ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΚΥΠΡΟΥ

<u>Παραδοτέο 2</u>: Τυπικές μέθοδοι περιγραφής και ανάλυσης πιθανοτικών συστημάτων

Ερευνητικό Πρόγραμμα ΕΝΔΙΚΤΗΣ

1. Εισαγωγή

Τις δύο τελευταίες δεκαετίες η περιοχή των τυπικών μεθόδων για το σχεδιασμό και την ανάλυση συστημάτων έχει αναπτυχθεί δραματικά και έχει σημειώσει μια τεράστια επιτυχία τόσο στην ανάπτυξη θεωρητικών μεθόδων περιγραφής [1, 2, 3] και ανάλυσης συστημάτων [1, 2, 3] όσο και στην παροχή πρακτικών εργαλείων για τους σκοπούς αυτούς [4, 5]. Τελευταία, το ενδιαφέρον των ερευνητών άρχισε να στρέφεται στον τομέα της δημιουργίας τυπικών μεθόδων για περιγραφή και ανάλυση *πιθανοτικών* συστημάτων που επιβάλλεται από την ανάγκη για δικαιολόγηση της πιθανοτικής συμπεριφοράς, όπως αυτή παρουσιάζεται σε τυχαιοποιημένα, κατανεμημένα και ανεκτικά σε λάθη συστήματα [6-10].

Τα πλεονεκτήματα των τυπικών μεθόδων τα οποία υποβοήθησαν στην επικράτησή τους για την ανάλυση συστημάτων περιλαμβάνουν τη δυνατότητα μοντελοποίησης συστημάτων και την ύπαρξη κοινής μεθοδολογίας για μελέτη της ορθότητας και της αξιοπιστίας που τα χαρακτηρίζει. Οι τυπικές μέθοδοι έχουν ιδιαίτερη χρησιμότητα σε συστήματα όπου πολλαπλές διεργασίες ενεργούν ταυτόχρονα και κατά συνέπεια δυνατόν να παρουσιάσουν μη-ντετερμινισμό και περισσότερες από μία διαφορετικές συμπεριφορές [1,11]. Οι τυπικές μέθοδοι εκτελούν ανάλυση σε ολόκληρο το σύστημα καταστάσεων που αντιστοιχεί σε ένα τέτοιο μοντέλο, με άλλα λόγια, αναλύουν όλες τις δυνατές εκτελέσεις του μοντέλου. Αυτό έρχεται σε αντίθεση με τις μεθόδους ανάλυσης συστημάτων που βασίζονται σε προσομοίωση (simulation) όπου ακριβή αποτελέσματα εξάγονται για μία ακριβώς εκτέλεση ενός συστήματος και όχι για το σύνολό τους. Στόχος αυτής της μελέτης είναι η επισκόπηση μοντέλων περιγραφής και ανάλυσης συστημάτων που έχουν προταθεί για την περιγραφή και ανάλυση πιθανοτικών συστημάτων και η εισήγηση μίας μεθοδολογίας για την περιγραφή δικτυακών εφαρμογών. Η έκθεση οργανώνεται ως εξής: η Παράγραφος 2 παρουσιάζει μοντέλα πιθανοτικών συστημάτων που έχουν προταθεί στη βιβλιογραφία. Στην Παράγραφο 3 περιγράφεται η άλγεβρα διεργασιών QoSPA. Στην Παράγραφο 4 παρουσιάζονται οι αλγόριθμοι που αναπτύξαμε για τον υπολογισμό συμπεριφορών long-run average και, τέλος, στην Παράγραφο 5 δίνεται ένα παράδειγμα μοντελοποίησης μέσω της QoSPA για ένα διακομιστή της τεχνολογίας DIFFSERV.

2. Μοντέλα Πιθανοτικών Συστημάτων

Η γενική μέθοδος που ακολουθήθηκε για τη δημιουργία τυπικών μεθόδων περιγραφής και ανάλυσης συστημάτων είναι η επέκταση υπαρχόντων μοντέλων και τεχνικών οι οποίες είχαν αποδειχθεί επιτυχημένες για μη-πιθανοτικά πρότυπα. Πιο κάτω παρουσιάζονται κάποια μοντέλα για περιγραφή συστημάτων, καθώς και τρόποι για ανάλυση τους βασισμένοι στα μοντέλα αυτά.

2.1. Markov Decision Processes

Το πρώτο μοντέλο που θα περιγράψουμε είναι αυτό των ***Markov Decision Processes*** (MDP) [12, 13]. Ένα MDP είναι μια γενίκευση μιας αλυσίδας Markov στην οποία για κάθε κατάσταση συσχετίζουμε και ένα σύνολο από πιθανές ενέργειες (actions). Η συμπεριφορά μιας Markov Decision Process αποτελείται από μια ατέλειωτη εναλλαγή από καταστάσεις και ενέργειες: Σε κάθε κατάσταση, επιλέγεται μη ντετερμινιστικά μια ενέργεια και ακολούθως χρησιμοποιείται η ανάλογη κατανομή των πιθανοτήτων για την επιλογή της επόμενης κατάστασης.

Πιο συγκεκριμένα, ένα MDP (*S, A, p*), αποτελείται από ένα πεπερασμένο σύνολο από καταστάσεις S, και δύο άλλα στοιχεία A, p τα οποία καθορίζουν και τη δομή των μεταβάσεων:

- Για κάθε $s \in S$, $A(s)$ είναι ένα μη άδειο πεπερασμένο σύνολο από ενέργειες διαθέσιμες στο S.
- Για κάθε $s,t \in S$ και $A(s)$, $p_{st(a)}$ είναι η πιθανότητα μετάβασης από την κατάσταση s στην κατάσταση t χρησιμοποιώντας την ενέργεια α. Για κάθε $s,t \in S$ και $A(s)$, $0 \le p_{st(a)} \le 1$ και $\sum_{t \in S} p_{st}(a) = 1$.

Ο έλεγχος του μοντέλου αυτού γίνεται με τη χρήση της λογικής PCTL [14], η οποία βασίζεται στην λογική CTL [3]. Ο έλεγχος του μοντέλου των χαρακτηριστικών του "*until*", εκτελείται διαμέσου μιας αναγωγής σε ένα γραμμικό πρόβλημα βελτιστοποίησης. Για την επίλυση των γραμμικών προβλημάτων βελτιστοποίησης, μπορούν να χρησιμοποιηθούν γνωστοί αλγόριθμοι, όπως π.χ. η μέθοδος simplex [15].

Πέραν του ελέγχου του μοντέλου που προαναφέραμε, είναι δυνατός ο υπολογισμός χαρακτηριστικών τύπου *long run average* για την ανάλυση του συστήματος [16]. Το

long run average ενός συστήματος, είναι η μέση συμπεριφορά του συστήματος, όταν αυτό τρέχει άπειρα πολύ χρόνο και με αυτό μπορούμε να ερευνήσουμε χαρακτηριστικά του συστήματος τα οποία σχετίζονται σε πολλές περιπτώσεις με την ανάλυση της απόδοσης και της ασφάλειας (reliability), αφού συμπεριλαμβάνουν ποσότητες όπως μέσος όρος απόκρισης, μέσος όρος χρόνου ανάμεσα σε αποτυχίες (failures) και δικαιοσύνη του δικτύου.

Για να μπορεί να γίνει η χρήση των long run average χαρακτηριστικών, πρέπει να γίνει η επέκταση του MDP σε ***Timed probabilistic system*** (TPS). Ένα TPS (*S, A, p, S_{in}, time, I*) είναι ένα MDP (*S, A, p*) με τρία επιπλέον στοιχεία:

- Ένα υποσύνολο $S_{in} \subseteq S$ αρχικών καταστάσεων
- Ένα labeling time το οποίο αναθέτει για κάθε $s \in S$ και $\alpha \in A(s)$ την αναμενόμενη ποσότητα χρόνου time $(s,a) \in \Re^+$ που θα ξοδευτεί στην κατάσταση *s* όταν επιλεγεί η ενέργεια α.
- Ένα labeling I το οποίο συσχετίζει για κάθε $x \in V$ και $s \in S$ την τιμή *I[[x]]* του *x* στο *s*.

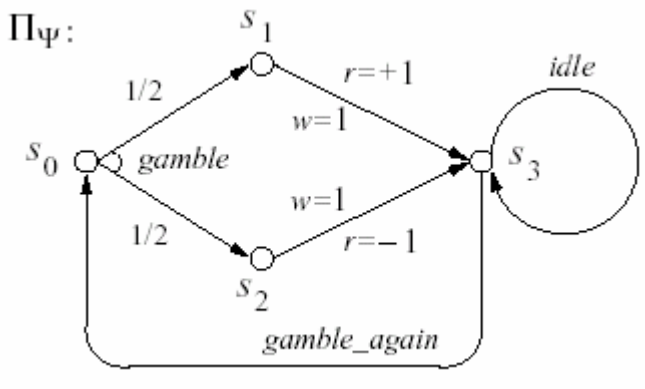
Η βασική ιδέα για την ανάλυση χαρακτηριστικών long run average στο μοντέλο αυτό είναι η εξής:

i. Η μέση συμπεριφορά που θέλουμε να αναλύσουμε μεταφράζεται ως ένα *πείραμα,* όπου το πείραμα ορίζεται ως μία πλειάδα $=(V, E, E_r, V_{in}, \lambda)$ και αντιστοιχεί σε ένα γράφο (V, E), με κόμβους το σύνολο V και ακμές το σύνολο $E \subseteq V \times V$ και τα ακόλουθα επιπλέον στοιχεία:
   - Ένα υποσύνολο $V_{in} \subseteq V$ αρχικών καταστάσεων
   - Ένα υποσύνολο $E_r \subseteq E - \{(v,v) \mid v \in V\}$ από ακμές τύπου reset που αντιστοιχούν στις ενέργειες
   - Ένα labeling λ, το οποίο αναθέτει σε κάθε $u \in V$ μια φόρμουλα λ(u) πάνω στα state variables V που συμβολίζει τερματισμό ενός στιγμιότυπου ενός πειράματος

ii. Το μοντέλο μας "συνδυάζεται" με το πείραμα του οποίου θα θέλαμε να μετρήσουμε τη long run average συμπεριφορά.

iii. Μέσω μιας αναγωγής σε ένα γραμμικό πρόβλημα βελτιστοποίησης υπολογίζονται οι ζητούμενες τιμές.

Ακολουθεί η περιγραφή ενός συστήματος – παιγνιδιού ως ένα TPS. Έστω ένα σύστημα παιγνιδιού, το οποίο για κάθε ρίξιμο (gamble) του παίχτη επιστρέφεται κέρδος ±1 με ίση πιθανότητα. Μετά από κάθε ρίξιμο ο παίχτης μπορεί να παραμείνει αδρανής ή να ξαναρίξει. Το σχήμα που ακολουθεί δείχνει το συνδυασμό του TPS το οποίο ανταποκρίνεται στο παιγνίδι και του πειράματος Ψ το οποίο ανταποκρίνεται στο μέσο όρο κέρδους κάθε ριξίματος.



2.2. Probabilistic Timed Automata

Ένα δεύτερο μοντέλο που χρησιμοποιείται για την περιγραφή πιθανοτικών συστημάτων είναι αυτό των ***Probabilistic Timed Automata*** (PTA) [17]. Τα πιθανοτικά αυτόματα πραγματικού χρόνου είναι μια επέκταση των αυτομάτωμ πραγματικού χρόνου με τη χρήση πιθανοτικών διακλαδώσεων στις ακμές. Ένα PTA εξοπλίζει ένα κλασσικό αυτόματο με διακριτές πιθανοτικές κατανομές, η υποστήριξη των οποίων είναι ακμές. Για να δημιουργηθεί ένα κατανοητό μοντέλο στο οποίο όλες οι ακμές της υποστήριξης είναι διαθέσιμες την ίδια χρονική στιγμή, χρησιμοποιούνται συνθήκες σε ρολόγια τα οποία επιτρέπουν μεταβάσεις με κατανομές παρά με μεμονωμένες ακμές. Η έννοια της αρχικοποίησης του ρολογιού συσχετίζεται με τις ακμές. Συμβατικά τα PTA μπορούν να θεωρηθούν ως MDPs με τη διευκόλυνση της αρχικοποίησης και δοκιμής των ρολογιών με το στυλ των κλασσικών αυτομάτων.

Η λογική που χρησιμοποιείται για τον έλεγχο του μοντέλου είναι η PTCTL, μια λογική που αποτελεί προέκταση της TCTL με τους πιθανοτικούς τελεστές της PTCL. Ο αλγόριθμος για τον έλεγχο του μοντέλου αναμιγνύει την κατασκευή ενός

πεπερασμένων καταστάσεων MDP, ένα σχήμα μετάφρασης από PTCTL σε PCTL και μια εφαρμογή του ελέγχου μοντέλου της PCTL.

