Εργασία Αλγόριθμοι Επικοινωνιών

2022-2023

Σκύρλα Αγάθη, ΑΜ:1064888, Έτος:6’
Τυροβολά Αθανασία, ΑΜ:1064887, Έτος:6’

1. Περιγραφή

Το κείμενο πραγματεύεται την μελέτη της δρομολόγησης των μηνυμάτων σε πολλαπλούς προορισμούς σε ένα τετράγωνο πλέγμα n-κόμβων με την χρήση δρομολόγησης ένα-προς-πολλά (one-to-many routing). Αναλύει δύο αλγορίθμους που χρησιμοποιούν την δρομολόγηση αυτή, έναν τυχαιοποιημένο (randomized), ο οποίος επιτυγχάνει τη βέλτιστη απόδοση και έναν ντετερμινιστικό (deterministic), όπου αυτός είναι πιο αργός κατά $O(log^2 n)$. Οι αλγόριθμοι αυτοί, χρησιμοποιούν buffer σταθερού μεγέθους σε κάθε κόμβο. Ακόμη, αναφέρεται και στην περιγραφή ενός βέλτιστου ντετερμινιστικού αλγορίθμου, ο οποίος ωστόσο, απαιτεί μεγάλα buffer μεγέθους $O(c)$. Έτσι ώστε, να είναι δυνατή η δρομολόγηση πολλών πολυ-εκπεμπόμενων μηνυμάτων με χρήση της δρομολόγησης ένα-προς-πολλά.

2. Εισαγωγή

2.1. Πρόβλημα Multicast

Όταν μία πληροφορία πρέπει να μοιραστεί μεταξύ πολλών προορισμών, η Multicast, ή αλλιώς πολυεκπομπή, απαιτεί την παράδοση ενός μηνύματος από την πηγή του σε πολλούς προορισμούς. Η τεχνική Multicast μπορεί να χρησιμοποιηθεί για την ταξινόμηση στοιχείων σε παράλληλα αλλά και κατανεμημένα συστήματα. Έχει μελετηθεί και έχει λυθεί το πρόβλημα της πολυεκπομπής, για ένα μόνο μήνυμα, από ορισμένες μελέτες που ανέπτυξαν κατάλληλες στρατηγικές, για την επιλογή των μονοπατιών που συνδέουν την πηγή του μηνύματος με τους προορισμούς του και για τον προγραμματισμό της πραγματικής παράδοσης, κατά μήκος των επιλεγμένων μονοπατιών αποτελεσματικά.. Όμως, για την δρομολόγηση μεγάλων συνόλων μηνυμάτων αντιμετωπίζουν το πρόβλημα, όπως και στη δρομολόγηση σημείο-προς-σημείο, με ανεξάρτητη δρομολόγηση.

Για να λυθεί το πρόβλημα αυτό, αναπτύχθηκαν άνω και κάτω όρια για τον χρόνο που απαιτείται για την δρομολόγηση ένα προς πολλά σε ένα τετράγωνο πλέγμα. Δηλαδή, για να παραδοθεί ένα σύνολο μηνυμάτων πολλαπλής εκπομπής στους προορισμούς τους, όπου κάθε μήνυμα προέρχεται από μια ξεχωριστή πηγή, αλλά έχει πολλούς προορισμούς. Η πολυπλοκότητα ενός τέτοιου προβλήματος είναι η συνάρτηση f(n,m,d,c), όπου n το μέγεθος του πλέγματος, $m \leq n$ είναι ο αριθμός των μηνυμάτων που θα δρομολογηθούν, d είναι ο μέγιστος αριθμός προορισμών ανά μήνυμα και c είναι ο μέγιστος αριθμός μηνυμάτων που πρέπει να ληφθούν από οποιονδήποτε κόμβο. Εάν το c=1 και d τυχαίο, το πρόβλημα μπορεί να λυθεί με αρκετά απλό τρόπο σε χρόνο $O(\sqrt{n})$ με ουρές σταθερού μεγέθους, χρησιμοποιώντας τυπική δρομολόγηση μετάθεσης. Για να περιοριστεί το πρόβλημα στη δρομολόγηση μετάθεσης, προτείνεται να δημιουργηθούν τόσα αντίγραφα ενός μηνύματος όσα έχει προορισμούς και συσχετίζοντας το ένα αντίγραφο με ξεχωριστό προορισμό. Αλλά στην περίπτωση όπου c >1 είναι πολύ πιο δύσκολη, καθώς η γενίκευση της παραπάνω τεχνικής οδηγεί σε χρήση μεγάλων buffer και δημιουργία ελάχιστων αντιγράφων. Η δρομολόγηση (d,c) απαιτεί χρόνο εκτέλεσης $\Theta(\sqrt{dcn})$, ο οποίος δεν είναι βέλτιστος για δρομολόγηση ένα-προς-πολλά.

