
HY-280

«ΘΕΩΡΙΑ ΥΠΟΛΟΓΙΣΜΟΥ»

θεμελιακές έννοιες της επιστήμης του υπολογισμού

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

Γεώργιος Φρ. Γεωργακόπουλος

Μέρος Α΄

Εισαγωγή, και η βασική θεωρία των πεπερασμένων αυτομάτων.

1^ο Γενική εισαγωγή και λίγα ιστορικά πρόσωπα.

Η μελέτη μιας επιστημονικής/μαθηματικής (και όχι μόνον) περιοχής, βοηθά στην κατανόηση της ιστορίας της, ισχύει όμως και το αντίστροφο: η ιστορία μιας περιοχής φωτίζει και την ίδια. Η ιστορία της θεωρίας του υπολογισμού όμως δεν έχει γραφεί ακόμα. Η σχετική διεθνής βιβλιογραφία σε αυτό το θέμα είναι ακόμα αποσπασματική, ίσως ακόμα και ρηχή. Δεν έχουμε λοιπόν να προσφέρουμε μια καλή σύνοψη του τί συνέβη και φτάσαμε να έχουμε μια θεωρία (και μια τεχνολογία, πια) του υπολογισμού. Αντ' αυτού θα απαριθμήσουμε μια σειρά επεισοδίων σχετικά με αυτή την εξέλιξη, μέσω μιας σειράς από εξέχουσες προσωπικότητες. Ο αναγνώστης έχει, πια, την δυνατότητα να τις αναζητήσει στο διαδίκτυο (βλ. wikipedia.org), και να βρει πλήθος από πληροφορίες, σχετικά με την ιστορική περιπέτεια του «υπολογισμού».

Αριστοτέλης, ο Σταγειρίτης	384-332 π.Χ. , Έλληνας, Αρχαία Μακεδονία και Αθήνα. Μεταξύ αναριθμητων συνεισφορών του, ο Αριστοτέλης έγραψε και ένα εγχειρίδιο για την λογική. Εκινείτο στο επίπεδο του «προτασιακού λογισμού», και όπως ο ίδιος έλεγε «περί της συμπερασματικής λογικής δεν εγνώριζε καμμιά προηγούμενη πηγή» (αν και κάποιοι είχαν, προηγουμένως, ασχοληθεί με τους κανόνες όχι της σκέψης αλλά της γλώσσας, και είχαν μιλήσει για το «συντακτικό» της). Η «Αριστοτελική Λογική» έμελε να παραμείνει σε εκείνο το αρχικό επίπεδο για πάνω από 2000 χρόνια.
Στωικοί	circa 300 π.Χ. - 200 μ.Χ. , Έλληνες (και όχι μόνο). Η σχολή των Στωικών είχε την δική της εκδοχή για την «λογική», η οποία έδινε θεμελιώδη έμφαση στη σχέση «συνεπάγεται» ($p \rightarrow q$ αν και τότε δεν χρησιμοποιούσαν αυτόν τον συμβολισμό). Η εκδοχή τους νόμιζαν ότι διέφερε από εκείνη του Αριστοτέλη, αυτό όμως συνέβαινε μόνον επιφανειακά.
Ευκλείδης, ο Αλεξανδρινός	Εποχή Πτολεμαίου (323-283 π.Χ.), Έλληνας, Αλεξάνδρεια Αιγύπτου. Έφορος της βιβλιοθήκης της Αλεξανδρείας, και κατά γενική ομολογία ο κρισιμότερος μαθηματικός όλων των εποχών. Με τα «Στοιχεία» του (circa 300 π.Χ.), όχι μόνο θεμελίωσε την γεωμετρία, αλλά και την ίδια την μαθηματική μέθοδο: σαφείς ορισμοί και ακριβείς αποδείξεις . Τα «Στοιχεία» ¹ περιλαμβάνουν και τον περίφημο «αλγόριθμο του Ευκλείδη» (για την εύρεση του μέγιστου κοινού διαιρέτη), έναν από τους πρώτους ρητά διατυπωμένους αλγορίθμους (μαζί με την απόδειξη ότι είναι σωστός!) – αν και ούτε εκείνη την εποχή τον έβλεπαν ως αλγόριθμο, ούτε και ποτέ (προ 20 ^{ου} αιώνα) τον ονόμαζαν έτσι...
Μεσαιωνική φιλοσοφία	Αν και επί σειρά αιώνων (sic) οι φιλόσοφοι και οι μαθηματικοί διαπίστωναν προβλήματα και ελλείψεις στα διαθέσιμα «εργαλεία» της λογικής, δεν επέτυχαν καμμιά ουσιώδη συνεισφορά.
George Boole	1815 -1864, Βρετανός. Το 1847 με το βιβλίο του <i>The laws of thought</i> επανέφερε το θέμα της λογικής, ακριβέστερα του προτασιακού λογισμού, έδειξε την ενότητα των απόψεων του Αριστοτέλη και των Στωικών, και έδειξε τις δυνατότητες αλγεβρικού χειρισμού της λογικής. Η λογική δεν ήταν πια μόνον <i>όργανο</i> των μαθηματικών, αλλά και <i>αντικείμενο</i> των μαθηματικών.
Gottlob Frege	1848-1925, Γερμανός. Φυσιογνωμία-κλειδί. Το 1879 με το βιβλίο του «Εννοιολογία» (!) επεξέτεινε ρητά και συνειδητά – επί τέλους – την προτασιακή λογική, περιγράφοντας τον <i>κατηγορηματικό λογισμό</i> , εισάγοντας τους λογικούς ποσοδείκτες (\forall και \exists), και δείχνοντας το πώς οι αποδείξεις μπορούν να γίνουν μέσω συμβολικών κατ' ουσίαν χειρισμών. Ο κατηγορηματικός λογισμός είναι σχεδόν αποκλειστικά το

¹ Εκτιμάται ότι από όλα τα βιβλία της ιστορίας, μόνον η Βίβλος έχει γνωρίσει περισσότερες εκδόσεις από ότι τα «Στοιχεία» του Ευκλείδη!