2.3. Labelled Concurrent Markov Chains

Τέλος, ένα τρίτο μοντέλο (το οποίο θα χρησιμοποιηθεί και στην έρευνα αυτή) είναι αυτό των ***Labeled Concurrent Markov Chains*** (LCMCs) [18]. Ένα LCMC είναι μια γενίκευση μιας αλυσίδας του Markov (Markov Chain), όπου υπάρχει η δυνατότητα πιθανοτικών και μη ντετερμινιστικών (non deterministic) συμπεριφορών-μεταβάσεων. Επομένως ένα LCMC είναι ένας γράφος του οποίου οι κόμβοι είναι είτε πιθανοτικοί είτε μη ντετερμινιστικοί. Οι ακμές (μεταβάσεις) που ξεκινούν από ένα πιθανοτικό κόμβο είναι όλες πιθανοτικές (με άθροισμα των πιθανοτήτων ίσο με 1), και από ένα μη ντετερμινιστικό κόμβο μη ντετερμινιστικές (κάθε ακμή ανταποκρίνεται σε μία ετικέτα/ενέργεια). Το μοντέλο αυτό μπορεί να χρησιμοποιηθεί για μοντελοποίηση συστημάτων τα οποία περιλαμβάνουν μη ντετερμινισμό, μέσω του οποίου μπορούμε να περιγράψουμε διαφορετικές επιλογές συμπεριφορών που πιθανόν να παρουσιάσει κάποιο σύστημα αλλά και πιθανοτική συμπεριφορά με την οποία μπορούμε να μοντελοποιήσουμε αβεβαιότητα ή και τυχαιοποίηση.

Πιο συγκεκριμένα, ένα LCMC είναι μια πλειάδα $(S_n, S_p, Act, \rightarrow_n, \rightarrow_p, s_0, t)$ και αποτελείται από δύο πεπερασμένα σύνολα S_n και S_p τα οποία αντιστοιχούν στις μη ντετερμινιστικές καταστάσεις και στις πιθανοτικές καταστάσεις, μια αρχική κατάσταση την s_0, ένα σύνολο από ετικέτες Act, και δύο άλλα στοιχεία $\rightarrow_n$ και $\rightarrow_p$ τα οποία καθορίζουν τη δομή των μεταβάσεων:

- $\rightarrow_n$ είναι το σύνολο των μη ντετερμινιστικών μεταβάσεων και

  $\rightarrow_n \subset S_n \text{ x Act x } (S_n \cup S_p)$,

- $\rightarrow_p$ είναι το σύνολο των πιθανοτικών μεταβάσεων και $\rightarrow_p \subset S_p \text{ x } (0,1] \text{ x } S_n$

  το οποίο ικανοποιεί $\sum_{(s,\pi,t)\in\rightarrow p} \pi = 1$ για κάθε $s \in S_n$ και

t είναι μια συνάρτηση η οποία παίρνει ως είσοδο ένα label $\alpha \in$ Act και έχει ως έξοδο το χρόνο που χρειάζεται για να γίνει μετάβαση από τον κόμβο s στον κόμβο t με label α.

$$t(\alpha) = \begin{cases} 1, & \text{αν η ενέργεια απαιτεί χρόνο} \\ 0, & \text{διαφορετικά} \end{cases}$$

Ο έλεγχος του μοντέλου για LCMC μπορεί να επιτευχθεί σε PCTL ή Πιθανοτική HML (Hennessy-Milner Logic). Μια πρόταση προς αυτή την κατεύθυνση περιλαμβάνει τη λογική PHML η οποία αποτελεί επέκταση της HML με τη χρήση του τελεστή until. Η λογική αυτή επεκτείνει την HML με το να επιτρέπει την ποσοτική ανάλυση των πιθανοτικών χαρακτηριστικών ενός συστήματος, καθώς και την παραμετροποίηση του τελεστή until με κανονικές εκφράσεις πάνω σε ονόματα γεγονότων (και όχι απλώς σε ένα όνομα). Επιπλέον, προσφέρεται και μια χρονικά οριοθετημένη έκδοση του τελεστή until, ώστε να είναι δυνατή και η χρησιμοποίηση των χαρακτηριστικών πραγματικού χρόνου.

Για να ελέγξουμε αν ένα LCMC T ικανοποιεί κάποια φόρμουλα $f \in L^{pr}_{HMLu}$, ο αλγόριθμος σημειώνει κάθε κατάσταση s του T με ένα σύνολο F $\subseteq$ closure(f) (όπου closure(f) δηλώνει το σύνολο των φόρμουλων {f' $\cup$ $\neg$f' | f' είναι μια υποφόρμουλα του f}), έτσι ώστε f' $\in$ F, s $\models$ f'. Το T ικανοποιεί το f, εάν και μόνο εάν το s_0, η αρχική κατάσταση του T, σημειωθεί με το f. Ο αλγόριθμος ξεκινά με τις ατομικές υποφόρμουλες της f και προχωρεί με τις πιο σύνθετες – πολύπλοκες υποφόρμουλες.

2.4. Γλώσσες μοντελοποίησης συστημάτων

Η απευθείας χρήση των πιο πάνω μοντέλων για την περιγραφή πολύπλοκων συστημάτων επεξεργασίας και μεταφοράς είναι πολύ δύσκολη. Για αντιμετώπιση αυτών των δυσκολιών τις τελευταίες δεκαετίες έχουν προταθεί διάφορα πρότυπα τα οποία διαθέτουν χαρακτηριστικά που επιτρέπουν την σταδιακή και συνδυαστική μοντελοποίηση πολύπλοκων συστημάτων. Ιδιαίτερη σημασία δόθηκε στα πιο κάτω σημεία

- Παραλληλισμός και επικοινωνία: Γλώσσες μοντελοποίησης συστημάτων θα πρέπει να διαθέτουν τελεστές για να περιγράφουν σύνθετα συστήματα, φυσικά κατανεμημένα, που δρουν παράλληλα και αλληλεπιδρούν μεταξύ τους για την επίτευξη κάποιων στόχων.
- Πιθανοτική και χρονική συμπεριφορά: Για την μελέτη της αβεβαιότητας και των χρονικών περιορισμών που πιθανόν να περιλαμβάνονται σε ένα σύστημα είναι απαραίτητη η ενθυλάκωση εννοιών τυχαιοποίησης και χρόνου στους τελεστές μιας γλώσσας μοντελοποίησης.

- Παραμετρικότητα και συνθετικότητα: Ένα σύστημα συνήθως απαρτίζεται από πολλά υποσυστήματα τα οποία δυνατόν να παρουσιάζουν την ίδια δομή αλλά διαφέρουν σε κάποιες σταθερές, ή παραμέτρους. Για εύκολη περιγραφή τέτοιων συστημάτων απαιτούνται ανάλογοι τελεστές.

- Αυστηρή σημασιολογία: Μία γλώσσα μοντελοποίησης συστημάτων πρέπει να συνοδεύεται από μια ακριβή και σαφή διατύπωση της συμπεριφοράς συστημάτων γραμμένων σε αυτή. Αυτό επιβάλλεται για την ύπαρξη εμπιστοσύνης σε οποιαδήποτε συμπεράσματα διεξαχθούν κατά την ανάλυση ενός μοντέλου.

Βασικότερα τυπικά πρότυπα σχεδιασμού συστημάτων περιλαμβάνουν τα πιο κάτω:

- Petri nets [19]: Βασίζονται σε συστήματα μεταβάσεων των οποίων η δυναμική συμπεριφορά επιτυγχάνεται μέσω περάσματος tokens. Έχουν επεκταθεί έτσι ώστε να περιγράφουν χρονική συμπεριφορά, αλλά υστερούν στο σημείο της εύκολης σύνθεσης και περιγραφής της πιθανοτικής συμπεριφορά συστημάτων.

- I/O Αυτόματα [20]: Συστήματα μεταβάσεων με πιθανοτική, χρονική και παραμετρική επένδυση. Όπως και τα Petri Nets δεν διευκολύνουν την ιεραρχική και παραμετρική σύνθεση συστημάτων ως συνδυασμό των υποσυστημάτων τους.

- State charts και message charts [21]: Γραφικές γλώσσες που έχουν επεκταθεί με χρονική και πιθανοτική συμπεριφορά. Παρόλο που επιτρέπουν εύκολα τη σύνθεση συστημάτων δεν επιτρέπουν την παραμετρική περιγραφή και την εύκολη επαναχρησιμοποίηση κώδικα.

- UML [22]: Γραφική γλώσσα με πλούσια σύνταξη που έχει αναπτυχθεί για ανάπτυξη λογισμικού. Το μειονέκτημά της είναι η έλλειψη αυστηρής σημασιολογίας που επιτρέπει την ανάλυση μετά από το σχεδιασμό.

- Άλγεβρες Διεργασιών [23-26]: Οι άλγεβρες διεργασιών είναι ένα πρότυπο σχεδιασμού και ανάλυσης συστημάτων που συνοδεύεται από καλά διατυπωμένη σημασιολογία και που επιτρέπει τη μοντελοποίηση και ανάλυση συστημάτων με συνδυαστικό τρόπο. Άλγεβρες διεργασιών που έχουν προταθεί μέχρι σήμερα είναι ικανές να περιγράψουν συστήματα που περιλαμβάνουν χαρακτηριστικά όπως η έννοια του χρόνου, σταθερές ή κινητές τοπολογίες και πιθανοτική συμπεριφορά.

Σύμφωνα με την πιο πάνω μελέτη, επιλέχθηκε το πρότυπο των αλγεβρών διεργασιών για τη μοντελοποίηση δικτυακών συστημάτων και αναπτύχθηκε η άλγεβρα διεργασιών QoSPA η οποία περιγράφεται σε επόμενη παράγραφο.

3. Η Άλγεβρα Διεργασιών QoSPA

Στην έρευνα αυτή θα χρησιμοποιηθεί η άλγεβρα διαδικασιών QoSPA (Quality of Service Process Algebra), με την οποία θα μπορούμε να περιγράψουμε τα συστήματα που μας ενδιαφέρουν και μέσω της σημασιολογίας της να μεταφράζουμε στο μοντέλο των Labeled Concurrent Markov Chains. Η QoSPA είναι εμπνευσμένη από τις άλγεβρες διεργασιών ACSR [27] και PACSR [28] οι οποίες δημιουργήθηκαν για τη μοντελοποίηση και ανάλυση συστημάτων πραγματικού χρόνου. Θα ακολουθήσει μια σύντομη περιγραφή της ACSR, και στη συνέχεια θα δοθούν η σύνταξη και η σημασιολογία της QoSPA.

Μια υψηλού επιπέδου άποψη για τα κατανεμημένα συστήματα πραγματικού χρόνου (distributed real-time systems) είναι ότι τα διάφορα κομμάτια (components) του συστήματος συναγωνίζονται για την πρόσβασή τους στους διάφορους πόρους (recourses), επικοινωνώντας μεταξύ τους όπως χρειάζεται. Για να μπορεί το πιο πάνω να περιγραφεί πλήρως, δημιουργήθηκε μια πραγματικού χρόνου διαδικασιακή άλγεβρα (real time process algebra), η ACSR. Κάθε διαδικασία μπορεί να εμπλακεί σε δύο ειδών ενέργειες: να επικοινωνήσει με άλλες διαδικασίες διαμέσου γεγονότων (events) τα οποία δεν χρειάζονται χρόνο ή να επικοινωνήσει διαμέσου ενεργειών (actions) τα οποία χρειάζονται χρόνο. Για να εκτελεστεί μια ενέργεια χρειάζεται πρόσβαση σε ένα σύνολο από πόρους και χρειάζεται μη μηδενική ποσότητα χρόνου η οποία μετράται από ένα καθολικό ρολόι (implicit global clock). Οι πόροι είναι σειριακά επαναχρησιμοποιήσιμοι, και η πρόσβαση σε αυτούς καθορίζεται από προτεραιότητες. Μια διαδικασία η οποία προσπαθεί να έχει πρόσβαση σε ένα πόρο ο οποίος χρησιμοποιείται από μια διαδικασία μεγαλύτερης προτεραιότητας μπλοκάρεται από το να συνεχίσει.