Η δρομολόγηση αυτή, διαφέρει από τη δρομολόγηση ένα-προς-πολλά, καθώς η δεύτερη δεν απαιτεί απαραίτητα να δημιουργούνται όλα τα αντίγραφα στην πηγή του μηνύματος και να μεταφέρονται χωριστά, μέσω του δικτύου στους αντίστοιχους προορισμούς του.

2.2. Δρομολόγηση ένα-προς-πολλά σε πλέγμα

Έστω ένα παράδειγμα n κόμβων πλέγματος δρομολόγησης ένα-προς-πολλά, με σύνολο μηνυμάτων otm(n,m,d,c), όπου $m \leq n$ ένα σύνολο μηνυμάτων. Κανένας κόμβος δεν είναι ο προορισμός περισσότερων από c μηνυμάτων συνολικά και κάθε κόμβος είναι η πηγή το πολύ ενός μηνύματος. Οι παράμετροι d και c αναφέρονται ως ο βαθμός και η συμφόρηση του συνόλου μηνυμάτων, αντίστοιχα, έτσι προκύπτει $d \leq n$ και $c \leq m$. Η μελέτη της δρομολόγησης αυτής στο πλέγμα, γίνεται σύμφωνα με το μοντέλο αποθήκευσης και προώθησης (store-and-forward). Κάθε μήνυμα βρίσκεται αρχικά στην ουρά εισόδου του κόμβου πηγής και παραμένει εκεί μέχρι να εισαχθεί στο δίκτυο. Το μήνυμα ταξιδεύει τυλιγμένο σε ένα πακέτο που αποτελείται από μια κεφαλίδα, που παραθέτει τις διευθύνσεις προορισμού, και ένα ωφέλιμο φορτίο, που περιέχει το πραγματικό σώμα του μηνύματος. Μόλις ένα πακέτο φτάσει σε έναν κόμβο προορισμού, το σχετικό μήνυμα αντιγράφεται στην ουρά εξόδου του κόμβου. Κάθε κόμβος του πλέγματος έχει ουρά εισόδου, ουρά εξόδου και buffer. Το πακέτο αφαιρείται από το δίκτυο μετά την επίσκεψη στον τελευταίο προορισμό.

3. Αποτελέσματα

Η έρευνα αυτή οδήγησε στα εξής:

- Η προφανής προσέγγιση της απλής αναπαραγωγής κάθε μηνύματος στον κατάλληλο αριθμό μηνυμάτων από σημείο-σε-σημείο και η δρομολόγηση αυτών ανεξάρτητα, δεν αποφέρει βέλτιστη απόδοση. Ένα αυθαίρετο σύνολο μηνυμάτων otm(n,m,d,c), με οποιοδήποτε σύνολο παραμέτρων n,m,d,c με $m \leq n$, $1 \leq d \leq n$ and $1 \leq c \leq m$, απαιτεί τουλάχιστον χρόνο $\Omega(\sqrt{cm + n})$ για να δρομολογηθεί, το οποίο αποτελεί συνδυασμό των εκτιμήσεων απλής διαμέτρου και εύρους ζώνης και το παραπάνω όριο είναι ανεξάρτητο από την παράμετρο d. Το κάτω όριο, παρουσιάζει εξάρτηση από το n, το οποίο αποτελεί το μέγεθος του πλέγματος