	<p>σύστημα λογικής που χρησιμοποιούμε τώρα στα μαθηματικά.</p> <p>Το 1893 (I τόμος) και 1903 (II τόμος) με το βιβλίο του <i>Βασικοί νόμοι της αριθμητικής</i> προσπάθησε να θεμελιώσει τους φυσικούς αριθμούς επί της λογικής. Μια κρίσιμη λογική αδυναμία στον II τόμο, άναψε την φωτιά της αναζήτησης επαρκώς ευσταθών θεμελίων για τα μαθηματικά, και οδήγησε στην συστηματική ανάλυση της μεθόδου των «αποδείξεων» και στη συνέχεια των «υπολογισμών».</p>
Giuseppe Peano	<p>1858-1932, Ιταλία.</p> <p>Το 1889 δημοσίευσε ένα συλλεκτικό έργο (με εργασίες από το 1860) επί της «αξιοματικής θεμελίωσης» των φυσικών αριθμών, για να δείξει ότι λίγες αρχικές παραδοχές αρκούν για να εξαχθεί αποδεικτικά όλη η θεωρία των αριθμών. Τα σχετικά αξιώματα καλούνται ακόμα «αξιώματα του Peano».</p>
Bertrand Russel	<p>1872-1970, Βρετανός.</p> <p>Το 1900 επεσήμανε το (περίφημο έκτοτε) «παράδοξο του Russel»², μια καίρια αντίφαση στο υπό δημοσίευση έργο του Frege (τί τύχη...). Ο ίδιος εργάστηκε επί μακρά σειρά ετών στην «αξιοματική θεμελίωση» των μαθηματικών, εμπεδώνοντας ότι ο λογισμός του Frege ήταν όντως, με σωστή χρήση, το κατάλληλο εργαλείο.</p>
David Hilbert	<p>1862-1943, Γερμανός.</p> <p>Εμβληματική φυσιογνωμία των μαθηματικών του 20^{ου} αιώνα. Όχι μόνον έκανε πολλά και βαθιά μαθηματικά, αλλά έκανε τα ίδια τα μαθηματικά του 20^{ου} αιώνα, προσδιορίζοντας τόσο τα μείζονα προβλήματα τους, αλλά και τις μεθόδους τους: ο Hilbert βάρυνε και στερέωσε την χρήση της ευκλείδειας αξιοματικής μεθόδου. Το σημείο κλειδί όσον αφορά στην ιστορία του υπολογισμού είναι ότι τις 10ετίες 1900-1920 υποστήριζε θερμά την «πεπερασμένη» (<i>finitary</i>) εκδοχή των μαθηματικών εργαλείων ότι δηλαδή με πεπερασμένες και διακριτές λογικές μεθόδους μπορούμε και θα είμαστε σε θέση να κρίνουμε την αλήθεια ή το ψεύδος κάθε δυνατής μαθηματικής πρότασης. Ως προς αυτό, διαψεύστηκε από έναν μαθητή του, τον K. Gödel.</p>
Kurt Gödel	<p>1906-1978, Αυστριακός.</p> <p>Εργάστηκε στη λογική και έδωσε τα θεωρήματα της «πληρότητας» (του κατηγορηματικού λογισμού <i>a-la</i> Frege), και της «μη-πληρότητας» (της αριθμητικής, διδ. διατριβή, 1931). Το 2^ο θεωρείται το πιο «shocking» θεώρημα του 20^{ου} αιώνα, αν όχι όλων των μαθηματικών μέχρι τώρα. (Εξ άλλου τα μαθηματικά του 20^{ου} αιώνα είναι το 95% όλων των μαθηματικών μέχρι τώρα...). Το θεώρημα αυτό λέει ότι η αριθμοθεωρία περιέχει προτάσεις που είναι μεν αληθινές, αλλά δεν μπορούμε να τις αποδείξουμε (με κανένα «εύλογο» σύστημα αξιωμάτων). Έδειξε λοιπόν με αυτό τον τρόπο ότι η αξιοματική-αποδεικτική μέθοδος έχει κάποιου είδους όρια, και επομένως έθετε το ζωτικό και «επείγον» ζήτημα, του ποιά είναι αυτά.</p>
Alan Turing	<p>1912-1954, Βρετανός.</p> <p>Ο «πατέρας» της θεωρίας υπολογισμού. Μετά το θεώρημα μη-πληρότητας του Gödel ένοιωσε το εξής ερώτημα: «ΟΚ – οι αποδείξεις έχουν όρια, αλλά μήπως άλλοι συμβολικοί χειρισμοί είναι σε θέση να τα καταφέρουν καλύτερα;» Για την απάντησή του έπρεπε να εξετάσει τον ισχυρότερο κατά το δυνατόν τύπο μηχανής ικανής για συμβολικούς χειρισμούς – την λεγόμενη πλέον μηχανή Turing (1936).</p> <p>Ο Turing εξήγησε γιατί αυτός ο τύπος μηχανής που εισηγήθηκε είναι ο ισχυρότερος στον οποίο μπορούμε να ελπίζουμε, αλλά ότι ακόμα και αυτός δεν είναι σε θέση να επιλύσει όλα τα προβλήματα που θα θέλαμε να λύσουμε</p>

² Το «παράδοξο του Russel»: αν ένα σύνολο θεωρείται «ομαλό» αν δεν περιέχει τον εαυτό του, τότε το σύνολο των ομαλών συνόλων τί είναι: ομαλό ή ανώμαλο; Δεν μπορεί να είναι τίποτε από τα δύο...

