένα από τους γειτονικούς του κόμβους με τις τιμές l και h τις οποίες κατέχει.*
3. *Κάθε φορά που ένας κόμβος i παραλαμβάνει ένα μήνυμα m, τότε ο i συγκρίνει τα πεδία m.αρχηγός και m.απόσταση με τις τοπικές πληροφορίες που κατέχει (*l_i *και* h_i*). Σε περίπτωση που (m.αρχηγός >* l_i*) ή (m.αρχηγός =* l_i *και m.απόσταση >* h_i *) ο κόμβος i αγνοεί το μήνυμα. Διαφορετικά, οι τοπικές πληροφορίες αναθεωρούνται και θέτουμε (*l_i *= m.αρχηγός και* h_i *= m.απόσταση + 1).*
4. *Κάθε φορά που οι τοπικές πληροφορίες ενός κόμβου αναθεωρούνται, τότε αυτός στέλνει ένα μήνυμα σε κάθε ένα από τους γειτονικούς του κόμβους με εξαίρεση τον κόμβο ο οποίος του έστειλε το μήνυμα στον οποίο οφείλεται η αλλαγή.*
5. *Όταν ένας κόμβος i εμφανίσει λήξη του χρονικού του ορίου (timeout) και ικανοποιεί* l_i *= i τότε εκλέγει τον εαυτό του ως αρχηγό και στέλνει ένα μήνυμα ενημέρωσης προς όλους τους γείτονές του, οι οποίοι θα προωθήσουν το μήνυμα προς τους δικούς τους γείτονες.*
6. *Όταν όλοι οι κόμβοι ενημερωθούν, το πρωτόκολλο τερματίζει.*

(α) Μοντελοποιήστε το πιο πάνω πρωτόκολλο στο εργαλείο UPPAAL σε δίκτυο που έχουμε 3 κόμβους σε μια πλήρως συνδεδεμένη τοπολογία χρησιμοποιώντας τις πιο κάτω σταθερές:

i.	INIT_TIMEOUT	10
ii.	TIMEOUT_DELAY	5
iii.	MSG_DELAY	3

(β) Ελέγξτε κατά πόσο το μοντέλο ικανοποιεί τις πιο κάτω προτάσεις:

i. Το μοντέλο δεν περιέχει ανεπιθύμητα αδιέξοδα (ως αδιέξοδο δεν εννοούμε τον φυσιολογικό τερματισμό του πρωτοκόλλου).

ii. Είναι δυνατόν το σύστημα να φτάσει σε μια κατάσταση όπου όλοι οι κόμβοι γνωρίζουν τον σωστό αρχηγό.

iii. Σε κάθε εκτέλεση ο κόμβος *i* θα γνωρίζει τον σωστό αρχηγό μετά από την πάροδο $timeout + TIMEOUT\_DELAY + d_i * MSG\_DELAY$ χρόνου, όπου d_i είναι η απόσταση ανάμεσα στον κόμβο *i* και τον αρχηγό.

(γ) Σχολιάστε την ποιότητα του μοντέλου σας όσον αφορά

i. Την ορθότητα του μοντέλου σας
ii. Την απλότητα του μοντέλου σας, δεδομένου ότι μεταξύ διάφορων μοντέλων με τις ίδιες δυνατότητες το πιο απλό είναι το πιο επιθυμητό
iii. Επεκτασιμότητα και επαναχρησιμοποίηση

(δ) BONUS: Δοκιμάστε ορισμένες διαφορετικές τοπολογίες. Ποιο είναι το μέγιστο μέγεθος δικτύου που μπορεί να αναλύσει το UPPAAL;