  <u>Απόδειξη:</u>

  Η υπόθεση γίνεται με βάση ότι $n \geq 4$, διαφορετικά το λήμμα είναι ασήμαντο. Έστω ένα τετράγωνο υποπλέγμα A με $n' = \max\{1, m/(4c)\}$ κόμβους και έστω ότι το B υποδηλώνει το υπόλοιπο πλέγμα. Αφού, $m \leq n$ έχουμε ότι $n' \leq n/4$. Έτσι, $|B| = n - n' \geq 3n/4$. Έστω, μία περίπτωση του otm(n,m,d,c) όπου τα m/4 μηνύματα που προέρχονται από κόμβους του B έχουν ακριβώς έναν προορισμό το καθένα στο A. Σαφώς, θα χρειαστεί χρόνος $\Omega(m/\sqrt{n'}) = \Omega(\sqrt{cm})$ για να εισαχθούν όλα αυτά τα μηνύματα στο A.

Έπειτα, με δύο λήμματα αποδεικνύουν ότι αρκεί η ανάπτυξη αλγορίθμων δρομολόγησης για σύνολα μηνυμάτων otm(n,m,d,c) με $m = n$ και $d \leq c$, αφού η δρομολόγηση όλων των άλλων συνόλων μηνυμάτων μπορεί να περιοριστεί στη δρομολόγηση αυτής της περίπτωσης χωρίς απώλεια απόδοσης και με buffer σταθερού μεγέθους.

- Εάν μπορούν να αποθηκευτούν έως και O(c) μηνύματα σε έναν κόμβο, μια κατάλληλη στρατηγική δρομολόγησης, η οποία αναπαράγει μηνύματα μόνο όταν είναι αρκετά κοντά στους προορισμούς τους, προσεγγίζει βέλτιστη απόδοση στη χειρότερη περίπτωση. Για μεγάλες τιμές c, αυτός ο αλγόριθμος δεν είναι πρακτικός, καθώς απαιτεί μεγάλα buffer μεγέθους O(c).
- Η επίτευξη βέλτιστου χρόνου χρησιμοποιώντας buffer σταθερού μεγέθους, είναι πολύ δύσκολη. Υπό αυτόν τον περιορισμό αναπτύχθηκε ένας βέλτιστος τυχαιοποιημένος αλγόριθμος και ένας ντετερμινιστικός αλγόριθμος, του οποίου η απόδοση είναι ο παράγοντας $O(\log^2 n)$ μακριά από το βέλτιστο.

Οι αλγόριθμοι βασίστηκαν σε διαχωρισμό των μηνυμάτων σε ομάδες c, ώστε τα μηνύματα με προορισμούς μικρές περιοχές του πλέγματος, να είναι ομοιόμορφα διασκορπισμένα. Ο τυχαιοποιημένος αλγόριθμος επιτυγχάνεται στην πιθανοτική ρύθμιση με τυχαία αντιστοίχιση μηνυμάτων σε ομάδες. Και στον ντετερμινιστικό χρησιμοποιώντας ένα πιο πολύπλοκο σχήμα που βασίζεται σε μια επαναλαμβανόμενη ανάθεση αντιγράφων μηνυμάτων σε ομάδες, που διέπεται από εξαιρετικά επεκτεινόμενα γραφήματα. Υποθετικά το μήνυμα διασχίζει μια σύνδεση σε σταθερό χρόνο. Οι τυχαιοποιημένοι και (υπό βέλτιστοι) ντετερμινιστικοί αλγόριθμοι που παρουσιάζονται στο κείμενο, μπορούν να ισχύουν για το εικονικό μοντέλο διακοπής, διότι χρησιμοποιούν buffer σταθερού μεγέθους.