Η έννοια του πόρου, η οποία είναι ήδη πολύ σημαντική για τον καθορισμό (specification) των συστημάτων πραγματικού χρόνου, επιπρόσθετα παρέχει ένα βολικό μηχανισμό (convenient abstraction mechanism) όσο αφορά την πιθανοτική πλευρά της συμπεριφοράς ενός συστήματος, όπως αποτυχία ενός φυσικού μέσου

(physical device). Τα φυσικά μέσα (επεξεργαστής, μνήμη, δίαυλοι επικοινωνίας κτλ) είναι τα είδη των αντικειμένων τα οποία θεωρούνται ως πόροι στην ACSR. Έτσι είναι φυσικό να χρησιμοποιούνται οι πόροι ως μέσο για την διερεύνηση της επίδρασης των αποτυχιών (failures) στην απόδοση ενός συστήματος. Τα βασικά συστατικά μιας άλγεβρας διεργασιών είναι η έννοια της *διεργασίας* και η έννοια του *καναλιού*, η πρώτη από τις οποίες επιτρέπει την περιγραφή υποσυστημάτων ενός συστήματος, τα οποία δρουν παράλληλα και ανεξάρτητα από άλλα υποσυστήματα, και το δεύτερο αποτελεί το σημείο επικοινωνίας και συντονισμού ανάμεσα σε διεργασίες. Οι τελεστές μιας άλγεβρας διεργασιών περιλαμβάνουν τη σειριακή και την παράλληλη σύνθεση και επιτρέπουν την ιεραρχική περιγραφή συστημάτων πράγμα εξαιρετικά χρήσιμο για τη συνθετική μοντελοποίηση των πολλαπλών στρωμάτων που πιθανόν να απαρτίζουν ένα δικτυακό σύστημα και, στη συνέχεια, τη συνθετική ανάλυση του συστήματος.

Στην άλγεβρα που αναπτύξαμε επεκτείνουμε την πιο πάνω άποψη παράλληλων συστημάτων. Υποθέτουμε ότι ένα σύστημα αποτελείται από ένα πεπερασμένο σύνολο από πόρους, τους οποίους χρησιμοποιούμε για να περιγράψουμε φυσικούς πόρους όπως κανάλια επικοινωνίας ή επεξεργαστές. Η χρήση πόρων ή και καναλιών συνοδεύονται από προτεραιότητες, οι οποίες εκφράζουν την προτεραιότητα με την οποία θέλουμε να διεκδικήσουμε τον πόρο ή το κανάλι και έτσι να εκφράσουμε στοιχεία προτεραιότητας που παρουσιάζονται π.χ. σε αρχιτεκτονικές DiffServ.

Η έννοια της πιθανότητας παρέχεται στην QoSPA μέσω της έννοιας του πόρου. Συγκεκριμένα, υποθέτουμε ότι οι πόροι ενός συστήματος δυνατόν να αποτύχουν κατά τη διάρκεια της εκτέλεσης του συστήματος, και συσχετίζουμε κάθε πόρο με μια πιθανότητα η οποία συλλαμβάνει την πιθανότητα αποτυχίας. Για κάποιο πόρο, έστω r, γράφουμε ως p(r) αυτή την πιθανότητα αποτυχίας, και ως $\bar{r}$ την εμφάνιση της αποτυχίας.

Όσον αφορά την έννοια του χρόνου, η QoSPA επιτρέπει σε κάθε διαδικασία να εμπλακεί σε τριών ειδών ενέργειες. Ένα από αυτά παίρνει μία μονάδα χρόνου, ενώ τα άλλα δύο δεν παίρνουν χρόνο:

- Χρονική ενέργεια: Συμβολίζεται ως $\varnothing$ και σημαίνει την πάροδο μιας μονάδας χρόνου. Σε μία παράλληλη σύνθεση διεργασιών για να περάσει μια μονάδα

χρόνου πρέπει όλες οι διεργασίες να είναι σε θέση να επιτρέψουν την πάροδο χρόνου.

- Γεγονότα: Τα γεγονότα στην QoSPA παρέχουν τη βάση για το συγχρονισμό των διαφόρων οντοτήτων. Ένα γεγονός ορίζεται ως μια δυάδα (α,p), όπου α είναι το label του γεγονότος και p η προτεραιότητά του. Τα labels αποτελούν το σύνολο L = $\ell \cup \bar{\ell} \cup \{\tau\}$, όπου αν α είναι ένα label τότε $\bar{\alpha}$ είναι το αντίστροφο label. Το ειδικό label τ παρουσιάζεται στις περιπτώσεις που δύο αντίστροφα labels εκτελούνται ταυτόχρονα. Συμβολίζουμε τα γεγονότα ως a, b, κλπ.

- Ενέργειες χρήσης πόρων: Οι ενέργειες αυτές απαιτούν πρόσβαση σε ένα σύνολο από πόρους οι οποίοι είναι δυνατόν να παρουσιάσουν βλάβη. Για παράδειγμα η ενέργεια {(r,p)} δηλώνει τη χρήση του πόρου r με προτεραιότητα p. Αντίθετα η ενέργεια {($\bar{r}$,p)} δηλώνει ότι ο πόρος r έχει αποτύχει και έχει προτεραιότητα p. Συμβολίζουμε ενέργειες χρήσης πόρων ως A, B. Επίσης γράφουμε ρ(A) για το σύνολο των πόρων που χρησιμοποιεί η ενέργεια A και p({r_1, …, r_n}) όλοι οι πόροι r_1, …, r_n να είναι διαθέσιμοι, δηλαδή p({r_1, …, r_n}) = p(r_1)·…· π(r_n).

Χρησιμοποιούμε τους ελληνικούς χαρακτήρες ***α*** και ***β*** για να δηλώσουμε ενέργειες οποιασδήποτε από τις πιο πάνω κατηγορίες.

3.1. Σύνταξη

Η γραμματική της QoSPA έχει ως εξής. Έστω ότι P, Q είναι διεργασίες της QoSPA. Οι πιο κάτω γραμματική περιγράφει τη σύνταξη των διεργασιών της QoSPA.

P ::= NIL | (a,p). P | A:P | ∅:P | b→P | P + Q
| P || Q | P\F | $[P]_I$ | rec X. P

Η διεργασία NIL αντιπροσωπεύει την αδρανή διεργασία. Στη γραμματική υπάρχουν τρεις προθεματικοί τελεστές, ένας για κάθε είδος ενεργειών. Ο πρώτος (a,n).P εκτελεί το γεγονός (a, n) και προχωρεί στο P, ενώ ο δεύτερος A:P εκτελεί την δέσμευση ενός πόρου και ακολούθως προχωρεί στο P. Τέλος ο ∅:P αδρανεί για μία μονάδα χρόνου και προχωρεί στο P. Η διεργασία b→P, προχωρεί στη διεργασία P δεδομένου ότι η συνθήκη b είναι αληθής. Η διεργασία P+Q, αντιπροσωπεύει την μη-ντετερμινιστική επιλογή μεταξύ δύο διεργασιών. Η διεργασία P||Q περιγράφει την παράλληλη

σύνθεση δύο διεργασιών. Στην περίπτωση αυτή οι διεργασίες εκτελούνται ανεξάρτητα ή αλληλεπιδρούν όταν εκτελούν γεγονότα και συγχρονίζονται στις χρονικές ενέργειες. Στην P\F, όπου $F \subseteq L$, τα labels F χρησιμοποιούνται μόνο από την P. Δηλαδή τα συστατικά της διεργασίας P μπορούν να χρησιμοποιήσουν αυτά τα labels για να αλληλεπιδρούν, αλλά δεν μπορούν να τα χρησιμοποιήσουν για την αλληλεπίδραση με το περιβάλλον της P. Το $[P]_I$, όπου το I είναι ένα σύνολο πόρων, δημιουργεί μια διεργασία η οποία προκρατεί τη χρήση των πόρων του I για αυτήν, επεκτείνοντας κάθε ενέργεια στο P με τους πόρους I με προτεραιότητα 0. Τέλος η διεργασία rec X.P καθορίζει αναδρομή.

3.2. Σημασιολογία

Η σημασιολογία της QoSPA δίνεται μέσω δύο συστημάτων μεταβάσεων που ορίζουν την πιθανοτική και τη μη-ντετερμινιστική συμπεριφορά διεργασιών. Για να γίνει αυτό χρησιμοποιούμε ως αντικείμενα ερμηνείας σχηματισμούς-ζεύγη της μορφής (P, W) που αντιπροσωπεύουν μία διεργασία P στον 'κόσμο' W, όπου το W συλλαμβάνει την κατάσταση στην οποία βρίσκονται οι πόροι του συστήματος (αν έχουν υποστεί πιθανοτική βλάβη ή όχι). Επομένως χωρίζουμε το σύνολο όλων των σχηματισμών σε δύο κλάσεις:

$$S_n = \{(P,W) \mid \mathrm{imr}(P) \subseteq W\}, \text{ και}$$

$$S_p = \{(P,W) \mid \mathrm{imr}(P) \not\subset W\},$$

όπου η συνάρτηση imr(P) επιστρέφει το σύνολο όλων των πόρων που απαιτούνται για χρήση από τη διεργασία P μέσα στην επόμενη χρονική στιγμή (ακριβής ορισμός μπορεί να δοθεί επαγωγικά στη δομή της P).

Σύμφωνα με αυτή την θεωρία, ο αρχικός σχηματισμός μιας διεργασίας P είναι ο (P, ∅), όπου δηλαδή ο κόσμος της διεργασίας είναι κενός. Για να υπολογίσουμε τον κόσμο της διεργασίας και τους διαθέσιμους πόρους προς αυτήν, πρέπει να εκτελεστεί μια πιθανοτική μετάβαση ως εξής:

$$\frac{(P,W) \in S_p, \quad B = \{r \text{ or } \bar{r} \mid r \in imr(P)\}, \quad \pi = p(A)}{(P,W) \xrightarrow{\pi} (P,B)}$$

Αφού γίνει αυτό, το κόσμος W δεν αλλάζει εκτός και αν πραγματοποιηθεί κάποια χρονική ενέργεια. Σε αυτό το νέο σχηματισμό οι διαθέσιμοι πόροι είναι γνωστοί και είναι δυνατόν να συνεχίσουμε με την εκτέλεση όλων των δυνατών ενεργειών. Ο κόσμος W δεν αλλάζει παρά μόνο αν πραγματοποιηθεί κάποια χρονική ενέργεια. Τότε, όπως θα δούμε και στους κανόνες που ακολουθούν, το σύστημα επαναφέρεται στον κενό κόσμο από όπου πιθανοτικές μεταβάσεις μπορούν και πρέπει να ξαναγίνουν για περαιτέρω εξέτασης του συστήματος.

Ένα παράδειγμα πιθανοτικών μεταβάσεων δίνεται από την πιο κάτω διεργασία. Έστω ότι έχουμε τους πόρους r_1 και r_2 και ισχύουν $p(r_1) = ½$ και $p(r_2) = ¼$. Τότε η διεργασία $P = \{(r_1,2),(\bar{r}_2,3)\} : Q$ έχει τις εξής μεταβάσεις:

$$(P,0) \xrightarrow{1/8} (P,\{r_1,r_2\}) \qquad (P,0) \xrightarrow{1/8} (P,\{\bar{r}_1,r_2\})$$

$$(P,0) \xrightarrow{3/8} (P,\{r_1,\bar{r}_2\}) \qquad (P,0) \xrightarrow{3/8} (P,\{\bar{r}_1,\bar{r}_2\})$$

Στον Πίνακα 1 ορίζουμε το μη-ντετερμινιστικό σύστημα μεταβάσεων της άλγεβρας διεργασιών. Αυτό ορίζεται από ένα σύνολο κανόνων που ερμηνεύουν κάθε ένα από τους τελεστές της άλγεβρας. Πιο κάτω εξηγούμε τους κανόνες:

- Οι κανόνες **ActI** και **ActT** και **Idle** για τους προκαθορισμένους τελεστές αποτελούν αξιώματα. Ο **ActI** αναφέρεται για τα στιγμιαία γεγονότα και ο **ActT** για τις ενέργειες πρόσβαση σε πόρους και ο **Idle** στην πάροδο μιας χρονικής στιγμής. Είναι φανερό πως στα στιγμιαία γεγονότα ο κόσμος W διατηρείται, ενώ, όπως αναφέραμε και πιο πάνω με την πάροδο χρόνου μηδενίζεται.
- Οι κανόνες **ChoiceL** και **ChoiceR**, είναι κανόνες μη-ντετερμινιστικής επιλογής. Είναι οι ίδιοι και για όλα τα είδη ενεργειών.
- Για την παράλληλη εκτέλεση διεργασιών ορίζουμε τέσσερις κανόνες. Οι πρώτοι τρεις **ParIL, ParIR** και **ParCom** είναι παρόμοιοι αναφέρονται στα στιγμιαία

γεγονότα. Συγκεκριμένα, **ParIL** και **ParIR** δείχνουν ότι όταν δύο γεγονότα πρόκειται να εκτελεστούν παράλληλα, εκτελούνται ανεξάρτητα μεταξύ τους. Για το συγχρονισμό, υπάρχει ο κανόνας **ParCom**, με προϋπόθεση ότι αν η P εκτελεί ένα γεγονός *a* τότε η Q θα εκτελεί την αλληλενέργεια $\bar{a}$. Αυτό επιτρέπει το συγχρονισμό ανάμεσα σε παράλληλες διεργασιών μέσω καναλιών. Ο τέταρτος κανόνας αναφέρεται στην παράλληλη εκτέλεση χρονικών ενεργειών. Για να εκδηλωθούν τέτοιες ενέργειες πρέπει όλες οι υποδιεργασίες να είναι έτοιμες να συγχρονιστούν. Αυτό βεβαιώνει πως το πέρασμα χρόνου συμβαίνει με βάση ένα καθολικό ρολόι.

