Γραμματικές Με Συμφραζόμενα για **NP**-πλήρεις Γλώσσες

Η ΜΕΤΑΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ ΕΞΕΙΔΙΚΕΥΣΗΣ

υποβάλλεται στην

ορισθείσα από την Γενική Συνέλευση Ειδικής Σύνθεσης

του Τμήματος Πληροφορικής Εξεταστική Επιτροπή

από τον

Σταμούλη Γεώργιο

ως μέρος των Υποχρεώσεων για τη λήψη του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΣΤΗΝ ΠΛΗΡΟΦΟΡΙΚΗ

ΜΕ ΕΞΕΙΔΙΚΕΥΣΗ

ΣΤΗΝ ΘΕΩΡΙΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

Φλεβάρης 2009

Περιεχομενα

ΣΧΗΜΑΤΑ

Περιληψη

Γεώργιος Σταμούλης του Γερασίμου και της Αναστασίας. MSc, Τμήμα Πληροφορικής, Πανεπιστήμιο Ιωαννίνων, Φλεβάρης 2009. Γραμματικές με Συμφραζόμενα για **NP**-πλήρης Γλώσσες. Επιβλέποντας: Νομικός Χρίστος.

Μελετάμε τη σχέση μεταξύ χαρακτηριστικών προβλημάτων της κλάσης πολυπλοκότητας **NP** και των Γλωσσών με Συμφραζόμενα, δηλαδή πώς διάφορα **NP**-πλήρη προβλήματα μπορούν να αναπαρασταθούν ως γλώσσες και να περιγραφούν από συντακτικούς κανόνες ευαίσθητους στα συμφραζόμενα.

Βασικό κίνητρο της παραπάνω μελέτης είναι η άγνωστη σχέση μεταξύ των κλάσεων πολυπλοκότητας **NP** και **NSPACE**(n), με την τελευταία να χαρακτηρίζει ακριβώς την κλάση των γλωσσών με συμφραζόμενα. Επιπλέον, μελετάμε την πιθανότητα ύπαρξης υπολογιστικών προβλημάτων της κλάσης πολυπλοκότητας **NP** με υπερ-γραμμικό πιστοποιητικό. Τέτοια προβλήματα δεν μπορούν να περιγραφούν από συντακτικούς κανόνες με συμφραζόμενα.

Extended Abstract in English

Stamoulis Georgios. MSc, Computer Science Department, University of Ioannina, Greece. February 2009. Context Sensitive Grammars for NP-complete languages. Thesis Supervisor: Christos Nomikos.

We study the relationship between **NP**-complete problems and *Context Sensitive Languages* i.e. how well known **NP**-complete problems (such as Partition, Directed Hamiltonian Path, Clique, 3D Matching and Satisfiability of Logical Expressions) can be described as languages and generated by Context Sensitive Grammars.

We take advantage of a nice property of these problems in **NP**, the *linear certificate* property, and we built all the "yes" instances of these problems by describing appropriate sets of Context Sensitive Grammatical rules. This work is motivated by the fact that the exact relationship between **NP** and **NSPACE**(n) is unknown, although we know that they represent different classes of problems/languages. The latter class is precisely the class of Context Sensitive Languages, and also can be viewed as the class of problems that can be verified in linear space when a (guessed) certificate of same size is considered. Also we investigate if it is possible to exist some problems in **NP** with super linear certificates. Such problems are very unlikely to be able to be described by Context Sensitive Grammars.

Μέρος I

Εισαγωγικά

Κεφαλαιο 1

Εισαγωγη

1.1 Γενικά

Στη παρούσα διατριβή μελετάμε τη σχέση που έχουν δύο διαφορετικά θεωρητικά μοντέλα ορισμού γλωσσών. Ως γλώσσα εννούμε ένα σύνολο από συμβολοσειρές, αριθμούς κτλ. Ο πιο τυπικός ορισμός μιας γλώσσας γίνεται μέσω μιας γραμματικής, δηλαδή ένα σύνολο κανόνων το οποίο μας λέει πώς (με ποιόν τρόπο) παράγονται οι λέξεις μιας γλώσσας. Ο εναλλακτικός ορισμός αναφέρεται σε μεθηματικές κατασκευές (αυτόματα) προσομείωσης ενός μοντέλου του υπολογιστή. Με βάση αυτά τα αυτόματα, μπορούμε να ορίσουμε τις γλώσσες ως τις συμβολοσειρές εκείνες τις οποίες μπορεί να αναγνωρίσει και να παράξει ένα συγκεκριμένο αυτόματο.

Με βάση τους δύο αυτούς ορισμούς, είναι δυνατό να ορίσουμε πολλές και διαφορετικές μεταξύ τους νέες γλώσσες. Στην διατριβή αυτή θεωρούμε δύο διαφορετικά μοντέλα από κάθε τέτοια προσέγγιση, των οποίων η ακριβής σχέση δεν είναι ακόμα γνωστή, δηλαδή το εάν παράγουν ή όχι τις ίδιες γλώσσες, και διερευνούμε τις περιπτώσεις που προκύπτουν από την σύγκριση αυτών.

1.2 Στόχοι της Διατριβής

Στόχος της παρούσας διατριβής είναι δώσει μια εναλλακτική προσέγγιση σε διάφορα ευρέως γνωστά και κατά κόρον πρακτικά υπολογιστικά προβήματα από τη ευρέως γνωστή κλάση πολυπλοκότητας **NP**. Πιο συγκεκριμένα, δείχνουμε το πώς μπορούν να δημιουργηθούν όλα τα "καλά" στιγμιότυπα των προβλημάτων αυτών με βάση τον αρχικό ορισμό των γλωσσών. Θεωρούμε τα προβλήματα αυτά ως γλώσσες με τα μέλη της εκάστοτε γλώσσας να αντιστοιχούν σε στιγμιότυπα το τρέχοντος προβλήματος που έχουν τη επιθυμητή ιδιότητα. Προσπαθούμε να διερευνήσουμε το κατά πόσον είναι δυνατόν να δημιουργηθούν αυτά τα στιγμιότυπα με βάση γραμματικές που αντιστοιχούν σε ένα συγκεκριμένο σύνολο γλωσσών - τις γλώσσες *με συμφραζόμενα*.

Ολα αυτά μας δίνουν μια διαίσθηση για την ακριβή σχέση των γλωσσών που ανήκουν στην κλάση **NP** με τις γλώσσες που παράγονται από γραμματικές με συμφραζόμενα. Δείχνουμε ότι τα προβλήματα που επιλέξαμε μπορούν να περιεγραφούν από γραμματικες με συμφραζόμενα, και άρα αυτό, σε πρώτο στάδιο, είναι ένα στοιχείο το οποίο μας επιτρέπει να εικάσουμε ότι *όλα* τα προβλήματα που ανήκου στην κλάση **NP** μπορούν να περιεγραφούν από τέτοιες γραμματικές.

Στο τελευταίο κεφάλαιο όμως, περιγράφουμε ένα τεχνιτό πρόβλημα, το οποίο, αν και δείχνουμε ότι ανήκει στην κλάση **NP**, εντούτοις δεν είναι καθόλου προφανές το πώς μπορεί να περιεγραφεί από μια γραμματική με συμφραζόμενα.

1.3 Δομή της Διατριβής

Η δομή της παρούσης διατριβής έχει ως εξής: Στο πρώτο μέρος της διατριβής ορίζουμε και περιγράφουμε όλα εκείνα τα απαραίτητα αποτελέσματα σχετικά με γραμματικές, κλάσεις πολυπλοκότητας κτλ καθώς επίσης και τι σχέση έχουν μεταξύ τους ώστε να φανεί πλέον ξεκάθαρα το κίνητρο πίσω από την προσέγγισή μας.

Στο δεύτερο μέρος, κάθε κεφάλαιο περιλαμβάνει ένα σύνολο κανόνων που παράγουν όλα τα "ναι" στιγμιότυπα για το εκάστοτε υπολογιστικό πρόβλημα, χρησιμοποιώντας κανόνες με συμφραζόμενα.

Τέλος, το τελευταίο κεφάλαιο περιέχει την περιγραφή ενός υποψήφιου προβλήματος το οποίο φαίνεται να διαχωρίζει τις δύο υποκείμενες κλάσεις μελέτης: την **NP** με την κλάση των γλωσσών που παράγωνται από γραμματικές με συμφραζόμενα.

Κεφαλαιο 2

Τυπικεσ Γλωσσεσ και Γραμματικεσ

2.1 Τυπικές Γλώσσες

Μια τυπική γλώσσα είναι γενικά ένα σύνολο από *λέξεις* οι οποίες αντλούν τα σύμβολά τους από ένα δεύτερο, για παράδειγμα ένα πεπερασμένο σύνολο συμβόλων, γραμμάτων ή αριθμών. Το σύνολο αυτό από το οποίο μπορούμε να "αντλήσουμε" αυτά τα γράμματα (σύμβολα) ονομάζεται το *αλφάβητο* της υποκείμενης γλώσσας μέσω του οποίου ορίζεται η γλώσσα. Μια τυπική γλώσσα συνήθως ορίζεται μέσω μιας *τυπικής γραμματικής*. Ο όρος Τυπική Γλώσσα είναι ένας καθαρά συντακτικός όρος, για το λόγο αυτό δεν υπάρχει απαραίτητα κάποια εξήγηση η οποία και τον ερμηνεύει. Για να ξεχωρίσουμε τις λέξεις - συμβολοσειρές οι οποίες ανήκουν στη γλώσσα από άλλες συμβολοσειρές οι οποίες απλά αποτελούνται από σύμβολα του αλφαβήτου της γλώσσας αυτής, οι αρχικές συχνά ονομάζονται *καλώς ορισμένες συμβολοσειρές* (ή, στη μαθηματική λογική, *καλώς ορισμένες προτάσεις*).

Οι τυπικές γλώσσες μελετώνται στο πεδίο της μαθηματικής λογικές, στην πληροφορική καθώς και στη γλωσσολογία. Η κύρια και πιο σημαντική πρακτική εφαρμογή των τυπικών γλωσσών στην επιστήμη της πληροφορικής είναι

για τον ακριβή ορισμό συντακτικά ορθών προγραμμάτων για κάποια γλώσσα προγραμματισμού. Ο κλάδος των μαθηματικών και της επιστήμης των υπολογιστών ο οποίος ασχολείται μόνο με αμιγώς συντακτικές ιδιότητες τέτοιων γλωσσών προγραμματισμού, όπως το πρότυπο εσωτερικ ̃ ση δομής μιας τέτοιας γλώσσας, είναι γνωστός ως Τυπική Θεωρία Γλωσσών.

Οι "λέξεις" μιας τυπικής γλώσσας έχουν συχνά μια συγκεκριμένη σημασιολογική ερμηνεία, αν και αυτή η ερμηνεία δεν είναι τυπικά μέρος της γλώσσας, Στην πράξη, αυτό είναι πάντοτε συνυφασμένο με τη δομή της γλώσσας αυτής, και οι τυπικές γραμματικές μπορούν να μας βοηθήσουν να αντιμετωπίσουμε το νόημα καλά ορισμένων λέξεων (με τον όρο τυπική γραμματική εννοούμε ένα σύνολο κανόνων οι οποίοι αναδρομικά ορίζουν μια γλώσσα). Ενα πολύ γνωστό παράδειγμα είναι ο ορισμός της αλήθειας κατά τον Tarski στην πρωτοβάθμια λογική, ενώ στο πεδίο της επιστήμης των υπολογιστών οι γεννήτορες μεταφραστών lex και yacc.

2.1.1 Λέξεις από ένα αλφάβητο

Ενα αλφάβητο, στο πλαίσιο των τυπικών γλωσσών, μπορεί να είναι ένα οποιοδήποτε σύνολο συμβόλων, αν και τις περισότερες φορές το αλφάβητο αυτό ορίζεται με τη συνήθη έννοια της λέξης, δηλαδή αποτελεί ένα σύνολο χαρακτήρων. Τα αλφάβητα, γενικά, μπορούν να είναι άπειρα: ένα τέτοιο παράδειγμα άπειρου αλφαβήτου είναι η το αλφάβητο που χρησιμοποιείται στη πρωτοβάθμια λογική η οποία συχνά περιγράφεται χρησιμοποιώντας ένα αλφάβητο στο οποίο, εκτός των συμβόλων $\vee, \wedge, \neg$ παρενθέσεις κτλ, περιέχει άπειρα (αλλά αριθμήσιμα) στοιχεία $x_0, x_1, \ldots$ τα οποία αντιπροσωπεύουν τις μεταβλητές. Τα στοιχεία ενός αλφαβήτου μερικές φορές ονομάζονται και "γράμματα".

Μια λέξη από ένα αλφάβητο μπορεί να είναι μια οποιαδήποτε άπειρη ακολουθία, ή συμβολοσειρά, γραμμάτων. Το σύνολο όλων αυτών των λέξεων - συμβολοσειρών από ένα αλφάβητο Σ, συνήθως συμβολίζεται ως Σ^* (το οποίο και αναφέρεται ως Kleene κλειστότητα). Για ένα αλφάβητο, ύπάρχει μόνο μία συμβολοσειρά μήκους 0, η κενή συμβολοσειρά, την οποία και θα συμβολίζουμε ως ϵ ή ε (μερικές φορές υιοθετήται ο όρος λ). Στη συνέχεια της διατριβής, θα συμβολίζουμε τη συμβολοσειρά αυτή με τον τυπικό τρόπο - ως ϵ.

2.1.2 Ορισμός Τυπικών Γλωσσών

Παρακάτω δίνουμε λεναν ορισμό των τυπικών γλωσσών:

Ορισμός 1 *Μια τυπική γλώσσα L σε κάποιο αλφάβητο Σ είναι ένα υποσύνολο του συνόλου Σ^*.*

Γενικά, στην επιστήμη των υπολογιστών και στον κλάδο των μαθηματικών στις οποίες η έρευνα δεν ασχολείται με φυσικές γλώσσες, ο όρος "τυπική" όταν ανφερόμαστε σε μιά γλώσσα, παραλείπεται.

Ενώ η θεωρία τυπικών γλωσσών συνήθως (αν όχι πάντα) ασχολείται με τυπικές γλώσσες οι οποίες περιγράφονται από ένα σύνολο συντακτικών κανόνων, ο πραγματικός ορισμός του όρου "τυπική γλώσσα" δεν είναι τίποτα περισσότερο ή τίποτα λιγότερο από τον ανωτέρω ορισμό: ένα σύνολο (ενδεχομένως άπειρο) από συμβολοσειρές (συνήθως, αν και όχι απαραίτητα, πεπερασμένου μήκους) από το υποκείμενο αλφάβητο. Στη πράξη, υπάρχουν πολλές γλώσσες οι οποίες μπορούν να περιγραφούν από ένα σύνολο συντακτικών κανόνων, για παράδειγμα το σύνολο των κανονικών γλωσσών ή των γλωσσών με συμφραζόμενα. Ο όρος μιας τυπικής γλώσσας μπορεί να βρίσκεται κοντύτερα στον ορισμό μιας "γλώσσας" που περιγράφεται από ένα σύνολο συνακτικών κανόνων. Με άλλα λόγια, μια δεδομένη τυπική γλώσσα μπορεί να θεωρηθεί ότι ορίζεται μέσω μιας τυπικής γραμματικής που περιγράφει την ιδιότητα των αντίστοιχων συμβολοσειρών που ανήκουν στη γλώσσα αυτή.

2.1.3 Παραδείγματα Τυπικών Γλωσσών

Το επόμενο σύνολο κανόνων περιγράφει μια τυπική γλώσσα L η οποία αντλεί τα σύμβολα των λέξεών της από εάν αλφάβητο $\Sigma = \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, +, =\}$:

- Κάθε μη κενή συμβολοσειρά η οποία δεν περιέχει τα σύμβολα $+$ ή $=$ και δεν αρχίζει με 0, ανήκει στη γλώσσα L.
- Η συμβολοσειρά 0 ανήκει στη γλώσσα L.
- Μια συμβολοσειρά η οποία περιέχει το σύμβολο $=$ ανήκει στη γλώσσα L εάν, και μόνον εάν, υπάρχει ακριβώς ένα σύμβολο $=$, και αυτό διαχωρίζει δύο έγκυρες συμβολοσειρές της γλώσσας L.
- Μια συμβολοσειρά η οποία περιέχει το σύμβολο $+$ ανήκει στη γλώσσα L εάν, και μόνον εάν, κάθε σύμβολο $+$ διαχωρίζει δύο έγκυρες συμβολοσειρές της γλώσσας L.
- Καμία άλλη συμβολοσειρά δεν ανήκει στη γλώσσα L πέραν αυτών που περιγράφηκαν παραπάνω.

Με βάση αυτούς τους κανόνες, η συμβολοσειρά "$123+456 = 0$" ανήκει στη γλώσσα μας, αλλά για παράδειγμα η συμβολοσειρά $+ = 222 = 333$ δεν ανήκει. Η παραπάνω τυπική γλώσσα περιγράφει φυσικούς αριθμούς, καλά ορισμένες

προτάσεις πρόσθεσης φυσικών αριθμών και, τέλος, καλά ορισμένες προτάσεις ισότητας φυσικών αριθμών. Από την άλλη μεριά, η παραπάνω γλώσσα εκφράζει το "πώς πρέπει να φαίνονται" αυτές οι προτάσεις (δηλαδή ορίζει το συντακτικό αυτών), αλλά όμως όχι το τι εκφράζουν (δηλαδή τη σημασιολογία αυτών). Για παράδειγμα, δεν προκύπτει πουθενά από τους παραπάνω κανόνες το τι ακριβώς σημαίνει το σύμβολο "0" ή ότι το σύμβολο "+" αντιστοιχεί στη λειτουργία της πρόσθετης όπως την ξέρουμε.

Οταν ασχολούμαστε με πεπερασμένες γλώσσες, δηλαδή γλώσσες οι οποίες περιέχουν έναν πεπερασμένο αριθμό συμβολοσειρών, τότε απλά μπορούμε να απαριθμήσουμε όλες τις δυνατές καλά ορισμένες συμβολοσειρές που ανήκου στη γλώσσα αυτή. Για παράδειγμα, μια σε μια τέτοια περίπτωση ανήκει η γλώσσα $L = \{b, aba, aabbaa, aaabbbaaa\}$ η οποία περιέχει μόνο αυτές τις τέσσερις συμβολοσειρές.

Από την άλλη πλευρά, ακόμα και εάν έχουμε ένα πεπερασμένο αλφάβητο, αυτό είναι ικανό να ορίζει μια γλώσσα με άπειρο αριθμό συμβολοσειρών, όπως για παράδειγμα η επόμενη γλώσσα L που ορίζεται στο αλφάβητο $\Sigma = \{\alpha, \beta\}$ και περιέχει έναν άπειρο αριθμό συμβολοσειρών: $\alpha\beta, \alpha\alpha\beta\beta, \alpha\alpha\alpha\beta\beta\beta, \ldots$. Κατά κανόνα οι τυπικές γλώσσες είναι άπειρες και μια απλή απαρίθμηση δεν είναι ικανή να μας δώσει τις συμβολοσειρες που ανήκουν στην υποκείμενη άπειρη γλώσσα, για το λόγω αυτό παρακάτω θα ορίσουμε ένα σύνολο κανόνων οι οποίοι θα περιγράφουν δομικές ιδιότητες των συμβολοσειρών που ανήκουν σε μια δεδομένη γλώσσα καθώς και τον τρόπο μου μπορούμε να κατασκευάσουμε τέτοιες συμβολοσειρες.

Μερικά παραδείγματα τέτοιων γλωσσών είναι τα παρακάτω:

- $L = \Sigma^*$, δηλαδή το σύνολο όλων των συμβολοσειρών που μπορούν να σχηματιστούν με τα σύμβολα του αλφαβήτου Σ.
- $L = \alpha^* = \alpha^n$, όπου η μεταβλητή n ανήκει στους φυσικούς αριθμούς και η έκφραση α^n σημαίνει επανάληψη του συμβόλου α n φορές (η γλώσσα αυτή αποτελείται από το σύνολο των συμβολοσειρών που περιέχουν μόνο ένα σύμβολο - το α).
- Το σύνολο όλων των συντακτικά ορθών προγραμμάτων μιας δεδομένης γλώσσας προγραμματισμού (το συντακτικό της οποίας σχεδόν πάντα ορίζεται από μια γραμματική με συμφραζόμενα).
- Το σύνολο όλων των πιθανών εισόδων με τις οποίες μια δεδομένη μηχανή Turing τερματίζει σε μια κατάσταση τερματισμού.

2.1.4 Λειτουργίες σε Γλώσσες και Συμβολοσειρές

Διάφορες λειτουργίες στις γλώσσες χρησιμοποιούνται πολύ συχνά. Αυτές περιέχουν τις κλασσικές λειτουργίες πάνω σε σύνολα, όπως τομή, ένωση και συμπλήρωμα, καθώς και διάφορες άλλες λειτουργίες οι οποίες ενεργούν πάνω σε συμβολοσειρές οι οποίες ανήκουν σε μια δεδομένη γλώσσα. Για παράδειγμα, θεωρούμε ότι έχουμε δύο γλώσσες L_1 και L_2 με το ίδιο αλφάβητο Σ. Τότε:

- ▷ Η παράθεση $L_1 \circ L_2$ ων δύο γλωσσών αποτελείται από όλες τις συμβολοσειρές οι οποίες μπορούν να γραφτούν στην μορφή $uv : \ u \in L_1, \ u \in L_2$.
- ▷ Η τομή των δύο γλωσσών, που συμβολίζεται ως $L_1 \cap L_2$ αποτελείται από όλες εκείνες τις συμβολοσειρές οι οποίες ανήκουν και στις δύο γλώσσες. Δηλαδή $w \in L_1 \cap L_2 \Leftrightarrow w \in L_1 \wedge w \in L_2$.
- ▷ Το συμπλήρωμα μιας γλώσσας L, το οποίο και το συμβολίζουμε ως $\neg L$ ως $\bar{L}$, ίναι μια νέα γλώσσα με το ίδιο αλφάβητο και περιέχει όλες εκείνες τις συμβολοσειρές του συγκεκριμένου κοινού αλφαβήτου που *δεν* ανήκουν στην αρχική γλώσσα L.
- ▷ Kleene κλειστότητα: η γλώσσα αυτή αποτελείται από όλες εκείνες τις συμβολοσειρές οι οποίες είναι παραθέσεις μηδέν ή περισσότερων συμβολοσειρών που ανήκουν στην αρχική γλώσσα. Δηλαδή, η Kleene κλειστότητα μιας γλώσσας L, η οποία συμβολίζεται ως L^*, περιέχει όλες εκείνες τις συμβολοσειρές w οι οποίες μπορούν να γραφτούν στην εξής μορφή: $L^* = \{w \in \Sigma : w = w_1 w_2 \dots w_n\}, w_i \in L, \ 1 \leq i \leq n$ για κάποιο $n \geq 0$.
- ▷ Η αντίστροφη γλώσσα L^r μιας γλώσσας L περιέχει όλες τις συμβολσοειρές της αρχικής γλώσσας στην αντίστροφη μορφή τους. Η αντίστροφη μιας συμβολοσειράς ορίζεται ως εξής:

  1. Για τη κενή συμβολοσειρά ϵ ισχύει ότι $\epsilon^r = \epsilon$.
  2. Για κάθε μη κενή συμβολοσειρά $w = w_1 w_2 \dots w_n$, η αντίστροφή της είναι $w^r = w_n w_{n-1} \dots w_1$.

  Αρα η γλώσσα L^r ορίζεται ως εξής: $L^r = \{w^r \mid w \in L\}$.

Αυτές οι λειτουργίες πάνω σε γλώσσες, και κατ'επέκταση σε συμβολειρές, χρησιμοποιούνται για να διερευνηθούν ιδιότητες κλειστότητας διαφόρων κλάσεων γλωσσών. Μια κλάση γλωσσών είναι κλειστή ως προς μια δεδομένη λειτουργία όταν η λειτουργία αυτή, εφαρμοζόμενη σε γλώσσες της κλάσης

αυτής, πάντα παράγει μια γλώσσα που ανήκει στην ίδια κλάση με την αρχική στην οποία εφαρμόστηκε η λειτουργία. Για παράδειγμα μπορεί να αποδεχτεί ότι οι γώσσες χωρίς συμφραζόμενα είναι κλειστές, για παράδειγμα, ως προς την ένωση και την παράθεση αλλα δεν είναι κλειστές ως προς την τομή ή το συμπλήρωμα (βλέπε Hopcroft, Ullman).

2.2 Γραμματικές

Η τυπική θεωρία γλωσσών είναι ένας κλάδος των μαθηματικών ο οποίος χρησιμοποιείται κύρια στην επιστήμη των υπολογιστών και στην γλωσσολογία με σκοπό να ορίσει τους κανόνες εκείνους με τους οποίους μια συγκεκριμένη (φυσική η τεχνιτή) γλώσσα μπορεί να παραχθεί. Δηλαδή, μια γραμματική περιγράφει τον τρόπο με τον οποίο συμβολοσειρές που ανήκουν σε μια γλώσσα μπορούν να παραχθούν. Με άλλα λόγια, μια γραμματική περιγράφει *τον τρόπο* με τον οποίο σχηματίζονται έγκυρες ακολουθίες συμβόλων (τα στοιχεία του αλφαβήτου) οι οποίες ανήκουν στην γλώσσα και οι οποίες αντιπροσωπεύουν έγκυρες λέξεις ή δηλώσεις στην υποκείμενη γλώσσα, όμως μια γραμματική δεν περιγράφη την *σημασιολογία* του τί αυτές οι λέξεις περιγράφουν.

Μια γραμματική συχνά αντιμετωπίζεται ώς ένας τρόπος με τον οποίο παράγονται όλες οι έγκυρες συμβολοσειρές μιας γλώσσας. Επίσης, με βάση μια γραμματική για μια συγκεκριμένη γλώσσα, μπορούμε να κατασκευάσουμε έναν αλγόριθμο ο οποίος λειτουργεί ως "αναγνωριστής" της γλώσσας αυτής, δηλαδή έναν αλγόριθμο ο οποίος, δοθείσας μιας συμβολοσειράς, ελέγχει έαν αυτή είναι μέλος της υποκείμενης γλώσσας. Για να περιγραφούν τέτοιοι αναγνωριστές γλωσσών, στην τυπική θεωρία γλωσσών χρησιμοποιούνται άλλες τυπικές κατασκευές γνωστές και ως *αυτόματα*.

Επιπλέον, μια γραμματική μπορεί να χρησιμοποιηθεί για την *ανάλυση* των συμβολοσειρών μιας γλώσσας, δηλαδή για μια δοθείσα συμβολοσειρά μπορεί να περιγράψει την εσωτερική της δομή, τον τρόπο με τον οποίο "κατασκευάστηκε" αυτή η συμβολοσειρά. Στην επιστήμη των υπολογιστών, αυτή η διαδικασία ονομάζεται *συντακτική ανάλυση* και χρησιμοποιείται κατά κόρον στους μεταφραστές γλωσσών προγραμματισμού. Οι περισσότερες (εάν όχι όλες) οι σύγχρονεςς γλώσσες προγραμματισμού έχουν πολύ στοιχειοθετημένη σημασιολογία, δηλαδή το νόημα των διαφόρων διατυπώσεων είναι δομημένο κατάλληλα στο εκάστοτε συντακτικό. Αρα, το πρώτο βήμα για να περιγραφεί η σημασία των διαφόρων δηλώσεων και διατυπώσεων σε μια γλώσσα προγραμματισμού είναι να αναλυθεί και ύστερα να επεξεργαστεί ως προς αυτή την αναλυτική μορφή, η οποία είναι γνωστή και ως συντακτικό δέντρο στην θεωρία γλωσσών και μεταφραστών στην επιστήμη των υπολογιστών.

2.3 Τυπικές γραμματικές

Μια γραμματική αποτελείται από ένα σύνολο *κανόνων μετασχηματισμού* των συμβολοσειρών που ονομάζονται και *παραγωγές*. Κάθε τέτοιος κανόνας σε μιά γραμματική αποτελείται από ένα σύμβολο που χωρίζεται από μια συμβολοσειρά από τον χαρακτήρα $\rightarrow$. Παραδείγματα κανόνων τέτοιας μορφής είναι, για παράδειγμα, οι $A \rightarrow AB$, $B \rightarrow \Gamma\gamma$ και $\Gamma \rightarrow \delta$. Τα σύμβολα που εμφανίζονται στο αριστερό μέλος ενός κανόνα ονομάζονται *μεταβλητές ή μη τερματικά σύμβολα*. Αυτά είναι τα σύμβολα που μπορούν να αντικατασταθού από διάφορες συμβολοσειρές με βάση πάντα τη γραμματική. Η συμβολοσειρά του δεξιού μέλους ενός κανόνα αποτελείται από μεταβλητές και *τερματικά σύμβολα*. Οι μεταβλητές συνήθως αναπαρίστανται με κεφαλαία γράμματα ενώ τα τερματικά σύμβολα είναι στην ουσία οι χαρακτήρες που αποτελούν το αλφάβητο της υποκείμενης γλώσσας και συχνά αναπαρίστανται με μικρά γράμματα, σύμβολα ή αριθμούς, εκτός και εάν αναφέρεται αλλιώς. τα τερματικά σύμβολα δεν μπορούν (από μόνα τους) να αντικατασταθούν από οποιοδήποτε άλλο σύμβολο. Μια συγκεκριμένη μεταβλητή ονομάζεται *αρχική μεταβλητή*. Συνήθως εμφανίζεται στο αριστερό μέλος του πρώτου κανόνα της γραμματικής. Το σύμβολο αυτό συχνά αναπαρίσταται με το γράμμα S.

Για να παραχθεί μια συμβολοσειρά της υποκείμενης γλώσσας, η γραμματική ξεκινά πάντα από τον αρχικό κανόνα ο οπόιος περιέχει το αρχικό σύμβολο, και έπειτα επαναληπτικά εφαρμόζει τους κανόνες της γραμματικής (οποίοδήποτε αριθμό φορών, με οποιαδήποτε σειρά εάν πρόκειται για διαφορετικούς κανόνες) για να επανεγγράψει τη συμβολοσειρά, με την νέα που προκύπτει μετά από κάθε εφαρμογή ενός κανόνα. Ως εφαρμογή ενός κανόνα σε μια συμβολοσειρά εννούμε την αντικατάσταση κάποιας μεταβλητής της τρέχουσας συμβολοσειράς μας από έναν κανόνα ο οποίος περιέχει στο αριστερό του μέλος τη συγκεκριμένη μεταβλητή.

Για παράδειγμα, εάν θεωρήσουμε ότι έχουμε μια γραμματική η οποία περιέχει τους παραπάνω τρεις κανόνες και η αρχική μεταβλητή είναι η A τότε, αρχικά, ξεκινώντας από την A παράγουμε τη συμβολοσειρά AB εφαρμόζοντας τον πρώτο κανόνα. Ο κανόνας αυτός εννοεί "η μεταβλητή A μπορεί να αντικατασταθεί με τη συμβολοσειρά AB". Επειτα, από τη συμβολοσειρά μας AB έχουμε δύο επιλογές: είτε να αντικαταστήσουμε το A όπως και στο πρώτο βήμα, είτε να αντικαταστήσουμε τη μεταβλητή B με τη συμβολοσειρά $\Gamma\gamma$. Η διαδικασία αυτή επεναλαμβάνεται οποιονδήποτε αριθμό φορών, δηλαδή σε κάθε βήμα επιλέγουμε μια μεταβλητή της τρέχουσας συμβολοσειράς μας και την αντικαθιστούμε με τη συμβολοσειρά του δεξιού μέλους του κανόνα εκείνου που περιέχει στο αριστερό μέλος τη μεταβλητή που μόλις επιλέξαμε. Εάν έχουμε μια γραμματική η οποία δίνει περισσότερες από μία διαφορετικές επιλογές αντικατάστασης για την ίδια μεταβλητή, τότε μπορούμε να συγχωνεύ-

σουμε όλες τις δυνατές επιλογές σε έναν κανόνα χωρίζοντας τις διαφορετικές επιλεογές με τον χαρακτήρα "|". Δηλαδή εάν έχουμε δύο κανόνες $B \to C$ και $B \to BD$ τότε μπορούμε να γράψουμε έναν κανόνα $B \to C \mid BD$ που σημαίνει ότι όταν αποφασίσουμε να αντικαταστήσουμε κάποια μεταβλητή B της τρέχουσας συμβολοσειράς μας, τότε μπορούμε να την αντικαταστήσουμε είτε με C είτε με BD.

Η γλώσσα αποτελείται από όλες τις συμβολοσειρές οι οποίες μπορούν να παραχθούν με τον συγκεκριμένο τρόπο. Κάθε μια συγκεκριμένη αλληλουχία έγγυρων επιλογών που εφαρμόζονται κατά τη διάρκεια της διαδικασίας μετασχηματισμού της συμβολοσειράς, μέχρι αυτή να φτάσει στην τελική-επιθυμητή μορφή της, παράγει και μία συγκεκριμένη συμβολοσειρά που ανήκει στη γλώσσα. Εάν υπάρχουν πολλοί και διαφορετικοί τρόποι παραγωγής μιας δεδομένης συμβολοσειράς με την παραπάνω διαδικασία, τότε η γραμματική ονομάζεται *διφορούμενη*. Προφανώς. μια γραμματική παράγει συμβολοσειρές από μια συγκεκριμένη γλώσσα με την έννοια ότι μια γραμματική δεν μπορεί να χρησιμοποιηθεί για την περιγραφή δύο διαφορετικών μεταξύ τους γλωσσών ταυτόχρονα.

Για παράδειγμα έστω ότι θέλουμε να δημιουργήσουμε μια γραμματική που να παράγει την γλώσσα που αποτελείται από όλες εκείνες τις λέξεις με γράμματα α και β, οι οποίες έχουν έναν αριθμό από συνεχόμενα α (πιθανόν 0) ακολουθούμενα από *τον ίδιο* αριθμό από β. Εστω ότι το αρχικό σύμβολο είναι το S. Τότε η γραμματική που ψάχνουμε είναι η εξής:

$$S \to \alpha S \beta$$
$$S \to \varepsilon$$

Στην παραπάνω περίπτωση αρχίζουμε με το αρχικό σύμβολο S και επιλέγουμε για εφαρμογή κάποιον από τους δύο κανόνες. Εάν επιλέξουμε τον πρώτο κανόνα, τότε αντικαθιστούμε το S με το $\alpha S \beta$. Επιλέγοντας τον δεύτερο κανόνα, αντικαθιστούμε το S με ε, δηλαδή την κενή συμβολοσειρά και τελειώνουμε: δεν έχει μείνει τίποτα να αντικαταστήσουμε. Στην περίπτωση που έχουμε επιλέξει τον πρώτο κανόνα, μπορούμε να επαναλάβουμε την παραπάνω επιλεγή οποιονδήποτε αριθμό φορών θέλουμε μέχρι να κλαταλήξουμε σε μια συμβολοσειρά από α και β στην οποία να μην μπορούμε να κάνουμε πλέον αλλαγές. Εάν δηλαδή επιλέξουμε την δεύτερη φορά ξανά τον πρώτο κανόνα αντικατάστασης $S \to \alpha S \beta$ και την τρίτη φορα τον κανόνα $s \to \varepsilon$ τότε καταλήγουμε στην συμβολοσειρά $\alpha\alpha\beta\beta$ στην οποία δεν μπορούμε να συνεχίσουμε τις αλλαγές. Αυτή την αλληλουχία επιλογών μπορούμε να την συμβολίσουμε πιο περιληπτικά ως εξής: $S \Rightarrow \alpha S \beta \Rightarrow \alpha\alpha S \beta\beta \Rightarrow \alpha\alpha\beta\beta$. Η

γλώσσα αυτής της γραμματικής είναι περιέχει της λέξεις-συμβολοσειρές του συνόλου $\{w \in \{\alpha, \beta\}^* | w = \alpha^n \beta^n,\ n \geq 0\}$.

Ο πρώτος τυπικός ορισμός που δόθηκε στις γραμματικές (δηλαδή τι είναι, τι περιγράφουν, πώς συμβολίζονται) δόθηκε στην δεκαετία του 50' από τον γλωσσολόγο, φιλόσοφο, πολιτικό ακτιβιστή και συγγραφέα Εβραϊκής καταγωγής Αμερικανό Αβραάμ Νοάμ Τσόμσκι.

Ορισμός 2 *Σύμφωνα με τον ορισμό που έδωσε ο Τσόμσκι, μια γραμματική G αποτελείται από τις εξής συνιστώσες:*

- ▷ *Από ένα μη κενό σύνολο N των μη τερματικών συμβόλων ή μεταβλητών.*
- ▷ *Ενα μη κενό σύνολο Σ των τερματικών συμβόλων ξένο ως προς το N.*
- ▷ *Ενα σύνολο κανόνων παραγωγής P της εξής μορφής:*

$$(\Sigma \cup N)^* N (\Sigma \cup N)^* \to (\Sigma \cup N)^*$$

- ▷ *Ενα συγκεκριμένο σύμβολο S - το αρχικό σύμβολο.*

Με βάση τα παραπάνω συστατικά, μια γραμματική ορίζεται ως μια διατεταγμένη τετράδα $G = (N, \Sigma, P, S)$ αυτών.

Δηλαδή, κάθε κανόνας παραγωγής απεικονίζει μια συμβολοσειρά σε μιά άλλη, όπου η αρχική συμβολοσειρά περιέχει τουλάχιστον ένα μη τερματικό σύμβολο (μεταβλητή). Στη περίπτωση που η δεύτερη συμβολοσειρά είναι η κενή, δηλαδή η συμβολοσειρά εκείνη που δεν περιέχει καθόλου σύμβολα, για να αποφεύγονται οι παρανοήσεις, αυτή η κενή συμβολοσειρά θα συμβολίζεται στους κανόνες με το γνωστό σύμβολο ϵ.

Η λειτουργία μιας γραμματικής μπορεί να οριστεί με βάση τις παρακάτω σχέσεις στις συμβολοσειρές

Ορισμός 3 *Εστω μια γραμματική $G = (N, \Sigma, P, S)$. τότε*

- *η δυαδική σχέση $\Rightarrow_G$ σε συμβολοσειρές από το σύνολο $(\Sigma \cup N)^*$ ορίζεται ως εξής:*

$$x \Rightarrow_G y \text{ εάνν } \exists u, v, w \in \Sigma^*, X \in N :$$

$$x = uXv \wedge y = uwv \wedge X \to w \in P$$

- *Η σχέση $\Rightarrow_G{}^*$ ορίζεται ως η μεταβατική κλειστότητα του συνόλου $(\Sigma \cup N)^*$.*

- *Η γλώσσα της γραμματικής G, οποία συμβολίζεται ως $L(G)$, ορίζεται ως όλες εκείνες οι συμβολοσειρές από το αλφάβητο Σ οι οποίες μπορούν να παραχθούν ξεκινόντας από το αρχικό σύμβολο S της γραμματικής μας και έπειτα εφαρμόζοντας τους κανόνες παραγωγής στο P μέχρι να καταλήξουμε σε μια συμβολοσειρά η οποία να μην περιέχει καθόλου μη τερματικά σύμβολα. Πιο τυπικά $L(G) = \{w \in \Sigma^* \mid S \Rightarrow_G^* w\}$.*

Παράδειγμα 1 *Εστω η γραμματική G όπου $N = \{S, B\}, \Sigma = \{a, b, c\}$, S είναι το αρχικό σύμβολο και το σύνολο P αποτελείται από τους παρακάτω κανόνες:*

1. $S \rightarrow aBSc$
2. $S \rightarrow abc$
3. $Ba \rightarrow aB$
4. $Bb \rightarrow bb$

Μερικά παραδείγματα παραγωγών συμβολοσειρών είναι τα εξής:

α. $S \Rightarrow \mathbf{abc}$

β. $\mathbf{S} \Rightarrow \mathbf{aBSc} \Rightarrow aB\mathbf{abc}c \Rightarrow a\mathbf{aB}bcc \Rightarrow aa\mathbf{bb}cc$

γ. $\mathbf{S} \Rightarrow \mathbf{aBSc} \Rightarrow aB\mathbf{aBSc}c \Rightarrow aBaB\ \mathbf{abc}\ cc \Rightarrow a\mathbf{aB}Babccc \Rightarrow aaB\ \mathbf{aB}\ bccc \Rightarrow aa\mathbf{aB}Bbccc \Rightarrow aaaB\mathbf{bb}ccc \Rightarrow aaa\mathbf{bb}bccc$

Η παραπάνω γραμματική ορίζει τη γλώσσα $L = \{a^n b^n c^n | n \geq 1\}$ όπου a^n είναι μια συμβολοσειρά με n συνεχόμενα a. Αρα, η γλώσσα αυτή είναι το σύνολο των συμβολοσειρών το οποίο αποτελείται από ένα ή περισσότερους χαρακτήρες a, ακολουθούμενους από τον ίδιο αριθμό χαρακτήρων b, ακολουθούμενους από τον ίδιο αριθμό χαρακτήρων c.

2.4 Η Ιεραρχία Τσόμσκι

Οταν ο Νόαμ Τσόμσκι αρχικά όρισε τις τυπικές γραμματικές το 1956, τις κατέταξε σε ένα σύνολο τύπων (κλάσεων) οι οποίες πλέον είναι γνωστές ως η *Ιεραρχία Τσόμσκι*. Η διαφορά ανάμεσα σε αυτές τις κλάσεις είναι οτι έχουν αυξανόμενη αυστηρότητα όσον αφορά τον ορισμό των κανόνων παραγωγής και μπορούν να περιγράψουν λιγότερες τυπικές γλώσσες.

Η ιεραρχία αυή είναι η ακόλουθη:

Γραμματικές τύπου 0 ή μη περιορισμένες γραμματικές: αυτές περιέχουν όλες τις τυπικές γραμματικές. Παράγουν όλες τις γλώσσες οι οποίες μπορούν να αναγνωριστούν από μια μηχανή Turing.Οι γλώσσες αυτές είναι επισης γνωστές ως *αναδρομικώς απαριθμήσιμες* γλώσσες. Αυτές είναι διαφορετικές από τις αναδροικές γλώσσες οι οποίες αποφασίζονται από μηχανές Turing οι οποίες πάντοτε τερματίζουν τη λειτουργία τους.

Γραμματικές τύπου 1 ή γραμματικές με συμφραζόμενα: οι γραμματικές αυτές παράγουν τις γλώσσες με συμφραζόμενα. Εχουν κανόνες τις μορφής $\alpha A\beta \rightarrow \alpha\gamma\beta$ όπου A ένα μη τερματικό σύμβολο της γραμματικής ενώ α, β, γ είναι συμβολοσειρές τερματικών και μη τερματικών συμβόλων με τις συμβολοσειρές α και β να μπορεί να είναι κενές αλλά η συμβολοσειρά γ είναι απαραίτητα μη κενή. Ο κανόνας $S \rightarrow \epsilon$ επιτρέπεται εάν η αρχική μεταβλητή S δεν εμφανίζεται στο δεξί μέρος κάνενός κανόνα του συνόλου κανόνων P. Οι γλώσσες που περιγράφονται από αυτές τις γραμματικές είναι ακριβώς οι ίδιες γλώσσες που αναγνωρίζονται από ένα γραμμικώς φραγμένο αυτόματο, δηλαδή από μια μη ντετερμινιστική μηχανή Turing της οποίας η ταινία φράσσεται από το μέγεθος της συμβολοσειράς εισόδου. Θα αναλύσουμε τις γραμματικές με συμφραζόμενα καθώς και τις υποκείμενες γλώσσες που αυτές παράγουν μαζί με πολλές ενδιαφέρουσες ιδιότητες αυτών με μεγάλη λεπτομέρεια σε επόμενο αντίστοιχο κεφάλαιο.

Γραμματικές τύπου 2 ή γραμματικές χωρίς συμφραζόμενα: οι γραμματικές αυτές παράγουν τις γλώσσες χωρίς συμφραζόμενα. Οι κανόνες που περιέχουν οι γραμματικές αυτές είναι της μορφής $A \rightarrow \gamma$ όπου A ένα μη τερματικό σύμβολο ενώ γ μια συμβολοσειρά τερματικών και μη τερματικών συμβόλων, ενδεχομένως κενή. Αυτές οι γλώσσες είναι ακριβώς οι γλώσσες που μπορούν να αναγνωριστούν από ένα μη ντετερμινιστικό αυτόματο εφοδιασμένο με μιά στοίβα. Οι γλώσσες χωρίς συμφραζόμενα είναι το θεωρητικό υπόβαθρο για το συντατκικό σχεδόν όλων των σύγχρονων γλωσσών προγραμματισμού κυριώς επειδή υπάρχουν πολύ γνωστοί και σχετικά γρήγοροι αλγόριθμοι για τη συνακτική ανάλυση ενός προγράμματος γραμμένου με κανόνες γραμμματικών χωρίς συμφραζόμενα.

Γραμματικές τύπου 3 ή κανονικές γραμματικές: οι γραμματικές αυτές παράγουν τις κανονικές γλώσσες. Τέτοιες γραμματικές περιορίζουν τους κανόνες τους στο να περιέχουν ένα απλό μη τερματικό σύμβολο στο αριστερό μέλος ενός κανόνα το οποίο και αντικαθίσταται από ένα απλό τερματικό σύμβολο, ενδεχομένος ακολουθούμενο (ή προηγούμενο, αλλά όχι ταυτόχρονα και τα δύο στην ίδια γραμματική) από ένα απλό μη

τερματικό σύμβολο. Ο κανόνας $S \rightarrow \epsilon$ επιτρέπεται σε αυτές τις γραμματικές εάν το αρχικό μη τερματικό σύμβολο S δεν εμφανίζεται στο δεξί μέλος κανενός κανόνα - τέτοιες γραμματικές δηλαδή μπορούν να δημιουργήσουν τη κενή συμβολοσειρά. Αυτές οι γλώσσες, οι κανονικές δηλαδή, είναι ακριβώς οι ίδιες γλώσσες οι οποίες *αποφασίζονται* (και όχι απλά αναγνωρίζονται όπως πριν) από πεπερασμένα αυτόματα. Επιπλέον, αυτή η οικογένεια τυπικών γλωσσών μπορεί να αναπαρασταθεί και από *κανονικές εκφράσεις*. Οι κανονικές γλώσσες γενικά χρησιμοποιούνται για να οριστούν λεκτικά πρότυπα στη διαδικασία της λεκτικής ανάλυσης γλωσσών προγραμματισμού.

Παρατηρούμε ότι το σύνολο των γλωσσών που αντιστοιχούν στις αναδρομικές γλώσσες δεν είναι μέλος της παραπάνω ιεραρχίας.

Επιπλέον, κάθε κανονική γλώσσα είναι και γλώσσα χωρίς συμφραζόμενα, κάθε γλώσσα χωρίς συμφραζόμενα είναι και γλώσσα με συμφραζόμενα, κάθε γλώσσα με συμφραζόμενα είναι και αναδρομική γλώσσα και κάθε αναδρομική γλώσσα είναι και αναδρομικώς απαριθμήσιμη γλώσσα. Επιπλέον, υπάρχουν αναδρομικώς απαριθμήσιμες γλώσσες οι οποίες δεν είναι με συμφραζόμενα, γλώσσες με συμφραζόμενα οι οποίες δεν είναι χωρίς συμφραζόμενα και γλώσσες χωρίς συμφραζόμενα οι οποίες δεν είναι κανονικές γλώσσες.

Στον επόμενο πίνακα φαίνοται οι τέσσερεις τύποι γραμματικών που αποτελούν την ιεραρχία του Τσόμσκι, τις γλώσσες που η κάθε μια παράγει, τον τύπο του αυτομάτου το οποίο αναγνωρίζει ή αποφασίζει την εκάστοτε γλώσσα καθώς και τον τύπο των κανόνων παραγωγής που πρέπει η κάθε γραμματική να περιέχει:

Γραμματική	Γλώσσα	Αυτόματο	Μορφή Κανόνων
Τύπου 0	Αναδρομικώς Απαρ.	Μηχανή Turing	$\alpha \rightarrow \beta$
Τύπου 1	Με Συμφραζ.	ΓΦΑ	$\alpha A \beta \rightarrow \alpha \gamma \beta$
Τύπου 2	Χωρίς Συμφραζ.	Αυτόμ. με Στοίβα	$A \rightarrow \delta$
Τύπου 3	Κανονικές	Πεπερ. Αυτόμ.	$A \rightarrow \alpha$, $A \rightarrow \alpha B$

όπου στον παραπάνω πίνακα $\alpha, \beta, \gamma, \delta \in (\Sigma \cup N)^*$, $\gamma \neq \epsilon$, $\alpha \in \Sigma$ και $A, B \in N$.

Κεφαλαιο 3

Αυτοματα και Μηχανεσ Turing

3.1 Μηχανές Turing

Το 1936 ο άγγλος μαθηματικός Alan Turing πρότεινε και περιέγραψε τυπικά ένα γενικό μοντέλο υπολογισμού το οποίο έμεινε γνωστό ως *μηχανή Turing*. Η μηχανή αυτή έχει πολλές ομοιότητες με το γενικό πεπερασμένο αυτόματο, όμως οι κύριες διαφορές, όπως η άπειρη και απεριόριστη μνήμη καθώς μπορούμε να αποθηκεύσουμε ότιδήποτε στις καταστάσεις αυτής, την καθιστούν ως το κατάλληλο μαθηματικό μοντέλο για την τυπική περιγραφή του υπολογισμού όπως αυτός εκτελείται από τους σύγχρονους γενικής χρήσης υπολογιστές. Μια μηχανη Turing μπορεί να κάνει και να υπολογίσει οτιδήποτε ένας υπολογιστής μπορεί. Ακριβώς όμως αυτή η ισχύς της μηχανής Turing την περιορίζει καθώς προκύπτουν προβλήματα τα οποία είναι αδύνατον να επιλυθούν από αυτή, και κατ'επέκταση από οποιονδήποτε υπολογιστή.

Κατα καιρούς προτάθηκαν διάφορα μαθηματικά μοντέλα προσομοίωσης του υπολογισμού. Όλα όμως εν τέλει αποδείχτηκαν ισοδύναμα με την μηχανή Turing, με την έννοια ότι οποιαδήποτε προβλήματα αυτά τα μοντέλα ήταν ικανά να λύσουν, υπήρχε μια μηχανή Turing η οποία επίσης τα έλυνε.

Διαισθητικά η λειτουργία μιας μηχανής Turing είναι πολύ απλή. Χρησιμοποιεί μια απεριόριστη (ως προς οποιοδήποτε άκρο αλλά παρακάτω θα χρησιμοποιούμε ένα ισοδύναμο μοντέλο η ταινία του οποίου είναι απεριόριστη ως προς το ένα μόνο άκρο, το δεξί χωρίς βλάβη της γενικότητας) ταινία και έχει μια απεριόριστη μνήμη. Είναι εφοδιασμένη με μια κεφαλή η οποία μπορεί να διαβάζει κάποιο σύμβολο μια ορισμένη στιγμή, και ανάλογα με το σύμβολο που διαβάζει και την κατάσταση που βρίσκεται σε δεδομένη χρονική στιγμή, μπορεί να αλλάξει το σύμβολο το οποίο διαβάζει και να κινηθεί κατά μια θέση δεξιά ή αριστερά ή να μείνει εκεί που είναι. Η έννοια της κατάστασης αντικατοπτρίζει την πληροφορία που έχει αποθηκεύσει η μηχανή μέχρι στιγμής. Αυτό της δίνει τη δυνατότητα να "θυμάται" διάφορα πράγματα, ώστε να κάνει τις κατάλληλες ενέργειες στο μέλλον. Αρχικά η ταινία περι˜χει την είσοδο, μια ακολουθία από σύμβολα, και κένά σύμβολα οπουδήποτε αλλού. Η μηχανή συνεχίζει να κάνει υπολογισμούς, δηλαδή να αποθηκεύει πληροφορίες και να διαβάζει σύμβολά, έως ότου αποφασίσει να σταματήσει παράγωντας μια έξοδο. Η έξοδος αυτή μπορεί να είναι "ναι" ή "όχι" εάν η αρχική συμβολοσειρά πλειρή κάποια συνθήκη, ή καποια άλλη έξοδος η οποία θα είναι συνάρτηση της αρχικής συμβολοσειράς. Φυσικά, σε πολλές περιπτώσεις, τίποτα δεν μπορεί να μας εγγυηθεί ότι όντως η μηχανή κάποτε θα τερματίσει. Στην περίπτωση αυτή η μηχανή συνεχίζει τον υπολογισμό της για πάντα.

Συνοψίζοντας τις διαφορές μεταξύ πεπερασμένων (ντετερμινιστικών ή μη) αυτομάτων και μηχανών Turing, αυτές είναι:

- ▷ Μια μηχανή Turing μπορεί να γράψει στην ταινία και να διαβάσει από αυτήν.
- ▷ Η κεφαλή εγγραφής-ανάγνωσης μπορεί να κινηθεί και προς τις δύο κατευθύνσεις, δηλαδή δεξιά και αριστερά από την τρέχουσα θέση της.
- ▷ Η ταινία είναι άπειρη.
- ▷ Οι ειδικές καταστάσεις αποδοχής ή απόρριψης τερματίζουν άμμεσα την λειτουργία της μηχανής Turing.

3.2 Τυπικός ορισμός των μηχανών Turing

Το πιο σημαντικό κομμάτι στην περιγραφή μιας μηχανής Turing είναι ο ορισμός της συνάρτησης μετάβασης, επειδή αυτή καθορίζει με πιο τρόπο θα λειτουργεί κάθε δεδομένη στιγμή η μηχανή, δηλαδή με πιον τρόπο θα συνεχίζεται ο υπολογισμός. Για μια μηχανή Turing η συνάρτηση μετάβασης δ έχει την εξής ερμηνεία: δεδομένου του τρέχοντος συμβόλου που διαβάζει η κεφαλή

από την ταινία και της τρέχουσας κατάστασης στην οποία η μηχανή βρίσκεται, η κεφαλή γράφει έναν νεό χαρακτήρα στην θέση την οποία βρίσκεται, αλλάζει την κατάσταση της μηχανής και μετακινείται κατάλληλα κατά μία θέση δεξιά ή αριστερά ή μένει εκεί που είναι.

Πιο τυπικά, έστω Q το σύνολο καταστάσεων την μηχανής και Γ το αλφάβητο της ταινίας τότε η συνάρτηση είναι της μορφής $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\to, \leftarrow, -\}$.

Ορισμός 4 *Μια μηχανή Turing είναι μια πεντάδα $(Q, \Sigma, \Gamma, \delta, s)$ όπου τα Q είναι ένα πεπερασμένο σύνολο καταστάσεων το οποίο περιέχει τις ειδικές καταστάσεις τερματισμού q_{halt}, q_{accept} και q_{reject}. Σ είναι το αλφάβητο εισόδου το οποίο δεν περιέχει το σύμβολο κενού $\sqcup$, Γ είναι το αλφάβητο της ταινίας που περιέχει το σύμβολο κενού $\sqcup$ και το σύμβολο της αρχής της συμβολοσειράς $\triangleright$ (το οποίο δείχνει το αριστερότερο άκρο της ταινίας όπου η κεφαλή δεν μπορεί να προσπελάσει προς τα αριστερά, κάτι που δεν επηρεάζει την γενικότητα και την ισχύ της μηχανής Turing), $s \in Q$ είναι η αρχική κατάσταση της μηχανής και $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\to, \leftarrow, -\}$είναι η συνάρτηση μετάβασης. Αφού η κεφαλή δεν μπορεί να πάει αριστερότερα του συμβόλου $\triangleright$, έχουμε ότι $\delta(q, \triangleright) = (p, \triangleright, \to)$, για οποιεσδήποτε καταστάσεις q, p, δηλαδή το σύμβολο $\triangleright$ δεν μπορεί να προσπελαθεί προς τα αριστερά και ποτέ δεν διαγράφεται.*

Πως όμως λειτουργεί μια μηχανή Turing; Αρχικά, μια μηχανή Touring, που περιγράφεται από τον παραπάνω ορισμό, δέχεται την είσοδο, μια συμβολοσειρα $w = \sigma_1\sigma_2 \ldots \sigma_n \in \Sigma^*$ το οποίο το αποθηκεύει στις n αριστερότερες θέσεις της ταινίας. Η αρχή της συμβολοσειράς δεικτοδοτείται από το σύμβολο $\triangleright$ το οποίο και τοποθετείται ακριβώς στα αριστερά της συμβολοσειράς εισόδου. Οι υπόλοιπες θέσεις της ταινίας περιέχουν το κενό σύμβολο $\sqcup$ το οποίο υποθέσαμε ότι δεν ανήκει στο αλφάβητο εισόδου, δηλαδή η αρχική συμβολοσειρά εισόδου δεν το περιέχει, και άρα η πρώτη εμφάνιση ενός τέτοιου συμβόλου σηματοδοτεί το τέλος της εισόδου. Η κεφαλή αρχικά τοποθετείται στην αριστερότερη θέση της ταινίας η οποία πάντα περιέχει το σύμβολο $\triangleright$. Από την στιγμή που αρχίζει η λειτουργία της μηχανής Turing, ο υπολογισμός της συνεχίζεται όπως περιγράφεται από την συνάρτηση μετάβασης, μέχρι να φτάσει σε μια κατάσταση τερματισμού, q_{halt}, q_{accept} και q_{reject}. Εάν καμιά τέτοια κατάσταση δεν έχει εμφανιστεί κατά τη διάρκεια του υπολογισμού, η μηχανή συνεχίζει. Εάν η μηχανή απλά τερματίσει στην κατάσταση q_{halt} με είσοδο $w = \sigma_1\sigma_2 \ldots \sigma_n$, τότε θεωρούμε ως έξοδο της μηχανής το περιεχόμενο της ταινίας κατά τον τερματισμό της και συμβολίζουμε $M(w) = y$ όπου M η μηχανή Turing και y μια συμβολοσειρά που αντιστοιχεί στο περιεχόμενο της ταινίας αμέσως μετά τον τερματισμό της μηχανής M.

Καθώς ο υπολογισμός της μηχανής Turing συνεχίζει, αλλαγές συμβαίνουν στην τρέχουσα κατάσταση, στο τρέχων σύμβολο που διαβάζει η κεφαλή καθώς και στα περιεχόμενα της ταινίας. Για να αντικατοπτριστούν κάθε χρονική στιγμή οι τιμές των τριών αυτών αντικειμένων, οριζουμε την έννοια της *διαμόρφωσης*. Η τρέχουσα διαμόρφωση, που πρέπει να παρασταθεί με συγκεκριμένο τρόπο, μας δίνει πληροφορία για την τρέχουσα διαμόρφωση του υπολογισμού, όπως αυτός περιγράφεται από τις τιμές των τριών παραπάνω αντικειμένων. Εστω ότι η τρέχουσα κατάσταση είναι η q, η κεφαλή διαβάζει κάποιο σύμβολο σ. Αριστερά του σ υπάρχει μια υποσυμβολοσειρά της συμβολοσειράς της ταινίας, u, ενώ το σύμβολο σ μαζί με την υπόλοιπη υποσυμβολοσειρά στα δεξιά του συμβολίζεται με v. Δηλαδή εάν η συμβολοσειρά της ταινίας τη δεδομένη χρονική στιγμή είναι η w, τότε $w = uv$. Αυτή η διαμόρφωση συμβολίζεται ως $(u\ q\ v)$. Για παράδειγμα, η διαμόρφωση $(111\ q_2\ 01)$ σημαίνει ότι η μηχανή βρίσκεται αυτή τη στιγμή στη κατάσταση q_2 και η κεφαλή διαβάζει το 0, ενώ η συμβολοσειρά που περιέχει η ταινία είναι η 11101.

Τι σχέση έχουν όμως μεταξύ τους οι διάφορες διαμορφώσεις μιας μηχανής;

Ορισμός 5 *Εστω M μιά μηχανή Turing. Λέμε ότι μια διαμόρφωση $(u\ q\ v)$ αποφέρει σε ένα βήμα την διαμόρφωση $(u'\ q'\ v')$, γεγονός το οποίο συμβολίζεται ως $(u\ q\ v) \vdash^M (u'\ q'\ v')$, εάν, διαισθητικά, ένα βήμα της μηχανής μας από την διαμόρφωση $(u\ q\ v)$ μας μεταφέρει στην διαμόρφωση $(u'\ q'\ v')$. Πιο τυπικά, ισχύει το ακόλουθο: αρχικά, έστω σ το αρχικό σύμβολο της συμβολοσειράς v και έστω ότι έοχυμε την μετάβαση $\delta(q, \sigma) = (p, \rho, K)$. Τότε, αρχικά, πρέπει να ισχύει ότι $q' = p$. Εχουμε τρεις περιπτώσεις:*

1. *$K = \rightarrow$: Τότε $v' = v$ εκτός από το πρώτο σύμβολο, το οποίο ήταν σ, το οποίο προσαρτάται τώρα ως rho στην συμβολοσειρά u. Δηλαδή $u' = u\rho$.*

2. *$K = \leftarrow$: Τότε $u' = u$ εκτός από το τελευταίο σύμβολο το οποίο προστίθεται στη συμβολοσειρά v' και της οποίας το πρώτο, πριν την προσάρτηση, σύμβολο μετατρέπεται από σ σε ρ.*

3. *Τέλος, εάν $K = -$ τότε $u' = u$ και $v' = v$ εκτός από το πρώτο σύμβολο το οποίο μετατρέπεται από σ σε ρ.*

Αφού ορίσαμε τη σχέση της "παραγωγής σε ένα βήμα" μεταξύ των διαμορφώσεων, μπορούμε να ορίσουμε τη σχέση "παράγει" ως την μεταβατική κλειστότητα της αρχικής σχέσης. Λέμε ότι μια διαμόρφωση C παράγει ή αποφέρει σε $k \geq 0, k \in \mathbb{N}$ βήματα τη διαμόρφωση C', το οποίο το συμβολίζουμε ως $C \vdash^k C'$, εάν υπάρχουν $k-1$ ενδιάμεσες καταστάσεις C_i έτσι ώστε να ισχύει $C \vdash C_1 \vdash \cdots \vdash C_{k-1} \vdash C'$. Τέλος, λέμε ότι η διαμόρφωση C αποφέρει ή παράγει τη διαμόρφωση C', το οποίο συμβολίζεται ως $C \vdash^* C'$, εάν υπάρχει $k \geq 0, k \in \mathbb{N}$ τέτοιο ώστε $C \vdash^k C'$.

3.3 Υπολογισμοί με Μηχανές Turing

Γενικά οι μηχανές Turing φαίνονται ιδανικά αυτόματα για να λύνουν συγκεκριμένου τύπους προβλημάτων ή να υπολογίζουν συναρτήσεις ως προς διάφορες συμβολοσειρές, δηλαδή φαίνονται ικανές να υπολογίζουν και να αποφασίζουν ή να αποδέχονται διάφορες γλώσσες:

Ορισμός 6 *Εστω μια γλώσσα $L \subset \Sigma^*$ στο αλφάβητο Σ. Εστω M μια μηχανή Turing τέτοια ώστε, για κάθε συμβολοσειρά $x \in \Sigma^*$ να μπορεί να λέει "ναι" εάν $x \in L$, ενώ εάν $x \notin L$ τότε η M να αποκρίνεται "όχι" (δηλαδή, μετά το τέλος της λειτουργίας της να μεταβαίνει σε μια ειδική κατάσταση αποδοχής q_{accept} ή απόρριψης q_{reject} αντίστοιχα). Τότε, η μηχανή M* **αποφασίζει** *τη γλώσσα L. Εάν μια γλώσσα L αποφασίζεται από μια μηχανή Turing ότε η L είναι* **αναδρομική** *γλώσσα.*

Εάν η μηχανή Turing M απαντάει (θετικά) μόνον όταν $x \in L$, ενώ στην περίπτωση που $x \notin L$ η M δεν τερματίζει ποτέ, τότε η μηχανή M **αποδέχεται** *την γλώσσα L. Μια γλώσσα που γίνεται αποδεκτή από κάποια μηχανή Turing, είναι* **αναδρομικώς απαριθμήσιμη.**

Αν και ο όρος της αποφασισιμότητας μιας γλώσσας φαίνεται λογικός και διαισθητικός, από την άλλη μεριά, ο όρος της αποδοχής παρουσιάζει μια ασυμετρία. Η μηχανή απαντάει μόνο όταν η συμβολοσειρά ανήκει στην υποκείμενη γλώσσα, ενώ αν δεν ανήκει τότε η μηχανή δε τερματίζει ποτέ. Πώς όμως είμαστε σίγουροι ότι η μηχανή με τη δεδομένη είσοδο δεν τερματίζει. Το αντίστοιχο πρόβλημα, δηλαδή αυτό του τερματισμού μιας μηχανής με είσοδο μια δεδομένη συμβολοσειρά είναι ίσως το πιο γνωστό μη επιλύσημο πρόβλημα. Αυτό μας δείχνει ότι η αποδοχή μιας γλώσσας δεν είναι μια κανονική αλγοριθμική έννοια, όμως μας βοηθάει να κατηγοριοποιήσουμε τα διάφορα προβλήματα. Ενα ανάλογο πρότυπο υπάρχει και στον ορισμό των μη ντετερμινιστικών μηχανών Turing, όπως εξ'άλλου θα δούμε και σε επόμενη ενότητα του παρόντος κεφαλαίου.

Οσον αφορά τη σχέση των αναδρομικών με τις αναδρομικώς απαριθμήσιμες γλώσσες, το παρακάτω αποτέλεσμα είναι εύκολο να αποδειχτεί:

Πρόταση 1 *Εάν μια γλώσσα L είναι αναδρομική, τότε είναι και αναδρομικώς απαριθμήσιμη.*

Απόδειξη: Εστω μια μηχανή Turing M η οποία αποφασίζει μια γλώσσα L, δηλαδή η γλώσσα L είναι ανδρομική. Θα κατασκευάσουμε μια νέα μηχανή Turing M' η οποία να *αποδέχεται* τη συγκεκριμένη γλώσσα ως εξής: η M' λειτουργεί τελείως αντίστοιχα με την M, με τη μόνη διαφορά ότι όταν η M ετοιμάζεται να εισέλθει σε μια κατάσταση απόρριψης της δεδομένης συμβοσειράς, η M' πέφτει σε μια άπειρη ανακύκλωση, για παράδειγμα αρχίζει να κινείται επ'άπειρον προς τα δεξιά, και δε τερματίζει ποτέ. □

3.4 Μη Ντερμινιστικές Μηχανές Turing

Στην παρούσα ενότητα θα ορίσουμε ένα μοντέλο μηχανών Turing που όπως δεν ανταποκρίνεται στην έννοια του πραγματικού υπολογισμού. Οι λόγοι που ορίζουμε τέτοια μη ρεαλιστικά μοντέλα είναι κυρίως για κατηγοριοποίηση και μελέτη διαφόρων προβλημάτων. Παρ'όλ'αυτά, το νέο μοντέλο θα είναι *ισοδύναμο* με το κλασσικό μοντέλο της μηχανής Turing, με την έννοια ότι κάθε μηχανή Turing είναι σε έση να προσομοιώσει το νέο μη ντετερμινιστικό μοντέλο αλλά χάνοντας εκθετική ισχύ, όσον αφορά τον χρόνο εκτέλεσης.

Ορισμός 7 *Μια μη νετερμινιστική μηχανή Turing είναι κα πάλι, όπως και στην περίπτωση των κλασσικών μηχανών Turing, μια διατεταγμένη τετράδα* $N = (K, \Sigma, \Delta, s)$. *Τα* K, Σ *και* s *ορίζονται όπως και πρίν, δηλαδή αντιστοιχούν στο σύνολο καταστάσεων, στο αλφάβητο και στην αρχική κατάσταση. Η κύρια διαφορά έγκειται στον ορισμό της συνάρτησης μετάβασης. Το γεγονός ότι μια μη ντετερμινιστική μηχανή, όπως εξ'άλλου υποδηλώνει και ο όρος "μη ντετερμινισμός", δεν έχει μια ακριβώς καθορισμένη επόμενη κίνηση τουλάχιστον σε ένα βήμα του υπολογισμού της αλλά ένα σύνολο επιλογής της επόμενης κίνησης, "αναγκάζει" τη* Δ *να μην είναι πια συνάρτηση αλλά* **σχέση**. *Δηλαδή* $\Delta \subset (K \times \Sigma) \times \Big((K \cup \{h, \text{“}\textbf{ναι}\text{”}, \text{“}\textbf{οχι}\text{”}\}) \times \Sigma \times \{-, \rightarrow, \leftarrow\}\Big)$, *κάτι το οποίο σημαίνει ότι για κάθε συνδιασμό καταστάσεως - συμβόλου ενδέχεται να υπάρχουν περισσότερες της μίας διαθέσιμες μεταβάσεις.*

Οι διαμορφώσεις μιας μη ντετερμινιστικής μηχανής Turing ορίζονται με τελείως ανάλογο τρόπο όπως και στη νετερμινιστική περίπτωση με τη μόνη διαφορά ότι η σχέση "αποφέρρει" 'η "παράγει" δεν είναι πλέον συνάρτηση αλλά *σχέση*. Κατά τ'άλλα, οι νόμιμες μεταβάσεις μεταξύ διαμορφώσεων ορίζονται τελείως ανάλογα.