Εάν και αυτά τα αποτελέσματα βασίστηκαν στην υπόθεση ότι, ένα μήνυμα διασχίζει μια σύνδεση σε σταθερό χρόνο. Μπορούν βέβαια εύκολα να αναδιαμορφωθούν, έτσι ώστε να διατηρηθεί το πιο ρεαλιστικό σενάριο όπου ο χρόνος μετάδοσης είναι μια γραμμική συνάρτηση του μήκους του μηνύματος, διατηρώντας τα ίδια χαρακτηριστικά βελτιστοποίησης.

4. Τυχαιοποιημένος Αλγόριθμος

Στη συνέχεια αναλύεται ένας βέλτιστος τυχαιοποιημένος αλγόριθμος δρομολόγησης ενός αυθαίρετου σετ μηνυμάτων otm(n,n,d,c) με $d \leq c$ με buffer σταθερού μεγέθους. Η βασική ιδέα του αλγόριθμου αυτού είναι, να χωρίζονται τα πακέτα σε c ομάδες, ώστε να μπορούμε να εξασφαλίσουμε οτι τα μηνύματα είναι πιθανό να παραδοθούν ομοιόμορφα σε $\Theta(c)$ ομάδες. Για να γίνει αυτό, αρχικά χωρίζουμε το πλέγμα σε c ομοιόμορφα κελιά και γύρω τους κυκλώνουμε κάθε ομάδα δημιουργώντας Χαμιλτονιανούς κύκλους οι οποίοι επιτρέπουν στα πακέτα κάθε ομάδας να επιχειρήσουν να παραδώσουν τα πακέτα στους προορισμούς τους εντός κάθε κελιού όταν η ομάδα φτάσει σε αυτό. Για παράδειγμα, για κάθε πακέτο x που μας δίνεται, το οποίο ανήκει στην ομάδα Gx με προορισμό το y σε κελί C, τα περισσότερα από τα υπολοιπα πακετα που πρεπει να φτάσουν σε κόμβους εντός του μπλοκ που περιέχει εναν κομβο y, είναι πιθανό να ανήκουν σε άλλες ομάδες από τη Gx, αρα είναι απίθανο η παράδοση του x στον κομβο y να παρεμποδιστεί από υψηλη συμφόρηση στο μπλοκ του y. Ωστόσο, η στρατηγική αυτή χρειάζεται ειδική διαχείριση ωστε να επιφέρει τα επιθυμητά βέλτιστα όρια σε χρόνο και μέγεθος buffer.

Ο αλγόριθμος αυτός είναι οργανωμένος σε φάσεις, σε κάθε μία από τις οποίες τα πακέτα επισκέπτονται ένα υποσύνολο από προορισμούς οι οποίοι δεν έχουν επισκεφτεί από άλλα πακέτα προηγουμένως, με τέτοιο τρόπο ώστε στο τέλος κάθε φάσης, ο συνολικός αριθμός παραδόσεων που πρέπει να γίνουν να μειώνεται. Γενικά υποθέτουμε πως c $\leq \frac{n}{16}$ εφόσον διαφορετικά η τετριμμένη στρατηγική, όπου όλα τα πακέτα κυκλοφορούν κατά μήκος ενός χαμιλτονιανού κύκλου του πλέγματος, αρκεί για να επιτευχθεί μέγιστη απόδοση.