	<p>«αλγοριθμικά».</p> <p>Κατά την διάρκεια του πολέμου εργάστηκε σε ένα είδος υπολογιστή, εξειδικευμένου στην επίλυση κρυπταναλυτικών προβλημάτων. Δεν ήταν ακριβώς ένας υπολογιστής όπως τον θέλουμε τώρα, αλλά ήταν αρκετός για να δείξει ότι ο υπολογισμός ήταν – μεταξύ άλλων – και ένα όπλο.</p>
John von Neuman	<p>1903-1957, Ούγγρος (και τελικά πολίτης των ΗΠΑ).</p> <p>Ένας εξαιρετικά ευφυής μαθηματικός, ο οποίος δεν άφησε κλάδο στον οποίο να μην προσφέρει κατά καιρίο τρόπο: ανάλυση, άλγεβρα, κβαντική φυσική, θεωρία παιγνίων, θεωρία υπολογισμού, κ.ά. Αμέσως μετά τον 2^ο παγκόσμιο πόλεμο, το 1945, ενεπλάκη στην ομάδα των ΗΠΑ για την κατασκευή του πρώτου «γνήσιου» υπολογιστή. Εκεί πρότεινε την υπολογιστική αρχιτεκτονική που φέρει το όνομά του, δηλαδή, της συσκευής με μία μνήμη, τόσο για το «πρόγραμμα» όσο και για τα «δεδομένα». Εν πολλοίς, αυτή την αρχιτεκτονική χρησιμοποιούμε ακόμα.</p>
Noam Chomsky	<p>1928 και εν ζωή, Αμερικανός.</p> <p>Προσωπικότητα όχι των μαθηματικών αλλά και της διεθνούς σκηνής. Ο Chomsky έγραψε πολλά για τις «τυπικές γλώσσες» την 10ετία '50-'60. Η μορφή αυτού του μαθήματος οφείλεται σε δικές του συνεισφορές, (λ.χ. στη χαρακτηριστική εργασία “<i>Three models for the description of a language</i>”, 1956).</p>
η 2^η πράξη	<p>Στη δεκαετία του 1960 τα ηνία της θεωρίας υπολογισμού ανέλαβε ένας κλάδος αυτής, η «θεωρία πολυπλοκότητας», η οποία δεν ενδιαφέρεται μόνον για το ποιά προβλήματα λύνονται υπολογιστικά, αλλά κυρίως για το πόσο «γρήγορα» ή «οικονομικά» αυτό επιτυγχάνεται.</p>

Η ιστορία (του «υπολογισμού») λοιπόν, όπως την διηγούνται τα επιτεύγματα των παραπάνω προσώπων είναι σε λίγες παραγράφους η εξής:

- Ήδη από την αρχαιότητα, (Αριστοτέλης, Ευκλείδης, Στωικοί, κ.ά), είχε επισημανθεί ότι υπάρχουν λογικοί κανόνες, και ότι βάσει αυτών μπορεί και πρέπει να ασκούνται τα μαθηματικά. Είχε ήδη επισημανθεί η πρακτική και όχι μόνον, αξία των κατασκευαστικών μεθόδων, τις οποίες ο αρχαίος κόσμος σεβόταν πολύ και διαρκώς επεδίωκε.
- Παρά ταύτα, η κατάσταση στο χώρο της λογικής και των κατασκευαστικών μαθηματικών παρέμεινε στάσιμη όσο αφορά την κατανόηση των σχετικών εννοιών για εξαιρετικά μακρύ χρονικό διάστημα, (μεσαιωνική περίοδος και όχι μόνον).
- Κατά τον 19^ο αιώνα η μελέτη της λογικής επανήλθε με μαθηματικά πια εργαλεία, και έγινα νέα καιρία, ιστορικού επιπέδου βήματα. Η αξιωματική και αποδεικτική μέθοδος του Ευκλείδη επαν-ισχυροποιήθηκε (Boole, Frege, Peano), και έγινε αποδεκτό ότι η απόδειξη όχι μόνον είναι μια «μηχανική» διαδικασία, αλλά καλύτερα που είναι τέτοια, διότι μόνον έτσι εξασφαλίζεται το αλάθητο και η βεβαιότητα για την ορθότητα της συλλογιστικής διαδικασίας (και άρα και των όποιων συμπερασμάτων).
- Επιφανείς μαθηματικοί, (όπως οι Russel και Hilbert), καταθέτουν όλη τους την εμπιστοσύνη στις «πεπερασμένες» αξιωματικές και αποδεικτικές μεθόδους, και κηρύσσουν ως πρόγραμμα των μαθηματικών την εύρεση μιας μεθόδου (διάβαζε: αλγόριθμου), για την διάγνωση της αλήθειας ή όχι, οποιουσδήποτε μαθηματικού ισχυρισμού.
- Διαπιστώθηκε γρήγορα, όμως, ότι η ορθή χρήση των λογικο-αποδεικτικών εργαλείων όχι μόνον δεν είναι προφανής (Russel), αλλά ότι ακόμα και αυτά υπόκεινται σε σοβαρούς περιορισμούς (Gödel).
- Τέλος, προκειμένου να διαπιστωθεί με τον πιο σαφή και τελεσίδικο τρόπο τί είδους κατασκευαστικές διαδικασίες έχουμε στη διάθεσή μας, και τί είδους χειρισμούς είμαστε σε θέση να κάνουμε «μηχανικά», ο Turing καταλήγει στις ομώνυμες μηχανές. Ακολουθούν και άλλοι, την ίδια εποχή (Kleene, Church, Markov, Neuman), οι οποίοι διευκρινίζουν το θέμα των «μηχανών», της «υπολογισιμότητας», των «προγραμμάτων» κττ., και ανοίγουν εφεξής το πεδίο της κατασκευής υπολογιστικών συσκευών. Και αυτές δεν άργησαν να φανούν – αλλά αυτό είναι άλλη ιστορία.

BEGRIFFSSCHRIFT,

EINE DER ARITHMETISCHEN NACHGEBILDETE



FORMELSPRACHE

DES REINEN DENKENS.

VON

DR. GOTTLÖB FREGE.

PROFESSOR DER MATHEMATIK AN DER UNIVERSITÄT JENA.

HALLE 4/8.

VERLAG VON LOUIS NEBERT.

1879.

PROCEEDINGS

OF

THE LONDON MATHEMATICAL SOCIETY.

SERIES 2.—VOL. 42.—PART 3.

CONTENTS.