Ο υπολογιμός μιας μηχανής Turing είναι στην ουσία ένα δένδρο τα κλαδία του οποίου αντιστοιχούν σε διαφορετικά μονοπάτια - μη ντετερμινιστικές επιλογές της μηχανής. Εάν *κάποιο* τέτοιο υπολογιστικό μονοπάτι οδηγεί σε αποδοχή της εισόδου, τότε η μηχανή αυτή αποδέχεται την είσοδό της. Αντίθετα, εάν *κανένα* τέτοιο υπολογιστικό μονοπάτι δεν οδηγεί σε αποδοχή, ή αντίστοιχα *όλα* οδηγούν σε απόρριψη, τότε η μηχανή απορρίπτει την είσοδό της.

Από τα παραπάνω, είναι τελείως σαφές ότι κάθε ντετερμινιστική μηχανή Turing είνια και μη ντετερμινιστική καθώς οι συναρτήσεις είναι ειδική περίπτωση σχέσεων. Δηλαδή οι νετερμινιστικές μηχανές Turing είναι στην οουσία μη ντετερμινιστικές μηχανές για τις οποίες το επόμενο βήμα είναι πάντοτε καλά καθορισμένο.

Θεώρημα 1 *Κάθε μη ντετερμινιστική μηχανή Turing η οποία εκετελείται σε χρόνο $t(n)$ έχει μια ισοδύναμη ντετερμινιστική μηχανή Turing, που αποφασίζει δηλαδή την ίδια γλώσσα, η οποία εκτελείται σε χρόνο $2^{\mathcal{O}(t(n)}$ για εισόδους μεγέθους n.*

Απόδειξη: Εστω μια μη ντετερμινιστική μηχανή Turing N η οποία εκετελείται σε χρόνο $t(n)$. Θέλουμε να κατασκευάσουμε έναν ντετερμινιστικό αλγόριθμο, ουσιαστικά μια ντετερμινιστική μηχανή Turing M, η οποία θα προσομειώνει τις μη ντετερμινιστικές επιλογές της μη ντετερμινιστικής μηχανής N.

Η προσομειούμενη ντετερμινιστική μηχανή M θεωρεί όλες τις ακολουθίες μη ντετερμινιστικών επιλογών, με αύξουσα διάταξη μήκους, και για κάθε μία τέτοια ακολουθία, προσομειώνει την μη ντετερμινιστική μηχανή N, η οποία για δωθέν πρόβλημα Π, αποφασίζει την αντίστοιχη γλώσσα σε χρόνο $t(n)$. Ο χρόνος $t(n)$ δεν είναι εκ των πρωτέρων γνωστός στην N, άρα πρέπει να θεωρήσει όλες τις ακολουθίες και όχι μόνο αυτές μήκους $t(|n|)$. Εάν οι αντίστοιχες επιλογές της εκάστοτε ακολουθίας που προσομειώνει η μηχανή M οδηγούν σε μια τελική "ναι" κατάσταση, τότε η μηχανή M σταματάει επίσης σε μια "ναι" κατάσταση τερματισμού, αλλιώς απλά προχωράει θεωρώντας την επόμενη ακολουθία μη ντετερμινιστικών επιλογών.

Για να δημιουργηθεί η επόμενη ακολουθία από την ντετερμινιστική μηχανή M, αρκεί να δούμε ότι κάθε μια ακολουθία επιλογών μπορεί να θεωρηθεί ως μια ακολουθία ακεραίων από το πεδίο $0, 1, \ldots, d-1$, όπου d είναι ο μέγιστος αριθμός επιλογών που μορεί να έχει σε κάποιο βημα η μη ντετερμινιστική μηχανή N. Αρα για τη δημιουργία της επόμενης ακολουθίας από την M αρκεί να δημιουργηθεί ο επόμενος ακέραιος στο d-αδικό σύστημα. Εάν σε κάποιο βήμα της προσομείωσης, η μηχανή M θεωρεί ακολουθίες μήκους έστω t, και καμία από αυτές δεν τερματίζει σε μια "ναι" κατάσταση και επιπλέον δεν υπάρχει καμία ακολουθία της οποίας να συνεχίζει ο υπολογισμός με επιλογές μήκους μεγαλύτερου του t, τότε η μηχανή M μπορεί να συμπεράνει ότι η μηχανή N απορρίπτει την είσοδό της. Αρα ο συνολικός χρόνος για να προσομειωθεί η μη ντετερμινιστική μηχανή N από την M είναι $\sum_{i=1}^{t(n)} d^i = \mathcal{O}(d^{t(n)})$ επί τον χρόνο που χρειάζεται για να δημιουργηθεί και να ελεγχθεί κάθε ακολουθία επιλογών, χρόνος ο οποίος είναι $\mathcal{O}(2^{t(n)})$. □

3.5 Κλάσεις Πολυπλοκότητας Χρόνου και Χώρου

Στην παρούσα ενότητα, θα ορίσουμε τις κλάσεις πολυπλοκότητας χώρου και χρόνου ενδιαφέροντός μας και θα παρουσιάσουμε ορίσμένα χρήσιμα αποτελέσματα.

3.5.1 Πολυπλοκότητα Χρόνου

Ακόμα και εάν ένα πρόβλημα είναι γενικά επιλύσιμο, αυτό σε καμία περίπωση δε σημαίνει ότι μπορεί να επιλυθεί σε λογικό χρόνο, τουλάχιστον με τη σημερινή τεχνολογία υπολογιστικών συστημάτων. Παρακάτω, ορίζουμε τον τρόπο με τον οποίο "μετριέται" ο υπολογιστικός χρόνος ενός υπολογιστικού μονέλου: τα βήματα που χρειάζονται ώστε να τερματίσει την λειτουργία του.

Εστω μια μηχανή Turing η οποία τερματίζει για όλες τις πιθανές εισόδους. Ο *χρόνος εκτέλεσης* της μηχανής αυτής είναι μια συνάρτηση $f : \mathbb{N} \to \mathbb{N}$ όπου η συνάρτηση αυτή, με είσοδο μια συμβολοσειρά $x : |x| = n$ αντιπροσωπεύει το μέγιστο αριθμό υπολογιστικών βημάτων που χρειάζεται ώστε η μηχανή να τερματίσει τη λειτουργία της για κάθε είσοδο. Αυτό ισχύει και για τις νετερμινιστικές και για τις μη ντετερμινιστικές μηχανές Turing.

Για τις ντερμινιστικές έχουμε τον εξής ορισμό:

Ορισμός 8 *Εστω $g : \mathbb{N} \to \mathbb{R}$ μια συνάρτηση. Ορίζουμε την κλάση πολυπλοκότητας* $\mathbf{TIME}(g(n))$ *να είναι ένα σύνολο που περιέχει τις γλώσσες οι οποίες αποφασίζονται σε χρόνο $\mathcal{O}(g(n))$ από κάποια μηχανή Turing, για εισόδους μεγέθους n.*

Λέμε ότι μια μη ντετερμινιστική μηχανή Turing N αποφασίζει μια γλώσσα L σε χρόνο $f(n)$, $f : \mathbb{N} \to \mathbb{N}$, εάν η N αφ'ενός αποφασίζει τη γλώσσα L και αφ'εταίρου κάθε υπολογιστικό μονοπάτι δεν έχει μήκος μεγαλύτεο από $f(|x|)$ πολογιστικά βήματαγια τη συμβολοσειρά εισόδου $|x|$. Δηλαδή, όταν τερματίζει η μηχανή, το κάνει μετά από $k \leq f(|x|)$ βήματα. Με άλλα λόγια, θεωρούμε ως "χρόνο" εκτέλεσης της μηχανής το *ύψος* του υπολογιστικού δέντρου της δεδομένης μηχανής, χωρίς να λαμβάνουμε υπ'όψιν το *πλάτος*, δηλαδή το πόσα διαφορετικά υπολογιστικά μονοπάτια αυτό έχει.

Ορισμός 9 *Εστω $t : \mathbb{N} \to \mathbb{R}$ μια συνάρτηση. Ορίζουμε την κλάση πολυπλοκότητας* $\mathbf{NTIME}(g(n))$ *να είναι ένα σύνολο που περιέχει τις γλώσσες οι οποίες αποφασίζονται σε χρόνο $\mathcal{O}(t(n))$ από κάποια* **μη ντετερμινιστική** *μηχανή Turing, για εισόδους μεγέθους n.*

Ισως η πιο σημαντική κλάση πολυπλοκότητας χρόνου είναι η κλάση **NP** η οποία ορίζεται ως εξής: $\mathbf{NP} = \bigcup_k \mathbf{NTIME}(n^k)$, δηλαδή το σύνολο εκείνων των γλωσσών που αποφασίζονται σε πολυωνυμικό χρόνο ως προς το μέγεθος της εκάστοτε εισόδου. Στην κλάση αυτή ανήκουν τα περισσότερα, αν όχι όλα, τα προβλήματα πρακτικού και θεωρητικού ενδιαφέροντος. Είναι γενικά η κλάση των προβλημάτων τα οποία έχουν πολυωνυμικού χρόνου, μη ντετερμινιστικούς αλγορίθμους. Για πολλά από αυτά, δεν είναι γνωστό εάν αυτοι οι πολυωνυμικού χρόνου μη ντετερμινιστικοί αλγόριθμοι μπορούν να μεταφραστούν σε πολυωνυμικού χρόνου νετερμινιστικούς αλγορίθμους.

3.5.2 Πολυπλοκότητα Χώρου

Μαζί με την πολυπλοκότητα χρόνου, η πολυπλοκότητα *χώρου* είναι μια σημαντικότατη κλάση με την οποία μετράμε την ανάθεση πόρων σε διάφορους πρακτικούς υπολογισμούς. Η πολυπλοκότητα αυτή αντικατοπτρίζει τη *μνήμη* που χρειάζεται ένα πρόβλημα για να εκτελεστεί. Οπως και στη περίπτωση της χρονικής πολυπλοκότητας, όσο πιο μικρή η ποσότητα μνήμης που χρειάζεται ένα δεδομένο πρόβλημα, τόσο πιο εφικό είναι, όταν αναφαρόμαστε στην χωρική πολυπλοκότητα. Αυτού του είδους η πολυπλοκότητα έχει πολλά κοινά στοιχεία με την αντίστοιχη χρονική και παράλληλα μας βοηθάει στην περ'αιτέρω ομαδοποιήση υπολογιστικών προβλημάτων.

Αρχικά, δοθέντος ενός υπολογιστικού προβλήματος, είναι αναγκαίο να ορίσουμε τυπικά τι είναι η πολυπλοκότητα χώρου του προβλήματος αυτού. Αυτό φυσικά θα γίνει χρησιμοποιώντας για μοντέλο τις μηχανές Turing:

Ορισμός 10 *Εστω M μια μηχανή Turing η οποία τερματίζει για όλες τις εισόδους. Διακρίνουμε δύο περιπτώσεις σχετικά με τον τρόπο υπολογισμού της μηχανής αυτής:*

1. *Η μηχανή M είναι ντετερμινιστική. Τότε η πολυπλοκότητα χώρου της δεδομένης μηχανής M είναι μια συνάρτηση $f : \mathbb{N} \to \mathbb{N}$, όπου $f(n)$ είναι το μέγιστο πλήθος διευθύνσεων μνήμης που χρειάζεται μηχανή M ώστε να ολοκληρώσει τον υπολογισμό της για εισόδους μεγέθους (μήκους) n. Εάν η πολυπλοκότητα χώρου μιας μηχανής Turing M είναι $f(n)$ τότε λέμε ότι η μηχανή εκτελείται σε χώρο $f(n)$.*

2. *Η μηχανή M είναι μη ντετερμινιστική. Τότε, η πολυπλοκότητα χώρου της M είναι $f(n)$ που αντιστοιχεί στο ο μέγιστο αριθμό διευθύνσεων μνήμης (κελιών ταινίας στη περίπτωση των μηχανών Turing) που χρειάζεται να προσπελάσει η μηχανή ώστε να ολοκληρώσει τον υπολογισμό της για εισόδους μεγέθους n.*

Παρακάτω ορίζουμε τις κλάσεις πολυπλοκότητας χώρου:

Ορισμός 11 *Εστω $t : \mathbb{N} \to \mathbb{R}$ μια συνάρτηση. Ορίζουμε την κλάση πολυπλοκότητας* **SPACE**$(t(n))$ *να είναι ένα σύνολο που περιέχει τις γλώσσες οι οποίες αποφασίζονται σε χώρο $\mathcal{O}(t(n))$ από κάποια* **ντετερμινιστική** *μηχανή Turing, για εισόδους μεγέθους n.*

και αντίστοιχα για τη μη ντετερμινιστική περίπτωση

Ορισμός 12 *Εστω $t : \mathbb{N} \to \mathbb{R}$ μια συνάρτηση. Ορίζουμε την κλάση πολυπλοκότητας* $\mathbf{NSPACE}(t(n))$ *να είναι ένα σύνολο που περιέχει τις γλώσσες οι οποίες αποφασίζονται σε χώρο $\mathcal{O}(t(n))$ από κάποια* **μη ντετερμινιστική** *μηχανή Turing, για εισόδους μεγέθους n.*

Η κύρια κλάση ενδιαφέροντός μας είναι η κλάση $\mathbf{NSPACE}(n)$, δηλαδή η κλάση εκείνη που περιέχει όλα τα προβλήματα τα οποία αποφασίζονται σε χώρο ακριβώς γραμμικό σε σχέση με το μέγεθος της εισόδου (την οποία θεωρούμε να έχει μέγεθος n). Οπως θα δούμε και στη συνέχεια, η συγκεκριμένη κλάση είναι στην ουσία η κλάση των *γλωσσών με συμφραζόμενα*.

3.5.3 Σχέση μεταξύ των κλάσεων πολυπλοκότητας

Τα προβλήματα όμως που ανήκουν στην κλάση **NP**, έχουν μια πολύ σημαντική ιδιότητα, την οποία και θα αξιοποιήσουμε κατά κόρον στο επόμενο κομμάτι της παρούσης διατριβής. Πριν αναλύσουμε την ιδιότητα αυτή , είναι απαραίτητο να ορίσουμε τι είναι οι *επαληθευτές πολυωνυμικού χρόνου*:

Ορισμός 13 *Ενας επαληθευτής για μιά γλώσσα A είναι ένας αλγόριθμος V όπου*

$$A = \{w \mid V(w, c) = \text{“ναι”}, \text{για κάποιο } c \in \Sigma(A)^*\}$$

Μετρούμε το χρόνο εκτέλεσης του υπολογισμού του επαληθευτή πάντα ως προς το μήκος της συμβολοσειράς εισόδου w. Αρα, ένας πολυωνυμικού χρόνου επαληθευτής εκτελείται σε χρόνο πολυωνυμικά φραγμένο ως προς το μέγεθος της συμβολοσειράς εισόδου του αλγορίθμου - μηχανής A. Η γλώσσα A είναι πολυωνυμικά επαληθεύσιμη εάν έχει κάποιον πολυωνυμικού χρόνου επαληθευτή.

Για να γίνει πιο κατανοητή η λειτουργία των επαληθευτών, ας θεωρήσουμε το παρακάτω παράδειγμα:

Παράδειγμα 2 *Εστω το πρόβλημα του (μη κατευθυνόμενου) μονοπατιού Χάμιλτον σε ένα γράφημα. Αυτό που επιθυμούμε είναι, δοθέντος ενός μη κατευθυνόμενου γραφήματος $G = (V, E)$ να απαντήσουμε θετικά ή αρνητικά στο ερώτημα του κατά πόσον το γράφημα αυτό έχει μονοπάτι Χάμιλτον, δηλαδή μονοπάτι που να περνάει ακριβώς μία φορά από κάθε κορυφή του γραφήματος.*

Προς το παρόν, δεν υπάρχει πολυωνυμικού χρόνου ντετερμινιστικός αλγόριθμος για το πρόβλημα αυτό. Μια πιθανότητα επίλυσης είναι η εξής: δοκιμάζουμε όλα τα δυνατά μονοπάτια, το οποία μπορεί να είναι εκθετικά σε

πλήθος, μήκους $|V|-1$ ακμών. Εάν κάποιο από αυτό κωδκοποιεί κατάλληλα κάποιο μονοπάτι Χάμιλτον, τότε απαντάμε "ναι". Εάν κανένα τέτοιο δυνατό μονοπάτι δεν ικανοποιεί τη συνθήκη του μονοπατιού Χάμιλτον, τότε απαντάμε "όχι". Από τη στιγμή που μπορέι να υπάρξουν εκθετικά πολλά διαφορετιά υποψήφια μονοπάτια, ο παραπάνω "κουτός" αλγόριθμος εκτελείται σε εκθετικό, ως προς την αναπαράσταση του γραφήματος, χρόνο.

Ομως, το παραπάνω πρόβλημα έχει την εξής ιδιότητα. Εάν κάποιος "μαγικός" αλγόριθμος, ένα μαύρο κουτί, μας δώσει ένα υποψήφιο μονοπάτι Χάμιλτον, ο έλεγχος του κατά πόσον είναι όντος ένα έγκυρο μονοπάτι Χάμιλτον, διαδικασία την οποία εξ'άλλου χρησιμοποιούμε και στον παραπάνω εκθετικό αλγόριθμο μπορεί να γίνει σε πολυωνυμικό χρόνο. Αν και δεν έχουμε πολυωνυμικού χρόνου αλγορίθμους για το πρόβλημα αυτό, εντούτοις, εάν κάποιος μας εφοδιάζει με πιθανά μονοπάτια Χάμιλτον, μπορούμε, σε πολυωνυμικό χρόνο, να επαληθεύουμε την εγκυρότητα αυτών: απλά κοιτάμε εάν το μονοπάτι έχει μήκος $|V|-1$ και δεν επαναλαμβάνει κάποια κορυφή. Με άλλα λόγια, η επαλήθευση ενός πιθανού μονοπατιού Χάμιλτον, είναι μια διαδικασία πολύ ευκολότερη από την εύρεση αυτού.

Αυτή η διαδικασία είναι στην ουσία ο "επαληθευτής" V του παραπάνω ορισμού, ενώ το προς έλεγχο μονοπάτι Χάμιλτον είναι η συμβολοσειρά c. Αυτή η συμβολοσειρά ονομάζεται και **πιστοποιητικό**, *αφού πιστοποιεί την ύπαρξη μονοπατιού Χάμιλτον. Στη γενική περίπτωση, αφού ο επαληθευτής εκτελείται σε πολυωνυμικό χρόνο, τότε προφανώς και το πιστοποιητικό δεν μπορεί να έχιε μήκος μεγαλύτερο από πολυωνυμικό.*

Το επόμενο αποτέλεσμα είναι χαρακτηριστικό της σχέσης προβλημάτων που ανήκουν στην κλάση **NP** και αυτών που έχουν πολυωνυμικούς επαληθευτές.

Θεώρημα 2 *Εστω $L \subseteq \Sigma^*$ μια γλώσσα στο αλφάβητο Σ. Τότε, $L \in \mathbf{NP}$ εάν και μόνον εάν υπάρχει πολυωνυμικού χρόνου επαληθευτής, έστω V, τέτοιος ώστε $L = \{w \mid V(w,c) = \text{"ναι"}, \text{για κάποιο } c \in \Sigma(A)^*,\ c = poly(n)\}$*

Απόδειξη: Εστω ότι έχουμε στη διάθεσή μας έναν τέτοιο επαληθευτή V. Θα κατασκευάσουμε έναν πολυωνυμικού χρόνου μη ντετερμινιστικό αλγόριθμο (δηλαδή μηχανή Turing) A που θα αποφασίζει τη γλώσσα L ως εξής: ο αλγόριθμος A με είσοδο $w \in \Sigma^*$ "μαντεύει" μια συμβολοσειρά c με μήκος το πολύ $|w|^k$ για κάποιο $k \in \mathbb{Z}$ και τότε χρησιμοποιεί τον επαληθευτή V ώστε να αποφασίσει το κατά πόσον $V(w,c) = \text{"ναι"}$. Εάν η απάντηση είναι όντος "ναι" τότε η μηχανή αποκρίνεται "ναι" αλλιώς αποκρίνεται "όχι".

Για την αντίστροφη περίπτωση, έστω ότι η γλώσσα L ανήκει στη κλάση πολυπλοκότητας **NP**. Δηλαδή, υπάρχει ένας πολυωνυμικού χρόνου μη ντετερμινιστικός αλγόριθμος ο οποίος αποφασίζει τη δεδομένη γλώσσα σε χρόνο

$|w|^k$ για κάποιο $k \in \mathbb{Z}$. Ορίζουμε τον επαληθευτή V ως εξής: $V(w, c) =$ "ναι" εάν και μόνον εάν η συμβολοσειρά c κωδικοποιεί έναν υπολογισμό αποδοχής του αλγορίθμου A με είσοδο w. Είναι ξεκάθαρο ότι ο αλγόριθμος επαλήθευσης V εκτελείται σε πολυωνυμικό χρόνο (αφού ο A είναι πολυωνυμικού χρόνου, κανένα έγκυρο υπολογιστικό μονοπάτι δεν έχει μεγαλύτερο από πολυωνυμικό μήκος) και επιπλέον χρειάζεται γραμμικός ως προς το μήκος της συμβολοσειράς κωδικοποίησης του υπολογισμού για να αποφανθεί εάν αυτή είναι μια έγκυρη συμβολοσειρά υπολογισμού αποδοχής. Αρα ο αλγόριθμος V απαντάει "ναι" εάν και μόνον εάν $A(w) =$ "ναι". □

Το παραπάνω αποτέλεσμα μας δίνει τη δυνατότητα να αντιληφθούμε διαισθητικά τί ακριβώς σημαίνει να βρίσκεται ένα πρόβλημα στην κλάση πολυπλοκότητας **NP**: κάθε "ναι" στιγμιότυπο κάθε προβλήματος που ανήκει στην **NP**, έχει τουλάχιστον ένα σύντομο (πολυωνυμικό) πιστοποιητικό c του γεγονότος ότι όντος είναι ένα "ναι" στιγμιότυπο. Προφανώς, τα "όχι" στιγμιότυπα δεν έχουν τέτοια πιστοποιητικά. Οπως είπαμε και πριν, μπορέι να μην έχουμε εφικτούς τρόπου να "νακαλύψουμε" αυτά τα πιστοποιητικά, αλλά είμαστε σίγουροι ότι αυτά υπάρχουν εάν το στιγμιότυπο είναι ένα "ναι" στιγμιότυπο.

Οπως είπαμε, για το πρόβλημα του μονοπατιού Χάμιλτον το πιστοποιητικό είναι το ίδιο το μονοπάτι Χάμιλτον το οποίο και υπάρχει εάν και μόνον εάν το υποκείενο γράφημα έχει μονοπάτι Χάμιλτον. Στο πρόβλημα της ικανοποιησιμότητας μιας λογικής έκφρασης, το πιστοποιητικό είναι η εκείνη η ανάθεση αληθοτιμών στις μεταβλητές αυτής η οποία κάνει αληθή τη πρόταση. Και πάλι, το πιστοποιητικό αυτό υπάρχει εάν και μόνο εάν η πρόταση είναι, ή μπορέι να γίνει, αληθής δηλαδή εάν έιναι ικανοποιήσιμη.

Οπως βλέπουμε, τα πιστοποιητικά γενικά φαίνεται να είναι μέρος της εισόδου ή τουλάχιστον να μην είναι ποτέ μεγαλύτερα από την είσοδο. Πάνω σε αυτή τη παρατήρηση μπορούμε να κάνουμε την υπόθεση ότι η κλάση πολυπλοκότητας **NP** περιέχει όλα τα προβλήματα που έχουν επαληθευτές, και άρα πιστοποιητικά, *γραμμικού χώρου* (και όχι πολυωνυμικού χρόνου). Το παρακάτω αποτέλεσμα όμως κινείται στην τελείως αντίθετη κατεύθυνση:

Θεώρημα 3 $\mathbf{NP} \neq \mathbf{NSPACE}(n)$

Για την απόδειξη του παραπάνω θεωρήματος, θα μας βοηθήσει πολύ η παρακάτω συνάρτηση η οποία ενεργεί πάνω σε συμβολοσειρές. Αυτή η συνάρτηση, την οποία λόγω της λειτουργίας της θα την ονομάσουμε pad από τον αγγλιό όρο της έννοιας "επέκταση", θα έχει δύο ορίσματα: μία συμβολοσειρά w και έναν αριθμό z. Λειτουργία της είναι να προσθέτει στο τέλος της συμβολοσειράς w έναν χαρακτήρα που δεν υπάρχει στο αλφάβητο της

w, έστω τον $\sqcup$, τόσες φορές όσες ώστε η τελική συμβολοσειρά να έχει μήκος τουλάχιστον z. Πιο τυπικά, η συνάρτηση $\text{pad} : \Sigma^* \times \mathbb{N} \to \Sigma^* \sqcup^*$ ορίζεται ως εξής: $\text{pad}(w, z) = w\sqcup^i,\ i = \max(0, z - |w|)$. Με άλλα λόγια $|\text{pad}(w, z)| = \max(|w|, z)$.

Τώρα, έστω μια γλώσσα L. Για κάθε συνάρτηση $f : \mathbb{N} \to \mathbb{N}$ ορίζουμε τη συνάρτηση pad που ενεργεί σε *γλώσσες* ως εξής:

$$\text{pad}(L, f(n)) = \{pad(x, f(|x|)),\ x \in L\}.$$

Μια πού σημαντική ιδιότητα της συνάρτησης pad την οποία θα χρησιμοποιήσουμε στην συνέχεια για τον διαχωρισμό των κλάσεων **NP** και $\mathbf{NSPACE}(n)$ είναι η παρακάτω, την οποία θα διατυπώσουμε με μορφή λήμματος:

Λήμμα 1 *Για κάθε γλώσσα L και για κάθε $k \in \mathbb{Z}$ σχύει ότι*

$$pad(L, n^k) \in \mathbf{NP} \Leftrightarrow A \in \mathbf{NP}$$

Απόδειξη: Αρχικά θα αποδείξουμε την πρώτη κατεύθυνση. Εστω ότι $\text{pad}(L, n^k)$ για κάποια γλώσσα L και για κάποιο $k \in \mathbb{Z}$. Αυτό σημαίνει ότι υπάρχει μη ντετερμινιστικός αλγόριθμος πολυωνυμικού χρόνου ο οποίος δοθείσας μιας συμβολοσειράς w απαντάει εάν $w \in \text{pad}(L, n^k)$. Εστω A αυτός ο αλγόριθμος. Για να δείξουμε ότι η γλώσσα $L \in \mathbf{NP}$ αρκεί να κατασκευάσουμε έναν πολυωνυμικού χρόνου μη νετετρμινιστικό αλγόριθμο που να την αποφασίζει. Δοθείσας μιας συμβολοσειράς x για να δούμε το κατά πόσον ανήκει ή όχι στην L αυτό που κάνουμε είναι το εξής: επεκτείνουμε τη x ώστε να φτάσει σε μήκος $|x|^k$ και μετά ελέγχουμε μέσω του αλγορίθμου A το κατά πόσον $x^k \in \text{pad}(L, n^k)$.

Για την αντίστροφη κατεύθυνση έστω A' ο πολυωνυμικού χρόνου μη ντετερμινιστικός αλγόριθμος ο οποίος αποφασίζει τη γλώσσα L. Με βάση αυτό τον αλγόριθμο, θα κατασκευάσουμε έναν νέο πολυωνυμικού χρόνου και μη ντετερμινιστικό αλγόριθμο B που να αποφασίζει τη γλώσσα $\text{pad}(L, n^k)$. Θέλουμε, δοθείσας μιας συμβολοσειράς w να αποφανθούμε κατά πόσον αυτή ανήκει στη γλώσσα $\text{pad}(L, n^k)$. Γράφουμε τη συμβολοσειρά w ως $x\sqcup^j$ και ελέγχουμε το κατά πόσον ισχύει $|w| = |x|^k$ προφανώς σε πολυωνικό χρόνο. Επειτα, χρησιμοποιούμε τον αλγόριθμο A' για να διαπιστώσουμε εάν $x \in L$. □

Τώρα μπορούμε να επιστρέψουμε στην απόδειξη του διαχωρισμού των δύο κλάσεων πολυπλοκότητας **NP** και $\mathbf{NSPACE}(n)$:

Απόδειξη του Θεωρήματος 3 Ισχυριζόμαστε, προς χάρην της ατόπου απαγωγής, ότι $\mathbf{NP} = \mathbf{NSPACE}(n)$. Εστω L μια γλώσσα η οποία ανήκει στην

κλάση $\mathbf{NSPACE}(n^2)$ αλλά όχι στην $\mathbf{NSPACE}(n)$. Μια τέτοια γλώσσα υπάρχει σύμφωνα με το θεώρημα της ιεραρχίας χώρου[1]. Η γλώσσα $\mathrm{pad}(L, n^2)$ ανήκει στην κλάση $\mathbf{NSPACE}(n)$ γιατί έχουμε αρκετό χώρο ώστε να τρέξουμε τον $\mathcal{O}(n^2)$ χώρου αλγόριθμο για τη γλώσσα L, χρησιμοποιώντας χώρο ο οποίος είναι *γραμμικός* ως προς την γλώσσα $\mathrm{pad}(L, n^2)$. Εξ'αιτίας της υποθέσεως και του προηγούμενου λήμματος έχουμε ότι $\mathrm{pad}(L, n^2) \in \mathbf{NP} \Rightarrow L \in \mathbf{NP} \Rightarrow L \in \mathbf{NSPACE}(n)$. Ατοπο, καθώς υποθέσαμε ότι $L \notin \mathbf{NSPACE}(n)$. Αρα $\mathbf{NP} \neq \mathbf{NSPACE}(n)$. $\square$

Το παραπάνω αποτέλεσμα, αν και διαχωρίζει τις δύο κλάσεις πολυπλοκότητας, εντούτοις δε μας δίνει κανένα αποτέλεσμα για το εάν κάποια από αυτές περικλύει την άλλη ή όχι. Αυτό που μας λέει είναι ότι απλά είναι διαφορετικές! Οπως είπαμε, η κλάση πολυπλοκότητας γραμμικού χώρου χαρακτηρίζει τις γλώσσες με συμφραζόμενα, κάτι που εξ'άλλου θα αποδείξουμε καισ το επόμενο κεφάλαιο. Αρα το παραπάνω αποτέλεσμα στην ουσία διαχωρίζει τις γλώσσες με συμφραζόμενα με τις γλώσσες στο **NP**. Αρα είτε υπάρχουν γλώσσες στο **NP** με υπερ - γραμμικό πιστοποιητικό, είτε υπάρχουν γλώσσες με συμφραζόμενα οι οποίες να μην μπορούν να αποφασιστούν σε πολυωνυμικό χρόνο μη ντετερμινιστικά.

Η παρούσα διατριβή προσεγγίζει την πρώτη περίπτωση. Αρχικά παρουσιάζουμε γραμματικές με συμφραζόμενα για έναν αριθμό προβλημάτων στο **NP**. Τα προβλήματα αυτά έχουν την ιδιότητα ότι αποτελούν τα πιο δύσκολα και χαρακτηριστικά προβλήματα της κλάσης **NP**. Είναι **NP**-πλήρη. Αφού παρουσιάσουμε γραμματικές με συμφραζόμενα για αυτά, θα προσπαθήσουμε να ορίσουμε προβλήματα τα οποία χρειάζονται υπερ - γραμμικά πιστοποιητικά για να επιλυθούν σε πολυωνιμικό χρόνο μη ντετερμινιστικά.

[1]Το θεώρημα της Ιεραρχίας χώρου αναφέρει ότι υπάρχουν γλώσσες αποφασίσιμες σε χώρο $\mathcal{O}(f(n))$ αλλά όχι σε χώρο $o(f(n))$.

Κεφαλαιο 4

Γραμματικες Και Γλωσσες Με Συμφραζομενα

Μια *γραμματική με συμφραζόμενα* (context sensitive grammar) είναι μια τυπική γραμματική στην οποία όλοι οι κανόνες αριστερού μέλους ενός οποιουδήποτε κανόνα παραγωγής βρίσκονται ανάμεσα σε συμφραζόμενα. Ολοι οι κανόνες μιας γραμματικής με συμφραζόμενα είναι της μορφής $\alpha A\beta \to \alpha\gamma\beta$, το οποίο διασισθητικά σημαίνει ότι όποτε έχουμε το σύμβολο A με συμφραζόμενα α και β, τότε μπορούμε να το αντικαταστήσουμε με τη συμβολοσειρά τερματικών και μη τερματικών συμβόλων γ.

Η κλάση των γλωσσών που παράγονται από γραμματικές με συμφραζόμενα είναι γενικότερη από την κλάση των γλωσσών που παράγονται από γραμματικές *χωρίς* συμφραζόμενα και έτσι οι γραμματικές με συμφραζόμενα περιγράφουν ευρύτερη κλάση γλωσσών. Κάθε γραμματική χωρίς συμφραζόμενα είναι επί της ουσίας μια γραμματική με συμφραζόμενα στην οποία δεν λαμβάνουμε καθόλου υπ'όψιν τα συμφραζόμενα, δηλαδή στον παραπάνω κανόνα $\alpha A\beta \to \alpha\gamma\beta$, το A μπορεί να αντικατασταθεί με γ ανεξάρτητα με το τι τα α και β μπορεί να είναι, δηλαδή για τους κανόνες χωρίς συμφραζόμενα ισχύει ότι τα α και β είναι πάντα οι κενές συμβολοσειρές ϵ.

Η έννοια της γραμματικής με συμφραζόμενα ως παραγώγου γλωσσών, πρωτο παρουσιάστηκε από Νόαμ Τσόμσκι το 1956 ως ένας τρόπος περιγραφής του συντακτικού *φυσικών γλωσσών* στις οποίες πράγματι είναι συχνή η περίπτωση όπου η παρουσία ή όχι μιας λέξης σε συγκεκριμένο σημείο εντός μιας πρότασης, εξαρτάται από τις γειτόνικές λέξεις, τα *συμφραζόμενα*. Μια τυπική γλώσσα η οπόια περιγράφεται από μια γραμματική με συμφραζόμενα, είναι *γλώσσα με συμφραζόμενα*.

4.1 Τυπικός ορισμός γραμματικών με συμφραζόμενα

Πιο τυπικά, μια γραμματική με συμφραζόμενα ορίζεται ακριβώς με τον ίδιο τρόπο που ορίσαμε τις τυπικές γραμματικές μόνο που επιτρέπουμε μόνο κανόνες της μορφής $\alpha A\beta \rightarrow \alpha\gamma\beta$. Για πληρότητα, παραθέτουμε πιο κάτω τον νατίστοιχο ορισμό:

Ορισμός 14 *Μια γραμματική με συμφραζόμενα G είναι μια τετράδα $G = (V, \Sigma, R, S)$ όπου*

- *V είναι το σύνολο των μεταβλητών ή μη τερματικών συμβόλων,*
- *Σ είναι το σύνολο των τερματικών συμβόλων(το αλφάβητο που αντιστοιχεί στην γλώσσα την οποία παράγει η δεδομένη γραμματική),*
- *R είναι το σύνολο των κανόνων παραγωγής: $R \subseteq (V \cup \Sigma)^* V (V \cup \Sigma)^* \times (V \cup \Sigma)^*$ όπου ο κάθε κανόνας είναι της μορφής $\alpha A\beta \rightarrow \alpha\gamma\beta$ με $\alpha, \beta \in (V \cup \Sigma)^*$, $A \in V$ και $\gamma \in (V \cup \Sigma)^+$, δηλαδή το A είναι ένα απλό μη τερματικό σύμβολο, τα α, β είναι συμβολοσειρές (ενδεχομένως κενές) τερματικών και μη τερματικών συμβόλων ενώ γ είναι μια μη κενή συμβολοσειρά τερματικών και μη τερματικών συμβόλων.*
- *S είναι η αρχική μεταβλητή $\in V$.*

Επιπλέον επιτρέπεται και κανόνας της μορφής $S \rightarrow \epsilon$, όπου ϵ είναι η κενή συμβολοσειρά, δεδομένου ότι ο κανόνας S δεν εμφανίζεται σε κανένα δεξιό μέλος οποιουδήποτε κανόνα παραγωγής. Η προσθήκη της κενής συμβολοσειράς στο σύνολο αυτών των συμβολοσειρών που μπορεί να παράγει μια γραμματική με συμφραζόμενα, επιτρέπει να θεωρούμε τις γλώσσες που παράγονται από γραμματικές χωρίς συμφραζόμενα ως γνήσιο υποσύνολο των γλωσσών που παράγονται από γραμματικές με συμφραζόμενα, αντι να κάνουμε την ασθενέστερη δήλωση ότι όλες οι γραμματικές χωρίς συμφραζόμενα και χωρίς $\rightarrow \epsilon$ κανόνες, είναι επίσης γραμματικές με συμφραζόμενα.

Ο όρος *με συμφραζόμενα* σε κανόνες τις μορφής $\alpha A\beta \to \alpha\gamma\beta$ εξηγείται από τις συμβολοσειρές α και β οι οποίες και αποτελούν τα *συμφραζόμενα* της μεταβλητής A και καθορίζουν το κατά πόσο η μεταβλητή A μπορεί να αντικατασταθεί από την (μη κενή) συμβολοσειρά τερματικών και μη-τερματικών συμβόλων γ ή όχι. Αυτοί οι κανόνες είναι διαφορετικοί από τους κανόνων γραμματικών *χωρίς* συμφραζόμενα στους οποίους τα συμφραζόμενα μιας μεταβλητής δεν λαμβάνονται καθόλου υπ'όψιν.

Εναλλακτικά (και ισοδύναμα) μπορούμε να χαρακτηρίσουμε τους κανόνες μιας γραμματικής με συμφραζόμενα ως εξής: όποτε έχουμε έναν κανόνα $x \to y \in R$, τότε $|x| \leq |y|$. Η υποκείμενη γλώσσα που παράγεται από την γραμματική αυτή είναι μια γλώσσα με συμφραζόμενα. Ο ορισμός όμως αυτός είναι τελείως ισοδύναμος με τον εξής ορισμό που δώσαμε παραπάνω:

Θεώρημα 4 *Οι δύο παραπάνω διαφορετικοί ορισμοί της μορφής των κανόνων μιας γραμματικής με συμφραζόμενα είναι ισοδύναμοι.*

Απόδειξη: Για να το δούμε αυτό, έστω μια γραμματική στην οποία όλοι οι κανόνες είναι της μορφής $x \to y$ με $|x| \leq |y|$. Αυτό σημαίνει ότι όταν αντικαθιστούμε την συμβολοσειρα (τερματικών και μη-τερματικών συμβόλων) x με την συμβολοσειρά y, το μήκος της καινούριας συμβολοσειράς *δεν μπορεί να είναι μικρότερο από την προηγούμενη*. Με άλλα λόγια δεν μπορούμε να συρικνώσουμε συμβολοσειρές.

Αρχικά, θα περιγράψουμε έναν τρόπο για τη μετατροπή κανόνων της παραπάνω μορφής, δηλαδή της μορφής $x \to y$ με $|x| \leq |y|$, στην δεύτερη μορφή. Δηλαδή, δοθέντος ενός κανόνα $x \to y$ θα περιγράψουμε τη διαδικασία μετατροπής αυτή σε ένα ισοδύναμο σύνολο κανόνων οι οποίοι να είναι όλοι της μορφής $\alpha A\beta \to \alpha\gamma\beta$ που οι συμβολοσειρές α, β μπορεί να είναι κενές. Ο κανόνας $x \to y$ μπορεί να γραφτεί στη μορφή $x_1x_2 \ldots x_n \to y_1y_2 \ldots y_q$, $q \leq n$. Κάθε τερματικό σύμβολο των συμβολοσειρών x και y το αντικαθιστούμε με ένα νέο σύμβολο-μεταβλητή, το οποίο δεν υπάρχει στο σύνολο των μεταβλητών μας. Δηλαδή $\forall\vartheta \in \Sigma$ και το οποίο ανήκη στις τρέχουσες συμβολοσειρές συμβόλων x και y, προσθέτουμε και μιά νέα μεταβλητή Θ. Αυτές οι μεταβλητές εμφανίζονται μόνο στους κανόνες $X'_i \to x_i$.

Αφού έχουμε αντικαταστήσει όλα τα τερματικά σύμβολα με νέα μη τερματικά, προχωράμε στη δημιουργεία των νέων κανόνων οι οποίοι έχουν τη δεύτερη μορφή. Η νέα συμβολοσειρά τώρα έχει τη μορφή $X_1X_2 \ldots X_n \to Y_1Y_2 \ldots Y_q$ (όπου εδώ έχουμε αντικαταστήσει με κεφαλαίο γράμμα όλα τα σύμβολα της αρχικής συμβολοσειράς, ακόμα και αυτά που αντιστοιχούν σε μεταβλητές για να τονίσουμε ότι οι δύο νέες συμβολοσειρές είναι συμβολοσειρές μόνο μεταβλητών: των ήδη υπάρχουσων και των νέων που αντικαθιστούν τα τερματικά σύμβολα).

Το νέο σύνολο κανόνων που θα είναι ισοδύναμο με τον παραπάνω μονό κανόνα είναι το εξής:

$$\begin{aligned}
X_1X_2\ldots X_n &\rightarrow \Lambda^i_1X_2X_3\ldots X_n\\
\Lambda^i_1X_2 &\rightarrow \Lambda^i_1\Lambda^i_2\\
\Lambda^i_2X_3 &\rightarrow \Lambda^i_2\Lambda^i_3\\
\vdots &\rightarrow \vdots\\
\Lambda^i_{n-1} &\rightarrow \Lambda^i_n\Lambda^i_{n+1}\ldots\Lambda^i_k\\
\Lambda^i_1 &\rightarrow Y_1\\
\vdots &\rightarrow \vdots\\
\Lambda^i_k &\rightarrow Y_k
\end{aligned}$$

όπου για τη νεα μεταβλητή Λ^l_k η ερμηνεία των δεικτών k και l είναι για να μας δείχνει ότι έχουμε προσθέσει μια νέα μεταβλητή για το k-οστό τερματικό σύμβολο του l-οστού κανόνα.

Η ερμηνεία των παραπάνων κανόνων είναι μάλλον προφανής: "σπάμε" τον αρχικό κανόνα σε ένα σύνολο $n+k$ κανόνων βοηθούμενοι από τις νέες μεταβλητές Λ^l_k όπου έχουν το ρόλο των συμφραζομένων ώστε να έρθουν οι κανόνες στην επιθυμητή μορφή. □

Ενα παράδειγμα είναι διαφωτιστικό. Εστω ότι έχουμε τον κανόνα $\alpha B\gamma\Delta \rightarrow E\gamma BAz$ και έστω ότι αυτός είναι ο πρώτος κανόνας της γραμματικής μας. Ακολουθούμε τη παραπάνω "συνταγή" και προσθέτουμε τους νέους ισοδύναμους κανόνες:. Αρχικά, αντικαθιστούμε κάθε τερματικό σύμβολο με μια νέα μεταβλητή: ο κανόνας μετατρέπεται στη μορφή $A'B\Gamma'\Delta \rightarrow E\Gamma'B\Delta Z'$. Προσθέτουμε τους αρχικούς κανόνες $A' \rightarrow \alpha$, $\Gamma' \rightarrow \gamma$, $Z' \rightarrow z$. Επειτα, προσθέτουμε τους εξής κανόνες:

$$\begin{aligned}
A'B\Gamma'\Delta &\rightarrow \Lambda^1_1B\Gamma'\Delta\\
\Lambda^1_1B &\rightarrow \Lambda^1_1\Lambda^1_2\\
\Lambda^1_2\Gamma' &\rightarrow \Lambda^1_2\Lambda^1_3\\
\Lambda^1_3\Delta &\rightarrow \Lambda^1_3\Lambda^1_4\Lambda^1_5
\end{aligned}$$

καθώς και τους τελικούς κανόνες

$$\begin{aligned}
\Lambda_1^1 &\rightarrow E \\
\Lambda_2^1 &\rightarrow \Gamma' \\
\Lambda_3^1 &\rightarrow B \\
\Lambda_4^1 &\rightarrow A \\
\Lambda_5^1 &\rightarrow Z'
\end{aligned}$$

Ένας κανόνας που χρησιμοποιείται συνεχώς στη συνέχεια, ως μέρος της περιγραφής κανόνων για τα αντίστοιχα προβλήματα, είναι ο κανόνας της μορφής $aX \rightarrow Xa$ (ή $Xa \rightarrow aX$), κανόνας που προσομειώνει τι κίνηση του δρομέα της μηχανής προς τα αριστερά (προς τα δεξιά). Αυτός ο κανόνας άνήκει στη πρώτη μορφή. Ενα ισοδύναμο σύνολο κανόνων στην εναλλακτική μορφή είναι το εξής: προσθέτουμε ένα μη τερματικό σύμβολο A' για το τερματικό σύμβολο a του κανόνα αυτού. Δηλαδή έχουμε $aX \rightarrow A'X$. Τώρα, η μεταβλητή X με αριστερό συμφραζόμενο το σύμβολο A' μετατρέπεται σε a: $A'X \rightarrow A'a$. Τέλος, χρησιμοποιούμε τον κανόνα $A' \rightarrow X$ για να πάρουμε τη τελική μορφή $A'a \rightarrow Xa$.

Για την αντίστροφη περίπτωση, έστω μια γραμματική στην οποία όλοι οι κανόνες είναι της μορφής $\alpha A\beta \rightarrow \alpha\gamma\beta$ όπου A είναι μια μεταβλητή (μη τερματικό σύμβολο) ενώ $\gamma \in (V \cup \Sigma)^+$. Αυτό σημαίνει ότι η μεταβλητή A, όταν βρίσκεται μεταξύ συμφραζομένων α και β, μπορεί να αντικατασταθεί από την (μη κενή) συμβολοσειρά τερματικών και μη τερματικών συμβόλων γ. Δεδομένου ότι η συμβολοσειρά γ δεν είναι η κενή, το μήκος της συμβολοσειράς $\alpha\gamma\beta$ είναι *τουλάχιστον* όσο και το μήκος της $\alpha A\beta$ (είναι ίσο στην περίπτωση όπου η μεταβλητή A αντικαθίσταται με μια καινούργια μεταβλητή ή ένα τερματικό σύμβολο). Δηλαδή, βλέπουμε ότι οι συμβολοσειρές *δεν συρικνώνονται*. Αρα οι κανόνες είναι της μορφής $x \rightarrow y$ με $|x| \leq |y|$.

Παράδειγμα 3 *Εστω η γλώσσα* $\{a^n b^n c^n : n \geq 0\}$. *Η γλώσσα αυτή ως γνωστόν δεν είναι χωρίς συμφραζόμενα (κάτι το οποίο μπορεί σχετικά εύκολα να αποδειχτεί με την χρήση του λήμματος άντλησης για τις γλώσσες χωρίς συμφραζόμενα). Παρακάτω δίδεται σύνολο κανόνων με συμφραζόμενα που παράγει συμβολοσειρες αυτής της γλώσσας και μόνον αυτής. Αρα αυτή η λώσσα είναι με συμφραζόμενα:*

$$
\begin{aligned}
S &\rightarrow aXBC \\
S &\rightarrow aBC \\
X &\rightarrow aXBC \\
X &\rightarrow aBC \\
CB &\rightarrow BC \\
aB &\rightarrow ab \\
bB &\rightarrow bb \\
bC &\rightarrow bc \\
cC &\rightarrow cc
\end{aligned}
$$

4.2 Γραμμικώς Φραγμένα Αυτόματα

Ενα γραμμικώς φραγμένο αυτόματο ΓΦΑ (linear bounded automaton-LBA) είναι μια περιοριστική εκδοχή της μη-ντετερμινιστικής μηχανής Turing. Οπως και η κλασική μηχανή Turing, αυτό αποτελείται από μια ταινία από κελιά τα οποία μπορούν να περιέχουν σύμβολα από εάν πεπερασμένο αλφάβητο (το εκάστοτε αλφάβητο της τρέχουσας γλώσσας), μια κεφαλή η οποία μπορεί να διαβάσει από η να γράψει στα κελιά της ταινίας κάθε στιγμή και η οποία μπορεί να μετακινηθεί αριστερά ή δεξιά η να παραμείνει στη θέση της, και ένα πεπερασμένο αριθμό από καταστάσεις. Διαφέρει από την παραδοσιακή μηχανή Turing στο ότι, αν και αρχικά η ταινία μπορεί να θεωρηθεί ότι έχει απεριόριστο μήκος, τελικά μόνο ένα πεπερασμένο, και συνεχές, κομμάτι της ταινίας, το μήκος του οποίου είναι μια γραμμική συνάρτηση του αρχικού μήκους της εισόδου, είναι προσπελάσιμο από την κεφαλή εγγραφής/ανάγνωσης.

Ενα γραμμικώς φραγμένο αυτόματο με άλλα λόγια είναι μια μηχανή Turing με περιορισμένη ποσότητα μνήμης. Τέτοιες μηχανές μπορούν να επιλύσουν προβλήματα τέτοια τα οποία απαιτούν μνήμη η οποία μπορεί να χωρέσει μέσα στην ταινία που χρησιμοποιείται για την είσοδο. Χρησιμοποιώντας ένα αλφάβητο ταινίας διαφορετικό και μεγαλύτερο από το αλφάβητο των συμβολοσειρών εισόδου, μπορούμε να αυξήσουμε την διαθέσιμη μνήμη κατά έναν σταθερό ($\mathcal{O}(1)$) παράγοντα. Αρα, λέμε ότι για μιά συμβολοσειρά εισόδου μήκους n η ποσότητα της διαθέσιμης μνήμης μπορεί να γίνει καριβώς n.

4.2.1 Ισοδυναμία Γλωσσών με Συμφραζόμενα και Γραμμικώς Φραγμένων Αυτομάτων

Το 1964 ο Ιάπωνας μαθηματικός S. Kuroda απέδειξε κάτι το εντυπωσιακό: η κλάση των γλωσσών με συμφραζόμενα και η κλάση των γλωσσών που ημι-αποφασίζονται από γραμμικώς φραγμένα αυτόματα ταυτίζονται. Σε πιο τυπικούς όρους απέδειξε το παρακάτω θεώρημα:

Θεώρημα 5 *Μια γλώσσα L είναι γλώσσα με συμφραζόμενα εάν, και μόνον εάν, υπάρχει ένα γραμμικώς φραγμένο αυτόματο M τέτοιο ώστε να αποδέχεται την ίδια γλώσσα L, δηλαδή τέτοιο ώστε $L = L(M)$.*

Οπως πάντα, έχουμε να αποδείξουμε δύο κατευθύνσεις. Ξεκινάμε από την πιο απλή περίπτωση της προσομείωσης μιας υποκείμενης γραμματικής που παράγει συμβολοσειρές της γλώσσας L από ένα γραμμικώς φραγμένο αυτόματο:

Λήμμα 2 *Δοθείσας μιας γραμματικής $G = (V, \Sigma, R, S)$ τέτοια ώστε $L(G) = L$, υπάρχει γραμμικώς φραγμένο αυτόματο M το οποίο να αποδέχεται την ίδια γλώσσα.*

Απόδειξη: Εστω μια γραμματική $G = (V, \Sigma, R, S)$ τέτοια ώστε $L(G) = L$. Θα κατασκευάσουμε μια *μη ντετερμινιστική* μηχανή Turing M γραμμικώς φραγμένη, η οποία και θα αποδέχεται (ημιαποφασίζει) την γλώσσα που παράγεται από την γραμματική G, δηλαδή δοθήσας μιας συμβολοσειράς $w \in \Sigma^*$, εάν $x \in L$ τότε η μηχανή θα τερματίζει και θα αποδέχεται την συμβολοσειρά ενώ έαν $x \notin L$ τότε η μηχανή θα εκτελείται επ'άπειρον. Στην πραγματικότητα , η μηχανή που θα κατασκευάσουμε θα είναι *μη ντετερμινιστική*.

Η μηχανή M που θα κατασκευάσουμε αποτελείται από τρείς ταινίες. Στην πρώτη ταινία θα αποθηκεύουμε την είσοδο w την οποία και θα προσπαθούμε να ανακατασκευάσουμε στην δεύτερη ταινία. Ουσιαστικά, προσπαθούμε να ανακατασκευάσουμε μια παραγωγή της w από το αρχικό μη τερματικό σύμβολο S στην γραμματική G. Αρα, στο πρώτο βήμα της, η μηχανή M γράφει στην δεύτερη ταινία το αρχικό σύμβολο S. Στη συνέχεια, η μηχανή M συνεχίζει με βήματα τα οποία προσπαθούν να προσομειώσουν κάποια παραγωγή της συμβολοσειράς w, την οποία θέλουμε να ελένξουμε εάν ανήκει ή όχι στην υποκείμενη γλώσσα L. Σε κάθε βήμα, η μηχανή M κάνει μια μη ντετερμινιστική επιλογή: μεταξύ των $|R| + 1$ πιθανών κανόνων, επιλέγει κάποιον από αυτούς. Εστω ότι σε κάποιο βήμα ο κανόνας που μη ντετερμινιστικά επέλεξε η μηχανή είναι ο $x \to y$ ($|x| \leq |y|$ αφού πρόκειται για κανόνα μιας γραμματικής με συμφραζόμενα). Η μηχανή M τότε αρχίζει να σαρώνει από αριστερά προς τα δεξιά την ενδιάμεση συμβολοσειρά της παραγωγής οποία είναι αποθηκευμένη στην δεύτερη ταινία της. Σε κάποιο σημείο, μη ντετερμινιστικά, η μηχανή σταματάει τη

σάρωση και αρχίζει να ελέγχει το κατά πόσον οι επόμενοι $|u|$ χαρακτήρες (εάν αυτοί υπάρχουν) ταυτίζονται με τους χαρακτήρες της συμβολοσειράς u. Εάν ναι, τότε η μηχανή αντικαθιστά τα σύμβολα της u με αυτά της v δημιουργώντας κατάλληλο χώρο εάν αυτό χρειαστεί (εάν $|u| < |v|$) μετακινώντας την υπόλοιπη προς τα δεξιά συμβολοσειρά κατά τον κατάλληλο αριθμό χαρακτήρων. Εάν ο παραπάνω έλεγχος αποτύχει, τότε η μηχανή M εισέρχεται σε έναν ατέρμονο υπολογισμό: η συγκεκριμένη προσπάθεια παραγωγής της συμβολοσειράς w απέτυχε (αλλά εφόσον η μηχανή είναι μη ντετερμινιστική, αυτό δεν μας πειράζει: αρκεί μια ακολουθία μεταβάσεων που να παράγει τελικά την w ώστε να αποφανθούμε ότι η w ανήκει στην υποκείμενη γλώσσα).

Η επιλογή που αντιστοιχεί στην $(|R| + 1)$-οστή κατάσταση της μηχανής Turing M, απλά ελέγχει το κατά πόσον η συμβολοσειρά που είναι γραμμένη στη δεύτερη ταινία (εργασίας) είναι ή όχι η ίδια με την αποθηκευμένη συμβολοσειρά εισόδου στην πρώτη ταινία (μόνο ανάγνωσης). Εάν ναι, τότε ο υπολογισμός σταματάει και η μηχανή M αποδέχεται την συμβολοσειρά εισόδου w. Αυτό σημαίνει ότι η συμβολοσειρά w μπορεί πράγματι να παραχθεί από την γραμματική G. Εάν όμως ο έλεγχος αποτύχει, δηλαδή δεν βρεθούν ίσες οι δύο συμβολοσειρές προς σύγκριση, τότε η μηχανή M εισέρχεται σε έναν ατέρμονο υπολογισμό και ελπίζουμε κάποια άλλη παραγωγή να μας οδηγήσει στην αποδοχή της συμβολοσειράς w.

Είναι άμεσο από την παραπάνω κατασκευή ότι οι μόνοι δυνατοί υπολογισμοί που αναγκάζουν την μηχανή M να τερματίσει είναι εκείνοι που αντιστοιχούν σε μια σωστή παραγωγή της συμβολοσειράς εισόδου w από τους ανίστοιχους κανόνες της γραμματικής G. Συνεπώς, η μηχανή Turing M αποδέχεται την συμβολοσειρά εισόδου w εάν και μόνον εάν η w μπορεί να παραχθεί από κανόνες της γραμματικής με συμφραζόμενα G, ή, με άλλα λόγια, εάν $w \in L(G)$.

Το επόμενο βήμα για την απόδειξη του λήμματος είναι να βρούμε πόσος *χώρος* χρειάζεται ώστε να ολοκληρωθεί ένας οποιοσδήποτε σωστός υπολογισμός της μηχανής M όταν αυτή αποδέχεται μια συμβολοσειρά εισόδου $w \in \Sigma^*$ με μήκος $|w| = n$.

Ολοι οι κανόνες της γραμματικής G είναι με συμφραζόμενα που με τη σειρά του σημαίνει ότι όταν έχουμε μια παραγωγή $x \Rightarrow y$ σε ένα βήμα, τότε $|x| \leq |y|,\ x, y \in \Sigma^*$ (εξ'αιτίας της μορφής των κανόνων της γραμματικής). Σε κάθε διαδοχική παραγωγή, αντικαθίσταται μια υπο-συμβολοσειρά της x από μια άλλη υποσυμβολοσειρά *τουλάχιστον ίσου μήκους* και το αποτέλεσμα είναι η νέα συμβολοσειρά y. Αρα έστω η αλληλουχία παραγωγών, που αποθηκεύονται στην δεύτερη ταινία της μηχανής Turing M: $S \Rightarrow x_1 \Rightarrow x_2 \Rightarrow \cdots \Rightarrow x_k = w$ (δηλαδή μετά από $k + 1$ παραγωγές, έχουμε την τελική συμβολοσειρά w ξεκινώντας από την κενή συμβολοσειρά). Με βάση την προηγούμενη παρατήρηση έχουμε ότι $1 = |S| \leq |x_1| \leq |x_2| \leq \cdots \leq |x_k| = w$. Αρα, βλέπουμε ότι σε

κάθε υπολογισμό αποδοχής, η μηχανή Turing M δεν χρειάζεται περισσότερο χώρο απ'ότι το μήκος της τελικής συμβολοσειράς που θέλουμε να παράξουμε. Αρα $L \in \mathbf{NSPACE}(n)$ και επειδή η γλώσσα L είναι μια τυχαία γλώσσα με συμφραζόμενα, συνεπάγεται ότι όλες οι γλώσσες με συμφραζόμενα μπορούν να αναγνωριστούν από ένα γραμμικώς φραγμένο αυτόματο. □

Τώρα απομένει να αποδειξουμε το αντίστροφο ώστε να τελειώσει η απόδειξη του θεωρήματος:

Λήμμα 3 *Εστω ένα γραμμικώς φραγμένο αυτόματο M το οποίο αποδέχεται (ημι-αποφασίζει) μια γλώσσα $L \subseteq \Sigma^*$. Τότε μπορούμε να κατασκευάσουμε μια γραμματική $G = (V, \Sigma, R, S)$ τέτοια ώστε για κάθε κανόνα $r \in R : \ r \equiv x \rightarrow y$ να ισχύει ότι $|x| \leq |y|$ (δηλαδή μια γραμματική με συμφραζόμενα) που να παράγει την αντίστοιχη γλώσσα L.*

Πριν όμως προχωρήσουμε στην απόδειξη του δεύτερου μέρους του θεωρήματος, καλό είναι να περιγράψουμε τη γενική διαδικασία παραγωγής κανόνων που αντιστοιχούν στη λειτουργία μιας δεδομένης μηχανής Turing, και έπειτα να προσαρμόσουμε αυτή την ανάλυση στην περίπτωσή μας ούτως ώστε οι παραγόμενοι κανόνες που αντιστοιχούν στις μεταβάσεις της υποκείμενης μηχανής να έχουν την επιθυμητή ιδιότητα που χαρακτηρίζει τους κανόνες με συμφραζόμενα: να είναι *αυξητικοί*.

Ας θεωρήσουμε ότι έχουμε στη διάθεσή μας μια μη ντετερμινιστική μηχανή Turing $\mathcal{M}$ με αλφάβητο $\mathcal{S} = \{\sigma_1, \ldots, \sigma_k\}$ και με σύνολο καταστάσεων $\mathcal{Q} = \{q_1, \ldots, q_n\}$. Θα περιγράψουμε, σε γενικές γραμμές, πώς μπορούμε να προσομοιώσουμε τη λειτουργία της μηχανής αυτής $\mathcal{M}$ από ένα σύνολο κανόνων στο αλφάβητο $\mathcal{S}' = \sigma_1, \ldots, \sigma_k, q_{k_0}, \ldots, q_{k+n+1}, \triangleleft, \triangleright$ όπου οι δύο επιπλέον καταστάσεις είναι ουσιαστικά, και όπως θα φανεί στη συνέχεια, καταστάσεις αρχής και τέλους ενώ τα δύο νέα σύμβολα $\triangleright$ και $\triangleleft$ συμβολίζουν το δεξί και το αριστερό αντίστοιχα άκρο της παραγόμενης συμβολοσειράς.

Κάθε στάδιο υπολογισμού της μηχανής $\mathcal{M}$ περιγράφεται απόλυτα από τη τρέχουσα διαμόρφωση της μηχανής αυτής. Σκοπός μας είναι να "κωδικοποιήσουμε" κατάλληλα κάθε διαμόρφωση με μιά κατάλληλη συμβολοσειρά (λέξη) του προηγουμένου αλφάβητου της υποκείμενης γραμματικής μας.

Εστω για παράδειγμα ότι η τρέχουσα κατάσταση της μηχανής, η διαμόρφωσή της, είναι η εξής: $\sigma_2\sigma_5\sigma_0\ q_3\ \sigma_3\sigma_1$, δηλαδή στην ταινία είναι αποθηκευμένη η τρέχουσα συμβολοσειρά $\sigma_2\sigma_5\sigma_0\sigma_3\sigma_1$, η τρέχουσα κατάστασή μας είναι η q_3 και ο δρομέας διαβάζει το σύμβολο σ_3. Επιλέγουμε η κωδικοποίηση της συμβολοσειράς αυτής να είναι η $\triangleright\sigma_2\sigma_5\sigma_0\ q_3\ \sigma_3\sigma_1\triangleleft$, δηλαδή ουσιαστικά η συμβολοσειρά της υποκείμενης διαμόρφωσης μαζί με τους χαρακτήρες προσδιορίσμού

αρχής και τέλους της συμβοοσειράς . Στη γενική περίπτωση θα έχουμε συμβολοσειρές της μορφής $\triangleright u\ q_i\ v \triangleleft$ όπου $k+1 \leq i \leq k+n+1$ και $u, v \subseteq \mathcal{S}'^*$.

Στη συνέχεια θα δείξουμε πώς να συσχετίζουμε κατάλληλους κανόνες για κάθε μετάβαση της μηχανής μας: οι κανόνες αυτοί (μεταβάσεις) θα προσομοιώνουν τον αντίκτυπο που έχουν οι αντίστοιχες μεταβάσεις στις υποκείμενες συμβολοσειρές (παρακάτω, και χωρίς βλάβη της γενικότητας, θεωρούμε ότι όταν η μηχανή θέλει να γράψει ένα σύμβολο και να κινηθεί προς μιά κατευθυνση τότε πρώτα γράφει το επιθυμητό σύμβολο και στο επόμενο βήμα μετακινεί το δρομέα, άντί να τα κάνει και τα δύο σε ένα βήμα):

1. Για κάθε μετάβαση της μορφής $\delta(q, \sigma_i) = (p, \sigma_j, -)$ τότε έχουμε έναν κανόνα της μορφής $q\sigma_i \rightarrow p\sigma_j$.

2. Για κάθε μετάβαση της μορφής $\delta(q, \sigma_i) = (p, \rightarrow)$ τότε έχουμε τους εξής κανόνες: $q\sigma_i\sigma_j \rightarrow \sigma_i p\sigma_j, \ \ \forall \sigma_j \neq \sigma_i$ και επιπλέον τον κανόνα $q\sigma_i \triangleleft \sigma_i p \sqcup \triangleleft$ (αυτός ο τελευταίος κανόνας περιγράφει την επέκταση της ταινίας, δηλαδή της συμβολοσειράς, κατά ένα επιπλέον τετράγωνο προς τα δεξιά).

3. Για κάθε μετάβαση της μορφής $\delta(q, \sigma_i) = (p, \leftarrow)$ τότε έχουμε τους εξής κανόνες: $\sigma_j q\sigma_i \rightarrow p\sigma_j\sigma_i, \ \ \forall \sigma_j \neq \sigma_i$.

4. Επιπλέον προσθέτουμε κανόνες για την αρχή του υπολογισμού μας (το οποίο ουσιαστικά αποτελεί και το τέλος της διαδικασίας παραγωγής) και το τέλος αυτού (αρχή της παραγωγής). Για να εξαναγκάσουμε την παραγωγή να ξεκινήσει ακριβώς από εκεί που θα τελείωνε ένας υπολογισμός αποδοχής προσθέτουμε τον κανόνα $\triangleright q_{n+k+1} \triangleleft \rightarrow S$ (στην πραγματικότητα πρέπει να προσθέσουμε τον αντίστοροφο αυτού του κανόνα και όλων των παραπάνω, κάτι που εξηγούμε στη συνέχεια).

5. οποτε δεν υπάρχει μετάβαση για το ζέυγος κατάστασης-συμβόλου $q_i\sigma_j$, τότε προσθέτουμε στο σύνολο των κανόνων μας την παραγωγή $q_i\sigma_j q_{n+1}\sigma_j$. Ετσι, η κατάσταση q_{n+1} θεωρείται η κατάσταση τερματισμου.

6. Τέλος προσθέτουμε τους επόμενους κανόνες: $q_{n+1}\sigma_i \rightarrow q_{n+1}$, $q_{n+1} \triangleleft \rightarrow q_0 \triangleleft$ και $\sigma_i q_0 \rightarrow q_0$, δηλαδή καταστάσεις οι οποίες διαγράφουν το τμήμα της τελικής συμβολοσειράς.

Για να γίνουν πιο σαφείς όλοι οι παραπάνω κανόνες, ας θεωρήσουμε ένα παράδειγμα. Εστω ότι η μηχανή μας βρίσκεται στη διαμόρφωση $\sigma_3\ q_1\ \sigma_2\sigma_5$. Η διαμόρφωση αυτή κωδικοποιείται από τη συμβολοσειρά $\triangleright\sigma_3 q_1\sigma_2\sigma_5\triangleleft$. Ας θεωρήσουμε τώρα ότι η μηχανή μας περιέχει τη μετάβαση $\delta(q_1, \sigma_2) = (q_4, \sigma_5)$. Τότε υπάχει ο αντίστοιχος κανόνας $q_1\sigma_2 \rightarrow q_4\sigma_5$, άρα έχουμε την παραγωγή $\triangleright\sigma_3 q_1\sigma_2\sigma_5\triangleleft \Rightarrow \triangleright\sigma_3 q_4\sigma_5\sigma_5\triangleleft$. Η συμβολοσειρά στα δεξιά αντιστοιχεί στη

διαμόρφωση που θα είχαμε εάν εφαρμόζαμε την αντίστοιχη μετάβαση στην αρχική διαμόρφωση. Εστω τώρα ότι η μηχανή μας περιέχει τη μετάβαση $\delta(q_4, \sigma_5) = (q_3, \rightarrow)$. τότε έχουμε τον εξής κανόνα: $q_4\sigma_5\sigma_5 \rightarrow \sigma_5 q_3 \sigma_5$, έτσι ώστε $\triangleright\sigma_3 q_4\sigma_5\sigma_5\triangleleft \Rightarrow \triangleright\sigma_3\sigma_5 q_3\sigma_5\triangleleft$.