- Οι κανόνες για περιορισμό, **ResT** και **ResI** αναφέρονται σε περιπτώσεις όπου ένα υποσύνολο των στιγμιαίων γεγονότων δεν μπορεί να χρησιμοποιηθεί από το σύστημα. Ο κανόνας **ResI** τονίζει το γεγονός ότι μπορούν να χρησιμοποιηθούν γεγονότα που βρίσκονται εκτός του υποσυνόλου που είναι σε περιορισμό. Ο κανόνας **ResT** επιτρέπει το γεγονός ότι οι ενέργειες παραμένουν ανεπηρέαστες από οποιοδήποτε είδος περιορισμού.
- Τέλος ο τελεστής **(recX.P, B)** δηλώνει αναδρομή. Η σημασιολογία του τελεστή αυτού δίνεται από τον κανόνα **Rec** και ο όρος P[recX.P / X] αντικαθιστά τον όρο «recX.P / X» κάθε φορά που πραγματοποιείται το X στη P.

ActI $(e.P,B)\xrightarrow{e}(P,B)$ **ActT** $(A:P,B)\xrightarrow{\mathrm{A}}(P,B)$ if $\rho(A)\subseteq(B)$

Idle $(\varnothing:P,B)\xrightarrow{\varnothing}(P,\varnothing)$ **Cond** $\dfrac{(P,B)\xrightarrow{\alpha}(P',B')}{(b\to P,B)\xrightarrow{\alpha}(P,B')}$ if $b=TRUE$

ChoiceL $\dfrac{(P,B)\xrightarrow{\alpha}(P',B')}{(P+Q,B)\xrightarrow{\alpha}(P',B')}$ **ChoiceR** $\dfrac{(Q,B)\xrightarrow{\alpha}(Q',B')}{(P+Q,B)\xrightarrow{\alpha}(Q',B')}$

ParT $\dfrac{(P,B)\xrightarrow{\varnothing}(P',B'),(Q,B)\xrightarrow{\varnothing}(Q',B')}{(P\parallel Q,B)\xrightarrow{\varnothing}(P'\parallel Q',\varnothing)}$

ParIL $\dfrac{(P,B)\xrightarrow{\alpha}(P',B')}{(P\parallel Q,B)\xrightarrow{\alpha}(P',B')}$ **ParIR** $\dfrac{(Q,B)\xrightarrow{\alpha}(Q',B')}{(P\parallel Q,B)\xrightarrow{\alpha}(Q',B')}$

$$\textbf{ParCom}\ \frac{(P,B)\xrightarrow{(a,p)}(P',B'),(Q,B)\xrightarrow{(\bar{a},q)}(Q',B')}{(P\parallel Q,B)\xrightarrow{(\tau,p+q)}(P'\parallel Q',B')}$$

$$\textbf{ResT}\ \frac{(P,B)\xrightarrow{A}(P',B')}{(P\setminus F,B)\xrightarrow{A}(P'\setminus F,B)} \qquad \textbf{ResI}\ \frac{(P,B)\xrightarrow{(a,n)}(P',B')}{(P\setminus F,B)\xrightarrow{(a,n)}(P'\setminus F,B')}\quad a,\bar{a}\notin F$$

$$\textbf{Rec}\ \frac{(P[recX.P/X],B)\xrightarrow{\alpha}(P',B')}{(recX.P,B)\xrightarrow{\alpha}(P',B')}$$

Πίνακας 1: Σημασιολογία της QoSPA

Σημειώνουμε ότι η σημασιολογία της QoSPA όπως την έχουμε ορίσει πιο πάνω αντιστοιχεί σε κάθε διεργασία ένα LCMC (Παράγραφος 2.3). Κατά συνέπεια μπορούμε να εφαρμόσουμε μοντελο-έλεγχο στο μοντέλο σύμφωνα με τις τεχνικές και τους αλγόριθμους που έχουν προταθεί στη βιβλιογραφία, για παράδειγμα [6, 28, 29]. Επίσης στο μοντέλο αυτό ισχύουν οι γνώστες έννοιες ισοδυναμίας συστημάτων όπως ισχυρή και ασθενής διπροσομοίωση [8, 30].

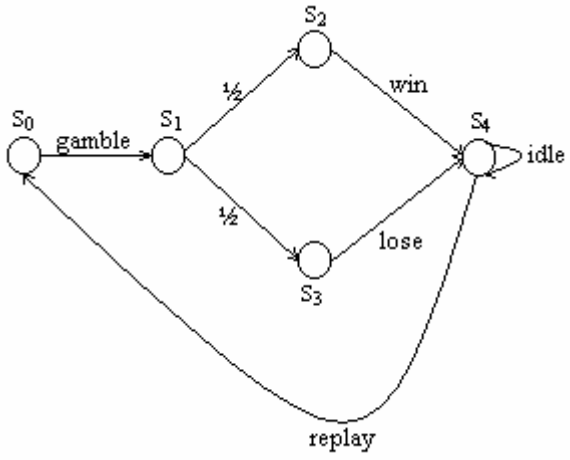
4. Υπολογισμός συμπεριφορών long-run average

Πέραν του μοντελοελέγχου έχουμε επεκτείνει το μοντέλο αυτό έτσι ώστε να μπορεί να γίνει ανάλυση των long run average χαρακτηριστικών του συστήματος (όπως και στο TPS, Παράγραφος, 2.1). Στην Παράγραφο αυτή διασκευάζουμε τον αλγόριθμο του TPS στο μοντέλο του LCMC. Για την επέκταση του μοντέλου χρησιμοποιείται η εξής έννοια του πειράματος. Ένα πείραμα είναι μια τριάδα Π = (Reset, Outcome, f), όπου

- Reset είναι ένα σύνολο από labels$\subseteq$Act, τα οποία ικανοποιούν τον τερματισμό του πειράματος Π,
- Outcome είναι ένα σύνολο από labels$\subseteq$Act, τα οποία αντιστοιχούν στις ποσότητες των οποίων θέλουμε να μετρήσουμε το long run average, και
- f είναι μια συνάρτηση η οποία f: Outcome$\rightarrow$R επιστρέφει το κόστος για κάθε label$\in$Outcome.

Ένα Extended Labeled Concurrent Markov Chain (ELCMC) είναι ένα LCMC (S_n, S_p, Act, $\rightarrow n$, $\rightarrow p$, s_0, t) με ένα πείραμα Π=(Reset, Outcome, f).

Το σχήμα που ακολουθεί δείχνει το LCMC L το οποίο ανταποκρίνεται στο παιγνίδι και το πείραμα Π το οποίο μετρά το μέσο όρο κέρδους κάθε ριξίματος του συστήματος που προαναφέραμε στο TPS μοντέλο.



Reset: win, lose

Outcome: win, lose

f(win) = +1

f(lose) = -1

Σ'αυτή την παράγραφο θα παρουσιάσουμε τον αλγόριθμο υπολογισμού των long-run average συμπεριφορών για το μοντέλο των LCMC Αρχικά θα δώσουμε τον

σε κάποιο ψηλό επίπεδο και στη συνέχεια στο υποκεφάλαιο 4.1 θα δώσουμε τους ορισμούς των sub-LCMCs και των end components, οι οποίοι θα χρησιμοποιηθούν στο επόμενο υποκεφάλαιο. Στο υποκεφάλαιο 4.2 προτείνουμε ένα αλγόριθμο για την εύρεση της long run average συμπεριφοράς ενός συστήματος δεδομένου ενός πειράματος. Τέλος στο υποκεφάλαιο 4.3 θα παρουσιάσουμε τη δομή της υλοποίησης μας, καθώς και μια ψηλού επιπέδου σύνδεση του αλγόριθμου και της υλοποίησης.

Αλγόριθμος σε ψηλό επίπεδο

Τα δεδομένα εισόδου του αλγόριθμού μας είναι ένα ELCMC και το αποτέλεσμά του είναι η μέγιστη και η ελάχιστη τιμή του long run average αποτελέσματος του πειράματος που περικλείεται στο ELCMC.

Όπως δείξαμε προηγουμένως, μας ενδιαφέρουν μόνο οι συμπεριφορές οι οποίες ικανοποιούν το κατηγόρημα I. Για να ικανοποιεί όμως μια συμπεριφορά το I πρέπει να εκτελεί απείρως πολλά πειράματα ή να εκτελεί απείρως πολλά actions. Δεδομένου

όμως ότι ο γράφος μας είναι πεπερασμένος η μόνη περίπτωση να ικανοποιείται το I είναι στον γράφο μας να υπάρχουν κάποιοι κύκλοι. Γι' αυτό το λόγο βρίσκουμε τους μέγιστους κύκλους του γράφου μας και επιλέγουμε από αυτούς όσους περιέχουν ακμές των οποίων τα labels ανήκουν στο Reset ή στο Outcome σύνολο. Αυτό γίνεται διότι αν δεν περιέχουν ακμές των οποίων τα labels ανήκουν στο Reset ή στο Outcome σύνολο τότε δεν θα υπήρχε νόημα να ψάξουμε την long run average συμπεριφορά.

Στη συνέχεια μετατρέπουμε όλους τους κύκλους που επιλέξαμε έτσι ώστε να ικανοποιούν το κατηγόρημα I με πιθανότητα 1 για όλες τις χρονοδρομολογήσεις. Το πιο πάνω γίνεται με αντικατάσταση κύκλων που δεν περιέχουν συμπεριφορές που μας ενδιαφέρουν με απλούς κόμβους. Τέλος βρίσκουμε τις μέγιστες και τις ελάχιστες τιμές του long run average αποτελέσματος του κάθε κύκλου και επιλέγουμε το μέγιστο από τα μέγιστα και το ελάχιστο από τα ελάχιστα.

4.1. Βοηθητικοί ορισμοί

Τα end components αντιπροσωπεύουν το σύνολο των καταστάσεων και ενεργειών, οι οποίες μπορούν να επαναληφθούν άπειρα πολλές φορές σε μια συμπεριφορά με πιθανότητα μεγαλύτερη του 0.

Ορισμός 1 (sub-LCMCs): Δεδομένου ενός LCMC L = (Sn, Sp, Act, $\rightarrow$n, $\rightarrow$p, s_0, t), ένα sub-LCMC είναι μια τριάδα (S, $\rightarrow$n', $\rightarrow$p'), όπου S $\subseteq$ (Sn $\cup$ Sp), $\rightarrow$n' $\subseteq \rightarrow$n και $\rightarrow$p' $\subseteq \rightarrow$p.

Ορισμός 2 (end component): Δεδομένου ενός LCMC L = (Sn, Sp, Act, $\rightarrow$n, $\rightarrow$p, s_0, t), ένα sub-LCMC (S, $\rightarrow$n', $\rightarrow$p') είναι end component εάν ευσταθούν οι πιο κάτω όροι:

- Κλειστότητα (closure):
  - για όλους τους κόμβους s$\in$S, αν $s \overset{\alpha}{\rightarrow}_n t \in \rightarrow_{n'}$, τότε t$\in$S.
  - $\forall s \in S \cap S_p \qquad \sum_{\left(s \overset{\pi}{\rightarrow} t\right) \in \rightarrow p'} \pi = 1$
- Συνεκτικότητα (connectivity): Εάν

$$\rightarrow_{n''} = \left\{ (s,t) \in S \times S \mid \exists\, a \in Act \wedge s \overset{\alpha}{\rightarrow}_{n'} t \right\}$$

$$\rightarrow_{p''} = \left\{ (s,t) \in S \times S \mid \exists\, \pi \in (0,1] \wedge s \overset{\pi}{\rightarrow}_{p'} t \right\}$$

τότε ο γράφος (S, $\rightarrow$n'', $\rightarrow$p''), είναι αυστηρά συνδεδεμένος (strongly connected).