Συνολικά, ο αλγόριθμος οργανώνεται σε K+1 φάσεις, όπου K=$log_{16}(\lceil c \cdot log^2 n/n \rceil)$. Σημειώνουμε πως για $c \leq n/log^2 n$ το K=0. Για $0 \leq i \leq K$ έχουμε $c_i = c/16^i$ και θεωρούμε τεμαχισμό του πλέγματος σε c_i τετραγωνα i-κελιά των n/c_i κόμβων το κάθε ένα, με την αρίθμηση να ταυτοποιεί έναν χαμιλτονιανό κύκλο των κελιών. Αντιθέτως αν υποθέσουμε πως ο τεμαχισμός κάθε κελιού σε τετράγωνα κελιά των $\sqrt{n/c_i}$ με $b_i = \sqrt{n/c_i}$ κόμβους. Κάθε μπλοκ έχει τουλάχιστον 4 κόμβους.

Για 0 ≤ i ≤ K-1, στην φάση i, τα πακέτα επισκέπτονται ένα υποσύνολο των προορισμών που δεν έχουν επισκεφτεί, έτσι ώστε στο τέλος κάθε φάσης, για καθε i-μπλοκ ο συνολικός αριθμός των πακέτων με προορισμούς που δεν έχουν επισκεφτεί στα πλέγματα είναι το πολύ $b_i c_{i+1} = b_i c_i/16$. Κατά τη φάση K, πραγματοποιούνται οι υπόλοιπες παραδόσεις.

Οι συγγραφείς υποστηρίζουν πως η ορθότητα και η ανάλυση χρόνου εκτέλεσης του αλγορίθμου βασίζονται σε 3 βασικά λήμματα τα οποία και αποδεικνύουν. Το πρώτο υποστηρίζει πως για $0 \leq i \leq K$, στην αρχή της φάσης i, ο αριθμός των πακέτων με προορισμούς που δεν έχουν επισκεφτεί μέσα σε κάθε i-μπλοκ είναι το πολύ $b_i c_i$, με υψηλή πιθανότητα. Το δεύτερο, παρουσιάζει πως στο τέλος της φάσης K , όλα τα παράγωγα έχουν φτιαχτεί, με υψηλή πιθανότητα. Και τέλος το τρίτο, αποδεικνύει πως για $0 \leq i \leq K$ η φάση i μπορεί να εκτελεστεί σε χρόνο $O(\sqrt{c_i n})$.

1. Για 0 ≤ i ≤ K, η φάση i εκτελείται όπως θα αναλύσουμε παρακάτω. Χωρίζουμε τα πακέτα σε c_i ομάδες $G_1, G_2, ..., G_{ci}$, όπου το κάθε ένα περιέχει το πολύ $2n/c_i$ πακέτα και διανέμουμε τις ομάδες ομοιόμορφα στους κόμβους ενός διακριτού κελιού c_i, ως εξής:
   a. Χωρίζουμε τα πακέτα σε c_i ομάδες $G'_1, G'_2, ..., G'_{ci}$, και αναθέτουμε κάθε πακέτο σε μία ομάδα με πιθανότητα $1/c_i$.

b. Για κάθε ομάδα G'_j με 1≤ j ≤c_i εξάγουμε $\lceil |G'_j|/(n/c_i) \rceil$ σετ των n/c_i πακέτων η κάθε μία. Διανέμουμε τα σετ αυτά ώστε να δημιουργηθούν οι c_i ομάδες $G_1, G_2, \ldots, G_{ci}$, με τέτοιο τρόπο ώστε κάθε ομάδα G_j είναι η ένωση το πολύ δύο από τα σετ μεγέθους (n/c_i).

c. Για 1≤ j ≤c_i στέλνουμε όλα τα πακέτα ομάδων G_j σε κελιά C_j, με το πολύ δύο πακέτα ανά κόμβο κελιού.

2. Παράλληλα για κάθε κελί C_j, επαναλαμβάνουμε τα ακόλουθα βήματα c_i φορές. a. Για κάθε πακέτο στο κελί, επιλέγουμε το πολύ rbi πακέτα (από αυτά στο κελί C_j) με προορισμούς εντός του μπλόκ και αφήνουμε τα πακέτα να επισκεφτούν όλους τους προορισμούς (εντός του μπλοκ).
b. Μεταφέρουμε κάθε πακέτο στην αντίστοιχη θέση στο επόμενο κελί C_{j+1}.