Το ότι αυτοί οι κανόνες προσομοιώνουν τον εκάστοτε υπολογισμό της μηχανής μας είναι αποτέλεσμα του παρακάτω θεωρήματος:

Θεώρημα 6 *Εστω $\mathcal{M}$ μιά μη ντετερμινιστική μηχανή Turing. Τότε για κάθε συμβολοσειρά u στο αλφάβητο της $\mathcal{M}$, η μηχανή $\mathcal{M}$ αποδέχεται τη συμβολοσειρά u εάν και μόνον εάν $\triangleright q_1 \sqcup u\triangleleft \Rightarrow^* \triangleright q_0\triangleleft$.*

Απόδειξη: Ας θεωρήσουμε αρχικά ότι η μηχανή μας αποδέχεται τη συμβολοσειρά u. Τότε, η αρχική διαμόρφωση της μηχανής είναι η $q_0 \sqcup u$ και μέσω αυτής, δυνητικά, θα φτάσει σε μια κατάσταση q_i και ο δρομέας θα διαβάζει ένα σύμβολο σ_j για τα οποία δεν υπάρχει καμία μετάβαση της συναρτήσεως δ. Αρα

$$\triangleright q_1 \sqcup u\triangleleft \Rightarrow^* \triangleright v q_i \sigma_j w\triangleleft \Rightarrow \triangleright v q_{n+1}\sigma_j w\triangleleft \Rightarrow^* \triangleright v q_0\triangleleft \Rightarrow^* \triangleright q_0\triangleleft$$

Ας θεωρήσουμε τώρα ότι η μηχανή $\mathcal{M}$ δεν αποδέχεται (απορρίπτει) τη συμβολοσειρά u. Τότε, από την αρχική διαμόρφωση $q_0 \sqcup u$ η μηχανή δε θα τερματίσει ποτέ.

Πράγματι, θεωρούμε $w_1 = \triangleright q_1 \sqcup u\triangleleft$ και επίσης ότι $w_1 \Rightarrow w_2 \Rightarrow \cdots \Rightarrow w_k$. Τότε, κάθε w_i, $1 \leq i \leq k$, είναι αναγκαίο να περιέχει κάποιο σύμβολο καταστάσεως q_j. Αρα, δεν είναι δυνατόν να παραχθεί με οποιονδήποτε τρόπο κάποια συμβολοσειρά η οποία να περιέχει την q_0 και άρα, δεν υπάρχει παραγωγή της συμβολοσειράς $\triangleright q_0\triangleleft$ από την w_1.

□

Για να ολοκληρωθεί η αποδειξή μας, είναι απαραίτητο να θεωρήσουμε όλους τους προαναφερθέντες κανόνες στην *αντίστροφη* διάταξή τους (δηλαδή έναν κανόνα $a \rightarrow b$ τον θεωρούμε ως $b \rightarrow a$). Αυτό γιατί, σκοπός των κανόνων είναι δυνητικά να φτιάξουν τη συμβολοσοειρά την οποία θέλουμε να δώσουμε ως είσοδο στη μηχανή και η οποία θα την αποδεχτεί ή όχι. Με άλλα λόγια, οι κανόνες μας κάνουν την αντίστροφη δουλεία από αυτή που κάνει η μηχανή. Οπως είδαμε, οι κανόνες φτιάχνουν εκείνες τις συμβολοσειρές τις οποίες αποδέχεται η υποκείμενη μηχανή. Οι κανόνες αυτοί, όταν τους θεωρησουμε όπως είπαμε στην αντίστροφη μορφή τους, προσομοιώνουν τις κινήσεις της μηχανής από το τέλος του υπολογισμού αυτής (αποδοχή) προς την αρχή (δημιουργεία της συμβολοσειράς αποδοχής).

Τώρα, θέλουμε να γυρίσουμε πίσω στην αρχική μας περίπτωση, αυτής δηλαδή στην οποία μας δίνεται ένα γραμμικώς φραγμένο M αυτόματο με το

υποκείμενο αλφάβητο Σ και μας ζητείται ένα σύνολο κανόνων που αποτελούν μια γραμματική με συμφραζόμενα Γ τέτοια ώστε $L(\Gamma) = L_\Sigma(M) - \epsilon$. Η κατασκευή των κανόνων αυτών θα είναι αντίστοιχη με την παραπάνω γενική κατασκευή, όμως τώρα η κωδικοποίηση θα πρέπει να είναι λίγο διαφορετική καθώς οι παραγωγές θα πρέπει να είναι απαραίτητα αυξανόμενου μήκους.

Στην παρακάτω συζήτηση, και χωρίς βλάβη της γενικότητας, θα θεωρούμε τις διαμορφώσεις του γραμμικώς φραγμένου αυτομάτου ως τριάδες της μορφής $(i, q, \triangleright w \triangleleft)$ όπου $0 \leq i \leq |w| + 1$ είναι η θέση του δρομέα, q είναι η τρέχουσα κατάσταση του αυτομάτου και $\triangleright w \triangleleft$, $w \in \Sigma^*$ είναι τα περιεχόμενα της ταινίας. Επιπλέον, κάνουμε τη σύμβαση η κατάσταση q_1 να είναι η αρχική κατάσταση, ενώ η κατάσταση $\tilde{q}$ η τελική κατάσταση. Αρχικά, θα θεωρούμε συμβολοσειρές $u \subseteq \Sigma^*$ μήκους τουλάχιστον δύο. Τέτοιες συμβολοσειρές μπορούν να γραφτούν στη μορφή $u = \alpha w \beta,\ \alpha, \beta \in \Sigma$. Σκοπός μας είναι να κωδικοποιήσουμε μία διαμόρφωση $(i, q, \triangleright \alpha w \beta \triangleleft)$ της μηχανής με μια συμβολοσειρά μήκους $|\alpha w \beta| = |w| + 2$. Για να μπορέσουμε να το κάνουμε αυτό, θεωρούμε πέντε διαφορετικές παραλλαγές γιά κάθε χαρακτήρα α του αλφαβήτου μας:

$$\alpha,\ \acute{\alpha},\ \grave{\alpha},\ \overleftarrow{\alpha},\ \overrightarrow{\alpha}$$

Η ερμηνεία των παραπάνων συμβόλων είναι η εξής:

$\acute{\alpha}$: ο χαρακτήρας α βρίσκεται στο αριστερό άκρο της συμβολοσειράς,

$\grave{\alpha}$: ο χαρακήρας α βρίσκεται στο δεξί άκρο της συμβολοσειράς

$\overleftarrow{\alpha}$: ο χαρακτήρας α βρίσκεται στο αριστερό άκρο της συμβολοσειράς, αλλά το σύμβολο το οποίο διαβάζεται από το δρομέα είναι το $\triangleright$, ένας χαρακτήρας προς τα αριστερά του α,

$\overrightarrow{\alpha}$: ο χαρακτήρας α βρίσκεται στο δεξί της συμβολοσειράς, αλλά το σύμβολο το οποίο διαβάζεται από το δρομέα είναι το $\triangleleft$, ένας χαρακτήρας προς τα δεξιά του α.

Τέλος, η τρέχουσα κατάσταση θα συμβολίζεται από έναν δείκτη στο σύμβολο που διαβάζει ο δρομέας τη δεδομένη χρονική στιγμή (π.χ. εάν ο δρομέας διαβάζει σ και βρισκόμαστε στη κατάσταση z τότε αυτό θα συμβολίζεται ως σ_z). Εάν, παρ'όλ'αυτά, το σύμβολο που διαβάζεται τη τρέχουσα χρονική στιγμή είναι $\triangleright$ ή $\triangleleft$, τότε ο δείκτης θα βρίσκεται στο διπλανό του σύμβολο, το οποίο, όπως είπαμε, θα χαρακτήρίζεται από ένα βέλος. Ετσι, εάν το αυτόματο έχει n καταστάσεις, τότε εισαγάγουμε $3(n+1) + 2n$ νέα σύμβολα για κάθε

χαρακτήρα $\alpha \in \Sigma$ (εδώ σημειώνουμε ότι οι χαρακτήρες $\overleftarrow{\alpha}$ και $\overrightarrow{\alpha}$ πάντοτε περιέχουν δείκτη καταστάσεως). Παρακάτω φαίνεται ένα παράδειγμα το οποίο κάνει την παραπάνω κωδικοποίηση πιο κατανοητή:

Διαμόρφωση	Κωδικοποίηση
$(3, q, \triangleright abcde \triangleleft)$	$\acute{a}bc_q d\grave{e}$
$(1, q, \triangleright abcde \triangleleft)$	$\acute{a}_q bcd\grave{e}$
$(5, q, \triangleright abcde \triangleleft)$	$\acute{a}bcd\grave{e}_q$
$(0, q, \triangleright abcde \triangleleft)$	$\overleftarrow{a}_q bcd\grave{e}$
$(6, q, \triangleright abcde \triangleleft)$	$\acute{a}bcd\overrightarrow{e}_q$

Τώρα, θα κατασκευάσουμε ένα σύνολο κανόνων G τέτοιο ώστε δοθέντων δύο διαφορετικών διαμορφώσεων γ, δ του γραμμικώς φραγμένου αυτομάτου M καθώς και των αντίστοιχων κωδικοποιήσεών τους, έστω $\bar{\gamma}, \bar{\delta}$ αντίστοιχα, να ισχύει ότι:

$$\gamma \vdash \delta \quad \Leftrightarrow \quad \bar{\gamma} \rightarrow_G \bar{\delta}$$

Θα ορίσουμε το σύνολο κανόνων G "κατασκευάζοντας" κατάλληλους κανόνες που αντιστοιχούν σε κάθε μετάβαση του αυτομάτου μας. Οι κατάλληλοι κανόνες αυτοί φαίνονται στον παρακάτω πίνακα:

Κανόνας μετάβασης του M	Κανόνας του συνόλου G
$\delta(q, a) = (p, b, -)$	$a_q \rightarrow b_p$
	$\acute{a}_q \rightarrow \acute{b}_p$
	$\grave{a}_q \rightarrow \grave{b}_p$
$\delta(q, a) = (p, a, \rightarrow)$	$a_q b \rightarrow ab_p$
	$\acute{a}_q b \rightarrow \acute{a}b_p$
	$a_q \grave{b} \rightarrow a\grave{b}_p$
	$\acute{a}_q \grave{b} \rightarrow \acute{a}\grave{b}_p$
	$\grave{a}_q \rightarrow \overrightarrow{a}_p$
$\delta(q, \triangleright) = (p, \triangleright, \rightarrow)$	$\overleftarrow{a}_q \rightarrow \acute{a}_p$
$\delta(q, a) = (p, a, \leftarrow)$	$ba_q \rightarrow b_p a$
	$b\grave{a}_q \rightarrow b_p \grave{a}$
	$\acute{b}a_q \rightarrow \acute{b}_p a$
	$\acute{b}\grave{a}_q \rightarrow \acute{b}_p \grave{a}$
	$\acute{a}_q \rightarrow \overleftarrow{a}_p$
$\delta(q, \triangleleft) = (p, \triangleleft, \leftarrow)$	$\overrightarrow{a}_q \rightarrow \grave{a}_p$

Από τη στιγμή που οι παραπάνω κανόνες προσομοιώνουν τη λειτουργία του αυτομάτου M με έναν προφανή τρόπο, βλέπουμε ότι το αυτόματο M θα

αποδεχτεί μια συμβολοσειρά $\alpha u\beta,\ u \in \Sigma^*, \alpha, \beta \in \Sigma$ μόνο στην περίπτωση που υπάρχει μια παραγωγή από την αρχική μας συμβολοσειρά $\acute{\alpha}_{q_1} u\grave{\beta}$ χρησιμοποιώντας το παραπάνω σύνολο κανόνων, προς μια συμβολοσειρά η οποία περιέχει τη κατάσταση $\tilde{q}$ σαν δείκτη ($\tilde{q}$, όπως είπαμε στη αρχή της συζήτησης, είναι η τελική κατάσταση). Επιπλέον αυτών, προσθέτουμε στο αλφάβητό μας ένα νέο αρχικό σύμβολο S και τους αντίστοιχους κανόνες εκκαθάρισης της τελικής συμβολοσειράς μας:

$$\alpha_{\tilde{q}} \to S, \quad \alpha S \to S, \quad S\alpha \to S$$

όπου $\alpha \in \{\alpha, \acute{\alpha}, \grave{\alpha}, \overleftarrow{\alpha}, \overrightarrow{\alpha}\},\ \forall \alpha \in \Sigma$. Από τη στιγμή που αυτοί οι κανόνες θα μετατρέψουν τις κωδικοποιήσεις των διαμορφώσεων οι οποίες περιέχουν τη τελική κατάσταση $\tilde{q}$ στο μοναδικό σύμβολο S, και επιπλέον από τη στιγμή που δεν υπάρχει άλλος τρόπος να βρεθούμε στο σύμβολο S χρησιμοποιώντας τους παραπάνω κανόνες, έχουμε το εξής αποτέλεσμα:

Λήμμα 4 *Το γραμμικώς φραγμένο αυτόματο M αποδέχεται τη συμβολοσειρά $\alpha u\beta,\ u \in \Sigma^*, \alpha, \beta \in \Sigma$, εάν και μόνον εάν*

$$\acute{\alpha}_{q_1} u\grave{\beta} \quad \Rightarrow_G^* \quad S$$

Τώρα, ορίζουμε ως R το σύνολο εκείνων των κανόνων οι οποίοι αποτελούν στους *αντίθετους* του παραπάνω συνόλου κανόνων. Τότε έχουμε:

Λήμμα 5 *Το γραμμικώς φραγμένο αυτόματο M αποδέχεται τη συμβολοσειρά $\alpha u\beta,\ u \in \Sigma^*, \alpha, \beta \in \Sigma$, εάν και μόνον εάν*

$$S \quad \Rightarrow_R^* \quad \acute{\alpha}_{q_1} u\grave{\beta}$$

Είμαστε πλέον έτοιμοι να ορίσουμε τη με συμφραζόμενα γραμματική μας, Γ. Τα τερματικά σύμβολα της γραμματικής αυτής θα είναι τα μέλη του συνόλου Σ. Οι μεταβλητές μας θα είναι δύο ειδών:

1. τα σύμβολα του αλφαβήτου του συνόλου κανόνων R τα οποία και δεν ανήκουν στο αλφάβητο Σ και
2. σύμβολα της μορφής $\dot{\alpha}\ \forall \alpha \in \Sigma$.

Τέλος, οι κανόνες της γραμματικής μας θα είναι οι κανόνες του συνόλου R μαζί με τους εξής κανόνες:

$$\acute{\alpha}_{q_1} \to \dot{\alpha}, \quad \dot{\alpha}\beta \to \alpha\dot{\beta}, \quad \dot{\alpha}\grave{\beta} \to \alpha\beta, \qquad \forall \alpha, \beta \in \Sigma$$

Προφανώς, κρίνοντας από τη μορφή των κανόνων (αυξανόμενου μήκους) είναι εύκολο να διαπιστώσουμε ότι η παραπάνω γραμματική είναι πράγματι με συμφραζόμενα. Επιπλέον, χρησιμοποιώντας το παραπάνω λήμμα, έχουμε ότι:

Λήμμα 6 *Εστω $w \in \Sigma^*$ μια συμβολοσειρά. Τότε $w \in L(\Gamma)$ εάν και μόνον εάν $|w| \geq 2$ και $w \in L(M)$.*

Εστω τώρα M ένα γραμμικώς φραγμένο αυτόματο και Γ η γραμματική με συμφραζόμενα που μόλις κατασκευάσαμε. Τότε, από το παραπάνω λήμμα, έχουμε ότι

$$L(M) = L(\gamma) \cup L_0$$

όπου L_0 είναι το σύνολο των συμβολοσειρών w έτοιες ώστε $|w| \leq 2$. Το σύνολο όμως L_0 είναι πεπερασμένο, δηλαδή αντιστοιχή σε πεπερασμένη γλώσσα. Είναι όμως γνωστό ότι όλες οι πεπερασμένες γλώσσες είναι κανονικές και κατ'επέκταση με συμφραζόμενα, καθώς οι κανονικές γλώσσες είναι γνήσιο υποσύνολο των γλωσσών με συμφραζόμενα (reference here). Αρα, η γλώσσα $L(M)$ ως ένωση γλωσσών με συμφραζόμενα, είναι και η ίδια γλώσσα με συμφραζόμενα καθώς, όπως θα δείξουμε σε επόμενη ενότητα, οι γλώσσες με συμφραζόμενα είναι κλοιστές ως προς την ένωση.

4.3 Προβλήματα απόφασης σχετικά με γλώσσες με συμφραζόμενα

Στην προηγούμενη ενότητα είδαμε ότι, ατίθετα με την περιορισμένη μνήμη τους, τα γραμμικώς φραγμένα αυτόματα είναι αρκετά ισχυρά καθώς ημι - αποφασίζουν την κλάση γλωσσών με συμφραζόμενα, μια αρκετά ευρεία κλάση. Στην παρούσα ενότητα θα αποδείξουμε μερικά πρωταρχικά αποτελέσματα σχετικά με τις ικανότητες ή μη τέτοιων μηχανών.

Ας θυμηθούμε κάποιες ιδιότητες των μηχανών Turing. Οταν κάποιος μας δώσει μια τέτοια μηχανή και μια είσοδο x για τον "αλγόριθμο" που υλοποιεί αυτή η μηχανή, ένα απολύτος φυσιολογικό και σημαντικό ερώτημα είναι το κατά πόσον αυτή η μηχανή θα τερματίσει ποτέ *με την συγκεκριμένη είσοδο.*

Το πιο απλό πράγμα που έχουμε να κάνουμε είναι να "τρέξουμε" την μηχανή με την είσοδο αυτή και να δούμε εάν τερματίζει ποτέ. Ομως, αν και όταν η μηχανή τερματίσει είμαστε σε σε θέση να απαντήσουμε το ερώτημα, από την άλλη πλευρά, εάν η μηχανή δεν τερματίζει δεν μπορούμε να πούμε τίποτα για το κατά πόσον θα τερματίσει ή όχι στο μέλλον. Με άλλα λόγια, αποδεικνύεται ότι το ανίστοιχο πρόβλημα απόφασης για μηχανές Turing είναι *μη επιλύσιμο*. Δεν μπορούμε να βρούμε αλγόριθμο ο οποίος δοθείσας μιας μηχανής Turing M και μιάς εισόδου x για την μηχανή αυτή, να μπορεί να μας πεί, έστω και σε υπερ-εκθετικό χρόνο, εάν η μηχανή θα τερματίσει ποτέ με την είσοδο x.

Από την άλλη μεριά, το αντίστοιχο πρόβλημα, όταν περιορίζουμε το μοντέλο υπολογισμού σε γραμμικώς φραγμένα αυτόματα, γίνεται επιλύσημο. Παρακάτω θα περιγράψουμε μια διαδικασία η οποία, δοθέντως ενός γραμμικώς φραγμένου αυτομάτου M και μιας συμβολοσειράς εισόδου w, είναι σε θέση να προσδιορίσει έαν η μηχανή M θα τερματίσει ποτέ με είσοδο την συμβολοσειρά αυτή w:

Θεώρημα 7 *Δοθέντος ενός γραμμικώς φραγμένου αυτομάτου M και μιας συμβολοσειράς εισόδου w, το πρόβλημα απόφασης του κατά πόσον η μηχάνή M τερματίζει ποτέ με την συγκεκριμένη είσοδο w, είναι αλγοριθμικά επιλύσιμο (δηλαδή είναι αποφασίσιμο).*

Για την απόδειξη του προηγουμένου θεωρήματος, πολύ χρήσιμο είναι το επόμενο λήμμα, το οποίο μας δίνει ένα άνω φράγμα στον αριθμό των διαφορετικών διαμορφώσεων που μπορεί να έχει ενα γραμμικώς φραγμένο αυτόματο.

Λήμμα 7 *Εστω $M = (Q, \Sigma, \Gamma, \delta, s)$ ένα γραμμικώς φραγμένο αυτόματο. Ο αριθμός των διαφορετικών διαμορφώσεων του αυτομάτου αυτού, όταν αυτό λειτουργεί σε μιά είσοδο w με μήκος $|w| = n$, είναι $|Q||w||\Sigma|^{|w|}$.*

Απόδειξη: Ας θυμηθούμε ότι μια διαμόρφωση μιας μηχανής Turing M είναι ουσιαστικά μια απεικόνιση του υπολογισμού της μηχανής σε κάποιο συγκεκριμένο βήμα αυτής. Μια τέτοια διαμόρφωση, όπως την ορίσαμε, αποτελείται από την κατάσταση στην οποία βρίσκεται ο πεπερασμένος έλεγχος, από τη θέση στην οποία βρίσκεται ο δρομέας της ταινίας την τρέχουσα χρονική στιγμή και, τέλος, από τα περιεχόμενα της ταινίας. Εν προκειμένω, η μηχανή M έχει Q δυνατές καταστάσεις, το μήκος της συμβολοσειράς εισόδου είναι n (χωρίς να θεωρούμε ως μέρος της συμβολοσειράς εισόδου τους δύο τερματικούς χαρακτήρες της μηχανής $\triangleright$ και $\triangleleft$) οπότε η κεφαλή της μηχανής μπορεί να βρίσκεται σε n διαφορετικές θέσεις και επιπλέον, οι πιθανές συμβολοσειρές εισόδου μήκους n που μπορούν να εμφανιστούν στην ταινία, με βάση το αλφάβητο Σ, είναι $|\Sigma|^{|w|}$. Το γινόμενο όλων αυτών των τριών ποσοτήτων αποτελεί και τον

αριθμό των διαφορετικών διαμορφώσεων που μπορεί να βρεθεί το γραμμικώς φραγμένο αυτόματο M με ταινία μεγέθους $|n|$. □

Τώρα μπορούμε να γυρίσουμε στην απόδειξη του προηγούμενου θεωρήματος:

Απόδειξη Θεωρήματος 7: Με σκοπό να αποφασίσουμε το κατά πόσον το γραμμικώς φραγμένο αυτόματο M αποδέχεται ή όχι την συμβολοσειρά εισόδου w, προσομοιώνουμε την μηχανή M με είσοδο w. Δηλαδή κατασκευάζουμε μια νέα μηχανή Turing M' η οποία "τρέχει" την αρχική μηχανή M. Κατά τη διάρκεια της προσομοίωσης, εάν η μηχανή M τερματίσει, ανεξάρτητα από το εάν αποδέχεται ή απορρίπτει την συμβολοσειρά εισόδου w, τότε η M' τερματίζει και αυτή με τη σειρά της στην κατάσταση "ναι". Δηλαδή το συγκεκριμένο γραμμικώς φραγμένο αυτόματο μαζί μα τη συγκεκριμένη συμβολοσειρά εισόδου w αποτελούν ένα "ναι" στιγμιότυπο του προβλήματος απόφασης τοθυ κατά πόσον ένα γραμμικώς φραγμένο αυτόματο αποδέχεται μια συγκεκριμένη συμβολοσειρά εισόδου w. Αυτό που απομένει είναι να δείξουμε το πώς μπορούμε να ανιχνεύουμε άπειρους βρόγχους ώστε να τερματίσουμε τη λειτουργία της μηχανής M' σε κατάσταση απόρριψης.

Η ιδέα ανίχνευσης της άπειρης ανακύκλωσης είναι η εξής: καθώς η μηχανή (γραμμικώς φραγμένο αυτόματο) M κάνει υπολογισμούς με βάση τη συμβολοσειρά εισόδου w, μεταβαίνει από διαμόρφωση σε διαμόρφωση. Εάν ποτέ η μηχανή M επαναλάβει κάποια διαμόρφωση, τότε θα την επαναλαμβάνει συνέχεια και άρα θα βρίσκεται σε άπειρη ανακύκλωση. Επειδή όμως η μηχανή μας είναι γραμμικώς φραγμένη, η ποσότητα διαθέσιμης ταινίας είναι και αυτή φραγμένη (ακριβώς όσο και το μέγεθος της συμβολοσειράς εισόδου). Από το παραπάνω λήμμα, έχουμε ότι το αυτόματο αυτό θα έχει ένα πεπερασμένο πλήθος διαφορετικών διαμορφώσεων. Αρα, ένα πολύ συγκεκριμένο χρονικό περιθώριο είναι διαθέσιμο στη μηχανή μας ώστε να τελειώσει τον υπολογισμό της. Εάν κατά τον υπολογισμό αυτόν, η μηχανή ξεπεράσει σε αριθμό βημάτων το όριο των διαμορφώσεων που υποδεικνύει το παραπάνω λήμμα, τότε είμαστε σίγουροι ότι σε κάποιο σημείο έχει επαναλάβει κάποια διαμόρφωση και άρα βρίσκεται σε ανακύκλωση του υπολογισμού της. Δηλαδή, το μόνο που έχουμε να κάνουμε για να εντοπίσουμε τέτοιες ανακυκλώσεις, είναι να προσομειώνουμε τη μηχανή μας για ένα συγκεκριμένο αριθμό βημτάτων - αυτών που υποδυκνύει το παραπάνω λήμμα.

□

Το θεώρημα 7 μας δείχνει ότι οι μηχανές Turing και τα γραμμικώς φραγμένα αυτόματα διαφέρουν με έναν σημαντικό τρόπο: ενώ για τις μηχανές Turing το προηγούμενο πρόβλημα απόφασης είναι μη αποφασίσιμο, εντούτοις, για τα γραμμικώς φραγμένα αυτόματα είναι αποφασίσιμο. Από την άλλη μεριά,

αυτό δεν σημαίνει ότι οι δύο αυτές υπολογιστικές μηχανές δεν έχουν ομοιότητες. Για παράδειγμα, το επόμενο θεώρημα δείχνει ότι το να αποφασίσουμε το κατά πόσον η γλώσσα που αντιστοιχεί σε κάποιο γραμμικώς φραγμένο αυτόματο είναι κενή ή όχι, είναι μη επιλύσιμο, όπως και στην περίπτωση των γενικών μηχανών Turing.

Θεώρημα 8 *Δοθέντος ενός γραμμικώς φραγμένου αυτομάτου M, αληθεύει ότι η υποκείμενη γλώσσα που αποφασίζει το αυτόματο αυτό είναι κενή;*

Για την απόδειξη του παραπάνω θεωρήματος, σημαντικό ρόλο παίζει η έννοια του *ιστορικό υπολογισμών αποδοχής*:

Ορισμός 15 *Εστω M μια μηχανή Turing και w μια συμβολοσειρά εισόδου για την μηχανή αυτή. Ενα ιστορικό υπολογισμού αποδοχής για την μηχανή M με είσοδο w είναι μια ακολουθία διαμορφώσεων $\vartheta_1, \vartheta_2, \ldots, \vartheta_z$ όπου ϑ_1 είναι η αρχική διαμόρφωση της μηχανής M με είσοδο w, ϑ_z είναι μια διαμόρφωση αποδοχής (της συμβολοσειράς w) και, για κάθε $\vartheta_i : \vartheta_{i-1} \vdash \vartheta_i, 1 < i \leq z$. Ενα ιστορικό υπολογισμού απόρριψης ορίζεται ανάλογα, μόνο που τώρα η διαμόρφωση ϑ_z είναι μια διαμόρφωση απόρριψης της συμβολοσειράς εισόδου w.*

Τώρα, επιστρέφουμε στην απόδειξη του θεωρήματος.

Απόδειξη: Η απόδειξη θα γίνει με μιά απλή αναγωγή ενός αντίστοιχου προβλήματος απόφασης για τις μηχανές Turing, στην περίπτωση των γραμμικώς φραγμένων μηχανών Turing.

Εστω A_{TM} και $E_{\Gamma\Phi A}$ τα αντίστοιχα προβλήματα απόφασης για τις μηχανές Turing και για τα γραμμικώς φραγμένα αυτόματα αντίστοιχα. Πιο συγκεκριμένα, ως A_{TM} ορίζουμε το πρόβλημα απόφασης στο οποίο, δοθείσας μιας μηχανής Turing M και μιας συμβολοσειράς εισόδου w, ζητείται να αποφασίσουμε εάν $w \in L(M)$. Αυτό το πρόβλημα είναι ένα από τα πιο γνωστά μη επιλύσημα προβλήματα. Θα δείξουμε ότι εάν μπορούσαμε να επιλύσουμε (ουσιαστικά να αποφασίσουμε την υποκείμενη γλώσσα) το πρόβλημα $E_{\Gamma\Phi A}$, τότε θα μπορούσαμε να αποφασίσουμε και το πρόβλημα A_{TM}.

Δοθείσας μιας μηχανής Turing M και μιας συμβολοσειράς εισόδου w, θα δείξουμε ότι μπορούμε να αποφανθούμε κατά πόσον η μηχανή M αποδέχεται την συμβολοσειρά w, κατασκευάζοντας ένα γραμμικώς φραγμένο αυτόματο Φ και μετά ελέγχοντας κατά πόσον $L(\Phi) = \emptyset$ με τον αλγόριθμο που ήδη υποθέσαμε ότι υπάρχει. Το γραμμικώς φραγμένο αυτόματο Φ που θα κατασκευάσουμε, θα αναγνωρίζει την γλώσσα που αποτελείται από όλα ιστορικά υπολογισμών αποδοχής για την μηχανή M σε είσοδο w. Προφανώς, εάν η

μηχανή M απορρίπτει την συμβολοσειρά εισόδου w, τότε $L(\Phi) = \emptyset$. ε διαφορετική περίπτωση η γλώσσα $L(\Phi)$ περιέχει τουλάχιστον μια συμβολοσειρά: αυτή που αντιστοιχεί σε κάποιο ιστορικό υπολογισμού αποδοχής της M σε είσοδο w. Με άλλα λόγια, εάν μπορούμε να αποφανθούμε κατά πόσον $L(\Phi) = \emptyset$, τότε είναι ξεκάθαρο ότι το πρόβλημα E_{TM} είναι επιλύσιμο.

Τώρα, πρέπει να παριγράψουμε πώς το γραμμικώς φραγμένο αυτόματο Φ κατασκευάζεται. Θα περιγράψουμε την κατασκευή αυτού, δοθέντος μιας μηχανής Turing M και μιας συμβολοσειράς εισόδου w και δεν θα αρκεστούμε απλά στην απόδειξη ύπαρξης αυτού. Παρ'όλα αυτά, σκοπός δεν είναι να "τρέξουμε" το αυτόματο, αλλά να εκμεταλευτούμε την περιγραφή αυτού του αυτομάτου, η οποία και θα αποτελεί είσοδο στον υποτιθέμενο αλγόριθμο ο οποίος θα ελέγχει κατά πόσον $L(\Phi) = \emptyset$. Οταν ο αλγόριθμος αυτός αποφασίσει το κατά πόσον $L(\Phi) = \emptyset$ ή όχι, εμείς απλά αντιστρέφουμε την απόδειξη και με αυτόν τον τρόπο έχουμε κατασκευάσει έναν αλγόριθμο που αποφασίζει το προηγούμενο πρόβλημα για γενικές μηχανές Turing. Ατοπο, καθώς το πρόβλημα αυτό είναι μη επιλύσημο.

Κατασκευάζουμε το γραμμικώς φραγμένο αυτόματο Φ έτσι ώστε αυτό να αποδέχεται τη συμβολοσειρά εισόδου y εάν η y κωδικοποιεί ένα (κάποιο) ιστορικό υπολογισμού αποδοχής της μηχανής M σε είσοδο w. Οπως είπαμε, ένα ιστορικό υπολογισμού αποδοχής δεν είναι τίποτε άλλο από μια ακολουθία διαμορφώσεων $\vartheta_1, \vartheta_2, \ldots, \vartheta_z$ στις οποίες μεταβαίνει διαδοχικά η μηχανή M όταν ο υπολογισμός αυτής οδηγεί στην αποδοχή της συμβολοσειράς εισόδου w. Θεωρούμε ότι αυτή η ακολουθία διαμορφώσεων κωδικοποιείται από μια απλή συμβολοσειρά, με τις διαμορφώσεις να διαχωρίζονται από κάποιον χαρακτήρα διαχωρισμού ο οποίος δεν ανήκει στο αλφάβητο της μηχανής M ούτε σε αυτή του αυτομάτου Φ (σχήμα 4.3).

Σχήμα 4.1: Μια πιθανή είσοδος στο αυτόματο Φ.

Το γραμμικώς φραγμένο αυτόματο Φ δουλεύει με τον εξής τρόπο: όταν δέχεται ως είσοδο μια συμβολοσειρά y, το Φ πρέπει να αποδεχτεί την y εάν αυτή είναι μια συμβολοσειρά η οποία αντιστοιχεί σε ιστορικό υπολογισμού αποδοχής της μηχανής M σε είσοδο w. Αρχικά, το Φ ανακτά τις συμβολοσειρές που υποτείθεται ότι κωδικοποιούν διαμορφώσεις, αφαιρώντας τους χαρακτήρες διαχωρισμού ♯. Ετσι το Φ έχει στη διάθεσή του τις συμβολοσειρές $\vartheta_1, \vartheta_2, \ldots, \vartheta_z$ για τις οποίες είναι αναγκαίο να ελένξει τρείς συνθήκες:

▷ Η συμβολοσειρά ϑ_1 αντιστοιχεί στην αρχική διαμόρφωση της μηχανής

M σε είσοδο w.

▷ Η ϑ_z είναι μια διαμόρφωση αποδοχής (της συμβολοσειράς w).

▷ Για κάθε $\vartheta_i : \vartheta_{i-1} \vdash \vartheta_i, 1 < i \leq z$.

Οι δύο πρώτες συνθήκες είναι πολύ εύκολο να ελεγθούν: στην πρώτη περίπτωση, τοα υτόματο Φ απλά ελέγχει εάν η συμβολοσειρά ϑ_1 ταυτίζεται με την συμβολοσειρά $s\,\sigma_1, \ldots, \sigma_n$ όπου $\sigma_1, \ldots, \sigma_n = w$ και s η αρχική κατάσταση της μηχανής Turing M σε είσοδο w. Για την δεύτερη περίπτωση, το μόνο που έχει να κάνει το αυτόματο Φ είναι να ψάξει εάν στη τελευταία διαμόρφωση ϑ_z υπάρχει η κατάσταση αποδοχής q_{accept} της συμβολοσειράς εισόδου w από τη μηχανή M.

Η τρίτη περίπτωση είναι και αυτή που είναι δυσκολότερο, τουλάχιστον όχι προφανές, να ελεγθεί. Για κάθε ζεύγος γειτονικών διαμορφώσεων $\vartheta_{i-1}, \vartheta_i, 1 < i \leq z$, το αυτόματο Φ ελέγχει το κατά πόσον $\vartheta_{i-1} \vdash \vartheta_i$. Αυτό γίνεται επαληθεύοντας ότι οι δύο γειτονικές διαμορφώσεις είναι ταυτόσημες εκτός ίσως από τα σημεία κάτω και εκατέροθεν της κεφαλής στην πρώτη διαμόρφωση ϑ_{i-1}. Εάν διαφέρουν, θα πρέπει οι διαφορές να αντικατοπτρίζουν τους κανόνες μετάβασης που συμφωνούν με την συνάρτηση μετάβασης της μηχανής M. Τότε, το αυτόματο Φ ελέγχει ότι η μετάβαση είναι έγκυρη, παλινδρομώντας μεταξύ των αντίστοιχων θέσεων ελέγχου και σημαδεύοντας κάθε φορά το τρέχων σύμβολο ελέγχου.

Τέλος, εάν και οι τρείς συνθήκες επαληθεύονται, το αυτόματο Φ αποδέχεται την είσοδό του.

Εστω τώρα ο υποτιθέμενος αλγόριθμος ο οποίος και ελέγχει κατά πόσον $L(\Phi) = \emptyset$ ή όχι. Ο δέχεται ως είσοδο την περιγραφή του αυτομάτου Φ και αποφασίζει εάν η υποκείμενη γλώσσα που αποφασίζει το Φ είναι κενή ή όχι. Εάν $L(\Phi) = \emptyset$, η μηχανή M δεν έχει κανέναν υπολογισμό αποδοχής της συμβολοσειράς εισόδου w. Αρα $w \notin L(M)$. Αντίθετα, εάν $L(\Phi) \notin \emptyset$ αυτό σημαίνει ότι υπάρχει μια συμβολοσειρά η οποία κωδικοποιεί ενα ιστορικό υπολογισμού αποδοχής της συμβολοσειράς w από τη μηχανή M. Αρα $w \in L(M)$. Ατοπο, καθώς το αντίστοιχο πρόβλημα απόφασης είναι μη επιλύσιμο. □

4.4 Ιδιότητες κλειστότητας γλωσσών με συμφραζόμενα

Στην ενότητα αυτή θα παρουσιάσουμε και θα αποδείξουμε αποτελέσματα κλειστότητας σε σχέση πάντα με τις γλώσσες με συμφραζόμενα. Οι τρεις πιο βασικές ιδιότητες, και με τις οποίες θα ασχοληθούμε, αφορούν τις "προφανείς" λειτουργίες: τομή, ένωση και συμπλήρωμα.

Θεώρημα 9 *Εστω L_1 και L_2 δύο γλώσσες με συμφραζόμενα. Τότε η γλώσσα $L' = L_1 \cup L_2$ είναι και αυτή γλώσσα με συμφραζόμενα.*

Απόδειξη: Εστω $L_1 = L(\Gamma_1)$ και $L_2 = L(\Gamma_2)$ όπου Γ_1, Γ_2 είναι οι αντίστοιχες γραμματικές με συμφραζόμενα οι οποίες έχουν ξένα μεταξύ τους σύνολα τερματικών συμβόλων και μεταβλητών. Εστω S_1 και S_2 να είναι οι δύο αρχικές μεταβλητές που αρχίζουν την εκάστοτε παραγωγή των δύο γραμματικών. Θεωρούμε μια νέα γραμματική, Γ, η οποία έχει ως σύνολο τερματικών συμβόλων την ένωση των αντίστοιχων συνόλων των δύο γραμματικών και αντίστοιχα για τις μεταβλητές (την ένωση των δύο αντίστοιχων συνόλων μεταβλητών). Η νέα αυτή γραμματική περιέχει την ένωση των κανόνων των δύο γραμμτικών. Προσθέτουμε μια νέα μεταβλητή, την S, και τον νέο κανόνα $S \rightarrow S_1 \mid S_2$. Είναι προφανές ότι $L(\Gamma) = Ł(\Gamma_1) \cup L(\Gamma_2) = L_1 \cup L_2$. □

Αφού αποδείξαμε την κλειστότητα των γλωσσών με συμφραζόμενα ως προς την ένωση, τώρα θεωρούμε την τομή γλωσσών με συμφραζόμενα. Σε αντίθεση με τις γλώσσες χωρίς συμφραζόμενα οι οποίες δεν είναι κλειστές ως προς την τομή, στην περίπτωση των γλωσσών με συμφραζόμενα θα αποδείξουμε το ακόλουθο αποτέλεσμα:

Θεώρημα 10 *Εστω L_1 και L_2 δύο γλώσσες με συμφραζόμενα. Τότε η γλώσσα $L' = L_1 \cap L_2$ είναι και αυτή γλώσσα με συμφραζόμενα.*

Απόδειξη: Εστω $L_1 = L(M_1)$ και $L_2 = L(M_2)$ οι γλώσσες δύο γραμμικώς φραγμένων αυτομάτων M_1 αι M_2 αντίστοιχα. Η γενική ιδέα της αποδειξης είναι μάλλον και η προφανής, όμως παρουσιάζει κάποια δυσκολία: για να ελένξουμε το κατά πόσον μια συμβολοσειρά w ανήκει και στις δύο γλώσσες (αυτόματα) ελέγχουμε αρχικά εάν το αυτόματο M_1 αποδέχεται τη συμβολοσειρά και, εάν ναι, στη συνέχεια ξαναελέγχουμε το κατά πόσο και το δεύτερο αυτόματο την αποδέχεται. Η κύρια δυσκολία είναι ότι το αυτόματο M_1, κατά τη διάρκεια της λειτουργίας του, μπορεί να καταστρέψει την είσοδο, δηλαδή τη συμβολοσειρά w. Επειδή διαχειριζόμαστε γραμμικώς φραγμένα αυτόματα, το μέγεθος της διαθέσιμης μνήνης είναι περιορισμένο (γραμμικό) και άρα δε μπορούμε να έχουμε δύο αντιγραφα της συμβολοσειράς w σε παράθεση ώστε να αξιοποιηθεί το καθένα από το αντίστοιχο αυτόματο. Η λύση βασίζεται στην εξής ιδέα: θεωρούμε ότι η ταινία μας είναι χωρισμένη σε "ζώνες" ή "τομείς", στη περίπτωσή μας δύο ζώνες. Το νέο γραμμικώς φραγμένο αυτόματο M που θα κατασκευάσουμε θα δουλεύει ως εξής:

1. Αρχικά, θα αντιγράφει την είσοδο σε αυτό έτσι ώστε αυτή να εμφανίζεται και στις δύο ζώνες της ταινίας. Για να τις ξεχωρίζουμε, από εδώ και στο εξής, θα αναφερόμαστε στην άνω και στην κάτω ζώνη.

2. Επειτα, το νέο αυτόματο M θα προσομοιώσει το αρχικό αυτόματο M_1 δουλεύοντας στην άνω ζώνη.

3. Εάν το αυτόματο M_1 αποδεχτεί τη συμβολοσειρά εισόδου, τότε το M προσομοιώνει το δεύτερο αυτόματο M_2 δουλεύοντας όμως τώρα στην κάτω ζώνη (στην οποία η αρχική συμβολοσειρά εισόδου βρίσκεται ανέπαφη).

Θεωρούμε ότι τα δύο αυτόματα M_1 και M_2 έχουν το ίδιο αλφάβητο $C = \{c_1, \ldots, c_n\} \cup \{\triangleright, \triangleleft, \sqcup\}$. Το νέο αυτόματο M που θα κατασκευάσουμε, θα είναι ένα γραμμικώς φραγμένο αυτόματο που θα έχει το ίδιο αλφάβητο με τα αυτόματα M_1 και M_2 και επιπλέον στο αλφάβητο αυτό θα προστεθούν τα σύμβολο $b^i_j,\ 0 \leq i, j \leq n$. Η διαισθητική ερμηνεία των νέων συμβόλων αυτών είναι η εξής: παρουσία του συμβόλου b^i_j υποδεικνύει ότι το σύμβολο c_i βρίσκεται στην άνω ζώνη ενώ το σύμβολο c_j στην κάτω ζώνη της ταινίας του αυτομάτου M. Τέλος, θεωρούμε ότι η q_1 είναι η αρχική κατάσταση του αυτομάτου M_1 και $\tilde{q}$ η τελική του, ενώ q_2 είναι η αρχική κατάσταση του αυτομάτου M_2. Επιπλέον των προηγουμένων, θαωρούμε ότι τα δύο σύνολα των καταστάσεων των δύο αυτομάτων μας, M_1 και M_2, είναι ξένα μεταξύ τους. Το νέο αυτόματο M έχει ως αρχική κατάσταση την κατάσταση q_0, η οποία δεν περιέχεται σε κανένα από τα δύο σύνολα καταστάσεων των δύο αυτομάτων M_1 και M_2, και ως τελική κατάσταση το M έχει την τελική κατάσταση του αυτομάτου M_2. Το αυτόματο M θα περιέχει τις ακόλουθες μεταβάσεις (για κάθε πιθανό σύμβολο $c_i,\ 0 \leq i \leq n$, όπου $c_0 \equiv \sqcup$).

(1) Αρχικοποίηση:

- $\delta(q_0, c_i) = (\bar{q}, b^i_i, \rightarrow)$
- $\delta(q_0, \triangleleft) = (\bar{q}', \triangleleft, \leftarrow)$
- $\delta(\bar{q}', b^i_i) = (\bar{q}', b^i_i, \leftarrow)$
- $\delta(\bar{q}', \triangleright) = (q_1, \triangleright, \rightarrow)$

Οι καταστάσεις $\bar{q}$ και $\bar{q}'$ είναι νέες καταστάσεις του αυτομάτου M και δεν περιλαμβάνονται στα σύνολα καταστάσεων των δύο αρχικών αυτομάτων M_1 και M_2. Οι παραπάνω μεταβάσεις εξαναγκάζουν το αυτόματο M να "αντιγράψει" τη συμβολοσειρά εισόδου και στις δύο ζώνες τις ταινίας του (ουσιαστικά μέσω των νέων συμβόλων b^i_j), και, αφούε τελιώσει με την αντιγραφή, να γυρίσει προς τα αριστερά στην αρχή της συμβολοσειράς.

(2) Για κάθε μετάβαση του αυτομάτου M_1, οι αντίστοιχες μεταβάσεις, οι οποίες προκύπτουν από την μετατροπή κάθε συμβόλου c_i σε $b^i_j,\ 0 \leq j \leq$

n, ανήκουν στο σύνολο μεταβάσεων του αυτομάτου M. Αυτές οι μεταβάσεις "αναγκάζουν" το αυτόματο M να προσομοιώσει τη λειτουργία του M_1 δουλεύοντας στην άνω ζώνη. Επιπλέον αυτών των μεταβάσεων, το αυτοματο M έχει τις επόμενες μεταβάσεις για $0 \leq i, j \leq n$:

- $\delta(\tilde{q}, b_j^i) = (\tilde{q}, b_j^i, \rightarrow)$
- $\delta(\tilde{q}, \triangleleft) = (\tilde{p}, \triangleleft, \leftarrow)$
- $\delta(\tilde{p}', b_j^i) = (\tilde{p}, c_j, \leftarrow)$
- $\delta(\tilde{p}, \triangleright) = (q_2, \triangleright, \rightarrow)$

Οπως και πριν, η κατάσταση $\tilde{p}$ δεν ανήκει στα σύνολα καταστάσεων των δύο αυτομάτων M_1 και M_2 αλλά αποτελεί μια νέα κατάσταση τοου αυτομάτου M. Οι παραπάνω μεταβάσεις αναγκάζουν το αυτόματο M να ανακτήσει την κάτω ζώνη και έπειτα να εισέλθει στην αρχική κατάσταση του αυτομάτου M_2, όταν διαβάσει το αριστερότερο σύμβολο της εισόδου.

(3) Τέλος, το αυτόματο M περιέχει όλες τις μεταβάσεις του αυτομάτου M_2

Η παραπάνω κατασκευή μας δίνει το ακόλουθο αποτέλεσμα: $L(M) = L(M_1) \cap L(M_2)$.

Τελειώνουμε τη συζήτηση της παρούσης ενότητας, καθώς και το τρέχον κεφάλαιο, με ένα τελευταίο πρόβλημα. Οταν η έρευνα πάνω στης γλώσσες με συμφραζόμενα ήταν στο απόγειό της, ένα από τα σημαντικότερα ανοιχτά ερωτήματα ήταν το κατά πόσον οι γλώσσες αυτές είναι κλειστές ως προς το συμπλήρωμα. Δηλαδή, δοθείσας μιάς γλώσσας με συμφραζόμενα, ήταν άγνωστο το κατά πόσο το συμπλήρωμα της γλώσσας αυτής, δηλαδή το σύνολο το συμβολοσειρών εκείνων που *δεν* ανήκουν στην αρχική γλώσσα, ήταν και αυτό με συμφραζόμενα ή όχι. Ενα αντίστοιχο αποτέλεσμα, πολύ εύκολο να αποδειχτεί, ήταν γνωστό για τις γλώσσες χωρίς συμφραζόμενα: οι γλώσσες χωρίς συμφραζόμενα *δεν* είναι κλειστές ως προς το συμπλήρωμα. Το αποτέλεσμα αυτό, έδειχνε στους ερευνητές ότι μάλλον και οι γλώσσες με συμφραζόμενα δεν είναι κλειστές ως προς το συμπλήρωμα. Ομως, απουσία μιας τυπικής απόδειξης, κανένας δεν ήταν σίγουρος και όλα περιστρέφονταν γύρω από την σφαίρα της εικασίας.

Ολα αυτά όμως μέχρι το 1987, οπότε και ο μαθηματικός Neil Immerman απέδειξε ένα εντυπωσιακό αποτέλεσμα: ότι οι κλάσεις πολυπλοκότητας *χώρου* είναι κλειστές ως προς το συμπλήρωμα. Από τη στιγμή όμως που οι γλώσσες με συμφραζόμενα περιγράφονται από μια τέτοια κλάση, όπως αποδείξαμε σε προηγούμενη ενότητα την κλάση $\mathbf{NSPACE}(n)$, τότε έχουμε το εξής εντυπωσιακό πόρισμα του αποτελέσματος του Neil Immerman: οι γλώσσες με

συμφραζόμενα είναι κλειστές ως προς το συμπλήρωμα! Στο ίδιο αποτέλεσμα έφτασε ανεξάρτητα την ίδια εποχή και ο Πολωνικής καταγωγής μαθηματικός R. Szelepcsényi, και από τότε το παραπάνω θεώρημα ονομάζεται θεώρημα των Immerman-Szelepcsényi.

Το θεώρημα αυτό, στην τυπική του μορφή, είναι το ακόλουθο:

Θεώρημα 11 *Εστω* $f(n) \geq \log n$ *μια κατάλληλη συνάρτηση πολυπλοκότητας. Τότε* $\mathbf{NSPACE}(f(n)) = \mathbf{coNSPACE}(f(n))$.

Η απόδειξη του παραπάνω θεωρήματος, αν και όχι ιδιαίτερα δύσκολη, εντούτης είναι μεγάλη σε μήκος και περιέχει πολλές τεχνικές λεπτομέρειες. Για μιά πλήρη απόδειξη παραπέμπουμε στα αντίστοιχα άρθρα [11], [**?**] ή σε πιο εκπαιδεύτικές αποδείξεις .

□

Μέρος ΙΙ

Γραμματικές Με Συμφραζόμενα για NP-πλήρεις Γλώσσες

Κεφαλαιο 5

Γραμματικη Με Συμφραζομενα για το προβλημα της Διαμερισης

5.1 Εισαγωγή

Στο παρόν κεφάλαιο θα περιγράψουμε ένα σύνολο κανόνων οι οποίοι θα μας δίνουν τη δυνατότητα να περιγράψουμε οποιοδήποτε σύνολο ακεραίων το οποίο να μπορεί να χωριστεί σε δύο ξένα μεταξύ τους υποσύνολα με ίσα υπο-αθροίσματα. Θέλουμε, με άλλα λόγια, να δημιουργήσουμε όλα τα "ναι" στιγμιότυπα για το πρόβλημα της διαμέρισης ενός συνόλου ακεραίων. Ο τυπικός ορισμός του προ˚βλήματος αυτού είναι ο εξής:

> **Διαμέριση:** Μας δίνεται ένα σύνολο ακεραίων X. Ζητείται να διαμεριστεί το σύνολο X σε δύο υποσύνολα X_1 και X_2 για τα οποία να ισχύει $\sum_{x_i \in X_1} x_i = \sum_{x_j \in X_2} x_j$. Ονομάζουμε το προηγούμενο πρόβλημα *Διαμέριση συνόλου ακεραίων*.

Η ιδέα είναι να δημιουργήσουμε αυτά τα δύο αθροίσματα αρχικά ώς δύο ίσους αριθμούς και ύστερα να τους "σπάσουμε" κατάλληλα και με τέτοιο τρόπο ώστε να είναι δυνατόν να δημιουργηθούν όλες εκείνες οι ακουλουθίες ακεραίων των οποίων το άθροισμα να μπορεί να μας δώσει τους δύο αυτούς αρχικούς αριθμούς οι οποίοι αντιπροσωπεύουν τα δύο ίσα υπο-αθροίσματα.

5.2 Αρχικοποίηση του Στιγμιοτύπου

Αρχικά θέλουμε ένα σύνολο κανόνων οι οποίοι και θα παράγουν έναν οποιονδήποτε αριθμό σε δύο αντίγραφα στο δυαδικό σύστημα αρίθμησης (ουσιαστικά θέλουμε μια γραμματική η οποία να παράγει όλες τις συμβολοσειρές τις μορφής $\{1\{x,x\}|x \in \{0,1\}^*\}$):

$$
\begin{aligned}
S &\rightarrow \{1AT \\
A &\rightarrow 0AC \mid 1AD \mid ,1 \\
C0 &\rightarrow 0C \\
C1 &\rightarrow 1C \\
D0 &\rightarrow 0D \\
D1 &\rightarrow 1D \\
CT &\rightarrow 0T \\
DT &\rightarrow 1T \\
T &\rightarrow \}B
\end{aligned}
$$

όπου η μεταβλητή B σηματοδοτεί τον τερματισμό της διαδικασίας παραγωγής του αριθμού.

Θα ασχοληθούμε με τους δύο αριθμούς που συμβολίζουν τα δύο ίσα υποαθροίσματα ξεχωριστά. Εστω $2N$ η δεκαδική τιμή που αντιστοιχεί στο άθροισμα των δύο ίσων αριθμών x. Αρχικά, θα προσπαθήσουμε να χωρίσουμε τον πρώτο αριθμό x σε ένα σύνολο θετικών ακεραίων των οποίων το αθροισμα να είναι ακριβώς $x(=\mathrm{N})$. Επειτα θα ασχοληθούμε με την ίδια ακριβώς διαδικασία με τον δεύτερο αριθμό x. Στο τέλος αυτής της διαδικασίας θα έχουμε δημιουργήσει ένα σύνολο ακεραίων $\{x_1, x_2, \ldots, x_k\}$, $k \leq N : x_1 + x_2 + \cdots x_k = 2N$ όπου οι i πρώτοι εξ'αυτών έχουν άθροισμα *ακριβώς* N $(i < k)$ και οι υπόλοιποι $k - i$ ακέραιοι έχουν άθροισμα πάλι *ακριβώς* N. Επειδή αυτό δεν συνιστά "καλό" στιγμιότυπο του προβλήματος της Διαμέρισης συνόλου ακεραίων, σε ένα τελικό στάδιο δημιουργούμε μια μετάθεση αυτών των k ακεραίων που δημιουργήσαμε καθώς, όπως θα εξηγήσουμε και στη συνέχεια, αυτό δεν αποτελεί ένα τυχαίο στιγμιότυπο του προβλήματός μας.

5.3 Παραγωγή του Στιγμιοτύπου

Αρχικά αποφασίζουμε να ασχοληθούμε μόνο με τον πρώτο αριθμό x και όταν τελειώσουμε το σπάσιμο αυτού, προχωράμε για το σπάσιμο του δεύτερου αριθμού x. Αρχικά χρειάζεται η μεταβλητή B αμέσως δείά του πρώτου κόμματος, δηλαδή να βρεθεί η γραμματική στην κατάσταση $\{x, Bx\}$:

$$
\begin{aligned}
\}B &\rightarrow B\} \\
0B &\rightarrow B0 \\
1B &\rightarrow B1 \\
,B &\rightarrow ,E
\end{aligned}
$$

όπου E είναι η κατάσταση εκείνη η οποία θα αποφασίζει μη ντετερμινιστικά εάν θα μειώσουμε κατά μία μονάδα την φορά τον αριθμό x και θα αυξήσουμε κατά 1 έναν δεύτερο αριθμό. Διασθητικά, σε κάθε βήμα θα αποφασίζω εάν θέλω να μειώσω τον αριθμό αριστερά της κατάστασης E κατά 1 και έπειτα να αυξήσω τον επόμενο αριθμό του κατα 1, δηλαδή δημιουργείται έτσι μια συμβολοσειρά y, zE τέτοια ώστε $y + z = N$. Σε κάθε βήμα αποφασίζω εάν θα μειωθεί το y (με ταυτόχρονη αύξηση του z πάντα κατά μία μονάδα) ή εάν θα αρχίσω να μειώνω το z και να αυξάνω έναν άλλο αριθμό. Στο τέλος της παραπάνω επαναληπτικής διαδικασίας θα έχω δημιουργήσει μια συμβολοσειρά $\{x_1, x_2, \ldots, x_i, x\} \mid i \leq N - 1 : x_1 + x_2 + \cdots x_i = N$.

Εστω ότι βρισκόμαστε στην κατάσταση y, zE τέτοια ώστε $y + z = N$. Η μεταβλητή E, όπως είπαμε, σε κάθε βήμα έχει δύο επιλογές: είτε να συνεχίσει να μειώνει κατά 1 τον y και ταυτόχρονα να αυξάνει κατά 1 τον z, είτε να αποφασίσει πως τελείωσε με το σπάσιμο (μείωση) του y και ήρθε η ώρα για να ακολουθηθεί η ίδια διαδικασία με τον αριθμό z. Στην περίπτωση αυτή πρέπει να δημιουργηθεί χώρος αμέσως δεξιά του z ο οποίος να είναι ικανός να χωράει τον νέο αριθμό που θα προκύπτει από την μείωση του z. Αυτός ο αριθμός σε καμία περίπτωση δεν μπορεί να έχει περισσότερα ψηφία απ'ότι ο z ο οποίος με την σειρά του δεν μπορεί να έχει περισσότερα ψηφία απ'ότι ο αρχικός αριθμός x. Ετσι, στην περίπτωση αυτή, κάθε φορά δηλαδή που αρχίζουμε να σπάμε έναν νέο αριθμό, γράφουμε δεξιά του τόσα μηδενικά όσα και ο αριθμός που μόλις αρχίσαμε να σπάμε και κατόπιν εφαρμόζουμε την πράξη της δυαδικής αύξησης κατά μία μονάδα σε αυτή τη συμβολοσειρά με τον κατάλληλο αριθμό μηδενικών επαναληπτικά. Επιπλέον προσθέτουμε στο τέλος αυτών των μηδενικών και τον χαρακρήρα διαχωρισμού ",":

$$
E \rightarrow F \mid G \mid R
$$

Δηλαδή, στην κατάσταση E στην οποία βρισκόμαστε αποφασίζουμε μη ντετερμινιστικά εάν θα συνεχίσουμε το σπάσιμου τους αριθμού y, γεγονός που μοντελοποιείται από την κατάσταση F, ή εάν θα αρχίσουμε να σπάμε τον αριθμό z (κατάσταση G). Στην κατάσταση R μεταβαίνουμε όταν αποφασίσουμε να ασχοληθούμε με το δεύτερο αντίγραφο του αριθμού x.

Εάν αποφασίσουμε να μεταβούμε στην κατάσταση G, δηλαδή αποφασίσουμε να αρχίσουμε το σπάσιμο του αριθμού z, τότε αρχικά είναι αναγκαίο να προσθέσουμε τον κατάλληλο αριθμό μηδενικών αμέσως δεξιά του αριθμού z. Αυτό γίνεται προσθέτοντας ένα νέο μηδενικό για κάθε μη σημαδεμένο ψηφίο του αριθμού z.

Πριν γίνει αυτό όμως, είναι απαρίτητο να ελένξουμε εάν μπορέι να σπάσει ο τρέχων αριθμός z, δηλαδή να δούμε εάν είναι ή όχι μηδέν:

$$\begin{aligned} 0G &\rightarrow G0 \\ 1G &\rightarrow 1G' \\ ,G &\rightarrow ,G'' \\ \{G &\rightarrow \{G'' \end{aligned}$$

Στην κατάσταση G' μεταβαίνουμε εάν συναντήσουμε κάποιον άσσο, γεγονός που υποδηλώνει ότι ο αριθμός δεν είναι μηδέν και άρα επιδέχεται μείωσης. Αντίθετα, εάν δούμε ότι ο αριθμός είναι μηδέν, τότε μεταβαίνουμε στην κατάσταση G'' κοπός της οποίας είναι να μας μεταφέρει στο δεξί άκρο του αριθμού και μετά να μεταβεί στην κατάσταση R ώστε να αρχίσουμε την διάσπαση του δευτέρου αντιγράφου του αρχικού αριθμού x:

$$\begin{aligned} G'0 &\rightarrow 0G' \\ G'1 &\rightarrow 1G' \\ G', &\rightarrow \bar{G}, \\ G''0 &\rightarrow 0G'' \\ G''1 &\rightarrow 1G'' \\ G'', &\rightarrow ,R \end{aligned}$$

Στην περίπτωση που ο αριθμός δεν είναι μηδέν, αρχίζουμε τη διαδικασία προσθήκης των απαρίτητων μηδενικών. Αρχικά, σημαδεύουμε το πρώτο μη σημαδεμένο ψηφίο του αριθμού z:

$$
\begin{aligned}
0\bar{G} &\rightarrow 0'H \\
1\bar{G} &\rightarrow 1'H
\end{aligned}
$$

Επειτα, μέσω της καταστάσεως H, προσπερνάμε όλα τα ήδη σημαδεμένα σύμβολα - ψηφία του τρέχοντος αριθμού κινούμενοι προς τα δεξιά, μέχρι να βρούμε τον χαρακτήρα "," οπότε και προσθέτουμε το τρέχον μηδενικό στη σωστή του θέση και κατόπιν μεταβαίνουμε στην κατάσταση J ώστε να γυρίσουμε προς τα αριστερά ώστε να συνεχιστεί η διαδικασία πρόσθεσης νέων μηδενικών για κάθε μη σημαδεμένο ψηφίο του αριθμού z, εάν έχει απομείνει βέβαια:

$$
\begin{aligned}
H0 &\rightarrow 0H \\
H1 &\rightarrow 1H \\
H, &\rightarrow J, 0 \\
0'J &\rightarrow J0 \\
1'J &\rightarrow J1 \\
0J &\rightarrow 0'H \\
1J &\rightarrow 1'H
\end{aligned}
$$

Η παραπάνω διαδικασία συνεχίζεται μέχρι να φτάσουμε στο αριστερό άκρο του αριθμού z, δηλαδή μέχρι να καταλάβουμε, μέσω της καταστάσεως J ότι τα ψηφία του αριθμού έχουν τελείώσει και άρα έχουμε προσθέσει όλα τα απαραίτητα μηδενικά στη σωστή τους θέση. Μόλις τελειώσει αυτή διαδικασία, μεταβαίνουμε στην κατάλληλη κατάσταση, K, στην οποία κινούμενοι προς τα δεξιά, ξεσημαδεύουμε όλα τα σημαδεμένα ψηφία του τρέχοντος αριθμού z και, μέσω κατάλληλων μεταβάσεων, προσπερνάμε τον νέο αριθμό που αποτελείται όνο από μηδενικά και μεταβαίνουμε τέλος στη κατάσταση F σκοπός της οποίας είναι να αρχίσουμε να αυξάνομε τον αριθμό στα αριστερά αυτής και να μειώσουμε τον προηγούμενο αριθμό z κατά μία μονάδα αντιστοιχα:

$$
\begin{aligned}
,J &\rightarrow ,K \\
\{J &\rightarrow \{K \\
K0' &\rightarrow 0K \\
K1' &\rightarrow 1K \\
K, &\rightarrow ,L \\
L0 &\rightarrow 0L \\
L, &\rightarrow F,
\end{aligned}
$$

Δηλαδή τώρα έχει δημιουργηθεί η συμβολοσειρά $\{x_1, \ldots, y, z, 00 \ldots 0F, x$ αφότου αποφασίσαμε να σπάσουμε τον z και άρα προσθέσαμε τα αναγκαία μηδενικά. Για τον λόγο αυτό, είμαστε στην κατάσταση F η οποία διαισθητικά συμβολίζει σε μια συμβολοσειρά, έστω $\alpha, \beta F$, την διαδικασία αύξησης κατά μία μονάδα της συμβολοσειράς β με ταυτόχρονη μείωση της α πάλι κατά μία μονάδα. Επειτα από αυτό, η F θα γίνει E η οποία διασθητικά αποφασίζει για την συνέχεια του σπασίματος του τρέχοντος αριθμού ή για την έναρξη του σπασίματος του επόμενου αριθμού (καταστάσεις F και G αντίστοιχα) ή εάν θα μεταβεί στην R για τον επόμενο αριθμό x.

Είναι αναγκαίο να περιγραφεί η διαδικασία αύξησης και ταυτόχρονης μείωσης δύο διαδοχικών συμβολοσειρών. Γενικά θα έχουμε μια συμβολοσειρά της μορφής x, yF. Θέλουμε να μειωθεί ο αριθμός x και *ταυτόχρονα* να αυξηθεί ο αριθμός y μέχρι να αποφασίσουμε πως θέλουμε να αφήσουμε τον x όπως είναι και να αρχίσουμε να μειώνουμε τον y (επιλογές που εκφράζονται από τις καταστάσεις F και G). Πρώτα περιγράφουμε τους κανόνες εκείνους οι οποίοι αυξάνουν έναν δυαδικό αριθμό:

$$
\begin{aligned}
1F &\rightarrow F0 \\
0F &\rightarrow M1
\end{aligned}
$$

Αρα, αυξάνοντας τον y με τον παραπάνω τρόπο δημιουργείται η συμβολοσειρά $x, y_1 M y_2$ με $y_1 + y_2 = (y + 1)$. Τώρα πρέπει να μειωθεί ο αριθμός x κατά ένα. Μεταφέρουμε αρχικά τον κανόνα (μεταβλητή) M αμέσως δεξιά του πρώτου χαρακτήρα "," που θα συναντήσει:

$$
\begin{aligned}
1M &\rightarrow M1 \\
0M &\rightarrow M0 \\
,M &\rightarrow N,
\end{aligned}
$$

Δηλαδή δημιουργείται η συμβολοσειρά xN, y_1y_2. Τώρα, χρειάζεται να μειωθεί ο αριθμός x κατά ένα:

$$
\begin{aligned}
1N &\rightarrow 0Q \\
0N &\rightarrow N1 \\
,N &\rightarrow ,N' \\
\{N &\rightarrow \{N'
\end{aligned}
$$

Αμα φτάσουμε στην κατάσταση N' ξέρουμε ότι ο τρέχων αριθμός είναι μηδέν: δεν επιδέχεται επιπλέον μείωση. Από την κατάσταση αυτή, είναι αναγκαίο, κινούμενοι προς τα δεξιά, να μετατρέψουμε όλους τους άσσους σε μηδενικά στον τρέχων αριθμό και μόλις φτάσουμε στον επόμενο αριθμό να "αναιρέσουμε" την προηγούμενη αύξησή του, μειώνοντάς τον κατά μία μονάδα: ο αριθμός αυτός δεν μπορεί να είναι μηδέν, και έπειτα να μεταβούμε στη κατάλληλη καάσταση G ή R των οποίων τη λειτουργία περιγράψαμε παραπάνω:

$$
\begin{aligned}
N'1 &\rightarrow 0N' \\
N', &\rightarrow ,N'' \\
N''1 &\rightarrow 1N'' \\
N''0 &\rightarrow 0N'' \\
N'', &\rightarrow \bar{N}, \\
0\bar{N} &\rightarrow \bar{N}1 \\
1\bar{N} &\rightarrow 0\tilde{N} \\
\tilde{N}0 &\rightarrow 0\tilde{N} \\
\tilde{N}1 &\rightarrow 1\tilde{N} \\
\tilde{N}, &\rightarrow G \mid R,
\end{aligned}
$$

Τώρα, από την κατάσταση Q, στην πιό πάνω περίπτωση, είναι απαραίτητο να κινηθούμε προς τα αριστερά για να δούμε εάν είναι εφικτή η μείωση του

τρέχοντος αριθμού σε επόμενο βήμα. Δηλαδή, η κατάσταση αυτή προχωράει προς τα αριστερά όσο βρίσκει μηδενικά. Μόλις βρεί άσσο τότε αυτόματα μεταβαίνει σε μιά νέα κατάσταση, την P, η οποία και συμβολίζει τη συνέχιση της μη ντετερμινιστικής διαδικασίας: ο τρέχων αριθμός μπορεί να μειωθεί σε επόμενο βήμα.

$$\begin{aligned} 0Q &\rightarrow Q0 \\ 1Q &\rightarrow 1P \\ ,Q &\rightarrow ,Q' \\ \{Q &\rightarrow \{Q' \end{aligned}$$

Στην κατάσταση Q' φτάσαμε στο αριστερό άκρο του αριθμού χωρίς να βρούμε κάποιον άσσο: ο αριθμός είναι μηδέν και δε μπορεί να μειωθεί σε επόμενο βήμα. Αρα προχωράμε τον δρομέα προς τη κατάλληλη θέση, στο δεξίο τέλος του επόμενου αριθμού, και μεταβαίνουμε στη κατάσταση G ώστε να μειωθεί ο τρέχον αριθμός εφόσον αυτό είναι δυνατόν:

$$\begin{aligned} Q'0 &\rightarrow 0Q' \\ Q'1 &\rightarrow 1Q' \\ Q', &\rightarrow ,Q'' \\ Q''0 &\rightarrow 0Q'' \\ Q''1 &\rightarrow 1Q'' \\ Q'', &\rightarrow G \mid R, \end{aligned}$$

Σε διαφορετική περίπτωση, δημιουργείται η συμβολοσειρά x_1Px_2, y_1y_2 όπου $x_1 + x_2 = x - 1$. Τώρα χρειάζεται να μεταφερθεί η μεταβλητή - δρομέας P στο δεξί άκρο της συμβολοσειράς ώστε να δημιουργηθεί η συμβολοσειρά x_1x_2, y_1y_2P:

$$
\begin{aligned}
P1 &\rightarrow 1P \\
P0 &\rightarrow 0P \\
P, &\rightarrow ,P' \\
P'1 &\rightarrow 1P' \\
P'0 &\rightarrow 0P' \\
P', &\rightarrow E,
\end{aligned}
$$

Τωρα πρέπει να αποφασιστεί εάν θα συνεχιστεί η παραπάνω διαδικασία μείωσης του x και ταυτόχρονης άυξησης του y, ή εάν θα αρχίσει το "σπάσιμο" (μείωση) του y και η ταυτόχρονη αύξηση ενός άλλου αριθμου z, κάτι που γίνεται με την παραπάνω μετάβαση $P', \rightarrow E$.