	PAGE
ON THE PROJECTIVE CLASSIFICATION OF SURFACES (<i>continued</i>): LEONARD ROYB	161
ON A WARING'S PROBLEM WITH SQUARES OF LINEAR FORMS: CHAO KO	171
A MEAN VALUE THEOREM FOR ANALYTIC FUNCTIONS: J. M. WHITTAKER	186
FUNCTIONS WITH DOMINANT SIMILARITIES OF THE GENERALIZED ALGEBRAIC-LOGARITHMIC TYPE: R. WILSON	195
ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM: A. M. TURING	230

Issued November 30th, 1936.

LONDON:

PRINTED AND PUBLISHED FOR THE SOCIETY BY

C. F. HODGSON & SON, LTD.,

2 NEWTON STREET, KINGSWAY, W.C.2.

2^ο Τρία κεντρικά παραδείγματα τρόπων υπολογισμού.

Θα αρχίσουμε αυτή την εισαγωγική ξενάγηση στη θεωρία του υπολογισμού, με μια επίσκεψη στις πρώτες σχετικές θεμελιακές έννοιες. Θα χρησιμοποιήσουμε μερικά απλά και σύντομα παραδείγματα, και συγκεκριμένα τρεις συνηθισμένους τύπους υπολογισμού: τον πρώτο μάλιστα τον μαθαίνουμε σε πολύ μικρή ηλικία... Κάθε βέλος → δείχνει στην εκάστοτε επόμενη φάση του υπολογισμού.

I. «Συμβολικοί χειρισμοί»: η δεκαδική πρόσθεση.

Ας θυμηθούμε πως γίνεται μία πρόσθεση με τον κλασσικό τρόπο – εκείνο κατά τον οποίο παραθέτουμε τα ψηφία των δύο προσθετέων το ένα κάτω από το άλλο. Πάνω από τον 1^ο προσθετέο σημειώνουμε τα «κρατούμενα»:

$$\begin{array}{r}
 \text{(*)} \text{-----} \text{(*)} \\
 \begin{array}{cccccc}
 000 & & 000 & & 010 & & 100 & & \text{(κρατούμενα)} \\
 --- & & --- & & --- & & --- & & \\
 384 & \rightarrow & 384 & \rightarrow & 384 & \rightarrow & 384 & \rightarrow & 384 \quad \text{OK} \\
 663 & & 663 & & 663 & & 663 & & 663 \\
 --- + & & --- + & & --- + & & --- + & & ---- + \\
 & & & & & & & & 1047 \\
 & & & & & & & & 047 \\
 & & & & & & & & 7 \\
 & & & & & & & & 47 \\
 & & & & & & & & 047 \\
 & & & & & & & & 1047 \\
 \text{(*)} \text{-----} \text{(*)}
 \end{array}
 \end{array}$$

Για τους σκοπούς μας αποτελεί θεμελιώδη παρατήρηση ότι ο παραπάνω υπολογισμός (η άθροιση δύο αριθμών) και όσοι θα δούμε παρακάτω, γίνονται βάσει κάποιων **οδηγιών** οι οποίες μας είναι γνωστές εκ των προτέρων, και τις οποίες ακολουθούμε κατά βήμα. Συναντούμε πλήθος τρόπων διατύπωσης αυτών των οδηγιών. Εδώ θα αναφερθούμε στις οδηγίες της πρόσθεσης στη δυαδική μορφή της αντί της δεκαδικής (για συντομία), και θα προσθέσουμε το $5_{10} = 101_2$ με το $3_{10} = 011_2$ με βάση τους κανόνες που θα γράψουμε στη συνέχεια. Ως δεδομένα γράφουμε τα ψηφία των προσθετέων εναλλάξ (το $101 + 011$ γράφεται 100111), και τα σύμβολα '=' στην αρχή και 'N' και το τέλος. Οι κανόνες άθροισης έχουν ως εξής:

$$\begin{array}{r}
 \text{(*)} \text{-----} \text{(*)} \\
 \begin{array}{lll}
 00N \rightarrow N0 & 00C \rightarrow N0 & (N = \text{όχι κρατούμενο,} \\
 01N \rightarrow N1 & 01C \rightarrow C0 & C = \text{έχω κρατούμενο}) \\
 10N \rightarrow N1 & 10C \rightarrow C0 & \\
 11N \rightarrow C0 & 11C \rightarrow C1 & \\
 =N \rightarrow = & =C \rightarrow =1 &
 \end{array} \\
 \text{(*)} \text{-----} \text{(*)}
 \end{array}$$

Ακολουθώντας αυτούς τους κανόνες έχουμε τα εξής βήματα για τη πρόσθεση $5_{10} + 3_{10} = 1000_2$:

$$\begin{array}{r}
 = \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad N \\
 = \quad 1 \quad 0 \quad 0 \quad 1 \quad C \quad 0 \\
 = \quad 1 \quad 0 \quad C \quad 0 \quad 0 \\
 = \quad C \quad 0 \quad 0 \quad 0 \\
 = \quad 1 \quad 0 \quad 0 \quad 0
 \end{array}$$

$1000_2 = 8_{10}$
 (τα καταφέραμε...)
 2^{ος} κανόνας
 στη 2^η στήλη

Σχήμα: Μια όψη της δυαδικής πρόσθεσης.

Αυτό που θέλουμε να κρατήσουμε από το παραπάνω σύντομο παράδειγμα, είναι ότι σε αυτόν τον τύπο «υπολογισμού» **χειριζόμαστε σύμβολα**, τα οποία βάσει οδηγιών απλώς είτε διαβάζουμε, είτε γράφουμε είτε σβήνουμε.