Λέμε ότι ένα end component (S, $\rightarrow$n, $\rightarrow$p) περικλείεται σε ένα sub-LCMC (S', $\rightarrow$n', $\rightarrow$p') εάν:

$$S \subseteq S' \quad \wedge \quad \rightarrow_n \subseteq \rightarrow_{n'} \quad \wedge \quad \rightarrow_p \subseteq \rightarrow_{p'}$$

Ένα end component (S, $\rightarrow$n, $\rightarrow$p) ονομάζεται μέγιστο (maximal) σε ένα sub-LCMC (S', $\rightarrow$n', $\rightarrow$p'), εάν δεν υπάρχει άλλο end component (S'', $\rightarrow$n'', $\rightarrow$p'') το οποίο περιέχεται στο (S', $\rightarrow$n', $\rightarrow$p') και περιλαμβάνει πλήρως (properly contains) το (S, $\rightarrow$n, $\rightarrow$p). Ορίζουμε ως maxEC(S', $\rightarrow$n', $\rightarrow$p') το σύνολο των μέγιστων end component στο (S', $\rightarrow$n', $\rightarrow$p'). Το σύνολο maxEC(S, $\rightarrow$n, $\rightarrow$p) μπορεί να υπολογιστεί σε χρόνο πολυωνυμικό $\left(|S| + |\rightarrow_n| + |\rightarrow_p|\right)$ χρησιμοποιώντας απλούς αλγόριθμους γράφων. Ο αλγόριθμος εύρεσης των maxEC(S, $\rightarrow$n, $\rightarrow$p) παρουσιάζεται στο κεφάλαιο 4.3 (κόμβος 3 στο δέντρο υλοποίησης).

Δεδομένου μιας συμπεριφοράς ω = $X_0Y_0X_1A_1$...., ορίζουμε:

$$S_\omega = \{ s \mid \overset{\infty}{\exists}\kappa . X\kappa = s \}$$

$$\rightarrow_{n\,\omega} = \left\{ s \overset{a}{\rightarrow}_n t \mid \overset{\infty}{\exists}\kappa\ X_\kappa \overset{a}{\rightarrow}_n X_{\kappa+1},\ X_\kappa = s,\ X_{\kappa+1} = t \right\}$$

$$\rightarrow_{p\,\omega} = \left\{ s \overset{\pi}{\rightarrow}_n t \mid \overset{\infty}{\exists}\kappa\ X_\kappa \overset{\pi}{\rightarrow}_n X_{\kappa+1},\ X_\kappa = s,\ X_{\kappa+1} = t \right\}$$

όπου $\overset{\infty}{\exists}\kappa$ σημαίνει υπάρχουν άπειρα πολλά διαφορετικά κ.

Το sub-LCMC (S_ω, $\rightarrow n_\omega$, $\rightarrow p_\omega$) αντιστοιχεί στις καταστάσεις και ενέργειες (actions, events, ή πιθανοτικές μεταβάσεις), οι οποίες επαναλαμβάνονται άπειρα πολλές φορές στη συμπεριφορά ω.

Θεώρημα 1 (Θεμελιώδης θεώρημα των end components): Για όλα τα $s \in S$ και όλες τις χρονοδρομολογήσεις,

$$\Pr_s^n\ ((S_\omega, \rightarrow_{n_\omega}, \rightarrow_{p_\omega}) \text{ είναι ένα } \textit{end component}) = 1$$

4.2. Αλγόριθμος εύρεσης long run average συμπεριφορών

4.2.1. Αλγόριθμος ελέγχου του μοντέλου (model checking algorithm):

Έστω το ELCMC E = (Sn, Sp, Act, $\rightarrow n$, $\rightarrow p$, s_0, t, Reset, Outcome, f), το οποίο είναι το αποτέλεσμα της σύνδεσης ενός LCMC L = (Sn, Sp, Act, $\rightarrow n$, $\rightarrow p$, s_0, t) και ενός πειράματος Π (Reset, Outcome, f).

Ορίζουμε τα όρια αποτελέσματος (threshold outcomes) $\bar{T}_s^+$ και $\bar{T}_s^-$ του E να είναι αντίστοιχα η μέγιστη και η ελάχιστη τιμή του long run average αποτελέσματος του πειράματος Π, τα οποία μπορούν να δημιουργηθούν με μη μηδενική πιθανότητα κάτω από μια χρονοδρομολόγιση, αρχίζοντας από τον κόμβο s_0.

Ορισμός 3 (όρια αποτελέσματος (threshold outcomes)): Για όλα τα $s \in S$, ορίζουμε τα όρια αποτελέσματος $\bar{T}_s^+$ και $\bar{T}_s^-$ ως:

$$\bar{T}_s^+ = \sup\left\{a \in \Re \mid \exists n.\Pr_s^n\left(I \wedge \lim_{n\to\infty} Hn(\omega) \geq a\right) > 0\right\}$$

$$\bar{T}_s^- = \inf\left\{a \in \Re \mid \exists n.\Pr_s^n\left(I \wedge \lim_{n\to\infty} Hn(\omega) \leq a\right) > 0\right\}$$

Η πραγματική τιμή του $\bar{\Pi} \bowtie \alpha$ (E) για όλα τα $s \in S$, μπορεί να υπολογιστεί αν συγκρίνουμε τα όρια αποτελεσμάτων με το α, σύμφωνα με το πιο κάτω θεώρημα:

Θεώρημα 2: Για $\bowtie \in \{\leq, <\}$ έχουμε

$$s \models \bar{\Pi} \bowtie \alpha(E) \text{ αν } \bar{T}_s^+ \bowtie \alpha.$$

Για $\bowtie \in \{\geq, >\}$ έχουμε

$$s \models \overline{\prod} \bowtie \alpha(E) \text{ ανν } \bar{T}_s^- \bowtie \alpha$$

Ο πιο κάτω αλγόριθμος υπολογίζει τα όρια αποτελέσματος. Χρησιμοποιεί τους αλγόριθμους 2, 3 και 4, τους οποίους θα περιγράψουμε αργότερα.

Αλγόριθμος 1 (Όρια αποτελέσματος):

Είσοδος: ELCMC E = (Sn, Sp, Act, $\rightarrow$n, $\rightarrow$p, s_0, Reset, Outcome, f) και κάποιο $s \in (Sn \cup Sp)$

Έξοδος: $\bar{T}_s^+$ και $\bar{T}_s^-$ για το s

Μέθοδος:

1. Έστω $\{(S_1, \rightarrow_{n_1}, \rightarrow_{p_1}), \ldots\ldots, (S_m, \rightarrow_{n_m}, \rightarrow_{p_m})\} = \max EC(E)$. Για κάθε $1 \leq i \leq n$, δημιουργούμε ένα LCMC Li = (Sn_i, Sp_i, Act_i, $\rightarrow n_i$, $\rightarrow p_i$, s_i), όπου $Act_i \subseteq Act$ και δημιουργείται λόγω των περιορισμών των S_i, $\rightarrow n_i$ και $\rightarrow p_i$. Για $s \in (Sn \cup Sp)$ έστω:

$$Ms = \{i \in [1...n] \mid Si \text{ είναι επιτεύξιμο στο E από το } s\}$$

Βάση του θεωρήματος 1, για μια συμπεριφορά ω, υπάρχει με πιθανότητα 1 ένα $i \in [1...n]$ τέτοιο ώστε $(S_\omega, \rightarrow_{n_\omega}, \rightarrow_{p_\omega}) \subseteq (S_i, \rightarrow_{n_i}, \rightarrow_{p_i})$. Εάν μια συμπεριφορά ω ικανοποιεί το I (η περίπτωση που μας αφορά), το όριο $\liminf_{n\to\infty} Hn(\omega)$ δεν εξαρτάται από οποιεσδήποτε αρχικές καταστάσεις της ω. Παρόμοια συμβαίνει και για το όριο $\limsup_{n\to\infty} Hn(\omega)$.

Έστω $\bar{T}_{t,i}^+$ και $\bar{T}_{t,i}^-$ να είναι τα όρια αποτελέσματος που σχετίζονται με την κατάσταση $t \in S_i$, τα οποία υπολογίζονται στο LCMC Li. Αφού κάθε Li $1 \leq i \leq n$ είναι αυστηρά συνδεδεμένο (strongly connected), μπορούμε να αποδείξουμε ότι τα $\bar{T}_{t,i}^+$ και $\bar{T}_{t,i}^-$ δεν εξαρτούνται από το $t \in S_i$. Έτσι θα γράφουμε απλώς $\bar{T}_i^+$ και $\bar{T}_i^-$. Από τα πιο πάνω συμπεραίνουμε ότι:

$$\overline{T}_s^+ = \max_{i \in Ms} \overline{T}_i^+ \qquad \overline{T}_s^- = \min_{i \in Ms} \overline{T}_i^- \tag{1}$$

Για να λύσουμε το πρόβλημα του ελέγχου του μοντέλου, αρκεί να υπολογίσουμε τα $\overline{T}_i^+$ και $\overline{T}_i^-$ για όλα τα $1 \le i \le n$.

2. Υπολόγισε το σύνολο

$$\ell = \left\{ i \in Ms \;\middle|\; \exists s,t \in S_i \wedge \exists a \in (\mathrm{Re}set \vee Outcome), s \xrightarrow{a}_n t \right\}$$

Για κάθε $i \notin \ell$, οι συμπεριφορές που προκύπτουν από το Li δεν ικανοποιούν το I. Έτσι για $i \in \{1...n\} - \ell$ θεωρούμε

$$\overline{T}_i^+ = -\infty \text{ και } \overline{T}_i^- = \infty \tag{2}$$

με σκοπό να μην επηρεάζεται το τελικό αποτέλεσμα.

3. Χρησιμοποιώντας τον αλγόριθμο 2, μετατρέπουμε κάθε $Li = (Sn_i, Sp_i, Act_i, \rightarrow n_i, \rightarrow p_i, s_i, t_i)$, $i \in \ell$ σε ένα LCMC $\tilde{L}i = (\tilde{S}n_i, \tilde{S}p_i, \tilde{A}ct_i, \rightarrow\tilde{n}_i, \rightarrow\tilde{p}_i, \tilde{s}_i, \tilde{t}_i)$, έτσι ώστε να ευσταθεί το κατηγόρημα I με πιθανότητα 1 για όλες τις χρονοδρομολογήσεις.

Έστω $\tilde{T}_i^+$ και $\tilde{T}_i^-$ να είναι τα όρια αποτελέσματος που υπολογίζονται στα LCMC $\tilde{L}i$ για $i \in \ell$. Για $i \in \ell$ μπορεί να αποδειχτεί ότι:

$$\overline{T}_i^+ = \tilde{T}_i^+ \qquad \overline{T}_i^- = \tilde{T}_i^- \tag{3}$$

4. Χρησιμοποιώντας τον αλγόριθμο 3 υπολογίζουμε τα σύνολα:

$$K^- = \left\{ i \in \ell \;\middle|\; \tilde{T}_i^- = +\infty \right\} \tag{4}$$

$$K^+ = \left\{ i \in \ell \;\middle|\; \tilde{T}_i^+ = +\infty \right\} \tag{5}$$

5. Το κατηγόρημα I ισχύει με πιθανότητα 1 στο LCMC $\tilde{L}i$ $i \in \ell$, για κάθε χρονοδρομολόγιση. Αυτό μας επιτρέπει να αγνοήσουμε το κατηγόρημα I όταν

εργαζόμαστε στο $\tilde{L}i$, γεγονός που μας οδηγεί στη σύνδεση του υπολογισμού των $\tilde{T}_i^+$ και $\tilde{T}_i^-$ και της λύσης ενός προβλήματος βελτιστοποίησης για ένα semi-Markov LCMC.

Ο αλγόριθμος 4 υλοποιεί την πιο πάνω σύνδεση για να υπολογιστούν τα $\tilde{T}_i^+$ και $\tilde{T}_i^-$ για όλα τα $i \notin (K^+ \cup K^-)$. Τα όρια των αποτελεσμάτων $\bar{T}_s^+$ και $\bar{T}_s^-$ στο L μπορούν να υπολογιστούν χρησιμοποιώντας τα (1), (2), (3), (4) και (5).

4.2.2. Απαλείφοντας τις μη-I συμπεριφορές

Ο αλγόριθμος 2 μετατρέπει ένα LCMC σε ένα άλλο σχετικό LCMC, στο οποίο το κατηγόρημα I ισχύει με πιθανότητα 1. Η ιδέα του αλγόριθμου είναι η ακόλουθη:

Από το Θεώρημα 1, το κατηγόρημα I παίρνει την τιμή λάθος (false) με πιθανότητα μεγαλύτερη του 0, εάν το LCMC περιέχει ένα επιτεύξιμο end component το οποίο δεν έχει ακμή edge∈(Reset∪Outcome). Με το να εξαλείψουμε αυτά τα end components, τότε μπορούμε να εξασφαλίσουμε ότι το I ισχύει με πιθανότητα 1. Τα απειλητικά end components εξαλείφονται με το να αντικαταστήσουμε κάθε ένα από αυτά με μια απλή κατάσταση, και με το να διώξουμε όλες τις ενέργειες που ανήκουν σε αυτά. Αφού οι καταστάσεις που δημιουργούμε δεν περιέχουν ακμή edge∈(Reset∪Outcome), μπορούμε να αποδείξουμε ότι η μετατροπή αφήνει τα όρια αποτελέσματος ανεπηρέαστα όπως εκφράστηκαν στο (3).