Μη ντετερμινιστικά, κάποια στιγμή, θα αποφασίσουμε να μεταβούμε στην κατάσταση R, δηλαδή να ασχοληθούμε με το δέυτερο αντίγραφο της συμβολοσειράς x. Οι αντίστοιχοι κανόνες είναι οι ίδιοι με την μόνη διαφορά ότι τώρα από την κατάσταση E, ούτε από καμία άλλη αντίστοιχη, δεν μπορούμε να μεταβούμε στην R.

Επιπλέον, όταν δούμε ότι δεν μπορούμε να μειώσουμε επιπλέον έναν αριθμό, βλέπε κατάσταση Q'' παραπάνω, καθώ ςπροχωράμε προς τα δεξιά, δε θα συναντήσουμε τον χαρακτήρα "," αλλά τον }, οπότε αντικαθιστούμε τον κανόνα αυτόν με τον

$$Q''\} \rightarrow \Omega\}$$

5.4 Μετάθεση των Αριθμών

Σε κάποιο σημείο η παραγωγή των επιθυμητών αριθμών που αποτελούν στοιχεία του στιγμιοτύπου του προβλήματος της διαμέρισης θα σταματήσει. Στο σημείο αυτό θα έχουμε δημιουργήσει την συμβολοσειρά $\{x_1, x_2, \cdots, x_k\}$ τέτοια ώστε $x_1+x_2+\cdots+x_i = N$ και $x_{i+1}+x_{i+2}+\cdots+x_k = N$ για κάποιο $i < N$. Ομως στην περίπτωση αυτού του στιγμιοτύπου, το πρόβλημα της διαμέρισης γίνεται πολυωνυμικά επιλύσιμο: αθροίζουμε όλα τα x_j, $0 \leq j \leq k$ και έπειτα αρχίζουμε να αθροίζουμε τα x_i μέχρι το κατάλληλο i, δηλαδή μέχρι εκεί που το άθροισμα των αριθμώς γίνεται ίσο με το μισό του προηγούμενα υπολογισμένου αθροίσματος. Για τον λόγο αυτό, χρειαζόμαστε μια *μετάθεση* των παραπάνω

αριθμών. Η διαδικασία της μετάθεσης σε κάθε βήμα, διαισθητικά, θα ποφασίζει έαν θα εναλλάξει δύο *συνεχόμενους* αριθμούς. Αυτή η διαδικάσία μπορεί να συνεχιστεί για όσες φορές καταστεί αναγκαίο, μεχρι να δημιουργηθεί το επιθυμητό στιγμιότυπο του προβλήματος της διαμέρισης συνόλου ακεραίων.

Στην γενική περίπτωση εναλλαγής δύο διαδοχικών αριθμών x, y, η συμβολοσειρά θα έχει την εξής γενική μορφή: $\triangleright x_1 y_2, y_1 x_2 \triangleleft$ όπου x_2 είναι το τμήμα της συμβολοσειάς x το οποίο έχει εναλλαγεί με το αντίστοιχο τμήμα της συμβολοσειράς y, δηλαδή y_2, και αντίστοιχα x_1 και y_1 είναι τα τμήματα των συμβολοσειρών x και y που απομένει να εναλλαγούν. Επιπλέον τα σύμβολα $\triangleright$ και $\triangleleft$ συμβολίζουν τους χαρακτήρες που είναι πριν και μετα την συμβολοσειρά x, y αι οι οποίοι μπορεί να είναι "," ή "{,}". Στο σημείο αυτό, πολύ βοηθητικό μας είναι το γεγονός ότι όλοι οι αριθμοί στην παραγώμενη συμβολοσειρά έχουν το ίδιο μήκος.

Πιο συγκεκριμένα, θέλουμε να αντιγράψουμε το δεξιότερο ψηφίο της συμβολοσειράς x_1, έστω σ_x, στον δεξιότερο χαρακτήρα του y_1, έστω σ_y, και στη συνέχεια να αντικαταστήσουμε το σ_x με σ_y. Σε ένα τυχαίο στιγμιότυπο εναλλαγής δύο αριθμών θα έχουμε $\triangleright x_1 \sigma_x y_2, y_1 \sigma_y x_2 \Phi \triangleleft$ όπου Φ η κατάσταση εναλλαγής των x, y.

Αρχικά θα έχουμε την συμβολοσειρά $\triangleright x, y \Omega \triangleleft$ όπου:

$$\begin{aligned} \Omega &\rightarrow \Pi \\ \Omega &\rightarrow \Psi \end{aligned}$$

με την κατάσταση Π να συμβολίζει την έναρξη εναλλαγής των δύο αριθμών x, y, ενώ η κατάσταση Ψ συμβολίζει την απόφαση της μη εναλλαγής των x, y αλλά τη συνέχιση της διαδικασίας εναλλαγής κάποιου άλλου ζεύγους αριθμών. Οταν εισέλθουμε στην κατάσταση Π, πρέπει να κοιτάξουμε εάν υπάρχει ο αριθμός x! Αυτό γιατί μπορεί να βρισκόμαστε σε μια κατάσταση στην οποία ο αριθμός y είναι ο πρώτος από τα αριστερά αριθμός και ως εκ τούτου δεν έχει κάποιον αλλο αριθμό αριστερά του για να εναλλαχθεί. Εάν όντως διαπιστώσουμε ότι η εναλλαγή που αποφασίσαμε να κάνουμε δεν μπορεί να γίνει τότε η κατάσταση Π μας οδηγεί στην Ψ, αλλιώς η Π οδηγεί στην Φ, στην κατάσταση δηλαδή της έγκυρης εναλλαγής:

$$
\begin{array}{rcl}
0\Pi & \rightarrow & \Pi 0 \\
1\Pi & \rightarrow & \Pi 1 \\
,\Pi & \rightarrow & ,\Pi' \\
\{\Pi & \rightarrow & \{\Pi'' \\
\Pi' 0 & \rightarrow & 0\Pi' \\
\Pi' 1 & \rightarrow & 1\Pi' \\
\Pi', & \rightarrow & \Phi, \\
\Pi'\} & \rightarrow & \Phi\} \\
\Pi'' 0 & \rightarrow & 0\Pi'' \\
\Pi'' 1 & \rightarrow & 1\Pi'' \\
\Pi'', & \rightarrow & \Psi, \\
\Pi''\} & \rightarrow & \Psi\}
\end{array}
$$

Οταν μπούμε στην φάση της αντιγραφής, ουσιαστικά πρέπει να εναλλάξουμε το σ_y με το σ_x (ή αντίστροφα, χωρίς καμία ουσιώδη διαφορά). Για ευκολία, όταν έχουμε ήδη εναλλάξει δύο ψηφία των αριθμών x και y, τότε τα σημαδεύουμε με κάποιο τρόπο (πχ με ") ετσι ώστε να γνωρίζουμε ανα πάσα στιγμή τα τμήματα των δύο αριθμών που έχουμε ήδη εναλλάξει, δηλαδή τα τμήματα x_2 και y_2 στον παραπάνω συμβολισμό. Η εναλλαγή αυτή έχει διάφορες φάσεις τις οποίες περιγράφουμε παρακάτω.

$$
\left.
\begin{array}{rcl}
0''\Phi & \rightarrow & \Phi 0'' \\
1''\Phi & \rightarrow & \Phi 1'' \\
,\Phi & \rightarrow & ,\Sigma
\end{array}
\right\} (\textit{αγνόησε το τμήμα } y_2)
$$

Η παραπάνω μεταβλητή Σ υποδηλώνει ότι έχει τελειώσει η αντιγραφή: δεν έχει απομείνει τίποτα άλλο να αντιγράψουμε. Τώρα το μόνο που απομένει να κάνουμε είναι να μετατρέψουμε τα σημαδεμένα σύμβολα σε κανονικά, όπως θα δούμε στο τέλος.

$$
\left.
\begin{array}{rcl}
0\Phi & \rightarrow & X_0 0' \\
1\Phi & \rightarrow & X_1 1'
\end{array}
\right\} (\sigma_y\textit{: σημάδεψέ το, απομνημόνευσέ το στην } X)
$$

$$
\left.
\begin{array}{rcl}
0X & \rightarrow & X0 \\
1X & \rightarrow & X1 \\
,X & \rightarrow & ,X
\end{array}
\right\} (\textit{αγνόησε το } y_1.\ X \in \{X_0, X_1\})
$$

$$\left.\begin{array}{rcl} 0''X & \rightarrow & X0'' \\ 1''X & \rightarrow & X1'' \end{array}\right\} (\textit{αγνόησε το } y_2)$$

$$\begin{array}{rcl} 0X_0 & \rightarrow & 0'\Theta_0 \\ 1X_0 & \rightarrow & 1'\Theta_1 \end{array}$$

Με τους παραπάνω δύο κανόνες, αφού έχω ήδη φτάσει στο σ_x, το αντικαθιστών με σ_y το οποίο θυμάμαι στην κατάσταση X (X_0 ή X_1). Επιπλέον, στην κατάσταση Θ απομνημονεύουμε το σ_x ούτος ώστε να το αντιγράψουμε πίσω στο σ_y. Αντίστοιχα ισχύουν και για την κατάσταση X_1.

$$\left.\begin{array}{rcl} \Theta 0'' & \rightarrow & 0''\Theta \\ \Theta 1'' & \rightarrow & 1''\Theta \\ \Theta, & \rightarrow & ,\Theta \end{array}\right\} (\textit{αγνόησε το } y_2.\ \Theta \in \{\Theta_0, \Theta_1\})$$

$$\left.\begin{array}{rcl} \Theta 0 & \rightarrow & 0\Theta \\ \Theta 1 & \rightarrow & 1\Theta \end{array}\right\} (\textit{αγνόησε το } y_1.\ \Theta \in \{\Theta_0, \Theta_1\})$$

$$\left.\begin{array}{rcl} \Theta_0 0' & \rightarrow & Y0'' \\ \Theta_0 1' & \rightarrow & Y0'' \\ \Theta_1 0' & \rightarrow & Y1'' \\ \Theta_1 1' & \rightarrow & Y1'' \end{array}\right\}$$

Με τους παραπάνω τέσσερεις κανόνες αντικαθιστούμε το σ_y με σ_x, σημαδεύουμε κατάλληλα για να θυμόμαστε το τέλος της συμβολοσειράς x_2 και εισερχόμαστε στην κατάσταση Y η οποία διαισθητικά υποδηλώνει ότι η αντικατάσταση έγινε σωστά. Προχωράμε έπειτα την μεταβλητή Y δεξιά του x_2 έτσι ώστε να συνεχιστεί η διαδικασία της εναλλαγής:

$$\begin{array}{rcl} Y0 & \rightarrow & 0Y \\ Y1 & \rightarrow & 1Y \\ Y, & \rightarrow & \Phi, \\ y\} & \rightarrow & \Phi\} \end{array}$$

Οταν τελειώση η διαδικασία της αντιγραφής, όταν δηλαδή μπούμε στην κατάσταση Σ, πρέπει να μετατρέψουμε τα σημαδεμένα σύμβολα σε αυτά που ήταν αρχικά:

$$
\begin{array}{rcl}
,\Sigma 0'' & \rightarrow & \Sigma, 0'' \\
0''\Sigma & \rightarrow & 0\Sigma \\
1''\Sigma & \rightarrow & 1\Sigma \\
,\Sigma 0 & \rightarrow & ,\Sigma' 0 \\
,\Sigma 1 & \rightarrow & ,\Sigma' 1 \\
\{\Sigma 0 & \rightarrow & \{\Sigma' 0 \\
\{\Sigma 1 & \rightarrow & \{\Sigma' 1 \\
\Sigma' 0 & \rightarrow & 0\Sigma' \\
\Sigma' 1 & \rightarrow & 1\Sigma' \\
\Sigma', & \rightarrow & ,\Sigma' \\
\Sigma' 0'' & \rightarrow & 0\Sigma' \\
\Sigma' 1'' & \rightarrow & 1\Sigma' \\
\Sigma'\} & \rightarrow & \Omega\}
\end{array}
$$

Κεφαλαιο 6

Γραμματικη Με Συμφραζομενα για το προβλημα του Κατευθυνομενου Μονοπατιου Hamilton

6.1 Εισαγωγή

Θέλουμε να δημιουργήσουμε, σε μορφή συνόλου κανόνων, όλα τα "ναι" στιγμιότυπα για το πρόβλημα απόφασης του κατά πόσον ένα *κατευθυνόμενο* γράφημα έχει μονοπάτι Hamilton. Δηλαδή να παρουσιάσουμε ένα σύνολο κανόνων από τους οποίους να είναι δυνατόν να δημιουργηθούν οποιαδήποτε κατευθυνόμενα γραφήματα τα οποία περιέχουν τουλάχιστον ένα μονοπάτι Hamilton (και μόνο αυτά τα γραφήματα). Το πρόβλημα απόφασης έαν ένα γράφημα περιέχει μονοπάτι Hamilton είναι το εξής:

> **Μονοπάτι Hamilton**: δοθέντος ενός κατευθυνόμενου γραφήματος $G = (V, E)$, αληθεύει ότι το G περιέχει ένα (απλό) κατευθυνόμενο μονοπάτι το οποίο περνάει ακριβώς μία φορά από καθε κορυφή του γραφήματος;

Η ιδέα της παραγωγής όλων αυτών των "ναι" στιγμιοτύπων είναι η εξής: αρχικά παράγουμε έναν (δυαδικό) αριθμό $N > 0$ ο οποίος και θα μας δίνει το *πλήθος* των κορυφών στο γράφημα $G = (V, E)$ ($|V| = N+1$). Επειτα, με βάση αυτόν τον αριθμό N, δημιουργούμε όλους τους αριθμούς $i, 0 \leq i < N$ οι οποίοι και θα αντιστοιχούν στα ονόματα αυτών των $N+1$ κόμβων. Με βάση αυτές τις $N+1$ κορυφές, κωδικοποιούμε κατάλληλα ένα μονοπάτι στο οποίο καθε μία από τις $N+1$ κορυφές εμφανίζεται ακριβώς *μία* φορά. Αφού δημιουργήσουμε το επιθυμητό μονοπάτι Hamilton, προσθέτουμε όλες τις υπόλοιπες ακμές οι οποίες, μαζί με το μονοπάτι Hamilton, συνθέτουν το επιθυμητό γράφημα, το οποίο λόγο του τρόπου κατασκευής του, αποτελεί ένα "ναι" στιγμιότυπο για το πρόβλημα απόφασης του εάν ένα γράφημα περιέχει μονοπάτι Hamilton.

6.2 Παραγωγή των κορυφών του γραφήματος

Αρχικά, θέλουμε τους κανόνες εκείνους οι οποίοι να δημιουργούν ένα οποιοδήποτε ακέραιο μεγαλύτερο της μονάδος σε δυαδική μορφή:

$$\begin{aligned} S &\rightarrow \{\{1A\}\} \\ A &\rightarrow 0A|1A|B \\ 0B &\rightarrow B0 \\ 1B &\rightarrow B1 \\ \{B &\rightarrow \{I \end{aligned}$$

Ο αριθμός αυτός, έστω N η τιμή του στο δεκαδικό σύστημα, θα αναπαριστά την μέγιστη ετικέτα ονομάτος των κορυφών του γραφήματος: ονοματίζουμε τις κορυφές με αριθμούς, η κορυφή 0 έχει τον ελάχιστο δείκτη και ο κορυφή N τον μέγιστο (σε δεκαδικές τιμές: η αναπαράσταση εσωτερικά στην γραμματική θα γίνεται στο δυαδικό σύστημα, όπως και οποιαδήποτε άλλη λειτουργία).

Αφού έχει τελειώσει η κατασκευή του αριθμού αυτού, βρισκόμαστε στην κατάσταση I της οποίας η λειτουργία μπορεί διαισθητικά να περιεγραφεί ώς εξής: ελέγχουμε τον αριθμό που είναι στα αριστερά μας. Εάν αυτός περιέχει *τουλάχιστον* έναν άσσο, τότε το θυμόμαστε το σε μια νέα κατάσταση. Αυτό το τεχνικό βήμα μας βοηθάει για τον εξής λόγο: Το γράφημα γενικά θα αναπαρίσταται ως $\{\{V\}, \{E\}\}$. Για να δημιουργήσουμε την λίστα ονομάτων των κορυφών του γραφήματος, πρέπει να προσθέσουμε στην συμβολοσειρά όλα τα ονομάτα των κορυφών αυτών, δηλαδή όλους τους ακεραίους από 0 έως N. Αυτό γίνεται επαναληπτικά ως εξής: αρχικά ελέγχουμε έαν ο πρώτος από αριστερά αριθμός μπορεί να μειωθεί, εαν περιέχει δηλαδή έναν τουλάχιστον άσσο.

Εάν ναι, τότε δημιουργούμε ένα αντίγραφό του αμέσως αριστερά το οποίο στη συνέχεια το μειώνουμε κατά ένα. Επαναλαμβάνουμε αυτή τη διαδικασία μέχρι ο τρέχον αριθμός να μην μπορεί να μειωθεί, γιατί προφανώς περιέχει μόνο μηδενικά. Τότε σταματάμε την παραπάνω διαδικασία: έχουμε δημιουργήσει όλους τους αριθμός από 0 έως N, και άρα και τα ονόματα των $N+1$ κορυφών του γραφήματος σε *αύξουσα* από αριστερά προς τα δεξιά διάταξη.

Πιο συγκεκριμένα:

$$
\begin{aligned}
I0 &\rightarrow 0I \\
I, &\rightarrow K, \quad (K : \text{είναι μηδέν}) \\
I1 &\rightarrow 1J \quad (J : \text{δεν είναι μηδέν}) \\
0J &\rightarrow J0 \\
1J &\rightarrow J1 \\
\{J &\rightarrow \{, C
\end{aligned}
$$

Στην κατάσταση C έχουμε μεταβεί μέσω της καταστάσεως J όταν διαπιστώσουμε ότι ο αριθμός δεν είναι μηδέν και άρα μπορεί να αντιγραφεί αι να μειωθεί κατά μία μονάδα. Θέλουμε τον αριθμό που βρίσκεται δεξιά της τρέχουσας θέσης του δρομέα, κάτι που συμβολίζεται από τη μεταβλητή C, να τον αντιγράψουμε αριστερά από το σύμβολο "," και κατόπιν να μειώσουμε το αντίγραφο που δημιουργήσαμε κατά μία μονάδα:

$$
\left.
\begin{aligned}
C0 &\rightarrow D_0 0' \\
C1 &\rightarrow D_1 1'
\end{aligned}
\right\} (\textit{σημάδεψε στην } D \textit{ το τρέχον σύμβολο αντιγραφής})
$$

$$
\left.
\begin{aligned}
C0' &\rightarrow 0'C \\
C1' &\rightarrow 1'C
\end{aligned}
\right\} (\textit{αγνόησε ότι έχει ήδη αντιγραφεί})
$$

$$
\left.
\begin{aligned}
C, &\rightarrow F, \\
C\} &\rightarrow F\}
\end{aligned}
\right\} (\textit{η αντιγραφή μόλις τελείωσε})
$$

Με τους παρακάτω κανόνες αντιγράφουμε το τρέχον σύμβολο στην κατάλληλη θέση και επανερχόμαστε στην κατάλληλη κατάσταση αντιγραφής (οι περίπτωση της κατάστασης D_1 είναι τελείως συμμετρική):

$$
\begin{aligned}
0'D_0 &\rightarrow D_0 0' \\
1'D_0 &\rightarrow D_0 1' \\
,D_0 &\rightarrow 0, C
\end{aligned}
$$

Αφού τελειώσει η διαδικασία της αντιγραφής, θα έχουμε δημιουργήσει μιά συμβολοσειρά της μορφής $\{\{\alpha, \alpha F, \alpha+1, \alpha+2, \ldots, N\}\}$ όπου ο κάθε αριθμός αναπαρίσταται στο δυαδικό σύστημα. Βρισκόμαστε στην κατάσταση F η οποία διαισθητικά δηλώνει ότι έχει τελειώση η διαδικασία αντιγραφής του αριθμού α στα αριστερά του. Είναι αναγκαίο στην παρούσα φάση να αντιμετωπίσουμε δύο περιπτώσεις που προέκυψαν: 1) να ξεσημεδέψουμε τα σημαδεμένα σύμβολα του πρώτα από τα δεξιά α, τα οποία προέκυψαν κατά τη διαδικασία της αντιγραφής του και 2) να μειώσουμε το πρώτο προς τα τα αριστερά α κατά ένα ώστε να δημιουργηθεί η επιθυμιτή στην τρέχουσα φάση συμβολοσειρά $\{\{\alpha-1, \alpha, \alpha+1, \alpha+2, \ldots, N\}\}$. Η παραπάνω διαδικασίες συνεχίζονται επαναληπτικά έως ότου δημιουργηθεί η λίστα ονομάτων των κορυφών του γραφήματος $\{\{0, 1, 2, \ldots, N\}\}$ (σε δυαδική πάντοτε αναπαράσταση). Παρακάτω αναλύονται τα επιθυμητά βήματα.

Αρχικά, θέλουμε να ξεσημαδέψουμε τον αριθμό που έχουμε ήδη σημαδέψει κατά τη διαδικασία της αντιγραφής:

$$
\begin{aligned}
0'F &\rightarrow F0 \\
1'F &\rightarrow F1 \\
,F &\rightarrow G,
\end{aligned}
$$

Στην κατάσταση G η τρέχουσα διαμόρφωση είναι η $\{\{\alpha G, \alpha, \alpha+1, \alpha+2, \ldots, N\}\}$ για κάποιο α, $0 < \alpha \leq N$. Αρχίζουμε την μείωση του α που βρίσκεται στα αριστερά της καταστάσεως G:

$$
\begin{aligned}
0G &\rightarrow G1 \\
1G &\rightarrow 0H
\end{aligned}
$$

Φέρνουμε το σύμβολο-δρομέα H στην αρχή της δημιουργηθήσας συμβολοσειράς:

$$
\begin{aligned}
0H &\rightarrow H0 \\
1H &\rightarrow H1 \\
\{H &\rightarrow \{I
\end{aligned}
$$

Δηλαδή τώρα είμαστε στην διαμόρφωση $\{\{I\alpha-1, \alpha, \alpha+1, \alpha+2, \ldots, N\}\}$. Οπως είπαμε, στην κατάσταση I ελέγχουμε εάν ο αριθμός αμέσως δεξιά της καταστάσεως-δρομέα I είναι μηδέν ή όχι. Εάν όχι, τότε συνεχίζουμε την διαδικασία αντιγραφής και μείωσης κατά μία μονάδα. Εάν ο αριθμός είναι μηδέν. τότε έχουμε τελειώσει με την παραγωγή της λίστας ονομάτων του γραφήματος και εισερχόμαστε σε μια κατάσταση παραγωγής του μονοπατιού Hamilton (κατάσταση K όπως περιγράφηκαν παραπάνω οι μεταβάσεις του I).

6.3 Δημιουργία του μονοπατιού Hamilton

Δεδομένου ότι έχουμε δημιουργήσει ορθά την λίστα ονομάτων των κορυφών του γραφήματος, αυτό που έπεται είναι 1) να δημιουργήσουμε ένα κατάλληλο μονοπάτι Hamilton και 2) να προσθέσουμε τις απαραίτητες επιπλέον ακμές ώστε να παριγράψουμε πλήρως το επιθυμητό γράφημα.

Πρώτα απ'όλα, είναι αναγκαίο να καθορίσουμε την αναπαράσταση που θα έχουν αυτές οι ακμές. Γενικά μια (κατευθυνόμενη) ακμή θα αναπαρίσταται ως ένα *διατεταγμένο* ζεύγος των δύο τερματικών κορυφών της. Δηλαδή, έαν οι κορυφές α και β συνδέονται με ακμή με κατεύθυνση προς την κορυφή β, τότε αυτή θα αναπαρίσταται στην συμβολοσειρά μας ως (α, β) και όχι ως (β, α) αλλά όχι και τα δύο *ταυτόχρονα*. Οπως είπαμε, το γράφημα θα είναι μια συμβολοσειρά της μορφής $\{\{V\}, \{E\}\}$. Εχουμε ήδη δημιουργήσει το σύνολο V. Το σύνολο E θα κατασκευαστεί, όπως είπαμε, σε δύο βήματα: 1) αρχικά οι ακμές που αποτελούν το μονοπάτι Hamilton και 2) οι υπόλοιπες ακμές. Γενικά, ένα μονοπάτι Hamilton θα είναι μια ακολουθία των ακμών που το αποτελούν της εξής μορφής: $(v_0, v_1), (v_1, v_2), \ldots, (v_{N-1}, v_N)$ έτσι ώστε να ισχύει $v_i \neq v_j \Leftrightarrow i \neq j$ και όπου τα v_i είναι ονομάτα κορυφών όπως αυτά δημιουργήθηκαν στο προηγούμενο βήμα.

Η δημιουργία των ακμών αυτών θα γίνει με τον επόμενο τρόπο. Πρώτα απ'όλα, αντιγράφουμε ολόκληρη την λίστα ονομάτων των κορυφών (που ουσιαστικά είναι όλοι οι ακέραιοι σε δυαδική μορφή από το 0 έως κάποιο N σε αύξουσα διάταξη) αμέσως δεξιά αυτής. Ετσι δημιουργούμε την συμβολοσειρά $\{\{0, 1, 2, \ldots, N\}, \{0, 1, 2, \ldots, N\}\}$. Επειτα, θα πρέπει να δημιουργήσουμε το τυχαίο στιγμιότυπο των ακμών ενός *οποιουδήποτε* μονοπατιού Hamilton. Αυτό

γίνεται με τον εξής τρόπο: αρχικά μεταθέτουμε τις κορυφές (μόνο στην δεύτερη συμβολοσειρά που αντιγράψαμε-την λίστα ονομάτων των κορυφών την αφήνουμε ως έχει). Εστω ότι δημιουργήθηκε η συμβολοσειρά $\{\{0,1,2,\ldots,N\}, \{\pi(0),\pi(2),\ldots,\pi(N)\}\}$. Με βάση αυτή τη συμβολοσειρά, δημιουργούμε τις ακμές του μονοπατιού Hamilton, δηλαδή δημιουργούμε την συμβολοσειρά $\{\{0,1,2,\ldots,N\}, \{(\pi(0),\pi(1)),(\pi(1),\pi(2)),\ldots,(\pi(N-1),\pi(N)\}\}$. Για να γίνει αυτό, διπλασιάζουμε (δηλαδή απλά αντιγράφουμε) όλες τις κορυφές από την 2^{η} μέχρι την $(N-1)$-οστή, δηλαδή όλες εκτός των $\pi(0)$ και $\pi(N)$, και έπειτα βάζουμε κατάλληλα τις παρενθέσεις.

Αρχικά παρατηρούμε ότι, αφού έχει σταματήσει η παραγωγή ονομάτων για τις κορυφές του γραφήματος, το πρώτο όνομα-αριθμός είναι το 0 ενώ αμέσως μετά το μηδέν αυτό υπάρχει η κατάσταση K που εντόπισε ότι όντος φτάσαμε στο μηδέν. Μεταφέρουμε τον δρομέα-κατάσταση K στο τέλος της συμβολοσειράς ώστε να δημιουργήσουμε το στιγμιότυπο $\{\{0,1,2,\ldots,N\}K\}$:

$$\begin{aligned} K0 &\rightarrow 0K \\ K1 &\rightarrow 1K \\ K, &\rightarrow ,K \\ K\} &\rightarrow \}\Lambda, \end{aligned}$$

Τώρα θα πρέπει να αντιγράψουμε την λίστα ονομάτων των κορυφών, αμέσως δεξιά του χαρακτήρα "," στα δεξιά της καταστάσεως Λ.

$$\left.\begin{aligned} 0'\Lambda &\rightarrow \Lambda 0' \\ 1'\Lambda &\rightarrow \Lambda 1' \\ ,'\Lambda &\rightarrow \Lambda ,' \\ \}'\Lambda &\rightarrow \Lambda \}' \\ \{'\Lambda &\rightarrow \Lambda \{' \end{aligned}\right\} (\textit{αγνόησε ότι έχει ήδη αντιγραφεί})$$

$$\left.\begin{aligned} 0\Lambda &\rightarrow M_0 0' \\ 1\Lambda &\rightarrow M_1 1' \\ ,\Lambda &\rightarrow M_, ,' \\ \}\Lambda &\rightarrow M_\} \}' \\ \{\Lambda &\rightarrow M_\{ \{' \end{aligned}\right\} (\textit{σημάδεψε στην } M \textit{ το τρέχον σύμβολο αντιγραφής})$$

$$\left.\begin{aligned} M_* *' &\rightarrow *' M_* \\ M_*, &\rightarrow \Lambda, * \end{aligned}\right\} (\textit{αντέγραψε το σωστό σύμβολο στη σωστή θέση})$$

όπου στους παραπάνω κανόνες όπου $M_* \in \{M_0, M_1, M_,, M_\}, M_\{\}$ και το σύμβολο $*' \in \{0', 1', ,', \}', \{', \{'\}$. Η παραπάνω διαδικασία τελείωνει μόλις αντιγράψουμε το αριστερό άγκυστρο { καί τίτε μεταβαίνουμε στην επόμενη κατάσταση η οποία και θα ξεκινήσει τη μετάθεση των κορυφών αυτών: $M_\{, \rightarrow \Lambda'\{$

Αφού τελειώσαμε με την αντiγραφή της λίστας ονομάτων των κορυφών του γραφήματος, επόμενο βήμα είναι να μεταθέσουμε τις κορυφές αυτές με στόχο να δημιουργήσουμε τελικώς ένα τυχαίο μονοπάτι Hamilton στο υποκείμενο γράφημα. Για την μετάθεση της λίστας αυτής ακεραίων χρησιμοποιούμε αυτούσιους τους κανόνες μετάθεσης που χρησιμοποιήσαμε και στο πρόβλημα της Διαμέρισης συνόλου ακεραίων αριθμών. Εστω ότι μετά από την διαδικασία αυτή, η οποία αρχίζει από την κατάσταση Λ', βρισκόμαστε στο στιγμιότυπο $\{\{0, 1, \ldots, N\}, \Xi\{\pi(0), \pi(1), \ldots, \pi(N)\}\}$. Θέλουμε να δημιουργήσουμε αντίγραφα των $\pi(1), \ldots, \pi(N-1)$. Αυτό που χρειάζεται να γίνει είναι να σημαδευτούν κατάλληλα οι δύο χαρακτήρες ",", ο ένας που βρίσκεται αμέσως δεξιά του $\pi(0)$ και ο άλλος αμέσως αριστερά του $\pi(N)$. Αυτό γίνεται ούτως ώστε να δημιουργηθούν δύο τερματικά σημεία, εσωτερικά των οποίων αντιγράφουμε (διπλασιάζουμε) όποιον αριθμό βρούμε:

$$
\begin{array}{ll}
\Xi\{ \rightarrow \{\Xi & \Pi 0 \rightarrow 0\Pi \\
\Xi 0 \rightarrow 0\Xi & \Pi 1 \rightarrow 1\Pi \\
\Xi 1 \rightarrow 1\Xi & \Pi, \rightarrow, \Pi \\
\Xi, \rightarrow,' \Pi & \Pi\} \rightarrow P\}
\end{array}
$$

Τώρα, από την κατάσταση P, πρέπει να σημαδέψουμε τον πρώτο χαρακτήρα "," αμέσως αριστερά του $\pi(N)$:

$$
\begin{array}{lcl}
0P & \rightarrow & P0 \\
1P & \rightarrow & P1 \\
,P & \rightarrow & \Sigma,'
\end{array}
$$

Τώρα πλέον μπορούμε από την κατάσταση Σ να αρχίσουμε τον διπλασιασμό των κατάλληλων κορυφών. Αρχικά, φέρνουμε το σύμβολο-δρομέα Σ στην κατάλληλη θέση:

$$
\begin{array}{ll}
0\Sigma \to \Sigma 0 & T0 \to 0T \\
1\Sigma \to \Sigma 1 & T1 \to 1T \\
,\Sigma \to \Sigma,, & T, \to Y, \\
,'\Sigma \to ,'T & T,' \to ,T
\end{array}
$$

Οταν βρισκόμαστε στην κατάσταση Σ και συναντίσουμε τον χαρακτήρα ",", τότε τον διπλασιάζουμε, ώστε ανάμεσα από τους δύο διαδοχικούς χαρακτήρες "," να αντιγραφεί ο εκάστοτε αριθμός. Στην κατάσταση Y αρχίζουμε να αντιγράφουμε τον τρέχοντα αριθμό αμέσως δεξιά του:

$$
\begin{array}{rcl}
0'Y & \to & Y0' \\
1'Y & \to & Y1' \\
0Y & \to & 0'\Phi_0 \\
1Y & \to & 1'\Phi_1 \\
,Y & \to & ,\Psi \\
,'Y & \to & ,'\Psi
\end{array}
$$

Οι καταστάσεις Φ_0 και Φ_1 θα αντιγράψουν το κατάλληλο σύμβολο στην σωστή θέση (οι κανόνες για την κατάσταση Φ_1 είναι αντίστοιχες):

$$
\begin{array}{rcl}
\Phi_0 0' & \to & 0'\Phi_0 \\
\Phi_0 1' & \to & 1'\Phi_0 \\
\Phi_0, & \to & Y, 0
\end{array}
$$

Οταν συναντήσουμε το δεξιό σύμβολο ",' " ή ",", δηλαδή όταν φτάσουμε στην κατάσταση Ψ η οποία και υποδήλώνει ότι η αντιγραφή - διπλασιασμός του αριθμού δεξιά του δρομέα έχει τελειώσει επιτυχώς, πρέπει να ξεσημαδέψουμε όλους τους σημαδεμένους χαρακτήρες. Αυτό γίνεται πολύ εύκολα:

$$
\begin{array}{rcl}
\Psi 0' & \to & \Psi 0 \\
\Psi 1' & \to & \Psi 1 \\
,Psi & \to & ,T
\end{array}
$$

οταν τελειώσει το ξεσημάδεμα των σημαδεμένων χαρακτήρων, μεταβαίνουμε στην κατάσταση T σκοπός της οποίας είναι να προσπεράσει προς τα δεξιά τον τρέχοντα αριθμό και έπειτα να μεταβεί στην κατάσταση Y ώστε να τον διπλασιάσει στα δεξιά αυτού. Η παραπάνω διαδικασία θα τελειώσει όταν, προχωρώντας μέσω της καταστάσεως T, συναντήσουμε το δεξί σημαδεμένο κόμμα ",' " δηλαδή το χαρακτήρα εκείνων ο οποίος διαχωρίζει την κορυφή $\pi(N)$ από την $\pi(N-1)$. Στην περίπτωση αυτή μεταβαίνουμε στην κατάσταση T' και μεταφερόμαστε στο δεξί άκρο της συμβολοσειράς ώστε να αρχίσει η διαδικασία εισαγωγής των κατάλληλων παρενθέσεων για να έχουμε την έγκυρη αναπαράσταησ των ακμών του μονοπατιού Hamilton:

$$
\begin{aligned}
T'0 &\rightarrow 0T' \\
T'1 &\rightarrow 1T' \\
T'\} &\rightarrow \Omega\} \\
\Omega &\rightarrow \omega\tilde{)} \\
0\omega &\rightarrow \omega 0 \\
1\omega &\rightarrow \omega 1 \\
,\omega &\rightarrow \omega', \\
0\omega' &\rightarrow \omega' 0 \\
1\omega' &\rightarrow \omega' 1 \\
,\omega' &\rightarrow \Omega,\tilde{(} \\
\{\omega' &\rightarrow \{\Theta\tilde{(}
\end{aligned}
$$

Παραπάνω, σημαδεύουμε τις παρενθέσεις των ακμώ ώστε να τις αναγνωρίζουμε ως ακμές του μονοπατιού Hamilton, κάτι που θα μας φανεί χρήσιμο στην επόμενη ενότητα.

Αρα έχουμε πλέον δημιουργήσει κάποιο (τυχαίο) μονοπάτι Hamilton της μορφής $\{\{0,1,2,\ldots,N\},\{\Theta(\pi(0),\pi(1)),(\pi(1),\pi(2)),\ldots,(\pi(N-1),\pi(N))\}\}$. Αυτό που μένει για να ολοκληρώσουμε την κατασκευή του γραφήματος είναι η προσθήκη των επιπλέον ακμών που το αποτελούν.

6.4 Προσθήκη των επιπλέον ακμών στο γράφημα

Υστερα από τη δημιουργία των ακμών που αποτελούν το μονοπάτι Hamilton, είναι αναγκαίο να δημιουργήσουμε όλες τις υπόλοιπες ακμές του γραφήματος. Δύο σημεία που πρέπει να προσέξουμε είναι 1) να μην δημιουργήσουμε την ίδια ακμή δύο φορές και 2) να μην δημιουργήσουμε ακμές της μορφής (α,α).

Βρισκόμαστε στην κατάσταση Θ σε μια διαμόρφωση της μορφής $\{\{0,1,2,\ldots,N\}, \{\Theta(\pi(0),\pi(1)),(\pi(1),\pi(2)),\ldots,(\pi(N-1),\pi(N))\}\}$. Η κατάσταση / δρομέας Θ βρίσκεται ακριβώς αριστερά από το αριστερό τέλος της πρώτης ακμής του μονοπατιού Hamilton που δημιουργήσαμε στην προηγούμενη ενότητα. Από την κατάσταση αυτή, μεταφερόμαστε στο δεξί άκρο της επόμενης προς τα δεξιά κορυφής και μεταβαίνουμ ε σε μιά νέα κατάσταση η οποία θα μας βοηθήσει να δημιουργήσουμε τις επιθυμητές ακμές:

$$
\begin{aligned}
\Theta 0 &\rightarrow 0\Theta \\
\Theta 1 &\rightarrow 1\Theta \\
\Theta\tilde{(} &\rightarrow \tilde{(}\Theta \\
\Theta, &\rightarrow ,\Theta \\
\Theta\tilde{)} &\rightarrow \tilde{)}\theta
\end{aligned}
$$

Στο σημείο αυτό είναι απαραίτητο να περιγάψουμε τη διαδικασία της δημιουργίας των επιπλέον επιθυμητών ακμών σε φυσική γλώσσα ώστε στη συνέχεια τα αντίστοιχα σύνολα κανόνων να είναι πιο ευανάγνωστα.

Γενικά σε κάθε βήμα θα θεωρούμε μία μία τις ακμές. Αρχικά, για κάθε μία θεωρούμενη ακμή, θα θεωρούμε τη δεύτερη συνιστώσα της ακμής αυτής. Τη συνιστώσα - κορυφή αυτή θα τη σημαδεύουμε κατάλληλα: αυτή θα είναι η κορυφή από την οποία θα δημιουργήσουμε εξερχόμενες ακμές προς κορυφές λεξικοραφικά μικρότερες και έπειτα προς λεξικογραφικά μεγαλύτερες από αυτήν. Για παράδειγμα, εάν η τρέχουσα ακμή θεώρησης είναι η $(111,100)$ τότε σημαδεύουμε τη δεύτερη συνιστώσα - κορυφή κατάλληλα, π.χ. ως $(111,\overline{100})$, και έπειτα αρχίζουμε τη δημιουργεία των κορυφών με εξερχόμενη την κορυφή αυτή. Αυτό θα γίνει ως εξής: γενικά θα έχουμε δύο καταστάσεις η κάθε μία από τις οποίες θα ενώνει την τρέχουσα κορυφή με μεγαλύτερες ή μικρότερες λεξικογραφικά κορυφές. Αρχικά θα ελέγχουμε εάν είναι δυνατή η δημιουργεία μιας νέας ακμής και εάν ναι τότε θα αποφασίζουμε μη ντετερμινιστικά σε κάθε βήμα εάν επιθυμούμε μια νέα κορυφή της τρέχουσας μορφής (προς μεγαλύτερη ή μικρότερη κορυφή).

Από την κατάσταση θ θα ελέγχουμε τη δεύτερη συνιστώσα της κορυφής που βρίσκεται στα αριστερά αυτής εάν είναι μηδέν, γιατί ως πρώτο βήμα θα δημιουργήσουμε όλες τις επιθυμητές κορυφές από αυτή προς κορυφές λεξικογραφικά μικρότερες:

$$
\begin{aligned}
\tilde{)}\theta &\rightarrow \theta\tilde{)} \\
0\theta &\rightarrow \theta 0 \\
1\theta &\rightarrow 1\theta' \quad (\text{δεν είναι μηδέν}) \\
,\theta &\rightarrow ,\theta'' \quad (\text{είναι μηδέν})
\end{aligned}
$$

άν μεταβούμε στην κατάσταση θ' τότε έχουμε τη δυνατότητα να δημιουργήσουμε επιπλέον ακμή. Σαν επόμενο βήμα, θα αποφασίσουμε μη ντετερμινιστικά για τι νέα αυτή ακμή αφού πρώτα φέρουμε τον δρομέα στη κατάλληλη θέση:

$$
\begin{aligned}
\theta' 1 &\rightarrow 1\theta' \\
\theta' 0 &\rightarrow 0\theta' \\
\theta'\tilde{)} &\rightarrow \tilde{)}\kappa
\end{aligned}
$$

Η κατάσταση κ αποφασίζει μη ντετερμινιστικά για τη προσθήκη μιάς νέας ακμής:

$$
\kappa \rightarrow \kappa_{\nu\alpha\iota} \mid \kappa_{o\chi\iota}
$$

Εάν μεταβούμε στη κατάσταση $\kappa_{\nu\alpha\iota}$, σημαίνει ότι μη ντετερμινιστικά αποφασίσαμε τη δημιουργεία νέας ακμής και τώρα αρχίζουμε τη δημιουργεία της. Αυτό θα γίνει αρχικά αντιγράφοντας τη τρέχουσα συνιστώσα δύο φορές προς τα δεξιά αυτής και βάζοντας τις κατάλληλες παρενθέσεις. Επειτα, μη ντετερμινιστικά θε μειώνουμε το δέυτερο αντίγραφο αυτής τουλάχιστον μιά φορά:

$$
\begin{aligned}
\kappa_{\nu\alpha\iota} &\rightarrow \lambda, (,) \\
\tilde{)}\lambda &\rightarrow \lambda\tilde{)} \\
1'\lambda &\rightarrow \lambda 1' \\
0'\lambda &\rightarrow \lambda 0' \\
1\lambda &\rightarrow 1'\lambda_1 \\
0\lambda &\rightarrow 0'\lambda_0 \\
,\lambda &\rightarrow ,\lambda'
\end{aligned}
$$

Δηλαδή, μεταβαίνουμε στην κατάσταση λ η οποία ψάχνει και βρίσκει τον ρώτο μη σημαδεμένο προς αντιγραφή χαρακτήρα. Τώρα είναι αναγκαίο να τον αντιγράψουμε στο κατάλληλο σημείο (όπως πάντα, περιγράφουμε μόνο την μια περίπτωση, έστω την λ_0):

$$\begin{aligned}
\lambda_0 1' &\rightarrow 1'\lambda_0 \\
\lambda_0 0' &\rightarrow 0'\lambda_0 \\
\lambda_0 \tilde{)}, (&\rightarrow \lambda \tilde{)}, (0
\end{aligned}$$

Μόλις διαπιστώσουμε ότι έχει τελειώσει η διαδικασία αντιγραφής, θα μεταβούμε στην κατάσταση λ', οπότε και θα πρέπει να μετακινηθούμε προς τα δεξιά, ξεσημαδεύοντας ταυτόχρονα όλους τους σημαδεμένους χαρακτήρες, ώστε να διπλασιάσουμε τον αριθμό που μόλις αντιγράψαμε:

$$\begin{aligned}
\lambda' 1' &\rightarrow 1\lambda' \\
\lambda' 0' &\rightarrow 0\lambda' \\
\lambda' \tilde{)}, (&\rightarrow \tilde{)}, (\mu \\
\mu 1 &\rightarrow 1'\mu_1 \\
\mu 0 &\rightarrow 0'\mu_0 \\
\mu_0 1 &\rightarrow 1\mu_0 \\
\mu_0 0 &\rightarrow 0\mu_0 \\
\mu_0, &\rightarrow ,\mu_0' \\
\mu_0' 0 &\rightarrow 0\mu_0' \\
\mu_0' 1 &\rightarrow 1\mu_0' \\
\mu_0') &\rightarrow \mu' 0) \\
0\mu' &\rightarrow \mu' 0 \\
1\mu' &\rightarrow \mu' 1 \\
,\mu' &\rightarrow \mu', \\
0'\mu' &\rightarrow 0'\mu \\
1'\mu' &\rightarrow 1'\mu \\
\mu, &\rightarrow ,\nu
\end{aligned}$$

Στην κατάσταση ν μεταβαίνουμε όταν έχει τελειώσει ο διπλασιασμός της τρέχουσας κορυφής. Δηλαδή, εάν η αρχική μας κατάσταση ήταν η $\tilde{(}\alpha, \beta\tilde{)}\theta$

τώρα βρισκόμαστε στο εξής στιγμιότυπο $\tilde{(}\alpha, \beta\tilde{)}, (\beta, \nu\beta)$. Τώρα είναι αναγκαίο να μειωθεί τουλάχιστον μία φορά η δεύτερη συνιστώσα της τρέχουσας ακμής (έχουμε ήδη ελένξει σε παραπάνω βήμα ότι δεν είναι μηδέν):

$$\begin{aligned}
\nu 1 &\rightarrow 1\nu \\
\nu 0 &\rightarrow 0\nu \\
\nu) &\rightarrow \nu') \\
0\nu' &\rightarrow \nu' 1 \\
1\nu' &\rightarrow 0\nu''
\end{aligned}$$

Τώρα από την κατάσταση ν'' είναι απαραίτητο να φέρουμε το δρομέα στη κατάλληλη θέση ώστε να συνεχιστεί μη ντετερμινιστικά η μείωση της τρέχουσης συνιστώσας, αφού προηγηθεί ο έλεγχος ισότητας με το μηδέν για να διαπιστώσουμε εάν αυτή η μείωση είναι εφικτή:

$$\begin{aligned}
\nu'' 1 &\rightarrow 1\nu'' \\
\nu'' 0 &\rightarrow 0\nu'' \\
\nu'') &\rightarrow \xi)
\end{aligned}$$

Στην κατάσταση ξ ελέγχουμε για ισότητα με το μηδέν:

$$\begin{aligned}
0\xi &\rightarrow \xi 0 \\
1\xi &\rightarrow 1\xi' \quad \text{(δεν είναι μηδέν)} \\
,\xi &\rightarrow ,\xi'' \quad \text{(είναι μηδέν)}
\end{aligned}$$

Εάν μεταβούμε στην κατάσταση ξ', δούμε δηλαδή ότι ο τρέχων αριθμός δεν είναι μηδέν, μεταφέρουμε τον δρομέα στην κατάλληλη θέση ώστε να αποφασίσουμε μη ντετερμινιστικά την μείωση αυτού:

$$\begin{aligned}
\xi' 1 &\rightarrow 1\xi' \\
\xi' 0 &\rightarrow 0\xi' \\
\xi') &\rightarrow)\pi \\
\pi &\rightarrow \nu' \mid \pi'
\end{aligned}$$

Εάν αποφασίσουμε ότι δεν θέλουμε μείωση, τότε μεταβαίνουμε στην κατάσταση π'.

Από την κατάσταση αυτή, την π' δηλαδή, είναι αναγκαίο να δημιουργήσουμε μια διαδικασία η οποία επαναληπτικά και μη ντετερμινιστικά, θα αντιγράφει την τρέχουσα ακμή στα δεξιά της και έπειτα θα μειώνει τη δέυτερη συνιστώσα αυτής. Με τον τρόπο αυτό, δημιουργούνται όλες εκενείνες οι ακμές που εξέρχονται από την τρέχουσα πρώτη συνιστώσα της ακμής προς αυστηρά μικρότερες λεξικογραφικά κορυφές χωρίς να υπάρχει ο κίνδυνος να δημιουργηθούν ίδιες ακμές.

Πιο συγκεκριμένα, έστω ότι βρισκόμαστε σε μια διαμόρφωση της μορφής $\tilde{(}\alpha, \beta\tilde{)}, (\beta, \beta - l)\pi'$. Από την κατάσταση π' αποφασίζουμε μη ντετερμινιστικά εάν θέλουμε νέα ακμή της μορφής $(\beta, \beta - k), k > l : \ \beta - k \geq 0$:

$$\pi' \ \rightarrow \ \pi'_{\nu\alpha\iota} \mid \pi'_{o\chi\iota}$$

Εάν μεταβούμε στην κατάσταση $\pi'_{\nu\alpha\iota}$ τότε είναι αρχικά αναγκαίο να δούμε εάν είναι δυνατή μιά τέτοια δημιουργεία νέας ακμής, με άλλα λόγια να ελέγξουμε για ισότητα με το μηδέν της ακμής που βρίσκεται αμέσως αριστερά της κατάστασης αυτής:

$$\begin{array}{rcll}
\pi'_{\nu\alpha\iota} & \rightarrow & \varpi & \\
)\varpi & \rightarrow & \varpi) & \\
0\varpi & \rightarrow & \varpi 0 & \\
1\varpi & \rightarrow & 1\varpi' & (\text{δεν είναι μηδέν}) \\
,\varpi & \rightarrow & ,\varpi'' & (\text{είναι μηδέν})
\end{array}$$

Εάν δούμε ότι ο αριθμός δεν είναι μηδέν, τότε έιναι αναγκαίο, καθώς έχουμε ήδη αποφασίσει τη δημιουργεία νέας ακμής, να αντιγράψουμε ολόκληρη τη τρέχουσα ακμή στα δεξιά της μας:

$$\begin{array}{rcl}
\varpi' 1 & \rightarrow & 1\varpi' \\
\varpi' 0 & \rightarrow & 0\varpi' \\
\varpi') & \rightarrow & \rho), ()
\end{array}$$

Προχωράμε δηλαδή προς τα δεξιά για να ξεκινήσουμε τη διαδικασία της αντιγραφής χαρακτήρα προς χαρακτήρα μέσω της καταστάσεως ρ και ταυτόχρονα προσθέτουμε τις κατάλληλες παρανθέσεις και τον διαχωριστικό χαρακτήρα ",". Είμαστε έτοιμοι να ξεκινήσουμε την αντιγραφή:

$$\begin{aligned}
0'\rho &\rightarrow \rho 0' \\
1'\rho &\rightarrow \rho 1' \\
,'\rho &\rightarrow \rho ,' \\
0\rho &\rightarrow 0'\rho_0 \\
1\rho &\rightarrow 1'\rho_1 \\
,\rho &\rightarrow ,'\rho_, \\
(\rho &\rightarrow \rho'
\end{aligned}$$

Μόλις βρούμε το πρώτο μη σημαδεμένο σύμβολο, το αποθηκεύουμε στην κατάστασή μας και προχωράμε ώστε να το αντιγράψουμε στο κατάλληλο σημείο (περιγράφουμε μόνο την περίπτωση ρ_0, όπως πάντα οι υπόλοιπες δύο περιπτώσεις είναι τελείως ανάλογες):

$$\begin{aligned}
\rho_0 0' &\rightarrow 0'\rho_0 \\
\rho_0 1' &\rightarrow 1'\rho_0 \\
\rho_0 ,' &\rightarrow ,'\rho_0 \\
\rho_0), (&\rightarrow \rho), (0
\end{aligned}$$

Στην κατάσταση ρ' μεταβαίνουμε μόλις τελειώσει η αντιγραφή αμς οπότε είναι απαρίτητο να μεταφερθούμε στη νέα μας ακμή για να μειώσουμε τουλάχιστον μια φορά τη δεύτερη συνιστώσα αυτής, ξεσημαδεύοντας κατά τη μετάβασή μας όλους του ενδιάμεσους σημαδεμένους χαρακτήρες:

$$
\begin{aligned}
\rho'0' &\rightarrow 0\rho' \\
\rho'1' &\rightarrow 1\rho' \\
\rho',' &\rightarrow ,\rho' \\
\rho'),(&\rightarrow),(\rho'' \\
\rho''0 &\rightarrow 0\rho'' \\
\rho''1 &\rightarrow 1\rho'' \\
\rho'', &\rightarrow ,\rho'' \\
\rho'') &\rightarrow \varrho)
\end{aligned}
$$

Στην κατάσταση ϱ αρχίζουμε τη μείωση κατά μία μονάδα του αριθμού που βρίσκεται αριστερά της καταστάσεως αυτής (ξέρουμε ήδη ότι δεν είναι μηδέν) ώστε αργότερα να ακολουθήσει η μη ντετερμινιστική μείωση αυτού, εφόσον βέβαια κάτι τέτοιο είναι δυνατόν:

$$
\begin{aligned}
0\varrho &\rightarrow \varrho 1 \\
1\varrho &\rightarrow 0\varrho'
\end{aligned}
$$

Από την κατάσταση ϱ είναι απαραίτητο να γυρίσουμε στο δεξί άκρο της τρέχουσας ακμής και να αποφασίσουμε μη ντετερμινιστικά εάν θέλουμε επιπλέον μείωση:

$$
\begin{aligned}
\varrho'0 &\rightarrow 0\varrho' \\
\varrho'1 &\rightarrow 1\varrho' \\
\varrho') &\rightarrow \sigma) \\
\sigma &\rightarrow \sigma_{\nu\alpha\iota} \mid \sigma_{o\chi\iota}
\end{aligned}
$$

Εάν αποφασίσουμε ότι θέλουμε να μειώσουμε επιπλέον τη δεύτερη συνιστώσα της ακμής που μόλις δημιουργήσαμε, δηλαδή μεταβούμε στην κατάσταση $\sigma_{\nu\alpha\iota}$, είναι αναγκαίο πρώτα να δούμε εάν μπορεί αυτή η συνιστώσα να μειωθεί:

$$
\begin{array}{rcll}
\sigma_{\nu\alpha\iota} & \rightarrow & \tau & \\
0\tau & \rightarrow & \tau 0 & \\
1\tau & \rightarrow & 1\tau' & (\text{δεν είναι μηδέν}) \\
,\tau & \rightarrow & ,\tau'' & (\text{είναι μηδέν})
\end{array}
$$

Από την κατάσταση τ', δηλαδή από την κατάσταση εκείνη στην οποία μεταβαίνουμε εάν δούμε ότι ο αριθμός επιδέχεται επιπλέον μείωση, μεταβαίνουμε στο δεξί άκρο ώστε να τον μειώσουμε, αφού έχουμε ήδη αποφασίσει μέσω της $\sigma_{\nu\alpha\iota}$ ότι θέλουμε μείωση:

$$
\begin{array}{rcl}
\tau' 1 & \rightarrow & 1\tau' \\
\tau' 0 & \rightarrow & 0\tau' \\
\tau') & \rightarrow & \varrho)
\end{array}
$$

Η διαδικασία μείωσης είναι δυνατόν να σταματήσει μόνο με δύο τρόπους: είτε στη περίπτωση που αποφασίσουμε μη ντετερμινιστικά να μην μειώσουμε άλλο τον αριθμό αυτό, δηλαδή τη δεύτερη συνιστώσα της τρέχουσας ακμής μας, είτε επειδή δεν μπορούμε να την μειώσουμε άλλο (επειδή προφανώς έχει ήδη γίνει μηδέν). Από τις δύο αυτές περιπτώσεις μεταβαίνουμε σε διαφορετικές καταστάσεις, γιατί οι λειτουργίες σε κάθε περίπτωση είναι διαφορετικές: εάν δούμε ότι ο αριθμός είναι μηδέν (κατάσταση τ''), και άρα δεν επιδέχεται περαιτέρω μείωση, τότε, αφού δεν μπορούμε να δημιουργήσουμε επιπλέον ακμή του τύπου αυτού, είναι απαραίτητο να μεταβούμε στην αρχική μας ακμή προς τα αριστερά, τα άκρα της οποίας είναι ήδη σημαδεμένα ώστενα την αναγνωρίσουμε, και έπειτα να αρχίζουμε την παραγωγή ακμών από την ίδια με πριν κορυφή προς κορυφές λεξικογραφικά *μεγαλύτερες*. Στην περίπτωση όμως που η μείωση σταματήσει επειδή έχουμε αποφασίσει μη ντετερμινιστικά για αυτό, κατάσταση $\sigma_{o\chi\iota}$, τότε είναι απαραίτητο να επαναλάβουμε την παραπάνω μη ντετερμινιστική διαδικασία δημιουργείας επιπλέον ακμής προς κορηφή μικρότερου δείκτη, εφ'όσον φυσικά κάτι τέτοιο είναι δυνατόν, δηλαδή να μεταβούμε στην κατάσταση π'.

Αρχίζουμε πρώτα από την δεύτερη περίπτωση που είναι και η πιο εύκολη:

$$
\begin{array}{rcl}
\sigma_{o\chi\iota}) & \rightarrow &)\pi'
\end{array}
$$

Αυτή η μετάβαση θα συνεχίσει την επαναληπτική και μη ντετερμινιστική διαδικασία δημιουργείας επιπλεόν ακμών προς κορυφές λεξικογραφικά μικρότερες όπως περιγράφηκε παραπάνω.

Εάν δούμε τώρα ότι ο αριθμός είναι μηδέν, και άρα δεν επιδέχεται περαιτέρω μείωση και άρα δεν μπορούμε να δημιουργήσουμε μια νέα ακμή, τότε είναι αναγκαίο όπως είπαμε να γυρίσουμε στην αρχική σημαδεμένη ακμή και να δημιουργήσουμε όλες τις επιθυμητές ακμές δευτέρου είδους. Η ακόλουθες μεταβάσεις ισχύου και για τις περιπτώσεις των ϖ'', ξ'', καταστάσεις στις οποίες μεταβαίνουμε για αντίστοιχο λόγο και θέλουμε να έοχυμε ανάλογη δράση με αυτή της τ'':

$$\begin{aligned} \tau'' &\rightarrow \upsilon \\ \xi'' &\rightarrow \upsilon \\ \varpi'' &\rightarrow \upsilon \end{aligned}$$

Τώρα, από την κατάσταση υ, είναι αναγκαίο να μεταφερθούμε στην πρώτη προς τα αριστερά σημαδεμένη ακμή ώστε να δημιουργήσουμε όλες τις ακμές από τι δεύτερη συνιστώσα αυτής προς κορυφές λεξικογραφικά μεγαλύετρες αυτή τη φορα:

$$\begin{aligned} 0\upsilon &\rightarrow \upsilon 0 \\ 1\upsilon &\rightarrow \upsilon 1 \\ ,\upsilon &\rightarrow \upsilon, \\ )\upsilon &\rightarrow \upsilon) \\ (\upsilon &\rightarrow \upsilon(\\ \tilde{)}\upsilon &\rightarrow \phi \end{aligned}$$

Η κατάσταση ϕ θα ξεκινήσει τη διαδικασία παραγωγής όλων των επιθυμητών ακμών του τύπου που μόλις περιγράψαμε. Πριν όμως αρχίσει αυτή η περιγραφή, είναι αναγκαίο να περιγράψουμε κάποιες περιπτώσεις από παραπάνω που δεν έχουν ακόμα περιγραφει. Αυτές είναι οι καταστάσεις $\pi'_{o\chi\iota}$ και $\kappa_{o\chi\iota}$ αι γενικά έοχυν το ίδιο αποτέλεσμα αλλά σε διαφορετικές φάσεις του υπολογισμού:

$\kappa_{o\chi\iota}$**:** Αρχικά, βρισκόμαστε σε στιγμιότυπο της μορφής $\tilde{(}\alpha, \beta\tilde{)}\kappa$ και χρησιμοποιήσαμε τη μετάβαση $\kappa \rightarrow \kappa_{o\chi\iota}$. Δηλαδή, στο αρχικό μη ντετερμινιστικό βήμα για το εάν θέλουμε ακμή από την κορυφή β προς μιά άλλη

κορυφή λεξικογραφικά μικρότερη, αποφασίσαμε ότι δεν θέλουμε τέτοια ακμή, άρα μεταβαίνουμε σε μια νέα κατάσταση, την ϕ όπως περιγράψαμε πιο πάνω, η οποία θα αρχίσει την μη ντετερμινιστική διαδικασία παραγωγής όλων των επιθυμητών ακμών του τύπου, δηλαδή από την κορυφή β προς κορυφές λεξικογραφικά μεγαλύτερες. Αρα προσθέτουμε στο σύνολο των κανόνων μας τον κανόνα

$$\kappa_{o\chi\iota} \rightarrow \phi$$

$\pi'_{o\chi\iota}$: Και στην κατάσταση αυτή αποφασίσαμε ότι δεν θέλουμε επιπλέον ακμές από την κορυφή β προς μιά άλλη κορυφή λεξικογραφικά μικρότερη, όμως η απόφαση αυτή έγινε σε μεταγενέστερο βήμα, αφού έχει ήδη δημιουργηθεί ένα σύνολο επιθυμητών ακμών του τύπου αυτού. Αρα, είναι απαραίτητο να φέρουμε το δρομέα στη σωστή θέση και να μεταβούμε στην κταάσταση ϕ ώστε να αρχίσει η μη ντετερμινιστική διαδικασία παραγωγής ακμών πό την κορυφή β προς κορυφές λεξικογραφικά μεγαλύτερες. Αρα προσθέτουμε τον παρακάτω κανόνα:

$$\pi'_{o\chi\iota} \rightarrow \upsilon$$

Οταν τελειώσουμε με τη διαδικασία παραγωγής ακμών από τη δεύτερη συνιστώσα της τρέχουσας σημαδεμένης ακμής προς κορυφές λεξικογραφικά μικρότερες αυτής, θα προχωρήσουμε στην δημιουργία των ακμών εκείνων που ξεκινούν και πάλι από τη δεύτερη συνιστώσα της τρέχουσα σημαδεμένης ακμής, κατευθύνονται όμως σε λεξικογραφικά μεγαλύτερες ακμές.

Οι αντίστοιχοι κανόνες είναι ίδιοι όπως και στην παραπάνω περίπτωση, με μιά ουσιώδη διαφορά όμως που θα περιγράψουμε ευθύς αμέσως. Στη προηγούμενη περίπτωση ελέγχαμε για ισότητα με το μηδέν για να δούμε εάν μπορούμε να δημιουργήσουμε μια ειπλέον ακμή. Στην περίπτωση μας όμως ο έλεγχος που πρέπει να κάνουμε είναι για ισότητα με τον αριθμό $|V|$ που κατασκευάσαμε στη πρώτη ενότητα του τρέχοντος κεφαλαίου. Εάν σε κάποιο σημείο βρούμε ότι η τρέχουσα κορυφή έχει μια εξερχόμενη ακμή προς τη κορυφή με τη μέγιστη ετικέτα, δηλαδή την $|V|$, τότε ξέρουμε ότι δεν μπορούμε να δημιουργήσουμε επιπλέον ακμές καθώς δεν έχουμε περιθώριο "αύξησης" του αντίστοιχου δείκτη. Παρακτάτω θα περιγράψουμε μόνο το πώς γίνεται ο έλεγχος ισότητας του τρέχοντος αριθμού με το μέγιστο αυτό αριθμό $|V|$.