II. «Συναρτησιακοί χειρισμοί»: ο υπολογισμός ενός παραγοντικού.

Η σειρά των σταδίων του υπολογισμού του $N!$ (συνάρτηση «παραγοντικό») για N λ.χ. ίσο με 4 έχει ως εξής:

$$\begin{aligned} & (*) \text{-----} (*) \\ & \qquad 4! \quad \rightarrow (3! \times 4) \\ & \qquad \qquad \rightarrow ((2! \times 3) \times 4) \\ & \qquad \qquad \rightarrow (((1! \times 2) \times 3) \times 4) \\ & \qquad \qquad \rightarrow (((1 \times 2) \times 3) \times 4) \\ & \qquad \qquad \rightarrow ((2 \times 3) \times 4) \\ & \qquad \qquad \rightarrow (6 \times 4) \\ & \qquad \qquad \rightarrow 24 \\ & \qquad \qquad \text{OK} \\ & (*) \text{-----} (*) \end{aligned}$$

Ποιές είναι οι σχετικές «οδηγίες»; Διότι δεν επαρκεί να γράψουμε ως οδηγίες για τον υπολογισμό του $N!$ την σχέση: $N! = 1 \times 2 \times \dots \times N-1 \times N$. Τί είδους «οδηγία» θα ήσαν τα αποσιωπητικά '...'; Ακόμα και εάν εδώ σημαίνει κάτι συγκεκριμένο σε εμάς, αυτά τα αποσιωπητικά θα σημαίνουν για εμάς κάτι διαφορετικό σε άλλο πλαίσιο, και δεν θα σημαίνουν τελικά τίποτε για μια μηχανή. Γνωρίζουμε όμως τον αναδρομικό ορισμό της συνάρτησης του παραγοντικού: συμβολίζουμε με $\Phi(v)$ την συνάρτηση του παραγοντικού, (με όρισμα το v), και γράφουμε:

$$\begin{aligned} & (*) \text{-----} (*) \\ & \qquad \Phi(v) = \begin{array}{|l} \text{εάν } v=1 \quad \text{τότε } \Phi(v) = 1 \\ \text{εάν } v>1 \quad \text{τότε } \Phi(v) = \Phi(v-1) \times v \end{array} \\ & (*) \text{-----} (*) \end{aligned}$$

Η εκτέλεση αυτών των οδηγιών γίνεται τότε:

$$\begin{aligned} & (*) \text{-----} (*) \\ & \qquad \Phi(4) = \Phi(4-1) \times 4 \qquad (v>1) \\ & \qquad = \Phi(3) \times 4 \qquad = \Phi(3-1) \times 3 \times 4 \qquad (v>1) \\ & \qquad = \Phi(2) \times 3 \times 4 \qquad = \Phi(2-1) \times 2 \times 3 \times 4 \qquad (v>1) \\ & \qquad = \Phi(1) \times 2 \times 3 \times 4 \qquad = 1 \times 2 \times 3 \times 4 \qquad (v=1) \\ & \qquad = 24 \qquad \text{OK} \\ & (*) \text{-----} (*) \end{aligned}$$

Αυτό που θέλουμε να κρατήσουμε από το παραπάνω παράδειγμα είναι ότι σε αυτόν τον «υπολογισμό» *χειριζόμαστε συναρτήσεις*, τις οποίες βάσει οδηγιών είτε εφαρμόζουμε αμέσως (αν είναι πολύ απλές, όπως λ.χ. η αφαίρεση του 1), είτε τις συνθέτουμε, είτε τις ανάγουμε σε υπολογισμό άλλων συναρτήσεων (ακόμα και αναδρομικά), κοκ.

III. «Υψηλές» γλώσσες προγραμματισμού»: η διχοτομική αναζήτηση.

Το τελευταίο παράδειγμά μας είναι ένας «αλγόριθμος» που είναι πανταχού παρών στην αλγοριθμική: η διχοτομική αναζήτηση. Έχουμε στη διάθεσή μας έναν πίνακα T με N στοιχεία (λ.χ. αριθμούς), τα οποία είναι ταξινομημένα κατ' αύξουσα σειρά. Θα θέλαμε για οποιαδήποτε στοιχείο σ να είμαστε σε θέση να διαπιστώνουμε «γρήγορα» εάν αυτό συμπεριλαμβάνεται στον πίνακά T ή όχι, (και γενικότερα εάν συμπεριλαμβάνεται στις θέσεις από L έως και R).

Το τέχνασμα της διχοτομικής αναζήτησης είναι να «κόψουμε» τον πίνακα σε δύο τμήματα: στις θέσεις από L έως M (με τα «μικρότερα» στοιχεία), και στις θέσεις από M έως R (με τα «μεγαλύτερα» στοιχεία) για κάποιο M ανάμεσα στα L και R . Στη συνέχεια διαπιστώνουμε, χάρι στη ταξινόμηση, σε ποιο από τα δύο τμήματα είναι (ή θα έπρεπε να είναι...) το αναζητούμενο στοιχείο σ , και επαναλαμβάνουμε την αναζήτηση με τον ίδιο τρόπο, περιοριζόμενοι σε αυτό το τμήμα.

Δίνουμε ένα παράδειγμα με $N=11$ στοιχεία, $L=1$, $R=N$, και $\sigma=55$. Σε κάθε φάση του «υπολογισμού» ορίζουμε το μεσαίο στοιχείο $M = (L+R)$ δια 2, και εάν το στοιχείο σ κριθεί...

- μικρότερο από το T[M], επανερχόμαστε στο τμήμα T[L=M+1 ... R].
- μεγαλύτερο από το T[M], επανερχόμαστε στο τμήμα T[L ... R=M-1].
- αλλιώς, είναι ίσο με το T[M] τότε OK, το έχουμε εντοπίσει.

L	M	R	OK	σ	T[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]	Περιέχεται
1	?	11	ψ	55	5	9	13	16	23	32	45	52	64	74	88	?
1	6	11	ψ	55	5	9	13	16	23	32	45	52	64	74	88	?
7	6	11	ψ	55	5	9	13	16	23	32	45	52	64	74	88	?
7	9	11	ψ	55	5	9	13	16	23	32	45	52	64	74	88	?
7	9	8	ψ	55	5	9	13	16	23	32	45	52	64	74	88	?
7	7	8	ψ	55	5	9	13	16	23	32	45	52	64	74	88	?
8	7	8	ψ	55	5	9	13	16	23	32	45	52	64	74	88	?
8	8	8	ψ	55	5	9	13	16	23	32	45	52	64	74	88	?
9	8	8	ψ	55	5	9	13	16	23	32	45	52	64	74	88	?
9	8	8	ψ	55	5	9	13	16	23	32	45	52	64	74	88	ΨΕΥΔΕΣ