Αλγόριθμος 2 (I-μετατροπή):

Είσοδος: LCMC L = (Sn, Sp, Act, $\rightarrow$n, $\rightarrow$p, s)

Έξοδος: LCMC $\tilde{L}$ = ($\tilde{S}$n, $\tilde{S}$p, $\tilde{A}$ct, $\rightarrow\tilde{n}$, $\rightarrow\tilde{p}$, $\tilde{s}$)

Μέθοδος:

Για κάθε s∈Sn, έστω

$$\rightarrow_n' = \left\{ s \overset{a}{\rightarrow}_n t \mid s,t \in S, \quad a \in Act, a \notin (\mathrm{Re}set \cup Outcome \right\}$$

το σύνολο των μη ντετερμινιστικών καταστάσεων των οποίων τα labels δεν ανήκουν ούτε στο Reset ούτε στο Outcome σύνολο του πειράματος.

Επίσης, έστω

$$\left\{(B_1, \rightarrow_{n_1}, \rightarrow_{p_1}), ..., (B_m, \rightarrow_{n_m}, \rightarrow_{p_m})\right\} = \max EC(Sn \cup Sp, \rightarrow_n', \rightarrow_p)$$

Το LCMC $\tilde{L}$ δημιουργείται από το L, με το να καταρρεύσουν όλα τα end components $(B_i, \rightarrow_{n_i}, \rightarrow_{p_i})$ σε μια απλή κατάσταση $\tilde{s}_i$, για κάθε 1≤ i ≤ m. Συμβολίζουμε την πιο πάνω κατάρρευση ως $(B_i, \rightarrow_{n_i}, \rightarrow_{p_i}) \cong \tilde{s}_i$. Το νέο σύνολο των καταστάσεων δίνεται από τη σχέση:

$$\tilde{S} = S \cup \{\tilde{s}_1, ..., \tilde{s}_m\} - \bigcup_{i=1}^{m} B_i$$

Το σύνολο των μη ντετερμινιστικών καταστάσεων ορίζεται ως

$$\rightarrow_{\tilde{n}} = \rightarrow_{\tilde{n}_1} \cup \rightarrow_{\tilde{n}_2} \cup \rightarrow_{\tilde{n}_3},$$

όπου

$$\rightarrow_{\tilde{n}_1} = \left\{ s \overset{a}{\rightarrow}_n t \;\middle|\; s,t \in S - \bigcup_{i=1}^{m} B_i, \quad s \overset{a}{\rightarrow}_n t \in \rightarrow_n \right\}$$

$$\rightarrow_{\tilde{n}_2} = \left\{ s \overset{a}{\rightarrow}_n \tilde{s}_i \;\middle|\; s \in S - \bigcup_{i=1}^{m} B_i, \; t \in B_i, \quad s \overset{a}{\rightarrow}_n t \in \rightarrow_n, (B_i, \rightarrow_{n_i} \rightarrow_{n_i}) \cong \tilde{s}_i \right\}$$

$$\rightarrow_{\tilde{n}_3} = \left\{ \tilde{s}_i \overset{a}{\rightarrow}_n s \;\middle|\; s \in S - \bigcup_{i=1}^{m} B_i, \; t \in B_i, \quad t \overset{a}{\rightarrow}_n s \in \rightarrow_n, (B_i, \rightarrow_{n_i} \rightarrow_{n_i}) \cong \tilde{s}_i \right\}$$

Το σύνολο των πιθανοτικών μεταβάσεων ορίζεται ως

$$\rightarrow_{\tilde{p}} = \rightarrow_{\tilde{p}_1} \cup \rightarrow_{\tilde{p}_2}$$

$$\rightarrow_{\tilde{p}_1} = \left\{ s \overset{\pi}{\rightarrow}_p t \;\middle|\; s,t \in S_p - \bigcup_{i=1}^{m} B_i, \quad s \overset{\pi}{\rightarrow}_p t \in \rightarrow_p \right\}$$

$$\to_{\tilde{p}_2} = \left\{ s \xrightarrow{\pi}_p \tilde{s}_i \;\middle|\; s \in S_p - \bigcup_{i=1}^{m} B_i, \quad t \in B_i, \quad s \xrightarrow{\pi}_p t \in \to_p, \quad (B_i, \to_{n_i} \to_{n_i}) \cong \tilde{s}_i \right\}$$

4.2.3. Υπολογισμός των συνόλων K^+ και K^-

Πιο κάτω δίδεται ο αλγόριθμος υπολογισμού των K^+ και K^-, ο οποίος βασίζεται στην αναζήτηση Reset και Outcome ακμών στα end components τα οποία υπολογίσαμε προηγουμένως και τα οποία μετατρέψαμε έτσι ώστε να ισχύει το κατηγόρημα Ι με πιθανότητα 1. Γενικά ένα end component ανήκει στο σύνολο K^+, εάν φιλτράρουμε τις μη ντετερμινιστικές ακμές του που ανήκουν στο σύνολο Reset και βρούμε σε αυτό ένα νέο end component το οποίο περιέχει ακμή (ακμές) η οποία ανήκει στο σύνολο Outcome. Ένα end component ανήκει στο σύνολο K^- εάν δεν υπάρχει σε αυτό ακμή edge$\in$Reset, ενώ υπάρχει ακμή edge'$\in$Outcome.

Αλγόριθμος 3 (υπολογισμός των K^+ και K^-):

Είσοδος: Το σύνολο ℓ.

Έξοδος: Τα σύνολα K^+ και K^-, όπως ορίζονται στα (4) και (5).

Μέθοδος:

Για κάθε $i \in \ell$

- Για κάθε $s,t \in \tilde{S}_i$, έστω

$$\to_{n_i}' = \left\{ s \xrightarrow[\tilde{n}]{a} t \;\middle|\; s,t \in \tilde{S}_i, \quad a \in Act, a \notin \mathrm{Re}set \right\}$$

να είναι το σύνολο των μη ντετερμινιστικών καταστάσεων του end component $(\mathrm{S_i}, \to_{n_i}, \to_{p_i})$, οι οποίες δεν ανήκουν στο σύνολο Reset.

Έστω

$$\left\{ \left(S_1, \to_{n_1}, \to_{p_1}\right), \ \ldots\ldots, \left(S_g, \to_{n_g}, \to_{p_g}\right) \right\} = \max EC((\tilde{\mathrm{S}}_\mathrm{i}, \to_{n_i}', \to_{p_i}))$$

Τότε, $i \in K^+$ ανν υπάρχει $j \in [1...g]$, $s,t \in S_j$ και $s \xrightarrow{a} t \in \rightarrow_j$ τέτοια ώστε $a \in Outcome$

- $i \in K^-$ ανν και οι δύο πιο κάτω συνθήκες ισχύουν:

  - Για όλα τα $s,t \in \tilde{S}_i$ και $s \xrightarrow{a}_{\tilde{n}} t$, $a \in Act, a \notin \mathrm{Re}set$

  - Υπάρχουν $s,t \in \tilde{S}_i$ τ τέτοια ώστε $s \xrightarrow{a}_{\tilde{n}} t$, $a \in Act, a \in \mathrm{Re}set$

4.2.4. Υπολογισμός των ορίων αποτελέσματος:

Αλγόριθμος 4 (υπολογισμός των $\tilde{T}_i^+$ και $\tilde{T}_i^-$):

Είσοδος: Το σύνολο ℓ.

Έξοδος: υπολογισμός των $\tilde{T}_i^+$ και $\tilde{T}_i^-$

Μέθοδος:

Όπως προαναφέρθηκε, σε αυτό το σημείο μπορούμε να συνδέσουμε τον υπολογισμό των $\tilde{T}_i^+$ και $\tilde{T}_i^-$ με τη λύση ενός προβλήματος βελτιστοποίησης για ένα semi-Markov LCMC.

Σε γενικές γραμμές για κάθε $i \in \ell$ δημιουργείται μια εξίσωση. Η λύση του συστήματος εξισώσεων που προκύπτει από τις πιο πάνω εξισώσεις είναι τα $\tilde{T}_i^+$ και $\tilde{T}_i^-$.

4.3. Υλοποίηση

Ο αλγόριθμος που υλοποιήσαμε προστέθηκε στο εργαλείο VERSA (Verification, Execution and Rewrite System for QoSPA).

Το εργαλείο VERSA είναι γραμμένο στη γλώσσα προγραμματισμού C++. Εκτός από τις βιβλιοθήκες της C++ έγινε και εκτεταμένη χρήση των βιβλιοθηκών της LEDA. Σκοπός του είναι η δημιουργία του LCMC δεδομένου μιας περιγραφής σε γλώσσα QoSPA. Το LCMC ακολούθως μπορεί να χρησιμοποιηθεί από τους χρήστες για επεξεργασία. Μία από τις διαδικασίες που μπορούμε να τρέξουμε είναι και η average με την οποία μπορούμε να εξάγουμε τα long run average χαρακτηριστικά του συστήματος.

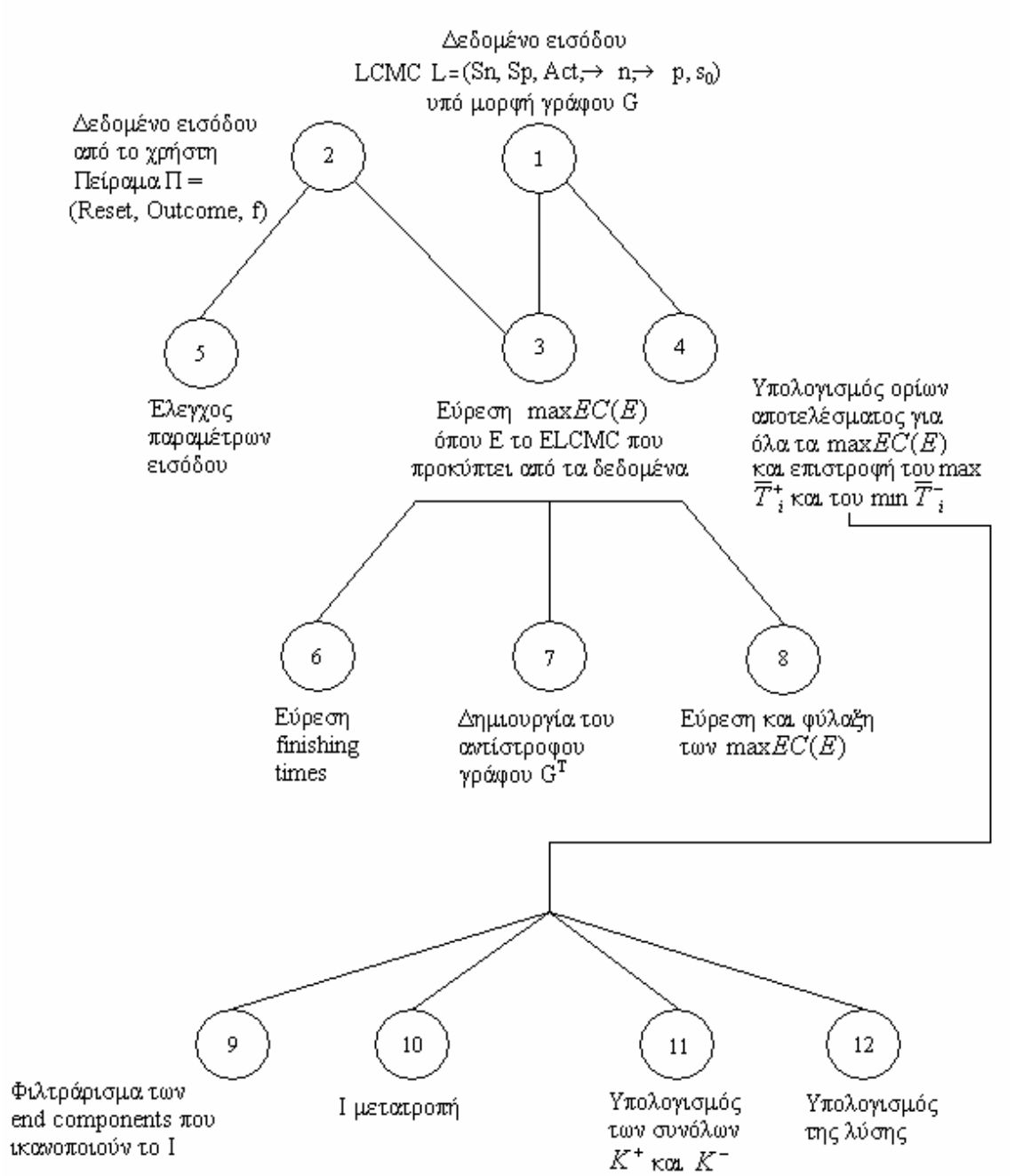
Το πιο κάτω σχεδιάγραμμα παρουσιάζει τη δομή της υλοποίησης του αλγόριθμου που δημιουργήσαμε. Κάθε φύλλο του δέντρου παρουσιάζει μια συνάρτηση που υλοποιήσαμε, ενώ κάθε ενδιάμεσος κόμβος είναι μια υψηλού επιπέδου αντιμετώπιση του προβλήματος. Πιο κάτω θα αναλυθεί σε μερικό βαθμό η υλοποίηση του προγράμματος, ενώ παράλληλα θα δείξουμε τη σύνδεση κάθε κόμβου με τον αλγόριθμο που προτείναμε στο 4.2.