Άρα καταλήγουμε στο συμπέρασμα πως ενα αυθαίρετο σετ μηνυμάτων otm(n,n,d,c) με $d \geq c$ μπορεί να δρομολογηθεί με βέλτιστο χρόνο O($\sqrt{cn}$) με υψηλή πιθανότητα, με χωρητικότητα buffer O(1) ανά κόμβο.

5. Ντετερμινιστικός αλγόριθμος

Ένας ακόμα αλγόριθμος που αναλύεται ειναι ο ντετερμινιστικός αλγόριθμος που δρομολογεί κάθε otm(n,n,d,c) σετ μηνυμάτων με d ≤ c, με χρόνο $O(log^2 n\sqrt{cn})$ χρησιμοποιώντας buffer σταθερού μεγέθους. Υποθέτουμε ότι $c \geq log^4 n$, εφόσον διαφορετικά η αμελητέα μείωση σε (d,c) δρομολόγηση με σταθερό buffer οπως περιγράφηκε στην εισαγωγή επαρκεί για να επιτευχθεί το πάνω όριο χρόνου δρομολόγησης.

Η ιδέα του αλγορίθμου είναι κάπως παρόμοια με αυτή του τυχαιοποιημένου αλγορίθμου που περιγράψαμε προηγούμενα. Χωρίζουμε τα πακέτα σε ομάδες, έτσι ώστε οι παραδόσεις μηνυμάτων σε κάθε μπλοκ μίας ομάδας να δημιουργεί λιγότερη συμφόρηση από το αρχικό πρόβλημα. Ωστόσο, δεν μπορούμε να βασιζόμαστε στην τύχη ώστε να παραδοθούν ομοιόμορφα τα πακέτα σε ένα μπλοκ ανάμεσα στις ομάδες, και συνεπώς καταφεύγουμε σε ένα σύνθετο σχήμα που βασίζεται σε μια επαναλαμβανόμενη ανάθεση αντιγράφων των πακέτων σε ομάδες που ρυθμίζονται από διευρυμένους γράφους.

Οι γράφοι αυτοί ορίζονται ως εξής: για λ σταθερό , 0 < λ < 1 και με ακέραιους r και c , με 1≤ r,c ≤ m , ένας διμερής (πολύ-)γράφος G=(U,V) είναι ένας (m, c, r, λ) -διευρυντής αν |U|= m, |V|=c < m, deg(U)= r και deg(V)= rm/c και αν για κάθε S⊆U με |S|≤ |V|/r, οι ακμές που ξεκινούν από το S φτάνουν ενα σετ Γ(S) από κόμβους του V μεγέθους |Γ(S)|≥λ|S|r.

Δοθέντος ενός σετ μηνυμάτων otm(n,n,d,c),οι παραπάνω γράφοι χρησιμοποιούνται σε ντετερμινιστικό αλγόριθμο όπως φαίνεται παρακάτω. Υποθετουμε ότι καθε επεξεργαστης αποθηκεύει μια μετάθεση που αντιπροσωπεύει έναν$(n/log\ n, c, r, \lambda)$ -διευρυντή G=(U,V) με r=Θ(logn). .Μία μόνο μετάθεση επαρκεί για μια γεωμετρική σειρά τιμών του c, κυμαινόμενη απο $log^4 n$ έως n Επιπλέον,η μετάθεση μπορεί να αποθηκευτεί κατανεμημένα ανάμεσα στους κόμβους χρησιμοποιώντας σταθερό χώρο σε κάθε κόμβο, και οι προσβάσεις στον συσχετισμένο γράφο που απαιτεί ο αλγόριθμος μπορούν να γίνουν χωρίς αύξηση του συνολικού χρόνου εκτέλεσης. Διαμοιράζουμε αυθαίρετα το σετ μηνυμάτων σε $log\ \cdot\ n$ κλάσεις των n/log·n μηνυμάτων το καθένα. Όπως προηγουμένως, θεωρούμε το πλέγμα χωρισμένο σε c κελιά μεγέθους n/c, αριθμημένα έτσι ώστε να δημιουργούν έναν χαμιλτονιανό κύκλο. Διαδοχικά, κάθε κελί χωρίζεται σε $\sqrt{(n/c)}$ τετράγωνα μπλοκ μεγέθους $b\ = \sqrt{(n/c)}$.