Σχήμα: η εκτέλεση μιας «δихοτομικής αναζήτησης»

(«κίτρινο» = οι μεταβολές τιμών, «γκρίζο» = η περιοχή αναζήτησης T[L...R], «πράσινο»: το T[M])

Τις όποιες (και σαφώς πιο περίπλοκες) *οδηγίες* της παραπάνω διχοτομικής αναζήτησης, θα τις γράφαμε σε μια «ψευδογλώσσα», ή καλύτερα σε μια γλώσσα «υψηλού επιπέδου» όπως παρακάτω. Η εκτέλεση θα γινόταν με τον τρόπο που διδασκόμαστε στα μαθήματα προγραμματισμού:

```

Αλγόριθμος: «Διχοτομική αναζήτηση».

Συνάρτηση Περιέχεται (T: πίνακας, L,R: φυσικός, σ: στοιχείο): ΑΛΗΘΕΣ/ΨΕΥΔΕΣ

// T πίνακας στοιχείων, ταξινομημένα κατ' αύξουσα σειρά .

{ OK ← ΨΕΥΔΕΣ
  Εφόσον (L ≤ R) και όχι OK
  { M ← (L+R) δια 2 // λ.χ. ((3+6) δια 2)=4
    Περίπτωση
    { σ < T[M]: { R ← M-1 }
      σ = T[M]: { OK ← ΑΛΗΘΕΣ }
      σ > T[M]: { L ← M+1 }
    } }
  Περιέχεται ← OK
}

```

Αυτό που θέλουμε να κρατήσουμε από το παράδειγμα της διχοτομικής αναζήτησης, είναι ότι στους «υπολογισμούς» αυτού του είδους *χειρίζομαστε ένα περιβάλλον «μεταβλητών»*, τις οποίες βάσει οδηγιών³ είτε ανακαλούμε προς χρήση, είτε ενημερώνουμε με νέες τιμές.

Θα χρειαστούμε μια μακρά πορεία για να προσδιορίσουμε και να «αναλύσουμε» μαθηματικά τα παραπάνω είδη οδηγιών και υπολογισμού. Φαίνεται όμως, ότι ο συμβολικός/γραμματικός τρόπος είναι ο πλέον θεμελιακός, διότι σε κάθε περίπτωση οι χειρισμοί μας δεν γίνονται «στον αέρα»: είτε με τον ένα είτε με τον άλλον τρόπο, κάτι θα «γράψουμε». Με αυτόν τον τρόπο υπολογισμού θα αρχίσουμε την ξενάγηση, ήδη από την επόμενη ενότητα.

³ Οι οποίες δεν είναι και τόσο απλές – θα πρέπει να παραδεχθούμε...

3^ο Ο «συμβολικός» τρόπος υπολογισμού – σύνταξη και σημασία.

Αναφερόμενοι στη προηγούμενες ενότητες οι εξής ορολογία και έννοιες θα είναι τα πρώτα θεμέλια της θεωρίας μας:

Αλφάβητο
(Σ)

Ένα αυθαίρετο πεπερασμένο σύνολο **συμβόλων** $\Sigma = \{\sigma_1, \sigma_2, \dots, \sigma_n\}$.
Συνήθως το αλφάβητο Σ περιέχει τουλάχιστον δύο σύμβολα, $|\Sigma| \geq 2$, αλλά μπορεί και να έχει μόνον ένα.

Χρησιμοποιούμε το αλφάβητο για την παράσταση των οποιωνδήποτε πληροφοριών, ή την περιγραφή των οποιωνδήποτε αντικειμένων, σκοπεύουμε να χειριστούμε υπολογιστικά. Τα συνηθισμένα γράμματα, τα αριθμητικά ψηφία και μερικά σημεία στίξης είναι υπεραρκετά για μια φυσική παράσταση οποιασδήποτε πληροφορίας. Θεωρητικά (και πολύ πρακτικά, π.χ. στα ολοκληρωμένα κυκλώματα) ένα δυαδικό αλφάβητο δύο συμβόλων, «0» και «1», είναι επαρκές.

Λέξη
(λ)

Ως λέξη θα θεωρούμε μια πεπερασμένη ακολουθία λ , συμβόλων εκ του αλφαβήτου που έχουμε επιλέξει: $\lambda = a_1 a_2 \dots a_n$, $a_k \in \Sigma$, $k = 1, \dots, n$. Το n είναι το **μήκος** της λέξης και θα το συμβολίζουμε $|\lambda|$. Επιτρέπουμε την τιμή $n = 0$, στην οποία περίπτωση θεωρούμε ότι η λέξη μας είναι **κενή** και θα την συμβολίζουμε με το σύμβολο \emptyset . Το σύνολο όλων των (πεπερασμένων, πάντοτε) λέξεων εξ ενός αλφαβήτου Σ θα το συμβολίζουμε με Σ^* .

Τί «είναι» οι λέξεις; Όταν υπολογίζουμε, υπάρχουν πράγματα που «θυμόμαστε», «σημειώσεις» που κρατάμε (λ.χ. στη μνήμη μιας συσκευής). Αυτά θα τα παριστάνουμε συμβολικά με **μία** λέξη. Επιλεγμένα σύμβολα που χρησιμοποιούνται ως σημεία «στίξης» είναι δυνατόν να ενώνουν συμβατικά πολλές λέξεις σε μία. Το παρόν κείμενο π.χ. μπορεί να θεωρηθεί ως **μία** λέξη με σημεία στίξης το «διάστημα», την «τελεία», τις «παύλες», τις «παρενθέσεις», κά.

Με μια λέξη παριστάνουμε τα δεδομένα που έχουμε, την διαθέσιμη σε μας πληροφορία, δηλαδή την περιγραφή των αντικειμένων που χειριζόμαστε. Το ότι αυτή είναι μια απλή μονοδιάστατη σειρά συμβόλων, ενώ μπορούμε να σκεφτούμε λ.χ. ακόμα και μια διδιάστατη «πληροφορία», δεν μας οδηγεί σε απώλεια γενικότητας, διότι αυτή η σειρά δεν αντιστοιχεί παρά στη χρονική σειρά με τα οποία ούτως ή άλλως θα τα εξετάσουμε. Ότι εμείς, ή μια υπολογιστική συσκευή, εξετάζει «**σε ένα βήμα**» της, αυτό το θεωρούμε ως ένα **σύμβολο**, και η σειρά της εξέτασης αντιστοιχεί στη λέξη που θεωρούμε ως τα δεδομένα μας. (Το πρακτικό αντίστοιχο μιας λέξης είναι τα γνωστά «αρχεία» των υπολογιστών μας – μια σειρά δηλαδή από «χαρακτήρες».)