- *Κόμβος 1:* Παρουσιάζει το πιθανοτικό σύστημα το οποίο αποτελεί δεδομένο εισόδου για το πρόγραμμά μας και είναι υπό μορφή γράφου (όπου το γράφο σε μελλοντικές αναφορές θα τον συμβολίζουμε με G). Ο γράφος αυτός δημιουργείται από το εργαλείο VERSA, αφού ο χρήστης περιγράψει σε αυτό το σύστημά του στη διαδικασιακή άλγεβρα PACSR (δες κεφάλαιο 4.4).

- *Κόμβος 2:* Αναφέρεται στο πείραμα για το οποίο ο χρήστης ενδιαφέρεται να βρει την long run average συμπεριφορά του συστήματος.

- *Κόμβος 3:* Αποτελεί την υλοποίηση του βήματος 1 του αλγόριθμου 1. Για την εύρεση των μέγιστων end components χρησιμοποιούνται οι κόμβοι-συναρτήσεις 6, 7 και 8.

- *Κόμβος 4:* Αναφέρεται στην εύρεση των $\overline{T}_s^+$ και $\overline{T}_s^-$ στο τέλος του βήματος 1 του αλγόριθμου 1 και χρησιμοποιεί τις διαδικασίες 9, 10, 11 και 12.



Σχήμα 3.4.1 - Δομή της υλοποίησης

- *Κόμβος 5:* Όπως προαναφέρθηκε χρησιμοποιείται για τον έλεγχο του δεδομένου εισόδου.
- *Κόμβος 6:* Υλοποιεί μια κατά βάθος διερεύνηση και ταξινομεί τους κόμβος του γράφου G σε φθίνουσα σειρά, ανάλογα με το χρόνο τερματισμού τους (finishing time). Με τον όρο χρόνος τερματισμού ενός κόμβου εννοούμε το

χρόνο στον οποίο έχει τελειώσει η επεξεργασία των κόμβων παιδιών του, όσο και του ίδιου. Μια μονάδα χρόνου παρέρχεται όταν κάνουμε προσπάθεια (επιτυχημένη ή αποτυχημένη) για να επισκεφτούμε ένα κόμβο.

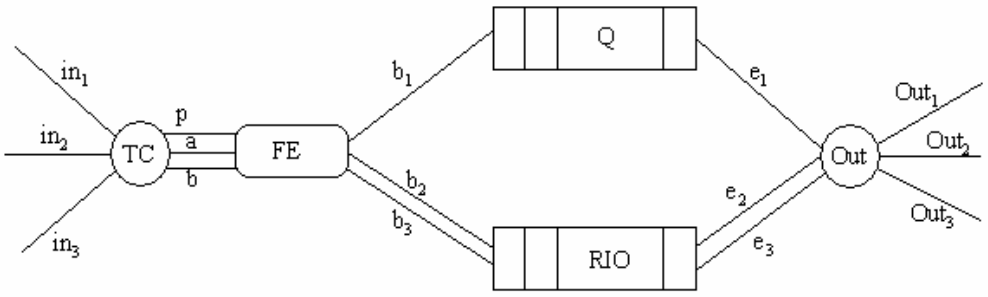
- *Κόμβος 7:* Δημιουργία του αντίστροφου γράφου G^T.
- *Κόμβος 8:* Το τελευταίο βήμα για την εύρεση των end components. Υλοποιεί μια κατά βάθος διερεύνηση στον αντίστροφο γράφο G^T χρησιμοποιώντας τους κόμβους σε φθίνουσα σειρά ανάλογα με το χρόνο τερματισμού τους.
- *Κόμβος 9:* Αναφέρεται στην εύρεση του συνόλου ℓ στο βήμα 2 του αλγόριθμου 1. Η υλοποίηση αποτελείται από μια κατά βάθος διερεύνηση στα end components που προέκυψαν από τον κόμβο 3. Αν ένα end component βρεθεί ότι περιέχει ακμή η οποία να ανήκει στα σύνολα Reset∪Outcome (από το οποίο συμπεραίνουμε ότι ικανοποιεί το κατηγόρημα I), τότε αυτά φιλτράρονται. Διαφορετικά, αν το end component i βρεθεί ότι δεν περιέχει ακμή η οποία να ανήκει στα σύνολα Reset∪Outcome, του αναθέτουμε τις τιμές $\overline{T}_i^+ = -\infty$ και $\overline{T}_i^- = +\infty$, έτσι ώστε να μην επηρεάζουν το τελικό αποτέλεσμα.
- *Κόμβος 10:* Η συνάρτηση 10 υλοποιεί το βήμα 3 του αλγόριθμου 1 (αλγόριθμος 2 I-μετατροπή). Ο τρόπος υλοποίησης έχει ως εξής. Με μια κατά βάθος διερεύνηση στα end components της συνάρτησης 9, σβήνουμε τις ακμές edge$\in$ Reset∪Outcome. Ακολούθως τρέχουμε τον αλγόριθμο του κόμβου 2 σε κάθε end component ξεχωριστά. Αν σε κάποιο end component i βρεθεί κάποιο νέο end component έστω j, τότε αντικαθιστούμε το j με ένα κόμβο node και για κάθε ακμή από ένα κόμβο v$\in$i και v$\notin$j προς ένα κόμβο u$\in$j δημιουργούμε μια ακμή μεταξύ v και node. Για κάθε ακμή από ένα κόμβο u$\in$j προς ένα κόμβο v$\in$i και v$\notin$j δημιουργούμε την αντίστοιχη ακμή μεταξύ node και v.
- *Κόμβος 11:* Υλοποιεί το βήμα 4 του αλγόριθμου 1 (αλγόριθμος 3 εύρεση των συνόλων K^+ και K^-). Πάλι έχουμε μια κατά βάθος διερεύνηση στα end components του που προκύπτουν από το βήμα 9. Αν σε ένα end component i δεν βρούμε ακμή edge$\in$Reset, τότε προσθέτουμε το i στο σύνολο K^-. Διαφορετικά, αν ένας κόμβος περιέχει edge$\in$Reset, τότε με μια κατά βάθος διερεύνηση σβήνουμε όλες τις Reset ακμές από το end component i και

τρέχουμε τον αλγόριθμο του κόμβου 2. Αν βρεθεί ένα νέο end component, τότε προσθέτουμε το αρχικό στο σύνολο K^+

- *Κόμβος 12:* Υλοποιεί το βήμα 5 του αλγόριθμου 1 (αλγόριθμος 4)

5. Μοντελοποίηση διακομιστή DIFFSERV στην QoSPA

Ο διακομιστής που θα περιγράψουμε στηρίζεται στο πρωτόκολλο διαφοροποιημένων υπηρεσιών (DIFFSERV) και σκοπός του είναι να παρέχει στους χρήστες μιας καλύτερης κλάσης ποιότητα υπηρεσίας. Πιο κάτω δίδεται ο διακομιστής σχηματικά, καθώς και η περιγραφή των πέντε συστατικών που τον αποτελούν. Ταυτόχρονα γίνεται και η «μετάφραση» του πιο πάνω διακομιστή σε γλώσσα QoSPA, η οποία είναι και η είσοδος στο πρόγραμμά μας. Η «μετάφραση» αυτή είναι ένας τρόπος αυστηρής περιγραφής του διακομιστή.



Σχήμα 4.4.1 Δομή του διακομιστή

- **TC – Ο ελεγκτής κυκλοφορίας:** Υπεύθυνος για την επεξεργασία της κίνησης που φτάνει στο router. Προωθεί τα πακέτα στο FE ανάλογα με τη διαθεσιμότητα του token t_a.

$$\begin{aligned} TC \quad \stackrel{def}{=} \quad & in_1 \,.\, \text{if ta then } \overline{p} \,.\, TC \text{ else } TC \\ & + in_2 \,.\, \text{if ta then } \overline{a} \,.\, TC \text{ else } \overline{b} \,.\, TC \\ & + in_3 \,.\, \overline{b} \,.\, TC \\ & + \varnothing : TC \end{aligned}$$

- **FE – Η μηχανή προώθησης:** Υπεύθυνη για την προώθηση των πακέτων σε μια εκ των δύο ουρών (priority και RIO). Τα premium πακέτα οδηγούνται στο priority queue, ενώ τα assured και best effort στο RIO queue.

$$\begin{aligned} FE \overset{def}{=} & \; p \,.\, \bar{b}_1 \,.\, FE \\ & + a \,.\, \bar{b}_2 \,.\, FE \\ & + a \,.\, \bar{b}_3 \,.\, FE \\ & + \varnothing : FE \end{aligned}$$

- **Q – Η ουρά προτεραιότητας:** Q_i είναι η στοίβα προτεραιότητας η οποία περιέχει i πακέτα. Το Q_i, $i \neq 0$, $i \neq n$, μπορεί να κάνει τρεις ενέργειες:
  - Να στείλει πακέτο και να γίνει Q_{i-1},
  - Να παραλάβει πακέτο και να γίνει Q_{i+1},
  - Να παραμείνει αδρανές (idle)

$$\begin{aligned} Q_0 \overset{def}{=} & \; b_1 \,.\, Q_1 \\ & + \varnothing : Q_0 \end{aligned}$$

$$\begin{aligned} Q_i \overset{def}{=} & \; b_1 \,.\, Q_{i+1} \\ & + \bar{e}_1 \,.\, Q_{i-1} \\ & + \varnothing : Q_i \end{aligned}$$

$$\begin{aligned} Q_n \overset{def}{=} & \; b_1 \,.\, \overline{ofq} \,.\, Q_{bfn} \\ & + \bar{e}_1 \,.\, Q_{n-1} \\ & + \varnothing : Q_n \end{aligned}$$

- **RIO – Η ουρά RIO:** Έχει τρεις παραμέτρους:

- Μια λίστα από bits, τα οποία αντιπροσωπεύουν τα περιεχόμενα της ουράς. Αν το bit είναι 0 τότε έχουμε assured πακέτο, διαφορετικά αν είναι 1 έχουμε best effort πακέτο.
- Ένα ακέραιο αριθμό i, ο οποίος συμβολίζει τον αριθμό των assured πακέτων
- Ένα ακέραιο αριθμό j, ο οποίος συμβολίζει τον αριθμό των best effort πακέτων.

Η διαφορά από την ουρά Q είναι το ότι μπορεί να χάσει πακέτα με πιθανότητα ανάλογη της κατάστασης της στοίβας. Η πιθανότητα είναι διαφορετική για τα assured και τα best effort πακέτα.

$$RIO_{[],0,0} \overset{def}{=} b_2 \,.\, RIO_{[0],1,1} + b_3 \,.\, RIO_{[1],0,1} + \varnothing : RIO_{[0],0,0}$$

$$RIO_{xs{+}{+}[y],i,j} \overset{def}{=} b_2 \,.\, (\{p_{i/cap}\} : RIO_{[0]{+}{+}xs{+}{+}[y],i+1,j+1} + \{\overline{p_{i/cap}}\} : \bar{d}_a \,.\, RIO_{xs{+}{+}[y],i,j}) + b_3 \,.\, (\{p_{j/cap}\} : RIO_{[1]{+}{+}xs{+}{+}[y],i,j+1} + \{\overline{p_{j/cap}}\} : \bar{d}_p \,.\, RIO_{xs{+}{+}[y],i,j}) + \text{if } Q_0 \text{ then if } y=0 \text{ then } \bar{e}_2 \,.\, RIO_{xs,i-1,j-1} \text{ else } \bar{e}_3 \,.\, RIO_{xs,i,j-1} + \varnothing : RIO_{xs{+}{+}[y],i,j}$$

$$RIO_{xs{+}{+}[y],i,cap} \overset{def}{=} b_2 \,.\, (\{p_{i/cap}\} : \overline{ofr} \,.\, RIO_{xs{+}{+}[y],i,cap} + \{\overline{p_{i/cap}}\} : \bar{d}_a \,.\, RIO_{xs{+}{+}[y],i,cap}) + b_3 \,.\, (\{p_{j/cap}\} : \overline{ofr} \,.\, RIO_{xs{+}{+}[y],i,cap} + \{\overline{p_{j/cap}}\} : \bar{d}_p \,.\, RIO_{xs{+}{+}[y],i,cap}) + \text{if } Q_0 \text{ then if } y=0 \text{ then } \bar{e}_2 \,.\, RIO_{xs,i-1cap-1} \text{ else } \bar{e}_3 \,.\, RIO_{xs,i,cap-1} + \varnothing : RIO_{xs{+}{+}[y],i,cap}$$

Out – Η μηχανή εξόδου: Βγάζει τα πακέτα που παραλαμβάνει από τις δύο πιο πάνω ουρές στον έξω κόσμο. Δίνει προτεραιότητα στη στοίβα προτεραιότητας. Μπορεί να μεταφέρει το πολύ s πακέτα ανά μονάδα χρόνου.

$$Out_0 \overset{def}{=} \varnothing : Out_s$$

$$Out_i \overset{def}{=} e_1 . \overline{out}_1 . Out_{i-1} + e_2 . \overline{out}_2 . Out_{i-1} + e_3 . \overline{out}_3 . Out_{i-1} + \varnothing : Out_s$$

- **Rout – Ένας διακομιστής:** Η παράλληλη συνένωση των πιο πάνω διαδικασιών. Επικοινωνεί με το περιβάλλον μέσο καναλιών.

$$Rout = (TC \parallel FE \parallel Q_0 \parallel RIO_{[],0,0} \parallel Out_s) \setminus p, a, b, b_1, b_2, b_3, e_1, e_2$$

Στα πιο κάτω παραδείγματα θεωρούμε ως γράφο το LCMC που δημιουργείται μέσω της σημασιολογίας της QoSPA. Για κάθε περίπτωση δίνεται το πείραμα Π του οποίου η ένωση με το LCMC μας δίνει το ELCMC που μας ενδιαφέρει.