Η δρομολόγηση κάθε κλάσης γίνεται ξεχωριστά, σε έναν αριθμό φάσεων. Στην πρώτη φάση για κάθε κλάση πραγματοποιούμε τον ακόλουθο αλγόριθμο.

1. Δημιουργούμε c ομαδες των Θ(n/c) πακέτων η κάθε μία. Δημιουργούμε r αντίγραφα κάθε πακέτου και αναθέτουμε τα αντίγραφα σε r απο τις c ομάδες όπως υπαγορεύεται από τον διευρυντή. Πιο συγκεκριμένα, αφήνουμε το ι-οστό πακέτο της κλάσης να ανταποκρίνεται στον κόμβο u_i του U, και την j-οστή ομαδα G_j να ανταποκρίνεται στον κόμβο u_j του V. Έπειτα τα r αντίγραφα του ι-οστού μηνύματος αναθέτονται στις ομάδες που σχετίζονται με τους κόμβους στο $G(\{u_i\})$.
2. Για 1≤j≤c ,στέλνουμε όλα τα αντίγραφα της ομαδα G_j στο κελι C_j, ομοιόμορφα κατανεμημένο στους κόμβους του κελιού.
3. Για κάθε κελί C_j παράλληλα, επαναλαμβάνονται τα ακόλουθα βήματα c φορές
    a. Για $a\ \succ\ 1/\lambda$ μία σταθερά. Για κάθε μπλοκ στο κελί, διαλέγουμε μέχρι abr πακέτα(απο αυτα στο κελι C_j) με προορισμούς εντός του μπλοκ.
    b. Χωρίζουμε το πολύ abr πακέτα επιλεγμενα για καθε μπλοκ σε ar παρτίδες μεγέθους το πολύ b η κάθε μία. Για 1≤k≤ ar και κάθε μπλοκ, στέλνουμε τα πακέτα στην k-οστή παρτίδα στο μπλοκ να επισκεφτεί όλους τους προορισμούς που υπάρχουν.
    c. Μετακινούμε κάθε πακέτο στην αντίστοιχη θέση στο επόμενο κελί C_{j+1}.

4. Συναυξάνουμε τα r αντίγραφα κάθε πακέτου σε ένα μόνο πακέτο, που περιέχει μόνο τους προορισμούς που δεν έχουν επισκεφθεί ακόμα.

Σημειώνεται πως οι μεταγενέστερες φάσεις αποτελούνται από επαναλήψεις των βημάτων 1-4 εκτός του 3α. Επιλέγουμε ab πακέτα για κάθε μπλοκ αντί για abr.

Έπειτα, αποδεικνύεται η παρακάτω υπόθεση: Μετά από την εκτέλεση του βήματος 4 δημιουργούνται πακέτα τα οποία λέμε πως είναι “alive” για ενα μπλοκ αν περιέχει προορισμούς εντός του μπλοκ αυτού. Ο αριθμός των πακέτων αποδεικνύεται πως είναι το πολύ c/r.

Χρησιμοποιώντας ένα αντίστοιχο επιχείρημα με το παραπάνω λήμμα αποδεικνύεται πως κάθε περαιτέρω φάση εγγυάται, για κάθε μπλοκ, ότι ο συνολικός αριθμός πακέτων στην κλάση με προορισμούς εντός του μπλοκ μειώνεται κατα b. Έτσι, $log_b(c/r) = O(logn/log(n/c)) = O(logn)$ περαιτέρω φάσεις ολοκληρώνουν τη δρομολόγηση της κλάσης.