Υπολογισμός
(ν)

Υπολογισμός ν , είναι (κατ' αρχάς) μια ακολουθία λέξεων $\nu_0, \nu_1, \nu_2, \dots, \nu_n, \dots$ δηλαδή μια απεικόνιση του \mathbf{N} (οι φυσικοί) στο Σ^* .

Διακρίνουμε τα εξής επί μέρους:

- κάθε λέξη της ακολουθίας ν_k , καλείται ένα **στάδιο** ή **φάση** υπολογισμού.
- κάθε ζεύγος διαδοχικών σταδίων (ν_k, ν_{k+1}) καλείται ένα **βήμα** υπολογισμού.
- κρατάμε κατ' αρχάς το ενδεχόμενο η ακολουθία/υπολογισμός να είναι άπειρος, αλλά στις πρακτικές περιπτώσεις μας ενδιαφέρει, φυσικά, αυτός να **περατούται**. Προς τούτο πρέπει να υπάρχει τελικό στάδιο $\nu_{\text{τελικό}}$ ώστε για $k > \text{τελικό}$ να ισχύει κάποια συνθήκη η οποία (κατά σύμβαση) να σημαίνει ότι ο υπολογισμός μας έχει περατωθεί: λ.χ. $\nu_k = \emptyset$. Το πρώτο τέτοιο ν είναι το **μήκος** του υπολογισμού. Εάν ο υπολογισμός ν περατούται θα γράφουμε με « $\nu \downarrow$ » (ή: υπολογισμός «συγκλίνει»): αλλιώς θα γράφουμε « $\nu \uparrow$ » (ή: υπολογισμός «αποκλίνει»)

Πρόγραμμα (π)

Τα προγράμματα ορίζουν τα επιτρεπτά βήματα ενός υπολογισμού, προσδιορίζουν δηλαδή το πότε δύο στάδια υπολογισμού ϕ και ϕ' μπορούν να θεωρηθούν διαδοχικά, $\phi = u_k$ και $\phi' = u_{k+1}$. Κάθε πρόγραμμα αντιστοιχεί σε μια σχέση λέξεων, που θα αποκαλούμε βήματα(π).

Είναι καίριο χαρακτηριστικό ενός υπολογισμού ότι τα στάδια του δεν διαδέχονται το ένα το άλλο κατά τυχαίο ή αυθαίρετο τρόπο – άλλως θα υπολογίζαμε τη λύση των προβλημάτων μας σε ένα (έστω μακρύ & αυθαίρετο) κατ' ευθείαν βήμα από τα δεδομένα στα επιθυμητά αποτελέσματα...

Τα βήματα ενός υπολογισμού δεν μας ενδιαφέρει να προκύπτουν αυθαίρετα, αυθόρμητα ή κατόπιν έμπνευσης: το κάθε επόμενο στάδιο υπολογισμού πρέπει σχετίζεται με το προηγούμενο σύμφωνα με κάποιες προκαθορισμένες οδηγίες, αυτές που αποκαλούμε **πρό-γραμμα**. Όλα τα ενδεχόμενα στάδια και βήματα υπολογισμού είναι όμως απείρου πλήθους και γι' αυτό δεν μπορούμε να περιγράψουμε ένα πρόγραμμα π παραθέτοντας όλα τα ζεύγη (u_k, u_{k+1}) αυτής της σχέσης: υπό αυτή την προοπτική η σχέση π αποτελεί απειρομέγεθες αντικείμενο.

Μπορούμε (;) όμως να την «συνοψίσουμε» σε μια συμβολική περιγραφή: τα προγράμματα δηλαδή είναι λέξεις $\pi \in \Sigma^*$, που περιγράφουν σχέσεις, τα βήματα(π). Π.χ. στα προηγούμενα γράψαμε «01C → C0» εννοώντας ότι:

«οποιοδήποτε $u_k = \alpha 01C \beta$, ($\alpha, \beta \in \Sigma^*$), μπορεί να γίνει $u_{k+1} = \alpha C0 \beta$ »,

Προφανώς, εάν χρειαζόμαστε έναν ολόκληρο υπολογισμό για να οδηγηθούμε στο επόμενο βήμα, τότε υποπίπτουμε σε φαύλο κύκλο: χρησιμοποιούμε υπολογισμούς για να ορίσουμε τα βήματα των υπολογισμών και ο ορισμός μας είναι δώρον-άδωρον.

Κάθε βήμα υπολογισμού (u_k, u_{k+1}) κατά το πρόγραμμα π , δηλαδή η «εφαρμογή» του π επί ενός τρέχοντος σταδίου u_k , θα πρέπει να είναι κάποιο **τετριμμένο καθήκον, εννόητο, χωρίς ανάγκη περαιτέρω ανάλυσης σε απλούστερα, και εκτελέσιμο μηχανικά χωρίς χρήση «διαίσθησης» ή «έμπνευσης»**: (όπως λ.χ. η προηγούμενη γραμματική αντικατάσταση του 01N με N1). Έτσι **γράφουμε** μεν μια λέξη (που είναι πεπερασμένο αντικείμενο) αλλά **εννοούμε** μια ολόκληρη συνάρτηση (που είναι άπειρο αντικείμενο).