Για να γίνει πιο κατανοητή η λειτουργία των παρακάτω κανόνων, θεωρούμε ότι βρισκόμαστε σε μια (τοπική) διαμόρφωση της μορφής $\tilde{(}\alpha,\beta\tilde{)}(\beta,\beta+3)\psi$ και έχουμε αποφασίσει μη ντετερμινιστικά να δημιουργήσουμε μια ακμή από την κορυφή β προς μία κορυφή $\gamma > \beta$ (δηλαδή με μεγαλύτερη ετικέτα - αριθμό),

ακριβώς όπως και πιο πάνω για κορυφές με μικρότερο δείκτη. Αυτό που χρειάζεται να κάνουμε τώρα είναι να συγκρίνουμε τον αριθμό αριστερά της κατάστασης ψ, δηλαδή τον $\beta+3$ στο παράδειγμά μας, με τον *πρώτο προς τα αριστερά αριθμό που θα συναντήσουμε από τη λίστα ονομάτων των κορυφών του γραφήματος*. Δηλαδή, προχωράμε το δρομέα προς τα αριστερά, αφού σημαδέψουμε τον τρέοχντα αριθμό που θέλουμε να συγκρίνουμε, μέχρι να συναντήσουμε τη λίστα κορυφών του γραφήματος που κατασκευάσαμε. Τότε, συγκρίνουμε ψηφίο προς ψηφίο τον αριθμό αυτό με τον αρχικό αριθμό . Η συγκριση συνεχίζει όσο τα αντίστοιχα ψηφία είναι τα ίδια και σταματάει μόλις βρούμε κάποιο διαφορετικο. Στη περίπτωση αυτή διαφέρουν, και άρα μπορούμε να συνεχίσουμε στην παραγωγή της ακμής, στην ερίπτωση όμως που φτάσουμε στο τέλος των δύο αριθμών και δεν έοχυμε βρει καμία διαφορα, γιατί ρπφανώς οι αριθμοί είναι ίδιοι, τότε δεν μπορούμε να δημιουργήσουμε επιπλέον ακμή και προχωράμε στην επόμενη προς τα δεξιά *σημαδεμένη* ακμή, αφού πρώτα ξεσημαδέψουμε τα σημαδεμένα ψηφία των δύ αριθμών που μόλις συγκρίθηκαν. Πιο συγκεκριμένα:

$$
\begin{aligned}
)\psi &\rightarrow \psi) \\
0\psi &\rightarrow \psi_0 0' \\
1\psi &\rightarrow \psi_1 1' \\
,\psi &\rightarrow ,\psi' \\
0\psi_0 &\rightarrow \psi_0 0 \\
1\psi_0 &\rightarrow \psi_1 1 \\
,\psi_0 &\rightarrow \psi_0, \\
)\psi_0 &\rightarrow \psi_0) \\
(\psi_0 &\rightarrow \psi_0(\\
\tilde{)}\psi_0 &\rightarrow \psi_0\tilde{)} \\
\tilde{(}\psi_0 &\rightarrow \phi_0\tilde{(} \\
\},\{\psi_0 0 &\rightarrow \chi_0\},\{
\end{aligned}
$$

Στην κατάσταση χ_0 (ή χ_1, αλλά χωρίς βλάβη της γενικότητας περιγράφουμε τη περίπτωση που το προς σύγκριση ψηφιο είναι το μηδέν) έοχυμε ήδη εισέλθει στη λίστα ονομάτων των κορυφών. Τώρα, είναι αναγκαίο να προσπεράσουμε όλους τους σημαδεμένους χαρακτήρες (που τους έχουμε ήδη δηλαδή συγκρίνει και τους έχουμε βρει ίδιους με αυτούς του αρχικού αριθμού - κόμβου ώστε να συνεχίσουμε τη διαδικασία) κμέχρι να βρούμε τον πρώτο μη

σημαδεμένο και να τον συγκρίνουμε με αυτόν που είναι αποθηκευμένος στη τρε˚χουσα κατάστασή μας:

$$\begin{aligned} 0'\chi_0 &\rightarrow \chi_0 0' \\ 1'\chi_0 &\rightarrow \chi_1 1' \\ 0\chi_0 &\rightarrow 0'\chi' \\ 1\chi_0 &\rightarrow 1\chi'' \end{aligned}$$

Οταν το σύμβολο προς σύγκριση είναι ίδιο με το σύμβολο που βρίσκουμε τότε μεταβαίνουμε στη κατάσταση χ' για να μεταφέρει το δρομέα στη κατάλληλη θέση ώστε να συνεχιστεί η σύγκριση:

$$\begin{aligned} \chi'0' &\rightarrow 0'\chi' \\ \chi'1' &\rightarrow 1'\chi' \\ \chi'\},\{ &\rightarrow \},\{\bar{\chi} \\ \bar{\chi}0 &\rightarrow 0\bar{\chi} \\ \bar{\chi}1 &\rightarrow 1\bar{\chi} \\ \bar{\chi}, &\rightarrow ,\bar{\chi} \\ \bar{\chi}(&\rightarrow (\bar{\chi} \\ \bar{\chi}) &\rightarrow)\bar{\chi} \\ \bar{\chi}\tilde{)} &\rightarrow \tilde{)}\bar{\chi} \\ \bar{\chi}\tilde{(} &\rightarrow \tilde{(}\bar{\chi} \\ \bar{\chi}0' &\rightarrow \psi 0' \\ \bar{\chi}1' &\rightarrow \psi 1' \end{aligned}$$

Οταν όμως τα δύο σύμβολα προς σύγκριση διαφέρουν, τότε μεταβαίνουμε στην κατάσταση χ'' σκοπός της οποίας είναι, αφού ξεσημαδέψει όλα τα σημαδεμένα σύμβολα, να μεταφερθεί στην κατάλληλη θέση ώστε να δημιουργηθεί η ακμή:

$$
\begin{aligned}
\chi''0' &\rightarrow 0\chi'' \\
\chi''1' &\rightarrow 1\chi'' \\
\chi''\},\{ &\rightarrow \},\{\bar{\chi}'' \\
\bar{\chi}''0 &\rightarrow 0\bar{\chi}'' \\
\bar{\chi}''1 &\rightarrow 1\bar{\chi}'' \\
\bar{\chi}'', &\rightarrow ,\bar{\chi}'' \\
\bar{\chi}''(&\rightarrow (\bar{\chi}'' \\
\bar{\chi}'') &\rightarrow)\bar{\chi}'' \\
\bar{\chi}''\tilde{)} &\rightarrow \tilde{)}\bar{\chi}'' \\
\bar{\chi}''\tilde{(} &\rightarrow \tilde{(}\bar{\chi}'' \\
\bar{\chi}0' &\rightarrow 0\zeta \\
\bar{\chi}1' &\rightarrow 1\zeta
\end{aligned}
$$

Τώρα, από την κατάσταση ζ είναι απαραίτητο να ξεσημαδέψουμε τα ήδη σημαδεμένα ψηφία και να μεταβούμε σε μιά κατάσταση η οποία θα μας δημιουργήσει την ακμή ακριβώς όπως και στη προηγούμενη περίπτωση προς λεξικογραφικά μικρότερες κορυφές:

$$
\begin{aligned}
\zeta0' &\rightarrow 0\zeta \\
\zeta1' &\rightarrow 1\zeta \\
\zeta\tilde{)} &\rightarrow \tilde{)}\zeta'
\end{aligned}
$$

Η μοναδική περίπτωση που απομένει να αναλυθεί είναι η κατάσταση ψ'. Στην κατάσταση αυτή μεταβαίνουμε όταν δεν βρούμε άλλα μη σημαδεμένα ψηφία, δηλαδή όταν συναντήσουμε τον χαρακτήρα ",". Τότε, ξέρουμε ότι οι δύο αριθμοί είναι ίσοι, άρα είναι αναγκαίο, αφού πρώτα ξεσημαδέψουμε όλους τους μη σημαδεμένους αριθμούς, να μεταφερθούμε στην επόμενη σημαδεμένη ακμή (ακμή του κύκλου Hamilton):

$$
\begin{array}{rcl}
\psi'0' & \rightarrow & 0\psi' \\
\psi'1' & \rightarrow & 1\psi' \\
\psi'0 & \rightarrow & 0\psi' \\
\psi'1 & \rightarrow & 1\psi' \\
\psi', & \rightarrow & ,\psi' \\
\psi'(& \rightarrow & (\psi' \\
\psi') & \rightarrow &)\psi' \\
\psi'\tilde{)} & \rightarrow & \tilde{)}\psi' \\
\psi'\tilde{(} & \rightarrow & \tilde{(}\bar{\psi} \\
\bar{\psi}0 & \rightarrow & 0\bar{\psi} \\
\bar{\psi}1 & \rightarrow & 1\bar{\psi} \\
\bar{\psi}, & \rightarrow & ,\bar{\psi} \\
\bar{\psi}\tilde{)} & \rightarrow & \tilde{)}\theta
\end{array}
$$

Κεφαλαιο 7

Γραμματικη Με Συμφραζομενα για το Προβλημα της Κλικας

7.1 Εισαγωγή

Στο παρόν κεφάλαιο θα παρουσιάσουμε σύνολα κανόνων τέτοια που να μας επιτρέπουν να δημιουργήσουμε ένα (απλό και μή κατευθυνόμενο) γράφημα $G = (V, E)$ καθώς και έναν ακέραιο αριθμό $k \geq 1$ τέτοιο ώστε το γράφημα G να περιέχει μια κλίκα μεγέθους *τουλάχιστον* k. Πιο συγκεκριμένα, θέλουμε να δημιουργήσουμε όλα τα "ναι" στιγμιότυπα για το πρόβλημα απόφασης της κλίκας, το οποίο διατυπώνεται ως εξής:

> **Κλίκα**: δοθέντος ενός γραφήματος $G = (V, E)$ και ενός ακεραίου $k \geq 1$, αληθεύει ότι το G περιέχει μία *κλίκα* (πλήρες επαγώμενο υπογράφημα) μεγέθους *τουλάχιστον* k;

Το πρώτο ζήτημα που καλούμαστε να επιλύσουμε είναι ο τρόπος με τον οποίο θα αναπαρίσταται το γράφημα. Γενικά θα υιοθετήσουμε τον τρόπο αναπαράστασης του γραφήματος που χρησιμοποιήσαμε και στο πρόβλημα εύρεσης μονοπατιού Hamilton με μια μικρή παραλλαγή: θα πρέπει να θεωρήσουμε και ως μέρος του στιγμιοτύπου και τον ακέραιο k αφού, κατά κάποιον τρόπο, θα προσπαθήσουμε να δημιουργήσουμε μια κλίκα μεγέθους k. Πιο συγκεκριμένα, η γενική αναπαράσταση του γραφήματος θα είναι της μορφής $\{k, \{V\}, \{E\}\}$ όπου $\{V\}$ είναι η λίστα ονομάτων των κορυφών του γραφήματος και $\{E\}$ η λίστα των ακμών στην οποία εμπεριέχεται και μία τουλάχιστον κλίκα μεγέθους τουλάχιστον k.

Αρχικά, είναι αναγκαίο να περιγράψουμε σε φυσική γλώσσα τα απαραίτητα βήματα και έπειτα, στην αντίστοιχη ενότητα, θα περιγραφούν αναλυτικά οι κανόνες εκείνοι που υλοποιούν το εκάστοτε βήμα.

- ▷ Δημιουργούμε, με ένα σύνολο κανόνων, έναν οποιοδήποτε ακέραιο $k > 1$. Ο αριθμός αυτός αντιστοιχεί στο μείζων μέγεθος κλίκας το οποίο θα έχει το γράφημα (δηλαδή σε επόμενα βήματα θα περιγράψουμε σύνολο κανόνων οι οποίοι και θα δημιουργούν μια τυχαία κλίκα μεγέθους ακριβώς k).

- ▷ Δημιουργούμε τις k κορυφές που θα αντιστοιχούν στις κορυφές τις κλίκας. Ουσιαστικά αυτό θα γίνει με την παράθεση όλων των ακεραίων αριθμών από το 0 έως και το $k - 1$.

- ▷ Επειτα, θα σημαδέψουμε το τέλος της λίστας ονομάτων των κορυφών της κλίκας με κάποιον ειδικό χαρακτήρα (π.χ. ";" αντί για το κλασσικό ","), και έπειτα θα συνεχίσουμε να παράγουμε τις υπόλοιπες κορυφές του γραφήματος μέχρι να αποφασίσουμε ότι τελειώσαμε μτ την λίστα ονομάτων των κορυφών, δηλαδή την λίστα V. Οι επόμενοι κόμβοι που θα παράξουμε θα είναι ουσιαστικά ακέραιοι στο δυαδικό σύστημα από το k μέχρι κάποιο επιθυμητό $|V|$.

- ▷ Οταν αποφασίσουμε ότι έχουμε τελειώσει με την παραγωγή των κορυφών της κλίκας καθώς και των υπόλοιπων κορυφών του γραφήματος, αρχίζουμε να παράγουμε την λίστα των ακμών E του γραφήματος:

  - αρχικά παράγουμε τις ακμές που αποτελούν την κλίκα. Δηλαδή για τις k κορυφές που αποτελούν την κλίκα, δημιουργούμε ακμές που να τις ανώνουμε όλες μεταξύ τους (λαμβάνοντας υπ'όψιν ότι η κάθε ακμή πρέπει να εμφανίζεται ακριβώς μία φορά και επιπλέον ότι δεν επιτρέπουμε βρόγχους).

– Για κάθε μία από τις υπόλοιπες $|V| - k$ κορυφές του γραφήματος, δημιουργούμε τις επιθυμητές ακμές (οι οποίες μπορεί να περιλαμβάνουν ακμές και προς τις κορυφές τις κλίκας, δίνοντας την δυνατότητα κάποια κορυφή διαφορετική από αυτες που αποτελούν την κλίκα να συνδεθεί με όλες τις κορυφές τις κλίκας σχηματίζοντας στην ουσία μια κλίκα μεγέθους $k+1$. Αυτό όμως δεν αποτελεί πρόβλημα καθώς επιθυμούμε να παράγουμε γραφήματα τα οποία έχουν κλίκα μεγέθους *τουλάχιστον* k)

Θα πρέπει να σημειώσουμε ότι, όταν παράγουμε όλους τους ακεραίους σε δυαδική μορφή από το 0 έως και το $k-1$, δεν σημαίνει ότι αυτές οι κορυφές θα αποτελούν και την κλίκα. Η κλίκα θα πρέπει να αποτελείται από *οποιεσδήποτε* k κορυφές. Απλά, το παραπάνω βήμα μπορούμε να το εκλάβουμε ώς δέσμευση χώρου για k κορυφές. Αργότερα, όταν θα έχουμε τελειώσει με την παραγωγή όλων των επιθυμητών κορυφών του γραφήματος, θα κάνουμε μια μετάθεση έτσι ώστε να είναι δυνατόν να έρθουν στην αρχική θέση των k κορυφών, οποιοσδήποτε συνδυασμός των $\binom{|V|}{k}$, όπου $|V| \geq k$ το τελικό συνολικό πλήθος των κορυφών του γραφήματος.

7.2 Παραγωγή του μείζονος μεγέθους κλίκας

Θέλουμε ένα σύνολο κανόνων οι οποίοι να δημιουργούν έναν οποιοδήποτε ακέραιο $k \geq 1$ ο οποίος και θα αντιστοιχεί στο επιθυμητό ελάσσον μέγεθος κλίκας για το γράφημα μας. Δηλαδή, θέλουμε ένα σύνολο κανόνων το οποίο να δημιουργεί όλες τις συμβολοσειρές $1\{0,1\}^*$ (μαζί με τις απαραίτητες, για την περιγραφή του όλου στιγμιοτύπου, αγκύλες):

$$\begin{aligned} S &\rightarrow \{1A\} \\ A &\rightarrow 1A \mid 0A \mid B \end{aligned}$$

Ο κανόνας B σηματοδοτεί το τέλος της παραγωγής του αριθμού k. Δηλαδή, μετά το τέλος αυτής της παραγωγής, θα έχουμε δημιουργήσει μια συμβολοσειρά της μορφής $\{101B\}$ για την περίπτωση όπου $k = 5$.

7.3 Παραγωγή της λίστας ονομάτων των κορυφών της κλίκας

Τώρα, μετά της δημιουργεία του επιθυμητού μεγέθους κλίκας, θέλουμε να δημιουργήσουμε την λίστα ονομάτων των κορυφών που αποτελούν την κλίκα: θέλουμε να δημιουγήσουμε k ακριβώς κόμβους οι οποίοι και θα αποτελούν την κλίκα. Αυτό θα γίνει με παράθεση όλων των ακεραίων από $0 \rightarrow (k-1)$. Γενικά, σε κάθε βήμα, θα ελέγχουμε έαν ο τρέχων αριθμός είναι μηδέν και, εάν όχι, θα τον αντιγράφουμε αριστερά και θα τον μειώνουμε κατά ένα. Η διαδικασία θα επαναλαμβάνεται ώσπου να φτάσουμε στον κόμβο 0, οπότε και θα έχουμε τελιώσει με την λίστα ονομάτων των κορυφών της υπο δημιουργείας κλίκας.

Για να δημιουργήσουμε την λίστα ονομάτων των κορυφών της κλίκας, αρχικα δημιουργούμε την συμβολοσειρά $\{k, \{C\}\}$, όπου C είναι η μεταβλητή η οποία θα ξεκινήσει την παραγωγή (στο παράδειγμά μας, αυτό θα αντιστοιχεί στην συμβολοσειρά $\{101, \{C\}\}$).

$$B \rightarrow ,\{C\}$$

Επειτα, αντιγράφουμε τον αριθμό k μέσα στα άγκιστρα τα οποία θα περιέχουν την λίστα ονομάτων των κορυφών του γραφήματος:

$$\begin{aligned}
\{C &\rightarrow E\{ \\
,E &\rightarrow E, \\
0'E &\rightarrow E0' \\
1'E &\rightarrow 1'E \\
0E &\rightarrow 0'E_0 \\
1E &\rightarrow 1'E_1 \\
\{E &\rightarrow \{F
\end{aligned}$$

Δηλαδή, ωε συνήθως, αποθηκεύουμε στην κατάλληλη κατάσταση το τρέχων σύμβολο προς αντιγραφή και πηγαίνουμε να το αντιγράψουμε στην κατάλληλη θέση (η περίπτωση της μεταβλητής E_1 είναι τελείως συμμετρική):

$$
\begin{aligned}
E_0 0' &\rightarrow 0' E_0 \\
E_0 1' &\rightarrow 1' E_0 \\
E_0 , &\rightarrow , E_0 \\
E_0 \{ &\rightarrow \{ C 0
\end{aligned}
$$

Η κατάσταση F υποδηλώνει ότι έχουμε τελειώσει με την τρέχουσα διαδικασία της αντιγραφής. Δηλαδή η τρέχουσα συμβολοσειρά είναι της μορφής $\{Fk', \{k\}\}$ (στο παράδειγμά μας $\{F1'0'1', \{101\}\}$). Τώρα, πρέπει να φέρουμε την κατάσταση / δρομέα F στην κατάλληλη θέση και, ταυτόχρονα, να ξεσημαδέψουμε όλα τα σημαδεμένα σύμβολα του k:

$$
\begin{aligned}
F0' &\rightarrow 0F \\
F1' &\rightarrow 1F \\
F, &\rightarrow , F \\
F\{ &\rightarrow \{ F \\
F0 &\rightarrow 0F \\
F1 &\rightarrow 1F \\
F\} &\rightarrow J\}
\end{aligned}
$$

Στην κατάσταση J πρέπει να μειώσω κατά μία μονάδα τον αριθμό που βρίσκεται στα αριστερά αυτής (ας παρατηρήσουμε ότι σε αυτό το βήμα δεν χρειάζεται να ελένξουμε έαν ο αριθμός που πάμε να μειώσουμε είναι μηδέν. Είμαστε σίγουροι ότι δεν είναι καθώς απότέλεί αντίγραφο του k το οποίο εξ'ορισμού δεν είναι μηδέν). Ας θυμηθούμε ότι προσπαθούμε να δημιουργήσουμε την λίστα ονομάτων των κορυφών της υπο δημιουργείας κλίκας, κάτι που θα γίνει με παράθεση όλων των ακεραίων από το 0 έως το $k-1$ σε δυαδική μορφή:

$$
\begin{aligned}
0J &\rightarrow J1 \\
1J &\rightarrow 0J' \\
J'0 &\rightarrow 0J' \\
J'\} &\rightarrow G\} \\
J', &\rightarrow G,
\end{aligned}
$$

Η διαισθητική λειτουργία της καταστάσεως G είναι να ελέγχει έαν ο αριθμός στα αριστερά της είναι μηδέν. Ουσιαστικά, ελέγχουμε εάν δύναται η αντιγραφή και μείωση του αριθμού (ας θυμηθούμε ότι προσπαθούμε να δημιουργήσουμε την λίστα ονομάτων των κορυφών της κλίκας με αντιγραφή και μείωση κατά μία μονάδα, οπότε ένας τέτοιος έλεγχος είναι απαραίτητος).

Αρα, η τρέχουσα συμβολοσειρά είναι της μορφής (στο παράδειγμά μας) $\{101, \{100G\}\}$ και θέλουμε να ελένξουμε, μέσω της καταστάσεως G εάν ο αριθμός στα αριστερά αυτής είναι μηδέν:

$$\begin{aligned} 1G &\rightarrow 1G_{\text{NO}} \\ 0G &\rightarrow G0 \\ \{G &\rightarrow \{G_{\text{YES}} \\ ,G &\rightarrow ,G_{\text{YES}} \end{aligned}$$

Η κατάσταση G_{NO} σημαίνει ότι ο υποκείμενος αριθμός *δεν* είναι μηδέν (αφού ενδιάμεσα εντοπίσαμε κάποιον άσσο), άρα και μπορούμε να τον αντιγράψουμε και ύστερα να τον μειώσουμε κατά μία μονάδα. Πρώτα φέρνουμε τον δρομέα / κατάσταση στην κατάλληλη θέση:

$$\begin{aligned} 1G_{\text{NO}} &\rightarrow G_{\text{NO}}1 \\ 0G_{\text{NO}} &\rightarrow G_{\text{NO}}0 \\ \{G_{\text{NO}} &\rightarrow \{,H \end{aligned}$$

Δηλαδή, δημιουργήσαμε την συμβολοσειρά $\{101, \{, H100\}\}$. Τώρα πρέπει να αντιγράψουμε τον αριθμό που βρίσκεται *δεξιά* της καταστάσεως H, στα αριστερά αυτής:

$$\begin{aligned} H0' &\rightarrow 0'H \\ H1' &\rightarrow 1'H \\ H0 &\rightarrow H_0 0' \\ H1 &\rightarrow H_1 1' \\ H\} &\rightarrow I\} \\ H, &\rightarrow I, \end{aligned}$$

Βρήκαμε το τρέχον σύμβολο προς αντιγραφή (αγνοώντας τα ήδη αντεγεγραμμένα που τα έχουμε σημαδέψει κατάλληλα) και τώρα πρέπει να αντιγράψουμε το τρέχων σύμβολο στην κάτάλληλη θέση (η περίπτωση της H_1 είναι τελείως συμμετρική και παραλείπεται):

$$\begin{aligned} 0'H_0 &\rightarrow H_0 0' \\ 1'H_0 &\rightarrow H_0 1' \\ ,H_0 &\rightarrow 0,H \end{aligned}$$

Η διαδικασία της αντιγραφής τελειώνει όταν βρούμε διαχωριστικό σύμβολο (“,” ή “}”) και τότε μεταβαίνουμε στην κατάσταση I και πρέπει να κάνουμε τα εξής: 1) να ξεσημαδέψουμε τα σημαδεμένα σύμβολα του αριθμού στα αριστερά μας που μόλις τον αντιγράψαμε, και 2) να μειώσουμε τον αριθμό που μόλις γράψαμε. Αρχικά φέρνουμε την κατάσταση / δρομέα στην κατάλληλη θέση, ξεσημαδεύοντας ταυτόχρονα:

$$\begin{aligned} 0'I &\rightarrow I0 \\ 1'I &\rightarrow I1 \\ ,I &\rightarrow J, \end{aligned}$$

Τώρα, η συμβολοσειρά που δημιουργήσαμε θα είναι (στο παράδειγμά μας) η $\{101, \{100J, 100\}\}$

Η διαδικασία αυτή συνεχίζεται επαναληπτικά παράγοντας, σε κάθε βήμα μιά νέα κορυφή με όνομα κάποιον ακέραιο σε δυαδική μορφή ο οποίος και είναι κατά μία μονάδα μικρότερος από τον προηγούμενο που παράχθηκε. Μετά το τέλος της παραπάνω διαδικασίας παραγωγής της λίστας ονομάτων των κορυφών της κλίκας, θα βρισκόμαστε αναγκαστικά στην κατάσταση G_{YES} η οποία και υποδηλώνει ότι ο υποκείμενος αριθμός είναι μηδέν και άρα ότι έχουμε τελειώσει μη την παραγωγή των κορυφών τις κλίκας. Στο παράδειγμά μας, η συμβολοσειρά θα είναι $\{101, \{G_{\text{YES}}000, 001, 010, 011, 100\}\}$.

7.4 Δημιουργία των επιπλέον κορυφών του γραφήματος

Αφού έχουμε τελειώσει με την παραγωγή των k κορυφών, είναι απαραίτητο να δημιουργήσουμε τις όποιες επιπλέον κορυφές θέλουμε για το γράφημά μας.

Κατ'αρχάς, η συμβολοσειρά μας μέχρι στιγμής (και πάντα στο παράδειγμά μας) θα είναι η $\{101, \{G_{\text{YES}}000, 001, 010, 011, 100\}\}$. Στην κατάσταση G_{YES} θα πρέπει να αποφασίσουμε εάν θέλουμε μιά νέα κορυφή στο γράφημα ή όχι. Παρατηρούμε ότι στο αρχικό αυτό βήμα, η κορυφή αυτή θα έχει ετικέτα 101. Εάν η απάντηση είναι "ναι" τότε πρέπει να τοποθετήσουμε έναν ειδικό χαρακτήρα στο τέλος της λίστας ονομάτων των κορυφών της κλίκας που έχουμε δημιουργήσει (αυτός θα είναι ο "," αντί του συνηθισμένου ",") και ύστερα να δημιουργήσουμε την επόμενη κορυφή. Στη συνέχεια θα ασχολήθούμε με το κατά πόσον θέλουμε επιπλέον κορυφές, αυτή τη φορά επαναληπτικά, χωρίς να χρειαστεί να σημαδεύουμε με ειδικούς χαρακτήρες διαχωρισμού.

Αρχικά, φέρνουμε την κατάσταση / δρομέα στο τέλος της λίστας ονομάτων των κορυφών της κλίκας:

$$\begin{aligned} G_{\text{YES}}0 &\rightarrow 0G_{\text{YES}} \\ G_{\text{YES}}1 &\rightarrow 1G_{\text{YES}} \\ G_{\text{YES}}, &\rightarrow ,G_{\text{YES}} \\ G_{\text{YES}}\} &\rightarrow K\} \end{aligned}$$

Την μεταβλητή K πρέπει να την εκλάβουμε ως την ερώτηση "θέλουμε να δημιουργήσουμε μία νέα κορυφή στο γράφημα;" Αρα η K μπορεί να έχει δύο δυνατές και επιθυμητές παραγωγές:

$$\begin{aligned} K &\rightarrow \Lambda, \\ K &\rightarrow \Lambda' \end{aligned}$$

όπου Λ είναι η κατάσταση δημιουργείας μιάς νέας κορυφής (για το λόγο αυτό τοποθετούμε και τον νέο χαρακτήρα διαχωρισμού ",") ενώ Λ' είναι η κατάστση όπου έχουμε αποφασίσει ότι τελειώσαμε με την παραγωγή όλων των κορυφών του γραφήματος, και είμαστε σε θέση να προχωρήσουμε στην παραγωγή των ακμών.

Στην κατάσταση Λ πρέπει να αντιγράψουμε τον δυαδικό αριθμό που βρίσκεται αριστερά της και ύστερα να τον αυξήσουμε κατά μία μονάδα: ο νεός αθτός αριθμός σε δυαδική αναπαράσταση θα αντιστοιχεί στο όνομα της νεάς επιθυμητής κορυφής του γραφήματος. Αρχικά, θέλουμε εκείνο το σύνολο κανόνων ώστε να αντιγράψουμε τον αριθμό που βρίσκεται στα αριστερά της καταστάσεως Λ, στα δεξιά αυτής:

$$
\begin{aligned}
0'\Lambda &\rightarrow \Lambda 0' \\
1'\Lambda &\rightarrow \Lambda 1' \\
0\Lambda &\rightarrow 0'\Lambda_0 \\
1\Lambda &\rightarrow 1'\Lambda_1 \\
,\Lambda &\rightarrow ,M \\
\{\Lambda &\rightarrow \{M
\end{aligned}
$$

Με τους παραπάνω κανόνες σημαδεύουμε το πρώτο μη σημαδεμένο σύμβολο, δηλαδή το τρέχων σύμβολο αντιγραφής αφού κάθε σημαδεμένο σύμβολο το έχουμε ήδη αντιγράψει, μέχρι να συναντήσουμε τον χαρακτήρα διαχωρισμού "," ή "{" που σημαίνει ότι έχουμε τελειώσει με την αντιγραφή του αριθμού και θέλουμε τώρα να αυξήσουμε κατά μία μονάδα το νέα αντίγραφο αυτού, ώστε να ανταποκρίνεται στον αμέσως επόμενο αριθμό, σε δυαδική αναπαράσταση, ως το όνομα της επόμενης κορυφής. Τώρα πρέπει να πάμε και να αντιγράψουμε τον τρέχων χαρακτήρα αντιγραφής στην κατάλληλη θέση (η περίπτωση για την κατάσταση Λ_1 είναι τελείως συμμετρική):

$$
\begin{aligned}
\Lambda_0 0' &\rightarrow 0'\Lambda_0 \\
\Lambda_0 1' &\rightarrow 1'\Lambda_0 \\
\Lambda_0, &\rightarrow \Lambda, 0 \\
\Lambda_0\mathbf{,} &\rightarrow \Lambda\mathbf{,}0
\end{aligned}
$$

Οταν έχουμε τελειώσει με την αντιγραφή (δηλαδή βρούμε το αριστερό διαχωριστικό του τρέχοντος αριθμού προς αντιγραφή) θα μεταβούμε στην κατάσταση M. Στην κατάσταση αυτή πρέπει να μετακινήσουμε τον δρομέα στο δεξιό τέλος του αριθμού που δημιουργήσαμε και έπειτα να τον αυξήσουμε κατά μία μονάδα. Κατα τη διάρκεια της μετακίνησης αυτής ξεσημαδεύουμε όλα τα σημαδεμένα σύμβολα:

$$
\begin{aligned}
M0' &\rightarrow 0M \\
M1' &\rightarrow 1M \\
M, &\rightarrow ,M \\
M\mathbf{,} &\rightarrow M\mathbf{,} \\
M0 &\rightarrow 0M \\
M1 &\rightarrow 1M \\
M\} &\rightarrow N\}
\end{aligned}
$$

Στο παράδειγμά μας, η συμβολοσειρά που θα έχουμε δημιουργήσει θα έχει τώρα την μορφή $\{101, \{000, 001, 010, 011, 100\mathbf{,}100N\}\}$. θέλουμε εκείνο το σύνολο κανόνων που θα αυξάνουν στον αριθμό δεξιά της καταστάσεως N κατά μία μονάδα:

$$
\begin{aligned}
1N &\rightarrow N0 \\
0N &\rightarrow 1\Xi \\
,N &\rightarrow ,1\Xi \\
\mathbf{,}N &\rightarrow \mathbf{,}1\Xi
\end{aligned}
$$

Η κατάσταση Ξ σηματοδοτεί και το τέλος της αύξησης του τρέχοντος αριθμού κατά μία μονάδα. Μία παρατήρηση: όταν δημιουργούμε τις k πρώτες κορυφές και σημαδέυουμε το τέλος αυτών, χρησιμοποιούμε μόνο $\log k$ ψηφία. Αυτό τα ψηφία ενδεχομένως να μην είναι αρκετά για να κατασκευάσουμε επιπλέον κορυφές, εκτός της κλίκας. Για παράδειγμα μπορέι να θέλουμε μια κλίκα μεγέθους 8 αλλά να θέλουμε ένα γράφημα με 128 κορυφές. Για το λόγο αυτή, στην παραπάνω διαδικασία αύξησης, όταν δούμε ότι αυξάνοντας έναν αριθμό φτάσουμε στο αριστερό άκρο του συναντώντας τον αντίστοιχο χαρακτήρα διαχωρισμού, τότε προσθέτουμε ένα επιπλέον ψηφίο. Αργότερα, θα εισάγουμε μια λειτουργία η οποία μετατρέπει όλους τους αριθμούς σε αριθμούς με ίσο αριθμό ψηφίων.

Κατόπιν μετακινούμε την κατάσταση / δρομέα Ξ στο δεξί τέλος του αριθμού που μόλις αυξήσαμε και μεταβαίνουμε σε μιά νέα κατάσταση με την οποία, όπως και πριν, μη ντετερμινιστικά θα αποφασίζουμε εάν θέλουμε νέα κορυφή ή όχι:

$$
\begin{aligned}
\Xi 0 &\rightarrow 0\Xi \\
\Xi\} &\rightarrow \Pi\} \\
\Pi &\rightarrow \Lambda, \\
\Pi &\rightarrow \Lambda'
\end{aligned}
$$

Οπως και πριν, η κατάσταση Λ αντιστοιχεί στην κατάσταση δημιουργίας μιάς νέας κορυφής με τον τρόπο που περιγράφηκε παραπάνω (δηλαδή αντιγραφή του αριθμού στά αριστερά της μεταβλητής Λ και ύστερα αύξηση κατά μία μονάδα). Οταν αποφασίσουμε πως έχουμε τελειώσει με την παραγωγή όλων των επιθυμητών κορυφών για το γράφημά μας, τότε μεταβαίνουμε στην κατάσταση Λ', η οποία σηματοδοτεί το τέλος της διαδικασίας παραγωγής των κορυφών. Στο παράδειγμά μας, έστω ότι επιθυμούμε δύο επιπλέον κορυφές πέραν αυτών της κλίκας, τότε η συμβολοσειρά μετά το πέρας της δημιουργείας όλων των κορυφών θα είναι η $\{101, \{000, 001, 010, 011, 100\mathbf{,}101, 110\Lambda'\}\}$.

Τώρα, από την κατάσταση Λ' είναι απαραίτητο να περιγράψουμε μια διαδικασία η οποία να μετατρέπει όλους τους αριθμούς σε αριθμούς με ίδιο μέγεθος. Η διαδικασία είναι επαναληπτική. Το πλήθος των ψηφίων που είναι αναγκαίο να έχει ο κάθε αριθμός είναι όσο και το πλήθος των ψηφίων του *τελευταίου* αριθμού. Όσο προχωράμε προς τα *αριστερά*, τόσο *μειώνεται* ο αριθμός των ψηφίων των αριθμών. Αυτό που κάνουμε είναι το εξής: ελέγχουμε ανα δύο διαδοχικούς αριθμούς. Αυτοί είτε θα έχουν το ίδιο αριθμό ψηφίων, είτε θα διαφέρουν κατά κάποιον αριθμό ψηφίων. Εάν έχουν τον ίδιο αριθμό ψηφίων, καμία ενέργεια δεν απαιτείται. Εάν όμως δούμε ότι διαφέρουν, τότε προσθέτουμε τους απαραίτητους χαρακτήρες 0 στο αριστερό άκρο του αριστερού αριθμού που συγκρίνουμε. Συνεχίζουμε μέχρι να φτάσουμε στο αριστερό άκρο της λίστας ονομάτων των κορυφών.

Πιο αναλυτικά:

$$
\begin{aligned}
\Lambda' &\rightarrow \iota \\
0\iota &\rightarrow \iota'0' \\
1\iota &\rightarrow \iota'1' \\
,\iota &\rightarrow \iota, \\
\mathbf{,}\iota &\rightarrow \iota\mathbf{,}
\end{aligned}
$$

$$
\begin{aligned}
1\iota' &\rightarrow \iota'0\\
0\iota' &\rightarrow \iota'1\\
,\iota' &\rightarrow \iota'',\\
\textbf{,}\iota' &\rightarrow \iota''\textbf{,}\\
1'\iota'' &\rightarrow \iota''0\\
0'\iota'' &\rightarrow \iota''1\\
1\iota'' &\rightarrow 1'\lambda\\
0\iota'' &\rightarrow 0'\lambda\\
,\iota'' &\rightarrow ,0\mu\\
\textbf{,}\iota'' &\rightarrow \iota\textbf{,}\\
\{\iota'' &\rightarrow \{0\kappa
\end{aligned}
$$

Η ερμηνεία των παραπάνω κανόνων είναι η εξής: έστω ότι θέλουμε να συγκρίνουμε τον αριθμό των ψηφίων δύο αριθμών α, β γνωρίζοντας ότι είτε $|\alpha| = |\beta|$ είτε ότι $|\alpha| = |\beta| - k$. Στην κατάσταση ι σημαδεύουμε τον πρώτο προς τα αριστερά μη σημαδεμένο ψηφίο του β και μεταβαίνουμε στην κατάσταση ι'. Σκοπός της καταστάσεως αυτής είναι να μεταβεί στο αντίστοιχο σημείο του αριθμού α και να ελένξει το αντίστοιχο ψηφίο. Εάν βρεί 0 ή 1 τότε συνεχίζει τη παραπάνω διαδικασία για το επόμενο ζεύγος ψηφίων των δύο αριθμών. Εάν όμως βρεί κάποιον διαχωριστικό χαρακτήρα, τότε γνωρίζει ότι ο α είναι κατά ένα ψηφίο μικρότερος του β οπότε προσθέτει το κατάλληλο μηδενικό, χωρίς να αλλάξει η ουσία του αριθμού και μεταβαίνει στην κατάσταση σύγκρισης του α με όποιον αριθμό υπάρχει στα δεξιά του. Εάν κατά τη διαδικασία εύρεσης του επόμενου ψηφίου από την ι του αριθμού β συναντίσουμε διαχωριστικό χαρακτήρα, τότε ξέρουμε ότι η συγκριση έχει τελειώσει: $|\alpha| = |\beta|$. Οπότε, προχωράμε στη σύκριση του α με τον αριθμό στα δεξιά του (εάν βέβαια υπάρχει).

Απομένει να παρουσιάσουμε τη λειτουργία των καταστάσεων λ και μ των οποίων η διαισθητική λειτουργία είναι η εξής: στην κατάσταση λ μεταβαίνουμε όταν βρούμε ψηφίο στον αριθμό α και άρα είναι απαραίτητο να γυρίσουε προς τον αριθμό β ώστε να πάρουμε το επόμενο ψηφίο αυτού προς σύγκριση, εάν βέβαια υπάρχει:

$$
\begin{aligned}
\lambda 0' &\rightarrow 0'\lambda \\
\lambda 1' &\rightarrow 1'\lambda \\
\lambda, &\rightarrow ,\lambda' \\
\lambda\mathbf{,} &\rightarrow \mathbf{,}\lambda' \\
\lambda' 0 &\rightarrow 0\lambda' \\
\lambda' 1 &\rightarrow 1\lambda' \\
\lambda' 0' &\rightarrow \iota 0' \\
\lambda' 1' &\rightarrow \iota 1'
\end{aligned}
$$

Αντίστοιχα, στην κατάσταση μ μεταβαίνουμε όταν, αντι για ψηφίο, συναντήσουμε διαχωριστικό χαρακτήρα διαφορετικό της δεξιάς αγκύλης, οπότε προσθέτουμε μηδέν και, αφού δεν βρίκαμε αγκύλη, σημαίνει ότι υπάρχει άλλος αριθμός αριστερά του α με τον οποίον είναι απαραίτητο να συγκρίνουμε ως προς το πλήθος των ψηφίων. Οπότε και μεταβαίνουμε στο δεξί άκρο του αριθμού α, ξεσημαδεύοντας όλους τους ενδιάμεσους χαρακτήρες (μαζί και του β) ώστε να αρχίσει η διαδικασία σύγκρισης του α με τον αριθμό στα δεξιά του:

$$
\begin{aligned}
\mu 0' &\rightarrow 0\mu \\
\mu 1' &\rightarrow 1\mu \\
\mu, &\rightarrow ,\mu' \\
\mu\mathbf{,} &\rightarrow \mathbf{,}\mu' \\
\mu' 0 &\rightarrow 0\mu' \\
\mu' 1 &\rightarrow 1\mu' \\
\mu' 0' &\rightarrow 0\mu' \\
\mu' 1' &\rightarrow 1\mu' \\
\mu', &\rightarrow \vartheta, \\
\mu'\} &\rightarrow \vartheta\} \\
0\vartheta &\rightarrow \vartheta' 0' \\
1\vartheta &\rightarrow \vartheta' 1' \\
,\vartheta &\rightarrow \iota, \\
\mathbf{,}\vartheta &\rightarrow \iota\mathbf{,}
\end{aligned}
$$

Στην τρέχουσα κατάσταση, έχουμε δημιουργήσει όλες τις επιθυμητές κορυφές με τον ίδιο αριθμό ψηφιών. Ενα επιπλέον βήμα είναι αναγκαίο: ενώ έχουμε δημιουργήσει ουσιαστικά *χώρο* για τις κορυφές της κλίκας, εντούτοις δεν έχουμε ακόμα αποφασίσει *ποιές* ακριβώς κορυφές θα αποτελούν την κλίκα. Ετσι, στο σημείο αυτό *μεταθέτουμε* τις κορυφές που έχουμε δημιουργήσει (τις k αρχικές συν τις επιπλέον που δημιουργήσαμε κατόπιν) ούτως ώστε να φέρουμε στην αρχική θέση των k κορυφών (που θα διαχωρίζονται με το ειδικό σύμβολο ",") εκείνες τις k κορυφές που αποτελούν την επιθυμητή μας κλίκα. Αυτό γίνεται με το ίδιο σύνολο κανόνων μετάθεσης που έχουμε ήδη περιγράψει στο πρόβλημα της Διαμέρισης και έχουμε επίσης χρησιμοποιήσει στο πρόβλημα του Μονοπατιού Hamilton.

Στο παράδειγμά μας, έστω ότι από τις εφτά κορυφές του γραφήματος, θέλουμε την κλίκα μας να την αποτελούν οι εξής πέντε κορυφές: 0,2,3,5 και 6 (όχι απαραίτητα με αύξουσα σειρά). Η μετάθεση των κορυφών μας δίνει τη δυνατότητα να δημιουργήσουμε την επιθυμητή συμβολοσειρά: αυτή θα είναι η $\{101, \{000, 010, 011, 101, 110, 001, 100\}\}$. Επίσης, οποιοσδήποτε άλλος συνδυασμός των πέντε αυτών κορυφών θα ήταν έγκυρος στο παράδειγμά μας.

7.5 Παραγωγή των ακμών του γραφήματος

Αφού έχουμε τελειώσειμε την παραγωγή των κορυφών του γραφήματος και έχουμε επιπλέον αποφασίσει ποιές k από τις $|V|$ κορυφές θα αποτελούν την επιθυμητή κλίκα, πρέπει να συνεχίσουμε τη κατασκευή του γραφήματος με το να προσθέσουμε τις απαραίτητες και τις επιθυμητές ακμές σε αυτό. Πρώτα πρέπει να δημιουργήσουμε τις ακμές εκείνες που θα αποτελούν την κλίκα (πλέον ξέρουμε ποιές κορυφές άποτελούν την κλίκα μας).

7.5.1 Δημιουργία των ακμών της κλίκας

Θέλουμε να δημιουργήσουμε όλες εκείνες τις ακμές μεταξύ των k κορυφών που αποτελούν την κλίκα. Σε πολύ γενικές γραμμές αυτό θα γίνει δημιουργώντας ακμές από κάθε κορυφή με όσες βρίσκονται στα δεξιά αυτής στην λίστα ονομάτων κορυφών της κλίκας. Πιο συγκεκριμένα, έστω ότι η τρέχουσα συμβολοσειρά μας είναι η $\{k, \{v_1, \ldots, v_k, v_{k+1}, \ldots, v_{|V|}\}\}$ όπου οι k πρώτες κορυφές, όπως υποδηλώνει εξ'άλλου και ο χαρακτήρας διαχωρισμού, είναι οι κορυφές της κλίκας μετά την μετάθση που έχουμε κάνει στην λίστα των κορυφών. Αρχίζοντας από την κορυφή v_1, επαναληπτικά για κάθε κορυφή v_i της κλίκας, δημιουργούμε όλες εκείνες τις ακμές της μορφής $\{v_i, v_j\} \ : \ i < j \leq k$. Με τον τρόπο αυτό δημιουργούνται όλες οι απαραίτητες ακμές και επιπλέον δεν διατρέχουμε κίνδυνο να δημιουργήσουμε την ίδια ακμή δύο ή περισσοτερες

φορές.

Πριν περιγράψουμε το σύνολο εκείνων των κανόνων που δημιουργούν αυτές τις απαραίτητες ακμές, είναι αναγκαίο να περιγραφεί γενικά και σε φυσική γλώσσα ο (επαναληπτικός) τρόπος με τον οποίο αυτές θα δημιουργηθούν.

1. Αρχικά, βρίσκουμε την τρέχουσα κορυφή από την οποία θα δημιουργήσουμε τις απαραίτητες ακμές με κορυφές στα δεξιά αυτής. Εστω v_i η κορυφή αυτή.

2. Ελέγχουμε εάν αυτή η κορυφή είναι η τελευταία κορυφή της λίστας κορυφών της κλίκας, δηλαδή η κορυφή v_k. Εάν όχι, τότε αντιγράφουμετις κορυφές $v_i, \ldots, v_k$ στην λίστα των ακμών του γραφήματος.

3. Αφού τελειώσουμε με την αντιγραφή των κατάλληλων κορυφών στην κατάλληλη θέση στην λίστα ακμών, προσθέτουμε την κορυφή v_i ενδιάμεσα από δύο διαδοχικές κορυφές της λίστας που μόλις αντιγράψαμε. Δηλαδή, εάν έχουμε αντιγράψει για παράδειγμα την λίστα v_i, v_{i+1}, v_{i+2} και $v_{i+3} = v_k$ τότε η συμβολοσειρά που δημιουργούμε είναι η v_i, v_{i+1}, v_i, v_{i+2}, v_i, v_{i+3}.

4. Τέλος, δημιουργούμε τις κατάλληλες ακμές με την προσθήκη των κατάλληλων συμβόλων { και }. Δηλαδή, μετασχηματίζουμε την προηγούμενη συμβολοσειρα στην $\{v_i, v_{i+1}\}$, $\{v_i, v_{i+2}\}$, $\{v_i, v_{i+3}\}$ μέσα στην λίστα των ακμών του γραφήματος.

Για να μπορέσουμε να περιγράψουμε τυπικά σε μορφή συνόλου κανόνων όλα τα παραπάνω, είναι απαραίτητο να σημαδεύουμε σε κάθε στάδιο της προηγούμενης επαναληπτικής διαδικασίας τον αριστερό χαρακτήρα διαχωρισμού της τρέχουσας κορυφής v_i (ας θυμηθούμε ότι στην παραπάνω περιγραφή, η κορυφή v_i αντιστοιχούσε στην τρέχουσα κορυφή από την οποία θα δημιουργήσουμε τις κατάλληλες ακμές με τις υπόλοιπες κορυφές της κλίκας στα δεξιά αυτής). Αυτό για να ανιχνεύουμε εύκολα τις κορυφές που θα αποτελούν την λίστα που θα αντιγράψουμε στο επόμενο βήμα στην λίστα των ακμών ούτως ώστε να δημιουργηθούν οι απαραίτητες ακμές (αφού ο δεξιός χαρακτήρας τερματισμού είναι ήδη σημαδεμένος με τον χαρακτήρα "**,**" που αντιστοιχεί στο τόλος της λίστας κορυφών της κλίκας).

Στο παράδειγμά μας, $\{101, \{000, 010, 011, 101, 110$**,**$001, 100\}\}$, έστω ότι χωρίς βλάβη τις γενικότητας και για λόγους επεξήγησης, θέλουμε να δημιουργήσουμε με τον παραπάνω τρόπο όλες τις ακμές τις κλίκας με μια τερματική κορυφή την 011 και με τις άλλες τερματικές κορυφές όσες έπονται αυτής στην λίστα κορυφών της κλίκας, δηλαδή τις 101 και 110. Σημεδεύουμε τον αριστερό χαρακτήρα διαχωρισμού της τρέχουσας κορυφής 011, δηλαδή

η συμβολοσειρά μας γίνεται $\{101, \{000, 010,' 011, 101, 110,001, 100\}\}$. Αντιγράφουμε τις ενδιάμεσες των δύο σημαδεμένων χαρακτήρων διαχωρισμού κορυφές στην λίστα ακμών. Η συμβολοσειρα θα έχει την εξής μορφή: $\{101, \{000, 010,' 011, 101, 110, 001, 100\}, \{011, 101, 110\}\}$. Έπειτα, αντιγράφουμε την κορυφή 011 ανάμεσα από δύο διαδοχικές κορυφές της λίστας που μόλις αντιγράψαμε: $\{101, \{000, 010,' 011, 101, 110, 001, 100\}, \{011, 101, 011, 110\}\}$. Τέλος, προσθέτουμε τα άγκυστρα στις κατάλληλες θέσεις και έχουμε την τελική επιθυμητή συμβολοσειρά: $\{101, \{000, 010,' 011, 101, 110 , 001, 100\}, \{\{011, 101\}, \{011, 110\}\}\}$

Οταν αποφασίσουμε ότι έχουμε δημιουργήσει όλες τις επιθυμητές κορυφές, και μετά την μετάθεση αυτών, θα βρισκόμαστε στο στιγμιότυπο της μορφής $\{k, \{v_1, \ldots, v_k, v_{k+1}, \ldots, v_{|V|}\}\Sigma'\}$. Στο σημείο αυτό πρέπει να δημιουργήσουμε την λίστα ακμών του γραφήματος:

$$\Sigma' \rightarrow \quad \Sigma, \{\}$$

δηλαδή δημιουργούμε τη συμβολοσειρά $\{k, \{v_1, \ldots, v_k, \ldots, v_{|V|}\}, \Sigma\{\}\}$ στην οποία τα άγκυστρα θα περιέχουν τιν λίστα των ακμών του γραφήματος.

Στο σημείο αυτό βρίσκουμε και σημαδεύουμε την τρέχουσα κορυφή v_i από την οποία θα δημιουργήσουμε τις απαραίτητες ακμές με κορυφές της κλίκας στα δεξιά αυτής. Το σημάδεμα της τρέχουσας κορυφής v_i θα γίνεται σημαδεύοντας τον πρώτο χαρακτήρα τερματισμού αμέσως *αριστερά* της κορυφής αυτής. Τέτοιος χαρακτήρας διαχωρισμού είναι ο "," τον οποίο και θα σημαδεύουμε ως ",'" ή ο { τον οποίον και αφήνουμε ως έχει. Κάθε φορά δηλαδή, σαρώνουμε την λίστα κορυφών τις κλίκας από τα δεξιά προς τα αριστερά, ψάχνοντας είτε για κάποια ήδη σημαδεμένη κορυφή την οποία και ξεσημαδεύουμε για να σημαδέψουμε την επόμενή της, με την προϋπόθεση ότι αυτή δεν είναι η v_k, είτε, εάν δε βρούμε σημαδεμένη κορυφή, ξέρουμε ότι είμαστε στο πρώτο βήμα της διαδικασίας όποτε η v_i είναι ουσιαστικά η v_1 και κατά συνέπεια αντιγράφουμε ολόκληρη την λίστα των κορυφών της κλίκας:

$$
\begin{aligned}
\}\Sigma &\rightarrow \Sigma\} \\
0\Sigma &\rightarrow \Sigma 0 \\
1\Sigma &\rightarrow \Sigma 1 \\
,\Sigma &\rightarrow \Sigma, \\
\mathbf{,}\Sigma &\rightarrow \Sigma\mathbf{,} \\
,'\Sigma &\rightarrow ,\Upsilon' \\
\{\Sigma &\rightarrow \{\Upsilon
\end{aligned}
$$

Η μεταβλητή Υ' έχει την εξής λειτουργία: αφού ξεσημαδέψουμε την ήδη σημαδεμένη κορυφή (έχουμε τελειώσει με τις ακμές από την κορυφή αυτή) προχωράμε για να σημαδέψουμε την επόμενή της. Στο σημείο αυτό πρέπει να ελένξουμε έαν αυτή η επόμενη είναι ή όχι η τελευταία της λίστας των κορυφών της κλίκας, οπότε και τελιώσαμε με την παραγωγή ακμων! Πιο συγκεκριμένα:

$$
\begin{aligned}
\Upsilon' 0 &\rightarrow 0\Upsilon' \\
\Upsilon' 1 &\rightarrow 1\Upsilon' \\
\Upsilon', &\rightarrow ,'\Upsilon \\
\Upsilon'\mathbf{,} &\rightarrow X\mathbf{,} \\
\Upsilon'\} &\rightarrow X\}
\end{aligned}
$$

όπου στην κατάσταση X μεταβαίνουμε όταν αντιληφθούμε ότι φτάσαμε στην τελέυταία κορυφή της λίστας κορυφών της κλίκας, δηλαδή στην v_k.

Αντίθετα, στην κατάσταση Υ πρέπει να αντιγράψουμε στην λίστα των ακμών του γραφήματος οτιδήποτε περικλείεται από τους δύο σημαδεμένους διαχωριστικούς χαρακτήρες "$,'$" και "$,$". Αρχικά, μεταφέρουμε τον δρομέα αμέσως πρίν από τον δεύυερο χαρακτήρα διαχωρισμού "$,$":

$$
\begin{aligned}
\Upsilon 0 &\rightarrow 0\Upsilon \\
\Upsilon 1 &\rightarrow 1\Upsilon \\
\Upsilon, &\rightarrow ,\Upsilon \\
\Upsilon\mathbf{,} &\rightarrow \Phi\mathbf{,} \\
\Upsilon\} &\rightarrow \Phi\}
\end{aligned}
$$

Τώρα στην κατάσταση Φ είμαστε πλέον σε θέση να αντιγράψουμε όλους τους ενδιάμεσους χαρακτήρες στην κατάλληλη θέση, δηλαδή μέσα στην λίστα των ακμών του γραφήματος. Πρώτα σημαδεύουμε το τρέχον σύμβολο προς αντιγραφή:

$$
\begin{aligned}
0'\Phi &\rightarrow \Phi 0' \\
1'\Phi &\rightarrow \Phi 1' \\
,'\Phi &\rightarrow \Phi ,' \\
0\Phi &\rightarrow 0'\Phi_0 \\
1\Phi &\rightarrow 1'\Phi_1 \\
,\Phi &\rightarrow ,''\Phi_, \\
,'\Phi &\rightarrow ,'\Psi
\end{aligned}
$$

όπου η κατάσταση Ψ σηματοδοτεί το τέλος της διαδικασίας αντιγραφής οπότε πρέπει να ξεσημαδέψουμε όλους τους σημαδεμένους χαρακτήρες. Τώρα αντιγράφουμε το σημαδεμένο σύμβολο στην κατάλληλη θέση (οι περιπτώσεις των κανόνων $\Phi 1'$ και $\Phi ,'$ είναι τελείως ανάλογες):

$$
\begin{aligned}
\Phi_0 0' &\rightarrow 0'\Phi_0 \\
\Phi_0 1' &\rightarrow 1'\Phi_0 \\
\Phi_0 ,'' &\rightarrow ,''\Phi_0 \\
\Phi_0 0 &\rightarrow 0\Phi_0 \\
\Phi_0 1 &\rightarrow 1\Phi_0 \\
\Phi_0 , &\rightarrow ,\Phi_0 \\
\Phi_0\},\{ &\rightarrow \Omega\},\{0
\end{aligned}
$$

Στην κατάσταση / μεταβλητή Ω πρέπει να γυρίσουμε για να αντιγράψουμε τους υπόλοιπους χαρακτήρες:

$$
\begin{aligned}
0\Psi &\rightarrow \Psi 0 \\
1\Psi &\rightarrow \Psi 1 \\
,\Psi &\rightarrow \Psi , \\
,\Psi &\rightarrow \Phi ,
\end{aligned}
$$

Οταν τελειώσουμε με την αντιγραφή των επιθυμητών κορυφών $v_i, \ldots, v_k$, βρισκόμαστε στην κατάσταση Ψ. Πρέπει τώρα να ξεσημαδέψουμε όλους τους σημαδεμένους χαρακτήρες και επιπλέον να προχωρήσουμε προς την λίστα των ακμών ώστε να συνεχιστεί η διαδικασία δημιουργίας των απαραίτητων ακμών της κλίκας:

$$\begin{aligned}
\Psi 0' &\rightarrow 0\Psi \\
\Psi 1' &\rightarrow 1\Psi \\
\Psi,'' &\rightarrow ,\Psi \\
\Psi\mathbf{,} &\rightarrow \mathbf{,}\Psi \\
\Psi 0 &\rightarrow 0\Psi \\
\Psi 1 &\rightarrow 1\Psi \\
\Psi, &\rightarrow ,\Psi \\
\Psi\},\{ &\rightarrow \},\{\Omega
\end{aligned}$$

Τώρα στην κατάσταση Ω έχουμε αντιγράψει στην λίστα των ακμών τις κορυφές $v_i, \ldots, v_k$. Με σκοπό να κατασκευάσουμε τις απαραίτητες ακμές, είναι αναγκαίο να αντιγράψουμε την κορυφή v_i ανάμεσα από δύο διαδοχικές κορυφές της λίστας των κορυφών που μόλις αντιγράψαμε. Πρέπει να ελένξουμε εάν υπάρχουν άλλες ενδιάμεσες κορυφές:

$$\begin{aligned}
\Omega 0 &\rightarrow 0\Omega \\
\Omega 1 &\rightarrow 1\Omega \\
\Omega, &\rightarrow ,\Omega' \\
\Omega' 0 &\rightarrow 0\Omega' \\
\Omega' 1 &\rightarrow 1\Omega' \\
\Omega', &\rightarrow \alpha,', \\
\Omega'\{ &\rightarrow \omega\{ \\
\Omega'\} &\rightarrow \omega\}
\end{aligned}$$

Εχουμε σημαδέψει το κατάλληλο σημείο ώστε να αντιγράψουμε την κορυφή v_i αμέσως μετά τον σημαδεμένο χαρακτήρα διαχωρισμού ",'. Στην κατάσταση α πρέπει να γυρίσουμε πίσω και να αντιγράψουμε ένα ένα τα ψηφία της κορυφής v_i. Παρατηρούμε ότι απλά χρειάζεται να προχωρήσουμε κατά δύο κορυφές προς τα αριστερά.

Στο προηγούμενο σύνολο κανόνων ενδέχεται να μην υπάρχουν άλλες ενδιάμεσες κορυφές. Αυτό το διαπιστώνουμε εάν αμέσως μετά την πρώτη κορυφή δεξία της v_i υπάρχει είτε ο χαρακτήρας $\{$, που υποδηλώνει ότι υπάρχουν άλλες ακμές από προηγούμενα στάδια και άρα δεν υπάρχει χώρος να δημιουργηθεί άλλη ακμή από τον v_i, είτε ο χαρακτήρας $\}$ που δηλώνει ότι ο τρέχων v_i είναι στην πραγματικότητα ο v_1. Και στις δύο περιπτώσεις έχουμε τελεώσει με την αντιγραφή του v_i στις κατάλληλες θέσεις και το μόνο που μας μένει είναι η εισαγωγή των αγκύστρων ώστε να αναπαρασταθούν σωστά οι ακμές, λειτουργία η οποία θα γίνει με ̃σω της καταστάσεως / μεταβλητής ω.

Τώρα είμαστε σε θέση να αντιγράψουμε την κορυφή v_i στη σημαδεμένη μας θέση. Πρώτα σημεδεύουμε το τρέχων σύμβολο προς αντιγραφή:

$$\begin{aligned}
0\alpha &\rightarrow \alpha 0 \\
1\alpha &\rightarrow \alpha 1 \\
,\alpha &\rightarrow \beta, \\
0'\beta &\rightarrow \beta 0' \\
1'\beta &\rightarrow \beta 1' \\
0\beta &\rightarrow 0'\beta_0 \\
1\beta &\rightarrow 1'\beta_1 \\
,\beta &\rightarrow ,\gamma
\end{aligned}$$

και κατά τα γνωστά το αντιγράφουμε στην κατάλληλη θέση (η κατάσταση β_1 όπως πάντα είναι τελείως συμμετρική):

$$\begin{aligned}
\beta_0 0' &\rightarrow 0'\beta_0 \\
\beta_0 1' &\rightarrow 1'\beta_0 \\
\beta_0, &\rightarrow ,\beta_0 \\
\beta_0 0 &\rightarrow 0\beta_0 \\
\beta_0 1 &\rightarrow 1\beta_0 \\
\beta_0,' &\rightarrow \alpha,'0
\end{aligned}$$

Οταν διαπιστώσουμε ότι έχουμε τελειώσει με την αντιγραφή της κορυφής v_i στην κατάλληλη θέση, θα βρισκόμαστε στην κατάσταση γ. Κατ'αρχάς, είναι αναγκαίο να ξεσημεδέψουμε όλους τους σημαδεμένους χαρακτήρες της τρέχουσας κορυφής που αντιγράψαμε και κατόπιν να πεταβούμε στην κατάσταση

Ω σκοπός της οποίας είναι επαναληπτικά να βρίσκει εάν μπορούμε να αντιγράψουμε την κορυφή v_i σε ενδιάμεσες κορυφές της λίστας των κορυφών που αντιγράψαμε σε προηγούμενο βήμα:

$$\begin{aligned}
\gamma 0' &\rightarrow 0\gamma \\
\gamma 1' &\rightarrow 1\gamma \\
\gamma, &\rightarrow ,\gamma \\
\gamma 0 &\rightarrow 0\gamma \\
\gamma 1 &\rightarrow 1\gamma \\
\gamma,' &\rightarrow ,\Omega
\end{aligned}$$

Μόλις τελειώσει η διαδικασία αντιγραφής της κορυφής v_i στις κατάλληλες θέσεις, ενδιάμεσα των κορυφών με τις οποίες θα συνδεθεί με ακμή, είμαστε στην κατάσταση ω: απομένει μόνο να προσθέσουμε τα κατάλληλα άγκυστρα στις σωστές θέσεις, ώστε νε έχουμε ορθή αναπαράσταση των ακμών. Ουσιαστικά αγνοούμε τους δύο αριθμούς που βρίσκονται στα αριστερά της καταστάσεως / δρομέα και, μετά το τλεός αυτών, προσθέτουμε τα κατάλληλα άγκυστρα:

$$\begin{aligned}
0\omega &\rightarrow \omega 0 \\
1\omega &\rightarrow \omega 1 \\
,\omega &\rightarrow \omega', \\
0\omega' &\rightarrow \omega' 0 \\
1\omega' &\rightarrow \omega' 1 \\
,\omega' &\rightarrow \omega\},\{ \\
,\{\omega' &\rightarrow \Sigma,\{\{
\end{aligned}$$

7.5.2 Παραγωγή των υπόλοιπων ακμών του γραφήματος

Οταν έχουμε τελειώσει με την παραγωγή όλων των απαρίτητων ακμών που θα αποτελούν την κλίκα, θα βρισκόμαστε στην κατάσταση X. Τώρα θέλουμε να δημιουργήσουμε επιπλέον ακμές, για τις κορυφές εκτό τις κλίκας (δηλαδή για όσες κορυφές βρισκονται *δεξιά* του χαρακτήρα διαχωρισμού ,). Γενικά θα έχουμε δύο περιπτώσεις:

1. Το γράφημα G θα είναι από μόνο του μια κλίκα των k κορυφών, δηλαδή μετά τη δημιουργεία όλων των k κορυφών της κλίκας δεν έχουμε δημιουργήσει επιπλέον κορυφες. Με άλλα λόγια $|V| = k$ και $X\} \rightarrow X_1$: δεν χρειάζονται επιπλέον ακμές.

2. Εχουμε επιπλέον $|V|-k$ κορυφές εκτός της κλίκας. Για όλες τις κορυφές αυτές θέλουμε να δημιουργήσουμε τις επιθυμητές ακμές (οι οποίες βέβαια μπορούν, ως δεύτερο τερματικό σημείο, να έχουν κορυφές της κλίκας): $X, \rightarrow ,X_2$.

Είναι προφανές ότι εάν βρεθούμε στην κατάσταση X_1 τότε έχει τελειώσει η παραγωγή του στιγμιότυπού μας. Οπότε θα ασχοληθούμε με την περίπτωση όπου έχουμε επιπλέον κορυφές για τις οποίες θέλουμε να δημιουργήσουμε τις επιθυμητές ακμές.

Οπως κάναμε και με την περίπτωση των ακμών που αποτελούν την μείζονα κλίκα, είναι αναγκαίο να περιγραφεί σε φυσική γλώσσα η επαναλήπτική διαδικασία δημιουργείας των επιθυμητών ακμών πριν αυτή μεταγραστεί σε σύνολο κανόνων:

1. Αρχικά, βρίσκουμε και σημαδεύουμε στον δεξιό χαρακτήρα διαχωρισμού, είτε είναι "," είτε "}, την τρέχουσα κορυφή, έστω v_j αυτή, από την οποία και θα δημιουργήσουμε όλες τις επιθυμητές ακμές με που θα την ενώνουν με κορυφές *αριστερά* αυτής. Δηλαδή βρίσκουμε, εάν υπάρχει, την πρώτη προς τα δεξιά μη σημαδεμένη κορυφη.

2. Μόλις βρούμε και σημαδέψουμε την τρέχουσα v_j αρχίζουμε να σαρώνουμε την λίστα κορυφών προς τα *αριστερά*. Για κάθε κορυφή που συναντάμε, μη ντετερμινιστικά αποφασίζουμε εάν θέλουμε να αποτελεί μέλος της γειτονιάς N της v_j, δηλαδή εάν θέλουμε να την ενώσουμε με ακμή με την κορυφή v_j. Εάν ναι, τότε σημαδεύουμε τον δεξιό της χαρακτήρα διαχωρισμού με κάποιο χαρακτήρα διαφορετικό από αυτόν που χρησιμοποιήσαμε στο βήμα (1) και προχωράμε στην επόμενη κορυφή προς τα αριστερά. Εάν όχι, απλά συνεχίζουμε τη σάρωση προς τα αριστερά.

3. Αφού τελειώσουμε με το σημάδεμα όλων εκείνων των κορυφών $v_q \in N(v_j)$, δηλαδή των κορυφών εκείνων που επιθυμούμε να ανήκουν στην γειτονιά της κορυφής v_j, αντιγράφουμε όλες αυτές τις κορυφές στην λίστα ακμών του γραφήματος. Μόλις τελειώσει αυτή η αντιγραφή, αντιγράφουμε (στην αρχή αυτής της λίστας) την κορυφή v_j. Επειτα δημιουργούμε τις κατάλληλες ακμές, διπλασιάζοντας κατάλληλα την κορυφή v_j και προσθέτοντας τα κατάλληλα άγκυστρα, ακριβώς με τον ίδιο τρόπο που χρησιμοποιήσαμε και στην παραγωγή των ακμών τις κλίκας στη προγούμενη ενότητα.

4. Οταν τελειώσουμε με την κατασκευή των επιθυμητών ακμών για την κορυφή v_j, γυρνάμε πίσω (προς τα αριστερά), ξεσημαδεύουμε, φέρνουμε τον δρομέα στην κατάλληλη αρχική θέση (για επανεκκίνηση της διαδικασίας) και πηγαίνουμε στο βήμα (1).

Βρισκόμαστε στην κατάσταση X_2. Αγνοούμε οτιδήποτε υπάρχει προς τα αριστερά της καταστάσεως αυτής μέχρι να συναντήσουμε τον πρώτο χαρακτήρα διαχωρισμού, είτε αυτός είναι "," είτε "}:

$$\begin{aligned} X_2 0 &\rightarrow 0X_2 \\ X_2 1 &\rightarrow 1X_2 \\ X_2, &\rightarrow \chi,' \\ X_2\} &\rightarrow \chi\} \end{aligned}$$

Στην κατάστασταση χ έχουμε βρεί την τρέχουσα κορυφή v_j, την έχουμε σημαδέψει, και μπορούμε πλέον να σαρώσουμε προς τα αριστερά την λίστα των κορυφών και να αποφασίσουμε με ποιές από αυτές θα ενώσουμε με ακμή την v_j:

$$\begin{aligned} 0\chi &\rightarrow \chi 0 \\ 1\chi &\rightarrow \chi 1 \\ ,\chi &\rightarrow \chi, \mid \chi,'' \quad \triangleright \text{μη ντετερμινιστικά σημαδεύω την τρέχουσα κορυφή} \\ \text{,}\chi &\rightarrow \chi\text{,} \mid \chi\text{,}'' \\ \{\chi &\rightarrow \{\zeta \end{aligned}$$

Μόλις τελειώσει η παραπάνω σάρωση των κορυφών του γραφήματος οι οποίες βρίσκονται στα αριστερά της τρέχουσας v_j, θα έχουμε σημαδέψει όλες τις επιθυμητές κορυφές, τις οποίες θέλουμε να ενώσουμε με ακμή με την v_j, στο δεξί διαχωριστικό τους σημείο και θα είμαστε στην κατάσταση ζ στην αρχή της λίστας ονομάτων των κορυφών του γραφήματος.

Τώρα, είναι αναγκαίο να αντιγράψουμε όλες αυτές τις σημαδεμένες κορυφές (πλη την v_j όπως είπαμε-αυτή θα την αντιγράψουμε τελευταία) στην λίστα των ακμών, ώστε σε επόμενο βήμα να κατασκευάσουμε κατάλληλα τις ακμές.

Από την κατάσταση ζ σαρώνουμε προς τα δεξιά μέχρι να βρούμε τον πρώτο σημαδεμένο χαρακτήρα διαχωρισμού ",'" ή ",'" οπότε και ξέρουμε ότι πρέπει να αντιγράψουμε την κορυφή που βρίσκεται δεξιά του χαρακτήρα αυτού στην

λίστα των ακμών (αφού αποφασίσαμε σε προηγούμενο βήμα να την ενώσουμε με την v_j):

$$
\begin{array}{rcll}
\zeta 0 & \rightarrow & 0\zeta & \\
\zeta 1 & \rightarrow & 1\zeta & \\
\zeta, & \rightarrow & ,\zeta & \\
\zeta,'' & \rightarrow & \theta, & \triangleright \text{ξεσημαδεύω} \\
\zeta\mathbf{,} & \rightarrow & \mathbf{,}\zeta & \\
\zeta\mathbf{,}'' & \rightarrow & \theta\mathbf{,} & \triangleright \text{ξεσημαδεύω} \\
\zeta,' & \rightarrow & \theta,' & \triangleright \text{βρήκαμε και αντιγράφουμε την } v_j \text{ χωρίς να ξεσημαδέψουμε} \\
\zeta\},\{ & \rightarrow & \},\{\Omega & \triangleright \text{κατασκευή των ακμών}
\end{array}
$$

Στην κατάσταση θ πρέπει να αντιγράψουμε την κορυφή δεξιά της θ στην λίστα των ακμών του γραφήματος. Κατά τα γνωστά, πρώτα βρήσκουμε τον πρώτο (προς τα αριστερά) μη σημαδεμένο χαρακτήρα-ψηφίο, τον σημαδεύουμε και παράλληλα θυμόμαστε στην κατάλληλη κατάσταση ποιόν χαρακτήρα μόλις διαβάσαμε ώστε να τον αντιγράψουμε κατάλληλα. Είναι αναγκαίο επίσης να μπορούμε με κάποιον τρόπο να αντιγράψουμε και τον χαρακτήρα διαχωρισμού μαζί με τον αριθμό στην λίστα των ακμών. Αυτό γίνεται εύκολα με το να βλέπουμε εάν είμαστε στον πρώτο *προς αντιγραφή* χαρακτήρα, οπότε και μεταβαίνουμε σε κατάλληλη κατάσταση η οποία και θα προσθέσει τον διαχωριστικό χαρακτήρα στο δεξί τέλος του αριοθμού που αντιγράφουμε στο τρέχον βήμα:

$$
\begin{array}{rcl}
0'\theta & \rightarrow & \theta 0' \\
1'\theta & \rightarrow & \theta 1' \\
0\theta 0' & \rightarrow & 0'\theta_0 0' \\
0\theta 1' & \rightarrow & 0'\theta_0 1' \\
0\theta, & \rightarrow & 0'\theta_{0,}, \\
1\theta 0' & \rightarrow & 1'\theta_1 0' \\
1\theta 1' & \rightarrow & 1'\theta_1 1' \\
1\theta, & \rightarrow & 1'\theta_{1,}, \\
,\theta & \rightarrow & ,\zeta' \\
\{\theta & \rightarrow & \{\zeta'
\end{array}
$$

Μόλις βρούμε τον τρέχον χαρακτήρα προς αντιγραφή, τον αποθηκεύουμε στην κατάλληλη κατάσταση και τον αντιγράφουμε στη σωστή θέση (χωρίς βλάβη της γενικότητας θεωρούμε μόνο τον κανόνα θ_0, οι υπόλοιποι είναι τελείως ανάλογοι):

$$\begin{aligned}
\theta_0 0 &\rightarrow 0\theta_0 \\
\theta_0 1 &\rightarrow 1\theta_0 \\
\theta_0, &\rightarrow ,\theta_0 \\
\theta_0,' &\rightarrow ,'\theta_0 \\
\theta_0,'' &\rightarrow ,''\theta_0 \\
\theta_0\mathbf{,} &\rightarrow \mathbf{,}\theta_0 \\
\theta_0\mathbf{,}'' &\rightarrow \mathbf{,}''\theta_0 \\
\theta_0\},\{ &\rightarrow \eta\},\{0
\end{aligned}$$

Μια παρατήρηση: οι κανόνες $\theta_{0,}$ και $\theta_{1,}$ χρειάζονται ώστε να αντιγράψουμε κατάλληλα τον χαρακτήρα διαχωρισμού στη σωστή θέση στην λίστα των ακμών. Δηλαδή οι κανόνες των δύο αυτών μεταβλητών είναι αντίστοιχοι με τοις προηγούμενους εκτός από τον τελευταίο κανόνα ο οποίος τώρα θα έχει την μορφή $\theta_{0,}\},\{\rightarrow \eta\},\{0$, στην πρώτη περίπτωση και $\theta_{1,}\},\{\rightarrow \eta\},\{1$, στην δεύτερη.