Στη συνέχεια, περιγράφεται η εφαρμογή του αλγορίθμου. Κατά την πρώτη φάση εφόσον θα δημιουργηθούν συνολικα Θ(n) αντίγραφα από κάθε κλάση, ($r = \Theta(logn)$ αντίγραφα για κάθε ένα από τα $n/logn$ μέλη της κλάσης), τα βήματα 1,2 και 4 μπορούν να εκτελεστούν μέσω δρομολογησης μετάθεσης και τμηματικούς υπολογισμούς προθέματος σε χρόνο $O(\sqrt{n})$ με σταθερό μέγεθος buffer. Με χρήση των ίδιων τεχνικών με το βήμα 2 του τυχαιοποιημένου αλγόριθμου, κάθε εκτελεση του 3a και 3b, απαιτεί χρόνο $O(\sqrt{n/c})$ όσο εξασφαλίζουμε ένα χρόνο εκτέλεσης του $O(r\sqrt{n/c})$ για κάθε εκτέλεση του 3b, πάλι χρησιμοποιώντας χωρητικότητα σταθερού μεγέθους. Έτσι, η πρώτη φάση ολοκληρώνεται σε χρόνο $O(logn\sqrt{cn})$. Σε κάθε επόμενη φάση, η πολυπλοκότητα μιας εκτέλεσης του 3b μειώνεται από $O(r\sqrt{n/c})$ σε $O(\sqrt{n/c})$, άρα η πολυπλοκότητα της φάσης γίνεται $O(\sqrt{nc})$.

Συμπερασματικά, η δρομολόγηση μίας κλασης πραγματοποιείται σε χρονο $O(logn\sqrt{cn})$, και ολόκληρο το σετ μηνυμάτων δρομολογείται σε χρόνο $O(log^2 n\sqrt{cn})$. Από το συμπέρασμα αυτό οι συγγραφείς καταλήγουν στο ακόλουθο θεώρημα:
Ένα αυθαίρετο otm(n,n,d,c) σετ μηνυμάτων με d≤ c μπορεί να δρομολογηθεί ντετερμινιστικά σε χρόνο $O(log^2 n\sqrt{cn})$ με σταθερή χωρητικότητα ανα κόμβο.

6. Αντιμετώπιση άγνωστων τιμών συμφόρησης

Τέλος, παρουσιάζεται η εξάλειψη των αγνώστων τιμών συμφόρησης στον αλγόριθμο που αναλύθηκε παραπάνω. Παρά το γεγονός ότι η επιλογή μία τυχαίας τιμής συμφόρησης είναι μία καλή υπόθεση, στην πραγματικότητα δεν μπορεί να εφαρμοστεί κάτι τέτοιο. Πρέπει, συνεπώς, να αφαιρεθεί η υπόθεση αυτή χωρίς όμως να αυξηθεί ο συνολικός χρόνος εκτέλεσης. Η μέση συμφόρηση καθορίζεται από το ρυθμό παράδοσης μηνυμάτων διαιρεμένο με n. Ο αλγόριθμος τρέχει για γεωμετρικά αυξανόμενες τιμές συμφόρησης ξεκινώντας από το c μέσο. Μια εκτέλεση για μια δεδομένη τιμή c ματαιώνεται όταν είτε ο χρόνος εκτέλεσης υπερβαίνει ένα όριο ανάλογο με τον επιθυμητό χρόνο για την τρέχουσα εικασία, είτε γίνεται υπέρβαση της επιτρεπόμενης χωρητικότητας buffer, ανάλογα με τον αλγόριθμο. Ξεκάθαρα, ο αλγόριθμος τερματίζει όταν μαντευθεί η πραγματική τιμή της συμφόρησης.