Ο αριθμός των overflow ανά μονάδα χρόνου: Από τις τρεις μετρήσεις που προτείναμε θα δείξουμε μόνο πως μπορεί να μετρηθεί η πρώτη για τον πιο πάνω διακομιστή. Ο συμβολισμός των overflow στην περιγραφή του διακομιστή δίδεται από τα σύμβολα $\overline{ofq}$ (συμβολίζει το overflow στην ουρά Q) και $\overline{ofr}$ (συμβολίζει το overflow στην ουρά RIO). Το πείραμα Π σε αυτή την περίπτωση είναι η τριάδα ({όλα τα actions}, {όλα τα είδη overflow}, {f(overflow)=1}). Εάν «τρέξουμε» τον αλγόριθμο μας σε αυτό το ELCMC, το $\overline{T}_s^-$ αποτέλεσμα είναι και το ζητούμενο. Η πιο πάνω μέτρηση θα μπορούσε να γίνει για κάθε κατηγορία ξεχωριστά, έτσι ώστε να βρούμε τον αριθμό των overflow ανά κατηγορία.

Δικαιοσύνη δικτύου: Η πιο πάνω μέτρηση αναφέρεται στη σύγκριση του ρυθμού αποστολής πακέτων σε κάθε ένα από τα in_1, in_2 και in_3 ξεχωριστά. Θα δούμε πως μπορεί να γίνει η μέτρηση για ένα από αυτά, έστω το in_1. Ξέρουμε ότι με κάθε αποτυχία αποστολής ενός πακέτου, στέλλεται μήνυμα από τον διακομιστή στην αντίστοιχη σύνδεση-αποστολέα του πακέτου για να ελαττώσει το ρυθμό αποστολής πακέτων. Το ιδανικό δίκτυο θα ήταν όλες οι συνδέσεις της ίδιας κλάσης να έχουν το ίδιο long run average αποστολής πακέτων. Έστω ότι στο διακομιστή μας υπάρχουν τρεις συνδέσεις-αποστολείς στην είσοδο in_1, τις οποίες ονομάζουμε in_{11} in_{12} in_{13}. Σε αυτή την περίπτωση θα τρέξουμε τρία πειράματα, όπου το πείραμα Πi είναι η τριάδα

({όλα τα actions}, {in_{1i}}, {f(in_{1i})=1}). Εάν «τρέξουμε» τον αλγόριθμο μας σε αυτά το ELCMC, μπορούμε να συγκρίνουμε τα αποτελέσματα των τριών πειραμάτων και να εξάγουμε τα συμπεράσματά μας.

Διαχωρισμός premium, assured και best effort συνδρομητών: Για να ελέγξουμε αν ένας συνδρομητής αξίζει να πληρώνει περισσότερα για να ανήκει σε μια καλύτερη κατηγορία υποθέτουμε σε κάθε σύνδεσμο in_1, in_2 και in_3 υπάρχει μόνο μια σύνδεση αποστολέας. Επίσης υποθέτουμε ότι αρχικά, όλες οι συνδέσεις έχουν τον ίδιο ρυθμό αποστολής πακέτων. Ένας τρόπος να το κάνουμε αυτό είναι να τρέξουμε τον αλγόριθμο μας στα ELCMC που προκύπτουν από τα πειράματα Πi = ({όλα τα actions}, {in_i}, {f(in_i)=1}) και το αποτέλεσμα κάθε φορά ανταποκρίνεται στο long run average ρυθμό αποστολής πακέτων ανά μονάδα χρόνου στην αντίστοιχη σύνδεση. Για να λειτουργεί με τον επιθυμητό τρόπο το δίκτυό μας, πρέπει $\overline{T}_{s1}^{-} > \overline{T}_{s2}^{-} > \overline{T}_{s3}^{-}$ και $\overline{T}_{s1}^{+} > \overline{T}_{s2}^{+} > \overline{T}_{s3}^{+}$. Η διαφορά των long run averages των τριών κατηγοριών θα είναι ανάλογη με τη συνδρομή που καταβάλλει κάθε κατηγορία.

Συνολικός αριθμός απορριπτόμενων πακέτων ανά μονάδα χρόνου: Η μέτρηση αυτή γίνεται παρόμοια με τη μέτρηση για τον αριθμό των overflow ανά μονάδα χρόνου. Το πείραμα Π σε αυτή την περίπτωση είναι η τριάδα ({όλα τα actions}, {όλα τα είδη overflow, πιθανοτικές απορρίψεις πακέτων}, {f(overflow)=1, f(πιθανοτικής απόρριψης)=1}). Εάν «τρέξουμε» τον αλγόριθμο μας σε αυτό το ELCMC, τα $\overline{T}_s^-$ και $\overline{T}_s^+$ αποτέλεσμα αντιστοιχούν στα όρια (καλύτερη και χειρότερη περίπτωση) του συστήματος. Τρέχοντας το σύστημα για διαφορετικά i και j μπορούμε να διαλέξουμε τα ιδανικά, τα οποία είναι αυτά που οδηγούν στα μικρότερα $\overline{T}_s^-$ και $\overline{T}_s^+$.

Το καλύτερο δίκτυο που θα προκύψει (όπου το καλύτερο είναι σχετικό με τον σκοπό που το χρειαζόμαστε) ίσως να μην είναι το καλύτερο για όλες τις παραμέτρους που αναφέραμε πιο πάνω. Αυτό οφείλεται στο γεγονός ότι η αλλαγή μιας παραμέτρου επηρεάζει περισσότερες από μια μέτρηση, άλλες θετικά και άλλες αρνητικά.

6. Συμπεράσματα

Έχουμε περιγράψει την άλγεβρα διεργασιών QoSPA η οποία έχει δημιουργηθεί για τη μοντελοποίηση συστημάτων που προέρχονται από εφαρμογές σε δικτυακά συστήματα. Η άλγεβρα αυτή επιτρέπει την ιεραρχική περιγραφή συστημάτων στοιχείο εξαιρετικά χρήσιμο για τη συνθετική μοντελοποίηση των πολλαπλών στρωμάτων που πιθανόν να απαρτίζουν ένα δικτυακό σύστημα. Επίσης συνδυάζει τους χρονικούς τελεστές και την πιθανοτική συμπεριφορά. Η σημασιολογία της άλγεβρας διεργασιών δίνεται ως συστήματα μεταβάσεων με μη-ντετερμινιστική και πιθανοτική συμπεριφορά (LCMC). Το μοντέλο αυτό έχει ήδη μελετηθεί στην περιοχή των τυπικών μεθόδων ανάλυσης συστημάτων και γι΄αυτό έχουν ήδη προταθεί μεθόδοι και αλγόριθμοι μοντελοελέγχου καθώς και εννοιών ισοδυναμίας, τις οποίες κληρονομεί το πρότυπο μας. Εμείς επεκτείναμε τις μεθόδους αυτές προτείνοντας αλγόριθμους υπολογισμού της ρυθμοαπόδοσης και των long-run average συμπεριφορών συστημάτων. Η καταλληλότητα της QoSPA για περιγραφή δικτυακών συστημάτων και των μεθόδων ανάλυσης για εκτίμηση της επίδοσης τους έχει δειχθεί μέσω της μελέτης του διακομιστή DIFFSERV.

Βιβλιογραφία

1. R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.
2. E. Clarke and R. Kurshnan. *Computer-aided Verification.* IEEE Spectrum, 33(6), 1996.
3. E. Clarke, O. Grumberg and D. Peled. *Model Checking*, MIT Press, 1999.
4. G. J. Holzmann. *The model checker SPIN*. IEEE Transactions on Software Engineering, 23(5). IEEE, 1997.
5. H. Hermanns, U. Herzog, U. Klehmet, V. Mertsiotakis and M. Siegle. *Compositional performance modeling with the TIPP-Tool*. Performance Evaluation, 39, 2000.
6. Bianco and R. de Alfaro. Model checking of probabilistic and nondeterministic systems. In Proceedings of *FST/TCS'95*, pages 499-513. Springer, 1995.
7. M. Bernardo and R. Gorrieri. *A tutorial on EMPA: A theory of concurrent processes with nondeterminism, priorities, probabilities and time*. In Theoretical Computer Science, 202(1-2):1-54, 1998.

8. K. Larsen and A. Skou. *Bisimulation through probabilistic testing*. Information and Computation, 94:1-28, 1991.

9. G. Lowe. Probabilistic *and prioritized models of timed CSP*. Theoretical Computer Science, 138:315-352, 1995.

10. Luca de Alfaro. *Formal Verification of Probabilistic Systems*. PhD thesis, December 1997.

11. C. A. R. Hoare, *Communicating Sequential Processes*. Prentice-Hall, 1985.

12. C. Derman. *Finite State Markovian Decision Processes*. Academic Press, 1970.

13. D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena Scientific, 1995.

14. C. Baier and M. Kwiatkowska. Model Checking for a Probabilistic Branching Time Logic with Fairness. Technical Report CSR-96-12, University of Birmingham, UK, 1996.

15. H. Karloff. *Linear Programming*. Progress in Theoretical Computer Science. Birkhauser, 1991.

16. Luca de Alfaro. How to Specify and Verify the Long-Run Average Behavior of Probabilistic Systems. In Proceedings *of LICS'98*. IEEE, 1998.

17. R. Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems.* PhD thesis, MIT, 1995.

18. M. Vardi. Automatic verification of probabilistic concurrent finite-state programs. In Proceedings of *FOCS'85*, pages 327-338. IEEE, 1995.

19. W. Reisig. *Petri-nets – an introduction*. EATCS Monograph on Theoretical Computer Science, volume 4. Springer-Verlag, 1985.

20. R. Segala and N. Lynch. Probabilistic simulations for Probabilistic Processes. In Proceedings of *CONCUR'94*, pages 481-496. Springer, 1994.

21. D. Harel. *Statecharts: a visual approach to complex systems.* Science of Computer Programming, 8(3):231-274, 1987.

22. G. Booch, J. Rumbaugh and I. Jacobson. *The Unified Modeling Language User Guide.* Addison Wesley, USA, 1999.

23. W. J. Fokkink. *Introduction to Process Algebra*. Texts in Theoretical Computer Science. Springer, 2000.

24. R. Milner. *Communicating and Mobile Systems*: *the π-Calculus*. Cambridge University Press, 1999.

25. J. Baeten amd P. Weijland. *Process Algebra*, volume 18 of Cambridge Tracts in Computer Science. Cambridge University Press, 1990.

26. J. A. Bergstra, A. Ponse and S.A. Smolka, editors, *Handbook of Process Algebra*. Elsevier Science Publishers, 2001.

27. P. Bremond-Gregoire, J. Choi and I. Lee. *A complete axiomatization of finite-state ACSR processes*. Information and Computation 138(2):124--159, 1997.

28. A. Philippou, O. Sokolsky, R.e Cleaveland, Insup Lee, S. Smolka. Probabilistic Resource Failure in Real-Time Process Algebra. In Proceedings of *CONCUR'98*, pages 389-404. Springer, 1998.

29. H. Hansson and B. Jonsson. *A logic for reasoning about time and probability*. Formal Aspects of Computing, 6:512—535. 1994.

30. A. Philippou, I. Lee, O. Sogolsky. Weak Bisimulation for Probabilistic Systems. In Proceedings of *CONCUR'00*. Springer, 2000.