Στην κατάσταση η πρέπει να γυρίσουμε πίσω ώστε να συνεχίσουμε την αντιγραφή της τρέχουσας κορυφής:

$$\begin{aligned}
0\eta &\rightarrow \eta 0 \\
1\eta &\rightarrow \eta 1 \\
,\eta &\rightarrow \eta, \\
,'\eta &\rightarrow \eta,' \\
0'\eta &\rightarrow \theta 0' \\
1'\eta &\rightarrow \theta 1'
\end{aligned}$$

Στην κατάσταση ζ' μεταβαίνουμε όταν διαπιστώσουμε ότι έχουμε τελειώσει με την αντιγραφή της τρέχουσας κορυφής οπότε προχωράμε προς τα δεξιά ώστε να βρούμε την νέα κορυφή προς αντιγραφή (εάν αυτή υπάρχει) και παράλληλα ξεσημαδεύουμε τους ήδη σημαδεμένους χαρακτήρες:

$$
\begin{array}{rcl}
\zeta'0' & \rightarrow & 0\zeta' \\
\zeta'1' & \rightarrow & 1\zeta' \\
\zeta', & \rightarrow & ,\zeta
\end{array}
$$

Κεφαλαιο 8

Γραμματικη Με Συμφραζομενα για το Προβλημα του Τριμερους Ταιριασματος

8.1 Εισαγωγή

Στο παρόν κεφάλαιο θα παρουσιάσουμε σύνολα κανόνων τέτοια ώστε να μας επιτρέπουν να κατασκευάσουμε τρία ξένα μεταξύ τους σύνολα μεγέθους n το καθένα και ένα πλήθος (διατεταγμένων) τριάδων από τα τρία αυτά σύνολα τέτοιο ώστε να υπάρχουν n τριάδες στο σύνολο των τριάδων μας τέτοιες ώστε όποιες δύο από τις τριάδες αυτές και να διαλέξουμε, αυτές δεν έχουν καμία συνιστώσα κοινή. Δηλαδή θέλουμε μια γραμματική η οποία να μας δίνει τη δυνατότητα να δημιουργήσουμε ένα οποιοδήποτε "ναι" στιγμιότυπο για το πρόβλημα του πλήρους (τριμερούς) ταιριάσματος το οποίο τυπικά ορίζεται ως εξής:

Τριμερές Ταίριασμα: Μας δίνονται τρία σύνολα, έστω X, Y και Z τέτοια ώστε $|X| = |Y| = |Z| = n$ και μια τριμερή σχέση $T \subseteq X \times Y \times Z$. Μας ζητείται να βρούμε n τριάδες της σχέσης T, για τις οποίες ισχύει ότι οποιεσδήποτε δύο από αυτές τις n τριάδες δεν έχουν καμία συνιστώσα κοινή. Ονομάζουμε το προηγούμενο πρόβλημα *Τριμερές Ταίριασμα*.

Το αντίστοιχο πρόβλημα απόφασης διατυπώνεται με τον προφανή τρόπο: δοθέντων τριών συνόλων όπως στον παραπάνω ορισμό, αληθεύει ότι μπορούμε να βρούμε ένα πλήρες τριμερές ταίριασμα;

Οπως πάντα, είναι αναγκαίο να περιγράψουμε έναν κατάλληλο τρόπο αναπαράστασης ενός έγκυρου στιγμιοτύπου για το πρόβλημα του τριμερούς ταιριάσματος. Πρώτα απ'όλα το στιγμιότυπο θα περιέχει τα στοιχεία των τριών συνόλων-τριάδων X, Y και Z. Τα στοιχεία των συνόλων αυτών θα αναπαρίστανται ως εξής: για το σύνολο X ως $x0, x1, \ldots$ όπου n είναι το πλήθος των στοιχείων του κάθε συνόλου και επιπλέον ο κάθε αριθμό γραμμένος στο δυαδικό σύστημα. Αντίστοιχα για τα υπόλοιπα σύνολα-τριάδες. Δεξιά των στοιχείων αυτών θα τοποθετηθούν οι τριάδες. Ο τρόπος αναπαράστασης μιας τριάδας είναι ο προφανής: $(x_i, y_j, z_k),\ 0 \leq i, j, k < n$.

8.2 Παραγωγή των τριών συνόλων X, Y και Z

Στην ενότητα αυτή θα περιγράψουμε ένα σύνολο κανόνων το οποίο θα δημιουργεί έναν οποιοδήποτε ακέραιο n μεγαλύτερο του μηδενός σε δυαδική αναπαράσταση. Ο ακέραιος αυτός έχει διπλό ρόλο. Αφ'ενός θα υποδηλώνει το πλήθος των στοιχείων των τριών συνόλων X, Y και Z και αφ'ετέρου θα μας βοηθήσει στην δημιουργεία των στοιχείων που αποτελούν τα σύνολα αυτά. Οπως είπαμε, τα στοιχεία των συνόλων αναπαρίστανται από ένα γράμμα το οποίο υποδηλώνει το σύνολο το οποίο ανήκει το στοιχείο, ακολουθούμενο από έναν αριθμό μικρότερο ή ίσο του n (πχ $x0101, z1001$ κ.ο.κ):

$$
\begin{aligned}
S &\rightarrow \{\{1A\}\} \\
A &\rightarrow 1A \mid 0A \mid B
\end{aligned}
$$

Ο κανόνας B σηματοδοτεί το τέλος της παραγωγής του αριθμού n. Δηλαδή, μετά το τέλος αυτής της παραγωγής, θα έχουμε δημιουργήσει μια συμβολοσειρά της μορφής $\{\{101B\}\}$ για την περίπτωση όπου $n = 5$. Βάζουμε διπλές αγκύλες καθώς το πρώτο ζεύγος αυτών περικλείει τα στοιχεία ενός συνόλου ενώ το δεύτερο το όλο στιγμιότυπο.

Τώρα μπορούμε να δημιουργήσουμε τα στοιχεία των συνόλων X, Y και Z. Θα αρχίσουμε δημιουργώντας όλα τα $x_i, 0 \leq i < n$ (ακριβώς n στοιχεία). Για να γίνει αυτό, κάνουμε χρήση μιας καταστάσεως/ρουτίνας, έστω C η οποία:

α. Ελέγχει εάν ο δυαδικός αριθμός δεξιά της καταστάσεως (που αντιπροσωπεύει τον δρομέα) είναι μηδέν ή όχι.

β. Εάν όχι, τότε μεταβαίνουμε σε μια νέα κατάσταση, έστω D, η οποία:

β_1. Θα αντιγράφει τον αριθμό που μόλις ελέγξαμε (και δεν είναι μηδέν) στα *αριστερά* αυτού,

β_2. θα τον μειώνει κατά μία μονάδα,

β_3. θα επιστρέφει στην προηγούμενη κατάσταση C για να επαναλάβει την ίδια διαδικασία μέχρις ότου βρει ότι έχει φτάσει στο μηδέν, όποτε δεν είναι δυνατή περαιρέρω μείωση.

γ. Εάν ναι, τότε κάνε πατάβαση στην κατάσταση E.

Αρχικά, είναι αναγκαίο να μειώσουμε κατά μία μονάδα τον ακέραιο n (αφού, όπως είπαμε, θέλουμε να δημιουργήσουμε όλα τα στοιχεία $x_i, 0 \leq i < n$):

$$\begin{aligned} 0B &\rightarrow B1 \\ 1B &\rightarrow 0B' \end{aligned}$$

Επειτα, από την κατάσταση B' θέλουμε να μεταφερθούμε στην αρχή της συμβολοσειράς:

$$\begin{aligned} 0B' &\rightarrow B'0 \\ 1B' &\rightarrow B'1 \\ \{B' &\rightarrow \{C \end{aligned}$$

Στην κατάσταση C ελέγχουμε (όπως είπαμε) εάν ο αριθμός δεξιάς αυτής είναι ή όχι μηδέν:

$$
\begin{aligned}
C0 &\rightarrow 0C \\
C1 &\rightarrow D1 \quad (\textit{ο αριθμός δεν είναι μηδέν}) \\
C\} &\rightarrow E\} \\
C, &\rightarrow E,
\end{aligned}
$$

Στις δύο τελευταίες περιπτώσεις, ο αριθμός βρέθηκε μηδέν, και μεταβαίνουμε στην κατάσταση E, την λειτουργία της οποίας θα περιγράψουμε σε λίγο. Οπως είπαμε, στην κατάσταση D μεταβαίνουμε όταν δούμε ότι ο αριθμός δεν είναι μηδέν. Τώρα, πρέπει να φέρουμε τον δρομέα στην κατάλληλη θέση ώστε να τον αντιγράψουμε και έπειτα να μειώσουμε το αντίγραφο κατά μία μονάδα:

$$
\begin{aligned}
0D &\rightarrow D0 \\
\{D &\rightarrow \{, Z
\end{aligned}
$$

Η λειτουργία της Z είναι η εξής: αντέγραψε τον αριθμό που βρίσκεται στα δεξιά της Z, στα αριστερά αυτής, προσθέτοντας τον κατάλληλο χαρακτήρα διαχωρισμού (“,”):

$$
\begin{aligned}
Z0 &\rightarrow Z_0 0' \\
Z1 &\rightarrow Z_1 1' \\
Z0' &\rightarrow 0'Z \\
Z1' &\rightarrow 1'Z \\
Z, &\rightarrow H, \\
Z\} &\rightarrow H\}
\end{aligned}
$$

Αφού σημαδέψαμε τον τρέχον χαρακτήρα αντιγραφής, τον αντιγράφουμε στο κατάλληλο σημείο (η περίπτωση της μεταβλητής Z_1 είναι τελείως ανάλογη):

$$
\begin{aligned}
0'Z0 &\rightarrow Z_0 0' \\
1'Z_0 &\rightarrow Z_0 1' \\
,Z_0 &\rightarrow 0, Z
\end{aligned}
$$

Οταν τελειώσει η διαντιγραφής του τρέχοντος αριθμού, το διαπιστώνουμε φτάνοντας στο δεξί άκρο του αριθμού αυτού το οποίο αποτελείται είτε από τον χαρακλτήρα "," είτε από τον "}", μεταβαίνουμε στην κατάσταση H. Ο ρόλος της H είναι να μεταβεί στο νέο αντίγραφο του αριθμού που μόλις δημιουργήσαμε και να το μειώσει κατά μία μονάδα (ξέρουμε ότι δεν είναι μηδέν). Στην περίπτωση αυτή, μετακινούμενοι προς τα αριστερά, ξεσημαδεύουμε όλους τους σημαδεμένους χαρακτήρες του τρέχοντος αριθμού που μόλις αντιγράψαμε, φέρνοντας ταυτόχρονα τον δρομέα στην κατάλληλη θέση ώστε να αρχίσει η μείωση του αντιγράφου:

$$\begin{aligned} 0'H &\rightarrow H0 \\ 1'H &\rightarrow H1 \\ ,H &\rightarrow \Theta, \end{aligned}$$

Η κατάσταση Θ αναλαμβάνει να κάνει την μείωση κατά μία μονάδα, με τον γνωστό τρόπο, του αριθμού που βρίσκεται στα αριστερά αυτής:

$$\begin{aligned} 0\Theta &\rightarrow \Theta 1 \\ 1\Theta &\rightarrow I0 \end{aligned}$$

Μολις τελειώσει η διαδικασία της μείωσης, εισερχόμαστε στην κατάσταση I σκοπός της οποίας είναι να μας μεταφέρει στο αριστερό άκρο του τρέχονταος αριθμού ούτως ώστε να επαναληθφεί η διαδικασία αντιγραφής και μείωσης κατά μια μονάδα, εφ'όσον αυτό είναι εφικτό, ώσπου να φτάσουμε στο δυαδικό μηδέν.

$$\begin{aligned} 0I &\rightarrow I0 \\ 1I &\rightarrow I1 \\ \{I &\rightarrow \{C \end{aligned}$$

Οταν τελειώσει η παραπάνω διαδικασία, βρισκόμενοι στην κατάσταση E, έχουμε όλους τους αριθμούς $i : 0 \leq i < n$, και μας απομένει να ονοματίσουμε κατάλληλα τα στοιχεία. Πριν όμως γίνει αυτό, είναι αναγκαίο

να αντιγράψουμε δύο φορές την συμβολοσειρά των αριθμών αυτών στα δεξιά αυτής. Το στιγμιότυπο την δεδομένη χρονική στιγμή θα έχει την μορφή $\{\{000E, 001, 010, 011, 100\}\}$ για την περίπτωση όπου αρχικά το n ισούται με 5 (101 στο δυαδικό και άρα δημιουργούμε 5 αριθμούς, από το 0 έως το δυαδικό 4).

Αρχικά, φέρνουμε τον δρομέα στην κατάλληλη θέση ώστε να ξεκινήσει η αντιγραφή του συνόλου:

$$\begin{aligned} 0E &\rightarrow E0 \\ 1E &\rightarrow E1 \\ \{E &\rightarrow K\{ \end{aligned}$$

Στην κατάσταση K αρχίζουμε την αντιγραφή του σύνόλου με τον γνωστό τρόπο όπου μόλις ο δρομέας σηναντίσει το τρέχον σύμβολο αντοιγραφής, έστω x, το σημαδεύει (x') και θυμάται το σύμβολο που πρέπει να αντιγράψει στην κατάστασή του (K_x):

$$\begin{aligned} K\{ &\rightarrow \{'K_{\{} \\ K0 &\rightarrow 0'K_0 \\ K1 &\rightarrow 1'K_1 \\ K, &\rightarrow ,'K_{,} \\ K0' &\rightarrow 0'K \\ K1' &\rightarrow 1'K \\ K,' &\rightarrow ,'K \\ K\} &\rightarrow \}'K_{\}} \end{aligned}$$

Μόλις βρούμε τον τρέχων χαρακτήρα αντιγραφής, πηγαίνουμε να τον αντιγράψουμε στο σωστό μέρος. Η διαδικασία είναι τελείως συμμετρική γιά όλα τα σύμβολα προς αντιγραφή, οπότε υποθέτουμε ότι, χωρίς βλάβη της γενικότητας, αντιγράφουμε τον χαρακτήρα x όπου $x \in \{0, 1, , , \{, \}\}$:

$$
\begin{aligned}
K_x 0 &\rightarrow 0 K_x \\
K_x 1 &\rightarrow 1 K_x \\
K_x , &\rightarrow , K_x \\
K_x \}, \{ &\rightarrow \}, \{ K'_x \\
K'_x 0 &\rightarrow 0 K'_x \\
K'_x 1 &\rightarrow 1 K'_x \\
K'_x , &\rightarrow , K'_x \\
K'_x \} &\rightarrow L x \}
\end{aligned}
$$

Η κατάσταση K λειτουργεί ως εξής: μόλις βρει τον τρέχων χαρακτήρα αντιγραφής, τον σημαδεύει και θυμάται στην αντίστοιχη κατάσταση τον χαρακτήρα που πρέπει να αντιγράψει. Επειτα, προσπερνάει το υπόλοιπο κομμάτι της συμβολοσειράς προς αντιγραφή μέχρι να συναντήσει τους χαρακτήρες που διαχωρίζουν τα δύο σύνολα: το ένα προς αντιγραφή και το άλλο που δημιουργείται αντιγράφοντας τους χαρακτήρες του πρώτου (κανόνας $K_x\}, \{ \quad \rightarrow \quad \}, \{K'_x$). Τότε πηγαίνουμε στην κατάσταση η οποία θα προσπεράει οτιδήποτε βρεί στα δεξιά της, μέχρι να βρεί τον τερματικό χαρακτήρα }, οπότε και αριστερά αυτού τοποθετεί το τρέχον σύμβολο x και μεταβαίνει στην κατάσταση L.

Οπως είπαμε, για κάθε σύμβολο $x \in \{0, 1, , , \{, \}\}$ δημιουργούμε την προηγούμενη οκτάδα χαρακτήρων, με μία μικρή εξαίρεση: εάν το τρέχων σύμβολο αντιγραφής είναι το { τότε αντικαθιστούμε τον κανόνα $K_x\}, \{ \quad \rightarrow \quad \}, \{K'_x$ με τον εξής $K_\{\} \quad \rightarrow \quad \}, \{L$. Ο λόγος είναι προφανής: είναι αναγκαίο να εισαχθεί το διαχωριστικό σύμβολο "," των δύο συνόλων, και αυό το κάνουμε όταν πάμε να αντιγράψουμε το πρώτο σύμβολο του πρώτου συνόλου στην κατάλληλη θέση.

Τωρα, από την κατάσταση L είναι αναγκαίο να γυρίσουμε πίσω μέχρις ότου βρούμε το πρώτο προς τα αριστερά σημαδεμένο σύμβολο: το δεξιό του είναι το νέο τρέχων σύμβολο αντιγραφής:

$$
\begin{aligned}
0L &\rightarrow L0 \\
1L &\rightarrow L1 \\
,L &\rightarrow L, \\
\},\{L &\rightarrow L\},\{ \\
\}',\{L &\rightarrow \}',M\{ \\
0'L &\rightarrow 0'K \\
1'L &\rightarrow 1'K \\
\{'L &\rightarrow \{'K
\end{aligned}
$$

Ο κανόνας $\}',\{L \quad \rightarrow \quad \}',M\{$ εξηγείται ως εξής: γενικά ο δρομέας, μέσω του κανόνα L, ψάχνει να βρει το πρώτο προς τα αριστερά σημαδεμένο χαρακτήρα ούτως ώστε να μεταβεί αμέσως στην κατάσταση K, δουλειά της οποίας είναι να βρεί τον επόμενο μη σημαδεμένο χαρακτήρα. Στην προηγούμενη περίπτωση όμως, ο τελευταίος σημαδεμένος χαρακτήρας είναι ο } δηλαδή ο τελευταίος της συμβολοσειράς που αντιγράφουμε στο τρέχον στάδιο. Με άλλα λόγια η αντιγραφή του τρέχοντος συνόλου έχει τελειώσει και μεταβαίνουμε στην κατάσταση M η οποία θα αρχίσει την παραγωγή του δέυτέρου αντιγράφου του συνόλου.

Τώρα είμαστε έτοιμοι για την παραγωγή του δευτέρου αντιγράφου του συνόλου. Οταν βρεθούμε στην κατάσταση M, ξέρουμε ότι έχουμε δημιουργήσει το πρώτο αντίγραφο και είομαστε έτοιμοι να δημιουργήσουμε το δεύτερο. Θα μπορούσαμε να φτιάξουμε ένα νέο σύνολο κανόνων, *σχεδόν* όμοιο με το προηγούμενο, το οποίο και θα εκτελεί την διαδικασία ατιγραφής. Ομως μπορούμε να τα καταφέρουμε καλύτερα, χωρίς να χρειαστεί να γράψουμε ένα παρόμοιο σύνολο, και σχετικά μεγάλο, κανόνων. Αυτό μπορεί να γίνει με την εξής λογική: όταν είμαστε στην κατάσταση L και συναντίσουμε την συμβολοσειρά $\}',\{$, ξέρουμε ότι έχει τελιώσει η παραγωγή του πρώτου αντιγράφου. Στην κατάσταση αυτή, μπορούμε απλά να μεταβούμε ξανά στην K και να αρχίσουμε την προηγούμενη διαδικασία αντιγραφής. Το πρόβλημα όμως τώρα είναι ότι μετά την τρέχουσα, δεύτερη, αντιγραφή, έχουμε τελειώσει και πρέπει να μεταβούμε σε μια νέα κατάσταση η οποία θα κάνει τα επόμενα βήματα. Πως θα ξέρει όμως η γραμματική ότι έχουμε πράγματι τελειώσει; Αν μεταβούμε κατευθείαν στην K, τότε επ'άπειρον θα συνεχίσουμε να δημιουργούμε αντίγραφα τους συνόλου των αριθμών προς τα δεξιά της συμβολοσειράς. Αρα με κάποιο τρόπο πρέπει να "σημαδέψουμε" την τρέχουσα συμβολοσειρά που αντιγράφουμε, ώστε όταν αντιληφθούμε ότι τελείωσε η αντιγραφή της, να μπορούμε

με σιγουριά να πούμε ότι έχει τελειώσει η *δεύτερη* αντιγραφή. Αυτό γίνεται προσθέτοντας στην γραμματική τους παρακάτω κανόνες:

$$\begin{aligned}
M\{ &\rightarrow K\bar{\{} \\
K\bar{\{} &\rightarrow \bar{\{}'K_{\bar{\{}} \\
\bar{\{}'L &\rightarrow \bar{\{}'K \\
\},\bar{\{}L &\rightarrow L\},\bar{\{} \\
\}',\bar{\{}L &\rightarrow \}',N\bar{\{} \\
M\bar{\{} &\rightarrow N\{
\end{aligned}$$

Η λογική είναι η εξής: απλά σημαδεύουμε τον πρώτο χαρακτήρα προς αντιγραφή ($\{$) και τον αντιγράφουμε *σημαδεμένο*. Ετσι, όταν καταλάβουμε ότι τελείωσε η αντιγραφή, μέσω του κανόνα $\}',\bar{\{}L \quad \rightarrow \quad \}',M\bar{\{}$, τότε μεταβαίνουμε στην M η οποία, βλέποντας τώρα στα δεξιά της τον σημαδεμένο πρώτο χαρακτήρα του τρίτου συνόλου το οποίο μόλις δημιουργήθηκε, ξέρει ότι έχουμε τελειώσει με την παραγωγή συνόλων και τότε μεταβαίνει σε μία νέα κατάσταση N η οποία, αφού ξεσημαδέψει όλα τα σημαδεμένα σύμβολα, θα βάλει τα αρχικά κατάλληλα γράμματα σε κάθε αριθμό των συνόλων, ώστε να δημιουργηθούν, στην μορφή που περιγράψαμε στην αρχή, τα τρία σύνολα X, Y και Z.

Επιπλέον, η κατάσταση $K_{\bar{\{}}$ λειτουργεί ακριβώς όπως και η $K_{\{}$ με τη μόνη διαφορά στον κανόνα $K_{\bar{\{}}\} \quad \rightarrow \quad \},\bar{\{}L$.

Τώρα είναι αναγκαίο να ξεσημαδέψουμε όλα τα σημαδεμένα σύμβολα, πριν προχωρήσουμε στην εισαγωγή των κατάλληλων γραμμάτων στην αρχή του κάθε αριθμού:

$$\begin{aligned}
,N &\rightarrow N, \\
\}'N &\rightarrow N\} \\
\{'N &\rightarrow N\{ \\
,'N &\rightarrow N, \\
0'N &\rightarrow N0 \\
1'N &\rightarrow N1 \\
\{N &\rightarrow \{P
\end{aligned}$$

Μετά το τέλος της παραπάνω διαδικασίας, το στιγμιότυπό μας θα είναι στη μορφή (για $n = 3$): $\{P\{000, 001, 010\}, \{000, 001, 010\}, \{000, 001, 010\}\}$.

Από την κατάσταση P, είναι αναγκαίο, προχωρώντας προς τα δεξιά, για κάθε αριθμό που συναντάμε, να προσθέτουμε στην αρχή του το κατάλληλο γράμμα x, y ή z. Αρχικά, προσθέτουμε το γράμμα x στις κατάλληλες θέσεις:

$$\begin{aligned}
P &\rightarrow P_x \\
P_x\{ &\rightarrow \{xP_x \\
P_x0 &\rightarrow 0P_x \\
P_x1 &\rightarrow 1P_x \\
P_x, &\rightarrow ,xP_x \\
P_x\}, &\rightarrow \}, P_y \\
P_y\}, &\rightarrow \}, P_z \\
P_z\} &\rightarrow \}R
\end{aligned}$$

Οι καταστάσεις P_y και P_z κατά τα άλλα, δουλεύουν ακριβώς με τον ίδιο τρόπο που δουλεύει και η κατάσταση P_x αι μόλις τελειώσει η διαδικασία, μεταβαίνουμε στη νέα κατάσταση R. Στο παράδειγμά μας, το στιγμιότυπο την τρέχουσα στιγμή θα είναι της μορφής: $\{\{x000, x001, x010\}, \{y000, y001, y010\}, \{z000, z001, z010\}R\}$.

8.3 Παραγωγή των τριάδων ενός έγκυρου τριμερούς ταιριάσματος

Στην ενότητα αυτή, θα δημιουργήσουμε όλες τις τριάδες που αποτελούν ένα έγκυρο τριμερές ταίριασμα για οποιοδήποτε στιγμιότυπο του προβλήματός μας. Πριν προχωρήσουμε στις τεχνικές λεπτομέρειες και τους κανόνες που θα μας οδηγήσουν στο επιθυμητό αποτέλεσμα, είναι αναγκαίο να περιγραφεί σε φυσική γλώσσα ο τρόπος, ο αλγόριθμος, παραγωγής αυτών των τριάδων, ώστε αργότερα να έιναι πιο εύκολα αναγνώσιμοι οι αντίστοιχοι κανόνες.

Ο αλγόριθμός μας σε κάθε βήμα, έστω στο i-οστό, θα αποφασίζει πια στοιχεία εκείνα x_{i_j}, y_{i_k} και z_{i_l} θα προσθέσει στην τρέχουσα, i-οστή, τριάδα που δημιουργεί. Αυτό γίνεται διαλέγοντας κατάλληλα τα στοιχεία με τον εξής τρόπο (χωρία βλάβη της γενικότητας θα δείξουμε πώς διαλέγουμε κάποιο κατάλληλο στοιχείο από το σύνολο X. Η μέθοδος για τα υπόλοιπα δύο σύνολα είναι τελειως ανάλογη):

- ▷ Φέρνουμε τον δρομέα αμέσως αριστερά του συνόλου X.

- ▷ Τα στοιχεία του συνόλου X διαχωρίζονται, με κατάλληλο σημάδεμα των ψηφίων του, σε έγκυρα, τα οποία μπορούμενα διαλέξουμε στην τρέχουσα φάση, και άκυρα, τα οποία έχουν ήδη χρησιμοποιηθεί σε προηγούμενη φάση για τη δημιουργεία κάποιας άλλης τριάδος.

- ▷ Μόλις βρεθούμε προ ενός έγκυρου αριθμού, στοιχείου του συνόλου X, αποφασίζουμε, μη ντετερμινιστικά, εάν το τρέχον στοιχείο θα το τοποθετήσουμε στην τρέχουσα τριάδα.

  - ○ Εάν ναι, τότε σημαδεύουμε τον αριθμό αυτό (προσοχή: το σημάδεμα θα γίνει με διαφορετικό τρόπο από αυτόν που είναι σημαδεμένα όλα τα άκυρα στοιχεία ώστε να διακρίνουμε ποιο στοιχείο αντιγράφουμε! Μόλις τελειώσει η αντιγραφή, θα το σημαδέψουμε με τον κανονικό τρόπο) και τον αντιγράφουμε στην κατάλληλη θέση.
  - ○ Εάν όχι, τότε συνεχίζουμε με τον ίδιο τρόπο προς τα δεξιά. Εάν όλοι οι αριθμοί είναι σημαδεμένοι έχουμε τελειώσει με την παραγωγή των τριάδων του τριμερούς ταιριάσματος.

- ▷ Εάν φτάσουμε στον τελευταίο αριθμό, απορρίπτοντας όλους τους ενδιάμεσους έγκυρους, και πάλι αποφασίσουμε όχι στον τελευταίο αριθμό, τότε οπισθοδρομούμε και προσθέτουμε στην τρέχουσα τριάδα το πρώτο προς τα αριστερά έγκυρο (μη σημαδεμένο) στοιχείο.

- ▷ Προχωράμε, με τον ίδιο τρόπο, για να επιλέξουμε στοιχεία των άλλων δύο συνόλων.

Για να αρχίσει η παραπάνω διαδικασία, είναι αναγκαίο αρχικά να φέρουμε τον δρομέα στην αρχή του στιγμιοτύπου:

$$
\begin{aligned}
0R &\rightarrow R0 \\
1R &\rightarrow R1 \\
\}R &\rightarrow R\} \\
,R &\rightarrow R, \\
xR &\rightarrow Rx \\
yR &\rightarrow Ry \\
zR &\rightarrow Ry \\
\},\{R &\rightarrow R\},\{ \\
\{\{R &\rightarrow \{\{S_x
\end{aligned}
$$

Στο παράδειγμά μας, ο τρέχων στιγμιότυπο θα είναι $\{\{S_x x000, x001, x010\}$, $\{y000, y001, y010\}$, $\{z000, z001, z010\}\}$. Στην κατάσταση S_x δηλαδή, πρέπει να αρχίσουμε την επιλογή των στοιχείων εκείνων από τα τρία σύνολα X, Y και Z τα οποία θα αποτελούν τις τριάδες του ταιριάσματος του στιγμιοτύπου του τριμερούς ταιράσματος, ξεκινώντας από το πρώτο σύνολο X. Οταν τελειώσουμε τη διαδικασία αυτή, επιλέξουμε δηλαδή το τρέχων στοιχείο του συνόλου X το οποίο θα είναι μέρος της τρέχουσας τριάδος του ταιριάσματος, συνεχίζουμε ώστε να επιλέξουμε στοιχεία για τις απόμενες δύο συνιστώσες της τρέχουσας τριάδος από τα δύο υπόλοιπα σύνολα Y και Z (μέσω των αντίστοιχων καταστάσεων S_y και S_z αντίστοιχα).

Θα περιγράψουμε πώς γίνεται η επιλογή και η αντιγραφή των στοιχείων του συνόλου X στην κατάλληλη θέση. Η διαδικασία είναι τελείως ανάλογη και για τα στοιχεία των δύο επόμενων συνόλων Y και Z.

Στην κατάσταση S_x που βρισκόμαστε, είναι αναγκαίο να ελένξουμε εάν ο αριθμός που βρίσκεται στα δεξιά της καταστάσεως S_x (εάν αυτός υπάρχει) είναι έγκυρος ή άκυρος:

$$\begin{aligned} S_x x &\rightarrow T_x x \\ S_x x' &\rightarrow U_x x' \\ S_x\} &\rightarrow W_x\} \end{aligned}$$

Οι παραπάνω κανόνες δικαιολογούνται ως εξής: όταν ο δρομέας δεί ότι σύμβολο στα δεξιά του *δεν* είναι σημαδεμένο, τότε μεταβαίνει στην κατάσταση T_x ώστε να αποφασίσει μη ντετερμινιστικά έαν θα προσθέσει αυτό το έγκυρο στοιχείο στην τρέχουσα τριάδα του ταιριάσματος. Εάν δει ότι το σύμβολο είναι σημαδεμένο τότε, μέσω της μετάβασης στην κατάσταση U_x, προχωράει για έλεγχο στο επόμενο στοιχείο, εάν υπάρχει. Εάν δεν υπάρχει άλλο επόμενο έγκυρο στοιχείο στη συνέχεια, ο δρομέας θα το αντιληφθεί φτάνοντας στο δεξί άκρο του συνόλου, δηλαδή τον χαρακτήρα }, και τότε, αφού στο τρέχον βήμα δεν έχει επιλέξει κανένα στοιχείο του συνόλου X, θα οπισθοδρομήσει ώστε να βρεί το πρώτο προς τα αριστερά μη σημαδεμένο στοιχείο για να το προσθέσει στην τρέχουσα τριάδα που κατασκευάζουμε.

Τώρα, είμαστε σε θέση να περιγράψουμε τους κανόνες που εκτελούν τις παραπάνω λειτουργίες. Αρχίζουμε από την κατάσταση T_x η οποία αποφασίζει μη ντετερμινιστικά εάν θα επιλέξει το στοιχείο στα δεξιά της ώς τη συνιστώσα από το σύνολο X στη δημιουργεία της τρέχουσας τριάδος:

$$
\begin{aligned}
T_x &\rightarrow T_{\nu\alpha\iota} \\
T_x &\rightarrow T_{o\chi\iota}
\end{aligned}
$$

όπου στην κατάσταση $T_{\nu\alpha\iota}$ επιλέγουμε τον τρέχοντα αριθμό, ενώ στην κατάσταση $T_{o\chi\iota}$ δεν τον επιλέγουμε και είναι αναγκαίο να προχωρήσουμε προς το επόμενο στοιχείο του συνόλου.

Από την κατάσταση $T_{\nu\alpha\iota}$ είναι αναγκαίο να αντιγράψουμε τον αριθμό στο δεξί άκρο του στιμιοτύπου μας:

$$
\begin{aligned}
T_{\nu\alpha\iota} &\rightarrow V \\
V* &\rightarrow *''V_* \\
V, &\rightarrow \alpha, \\
V\} &\rightarrow \alpha\}
\end{aligned}
$$

όπου $* \in \{x, 0, 1\}$. Στην κατάσταση V δηλαδή, αποθηκεύουμε το τρέχων σύμβολο που θα αντιγράψουμε. Οταν φτάσουμε στο δεξί άκρο του τρέχωντος στοιχείου, κάτι που το καταλαβαίνουμε όταν συναντήσουμε τον χαρακτήρα "," ή "}", τότε ξάρουμε ότι έχει τελειώσει η αντιγραφή του τρέχωντος αριθμού και είναι αναγκαίο να ξεσημαδέψουμε όλους τους χαρακτήρες του στοιχείου αυτού και να τους σημαδέψουμε ως έγκυρους μέσω της καταστάσεως α (ας θυμηθούμε ότι σημαδεύουμε με διαφορετικό τρόπο τους άκυρους αριθμούς-στοιχεία από τα στοιχεία των οποίων τους χαρακτήρες αντιγράφουμε έναν-έναν).

Στην κατάσταση V_* είναι αναγκαίο να αντιγράψουμε τον χαρακτήρα $*$ στην κατάλληλη θέση. Αγνοούμε όλους τους ενδιάμεσου χαρακτήρες και τον αντιγράφουμε στο τέλος του στιγμιοτύπου, πρίν από την παρένθεση της τρέχουσας τριάδος:

$$
\begin{aligned}
V_*\chi &\rightarrow \chi V_* \\
V_*\}\} &\rightarrow \beta *\}\} \\
V_x\}\} &\rightarrow ,\beta\{x\} \\
V_y\}\} &\rightarrow ,\beta y\}\} \\
V_z\}\} &\rightarrow ,\beta z\}\}
\end{aligned}
$$

όπου $\chi \in \{0, 1, 0', 1'\text{“}, \text{”}, \text{“}\}, \{\text{”}\}$. Δηλαδή αγνοούμε τα πάντα και μόλις συναντήσουμε τις δύο παρενθέσεις $\}\}$ τότε αντιγράφουμε τον χαρακτήρα $*$ και μεταβαίνουμε στην κατάσταση β η οποία θα αναλάβει να μεταφέρει τον δρομέα στο κατάλληλο σημείο ώστε να συνεχιστεί η αντιγραφή (η πρώτη παρένθεση αντιστοιχεί στην δεξιά παρένθεση του συνόλου z ενώ η δεύτερη, την δεξιά παρένθεση που περικλύει το στιγμίοτυπο):

$$\begin{aligned}
\chi\beta &\rightarrow \beta\chi \\
0''\beta &\rightarrow 0''V \\
1''\beta &\rightarrow 1''V \\
x''\beta &\rightarrow x''V \\
y''\beta &\rightarrow y''V \\
z''\beta &\rightarrow z''V
\end{aligned}$$

όπου $\chi \in \{0, 1, 0', 1'\text{“}, \text{”}, \text{“}\}, \{\text{”}, x, y, z\}$.

Παρατηρούμε ότι με τον ποιο πάνω τρόπο καταφέρνουμε όχι μόνο να αντιγράψουμε σωστά το τρέχον ψηφίο, αλλά να βάλουμε και τα σωστά άγκυστρα στις κατάλληλες θέσεις που θα περικλύουν την τρέχουσα τριάδα, καθώς και τους κατάλληλους χαρακτήρες διαχωρισμού μεταξύ των τριών στοιχείων της τριάδος.

Ξέρουμε ότι έχουμε τελειώσει με την αντιγραφή όταν βρεθούμε στην κατάσταση α οπότε και θα πρέπει να ξεσημαδέψουμε τους χαρακτήρες και να τους σημαδέψουμε ταυτόχρονα με εκείνον τον τρόπο ώστε να μπορούμε να τους αναγνωρίζουμε ώς άκυρους σε μελλοντικά βήματα:

$$\begin{aligned}
0''\alpha &\rightarrow \alpha 0' \\
1''\alpha &\rightarrow \alpha 0' \\
x''\alpha &\rightarrow \gamma x' \\
y''\alpha &\rightarrow \gamma y' \\
z''\alpha &\rightarrow \gamma z'
\end{aligned}$$

Η λειτουργία της καταστάσεως γ η οποία και θα περιγραφεί αναλυτικώς στη συνέχεια, είναι η εξής διαισθητικά: μόλις έχουμε επιλέξει κάποιο στοιχείο του συνόλου X, το έχουμε αντιγράψει και το έχουμε σημαδέψει καταλλήλως. Αρα είναι αναγκαίο να επιλεγούν τα στοιχεία των δύο επόμενων συνόλων που

θα αποτελούν την τρέχουσα τριάδα του τριμερούς ταιριάσματος. Αρα η κατάσταση γ μας μεταφέρει στην αρχή του επόμενου συνόλου, εάν υπάρχει (διότι μπορεί να τελιώσαμε στο τρέχον στάδιο την αντιγραφή κάποιου στοιχείου από το σύνολο Z):

$$
\begin{array}{rcl}
\gamma x' & \rightarrow & x'\gamma \\
\gamma y' & \rightarrow & y'\gamma \\
\gamma z' & \rightarrow & z'\gamma \\
\gamma x & \rightarrow & x\gamma \\
\gamma y & \rightarrow & y\gamma \\
\gamma z & \rightarrow & z\gamma \\
\gamma 0' & \rightarrow & 0'\gamma \\
\gamma 1' & \rightarrow & 1'\gamma \\
\gamma 0 & \rightarrow & 0\gamma \\
\gamma 1 & \rightarrow & 1\gamma \\
\gamma, & \rightarrow & ,\gamma \\
\gamma\},\{y & \rightarrow & \},\{S_y y \\
\gamma\},\{z & \rightarrow & \},\{S_z z \\
\gamma\},\{x & \rightarrow & \delta\},\{x
\end{array}
$$

Οι αρχικοί κανόνες είναι είναι προφανείς: αγνοούν οτιδήποτε ώστε να προχωρήσει προς τα δεξιά ο δρομέας. Οταν συναντήσουμε του χαρακτήρες "$\},\{$" τότε, ανάλογα με το γράμμα που έπεται, μεταβαίνουμε και στην αντίστοιχη κατάσταση S_y ή S_z για την επιλογή στοιχείου από το αντίστοιχο σύνολο. Ομως, όπως ήδη είπαμε, μπορεί να βρισκόμαστε ήδη στο τελευταίο σύνολο Z. Τότε, αυτό που θα συναντήσουμε μετακινούμενοι προς τα δεξιά, είναι η αρχική τριάδα του συνόλου των τριάδων του ταιριάσματος, που ως πρώτη συνιστώσα θα έχει κάποιο στοιχείο του συνόλου X, και όταν συναντήσουμε μια τέτοια κατάσταση, μεταβαίνουμε στην δ της οποίας ο ρόλος είναι να μας μεταφέρει στην αρχή του συνόλου X ώστε να μεταβούμε στο επόμενο βήμα και να δημιουργήσουμε την επόμενη τριάδα του τριμερούς ταιριάσματος:

$$
\begin{aligned}
0\delta &\rightarrow \delta 0\\
1\delta &\rightarrow \delta 1\\
0'\delta &\rightarrow \delta 0'\\
1'\delta &\rightarrow \delta 1'\\
x\delta &\rightarrow \delta x\\
y\delta &\rightarrow \delta y\\
z\delta &\rightarrow \delta z\\
x'\delta &\rightarrow \delta x'\\
y'\delta &\rightarrow \delta y'\\
z'\delta &\rightarrow \delta z'\\
,\delta &\rightarrow \delta,\\
\},\{\delta &\rightarrow \delta\},\{\\
\{\{\delta &\rightarrow \{\{S_x
\end{aligned}
$$

Τώρα, είναι αναγκαίο να περιγραφεί η λειτουργία της καταστάσεως U_x η οποία διαισθητικά είναι η εξής: έχουμε ήδη ανακαλύψει ότι το στοιχείο στα δεξιά του δρομέα μας είναι μη έγκυρο (μέσω της καταστάσεως S_x) και, ως εκ τούτου, έχουμε μεταφερθεί στην κατάσταση U_x της οποία ο σκοπός είναι να ελένξει το επόμενο στοιχείο του συνόλου, εάν φυσικά κάτι τέτοιο υπάρχει (αντίστοιχες είναι και οι λειτουργίες των U_y και U_z):

$$
\begin{aligned}
U_x 0' &\rightarrow 0' U_x\\
U_x 1' &\rightarrow 1' U_x\\
U_x, &\rightarrow , S_x\\
U_x\} &\rightarrow W_x\}
\end{aligned}
$$

Δηλαδή, με τους παραπάνω κανόνες, βρίσκουμε και ελέγχουμε, μέσω της S_x, το επόμενο στοιχείο του τρέχοντος συνόλου. Εάν δεν υπάρχει επόμενο στοιχείο, θα βρεθούμε στον χαρακτήρα } οπότε, όπως περιγράψαμε και πιο πάνω, είναι αναγκαίο να γυρίσουμε προς τα αριστερά και να επιλέξουμε το πρώτο έγκυρο στοιχείο που θα συναντήσουμε, εάν φυσικά υπάρχει (κατάσταση W_x). Αλλιώς, εάν δεν υπάρχει δηλαδή, έχουμε ήδη τελειώσει με την παραγωγή

των τριάδων του τριμερούς ταιριάσματος (καθώς όλα τα στοιχεία του τρέχοντος συνόλου είναι άκυρα και τα σύνολα έχουν την ίδια πληθυκότητα και απομένει να δημιουργήσουμε τις επιπλέον τριάδες του στιγμιοτύπου μας.

Τώρα, μπορούμε πλέον να περιγράψουμε την κατάσταση W_x, την λειτουργία της οποίας περιγράψαμε πιο πάνω:

$$\begin{aligned} 0'W_x &\rightarrow W_x0' \\ 1'W_x &\rightarrow W_x1' \\ x'W_x &\rightarrow W_xx' \\ 0W_x &\rightarrow W'0 \\ 1W_x &\rightarrow W'1 \\ ,W_x &\rightarrow W_x, \\ \{\{W_x &\rightarrow \{\{\Omega \end{aligned}$$

Παρατηρούμε ότι η περίπτωση xW_x δεν λείπει: δεν πρόκειται ποτέ να συναντήσουμε μη σημαδεμένο τον χαρακτήρα x στην κατάσταση W_x καθώς ο χαρακτήρας x βρίσκεται πάντοτε στο αριστερό άκρο του αριθμού, ενώ εμείς κεινούμαστε προς τα αριστερά. Αρα, όταν πρωτοσυναντήσουμε έναν έκγυρο αριθμό, αυτό θα γίνει μέσω κάποιου ψηφίου και αμέσως θα αλλάξουμε κατάσταση (W') ώστε να τον αντιγράψουμε στην κατάλληλη θέση:

$$\begin{aligned} W'0 &\rightarrow 0W' \\ W'1 &\rightarrow 1W' \\ W'x &\rightarrow Vx \end{aligned}$$

Ενόσω συνεχίζουμε να ψάχνουμε προς τα αριστερά για το πρώτο έγκυρο στοιχείο του τρέχοντος συνόλου, ενδέχεται τέτοιο στοιχείο να μην υπάρχει, με την έννοια ότι όλα είναι σημαδεμένα. Σε αυτή τη περίπτωση έχουμε τελειώσει με την παραγωγή των τριάδων του τριμερούς ταιριάσματος, καθώς δεν απομένει κάποιο στοιχείο του συνόλου X να επιλέξουμε και άρα, αφού και τα τρία σύνολα έχουν την ίδια πληθυκότητα, δεν μπορούμε να σχηματίσζουμε νέα έγκυρη ως προς το ταίριασμα τριάδα. Τότε μεταβαίνουμε στην κατάσταση Ω η οποία θα αρχίσει την παραγωγή των υπόλοιπων τριάδων του ταιριάσματος.

Οπως είπαμε, όταν φτάσουμε προ ενός μη σημαδεμένου αριθμού, μη ντετερμινιστικά αποφασίζουμε εάν θα αποτελεί το τρέχον στοιχείο του συνόλου X

της τρέχουσας τριάδος που δημιουργούμε (κατάσταση $T_{\nu\alpha\iota}$) ή όχι (κατάσταση $T_{o\chi\iota}$). Ενώ περιγράψαμε την λειτουργία της καταστάσεως $T_{\nu\alpha\iota}$, απομένει να περιγράψουμε αυτή της $T_{o\chi\iota}$. Στην περίπτωση που βρεθουμε στην κατάσταση αυτή, το μόνο που μπορούμε να κάνουμε είναι να προχωρήσουμε στην επόμενο αριθμό, εάν φυσικά υπάρχει, και να επιστρέψουμε στην σαρχική κατάσταση ελέγχου του αριθμού αυτού ως προς την εγκυρότητά του:

$$\begin{aligned}
T_{o\chi\iota}x &\rightarrow x\,T_{o\chi\iota}\\
T_{o\chi\iota}y &\rightarrow y\,T_{o\chi\iota}\\
T_{o\chi\iota}z &\rightarrow z\,T_{o\chi\iota}\\
T_{o\chi\iota}0 &\rightarrow 0\,T_{o\chi\iota}\\
T_{o\chi\iota}1 &\rightarrow 1\,T_{o\chi\iota}\\
T_{o\chi\iota}, &\rightarrow ,\,S_x\\
T_{o\chi\iota}\} &\rightarrow W_x\}
\end{aligned}$$

Η παραπάνω διαδικασία ακολουθείται, με τις προφανείς προσαρμογές, και για τα στοιχεία των επόμενων δύο συνόλων. Οταν τελειώσει η παραπάνω διαδικασία, στο i-οστό βήμα, θα έχουμε αντιγράψει στο τέλος της συμβολοσειράς του στιγμιοτύπου τρεις αριθμούς-στοιχεία, ένα από κάθε σύνολο, έστω x_{i_j}, y_{i_k} και z_{i_l} και επιπλέον η τριάδα ατή έχει τη σωστή μορφή $\{x_{i_j}, y_{i_k}, z_{i_l}\}$.

Παρατηρούμε ότι στην κατάσταση Ω μεταβαίνουμε όταν έχουμε τελειώσει με την παραγωγή της τρέχουσας τριάδος. Είμαστε πλέον έτοιμοι να αρχίσουμε την παραγωγή των επιπλέον επιθυμητών τριάδων του στιγμιοτύπου μας.

8.4 Παραγωγή των υπόλοιπων τριάδων του στιγμιοτύπου

Πριν προχωρήσουμε στην παραγωγή των υπολοίπων τριάδων του στιγμιοτύπου μας, είναι αναγκαίο, όπως πάντα, να περιγράψουμε τη βασική ιδέα του τρόπου ο ποίος θα μας δώσει τη δυνατότητα να παράξουμε όλες τις επιπλέον τριάδες του στιγμιοτύπου μας.

Στο i-οστό βήμα-στάδιο της διαδικασίας αυτής, θα "δουλεύουμε" με τη i-οστή τριάδα του ταιριάσματος που κατασκευάσαμε στη προηγούμενη ενότητα. Η τριάδα αυτή έστω ότι είναι η $\{x_{i_j}, y_{i_k}, z_{i_l}\}$. Η μέθοδος που θα περιγράψουμε στη συνέχεια, θα μας δίνει τη δυνατότητα να κατασκευάζουμε στο i-οστό βήμα όλες τις επιθυμητές τριάδες των οποίων η πρώτη (x) συνιστώσα είναι η x_{i_j}:

- Γενικά, όταν δουλεύουμε με κάποια τριάδα του ταιριάσματος που έχουμε

ήδη δημιουργήσει, θα έχουμε δύο μεταβλητές - καταστάσεις: την Φ και την Ψ.

- η λειτουργία της καταστάσεως Φ είναι διαισθητικά η εξής: ο ρόλος της είναι να δημιουργεί τις επιθυμητές εκείνες τριάδες $\{x_{i_p}, y_{i_q}, z_{i_r}\}$ για τις οποιές ισχύει ότι $x_{i_j} \equiv x_{i_p}$ και επιπλέον για τις υπόλοιπες δύο συνιστώσες να ισχύει ότι εάν παραθέσουμε τους δύο αντίστοιχους αριθμούς, ο συνολικός αριθμός να είναι *λεξικογραφικά μικρότερος* από τον αντίστοιχο της i-οστής τριάδος του ταιριάσματος. Δηλαδή για τα στοιχεία y_{i_q} και z_{i_r} να ισχύει ότι είτε $y_{i_q} \prec y_{i_k}$ και το z_{i_r} οτιδήποτε, είτε να ισχύει ότι $y_{i_q} \equiv y_{i_k}$ και $z_{i_r} \prec z_{i_l}$.

- αντίστοιχα για την κατάσταση Ψ η διαισθητική της λειτουργία είναι η εξής: ο ρόλος της είναι να δημιουργεί τις επιθυμητές εκείνες τριάδες $\{x_{i_p}, y_{i_q}, z_{i_r}\}$ για τις οποιές ισχύει ότι $x_{i_j} \equiv x_{i_p}$ και επιπλέον για τις υπόλοιπες δύο συνιστώσες να ισχύει ότι εάν παραθέσουμε τους δύο αντίστοιχους αριθμούς, ο συνολικός αριθμός να είναι *λεξικογραφικά μεγαλύτερος* από τον αντίστοιχο της i-οστής τριάδος του ταιριάσματος. Δηλαδή για τα στοιχεία y_{i_q} και z_{i_r} να ισχύει ότι είτε $y_{i_q} \succ y_{i_k}$ και το z_{i_r} οτιδήποτε, είτε να ισχύει ότι $y_{i_q} \equiv y_{i_k}$ και $z_{i_r} \succ z_{i_l}$.

Με άλλα λόγια οι καταστάσεις Φ και Ψ παράγουν τις τριάδες εκείνες οι οποίες έχουν ώς πρώτη συνιστώσα το στοιχείο της αρχικής i-οστής τριάδος του ταιριάσματος ενώ το υπόλοιπο κομμάτι (με τις συνιστώσες y και z) είναι λεξικογραφικά *μικρότερο* και *μεγαλύτερο* αντίστοιχα.

Με τον τρόπο αυτό, είναι προφανές ότι δημιουργούνται *όλες* οι επιθυμητές τριάδες που θέλουμε να προσθέσουμε στο στιγμιότυπό μας. Αφού θεωρούμε μία-μία τις τριάδες του ταιριάσματος, δυνητικά θα δημιουργήσουμε όλες τις επιθυμητές τριάδες για όλα τα $x \in X$. Επιπλέον, όπως θα φανεί πιο καθαρά και στη συνέχεια, δεν είναι δυνατόν να παραχθούν διπλότυπα τριάδων, αφού κάθε φορά εφαρμόζουμε την εκάστοτε διαδικασία στη τριάδα που μόλις δημιουργήσαμε. Ο τρόπος παραγωγής των νέων τριάδων εξασφαλίζει ότι θα είναι διαφορετικές από το αντίστοιχο πρότυπό τους και άρα όλες οι τριάδες θα είναι διαφορετικές μεταξύ τους!

Για να γίνει πιο εύκολα κατανοητή η παραπάνω μέθοδος, ας θεωρήσουμε ως παράδειγμα ότι βρισκόμαστε στο i-οστό στάδιο της διαδικασίας και η τρέχουσα τριάδα που δουλεύουμε είναι η $\{x01, y10, z10\}$. Με βάση αυτή τη τριάδα ωε πρότυπο, ο σκοπός μας είναι να δημιουργηθούν όλες εκείνες οι τριάδες που έχουν ως πρώτη (x) συνιστώσα το στοιχείο $x01$. Η κατάσταση Φ θα κρατήσει την πρώτη συνιστώσα $(x01)$ σταθερή και θα δημιουργήσει όλες τις επιθυμητές τριάδες οι οποίες είναι λεξικογραφικά μικρότερες της παραπάνω τριάδος π.χ τις

$\{x01, y10, z00\}$ και $\{x01, y01, z10\}$. Αντίστοιχα η κατάσταση Ψ θα δημιουργήσει όλες εκείνες τις επιθυμητές τριάδες οι οποίες είνα λεξικογραφικά μεγαλύτερες της παραπάνω τριάδος π.χ τις τριάδες $\{x01, y10, z11\}$, $\{x01, y11, z00\}$ και $\{x01, y11, z10\}$.

Αφού περιγράψαμε σε πολύ γενικές γραμμές τη βασική ιδέα του τρόπου με τον οποίο θα δημιουργηθούν οι επιθυμητές τριάδες, και πριν περάσουμε στην παράθεση των αντίστοιχων κανόνων, είναι αναγκαίο να περιγραφεί βήμα προς βήμα η διαδικασία πιο αναλυτικά και σε φυσική γλώσσα. Επειτα, θα "μεταφράσουμε" τα αντίστοιχα βήματα σε κανόνες που θα μας δίνουν το επιθυμητό αποτέλεσμα. Θα περιγράψουμε την διαδικασία για την περίπτωση όπου επιθυμούμε τριάδες λεξικογραφικά μικρότερες της τριάδος προτύπου που θεωρούμε στο εκάστοτε τρέχον βήμα. Η επέκταση για παραγωγή τριάδων λεξικογραφικά μεγαλύτερων είναι τελείως προφανής.

1. Εστω ότι η τρέχουσα τριάδα που δουλεύουμε είναι η $\{x_{i_j}, y_{i_k}, z_{i_l}\}$ (δηλαδή βρισκόμαστε στο i-οστό βήμα της διαδικασίας).

2. Αρχικά, αποφασίζουμε μη ντετερμινιστικά εάν επιθυμούμε μια νέα τριάδα λεξικογραφικά μικρότερη της αρχικής.

   ▷ Εάν ναι, τότε ελέγχουμε κατά πόσον είναι δυνατή η παραγωγή νέας τριάδος. Ο έλεγχος αυτός απαιτεί *τουλάχιστον* ένα στοιχείο εκ των y_{i_k} και z_{i_l} να είναι διάφορο του μηδενός:

   – Εάν ο έλεγχος πετύχει, τότε προχωράμε στο βήμα 3. Αλλιώς

   – Και οι δύο συνιστώσες y_{i_k} και z_{i_l} είναι μηδέν, οπότε δεν μπορούμε να δημιουργήσουμε μια νέα λεξικογραφικά μικρότερη τριάδα και άρα προχωράμε στην επόμενη τριάδα του ταιριάσματος (εάν υπάρχει) ώστε να εφαρμόσουμε την ίδια διαδικασία.

   ▷ Εάν όχι, τότε προχωράμε στη επόμενη τριάδα του ταιριάσματος.

3. Είμαστε σε θέση να δημιουργήσουμε μια νέα λεξικογραφικά μικρότερη τριάδα. Στο τρέχον βήμα αντιγράφουμε την τριάδα πρότυπο $\{x_{i_j}, y_{i_k}, z_{i_l}\}$ δεξιά του προτύπου και μετακινούμε τον δρομέα αμέσως αριστερά της συνιστώσας y_{i_k} στο αντίγραφο που μόλις δημιουργήσαμε.

4. Αποφασίζουμε μη ντετερμινιστικά εάν θα μειώσουμε το στοιχείο y_{i_k} ή όχι:

   ▷ Εάν αποφασίσουμε να το μειώσουμε, τότε ελέγχουμε εάν μπορεί να μειωθεί (γιατί μπορεί να είναι και μηδέν):

- Εαν μπορεί να μειωθεί, το μειώνουμε κατά το δοκούν μη ντετερμινιστικά και προχωράμε για την αυξομείωση του z_{i_l}.
- Εάν δε μπορεί να μειωθεί (προφανώς επειδή είναι μηδέν), τότε ξέρουμε ότι $z_{i_l} \neq 0$ (αφού *τουλάχιστον* ένα στοιχείο εκ των y_{i_k} και z_{i_l} να είναι διάφορο του μηδενός άρα το πολύ ένα είναι μηδέν). Αρα προχωράμε και μειώνουμε μη ντετερμινιστικά το στοιχείο z_{i_l}.

▷ Εάν αποφασίσουμε να μη μειώσουμε το στοιχείο y_{i_k} τότε μετακινούμε τον δρομέα στην αρχή του στοιχείου z_{i_l} και ελέγχουμε εάν μπορεί να μειωθεί:

- Εάν μπορεί να μειωθεί τότε το μειώνουμε μη ντετερμινιστικά κατά βούληση.
- Εάν δεν μπορεί να μειωθεί (προφανώς επειδή $z_{i_l} = 0$) τότε ξέρουμε ότι $y_{i_k} \neq 0$ και μεταφέρουμε τον δρομέα στην αρχή του y_{i_k} ώστε, αναιρώντας την προηγούμενη μη ντετερμινιστική επιλογή της μη μείωσης του y_{i_k}, να αρχίσουμε να τον μειώνουμε μη ντετερμινιστικά

5. Εφαρμόζουμε την παραπάνω διαδικασία στην τριάδα που μόλις δημιουργήσαμε μέχρις ότου να μη μπορούμε να κατασκευάσουμε επιπλέον τριάδες (επειδή και οι δύο συνιστώσες εκφυλίστηκαν στο μηδέν) είτε μέχρι να αποφασίσουμε μη ντετερμινιστικά ότι τελειώσαμε με τις τριάδες του στιγμιοτύπου που έχουν ως πρώτη (x) συνιστώσα το στοιχείο x_{i_j}. Σε κάθε περίπτωση, μεταβαίνουμε στην επόμενη τριάδα του ταιριάσματος για να κατασκευάσουμε όλες τις επιθυμητές τριάδες με πρώτη συνιστώσα την αντίστοιχη x συνιστώσα της τρέχουσας τριάδος του ταιριάσματος.

Τώρα είμαστε πλέον σε θέση να περιγράψουμε τους κανόνες εκείνους οι οποίοι θα υλοποιούν την παραπάνω διαδικασία. Σε πρώτη φάση, είναι απαραίτητο να μεταφέρουμε τον δρομέα στην αρχή της πρώτης τριάδος του ταιριάσματος που δημιουργήσαμε στη προηγούμενη ενότητα. Για να γίνει αυτό, μετακινούμε τον δρομέα προς τα δεξιά, μέχρι να συναντήσουμε τα στοιχεία του συνόλου Z. Μόλις συναντήσουμε αυτό το σύνολο, το θυμόμαστε σε μιά κατάσταση: ξέρουμε ότι αμέσως μετά το τέλος αυτού του συ˜νόλου ακολουθούν οι τριάδες του ταιριάσματος.

Ας θυμηθούμε σε ποιά κατάσταση είχαμε αφήσει το στιγμιότυπό μας: μόλις είχαμε ολοκληρώση την παραγωγή όλων των τριάδων του ταιριάσματος για το στιγμιότυπό μας και αυτό το καταλάβαμε καθώς δεν υπήρχε διαθέσιμο στοιχείο στο σύνολο X ώστε να χρησιμοποιηθεί για μιά νέα τριάδα του ταιριάσματος. Δηλαδή όλα τα στοιχεία του συνόλου X (και κατ'επέκταση και των άλλων

συνόλων) είναι σημαδεμένα. Αρα, είναι αναγκαίο να μεταφέρουμε τον δρομέα στο επιθυμητό σημείο και, παράλληλα, να ξεσημαδεύουμε τους σημαδεμένους χαρακτήρες των στοιχείων των τριών συνόλων:

$$
\begin{aligned}
\Omega x' &\rightarrow x\Omega \\
\Omega 0' &\rightarrow 0\Omega \\
\Omega 1' &\rightarrow 1\Omega \\
\Omega y' &\rightarrow y\Omega \\
\Omega, &\rightarrow ,\Omega \\
\Omega\}, \{ &\rightarrow \}, \{\Omega \\
\Omega z' &\rightarrow z\Omega' \\
\Omega' z' &\rightarrow z\Omega' \\
\Omega' 0' &\rightarrow 0\Omega' \\
\Omega' 1' &\rightarrow 1\Omega' \\
\Omega', &\rightarrow ,\Omega \\
\Omega'\}, \{ &\rightarrow \}, \{\Gamma
\end{aligned}
$$

Στην κατάσταση Γ στη παρούσα φάση, είμαστε προ της πρώτης τριάδος του ταιριάσματος, έστω ότι αυτή είναι η $\{x_{1_p}, y_{1_q}, z_{1_r}\}$. Θέλουμε να αποφασίσουμε μη ντετερμινιστικά εάν επιθυμούμε μιά νέα τριάδα αρχικά λεξικογραφικά μικρότερη της τρέχουσας τριάδος, και στη συνέχεια, έάν επιιθυμούμε μια τριάδα λεξικογραφικά μεγαλύτερη της τρέχουσας τριάδος:

$$
\begin{aligned}
\Gamma &\rightarrow \Phi \\
\Phi &\rightarrow \Phi_{\nu\alpha\iota} \\
\Phi &\rightarrow \Phi_{o\chi\iota} \\
\Phi_{o\chi\iota} &\rightarrow \Psi \\
\Psi &\rightarrow \Psi_{\nu\alpha\iota} \\
\Psi &\rightarrow \Psi_{o\chi\iota}
\end{aligned}
$$

Σημείωση: θα πρέπει να έχουμε μιά μετάβαση η οποία μόλις τελειώσει την παραγωγή όλων των επιθυμητών λεξικογραφικά μικρότερων τριάδων μέσω της καταστάσεως Φ, να μεταβαίνει στην κατάσταση Ψ για να δημιουργήσει τις όποιες τριάδες επιθυμούμε οι οποίες είναι λεξικογραφικά μεγαλύτερες της

τριαδός προτύπου που θεωρούμε στον τρέχον βήμα μας. Τώρα είναι αναγκαίο να εξηγήσουμε τις λειτουργίες των παραπάνω καταστάσεων. Επιπλέον, εάν αποφασίσουμε οτι θέλουμε λεξικογραφικά μικρότερες τριάδες σε σχέση με την τρέχουσα, δηλαδή μεταβούμε στην κατάσταση $\Phi_{οχι}$, τότε είναι αναγκαίο να σημαδέψουμε την τρέχουσα τριάδα, ώστε να την αναγνωρίσουμε στη συνέχεια, όταν, και εάν, θελήσουμε μέσω της καταστάσεως Ψ να δημιουργήουμε όλες τις λεξικογραφικά μεγαλύτερες τριάδες σε σχέση με την τρέχουσα τριάδα:

$$\{\Phi_{ναι} \rightarrow \bar{\{}\Phi_{ναι}$$

Οπως φαίνεται και από του παραπάνω κανόνες, όταν φτάσουμε προ της τρέχουσας τριάδος μεταβαίνουμε στην κατάσταση Φ για την παραγωγή όλων των επιθυμητών λεξικογραφικά μικρότερων τριάδων. Ετσι, μη ντετερμινιστικά απόφασίζουμε εάν θέλουμε κάποια τέτοια τριάδα και, ανάλογα, μεταβαίνουμε στην κατάλληλη κατάσταση: $\Phi_{ναι}$ εάν θέλουμε μια νέα τριάδα λεξικογραφικά μικρότερη της τρέχουσας ή $\Phi_{οχι}$ εάν δε θέλουμε κάποια νέα τριάδα. Αν αποφασίσουμε να μεταβούμε στην κατάσταση $\Phi_{οχι}$, τότε είναι αναγκαίο να μεταβούμε άμεσα στην κατάσταση Ψ και να εκτελέσουμε τις ίδιες λειτουργίες για την παραγωγή λεξικογραφικά μεγαλύτερων τριάδων.

Μόλις μεταβούμε στην κατάσταση $\Phi_{ναι}$, πριν ακόμα δημιουργήσουμε την νέα τριάδα που επιθυμούμε, είναι απαραίτητο να ελένξουμε το κατά πόσον είναι δυνατή η δημιουργία μιας νέας τριάδος. Αυτός ο έλεγχος (που θα γίνει μέσω της καταστάσεως Φ') απαιτεί *τουλάχιστον* μία από τις y ή z συνιστώσες της τρέχουσας τριάδος να είναι διαφορετική του μηδενός του μηδενός:

$$\begin{aligned}
\Phi_{ναι}x &\rightarrow x\Phi_{ναι} \\
\Phi_{ναι}0 &\rightarrow 0\Phi_{ναι} \\
\Phi_{ναι}1 &\rightarrow 1\Phi_{ναι} \\
\Phi_{ναι}, &\rightarrow ,\Phi_{ναι} \\
\Phi_{ναι}y &\rightarrow y\Phi' \\
\Phi'0 &\rightarrow 0\Phi' \\
\Phi'z &\rightarrow z\Phi' \\
\Phi', &\rightarrow ,\Phi' \\
\Phi'1 &\rightarrow 1\Delta_{ναι} \\
\Phi'\} &\rightarrow \Delta_{οχι}\}
\end{aligned}$$

Η λειτουργία των παραπάνω κανόνων είναι η εξής: αρχικά προσσπερνάμε όλο το στοιχείο x της τρέχουσας τριάδος ώστε να ελένξουμε τα στοιχεία y και z. Αρχίζουμε να ελέγχουμε ένα προς ένα τα ψηφία τους μέχρι σε κάποιο σημείο να σηναντίσουμε κάποιον άσσο το οποίο σημαίνει ότι το τρέχον στοιχείο (δεν μας ενδιαφέρει πιο είναι) *δεν* είναι μηδέν, άρα τούλάχιστον ένα στοιχείο εκ των y και z μπορεί να μειωθεί ώστε να δημιουργηθεί μια λεξικογραφικά μικρότερη συμβολοσειρά. Αμέσως μεταβαίνουμε στην κατάσταση $\Delta_{ναι}$. Αντίθετα, εάν διαπιστώσουμε ότι όλοι οι χαρακτήρες είναι μηδέν, τότε μεταβαίνουμε στην συμβολοσειρά $\Delta_{οχι}$ η οποία θα μας οδηγήσει στην κατάσταση Ψ για να κατασκευάσουμε τυχόν επιθυμητές τριάδες λεξικογραφικά μεγαλύτερες της τρέσουσας τριάδος:

$$
\begin{aligned}
0\Delta_{οχι} &\rightarrow \Delta_{οχι}0\\
1\Delta_{οχι} &\rightarrow \Delta_{οχι}1\\
y\Delta_{οχι} &\rightarrow \Delta_{οχι}y\\
z\Delta_{οχι} &\rightarrow \Delta_{οχι}z\\
,\Delta_{οχι} &\rightarrow \Delta_{οχι},\\
x\Delta_{οχι} &\rightarrow \Psi x
\end{aligned}
$$

Τώρα, στην κατάσταση $\Delta_{ναι}$, έχουμε βρεί ότι μπορούμε να δημιουργήσουμε μια νέα λεξικογραφικά μικρότερη τριάδα, οπότε είναι απαραίτητο να αρχίσουμε τη δημιουργία της. Οπως είπαμε, ο τρόπος που θα δημιουργηθεί είναι ο εξής: αρχικά αντιγράφουμε την τρέχουσα τριάδα στα δεξιά της και μετά αρχίζουμε την κατάλληλη αλλαγή των συνιστωσών ώστε να δημιουργηθεί η νέα επιθυμητή τριάδα. Αρχικά περιγράφουμε την αντιγραφή η οποία θα γίνει με τον γνωστό μας τρόπο μέσω της καταστάσεως ζ (πρώτα είναι απαραίτητο να μεταφερθεί ο δρομέας στην κατάλληλη θέσημ στην αρχή της τρέχουσας τριάδος προς αντιγραφή):

$$
\begin{aligned}
0\Delta_{ναι} &\rightarrow \Delta_{ναι}0\\
1\Delta_{ναι} &\rightarrow \Delta_{ναι}1\\
y\Delta_{ναι} &\rightarrow \Delta_{ναι}y\\
z\Delta_{ναι} &\rightarrow \Delta_{ναι}z\\
,\Delta_{ναι} &\rightarrow \Delta_{ναι},\\
x\Delta_{ναι} &\rightarrow \zeta x
\end{aligned}
$$

τώρα είμαστε έτοιμοι να αρχίσουμε την διαδικασία της αντιγραφής:

$$
\begin{aligned}
\zeta * &\rightarrow *'\zeta_* \\
\zeta_* \# &\rightarrow \#\zeta_* \\
\zeta_x \} &\rightarrow \eta, \{x\}, \\
\zeta_* \}, \{ &\rightarrow \}, \{\zeta'_* \\
\zeta'_* \# &\rightarrow \#\zeta'_* \\
\zeta'_* \} &\rightarrow \eta * \} \\
\#\eta &\rightarrow \eta \# \\
\}, \{\eta &\rightarrow \eta\}, \{ \\
\#'\eta &\rightarrow \#'\zeta \\
\zeta \} &\rightarrow \theta\}
\end{aligned}
$$

όπου $*, \# \in \{y, z, 0, 1\text{“,”}\}$ και $\#' \in \{x', y', z', 0', 1'\text{“,'”}\}$. Με άλλα λόγια, βρίσκουμε το τρέχον σύμβολο προς αντιγραφή, το σημαδεύουμε και θυμόμαστε ποιό είναι στην κατάλληλη κατάσταση. Επετιτα το αντιγράφουμε στο κατάλληλο σημείο και γυρίζουμε πίσω ώστε να συνεχίσουμε την παραπάνω διαδικασία. Τελειώνουμε όταν ο τρέχον χαρακτήρας δεξιά του δρομέα είναι το δεξί άγκυστρο } οπότε και μεταβαίνουμε στην κατάσταση θ ο ρόλος της οποίας είναι να ξεσημαδέψει τους σημαδεμένους χαρακτήρες:

$$
\begin{aligned}
0'\theta &\rightarrow \theta 0 \\
1'\theta &\rightarrow \theta 1 \\
,'\theta &\rightarrow \theta, \\
y'\theta &\rightarrow \theta y \\
z'\theta &\rightarrow \theta z \\
x'\theta &\rightarrow \theta' x
\end{aligned}
$$

Στην κατάσταση θ' είμαστε έτοιμοι να μεταβούμε στο νέο αντίγραφο της τρέχουσας τριάδας και να κάνουμε τις αναγκαίες αλλαγές ώστε να δημιουργηθεί η επιθυμητή, λεξικογραφικά μικρότερη, τριάδα. Αρχικά, μετακινούμε το δρομέα στην αρχή του αντιγράφου που μόλις δημιουργήσαμε:

$$
\begin{aligned}
\theta' x &\rightarrow x\theta' \\
\theta' 0 &\rightarrow 0\theta' \\
\theta' 1 &\rightarrow 1\theta' \\
\theta' , &\rightarrow , \theta' \\
\theta' y &\rightarrow y\theta' \\
\theta' z &\rightarrow z\theta' \\
\theta' \}, \{ &\rightarrow \}, \{ \vartheta
\end{aligned}
$$

Τώρα, μέσω της καταστάσεως
$vartheta$, είναι απαραίτητο να μετακινηθούμε προς την τρέχουσα y συνιστώσα για να αρχίσει η λειτουργία της δημιουργίας της επιθυμητής τριάδος:

$$
\begin{aligned}
\vartheta x &\rightarrow x\vartheta \\
\vartheta 0 &\rightarrow 0\vartheta \\
\vartheta 1 &\rightarrow 1\vartheta \\
\vartheta , &\rightarrow , \vartheta \\
\vartheta y &\rightarrow \kappa y
\end{aligned}
$$

Τώρα, όπως ήδη περιγράψαμε στη σκιαγράφιση της μεθόδου, είναι αναγκαίο να αποφασίσουμε μη ντετερμινιστικα εάν θα μειώσουμε τη συνιστώσα y ή όχι ώστε να δημιουργηθέι μια τριάδα λεξικογραφικά μικρότερη από την αρχική (υπ'όψιν ότι δεν ξέρουμε εάν η τρέχουσα y συνιστώσα είναι ή όχι μηδέν):

$$
\kappa \rightarrow \kappa_{\nu\alpha\iota} \mid \kappa_{o\chi\iota}
$$

Προφανώς, στην κατάσταση $\kappa_{\nu\alpha\iota}$ αποφασίζουμε να μειώσουμε, εφ'όσον είναι δυνατόν, το στοιχείο y, ενώ στην κατάσταση $\kappa_{o\chi\iota}$ αποφασίζουμε να το αφήσουμε αναλλοίωτο, μειώνοντας ανάλογα το στοιχείο z, πάλι εφ'όσον αυτό είναι δυνατόν. Αρχικά περιγράφουμε τη διαδικασία που ακολουθούμε στην περίπτωση που η μη ντετερμινιστική επιλογή μας φέρει στην κατάσταση $\kappa_{\nu\alpha\iota}$. Σε πρώτη φάση, είναι απαραίτητο να ελένξουμε κατά πόσον ειναι δυνατή η μείωση της y συνιστώσας:

$$
\begin{aligned}
\kappa_{\nu\alpha\iota} y &\rightarrow y \kappa_{\nu\alpha\iota} \\
\kappa_{\nu\alpha\iota} 0 &\rightarrow 0 \kappa_{\nu\alpha\iota} \\
\kappa_{\nu\alpha\iota} 1 &\rightarrow 1 \kappa' \\
\kappa_{\nu\alpha\iota}, &\rightarrow , \kappa''
\end{aligned}
$$

Δηλαδή, μετακινούμε προς τα δεξιά τον δρομέα ελέγχοντας ταυτόχρονα εάν η τρέχουσα y συνιστώσα είναι ή όχι μηδέν. Εάν στην πορεία συναντήσουμε κάποιον άσσο, μέσα στα όρια του y προφανώς, τότε μεταβαινουμε στην κατάσταση κ' η οποία ξέρει ότι η y συνιστώσα δεν είναι μηδέν. Αρα τώρα, είναι αναγκαίο να αρχίσουμε να μειώνουμε μη ντετερμινιστικά τη τρέχουσα y συνιστώσα. Αντίθετα, εάν δε βρούμε κανέναν άσσο στη συνιστώσα y, τότε προφανώς αυτή η συνιστώσα είναι μηδέν. Μεταβαίνουμε στην κατάσταση κ'' η οποία, ξέροντας ότι το τρέχον y είναι μηδέν, θα συνεχίσει προς τα δεξιά για να μειώσει τη z συνιστώσα η οποία όπως έχουμε ήδη πει δεν μπορεί να είναι μηδέν. Αρχικά περιγράφουμε την λειτουργία της κ':

$$
\begin{aligned}
\kappa' 0 &\rightarrow 0 \kappa' \\
\kappa' 1 &\rightarrow 1 \kappa' \\
\kappa', &\rightarrow \lambda,
\end{aligned}
$$

Ουσιαστικά, φέρνουμε τον δρομέα στο δεξί άκρο της τρέχουσας y συνιστώσας και, μόλις γίνει αυτό, μεταβαίνουμε στην κατάσταση λ της οποίας ο ρόλος είναι να μειώσει κατά μία μονάδα τη τρέχουσα συνιστώσα y:

$$
\begin{aligned}
0\lambda &\rightarrow \lambda 1 \\
1\lambda &\rightarrow \lambda' 0 \\
0\lambda' &\rightarrow \lambda' 0 \\
1\lambda' &\rightarrow \lambda' 1 \\
y\lambda' &\rightarrow \mu y
\end{aligned}
$$

Δηλαδή, μειώνουμε κατάλληλα κατά μία μονάδα την τρέχουσα y συνιστώσα και έπειτα, μεταβαίνουμε στο αριστερό άκρο της συνιστώσας αυτής

ώστε να αποφασίσουμε πάλι μη ντετερμινιστικά εάν θέλουμε επιπλέον μείωση ή όχι και κατά πόσον αυτή η επιθυμητή μείωση μπορεί να συντελεστεί. Αυτές οι λειτουργίες μοντελοποιούνται από την κατάσταση μ:

$$\mu \;\rightarrow\; \mu_{\nu\alpha\iota} \mid \mu_{o\chi\iota}$$

Η ερμηνεία του παραπάνω κανόνα είναι προφανής: επιλέγουμε μη ντετερμινιστικά εάν επιθυμούμε παραιτέρω μείωση της y συνιστώσας, οπότε και θα πρέπει να κάνουμε τον αναγκαίο έλεγχο για το εάν είναι δυνατή η μείωση, ή όχι:

$$\begin{aligned}
\mu_{\nu\alpha\iota} y &\rightarrow y\mu_{\nu\alpha\iota} \\
\mu_{\nu\alpha\iota} 0 &\rightarrow 0\mu_{\nu\alpha\iota} \\
\mu_{\nu\alpha\iota} 1 &\rightarrow 1\mu' \\
\mu_{\nu\alpha\iota}, &\rightarrow ,\mu''
\end{aligned}$$

Μόλις βρούμε άσσο, ξέρουμε ότι η μείωση είναι δυνατή οπότε είναι αναγκαίο να προχωρήσουμε στο δεξί άκρο της τρέχουσας y συνιστώσας και να αρχίσουμε τη μείωση:

$$\begin{aligned}
\mu' 0 &\rightarrow 0\mu' \\
\mu' 1 &\rightarrow 1\mu' \\
\mu', &\rightarrow \lambda,
\end{aligned}$$

Αντίθετα, εάν βρούμε ότι η τρέχουσα y συνιστώσα είναι μηδέν, τότε μεταβαίνουμε στην κατάσταση μ'' η οποία, δεδομένου ότι έχουμε ήδη μειώσει τουλάχιστον μία φορά τη συνιστώσα y θα αποφασίσει μη ντετερμινιστικά για τη συνιστώσα z εάν επιθυμεί αύξηση ή μείωση ή τίποτε από τα δύο. Και στις τρείς αυτές περιπτώσεις, έχουμε εξασφαλίσει ότι η τρέχουσα τριάδα που έχουμε κατασκευάσει είναι λεξικογραφικά μικρότερη της αρχικής τριάδος προτύποϋ. Επίσης, παρατηρούμε ότι η λειτουργία των καταστάσεων $\mu_{o\chi\iota}$ και μ'' είναι ακριβώς η ίδια: και στις δύο περιπτώσεις έχουμε μειώσει τουλάχιστον μία φορα την τρέχουσα y συνιστώσα και έχουμε αποφασίσει, για διαφορετικούς

λόγους, να μεταφερθούμε για αυξομείωση στη z συνιστώσα. Αρα, αρκεί να περιγράψουμε τη μια από αυτές τις δύο καταστάσεις:

$$\begin{aligned}
\mu_{o\chi\iota} y &\rightarrow y \mu_{o\chi\iota} \\
\mu_{o\chi\iota} 0 &\rightarrow 0 \mu_{o\chi\iota} \\
\mu_{o\chi\iota} 1 &\rightarrow 1 \mu_{o\chi\iota} \\
\mu_{o\chi\iota}, &\rightarrow , \mu_{o\chi\iota} \\
\mu_{o\chi\iota} z &\rightarrow \xi z
\end{aligned}$$

όπου η κατάσταση ξ θα αποφασίζει μη ντετερμινιστικά εάν θα μειώσει, θα αυξήσει ή θα αφίσει τη z συνιστώσα ως έχει.

Στα προηγούμενα βήματα περιγράψαμε τι γίνεται όταν η y συνιστώσα στο τρέχον βήμα δεν είναι μηδέν. Απομένει να εξηγήσουμε τι κάνουμε όταν βρούμε ότι αυτή η συνιστώσα είνα μηδέν, περίπτωση που μοντελοποιείται από την κατάσταση κ''. Στη περίπτωση αυτή, έχουμε ήδη εξηγήσει ότι η συνιστώσα z δεν μπορεί να είναι μηδέν, οπότε μεταφέρουμε το δρομέα στη συνιστώσα αυτή, στο δεξί της άκρο, και αρχίζουμε να τη μειώνουμε μη ντετερμινιστικά:

$$\begin{aligned}
\kappa'' z &\rightarrow z \kappa'' \\
\kappa'' 0 &\rightarrow 0 \kappa'' \\
\kappa'' 1 &\rightarrow 1 \kappa'' \\
\kappa'', &\rightarrow \lambda_z ,
\end{aligned}$$

Η κατάσταση λ_z, σε πλήρη αντιστοιχεία με την κατάσταση λ, θα μειώνει κατά μία μονάδα τον αριθμό που βρίσκεται στα αριστερά αυτής και στη συνέχεια, μη ντετερνινιστικά, θα αποφασίζει εάν θέλει νε μειώσει, εφ'όσον κάτι τέτοιο είναι δυνατό, τον ίδιο αριθμό επαναληπτικά:

$$\begin{aligned}
0 \lambda_z &\rightarrow \lambda_z 1 \\
1 \lambda_z &\rightarrow \lambda'_z 0 \\
0 \lambda'_z &\rightarrow \lambda'_z 0 \\
1 \lambda'_z &\rightarrow \lambda'_z 1 \\
z \lambda'_z &\rightarrow \mu_z z
\end{aligned}$$

Οπως και στη προηγούμενη περίπτωση, η κατάσταση μ_z θα αποφασίζει μη ντετερμινιστικά για το εάν θα μειώσει περεταίρω ή όχι τη συνιστώσα z:

$$\mu_z \rightarrow \mu_{z_{\nu\alpha\iota}} \mid \mu_{z_{o\chi\iota}}$$

Εαν μεταβούμε στην κατάσταση $\mu_{z_{\nu\alpha\iota}}$ τότε, πριν από οποιαδήποτε μείωση, είναι αναγκαίο να ελένξουμε έαν είναι δυνατή μια τέτοια επιθυμητή μείωση:

$$\begin{aligned}
\mu_{z_{\nu\alpha\iota}} z &\rightarrow z\mu_{z_{\nu\alpha\iota}} \\
\mu_{z_{\nu\alpha\iota}} 0 &\rightarrow 0\mu_{z_{\nu\alpha\iota}} \\
\mu_{z_{\nu\alpha\iota}} 1 &\rightarrow 1\mu'_z \\
\mu_{z_{\nu\alpha\iota}} \} &\rightarrow \omega\} \\
\mu_{z_{\nu\alpha\iota}} \}\} &\rightarrow \omega\}\}
\end{aligned}$$

Σε πλήρη αναλογία με τους αντίστοιχους κανόνες για τη περίπτωση της y συνιστώσας, μόλις βρούμε άσσο, ξέρουμε ότι η μείωση είναι δυνατή οπότε είναι αναγκαίο να προχωρήσουμε στο δεξί άκρο της τρέχουσας z συνιστώσας και να αρχίσουμε τη μείωση:

$$\begin{aligned}
\mu'_z 0 &\rightarrow 0\mu'_z \\
\mu'_z 1 &\rightarrow 1\mu'_z \\
\mu'_z, &\rightarrow \lambda_z,
\end{aligned}$$

Επιπλέον, η λειτουργία των καταστάσεων $\mu_{z_{o\chi\iota}}$ και μ''_z είναι ακριβώς η ίδια: και στις δύο περιπτώσεις έχουμε μειώσει τουλάχιστον μία φορά την τρέχουσα z συνιστώσα και έχουμε αποφασίσει, για διαφορετικούς λόγους, να μεταφερθούμε στην κατάσταση εκείνη η οποία θα μας δώσει τη δυνατότητα να συνεχίσουμε τη δημιουργία όλων των λεξικογραφικά μικρότερων τριάδων σε σχέση με την αρχική τριάδα-πρότυπο του τρέχοντος βήματος, μέχρις ότου αποφασίσαμε ότι έχουμε τελειώσει με την παραγωγή όλων των τριάδων αυτού του τύπου οπότε και θα μεταβούμε στην κατάσταση Ψ για να δημιουργήσουμε τις επιθυμητές λεξικογραφικά μεγαλύτερες τριάδες. (όπου ω στην παραπάνω και παρακάτω περίπτωση η κατάσταση εκείνη που θα μας φέρει στη σωστή θέση

ώστε να συνεχιστεί η παραγωγή όλων των λεξικογραφικά μικρότερων τριάδων που επιθυμούμε, κάνοντας απλά αλλαγή στο ποιά θα θεωρούμε τρέχουσα τριάδα στο τρέχον βήμα):

$$\begin{aligned}
\mu_{z_{o\chi\iota}} z &\rightarrow z \mu_{z_{o\chi\iota}} \\
\mu_{z_{o\chi\iota}} 0 &\rightarrow 0 \mu_{z_{o\chi\iota}} \\
\mu_{z_{o\chi\iota}} 1 &\rightarrow 1 \mu_{z_{o\chi\iota}} \\
\mu_{z_{o\chi\iota}} \} &\rightarrow \omega \} \\
\mu_{z_{o\chi\iota}} \} \} &\rightarrow \omega \} \}
\end{aligned}$$

Οπως ήδη είπαμε, η κατάσταση ω είναι εκείνη η κατάσταση η οποία θα μεταφέρει στην δρομέα στην αρχή της τρέχουσας τριάδος ώστε να συνεχιστεί η διαδικασία παραγωγής λεξικογραφικά μικρότερων τριάδων ακολουθώντας την παραπάνω διαδικασία:

$$\begin{aligned}
0\omega &\rightarrow \omega 0 \\
1\omega &\rightarrow \omega 1 \\
,\omega &\rightarrow \omega , \\
x\omega &\rightarrow \omega x \\
y\omega &\rightarrow \omega y \\
z\omega &\rightarrow \omega z \\
\{\omega &\rightarrow \{\Phi
\end{aligned}$$

Οπως φαίνεται από το παραπάνω σύνολο κανόνων, μόλις συναντήσουμε το αριστερό άγκυστρο, τότε ξαναμεταβαίνουμε στην κατάσταση Φ η οποιά θα αποφασίσει μη ντετερμινιστικά για τη συνέχεια της παραγωγής λεξικογραφικά μικρότερων τριάδων, εφαρμόζοντας όλη την παραπάνω διαδικασία στη τριάδα που δημιουργήσαμε στο προηγούμενο βήμα μας.

Τώρα, είναι απαραίτητο να περιγραφεί η λειτουργία της καταστάσεως ξ η οποία διαισθητικά είναι η εξής: έχουμε ήδη αποφασίσει σε προηγούμενο μη ντετερμινιστικό βήμα να μειώσουμε τη τρέχουσα y συνιστώσα, την μειώσαμε στο επιθυμητό μέτρο και τώρα είναι αναγκαίο να ασχοληθούμε με την z συνιστώσα. Αφού έχουμε ήδη μειώσει το y, όπως έχουμε ήδη πει, έχουμε τη δυνατότητα να αναθέσουμε στη τρέχουσα z συνιστώσα οποιαδήποτε τιμή σε

σχέση αυτή που είχε η αντίστοιχη z συνιστώσα στη τριάδα πρότυπο του τρέχοντος βήματος: την ίδια, μεγαλύτερη ή μικρότερη. Ολες αυτές οι λειτουργίες μοντελοπιούνται από τη κατάσταση ξ η οποία βρίσκεται και στη σωστή θέση: δεν χρειάζεται να μετακινήσουμε τον δρομέα, γιατί αυτός ήδη βρίσκεται στη σωστή θέση πρό του χαρακτήρος z.

Αρχικά, μέσω της καταστάσεως ξ πρέπει να αποφασίσουμε εάν θα αυξήσουμε, θα μειώσουμε ή εάν θα αφήσουμε ως έχει το τρέχον z στοιχείο:

$$\xi \rightarrow \xi_1 \mid \xi_2 \mid \xi_3$$

όπου ξ_1 είναι η κατάσταση που μεταβαίνουμε εάν θέλουμε να αφήσουμε αναλλοίωτο το στοιχείο z, ξ_2 η κατάσταση στην οποία μεταβαίνουμε εάν θάλουμε να μειώσουμε το τρέχον στοιχείο z και τέλος, ξ_3 η κατάσταση στην οποία μεταβαίνουμε εάν θάλουμε να αυξήσουμε το τρέχον στοιχείο z.

Αρχίζουμε την περιγραφή μας από την κατάσταση ξ_1:

$$\begin{aligned}
0\xi_1 &\rightarrow \xi_1 0\\
1\xi_1 &\rightarrow \xi_1 1\\
,\xi_1 &\rightarrow \xi_1 ,\\
x\xi_1 &\rightarrow \xi_1 x\\
y\xi_1 &\rightarrow \xi_1 y\\
\{\xi_1 &\rightarrow \{\Phi
\end{aligned}$$

Δηλαδή, αφού αποφασίσαμε να κρατήσουμε αναλλοίωτο το στοιχείο z, απλά μεταφέρουμε τον δρομέα προς τα αριστερά, στο αριστερό άκρο της τρέχουσας τριάδος, ώστε να συνεχιστεί η μη ντετερμινιστική διαδικασία παραγωγής λεξικογραφικά μικρότερων τριάδων.

Εάν στο προηγούμενο μη ντετερμινιστικό βήμα αποφασίσουμε να μεταβούμε στη κατάσταση ξ_2 τότε είναι αναγκαίο να μειωθεί η τρέχουσα z συνιστώσα, εφ'όσον βέβαια κάτι τέτοιο είναι δυνατό:

$$
\begin{aligned}
\xi_2 z &\rightarrow z\xi_2 \\
\xi_2 0 &\rightarrow 0\xi_2 \\
\xi_2 1 &\rightarrow 1\xi_2' \\
\xi_2 \} &\rightarrow \omega \} \\
\xi_2' \} &\rightarrow \lambda_z \}
\end{aligned}
$$

Δηλαδή, μεταβαίνουμε στο δεξί άκρο του τρέχοντος z στοιχείου ενώ παράλληλα ελέγχουμε το κατά πόσον αυτό είναι ή όχι μηδέν και μεταβαίνουμε στην αντίστοιχη κατάσταση ξ_2 ή ξ_2' αντίστοιχα. Στην πρώτη περίπτωση μεταβαίνουμε στην κατάσταση ω όπως και πριν για να συνεχιστεί η μη ντετερμινιστική διαδικασία παραγωγής λεξικογραφικά μικρότερων τριάδων ενώ στη δεύτερη περίπτωση, όταν δηλαδή βρούμε ότι ο αριθμός μπορεί να μειωθεί, τότε μεταβαίνουμε στην κατάσταση λ_z την οποία περιγράψαμε σε προηγούμενο βήμα της οποίας η λειτουργία είναι να μειώνει κατά μία μονάδα τον αριθμό που βρίσκεται στα αριστερά τής και στη συνέχεια, μη ντετερνινιστικά, να αποφασίζει εάν θέλει νε μειώσει, εφ'όσον κάτι τέτοιο είναι δυνατό, τον ίδιο αριθμό επαναληπτικά.

Η τρίτη και τελευταία περίπτωση αφορά την περιγραφή της κατάστασης ξ_3 στην οποία, όπως είπαμε, μεταβαίνουμε όταν αποφασίσουμε να αυξήσουμε μη ντετερμινιστικα, και εφόσον είναι βέβαια εφικτό, το τρέχον στοιχείο z.

Για να καταστεί αυτό εφικτό, είναι αναγκαίο να έχουμε μια κατάσταση η οποία και θα ελέγχει εάν είναι δυνατόν να αυξηθεί το τρέχον z στοιχείο. Αυτό θα γίνει συγκρίνοντας ένα προς ένα τα ψηφιά του τρέχοντος z στοιχείου με το μεγαλύτερο στοιχείο του συνόλου Z το οποίο και θα είναι το τελευταίο (δεξιότερο) στο αντίστοιχο σύνολο στοιχείων.

Αρχικά, είναι απαραίτητο να βρούμε και να σημαδέψουμε αυτό το στοιχείο, το μεγαλύτερο, του συνόλου Z. Για τον λόγο αυτό μετακινούμε το δρομέα στην αρχή της συμβολοσειράς (τέρμα αριστερά) και στη συνέχεια τον προχωράμε προς τα δεξιά μέχρι να "εισέλθουμε" στο σύνολο Z, και τότε απλά προχωράμε μέχρι να συναντήσουμε τον χαρακτήρα "}": ο αριθμός αμέσως αριστερά του δρομέα είναι ο αριθμός που ψάχνουμε. Επιπλέον, είναι αναγκαίο να σημαδέψουμε και τον πρώτο (z) χαρακτήρα του τρέχοντος αριθμού, ώστε να τον βρίσκουμε εύκολα όταν φα κάνουμε την σύγκριση με διαδοχικές παλινδρομήσεις του δρομέα:

$$
\begin{aligned}
\xi_3 z &\rightarrow \nu z' \\
0\nu &\rightarrow \nu 0 \\
1\nu &\rightarrow \nu 1 \\
x\nu &\rightarrow \nu x \\
y\nu &\rightarrow \nu y \\
z\nu &\rightarrow \nu z \\
,\nu &\rightarrow \nu, \\
\},\{\nu &\rightarrow \nu\},\{ \\
\{\{\nu &\rightarrow \{\{\pi
\end{aligned}
$$

Στην κατάσταση π έχουμε αντιληφθεί ότι βρισκόμαστε στην αρχή της συμβολοσειράς, οπότε τώρα είναι απαραίτητο να βρούμε και να σημαδέψουμε το τελευταίο (και άρα το μεγαλύτερο) στοιχείο του συνόλου Z:

$$
\begin{aligned}
\pi x &\rightarrow x\pi \\
\pi 0 &\rightarrow 0\pi \\
\pi 1 &\rightarrow 1\pi \\
\pi, &\rightarrow ,\pi \\
\pi\},\{ &\rightarrow \},\{\pi \\
\pi y &\rightarrow y\pi \\
\pi z &\rightarrow z\pi' \\
\pi' 0 &\rightarrow 0\pi' \\
\pi' 1 &\rightarrow 1\pi' \\
\pi'\} &\rightarrow \pi''\}
\end{aligned}
$$

Με παραπάνω σύνολο κανόνων, φέραμε το δρομέα στο τέλος του συνόλου Z και τώρα είναι απαραίτητο να σημαδέψουμε το τελευταίο του στοιχείο το οποίο, όπως είπαμε, είναι και το μεγαλύτερο στοιχείο, αυτό δηλαδή με το οποίο είναι αναγκαίο να συγκρίνουμε το στοιχείο z που θέλουμε να αυξήσουμε στο τρέχον μας βήμα. Τώρα απλά το σημαδεύουμε, σημαδεύοντας απλώς τον χαρακτήρα z στο αριστερό άκρο αυτού του αριθμού:

$$
\begin{aligned}
0\pi'' &\rightarrow \pi''0 \\
1\pi'' &\rightarrow \pi''1 \\
z\pi'' &\rightarrow z'\varpi
\end{aligned}
$$

Τώρα είμαστε έτοιμοι να επιστρέψουμε στο στοιχείο z στε να αρχίσει η διαδικασία της σύγκρισης:

$$
\begin{aligned}
\varpi x &\rightarrow x\varpi \\
\varpi 0 &\rightarrow 0\varpi \\
\varpi 1 &\rightarrow 1\varpi \\
\varpi, &\rightarrow ,\varpi \\
\varpi\},\{ &\rightarrow \},\{\varpi \\
\varpi y &\rightarrow y\varpi \\
\varpi z &\rightarrow z\varpi \\
\varpi z' &\rightarrow z'\rho
\end{aligned}
$$

Στην κατάσταση ρ προχωράμε μέχρι να βρούμε τον πρώτο μη σημαδεμένο χαρακτήρα του τρέχοντος στοιχείου z που θέλουμε να αυξήσουμε:

$$
\begin{aligned}
\rho 0' &\rightarrow 0'\rho \\
\rho 1' &\rightarrow 1'\rho \\
\rho 0 &\rightarrow \rho_0 0' \\
\rho 1 &\rightarrow \rho_1 1' \\
\rho\} &\rightarrow \sigma\}
\end{aligned}
$$

Δηλαδή, αγνοούμε όλους τους σημαδεμένους χαρακτήρες (τους οποίους έχουμε συγκρίνει σε προηγούμενα βήματα) και μόλις βρούμε τον πρώτο μη σημαδεμένο τον σημαδέβουμε και τον αποθηκεύουμε σε μια νέα, αντίστοιχη του αριθμού, κατάσταση και αρχίζουμε να κάνουμε τη σύγκριση με το αντίστοιχο ψηφίο του μεγαλύτερου αριθμού του συνόλου Z. Αντίθετα, εάν δε βρούμε μη σημαδεμένο χαρακτήρα και συναντήσουμε το δεξί άγκυστρο, τότε

ξάρουμε ότι οι δύο αριθμοί είναι ίσοι και άρα δεν μπορούμε να κάνουμε επιπλέον αύξηση. Πράγματι, αυτό σημαίνει ότι έοχυμε συγκρίνει όλα τα ψηφία και τα έχουμε βρεί ίδια. Αρα οι δύο αριθμοί είναι ίσοι και επειδή ο ένας είναι ο μέγιστος, άρα και ο δεύτερος έχει τη μέγιστη δυνατή τιμή και δεν μπορεί να αυξηθεί. Τότε μεταβαίνουμε στην κατάσλληλη κατάσταση σ.

Τώρα είναι αναγκαίο να περιγράψουμε τον έλεγχο των δύο ψηφίων. Αυτό που κάνουμε είναι να φέρουμε το δρομέα στο πρώτο προς τα αριστερά σημαδεμένο χαρακτήρα και εν ελέγξοουμε τον δεξιό του χαρακτήρα με αυτόν που έοχυμε αποθηκευμένο στην κατάστασή μας. Περιγράφουμε μόνο την περίπτωση της καταστάσεως ρ_0. Η περίπτωση της ρ_1 είναι τελείως ανάλογη:

$$
\begin{aligned}
0\rho_0 &\rightarrow \rho_0 0\\
1\rho_0 &\rightarrow \rho_0 1\\
x\rho_0 &\rightarrow \rho_0 x\\
y\rho_0 &\rightarrow \rho_0 y\\
z\rho_0 &\rightarrow \rho_0 z\\
,\rho_0 &\rightarrow \rho_0 ,\\
\},\{\rho_0 &\rightarrow \rho_0\},\{\\
0'\rho_0 &\rightarrow 0'\rho'_0\\
1'\rho_0 &\rightarrow 1'\rho'_0\\
z'\rho_0 &\rightarrow z'\rho'_0\\
\rho'_0 0 &\rightarrow 0'\varpi\\
\rho'_0 1 &\rightarrow z'\varrho
\end{aligned}
$$

Αν στον παραπάνω έλεγχο βρούμε τα δύο ψηφία ίδια, τότε πρέπει να γυρίσουμε να επαναλάβουμε τον έλεγχο για τα επόμενα δύο ψηφία. Αλλίως, άμα δηλαδή βρούμε ότι τα δύο ψηφία διαφέρουν, τότε ξέρουμε ότι το στοιχείο z που θέλουμε να αυξήσουμε επιδέχεται αύξησης και μεταβαίνουμε στην κατάσταση ϱ της οποίας η λειτουργία είναι να μας μεταφέρει στο σωστό σημείο, να ξεσημαδέψει και να κάνει την αύξηση, αφού πρώτα ξεσημαδέψει το τρέχον μέγιστο στοιχείο:

$$
\begin{aligned}
0'\varrho &\rightarrow \varrho 0\\
1'\varrho &\rightarrow \varrho 1\\
z'\varrho &\rightarrow z\tau
\end{aligned}
$$

Τώρα, στην κατάσταση τ προχωράμε τον δρομέα προς το τρέχον z στοιχείο ώστε να το ξεσημαδέψουμε και να το αυξήσουμε κατά μία μονάδα και ύστερα να επαναλάβουμε την παραπάνω επαναληπτική διαδικασία:

$$\begin{aligned}
\tau x &\rightarrow x\tau \\
\tau 0 &\rightarrow 0\tau \\
\tau 1 &\rightarrow 1\tau \\
\tau, &\rightarrow ,\tau \\
\tau\},\{ &\rightarrow \},\{\tau \\
\tau y &\rightarrow y\tau \\
\tau z &\rightarrow z\tau \\
\tau z' &\rightarrow z\tau' \\
\tau 0' &\rightarrow 0\tau' \\
\tau 1' &\rightarrow 1\tau' \\
\tau\} &\rightarrow \upsilon\}
\end{aligned}$$

Η λειτουργία της καταστάσεως υ είναι να αυξήσει τον αριθμό-στοιχείο ξατά μία μονάδα:

$$\begin{aligned}
1\upsilon &\rightarrow \upsilon 0 \\
0\upsilon &\rightarrow \upsilon' 1 \\
1\upsilon' &\rightarrow \upsilon' 0 \\
0\upsilon' &\rightarrow \upsilon' 1 \\
z\upsilon' &\rightarrow \xi' z
\end{aligned}$$

Τώρα, μπορούμε είτε να αποφασίσουμε να αυξήσουμε ξανά το τρέχον στοιχείο, και άρα μεταβαίνουμε στην κατάσταση ξ_3 είτε να αποφασίσουμε ότι τελειώσαμε με την αύξηση και αρα τελειώσαμε με την παραγωγή της τρέχουσας τριάδος (κατάσταση ς):

$$
\begin{aligned}
\xi' &\rightarrow \xi_3 \mid \varsigma \\
x\varsigma &\rightarrow \varsigma x \\
y\varsigma &\rightarrow \varsigma y \\
0\varsigma &\rightarrow \varsigma 0 \\
1\varsigma &\rightarrow \varsigma 1 \\
,\varsigma &\rightarrow \varsigma, \\
\{\varsigma &\rightarrow \Phi\{
\end{aligned}
$$

Αυτό που απομένει ώστε να ολοκληρωθεί το παραπάνω σύνολο κανόνων είναι η εξήγηση της καταστάσεως σ. Οταν βρεθούμε σε αυτή τη περίπτωση ξέρουμε ότι δεν μπορούμε να κάνουμε επιπλέον αυξήσεις στο τρέχον z στοιχείο και άρα απλά έχουμε τελειώσει με τη τρέχουσα τριάδα και συνεχιζουμε την αρχική μας διαδικασία:

$$
\begin{aligned}
1'\sigma &\rightarrow \sigma 1 \\
0'\sigma &\rightarrow \sigma 0 \\
z'\sigma &\rightarrow \sigma z \\
,\sigma &\rightarrow \sigma, \\
y\sigma &\rightarrow \sigma y \\
x\sigma &\rightarrow \sigma x \\
1\sigma &\rightarrow \sigma 1 \\
0\sigma &\rightarrow \sigma 0 \\
\{\sigma &\rightarrow \sigma'\tilde{\{}
\end{aligned}
$$

Στην κατάσταση σ' θέλουμε να ξεσημαδέψουμε το ήδη σημαδεμένο μέγιστο στοιχείο του συνόλου Z:

$$
\begin{aligned}
y\sigma' &\rightarrow \sigma' y \\
x\sigma' &\rightarrow \sigma' x \\
1\sigma' &\rightarrow \sigma' 1 \\
0\sigma' &\rightarrow \sigma' 0 \\
,\sigma' &\rightarrow \sigma', \\
\},\{\sigma' &\rightarrow \},\{\sigma' \\
1'\sigma' &\rightarrow \sigma' 1 \\
0'\sigma' &\rightarrow \sigma' 0 \\
z'\sigma' &\rightarrow \zeta\phi
\end{aligned}
$$

Και τώρα μεταφέρουμε το δρομέα στη τριάδα που μόλις δημιουργήσαμε ώστε να συνεχίσουμε τη διαδικασία (την τριάδα αυτή την αναγνωρίζουμε αφού τη σημαδέψαμε στο προηγούμενο σύνολο κανόνων ώστε να την βρούμε εύκολα):

$$
\begin{aligned}
\phi x &\rightarrow x\phi \\
\phi 0 &\rightarrow 0\phi \\
\phi 1 &\rightarrow 1\phi \\
\phi, &\rightarrow ,\phi \\
\phi y &\rightarrow y\phi \\
\phi z &\rightarrow z\phi \\
\phi\},\{ &\rightarrow \},\{\phi \\
\phi\tilde{\{} &\rightarrow \Phi\{
\end{aligned}
$$

Ολα τα παραπάνω είναι αποτέλεσμα της αρχικής μη ντετερμινιστικής απόφασης να μειωθεί αρχικά η y συνιστώσα της τρέχουσας τριάδος, εφόσον βέβαια κάτι τέτοιο είναι εφικτό, και έπειτα να αλλάξουμε κατάλληλα τη z συνιστώσα, ανάλογα με το κατά πόσον ήταν εφικτή η μείωση της y συνιστώσας.

Ομως, σε εκείνο το μη ντετερμνιστικό βήμα, το οποίο το μοντελοποιήσαμε με την κατάσταση κ, είναι πιθανό να επιλέξουμε να αφήσουμε αναλλοίωτη τη y συνιστώσα της τρέχουσας τριάδος και να προσπαθήσουμε, εφόσον πάντα κάτι τέτοιο είναι δυνατό, να μειώσουμε τη z συνιστώσα που έπειται. Σε αυτή τη

περίπτωση μεταβαίνουμε στην κατάσταση $\kappa_{o\chi\iota}$ της οποίας η λειτουργία είναι ακριβώς αυτή: να ελέγξει το κατά πόσον είναι δυνατή η μείωση της z συνιστώσας και έπειτα, εφόσον αυτή είναι δυνατή, να τη μειώση κατά βούληση, ενώ, από την άλλη μεριά, εάν κάτι τέτοιο δεν είναι εφικτό να οπισθοδρομίσει, αναιρώντας την προηγούμενη μη ντετερμινιστική επιλογή να αφήσει αναλλοίωτη τη y συνιστώσα και να αρχίσει να μειώνει αυτή. Οπως έχουμε εξηγήσει επανηλημένα, στην προηγούμεη περίπτωση η y συνιστώσα δεν είναι δυνατόν να είναι μηδέν, οπότε μία τουλάχιστον μείωση στη συνιστώσα αυτή είναι πάντοτε εφικτή.

Για να ολοκληρωθεί λοιπόν η παρουσίαση της τρέχουσας ενότητας, είναι απαραίτητο να περιγραφεί η λειτουργία της καταστάσεως $\kappa_{o\chi\iota}$. Οπως είπαμε, η λειτουργία αυτής της καταστάσεως είναι να δημιουργεί τριάδες της μορφής $\{x_{p_i}, y_{q_i}, z_{r_i}\}$ από την αρχική τριάδα πρότυπο $\{x_{j_i}, y_{k_i}, z_{l_i}\}$ τέτοιες ώστε να ισχύει $x_{p_i} \equiv x_{j_i}$, $y_{q_i} \equiv y_{k_i}$ και $z_{r_i} \prec z_{l_i}$, δηλαδή τριάδες λεξικογράφικά μικρότερες από την αρχική τριάδα πρότυπο που όμως διατηρούν αναλλοίωτες τις δύο συνιστώσες x και y, εφόσον βέβαια κάτι τέτοιο είναι εφικτό.

Ο δρομέας μας τη δεδομένη χρονική στιγμή βρίσκεται προ του χαρακτήρα y της αντίστοιχης συνιστώσας. Αρα, αρχικά είναι αναγκαίο να μεταφέρουμε το δρομέα στη z συνιστώσα και, παράλληλαν να κάνουμε τον έλεγχο για το αν η τρέχουσα z συνιστώσα είναι δυνατό να μειωθεί:

$$
\begin{array}{rcl}
\kappa_{o\chi\iota}y & \rightarrow & y\kappa_{o\chi\iota} \\
\kappa_{o\chi\iota}0 & \rightarrow & 0\kappa_{o\chi\iota} \\
\kappa_{o\chi\iota}1 & \rightarrow & 1\kappa_{o\chi\iota} \\
\kappa_{o\chi\iota}, & \rightarrow & ,\kappa'_{o\chi\iota} \\
\kappa'_{o\chi\iota}z & \rightarrow & z\kappa'_{o\chi\iota} \\
\kappa'_{o\chi\iota}0 & \rightarrow & 0\kappa'_{o\chi\iota} \\
\kappa'_{o\chi\iota}1 & \rightarrow & 1\kappa''_{o\chi\iota} \\
\kappa'_{o\chi\iota}\} & \rightarrow & \varphi\}
\end{array}
$$

Δηλαδή, προχωράμε προς τα δεξιά ελέγχοντας ταυτόχρονα τη τρέχουσα z συνιστώσα. Μολις συναντίσουμε κάποιον άσσο, μεταβαίνουμε στη κατάσταση $\kappa''_{o\chi\iota}$ λειτουργία της οποίας είναι να μας φέρει στο τέλος (δεξί άκρο) της τρέχουσας z συνιστώσας ώστε να αρχίσει η διαδικασία μείωσης αυτής:

$$
\begin{aligned}
\kappa''_{o\chi\iota}0 &\rightarrow 0\kappa_{o\chi\iota} \\
\kappa''_{o\chi\iota}1 &\rightarrow 1\kappa_{o\chi\iota} \\
\kappa''_{o\chi\iota}\} &\rightarrow \lambda_z\}
\end{aligned}
$$

Αντίθετα, εάν δεν βρούμε άσσο, πράγμα που σημαίνει ότι η τρέχουσα z συνιστώσα είναι μηδέν, τότε μεταβαίνουμε στην κατάσταση φ λειτουργία της οποίας είναι να μας φέρει στο *τέλος* της y συνιστώσας ώστε να αρχίσει η μείωση αυτής:

$$
\begin{aligned}
0\varphi &\rightarrow \varphi 0 \\
1\varphi &\rightarrow \varphi 1 \\
z\varphi &\rightarrow \varphi z \\
,\varphi &\rightarrow \lambda,
\end{aligned}
$$

Οπως παρατηρούμε, και οι δύο διαδικασίες μείωσης των συνιστωσών z ή y, γίνωνται μέσω των καταστάσεων λ_z και λ αντίστοιχα, καταστάσεις που έχουμε ήδη περιγράψει σε προηγούμενα βήματα.

Κεφαλαιο 9

Γραμματικη Με Συμφραζομενα για το Προβλημα της Ικανοποιησιμοτητας μιας Λογικης Εκφρασης

9.1 Εισαγωγή

Στο τρέχον κεφάλαιο θα παρουσιάσουμε σύνολα κανόνων τα οποία θα μας δίνουν τη δυνατότητα να κατασκευάσουμε μια οποιαδήποτε ικανοποιήσιμη λογική έκφραση ϕ σε *συζευκτική κανονική μορφή*. Με άλλα λόγια, τα σύνολα κανόνων που θα περιγράψουμε θα παράγουν λογικές εκφράσεις ϕ για τις οποίες να υπάρχει πάντα κάποια, όχι απαραίτητα μοναδική, ανάθεση αληθοτιμών T στις μεταβλητές των τέτοια ώστε $T \models \phi$.

Πριν αρχίσουμε την παράθεση εκείνων των κανόνων οι οποίοι θα μας οδηγήσουν στο επιθυμητό αποτέλεσμα, είναι απαραίτητο να περιγράψουμε τον τρόπο με τον οποίο θα αναπαριστούμε τα στιγμιότυπά μας τα οποία θα παράγει η γραμματική μας ως μια συμβολοσειρά. Γενικά, η μορφή του τυχαίου

στιγμιοτύπου θα είναι η εξής: στην αρχή (αριστερό άκρο) θα παραθέτουμε τις μεταβλητές οι οποίες θα είναι υπάρχουσες στην τρέχουσα λογική έκφραση. Ο τρόπος αναπαράστασης των μεταβλητών αυτών θα είναι το σύμβολο x ακολουθούμενο από την δυαδική αναπαράσταση του αύξοντος αριθμού της εκάστοτε μεταβλητής (μειωμένου κατά μία μονάδα). Δηλαδή, για να γίνει κατανοητό, εάν έχουμε πέντε μεταβλητές, αυτές θα έχουν τη μορφή $x000$, $x001$, $x010$, $x011$ και $x100$.

Επειτα από την παράθεση όλων των αναγκαίων και επιθυμητών μεταβλητών, οι οποίες θα εμφανίζονται είτε αυτούσιες είτε οι αρνήσεις αυτών στην υπό παραγωγή λογική έκφραση, θα ακολουθούν τα clauses της λογικής έκφρασης. Χωρίς βλάβη της γενικότητας, καθώς θα αποδείξουμε παρακάτω την ισοδυναμία των δύο διαφορετικών μορφών των λογικών εκφράσεων αυτών, θα επιτρέπουμε στα clauses αυτά να περιέχουν *ακριβώς* τρία literals - μεταβλητές ή αρνήσεις αυτών.

Ο τρόπος αναπαράστασης των clauses θα γίνει κατανοητός μέσα από ένα παράδειγμα: ένα clause της μορφής $(x_2 \vee x_5 \vee \neg x_0)$ θα αναπαρίσταται από την υποσυμβολοσειρά ως $(x010, x101, \bar{x}000)$.

Αφού περιγράψαμε, σε αδρές γραμμές, τον τρόπο αναπαράστασης των στιγμιοτύπων που θα παράγουν οι κανόνες μας, είναι απαραίτητο να περιγράψουμε την διαδικασία εκείνη, σε φυσική γλώσσα, η οποία θα παράγει συμβολοσειρές οι οποίες θα αντιστοιχούν σε ικανοποιήσιμες λογικές εκφράσεις:

- ▷ Αρχικά, θα κατασκευάσουμε την πρώτη μεταβλητή του τρέχοντος στιγμιοτύπου μας στη μορφή που περιγράψαμε προηγουμένως. Οι υπόλοιπες επιθυμητές μεταβλητές θα παραχθούν αργότερα μέσω του αντίστοιχου μη ντετερμινιστικού βήματος παραγωγής νέων μεταβλητών.

- ▷ Μόλις τελειώσει η παραγωγή της πρώτης μεταβλητής, θα αρχίσει η παραγωγή του συνόλου των clauses τα οποία αντιστοιχούν στην πρώτη αυτή μεταβλητή μας και τα οποία θα κατασκευαστούν με τέτοιο τρόπο ούτως ώστε η τελική συνολική έκφραση να είναι ικανοποιήσιμη, να υπάρχει δηλαδή κάποια, τουλάχιστον μια, ανάθεση αληθοτιμών που να ικανοποιεί την τρέχουσα έκφραση. με άλλα λόγια, φροντίζουμε με κάποιον τρόπο ο οποίος θα γίνει αντιληπτός στη συνέχεια, όλα τα clauses είτε να είναι ικανοποιήσιμα είτε ότι θα ικανοποιηθούν σε επόμενο βήμα. Τα υπόλοιπα clauses θα κατασκευαστούν αφού παραχθούν οι αντίστοιχες μεταβλητές.

- ▷ Ο τρόπος παραγωγής των clauses ώστε να έχουν την επιθυμητή ιδιότητα, δηλαδή της ικανοποιησιμότητας, είναι ο εξής:

  - ○ Στο i-οστό βήμα θεωρούμε την μεταβλητή x_i. Αφού αποφασίσουμε για την ανάθεση της αληθοτιμής της μεταβλητής αυτής, για την

αληθοτιμή δηλαδή η οποία αποτελεί μέρος της ανάθεσης εκε'ινης με την οποία η συνολική έκφραση θα έχει αληθή τιμή, αρχίζουμε να κατασκευάζουμε όλα τα επιθυμητά clauses τα οποία θα περιέχουν την τρέχουσα μεταβλητή.

- Σε πρώτη φάση, κατασκευάζουμε εκείνα τα clasues τα οποία ικανοποιούνται από το αληθές literal της μεταβλητής. Δηλαδή, εάν αποφασίσουμε η μεταβλητή να είναι αληθής, τότε κατασκευάζουμε όσα clauses θέλουμε τα οποία ικανοποιούνται από την x_i αλλιώς κατασκευάζουμε clauses τα οποία ικανοποιούνται από την άρνηση αυτής ($\bar{x}_i$). Τα clauses αυτά έχουν την ιδιότητα ότι είναι αληθή εν τη γενέση τους. Για αποφυγή συγχύσεων με τα clauses τα οποία *δεν* έχουν ικανοποιηθεί ακόμα, τα clauses αυτά θα έχουν τη μορφή $[x_i, \sqcup, \sqcup]$ (ή $[\bar{x}_i, \sqcup, \sqcup]$). Χρησιμοποιούμε τον χαρακτήρα $\sqcup$ για να υποδηλώσουμε ότι οι αντίστοιχες θέσεις είναι κενές και μπορούν να συμπληρωθούν αργότερα από οποιοδήποτε literal, αληθές η ψευδές.
- Επειτα, κατασκευάζουμε τα clauses εκείνα τα οποία απλά θα περιέχουν το ψευδές literal της τρέχουσας μεταβλητής (π.χ εάν αποφασίζουμε στο προηγούμενο μη ντετερμινιστικό βήμα ότι $x_i =$ αληθής τότε στο τρέχον βήμα κατασκευάζουμε τα clauses εκείνα τα οποία θα περιέχουν το literal $\bar{x}_i$). Προφανώς, αυτά τα clauses δεν είναι ικανοποιήσιμα στο τρέχον βήμα οπότε τα σημαδέυουμε ανάλογα για αποφυγή συγχήσεων με σκοπό να τα ικανοποιήσουμε σε επόμενο βήμα με κάποιο αληθές literal. Ο συμβολισμός που χρησιμοποιούμε για να διακρίνουμε τη διαφορετικότητά τους (από τα clauses του προηγουμένου βήματος) είναι ο $\{\bar{x}_i, \sqcup, \sqcup\}$ (ή $\{x_i, \sqcup, \sqcup\}$). Μολις συμπληρώσουμε μία εκ των κενών θέσεων με κάποιο αληθές iteral, τότε αυτόματα μετατρέπουμε τον συμβολισμό ώστε να ξέρουμε ότι το τρέχον clause έγινε πλέον ικανοποιήσιμο (μετατρέποντας ουσιαστικά τα [(]) σε { (}) αντίστοιχα).

- Αφού τελειώσει η παραγωγή όλων των επιθυμητών clauses που περιέχουν τη μεταβλητή x_i τότε:
  - Αρχικά θεωρούμε το αληθές literal της τρέχουσας μεταβλητής. Αρχίζουμε να το τοποθετούμε μη ντετερμινιστικά σε όσα clauses δούμε ότι έχουν "χώρο" (τον χαρακτήρα $\sqcup$), προσέχοντας την περίπτωση που αυτά τα clauses είναι *μη* ικανοποιήσιμα μέχρι στιγμής. Στη περίπτωση αυτή, αλλάζουμε όπως αναφέραμε παραπάνω τους διακριτικούς χαρακτήρες του τρέχοντος

clause. Αν τοποθετήσουμε το τρέχον literal σε ήδη ικανοποιημένο clause, ότε το αφήνουμε αυτό αναλλοίωτο.

- Αντίστοιχα προχωράμε και με τη θεώρηση του ψευδούς literal της τρέχουσας μεταβλητής με μιά διαφορά: δεν τοποθετούμε το τρέχον ψευδές literal ως τρίτη συνηστώσα σε κάποιο clause το οποίο δεν έχει μέχρι τώρα ικανοποιηθεί (αλλιώς θα είχαμε ένα clause το οποίο προφανώς και δεν ικανοποιήται από την επιθυμητή ανάθεση αληθοτιμών).
- Γενικά, οποιοδήποτε literal και αν θεωρήσουμε στο προηγούμενο βήμα, είναι αναγκαίο να προσέξουμε την παρακάτω περίπτωση: εάν η τοποθέτηση του τρέχοντος literal (αληθούς ή ψευδούς, δεν έχει σημασία) γίνει στη *δεύτερη* συνιστώσα του clause, τότε καμία επιπλέον ενέργεια δεν πρέπει να ακολουθήσει. Αντίθετα όμως, όταν πάμε να τοποθετήσουμε το τρέχον literal στην *τρίτη* συνιστώσα κάποιου clause (το αληθές literal σε οποιουδήποτε τύπου clause, ενώ το ψευδές litreal μόνο στα ήδη ικανοποιημένα clauses), προφανώς επειδή οι δύο προηγούμενες συνιστώσες έχουν καλυφθεί σε προηγούμενα βήματα, είναι απαραίτητο να ελεγχθεί το κατά πόσον έχει ήδη δημιουργηθεί το clause που πάμε να φτάξουμε στο τρέχον βήμα. Αυτό θα γίνει με τον εξής τρόπο:
  Το τρέχον clause την τρίτη συνιστώσα του οποίου θέλουμε να συμπληρώσουμε θα ανήκει σε έναν από τους δύο τύπους: είτε θα είναι ήδη ικανοποιημένο (και θα περικλείεται από τους χαρακτήρες { και }) είτε όχι (οπότε και θα το περικλείουν οι χαρακτήρες [και]). Σε κάθε περίπτωση αυτό που κάνουμε είναι να αντιγράψουμε το τρέχον literal στη θέση αυτή (της τρίτης συνιστώσας δηλαδή) και έπειτα να μεταβούμε σε μία νέα κατάσταση ρόλος της οποίας είναι να ελέγξει χαρακτήρα προς χαρακτήρα *όλα* τα ήδη δημιουργημένα clauses του *ιδίου* τύπου. Εάν όλα τα clauses αυτά είναι διαφορετικά από αυτό που θέλουμε να το δημιουργήσουμε, τότε μπορούμε άφοβα να τα δημιουργήσουμε αφού ξέρουμε ότι είναι ίδιο με κανένα άλλο. Εάν όμως βρούμε ότι το προς δημιουργία clause χει ήδη κατασκευαστεί σε προηγούμενο βήμα τότε αυτό που κάνουμε είναι το εξής: απλά σβήνουμε όλους τους $\lceil \log N \rceil + 1$ χαρακτήρες [1] του αριθμού αυτού από την τρίτη συνιστώσα του clause που θέλουμε να δημιουργήσουμε και τους αντικαθιστούμε όλους με τον χαρακτήρα $\sqcup$. Ετσι, το τρέχον clause στην τρίτη συ-

[1] όπου N είναι ο αύξων αριθμός της τρέχουσας μεταβλητής

νιστώσα του θε έχει $\lceil \log N \rceil + 1$ χαρακτήρες κενού, πράγμα που πρέπει να το έχουμε υπ'όψιν μας στα αρχικά στάδια κατασκευής των κανόνων.

▷ Αφού τελειώσουμε με την παραγωγή όλων των επιθυμητών clauses για την τρέχουσα μεταβλητή μας, και αφού τοποθετήσουμε τα δύο αντίστοιχα literals στις επιθυμητές θέσεις άλλων clauses, μπορούμε να συνεχίσουμε την μη ντετερμινιστική παραγωγή μεταβλητών και clauses. Η συνθήκη τερματισμού αυτής της μη ντετεμινιστικής διαδικασίας είναι η εξής: μόλις έχουμε τελείωσει με όλες τια απαραίτητες "διεργασίες" σχετικά με την τρέχουσα μεταβλητή, τότε σαρώνουμε όλη τη συμβολοσειρά του στιγμιοτύπου και βλέπουμε εάν υπάρχουν clauses με κενές θέσεις. Εάν όχι, τότε μπορούμε μη ντετερμινιστικά να αποφασίσουμε για την παραγωγή μιας νέας μεταβλητής ή να αποφασίσουμε ότι τελείωσαμε με την υπό κατασκευή πρότασή μας. Εάν πάλι βρούμε κενές θέσεις, τότε *ντετερμινιστικά* πλέον κατασκευάζουμε μια νέα μεταβλητή και ακολουθούμε τα παραπάνω βήματα.

9.2 Αρχικοποίηση του στιγμιοτύπου

Το τρέχον βήμα θα παράξουμε *ντετερμινιστικά* την πρώτη μεταβλητή του στιγμιοτύπου μας και έπειτα *μη ντετερμινιστικά* τα δύο ειδών clauses που αντιστοιχούν στο ψευδές και αληθές literal της τρέχουσας μεταβλητής, όπως περιγράψαμε παραπάνω. Σε επόμενο βήμα θα παράξουμε μη ντετερμινιστικά τις υπόλοιπες επιθυμητές μεταβλητές και clauses που επιθυμούμε.

Οπως είπαμε, τις μεταβλητές θα τις κατασκευάζουμε μία-μία σε *αύξουσα* διάταξη (ως προς τους αύξοντες αριθμούς αυτών). Αυτό σημαίνει ότι δεν είναι γνωστό εκ των πρωτέρων το πόσα ψηφία θα χρειαστούμε για την αναπαράσταση της τελευταίας μεταβλητής. Παρ'όλα αυτά, η αύξηση θα γίνει με το συνήθη τρόπο, προσέχοντας όμως να αυξάνουμε κατά ένα τα ψηφία της αναπαράστασης του τρέχοντος αριθμού μεταβλητής, όποτε κάτι τέτοιο καταστεί αναγκαίο.

Αρχικά, παράγουμε την πρώτη μεταβλητή και, ταυτόχρονα, αρχικοποιούμε το στιγμιότυπό μας (σημαδεύουμε τον τελευταίο χαρακτήρα της συμβολοσειράς, δηλαδή το δεξιότερο άγκυστρο, καθώς και το δεξιότερο άγκυστρο της υποσυμβολοσειράς του συνόλου των μεταβλητών ώστε να τους διακρίνουμε όταν σαρώνουμε σε μελλοντικά βήματα τη συμβολοσειρά μας):

$$A \;\rightarrow\; \{\{Bx_0\}'\bar{\}}$$

Αυτό που κάνουμε είναι το εξής: γράφουμε στο σύνολο που θα περιέχει τις μεταβλητές μας την πρώτη μεταβλητή μας, την $x0$. Στην κατάσταση B θα αναθέσουμε αληθοτιμή στην μεταβλητή αυτή (γενικά η κατάσταση B θα αναθέτει αληθοτιμή την μεταβλητή που βρίσκεται στα *δεξιά* αυτής):

$$B \rightarrow B_{true} \mid B_{false}$$

Με κάποιον τρόπο θα πρέπει να "θυμόμαστε" τι αληθοτιμή έχουμε δώσει στην τρέχουσα μεταβλητή μας. Ο τρόπος που το επιτυγχάνουμε αυτό είναι ο εξής: εάν αποφασίσουμε να δώσουμε την αληθοτιμή **true** στη μεταβλητή μας, τότε την αφήνουμε ως έχει ενώ, αντίθετα, εάν αποφασίσουμε να δώσουμε την αληθοτιμή **false** τότε σημαδεύουμε ανάλογα την μεταβλητή μας στον πρώτο (x) χαρακτήρα αυτής. Με τον τρόπο αυτό ξέρουμε ότι σε σημαδεμένες μεταβλητές αντιστοιχεί αληθοτιμή **false** (δηλαδή το αρνητικό literal της τρέχουσας μεταβλητής είναι αληθές):

$$\begin{aligned} B_{true} &\rightarrow C \\ B_{false}x &\rightarrow C\bar{x} \end{aligned}$$

Αφού τελειώσουμε με την επιθυμητή ανάθεση αληθοτιμής στην τρέχουσα μεταβλητή μας, είναι απαραίτητο να αρχίσουμε να κατασκευάζουμε μη ντετερμινιστικά τα δύο ειδών clauses που αντιστοιχούν στην τρέχουσα μεταβλητή μας. Η περίπτωση της πρώτης μεταβλητής είναι ελαφρά διαφορετική από την αντίστοιχη περίπτωση των υπολοίπων μεταβλητών, όποτε και θα την παρουσιάσουμε ξεχωριστά.

Αρχικά, φέρνουμε τον δρομέα στην κατάλληλη θέση (δηλαδή στο τέλος του συνόλου των μεταβλητών):

$$\begin{aligned} Cx0\}' &\rightarrow x0\}'D_1 \\ C\bar{x}0\}' &\rightarrow C\bar{x}\}'D_1 \end{aligned}$$

Στην κατάσταση D_1 αποφασίζουμε μη ντετερμινιστικά εάν θέλουμε clauses τα οποία να ικανοποιούνται από το θετικό literal της *πρώτης* μεταβλητής:

$$D_1 \;\rightarrow\; D_{1_{\nu\alpha\iota}} \mid D_{1_{o\chi\iota}}$$

Προφανώς, στην κατάσταση $D_{1_{\nu\alpha\iota}}$ αποφασίσαμε ότι θέλουμε clause το οποίο να ικανοποιείται από την τρέχουσα (πρώτη) μεταβλητή και είναι αναγκαίο να το κατασκευάσουμε, σε αντίθεση με την κατάσταση $D_{1_{o\chi\iota}}$ όπου δεν κατασκευάζουμε κανένα clause τέτοιου τύπου και προχωράμε στην μη ντετερμινιστική κατασκευή των clauses του δευτέρου τύπου, κατάσταση στην οποία θα προχωρήσουμε και μόλις τελειώσει η μη ντετερμινιστική κατασκευή των clauses του πρώτου τύπου. Αρχίζουμε την περιγραφή από την κατάσταση $D_{1_{\nu\alpha\iota}}$ όπου πρέπει να κατασκευάσουμε το επιθυμητό clause το οποίο ικανοποιείται από το θετικό literal της πρώτης μεταβλητής:

$$D_{1_{\nu\alpha\iota}} \;\rightarrow\; E_1$$

Αυτό που κάνουμε είναι το εξής: γράφουμε το clause στην κατάλληλη μορφή του, μαζί με τον διαχωριστικό χαρακτήρα του κόμματος πριν από αυτό. Για να ολοκληρωθεί το τρέχον βήμα, είναι απαραίτητο να αντιγράψουμε στην πρώτη συνιστώσα του τρέχοντος clause το θετικό literal (τη σημαδεμένη ή μη μεταβλητή) του τρέχοντος βήματος, το οποίο θα είναι και το πρώτο προς τα αριστερά literal που θα συναντίσουμε:

$$\begin{aligned} x0\}'E_1 &\rightarrow x0\}'D_1, \{x0, \sqcup, \sqcup\} \\ \bar{x}0\}'E_1 &\rightarrow \bar{x}0\}'D_1, \{\bar{x}0, \sqcup, \sqcup\} \end{aligned}$$

Η ερμηνεία των παραπάνω κανόνων είναι προφανής: απλά βλέπουμε πιο literal της τρέχουσας μεταβλητής είναι αληθές και το αντιγράφουμε στη σωστή θέση του τρέχοντος clause, μεταβαίνοντας ταυτόχρονα στη κατάσταση D_1 για την επαναληπτική δημιουργεία όλων των clauses που ανηκουν στο πρώτο τύπο (ήδη ικανοποιημένα από το αληθές literal) σχετικά με την πρώτη μεταβλητή μας.

Δυνητικά θα μεταβούμε στην κατάσταση $D_{1_{o\chi\iota}}$, όταν δηλαδή έχουμε τελειώσει με την παραγωγή όλων (ενδεχομένος και κανενός) των επιθυμητών clauses του πρώτου τύπου. Στην περίπτωση αυτή, είναι αναγκαίο να κατασκευαστούν τα επιθυμητά clauses του δευτέρου τύπου που αντιστοιχούν στην πρώτη μας μεταβλητή, κάτι που θα γίνει μέσω της καταστάσεως F:

$$
\begin{aligned}
D_{1_{o\chi\iota}} &\rightarrow F_{\nu\alpha\iota} \mid F_{o\chi\iota} \\
x0\}'F_{\nu\alpha\iota} &\rightarrow x0\}'D_{1_{o\chi\iota}}, [\bar{x}0, \sqcup, \sqcup] \\
\bar{x}0\}'F_{\nu\alpha\iota} &\rightarrow \bar{x}0\}'D_{1_{o\chi\iota}}, [x0, \sqcup, \sqcup]
\end{aligned}
$$

Οι παραπάνω κανόνες χρειάζονται κάποια εξήγηση. Αρχικά, μόλις αποφασίσαμε ότι έχουμε τελειώσει τη μη ντετερμινιστική παραγωγή όλων των επιθυμητών clauses του πρώτου τύπου για την πρώτη μεταβλητή μας, μεταβαίνουμε στην κατάσταση $D_{1_{o\chi\iota}}$. Από την κατάσταση αυτή, είναι απαραίτητο να αποφασίσουμε εάν θέλουμε ή όχι νέο clause δευτέρου τύπου. Αυτή η κατάσταση μοντελοποιείται από τις δύο αντίστοιχες μεταβλητές / καταστάσεις $F_{\nu\alpha\iota}$ και $F_{o\chi\iota}$. Στην πρώτη περίπτωση, θέλουμε να δημιουργήσουμε ένα clause το οποίο να αντιστοιχεί στην πρώτη μεταβλητή και να περιέχει το ψευδές Literal αυτής. Με άλλα λόγια, κοιτάζουμε πιο literal έχουμε σημειώσει ως αληθές στο σύνολο των μεταβλητών και τοποθετούμε στο τρέχον μας clause το *αντίθετό του* και μεταβαίνουμε επαναληπτικά σε μια εκ των καταστάσεων $F_{\nu\alpha\iota}$ ή $F_{o\chi\iota}$. Στη δεύτερη περίπτωση, έχουμε τελειώσει με την παραγωγή όλων των επιθυμητών clauses που αντιστοιχούν στην πρώτη μεταβλητή μας και είμαστε έτοιμοι να αρχίσουμε το επόμενο βήμα της παραγωγής των στιγμιοτύπων μας: την παραγωγή των υπολοίπων μεταβλητών και clauses.

9.3 Παραγωγή όλων των επιθυμητών μεταβλητών και των clauses του στιγμιοτύπου

Για την παραγωγή των επιπλέον μεταβλητών ακολουθούμε την βασική ιδέα που σκιαγραφήσαμε παραπάνω: σαρώνουμε την συμβολοσειρά ψάχνοντας για κενές θέσεις στα clauses που έχουμε ήδη δημιουργήσει. Εάν βρούμε κενή θέση τότε, νετερμινιστικά πλέον, παράγουμε μια νέα μεταβλητή μαζί με τα επιθυμητά αντίστοιχα clauses. Αντίθετα, εάν δε βρούμε κενή θέση τότε μπορούμε μη ντετερμινιστικά να επιλέξουμε ανάμεσα στο να τελειώσουμε την παραγωγή του στιγμιοτύπου μας και στο να συνεχίσουμε παράγοντας επιπλέον μεταβλητες και clauses.

Για να βρεθούμε στην κατάσταση παραγωγής μιάς νέας μεταβλητής σημαίνει ότι στο προηγούμενο βήμα αποφασίσαμε να μη δημιουργήσουμε νέο clause δευτέρου τύπου (κατάσταση $F_{o\chi\iota}$). Από την κατάσταση αυτή, είναι αναγκαίο να σαρώσουμε την λίστα των clauses ψάχνοντας για κενές θέσεις σε αυτά:

$$
\begin{aligned}
F_{o\chi\iota} &\rightarrow G \\
G\{ &\rightarrow \{G \\
Gx &\rightarrow xG \\
G\bar{x} &\rightarrow \bar{x}G \\
G1 &\rightarrow 1G \\
G0 &\rightarrow 0G \\
G, &\rightarrow ,G \\
G\} &\rightarrow \}G \\
G[&\rightarrow [G \\
G] &\rightarrow]G \\
G\bar{\}} &\rightarrow H\bar{\}} \\
G\sqcup &\rightarrow I\sqcup
\end{aligned}
$$

Η εξήγηση των παραπάνω κανόνων είναι η εξής: σαρώνουμε από αριστερά προς τα δεξιά την συμβολοσειρά μας μέχρι είτε να βρούμε κάποια κενή θέση (χαρακτήρας $\sqcup$) σε κάποιο clause οποιουδήποτε τύπου, είτε μέχρι να βρούμε τον τελευταίο χαρακτήρα της συμβολοσειράς τον οποίο και έχουμε συμβολίσει με τον χαρακτήρα }. Σε κάθε περίπτωση, μεταβαίνουμε στην αντίστοιχη κατάσταση (στην I στην πρώτη περίπτωση ή στην H στη δεύτερη).

Αρχίζουμε πρώτα την περιγραφή της καταστάσεως H η οποία είναι πολύ πιο εύκολη. Μόλις βρεθούμε στην εν λόγω κατάσταση, αποφασίζουμε μη ντετερμινιστικά εάν έχει τελείωσει η παραγωγή του στιγμιοτύπου μας, ή εάν θέλουμε να συνεχίσουμε να παράγουμε μεταβλητές και clauses (κατάσταση H). Στην πρώτη περίπτωση μεταβαίνουμε σε μια κατάσταση (K) η οποία θα ξεσημαδέψει τα όποια σύμβολα έχουμε σημαδέψει για ευκολία σε προηγούμενα βήματα και ύστερα θα τερματίσει:

$$H \rightarrow I \mid \omega$$

Επειδή η κατάσταση ω είναι η κατάσταση η οποία σηματοδοτεί το τέλος της παραγωγής της συμβολοσειράς μας, θα την περιγράψουμε στο τελος του κεφαλαίου. Προς το παρόν, θα περιγράψουμε την κατάσταση I τη διαισθητική λειτουγία της οποίας έχουμε περιγράψει αλλά για πληρότητα επαναλαμβάνουμε: έχουμε βρεί κάποιο clause με κενή κάποια συνιστώσα του και άρα πρέπει να

δημιουργήσουμε μια νέα μεταβλητή, να κατασκευάσουμε τα επιθυμητά clauses που αντιστοιχούν στην μεταβλητή αυτή και τέλος να τοποθετήσουμε τη μεταβλητή αυτή, ή την άρνησή της, σε clauses τα οποία έχουμε δημιουργήσει σε προηγούμενα βήματα.

Αρχικά φέρνουμε τον δρομέα στη κατάλληλη θέση, δηλαδή αμέσως αριστερά της πρώτης προς τα αριστερά μεταβλητής που θα συναντήσουμε στο σύνολο των μεταβλητών μας:

$$
\begin{aligned}
0I &\rightarrow I0 \\
1I &\rightarrow I1 \\
xI &\rightarrow Ix \\
\bar{x}I &\rightarrow I\bar{x} \\
,I &\rightarrow I, \\
\}I &\rightarrow I\} \\
\{I &\rightarrow I\{ \\
[I &\rightarrow I[\\
]I &\rightarrow I] \\
\}'I &\rightarrow I',\}'
\end{aligned}
$$

Ουσιαστικά, με το παραπάνω σύνολο κανόνων, μεταφέρουμε το δρομέα, μέσω της καταστάσεως I στο δεξί άκρο του συνόλου των μεταβλητών μας, αγνοώντας όσους χαρακτήες συναντήσουμε ενδιάμεσα. Τώρα, είναι αναγκαίο να μεταφερθεί ο δρομέας αριστερά της μεταβλητής αυτής ώστε να αρχίσει στο επόμενο βήμα η αντιγραφή και έπειτα η αύξηση του αύξοντος αριθμού αυτής:

$$
\begin{aligned}
0I' &\rightarrow I'0 \\
1I' &\rightarrow I'1 \\
xI' &\rightarrow \Lambda x \\
\bar{x}I' &\rightarrow \Lambda\bar{x} \\
0'I' &\rightarrow 0'\Lambda \\
1'I' &\rightarrow 1'\Lambda \\
x'I' &\rightarrow x'\Lambda \\
\bar{x}'I' &\rightarrow \bar{x}'\Lambda
\end{aligned}
$$

Όπου Λ είναι η διαδικασία που θα αντιγράψει την τρέχουσα μεταβλητή, ψηφίο προς ψηφίο, στα δεξιά αυτής, χωρίς όμως το τυχόν σημαδεμένο αρχικό σύμβολο της τρέχουσας μεταβλητής:

$$\begin{aligned}
\Lambda x &\rightarrow x'\Lambda_x \\
\Lambda \bar{x} &\rightarrow \bar{x}'\Lambda_x \\
\Lambda 0 &\rightarrow 0'\Lambda_0 \\
\Lambda 1 &\rightarrow 1'\Lambda_1 \\
\Lambda, &\rightarrow \Lambda',
\end{aligned}$$

Δηλαδή, αποθηκεύουμε τον τρέχοντα χαρακτήρα στην αντίστοιχη κατάσταση και προχωράμε τον δρομέα προς τα δεξιά ώστε να τον αντιγράψουμε στο κατάλληλο σημείο, αγνοώντας όλους τους ενδιάμεσους χαρακτήρες (πα περιγράψουμε χωρίς βλάβη της γενικότητας μόνο την περίπτωση της κατάστασης Λ_0, οι υπόλοιπες περιπτώσεις είναι τελείως ανάλογες):

$$\begin{aligned}
\Lambda_0 0 &\rightarrow 0\Lambda_0 \\
\Lambda_0 1 &\rightarrow 1\Lambda_0 \\
\Lambda_0, &\rightarrow ,\Lambda_0 \\
\Lambda_0 x &\rightarrow x\Lambda_0 \\
\Lambda_0\}' &\rightarrow I'0\}'
\end{aligned}$$

Στην κατάσταση Λ' είναι απαραίτητο να γυρίσουμε και να ξεσημαδέψουμε όλους τους σημαδεμένους χαρακτήρες και να γυρίσουμε τον δρομέα στην προηγούμενή του θέση:

$$\begin{aligned}
0'\Lambda' &\rightarrow \Lambda'0 \\
1'\Lambda' &\rightarrow \Lambda'1 \\
x'\Lambda' &\rightarrow x\Lambda'' \\
\bar{x}'\Lambda' &\rightarrow \bar{x}\Lambda'' \\
\Lambda''0 &\rightarrow 0\Lambda'' \\
\Lambda''1 &\rightarrow 1\Lambda'' \\
\Lambda'', &\rightarrow ,M
\end{aligned}$$

Οταν τελειώσει η αντιγραφή και το ξεσημάδεμα, θα μεταβούμε στην κατάσταση M σκοπός της οποίας είναι να αυξήσει κατά μία μονάδα τον αύξοντα αριθμό της τρέχουσας μεταβλητής μας. Οπως έχουμε ήδη πει, το βήμα αυτό ενδέχεται να αυξήσει κατά ένα τα ψηφία της αναπαράστασης του αριθμού αυτού. Οπως πάντα, είναι απαραίτητο να φέρουμε το δρομέα στο δεξί άκρο της τρέχουσας μεταβλητής, αγνοώντας όλους τους ενδιάμεσους χαρακτήρες:

$$\begin{aligned} Mx &\rightarrow xM \\ M0 &\rightarrow 0M \\ M1 &\rightarrow 1M \\ M\}' &\rightarrow N\}' \end{aligned}$$

Στην κατάσταση N αρχίζουμε την αύξηση του αριθμού που βρίσκεται στα αριστερά αυτής:

$$\begin{aligned} 0N &\rightarrow \Xi 1 \\ 1N &\rightarrow N0 \\ xN &\rightarrow x1\Xi \end{aligned}$$

Η κατάσταση Ξ είναι η κατάσταση ολοκλήρωσης της διαδικασίας αύξησης. Τώρα, αυτό που είναι αναγκαίο να κάνουμε, είναι να αποφασίσουμε τι αληθοτιμή θα αναθέσουμε στη τρέχουσα μεταβλητή, με τον ίδιο τρόπο που το κάναμε και για την πρώτη μεταβλητή μας. Πρώτ'απ'όλα όμως, είναι αναγκαίο να φέρουμε το δρομέα στη σωστή του θέση για την έναρξη της προαναφερθήσας διαδικασίας:

$$\begin{aligned} 0\Xi &\rightarrow \Xi 0 \\ 1\Xi &\rightarrow \Xi 1 \\ x\Xi &\rightarrow \Pi x \\ \Pi &\rightarrow \Pi_{true} \mid \Pi_{false} \\ \Pi_{true} &\rightarrow P \\ \Pi_{false}x &\rightarrow P\bar{x} \end{aligned}$$

Δηλαδή, αναθέτουμε αληθοτιμή, αλλάζοντας κατάλληλα, εάν αυτό είναι αναγκαίο, το πρώτο σύμβολο της τρέχουσας μεταβλητής, ώστε να θυμόμαστε σε μελλοντικά βήματα την ανάθεση αυτή.

Αφού τελειώσουμε με την επιθυμητή ανάθεση αληθοτιμής στην τρέχουσα μεταβλητή μας, είναι απαραίτητο να αρχίσουμε να κατασκευάζουμε μη ντετερμινιστικά τα δύο ειδών clauses που αντιστοιχούν στη μεταβλητή αυτή, με παρόμοιο τρόπο με αυτόν της πρώτης μεταβλητής. Φέρνουμε το δρομέα στη κατάλληλη θέση:

$$\begin{aligned} Px &\rightarrow xP \\ P\bar{x} &\rightarrow \bar{x}P \\ P0 &\rightarrow 0P \\ P1 &\rightarrow 1P \\ M\}' &\rightarrow \}'J \end{aligned}$$

Στην κατάσταση J αποφασίζουμε μη ντετερμινιστικά έαν θέλουμε κάποιο clause πρώτου τύπου (που ικανοποιείται δηλαδή από το αληθές literal της τρέχουσας μεταβλητής):

$$J \rightarrow J_{ναι} \mid J_{οχι}$$

Στην κατάσταση $J_{ναι}$ αποφασίζουμε ότι θέλουμε ένα τέτοιο clause οπότε αρχίζουμε την κατασκευή του:

$$J_{ναι} \rightarrow K, \tilde{\{}, \sqcup, \sqcup\tilde{\}}$$

Δημιουργούμε δηλαδή, όχι στην πλήρη μορφή του, το πρότυπο clause, σημαδεύοντας όμως τα δύο άγκυστρα που το περικλείουν ώστε να μπορούμε να το αναγνωρίζουμε σε επόμενο βήμα, όπως περιγράψαμε και στη σκιαγράφηση της μεθόδου. Αυτό που απομένει είναι να αντιγράψουμε το αληθές literal της τρέχουσας μεταβλητής που βρίσκεται στο δεξί άκρο του συνόλου των μεταβλητών, στη πρώτη συνιστώσα του clause που μόλις δημιουργήσαμε:

$$
\begin{aligned}
\}'K &\rightarrow K\}' \\
0'K &\rightarrow K0' \\
1'K &\rightarrow K1' \\
0K &\rightarrow 0'K_0 \\
1K &\rightarrow 1'K_1 \\
xK &\rightarrow x'K_x \\
\bar{x}K &\rightarrow \bar{x}'K_{\bar{x}} \\
,K &\rightarrow ,L
\end{aligned}
$$

Δηλαδή, βρήσκουμε το πρώτο μη σημαδεμένο χαρακτήρα και τον αποθηκεύουμε στη κατάλληλη κατάσταση ώστε να τον αντιγράψουμε στο κατάλληλο σημείο. Η λειτουργία των καταστάσεων K_0, K_1, K_x και $K_{\bar{x}}$ είναι ακριβώς αυτή. Παρακάτω θα περιγράψουμε τη λειτουργία της K_0. Οι υπόλοιπες καταστάσεις είναι τελείως ανάλογες:

$$
\begin{aligned}
K_0 0' &\rightarrow 0'K_0 \\
K_0 1' &\rightarrow 1'K_0 \\
K_0\}', \tilde{\{} &\rightarrow K\}', \tilde{\{}0
\end{aligned}
$$

Μόλις τελειώσει η αντιγραφή θα μεταβούμε στην κατάσταση L. Τότε, είναι απαραίτητο να ξεσημαδέψουμε όλα τα σημαδεμένα σύμβολα και να φέρουμε στο δρομέα στη κατάλληλη θέση ώστε να επαναλάβουμε το προηγούμενο μη ντετερμινιστικό βήμα:

$$
\begin{aligned}
Lx' &\rightarrow xL \\
L\bar{x}' &\rightarrow \bar{x}L \\
L0' &\rightarrow 0L \\
L1' &\rightarrow 1L \\
L\}' &\rightarrow \}'J
\end{aligned}
$$

Το παραπάνω μη ντετερμινιστικό βήμα παραγωγής clauses πρώτου τύπου, που μοντελοποιείται από την κατάσταση J, δυνητικά θα μας οδηγήσει στην

κατάσταση $J_{o\chi\iota}$, όταν δηλαδή έχουμε κατασκευάσει τα επιθυμητά clauses του τύπου αυτού για τη τρέχουσα μεταβλητή μας. Τώρα, είναι αναγκαίο να επαναλάβουμε την παραπάνω διαδικασία για τα clauses του δευτέρου τύπου που επιθυμούμε να κατασκευάσουμε για την τρέχουσα μεταβλητή μας:

$$\begin{aligned} J_{o\chi\iota} &\rightarrow J_{2_{\nu\alpha\iota}} \mid J_{2_{o\chi\iota}} \\ J_{2_{\nu\alpha\iota}} &\rightarrow Q, \tilde{[}, \sqcup, \tilde{\sqcup]} \end{aligned}$$

Οπως και πριν, Δημιουργούμε δηλαδή, όχι στην πλήρη μορφή του, το πρότυπο clause, σημαδεύοντας όμως τα δύο άγκυστρα που το περικλείουν ώστε να μπορούμε να το αναγνωρίζουμε σε επόμενο βήμα. Τώρα, είναι απαραίτητο να αντιγράψουμε το *αντίθετο* του literal που είναι γραμμένο στο σύνολο των μεταβλητών μας και αντιστοιχεί στη τρέχουσα μεταβλητής μας:

$$\begin{aligned} \}'Q &\rightarrow Q\}' \\ 0'Q &\rightarrow Q0' \\ 1'Q &\rightarrow Q1' \\ 0Q &\rightarrow 0'Q_0 \\ 1Q &\rightarrow 1'Q_1 \\ xQ &\rightarrow x'Q_{\bar{X}} \\ \bar{x}Q &\rightarrow \bar{x}'K_x \\ ,Q &\rightarrow ,L' \end{aligned}$$

Χωρίς βλάβη της γενικότητας, περιγράφουμε την κατάσταση Q_1 οποία αντιγράφει τον αποθηκευμένο χαρακτήρα ("1") στην κατάλληλη θέση:

$$\begin{aligned} Q_1 0' &\rightarrow 0'Q_1 \\ Q_1 1' &\rightarrow 1'Q_1 \\ Q_1\}', \tilde{[} &\rightarrow K\}', \tilde{[}0 \end{aligned}$$

Μόλις τελειώσει η αντιγραφή, μεταβαίνουμε στην κατάσταση L' λειτουργία της οποίας είναι να ξεσημαδέψει όλους τους σημαδεμένους χαρακτήρες και να

φέρει στο δρομέα στη κατάλληλη θέση ώστε να επαναληφθεί το προηγούμενο μη ντετερμινιστικό βήμα παραγωγής clauses του δευτέρου τύπου:

$$
\begin{aligned}
L'x' &\rightarrow xL' \\
L'\bar{x}' &\rightarrow \bar{x}L' \\
L'0' &\rightarrow 0L' \\
L'1' &\rightarrow 1L' \\
L'\}' &\rightarrow \}'J_{o\chi\iota}
\end{aligned}
$$

Η παραπάνω διαδικασία παραγωγής clauses οποιουδήποτε τύπου θα τελειώσει όταν βρεθούμε στην κατάσταση $J_{2_{o\chi\iota}}$. Από την κατάσταση αυτή, αυτό που θέλουμε είναι να τοποθετήσουμε την τρέχουσα μεταβλητή, ή την άρνηση αυτής, σε άλλα clauses τα οποία έχουμε δημιουργήσει σε προηγούμενο βήμα και τα οποία έχουν κενή κάποια συνιστώσα τους. Τα clauses αυτά τα διαχωρίζουμε από τα clauses που μόλις κατασκευάσαμε στο τρέχον βήμα μας από το γεγονός ότι *δεν* είναι σημαδεμένα.

Η διαδικασία αυτή θα μοντελοποιηθεί μέσω δύο μεταβλητών-καταστάσεων: η πρώτη κατάσταση (R_1) θα τοποθετεί το αληθές literal της τρέχουσας μεταβλητής μας μη ντετερμινιστικά σε κάποιο clause, όπου υπάρχει χώρος, αλλάζοντας εάν καταστεί αναγκαίο τα άγκυστρα του clause αυτού. Η δεύτερη (R_2) θα κάνει την ίδια δουλειά για το αρνητικό literal όμως, προσέχοντας να μην τοποθετήσει το ψευδές literal αυτό ως τρίτη συνιστώσα σε κάποιο μη ικανοποιημένο έχρι στιγμής clause.

Από την κατάσταση $J_{2_{o\chi\iota}}$ μεταβαίνουμε στην πρώτη μας κατάσταση:

$$
J_{2_{o\chi\iota}} \rightarrow J'
$$

Από την κατάσταση αυτή (J'), μεταφέρουμε τον δρομέα στο πρώτο μη σημαδεμένο clause:

$$
\begin{aligned}
J'\tilde{\{} &\rightarrow \tilde{\{}J' \\
J'\tilde{[} &\rightarrow \tilde{[}J' \\
J'x &\rightarrow xJ' \\
J'\bar{x} &\rightarrow \bar{x}J' \\
J'0 &\rightarrow 0J' \\
J'1 &\rightarrow 1J' \\
J'\sqcup &\rightarrow \sqcup J' \\
J'\tilde{\}} &\rightarrow \tilde{\}}J' \\
J'\tilde{]} &\rightarrow \tilde{]}J' \\
J', &\rightarrow ,J' \\
J'\{ &\rightarrow \{R_1 \\
J'[&\rightarrow [R_1 \\
J'\bar{\}} &\rightarrow J''\bar{\}}
\end{aligned}
$$

Δηλαδή, από την κατάσταση J' αγνοούμε όλους τους ενδιάμεσους χαρακτήρες μέχρι να συναντήσουμε κάποιο μη σημαδεμένο clause, οπότε και μεταβαίνουμε στην κατάσταση R_1, ή μέχρι να βρούμε το τέλος της συμβολοσειράς μας, οπότε και μεταβαίνουμε στην κατάλληλη κατάσταση (J''), που σημαίνει ότι δεν υπάρχουν άλλα διαθέσιμα clauses οπότε μπορούμε να προχωρήσουμε στην μη ντετερμινιστική παραγωγή μιας νέας μεταβλητής (κάτι που μοντελοποιείται από την κατάσταση H που περιγράψαμε παραπάνω):

$$
J'' \rightarrow H
$$

Στην παρούσα φάση, εάν από την κατάσταση H αποφασίσουμε μη ντετερμινιστικά να μεταβούμε στην κατάσταση I, η οποία θα μας κατασκευάσει μια νεά μεταβλητή, είναι απαραίτητο, καθώς θα προχωράμε προς τα αριστερά ώστε να βρεθούμε στο σύνολο των μεταβλητών μας, να ξεσημαδέψουμε τα clauses που σημαδέψαμε όταν θεωρούσαμε την τελευταία μεταβλητή που κατασκευάσαμε. Αρα στους αντίστοιχους κανόνες της μεταβλητής I είναι απαραίτητο τα προσθέσουμε τους κάτωθι:

$$
\begin{aligned}
\tilde{\}}I &\rightarrow I\} \\
\tilde{\{}I &\rightarrow I\{ \\
\tilde{[}I &\rightarrow I[\\
\tilde{]}I &\rightarrow I]
\end{aligned}
$$

Μόλις βρεθούμε στην κατάσταση R_1 σημαίνει ότι έχουμε βρει ένα μη σημαδεμένο clause κάποιου τύπου, δεν μας ενδιαφέρει προς το παρόν τι τύπου είναι. Πριν αποφασίσουμε μη ντετερμινιστικά να προσθέσουμε το τρέχον αληθές literal το clause αυτό, είναι αναγκαίο να δούμε πρώτα εάν υπάρχει χώρος:

$$
\begin{aligned}
R_1 x &\rightarrow x R_1 \\
R_1 \bar{x} &\rightarrow \bar{x} R_1 \\
R_1 0 &\rightarrow 0 R_1 \\
R_1 1 &\rightarrow 1 R_1 \\
R_1, &\rightarrow , R_1 \\
R_1 \sqcup &\rightarrow R' \sqcup' \\
R_1\} &\rightarrow \} R'' \\
R_1] &\rightarrow] R''
\end{aligned}
$$

Στην κατάσταση R' μεταβαίνουμε εάν βρούμε ότι *υπάρχει* χώρος, ενώ στην αντίθετη περίπτωση μεταβαίνουμε στην κατάσταση R'' ώσετ να συνεχίσουμε στο επόμενο clause, εάν υπάρχει, ή να ακολουθήσουμε την ίδια διαδικασία για το αρνητικό literal της τρέχουσας μεταβλητής. Επειδή αυτή η περίπτωση είναι απλούστερη, την περιγραφουμε αμέσως:

$$
\begin{aligned}
R'', \{ &\rightarrow \{ R \\
R'', [&\rightarrow [R \\
R'' \bar{\}} &\rightarrow r \bar{\}}
\end{aligned}
$$

Στην κατάσταση r είναι απαραίτητο να γυρίσουμε προς τα πίσω και να επαναλάβουμε τα ανωτέρω, αυτή τη φορά για το αρνητικό literal της τρέχουσας μεταβλητής μας:

$$
\begin{array}{rcl}
]r & \rightarrow & r] \\
\}r & \rightarrow & r\} \\
0r & \rightarrow & r0 \\
1r & \rightarrow & r1 \\
xr & \rightarrow & rx \\
\bar{x}r & \rightarrow & r\bar{x} \\
,r & \rightarrow & r, \\
\sqcup r & \rightarrow & r \sqcup \\
\{r & \rightarrow & r\{ \\
[r & \rightarrow & r[\\
\tilde{\}}r & \rightarrow & \tilde{\}}R_2 \\
\tilde{]}r & \rightarrow & \tilde{]}R_2 \\
\bar{\}}r & \rightarrow & \bar{\}}R_2
\end{array}
$$

Πριν περιγράψουμε την κατάσταση R_2, είναι απαραίτητο να συνεχίσουμε με την περιγραφή της καταστάσεως R', κατάσταση στην οποία και μεταβαίνουμε εάν βρούμε χώρο στο τρέχον clause που ελέγχουμε. Από τη στιγμή που βρήκαμε τέτοιο χώρο, θα ακολουθήσουμε τα βήματα που αναφέραμε στη σκιαγράφηση: *πρώτα* θα αντιγράψουμε στη τρέχουσα θέση το τρέχον θετικό literal και *έπειτα* θα κάνουμε τον απαραίτητο έλεγχο για το εάν υπάρχει το clause αυτό. Για αποφυγή όμως άσκοπων συγκρίσεων, ο έλεγχος αυτό θα γίνεται μόνο εάν το τρέχον κανό σύμβολο που βρήκαμε βρίσκεται στη τρίτη συνιστώσα του clause.

Αρχίζουμε με την αντιγραφή του τρέχοντος literal, που βρίσκεται στο δεξί τέλος του συνόλου των μεταβλητών, στη τρέχουσα θέση. Εχουμε ήδη σημαδέψει τον χαρακτήρα $\sqcup$ ώστε να βρίσκουμε τη θέση μας και πηγαίνουμε να αντιγράψουμε έναν έναν τους χαρακτήρες του τρέχοντος literal:

$$
\begin{array}{rcl}
0R' & \rightarrow & R'0 \\
1R' & \rightarrow & R'1 \\
xR' & \rightarrow & R'x \\
\bar{x}R' & \rightarrow & R'\bar{x} \\
0'R' & \rightarrow & R'0' \\
1'R' & \rightarrow & R'1' \\
x'R' & \rightarrow & R'x' \\
\bar{x}'R' & \rightarrow & R'\bar{x}' \\
,R' & \rightarrow & R', \\
]R' & \rightarrow & R'] \\
\}R' & \rightarrow & R'\} \\
\{R' & \rightarrow & R'\{ \\
[R' & \rightarrow & R'[\\
\tilde{\}}R' & \rightarrow & \tilde{\}}R' \\
\tilde{]}R' & \rightarrow & \tilde{]}R' \\
\tilde{\{}R' & \rightarrow & \tilde{\}}R' \\
\tilde{[}R' & \rightarrow & \tilde{]}R' \\
\}'R' & \rightarrow & \}'S
\end{array}
$$

Ας θυμηθούμε ότι με τον χαρακτήρα $\}'$ σημαδέψαμε το τέλος του συνόλου των μεταβλητών μας οπότε, μόλις το συναντίσουμε, ξέρουμε ότι το στοιχείο που βρίσκεται στα αριστερά αυτού του χαρακτήρα είναι αυτό που θέλουμε να γράψουμε στη σημαδεμένη θέση του τρέχοντος clause. Η αντιγραφή θα γίνει με τον γνωστό τρόπο, ένα προς ένα τα ψηφιά:

$$
\begin{aligned}
0'S &\rightarrow S0' \\
1'S &\rightarrow S1' \\
0S &\rightarrow 0'S_0 \\
1S &\rightarrow 1'S_1 \\
xS &\rightarrow x'S_x \\
\bar{x}S &\rightarrow \bar{x}'S_{\bar{x}} \\
x'S &\rightarrow S'x' \\
\bar{x}'S &\rightarrow S'\bar{x}'
\end{aligned}
$$

όπου στην κατάσταση S' μεταβαίνουμε όταν διαπιστώσουμε ότι έχει τελειώσει η αντιγραφή οπότε και πρέπει να μεταβούμε στο σημείο που μόλις αντιγράψαμε το τρέχον literal αφού πρώτα ξεσημαδέψουμε όλους τους σημαδεμένους χαρακτήρες. Πριν προχωρήσουμε όμως στη περιγραφή της καταστάσεως αυτής, είναι χρήσιμο να περιγράψουμε πώς γίνεται η αντιγραφή. Χωρίς βλάβη της γενικότητας, αρκει να περιγράψουμε μία περίπτωση μόνο, έστω την S_0:

$$
\begin{aligned}
S_0 0 &\rightarrow 0 S_0 \\
S_0 1 &\rightarrow 1 S_0 \\
S_0 0' &\rightarrow 0' S_0 \\
S_0 1' &\rightarrow 1' S_0 \\
S_0 \}' &\rightarrow \}' S_0 \\
S_0 , &\rightarrow , S_0 \\
S_0 \{ &\rightarrow \{ S_0 \\
S_0 [&\rightarrow [S_0 \\
S_0 \} &\rightarrow \} S_0 \\
S_0] &\rightarrow] S_0 \\
S_0 \bar{0} &\rightarrow R' \bar{0} \bar{0} \\
S_0 \bar{1} &\rightarrow R' \bar{1} \bar{0} \\
S_0 \sqcup' &\rightarrow R' \bar{0} \\
S_0 \tilde{\{} &\rightarrow \tilde{\{} S_0 \\
S_0 \tilde{[} &\rightarrow \tilde{[} S_0 \\
S_0 \tilde{\}} &\rightarrow \tilde{\}} S_0 \\
S_0 \tilde{]} &\rightarrow \tilde{]} S_0 \\
S_0 \sqcup &\rightarrow \sqcup S_0
\end{aligned}
$$

Οπως βλέπουμε, κάθε έναν χαρακτήρα που αντιγράφουμε τον σημαδεύουμε ($\bar{0}$ ή $\bar{1}$) ώστε να μπορούμε να βρίσκουμε εύκολα πιο είναι το σημείο που πρέπει να εναποθέσουμε τον τρέχοντα χαρακτήρα αντιγραφής.

Η μόνη διαφοροποίηση μεταξύ των διαφόρων καταστάσεων (S_0, S_1 κτλ) είναι στις πριπτώσεις S_x και $S_{\bar{x}}$ που απλά αγνοούμε σε κάθε περίπτωση να σημαδέψουμε το τρέχον σύμβολο, καθώς ξέρουμε ότι είναι το τελευταίο που αντιγράφουμε, οπότε δεν χρειάζεται κανένα βοηθητικό σημάδεμα. Αντιγράφουμε απλά τον χαρακτήρα *ώς έχει*.

Οπως είπαμε, μόλις διαπιστώσουμε ότι έχει τελειώσει η αντιγραφή, μεταβαίνουμε στην κατάσταση S' σκοπός της οποίας είναι να ξεσημαδέψει όλους τους σημαδεμένους χαρακτήρες και να φέρει το δρομέα στη κατάλληλη θέση ώστε να αρχίσει η διαδικασία σύγκρισης του τρέχοντος clause με όλα τα υπόλοιπα:

$$
\begin{aligned}
S'0' &\rightarrow 0S' \\
S'1' &\rightarrow 1S' \\
S'x' &\rightarrow xS' \\
S'\bar{x}' &\rightarrow \bar{x}S' \\
S'\}' &\rightarrow \}'S' \\
S', &\rightarrow ,S' \\
S'\{ &\rightarrow \{S' \\
S'[&\rightarrow [S' \\
S'\} &\rightarrow \}S' \\
S'] &\rightarrow]S' \\
S'\tilde{\{} &\rightarrow \tilde{\{}S' \\
S'\tilde{[} &\rightarrow \tilde{[}S' \\
S'\tilde{\}} &\rightarrow \tilde{\}}S' \\
S'\tilde{]} &\rightarrow \tilde{]}S' \\
S'0 &\rightarrow 0S' \\
S'1 &\rightarrow 1S' \\
S'\sqcup &\rightarrow \sqcup S' \\
S'\bar{0} &\rightarrow 0S'' \\
S'\bar{1} &\rightarrow 1S'' \\
S''\bar{0} &\rightarrow 0S'' \\
S''\bar{1} &\rightarrow 1S'' \\
S''\} &\rightarrow T_1\} \\
S''] &\rightarrow T_1] \\
S'', &\rightarrow T_2,
\end{aligned}
$$

Η κατάσταση T_1 είναι αυτή η οποία θα συγκρίνει το τρέχον clause με *όλα* τα υπόλοιπα clauses ρος τα αριστερά του τρέχοντος *μόνο εάν κάτι τέτοιο είναι αναγκαίο*, δηλαδή μόνο εάν η τοποθέτηση του literal έχει γίνει στη τρίτη συνιστώσα του τρέχοντος clause. Για το λόγο αυτό, στο παραπάνω σύνολο κανόνων, έχουμε προνοήσει να μεταβούμε στην κατάλληλη κατάσταση, μέσω των δεξιών συμφραζομένων της τρέχουσας κατάστασής μας (S''). Εάν μεταβούμε *ντετερμινιστικά* πλεόν στην κατάσταση T_2, δεν έχουμε να κάνουμε κανέναν έλεγχο ύπαρξης διπλότυπου clause. Το μόνο που απομένει είναι να

αλλάξουμε τους χαρακτήρες που περικλύουν το τρέχον clause, εάν φυσικά κάτι τέτοιο είναι αναγκαίο.

Οπως ήδη αναφέραμε, είμαστε στο βήμα στο οποίο προσθέτουμε το *αληθές* literal της τρέχουσας μεταβλητής μας στα clauses τις επιλογής μας. Αρα, δεδομένου ότι προσθέτουμε αληθές literal το βήμα της προηγούμενης παραγράφου συνίσταται από έναν απλό έλεγχο:

$$
\begin{aligned}
0T_2 &\rightarrow T_20 \\
1T_2 &\rightarrow T_21 \\
xT_2 &\rightarrow T_2x \\
\bar{x}T_2 &\rightarrow T_2\bar{x} \\
,T_2 &\rightarrow T_2, \\
[T_2 &\rightarrow \{T' \\
\{T_2 &\rightarrow \tilde{[}U
\end{aligned}
$$

όπου στην κατάσταση T' μεταβαίνουμε εάν το clause είναι ήδη ικανοποιημένο, οπότε δεν χρειάζεται να κάνουμε τίποτε άλλο στο τρέχον βήμα, ενώ στην κατάσταση U μεταβαίνουμε όταν δούμε ότι το τρέχον clause *δεν* ήταν ικανοποιήσιμο και απαιτείται να αλλαχθεί και ο δεξιός αντίστοιχος χαρακτήρας ώστε να έρθει το clause στην επιθημητή και σωστή μορφή του:

$$
\begin{aligned}
Ux &\rightarrow xU \\
X\bar{x} &\rightarrow \bar{x}U \\
U0 &\rightarrow 0U \\
U1 &\rightarrow 1U \\
U\} &\rightarrow \tilde{]}T'
\end{aligned}
$$

Με τον παραπάνω τρόπο, αλλάξαμε τους διακριτικούς χαρακτήρες του τρέχοντος clause αι επίσης το σημαδέψαμε κατάλληλα ώστε να ξέρουμε ότι έχει ήδη τοποθετηθεί σε αυτό κάποιο literal ης τρέχουσας μεταβλητής μας, για να αναγνωρίζουμε το clause σε μελλοντικά υπο-βήματα του τρέχοντος βήματός μας.

Και στις δύο προηγούμενες περιπτώσεις, είτε δηλαδή αλλάξουμε τους διακριτικούς χαρακτήρες του τρέχοντος clause είτε όχι, στο τέλος των δύο αυτών περιπτώσεων μεταβαίνουμε στην κατάσταση T', στην οποία κατάσταση πρέπει

να μεταβούμε στο επόμενο (και μη σημαδεμένο clause) ώστε να συνεχιστεί η μη ντετερμινιστική διαδικασία εισαγωγής του τρέχοντος θετικού (αληθούς) literal σε όποια κενή θέση, μη ντετερμινιστικά, κατάσταση που έχουμε περιγράψει προηγουμένως και μοντελοποιείται από την J':

$$T' \rightarrow J'$$

Απομένει τώρα η περιγραφή της καταστάσεως T_1, κατάσταση στην οποία μεταβαίνουμε όταν θέλουμε να συγκρίνουμε το τρέχον clause με όλα τα υπόλοιπα στα *αριστερά* αυτού. Πιο συγκεκριμένα, όταν βρεθούμε στην κατάσταση αυτή, έχουμε αντιγράψει το τρέχον (αληθές) literal στην επιθυμητή θέση. Η θέση αυτή είναι η τρίτη συνιστώσα κάποιου clause το οποίο κατ'ανάγκη είναι ικανοποιήσιμο. Αυτό που θέλουμε είναι να συγκρίνουμε χαρακτήρα προς χαρακτήρα το τρέχον clause με όσα clauses προς τα αριστερά αυτού είναι σημαδεμένα. Αυτό γιατί μας ενδιαφέρουν clauses τα οποία περιέχουν το εν λόγω literal (και άρα είναι σημαδεμένα) και επιπλέον, αφού η θεώρηση των clauses γίνεται από αριστερά προς τα δεξιά, τα clauses που μπορεί να περιέχουν το literal αυτό θα βρίσκονται απαραίτητα προς τα αριστερά μας.

Η σύγκριση θα γίνει με τον γνωστό τρόπο: βρίσκουμε και θυμόμαστε τον πρώτο μη σημαδεμένο χαρακτήρα του τρέχοντος clause και προχωράμε προς τα αριστερά μέχρι να βρούμε τον αντίστοιχο τρέχον ψηφίο του άλλου clause, το οποίο και θέλουμε να συγκρίνουμε για ομοιότητα με το τρέχον μας:

$$\begin{aligned}
T_1 &\rightarrow V \\
0'V &\rightarrow V0' \\
1'V &\rightarrow V1' \\
x'V &\rightarrow Vx' \\
\bar{x}'V &\rightarrow V\bar{x}' \\
,'V &\rightarrow V,' \\
0V &\rightarrow V_0 0' \\
1V &\rightarrow V_1 1' \\
xV &\rightarrow V_x x' \\
\bar{x}V &\rightarrow V_{\bar{x}} \bar{x}' \\
,V &\rightarrow V_,,'
\end{aligned}$$

Αφού αποθηκεύσαμε στην κατάλληλη κατάσταση το τρέχον σύμβολο σύγκρισης, είναι αναγκαίο να το συγκρίνουμε με τον αντίστοιχό του χαρακτήρα στο έτερο clause (όπως πάντα, θα περιγράψουμε μόνο την μία περίπτωση-χωρίς βλάβη της γενικότητας έστω την V_x):

$$
\begin{aligned}
0V_x &\rightarrow V_x0\\
1V_x &\rightarrow V_x1\\
xV_x &\rightarrow V_xx\\
\bar{x}V_x &\rightarrow V_x\bar{x}\\
,V_x &\rightarrow V_x,\\
\{V_x &\rightarrow V_x\{\\
[V_x &\rightarrow V_x[\\
\}V_x &\rightarrow V_x\}\\
]V_x &\rightarrow V_x]\\
\tilde{\{}V_x &\rightarrow \tilde{\{}W\\
\tilde{[}V_x &\rightarrow \tilde{[}W\\
\}'V_x &\rightarrow \}'W''\\
\tilde{\}}V_x &\rightarrow V_x'\tilde{\}}\\
\tilde{]}V_x &\rightarrow V_x'\tilde{]}\\
0'V_x' &\rightarrow V_x'0'\\
1'V_x' &\rightarrow V_x'1'\\
x'V_x &\rightarrow V_x'x'\\
\bar{x}'V_x' &\rightarrow V_x'\bar{x}'\\
\sqcup V_x' &\rightarrow \sqcup W\\
,'V_x' &\rightarrow V_x','\\
0V_x' &\rightarrow 0W\\
1V_x' &\rightarrow 1W\\
xV_x' &\rightarrow x'W'\\
\bar{x}V_x' &\rightarrow \bar{x}W\\
,V_x' &\rightarrow ,W'
\end{aligned}
$$

Το παραπάνω σύνολο κανόνων, αν και μεγάλο, παρ'όλ'αυτά είναι ευκολονόητο: αρχικά, αγνοούμε οποιονδήποτε χρακτήρα, μέχρι να φτάσουμε στο

δεξιό διακριτικό χρακτήρα του πρώτου σημαδεμένου clause που θα συναντίσουμε προς τα αριστερά. Μόλις τον συναντίσουμε, μεταβαίνουμε στην κατάσταση V'_* ώστε να θυμόμαστε το προηγούμενο γεγονός. Επειτα, αγνοούμε πάλι όλους τους σημαδεμένους, και κατ'επέκταση ήδη ελεγμένους, χαρακτήρες και συγκρίνουμε τον πρώτο μη σημαδεμένο που θα συναντήσουμε με αυτόν που έχουμε αποθηκεύσει στην κατάστασή μας. Εάν είναι ίδιοι, η σύγκριση πρέπει να προχωρήσει με τους επόμενους δύο χαρακτήρες από τα δύο αντίστοιχα clauses, κάτι που γίνεται με την μετάβαση στην κατάσταση W'. Αντίθετα, εάν οι χαρακτήρες αυτοί διαφέρουν, τότε είναι προφανές ότι και τα δύο clauses διαφέρουν όποτε σταματάμε τη σύγκριση μεταβαίνοντας στην κατάλληλη κατάσταση W.

Δύο σημεία άξια προσοχής είναι τα εξής: 1) καθώς μετακινούμε το δρομέα μας προς τα αριστερά, ενδεχομένως να μη συναντήσουμε κανένα σημαδεμένο clause. Αυτό θα το καταλάβουμε μόλις φτάσουμε στον δεξί τερματικό χαρακτήρα του συνόλου των μεταβλητών μας ($\}'$). Μόλις τον συναντήσουμε, ξέρουμε ότι δεν έχει μείνει κάποιο υποψήφιο clause να συγκρίνουμε με τον τρέχον μας, οπότε μπορούμε να είμαστε σίγουροι ότι δεν υπάρχει κίνδυνος δημιουργείας διπλοτύπου clause. Αρα, μπορούμε να συνεχίσουμε το βήμα προσθήκης του τρέχοντος αληθούς literal στα επιθυμητά clauses, κάτι που θα γίνει αρχικώς μέσω της μετάβασης στην κατάλληλη κατάσταση (W''). 2) Μόλις τελειώσει η σύγκριση των δύο clauses (του τρέχοντος και του πρώτου προς τα αριστερά αυτού), είναι απαραίτητο να ξεσημαδέψουμε το clause αυτό. Ο λόγος είναι ότι πλεόν αυτό το clause ξέρουμε ότι είναι ίδιο με αυτό που θέλουμε να δημιουργήσουμε και, επιπλέον, αν το αφήσουμε σημαδεμένο κινδυνεύουμε να το ξανασυγκρίνουμε σε επόμενο βήμα! Ο λόγος που κρατούσαμε τα clauses αυτά σημαδεμένα ήταν ακριβώς αυτός: να τα αναγνωρίζουμε όταν θέλουμε να κάνουμε σύγκριση. Απαξ και έγινε η σύγκριση, δεν υπάρχει κανένας λόγος να τα κρατάμε σημαδεμένα.

Θα περιγράψουμε μία μία τις τρεις περιπτώσεις αυτές, που μοντελοποιούνται από τις καταστάσεις W, W' και W'' αρχίζοντας από την W. Στην κατάσταση αυτή, γνωρίζουμε ότι τα δύο clauses διαφέρουν, οπότε είναι αναγκαίο, αφού πρώτα ξεσημαδέψουμε το clause αυτό, να μετακινήσουμε το δρομέα στα δεξιά του τρέχοντος clause ώστε να συνεχιστεί η προηγούμενη διαδικασία σύγκρισης αυτού του clause με όλα τα υπόλοιπα σ ημαδεμένα προς τα αριστερά clauses ου στιγμιοτύπου μας, ξεσημαδεύοντας παράλληλα ότι σημαδεμένους χαρακτήρες συναντίσουμε:

$$
\begin{aligned}
0W &\rightarrow W0 \\
1W &\rightarrow W1 \\
xW &\rightarrow Wx \\
\bar{x}W &\rightarrow W\bar{x} \\
\sqcup W &\rightarrow W \sqcup \\
,W &\rightarrow W, \\
\tilde{\{}W &\rightarrow \{W_1 \\
\tilde{[}W &\rightarrow [W_1
\end{aligned}
$$

Τώρα μετακινούμαστε προς τα δεξιά ώστε να ξεσημαδέψουμε και το δεξιό διακριτικό χαρακτήρα:

$$
\begin{aligned}
W_1x &\rightarrow xW_1 \\
W_1\bar{x} &\rightarrow \bar{x}W_1 \\
W_10 &\rightarrow 0W_1 \\
W_11 &\rightarrow 1W_1 \\
W_1, &\rightarrow ,W_1 \\
W_1x' &\rightarrow xW_1 \\
W_1\bar{x}' &\rightarrow \bar{x}W_1 \\
W_10' &\rightarrow 0W_1 \\
W_11' &\rightarrow 1W_1 \\
W_1,' &\rightarrow ,W_1 \\
W_1\sqcup &\rightarrow \sqcup W_1 \\
W_1\tilde{\}} &\rightarrow \}W_2 \\
W_1\tilde{]} &\rightarrow]W_2
\end{aligned}
$$

Από τη κατάσταση W_2 μετακινούμαστε προς τα δεξιά μέχρι να βρούμε το τρέχον μας clause το οποίο θα το αναγνωρίσουμε βλέποντας τους σημαδεμένους του χαρακτήρες τους οποίους και θα ξεσημαδέψουμε ώστε να συνεχιστεί η σύγκριση με τα υπόλοιπα clauses:

$$
\begin{aligned}
W_2 x &\rightarrow x W_2 \\
W_2 \bar{x} &\rightarrow \bar{x} W_2 \\
W_2 0 &\rightarrow 0 W_2 \\
W_2 1 &\rightarrow 1 W_2 \\
W_2 , &\rightarrow , W_2 \\
W_2 x' &\rightarrow x W_2' \\
W_2 \bar{x}' &\rightarrow \bar{x} W_2' \\
W_2 0' &\rightarrow 0 W_2' \\
W_2 1' &\rightarrow 1 W_2' \\
W_2 ,' &\rightarrow , W_2' \\
W_2' 0' &\rightarrow 0 W_2' \\
W_2' 1' &\rightarrow 1 W_2' \\
W_2' ,' &\rightarrow , W_2' \\
W_2 \sqcup &\rightarrow \sqcup W_2 \\
W_2 \} &\rightarrow \} W_2 \\
W_2] &\rightarrow] W_2 \\
W_2 \{ &\rightarrow \} W_2 \\
W_2 [&\rightarrow] W_2 \\
W_2' \} &\rightarrow \} V \\
W_2'] &\rightarrow] V
\end{aligned}
$$

Η εξήγηση των παραπάνω κανόνων είναι η εξής: στην κατάσταση W_2 αγνοούμε όλους τους ενδιάμεσους χαρακτήρες μέχρι να εισέλθουμε στο τρέχον μας clause, κάτι το οποίο καταλαβαίνουμε όταν συναντήσουμε τον πρώτο σημαδεμένο χαρακτήρα, ο οποίος μπορεί να είναι είτε ο x ή $\bar{x}$ είτε κάποιο άλλο ψηφίο (του κόμματος συμπεριλαμβανομένου). Μόλις γίνει αυτό, μεταβαίνουμε στην κατάσταση W_2' στην οποία μετακινούμαστε προς το τέλος του clause αυτού, ξεσημαδεύοντας όλους τους σημαδεμένους χαρακτήρες που θα συναντήσουμε και μόλις βρεθούμε στο τέλος του clause, μεταβαίνουμε στην κατάσταση V ώστε να συνεχιστεί η επαναληπτική σύγκριση του τρέχοντός μας clause ε όλα τα προς τα αριστερά σημαδεμένα.

Τώρα, μπορούμε να συνεχίσουμε μα την περιγραφή της καταστάσεως W' στην οποία μεταβαίνουμε όταν η σύγκριση των δύο αντίστοιχων χαρακτήρων των επιμέρους clauses *πετύχει*. Στην περίπτωση αυτή, είναι απαραίτητο να

μετακινήσουμε τον δρομέα προς τα δεξιά, προς το τρέχον μας clause, ώστε να βρούμε και να αποθηκεύσουμε στην κατάστασή μας τον επόμενο για έλεγχο χαρακτήρα και ύστερα να μεταβούμε στην κατάλληλη κατάσταση η οποία θα μας συγκρίνει όπως και πριν, τους δυο αντίστοιχους χαρακτήρες μας:

$$
\begin{aligned}
W'x' &\rightarrow x'W' \\
W'\bar{x}' &\rightarrow \bar{x}'W' \\
W'0' &\rightarrow 0'W' \\
W'1' &\rightarrow 1'W' \\
W',' &\rightarrow ,'W' \\
W'\tilde{\}} &\rightarrow \tilde{\}}\tilde{W} \\
W'\tilde{]} &\rightarrow \tilde{]}\tilde{W} \\
\tilde{W}\{ &\rightarrow \}\tilde{W} \\
\tilde{W}[&\rightarrow]\tilde{W} \\
\tilde{W}\} &\rightarrow \}\tilde{W} \\
\tilde{W}] &\rightarrow]\tilde{W} \\
\tilde{W}0 &\rightarrow 0\tilde{W} \\
\tilde{W}1 &\rightarrow 1\tilde{W} \\
\tilde{W}, &\rightarrow ,\tilde{W} \\
\tilde{W}\sqcup &\rightarrow \sqcup\tilde{W} \\
\tilde{W}0' &\rightarrow V0' \\
\tilde{W}1' &\rightarrow V1' \\
\tilde{W},' &\rightarrow V,' \\
\tilde{W}x' &\rightarrow Vx' \\
\tilde{W}\bar{x}' &\rightarrow V\bar{x}'
\end{aligned}
$$

Η λειτουργία των παραπάνω κανόνων είναι προφανής: αρχικά, μετακινούμενοι προς τα δεξιά, αγνοούμε όλους τους χαρακτήρες του clause στο οποίο βρισκόμαστε, και οι οποίοι έχουν ήδη συγκριθεί με τους αντίστοιχους του τρέχοντος clause, *χωρίς να τους ξεσημαδέψουμε*. Αυτό γιατί, η σύγκριση θα συνεχιτεί, οπότε είναι αναγκαίο να έχουμε σημαδεμένο τον τελευταίο χαρακτήρα που έχουμε ήδη συγκρίνει καθώς στο επόμεν βήμα θα συγκρίνουμ με τον επόμενο προς τα αριστερά αυτού. Επειτα, αγνοούμε πάλι όλα τα ενδιάμεσα clauses, μέσω της καταστάσεως $\tilde{W}$, μέχρι να βρούμε το τρέχον clause μέσω του πρώτου σημαδεμένου χαρακτήρα που θα συναντήσουμε, οπότε και

μεταβαίνουμε στην κατάσταση V ώστε να συνεχιστεί η σύγκριση μεταξύ των δύο clauses.

Τελείωνουμε την περιγραφή των τριών παραπάνω καταστάσεων με την W'', κατάσταση στην οποία μεταβαίνουμε όταν δούμε ότι δεν υπάρχουν επιπλέον σημαδεμένα, και άρα υποψήφια προς σύγκριση, clauses. Στην περίπτωση αυτή, είναι αναγκαίο να μεταφερθούμε προς το τρέχον μας clause, να αλλάξουμε, εάν κάτι τέτοιο είναι φυσικά αναγκαίο, τους διακριτικούς χαρακτήρες του τρέχοντος clause (από μη ικανοποιήσιμο δηλαδή σε ικανοποιήσιμο) και έπειτα να συνεχίσουμε την παραπάνω επαναληπτική διαδικασία τοποθέτησης του τρέχοντος αληθούς literal σε άλλα προς τα δεξιά clauses:

$$
\begin{aligned}
W''x &\rightarrow xW'' \\
W''\bar{x} &\rightarrow \bar{x}W'' \\
W''0 &\rightarrow 0W'' \\
W''1 &\rightarrow 1W'' \\
W'', &\rightarrow ,W'' \\
W''\{ &\rightarrow \}W'' \\
W''[&\rightarrow]W'' \\
W''\} &\rightarrow \}W'' \\
W''] &\rightarrow]W'' \\
W''x' &\rightarrow Xx' \\
W''\bar{x}' &\rightarrow X\bar{x}' \\
W''0' &\rightarrow X0' \\
W''1' &\rightarrow X1' \\
W'',' &\rightarrow X,'
\end{aligned}
$$

Η εξήγηση των παραπάνω κανόνων είναι η εξής: αρχικά αγνούμε όλα τα ενδιάμεσα clauses μέχρι να συναντήσουμε τον πρώτο σημαδεμένο χαρακτήρα, γεγονός με το οποίο καταλαβαίνουμε ότι βρισκόμαστε στο τρέχον μας clause, οπότε και μεταβαίνουμε στην κατάσταση X. Αυτό που επιθυμούμε να κάνουμε μέσω αυτής της καταστάσεως είναι να αλλάξουμε, εάν κάτι τέτοιο είναι αναγκαίο, τους χαρακτήρες που περικλείουν το τρέχομ μας clause και ύστερα να συνεχίσουμε τη διαδικασία τοποθέτησης του τρέχοντος αληθούς μας literal σε επόμενα clauses:

$$
\begin{aligned}
0X &\rightarrow X0 \\
1X &\rightarrow X1 \\
xX &\rightarrow Xx \\
\bar{x}X &\rightarrow X\bar{x} \\
,X &\rightarrow X, \\
\{X &\rightarrow \tilde{[}X' \\
[X &\rightarrow \tilde{[}X'
\end{aligned}
$$

Εάν, μεταβαίνοντας προς τα αριστερά, συναντήσουμε χαρακτήρα μη ικανοποιήσιμου clause αυτομάτως τον μετατρέπουμε σε αυτόν που υποδηλώνει ικανοποιήσιμο clause. Επειτα, μέσω της καταστάσεως X', μεταβαίνουμε προς τα δεξιά, ξεσημαδεύοντας τους τυχών σημαδεμένους χαρακτήρες:

$$
\begin{aligned}
X'x &\rightarrow xX' \\
X'\bar{x} &\rightarrow \bar{x}X' \\
X'0 &\rightarrow 0X' \\
X'1 &\rightarrow 1X' \\
X', &\rightarrow ,X' \\
X'x' &\rightarrow xX' \\
X'\bar{x}' &\rightarrow \bar{x}X' \\
X'0' &\rightarrow 0X' \\
X'1' &\rightarrow 1X' \\
X',' &\rightarrow ,X' \\
X'\} &\rightarrow \tilde{]}J' \\
X'] &\rightarrow \tilde{]}J'
\end{aligned}
$$

Δηλαδή, αφού ξεσημαδέψουμε όλους τους ενδιάμεσους χαρακτήρες, αλλάζουμε εάν είναι αναγκαίο τον δεξιό τερματικό χαρακτήρα του clause και μεταβαίνουμε στην κατάσταση J' την οποία έχουμε περιγάψει σε προηγούμενο βήμα και λειτουργία της οποίας είναι να μετακινήσει τον δρομέα προς το πρώτο μη σημαδεμένο clause προς τα δεξιά.

Η πιο σημαντική περίπτωση, την οποία δεν αναφέραμε πιο πάνω, είναι η εξής: να βρεθούν ίδια τα δύο clauses που συγκρίνουμε. Κατ'αρχήν, είναι απαραίτητο να πούμε πότε συμβαίνει αυτό. Αυτό που κάνει η παραπάνω διαδικασία, είναι να βρίσκει τον πρώτο μη σημαδεμένο χαρακτήρα από τον τρέχον clause και να τον συγκρίνει με τον αντίστοιχο του προς σύγκριση clause. Οταν ψάχουμε να βρούμε τον τρέχομντα χαρακτήρα προς σύγκριση στο clause μας, ενδέχεται να έχουμε φτάσει στο αριστερό άκρο (δηλαδή στην αρχή) του clause. Οταν συμβεί αυτό, ξέρουμε ότι όλοι οι προηγούμενοι χαρακτήρες του clause μας ταυτίζονται με τους αντίστοιχους του έτερου clause. Και επειδή ο τελευταίος χαρακτήρας που συγκρίναμε ήταν είτε x είτε $\bar{x}$, ξέρουμε ότι δεν υπάρχει στο προς σύγκριση clause άλλος χαρακτήρας προς τα αριστερά του τελευταίου που συγκρίναμε, άρα τα δύο clauses ταυτίζονται. Πρώτη προτεραιότητα είναι να περιγράψουμε αυτή τη συνθήκη ταύτισης και της μετάβασης στην αντίστοιχη κατάλληλη κατάστασης (ας θυμηθούμε ότι η κατάσταση η οποία ψάχνει για τον πρώτο μη σημαδεμένο χαρακτήρα, και άρα η κατάσταση που θα αντιληφθεί το τέλος του clause μέσω της έλλειψης μη σημαδεμένων χαρακτήρων, είναι η V):

$$\begin{aligned} \{V &\rightarrow \{\alpha \\ [V &\rightarrow [\alpha \end{aligned}$$

Τώρα, μέσω της καταστάσεως α, οι λειτουργίες που πρέπει να επιτελέσουμε είναι οι κάτωθι: αρχικά να ξεσημαδέψουμε όλους τους χαρακτήρες του τρέχοντος clause. Επειτα να "σβήσουμε" (αντικαθιστώντας βέβαια τους χαρακτήρες με το $\sqcup$) το τρέχον αληθές literal που γράψαμε στη τρίτη συνιστώσα του τρέχοντος clause και, τέλος, να ξεσημαδέψουμε τους χαρακτήρες του έτερου clause σύγκρισης.

Αρχίζουμε από την πρώτη λειτουργία (το σημάδεμα του τελευταίου χαρακτήρα θα μας βοηθήσει στο επόμενο βήμα για να βρίσκουμε εύκολα το τρέχον clause):

$$
\begin{aligned}
\alpha x' &\rightarrow x\alpha \\
\alpha \bar{x}' &\rightarrow \bar{x}\alpha \\
\alpha 0' &\rightarrow 0\alpha \\
\alpha 1' &\rightarrow 1\alpha \\
\alpha,' &\rightarrow ,\alpha \\
\alpha\} &\rightarrow \alpha_1\dot{\}} \\
\alpha] &\rightarrow \alpha_1\dot{]}
\end{aligned}
$$

Στην κατάσταση α_1 αρχίζουμε το "σβήσιμο" των χαρακτήρων της τρίτης συνιστώσας του τρέχοντow clause και της ταυτόχρονης αντικατάστασή τους από τον χαρακτήρα $\sqcup$:

$$
\begin{aligned}
0\alpha_1 &\rightarrow \alpha_1 \sqcup \\
1\alpha_1 &\rightarrow \alpha_1 \sqcup \\
x\alpha_1 &\rightarrow \alpha_1 \sqcup \\
\bar{x}\alpha_1 &\rightarrow \alpha_1 \sqcup \\
,\alpha_1 &\rightarrow \alpha_2,
\end{aligned}
$$

Τέλος, στην κατάσταση α_2, θέλουμε να μετακινήσουμε το δρομέα μέχρι το έτερο clause σύγκρισης ώστε να ξεσημαδέψουμε όλους τους σημαδεμένους χαακτήρες αυτού:

$$
\begin{aligned}
0\alpha_2 &\rightarrow \alpha_2 0\\
1\alpha_2 &\rightarrow \alpha_2 1\\
\sqcup\alpha_2 &\rightarrow \alpha_2 \sqcup\\
x\alpha_2 &\rightarrow \alpha_2 x\\
\bar{x}\alpha_2 &\rightarrow \alpha_2 \bar{x}\\
,\alpha_2 &\rightarrow \alpha_2 ,\\
\{\alpha_2 &\rightarrow \alpha_2 \{\\
[\alpha_2 &\rightarrow \alpha_2 [\\
\tilde{\{}\alpha_2 &\rightarrow \alpha_3 \tilde{\{}\\
\tilde{[}\alpha_2 &\rightarrow \alpha_2 \tilde{[}\\
\}\alpha_2 &\rightarrow \alpha_2 \}\\
]\alpha_2 &\rightarrow \alpha_2]\\
\tilde{\}}\alpha_2 &\rightarrow \alpha_2 \tilde{\}}\\
\tilde{]}\alpha_2 &\rightarrow \alpha_2 \tilde{]}\\
0'\alpha_2 &\rightarrow \alpha_2 0\\
1'\alpha_2 &\rightarrow \alpha_2 1\\
x'\alpha_2 &\rightarrow \alpha_2 x\\
\bar{x}'\alpha_2 &\rightarrow \alpha_2 \bar{x}\\
,'\alpha_2 &\rightarrow \alpha_2 ,
\end{aligned}
$$

Στην κατάσταση α_3 θα βρεθούμε μόλις συναντήσουμε τον αριστερό τερματικό χαρακτήρα του clause προς σύγκριση. Ξέρουμε ότι το πρώτο ικανοποιήσιμο clause προς τα αριστερά που θα σηναντίσουμε, είναι το τρέχον clause σύγκρισης, άρα μόλις βρούμε το αριστερό του άκρο, σταματάμε τη διαδικασία ξεσημαδέματος των χαρακτήρων αυτού. Τώρα, είναι απαραίτητο να συνεχίσουμε τη διαδικασία τοποθέτησης του τρέχοντος θετικού literal σε άλλα clauses που έχουν χώρο προς τα δεξιά αυτού που θεωρήσαμε τρέχον στο τρέχον βήμα μας (και το οποίο έχουμε σημαδέψει κατάλληλα ώστε να το βρίσκουμε εύκολα):

$$
\begin{aligned}
\alpha_3 x &\rightarrow x\alpha_3 \\
\alpha_3 \bar{x} &\rightarrow \bar{x}\alpha_3 \\
\alpha_3 0 &\rightarrow 0\alpha_3 \\
\alpha_3 1 &\rightarrow 1\alpha_3 \\
\alpha_3, &\rightarrow ,\alpha_3 \\
\alpha_3\{ &\rightarrow \{\alpha_3 \\
\alpha_3[&\rightarrow [\alpha_3 \\
\alpha_3\} &\rightarrow \}\alpha_3 \\
\alpha_3] &\rightarrow]\alpha_3 \\
\alpha_3\dot{\}} &\rightarrow \}J' \\
\alpha_3\dot{]} &\rightarrow]J'
\end{aligned}
$$

Κεφαλαιο 10

NP Γλωσσες με Πολυωνυμικα Πιστοποιητικα

10.1 Εισαγωγή

Οπως είχαμε αποδείξει στο κεφάλαιο 3 σχετικά με τη σχέση των κλάσεων **NP** και **NSPACE**(n), αυτές οι κλάσεις είναι διαφορεικές, χωρίς όμως να έχουμε επιπλέον στοιχεία σχετικά με τη περαιτέρω σχέση αυτών. Στα προηγούμενα κεφάλαια δείξαμε ότι συγκεκεριμένα προβλήματα που ανήκουν στην κλάση πολυπλοκότητας **NP**, και μάλιστα είναι **NP**-πλήρη δηλαδή τα δυσκολότερα ως προς το βαθμό επιλυσημότητας μέσα στην κλάση **NP**, μπορούν να παραχθούν από γραμματικές με συμφραζόμενα και άρα ανήκουν στην κλάση πολυπλοκότητας **NSPACE**(n).

Αυτό ίσως να μας δίνει μια ένδειξη σχετικά με την ακριβή σχέση των δύο αυτών κλάσεων πολυπλοκότητας: αφού τα δυσκολότερα προβλήματα που ανήκουν στην κλάση **NP** παράγονται από γραμματικές με συμφραζόμενα, είναι λογικό να αναμένουμε ότι *όλα* τα προβλήματα της κλάσης **NP** μπορούν να παραχθούν από ανίστοιχους κανόνες και, κατ'επέκταση, να ανήκουν στην κλάση πολυπλοκότητας **NSPACE**(n). Δηλαδή με άλλα λόγια, αυτά τα αποτελέσματα ισχυροποιούν τη πιθανότητα να ισχύει ότι **NP** $\subseteq$ **NSPACE**(n).

10.2 Προβλήματα στο NP με υπερ-γραμμικά πιστοποιητικά

Στην ενότητα αυτή θα παρουσιάσουμε προβλήματα τα οποία φαίνεται να καταρρίπτουν την παραπάνω εικασία. Πιο συγκεκριμένα, θα ορίσουμε ένα πρόβλημα το οποίο από την μία πλευρά θα ανήκει στην κλάση **NP**, από την άλλη πλευρά όμως φαίνεται να χρειάζεται *πολυωνικού χώρου πιστοποιητικό* για να επιλυθεί μη ντετερμινιστικά σ επολυωνυμικό χρόνο. Η βασική ιδέα είναι, δοθέντος ενός στιγμιοτύπου, να *μεγαλώσουμε πολυωνυμικά* το στιγμιότυπο ώστε αυτόματα να μεγαλώσει και το αντίστοιχο πιστοποιητικό.

Το πρόβλημα που θα ορίσουμε θα αφορά γραφήματα. Τον τρόπο με τον οποίο θα μεγαλώνουμε τα στιγμιότυπα ενός γραφήματος τον περιγράφη η παρακάτω κατασκευή η οποία ορίζει την *επέκταση* ενός γραφήματος:

Ορισμός 16 *Εστω $G = (V, E)$ ένα απλό και μη κατευθυνόμενο γράφημα. Ως* **επέκταση** *του γραφήματος G, που θα την συμβολίζουμε ως $E\pi(G)$, ορίζουμε ένα νέο γράφημα G' το οποίο προκύπτει από το αρχικό γράφημα ως εξής:*

- α *Το σύνολο των κορυφών του γραφήματος $G' = E\pi(G)$ αποτελείται από το σύνολο των διατεταγμένων τριάδων των κορυφών του G. Δηλαδή οι κορυφές του G' αποτελούνται από όλες τις δυνατές τριάδες της εξής μορφής: $(x_i, x_j, x_k) : \ x_i, x_j, x_k \in V(G)$.*

- β *Δύο κορυφές του νέου γραφήματος (x_i, x_j, x_k) και (y_q, y_p, y_r) ενώνονται με ακμή στο νέο γράφημα εάν και μόνον εάν υπάρχει ακριβώς μία από τις επόμενες ακμές στο αρχικό γράφημα: $\{x_i, y_q\}, \{x_j, y_p\}, \{x_k, y_r\}$. Εάν καμία ή περισσότερες από μία από αυτές τις ακμές υπάρχουν στο αρχικό γράφημα, τότε οι αντίστοιχες κορυφές δεν ενώνονται στο νέο γράφημα G'.*

Με βάση την παραπάνω κατασκευή, ορίζουμε το παρακάτω πρόβλημα:

Ορισμός 17 *Δοθέντος ενός μη κατευθυνόμενου γραφήματος $G = (V, E)$, αληθεύει ότι η επέκταση $E\pi(G)$ υτού του γραφήματος έχει μονοπάτι Χάμιλτον;*

Το παραπάνω πρόβλημα ανήκει προφανώς στην κλάση **NP**. Δοθέντος ενός γραφήματος $G = (V, E)$, κατασκευάζουμε σε πολυωνυμικό χρόνο την επέκτασή του κια κατόπιν "μαντεύουμε" ένα μονοπάτι Χάμιλτον για το νέο αυτό γράφημα.

Αυτό που δεν είναι καθόλου προφανές, είναι το πώς μπορεί να "συμπυκνωθεί" το πιστοποιητικό του μονοπατιού Χάμιλτον του νέου γραφήματος ώστε να γίνει γραμμικό ως προς το αρχικό γράφημα. Το νέο αυτό γράφημα θα

περιέχει $|V(G)|^3$ κορυφές και άρα εάν έχει μονοπάτι Χάμιλτον, αυτό θα αναπαρίσταται από $|V(G)|^3 - 1$ κορυφές. Αυτόματα, το νέο πιστοποιητικό είναι πολυωνυμικά μεγαλύτερο από το μέγεθος του αρχικού γραφήματος το οποίο είναι $\mathcal{O}(|V(G)|^2)$.

Βιβλιογραφια

[1] Balcázar, J.L. , Diáz, J. and Gabarró,J. *Structural Complexity*, EATCS Monographs on Theoretical Computer Science. Springer Verlag, 1988(I) and 1990(II).

[2] Chomsky N., Systems of Syntactic Analysis, *J. Symb. Log.* 18(3): 242-256 (1953)

[3] Chomsky N., Three Models for the Description of Languages, *IRE Transactions on Information Theory* 2(3) (1956), 113-124

[4] Chomsky N., Miller G.A., Finite State Languages, *Information and Control* 1(2): 91-112 (1958)

[5] Chomsky N., A Note on Phrase Structure Grammars, *Information and Control* 2(4): 393-395 (1959)

[6] Chomsky N., On Certain Formal Properties of Grammars, *Information and Control* 2(2): 137-167 (1959)

[7] Davis, M.D, Sigal R., and Weyuker E.J., *Computability, Complexity and Languages: Fundamentals of Theoretical Computer Science*, Academic Press, Harcourt, Brace & Company, 1983.

[8] Garey, M. R. and Johnson, D. S., *Computers and Intractability: A Guide to the Theory of NP-completness*, W.H. Freeman, 1979.

[9] Harrison M.A., *Introduction to Formal Language Theory*, Reading, Mass.: Addison-Wesley, 1978

[10] Hopcroft J. E. and Ullman J. D., *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley 1979

[11] Immerman N., Nondeterministic Space is Closed Under Complementation , *SIAM J. Comput.* 17(5) (1989) 935–938.

[12] Kuroda S.Y., Classes of Languages and Linear-Bounded Automata, *Information and Control* 7(2): 207-223 (1964)

[13] Lewis, H. and Papadimitrioy, C. H., *Elements of the Theory of Computation*, Prentice-Hall, 1981.

[14] Papadimitriou C. H., *Computational Complexity*, Reading,Mass.: Addison Wesley, 1994.

[15] Szelepcsényi R., The method of forcing for nondeterministic automata , *Bulletin of the EATCS* . **33** (1987) 96–99.

Βιογραφικο

Ο Σταμούλης Γεώργιος γεννήθηκε το 1983 στην Αθήνα. Αποφοίτησε από το 4ο Ενιαίο Λύκειο Αγρινίου το 2001 και την ίδια χρονιά εισήχθη στο προπτυχιακό πρόγραμμα σπουδών του Τμήματος Πληροφορικής του Πανεπιστημίου Ιωαννίνων. Ολοκλήρωσε τις προπτυχιακές του σπουδές τον Φλεβάρη του 2006 και έπειτα από τον Φεβρουάριο του 2006 παρακολουθεί το μεταπτυχιακό πρόγραμμα σπουδών του ίδιου Τμήματος. Τα ευρευνητικά του ενδιαφέροντα εστιάζονται στην Υπολογιστική Πολυπλοκότητα, και Τυπική Θεωρία Γλωσσών και στους Προσεγγιστικούς Αλγορίθμους.