Σημείωση: Είναι πράγματι προκλητικό να ταυτίσει κάποιος τους λεγόμενους **αλγόριθμους** με όλα τα προγράμματα: τότε όμως πώς θα συμβιάσουμε το σαφές εμπειρικό γεγονός, που αντιλαμβάνονται αμέσως όλοι όσοι προγραμματίζουν, ότι έχουμε την δυνατότητα να προγραμματίσουμε σε εντελώς διαφορετικές «γλώσσες προγραμματισμού» τον ίδιο, κατά κάποια έννοια, αλγόριθμο; Αυτό σημαίνει ότι η έννοια «αλγόριθμος» είναι κάτι πιο γενικό ή αφηρημένο από ότι η έννοια «πρόγραμμα», εδώ όμως δεν θα θίξουμε αυτό το θέμα: θα εξετάσουμε διάφορους γενικούς τρόπους υπολογισμού και προγραμματισμού, ταυτίζοντας (εδώ χωρίς βλάβη) την έννοια «αλγόριθμος» με την έννοια «πρόγραμμα».



Συντακτικοί και σημασιολογικοί κανόνες.

Συντακτικοί κανόνες:

Υπάρχουν απείρου πλήθους προβλήματα την λύση των οποίων θα θέλαμε να υπολογίζουμε, και προφανώς χρειαζόμαστε προς τούτο άπειρο πλήθος προγραμμάτων που θα θέλαμε να γράψουμε: δεν μπορούμε δηλαδή να έχουμε έναν πλήρη εξαντλητικό κατάλογο όλων των «προγραμμάτων».

Αντί γι' αυτό έχουμε ένα σύστημα κανόνων με βάσει το οποίο μπορούμε να συντάξουμε όλα τα δυνατά προγράμματα (ή: οδηγίες, ή εντολές, κττ) που επιτρέπει ο τρόπος προγραμματισμού που έχουμε προτιμήσει. Οι κανόνες αυτοί λέγονται **συντακτικοί**

κανόνες. Προφανώς η εφαρμογή τους θα πρέπει να είναι απλή και άμεσα κατανοητή.

Κάθε πρόγραμμα λοιπόν, π , συντεταγμένο σύμφωνα με τους κανόνες μας, υποδηλώνει ένα τρόπο να κάνουμε βήματα στον υπολογισμό μας: υποδηλώνει δηλαδή μια **σχέση μετάβασης** από στάδιο υπολογισμού σε επόμενο στάδιο υπολογισμού:

$$\text{βήματα}(\pi) = \{(\phi', \phi'') : \phi', \phi'' \text{ διαδοχικά στάδια ή φάσεις υπολογισμού} \}$$

Ο αυστηρός και ακριβής ορισμός του πώς υπολογίζουμε για κάθε πρόγραμμα π , το «νόημά» του, τον τρόπο δηλαδή της επίδρασής του στο εκάστοτε στάδιο u_k , αποτελεί τους **σημασιολογικούς κανόνες** του προγραμματικού τρόπου που έχουμε προτιμήσει, και δίδονται από τον ορισμό, με οποιονδήποτε εξυπηρετικό τρόπο, της σχέσης $\text{βήματα}(\pi)$.

Π.χ., στο 1^ο παράδειγμα της προηγούμενης ενότητας, τα «προγράμματα» ήσαν σύνολα από οδηγίες της μορφής $\lambda \rightarrow \lambda'$, δηλαδή ζεύγη λέξεων (λ, λ') , και το νόημα ενός ζεύγους ήταν:

«εάν το στάδιο u_k περιέχει τη λέξη λ τότε ως επόμενο στάδιο u_{k+1} επιτρέπεται αυτό που προκύπτει εάν αντικαταστήσουμε τη λ με τη λ' »

ΣΥΝΤΑΞΗ	ΣΗΜΑΣΙΑ:
« $\lambda \rightarrow \lambda'$ »	« $\alpha \lambda \tau \rightarrow \alpha \lambda' \tau$ », $\alpha, \tau \in \Sigma^*$
	βήματα(π) =
	$1 \lambda 0 0 \rightarrow 1 \lambda' 0 0$
	$1 0 1 \lambda 1 \rightarrow 1 0 1 \lambda' 1$
	$\lambda 1 \rightarrow \lambda' 1$
	$1 \lambda 0 \rightarrow 1 \lambda' 0$
	...
	$1 0 1 \lambda \rightarrow 1 0 1 \lambda'$
	ΚΟΚ

(Βλ. στο 1^ο παράδειγμα της προηγούμενης ενότητας, στο σχετικό σχήμα, την διαγραφή του **01C**, και την αντικατάστασή του από το **C0**).

Στο σημείο αυτό φαίνεται καθαρά πόσο «απλός» είναι ο γραμματικός τρόπος υπολογισμού. Για να γράψουμε σε αυτόν ένα πρόγραμμα (= η **σύνταξη**) αρκεί:

- να διαλέξουμε ένα αλφάβητο συμβόλων Σ ,
- να διαλέξουμε ένα πεπερασμένο σύνολο π από ζεύγη (λ, λ') λέξεων του Σ^* , και τέλος: έχουμε ένα «συντακτικώς ορθό» πρόγραμμα π .

Και για να ερμηνεύσουμε ένα πρόγραμμα π (= η **σημασία**) αρκεί:

- να εξετάζουμε το εκάστοτε στάδιο υπολογισμού u_k εάν περιέχει την (υπο)λέξη λ ,
- να την αντικαθιστούμε με τη λέξη λ' παράγοντας το νέο στάδιο u_{k+1} , και τέλος: έχουμε εκτελέσει ένα επιτρεπτό βήμα του υπολογισμού u ως προς π .

Για να εκτιμήσετε αυτή την απλότητα αρκεί να ανακαλέσετε τυχόν μαθήματα που έχετε παρακολουθήσει για μια γλώσσα προγραμματισμού «υψηλού επιπέδου», όπως λ.χ. η C, και να προσέξετε το πόσο δύσκολη είναι η εξήγηση του ποιά προγράμματα C είναι σωστά γραμμένα (= η **σύνταξη** της C), και ποιά όχι. Ή ακόμα χειρότερα, το πόσο δύσκολη είναι η εξήγηση του τί αποτέλεσμα θα έχει κάθε (;) εντολή της C, σε

κάθε (;) περιβάλλον μεταβλητών (= η σημασία της C) ⁴.

Η κατακλείδα είναι μια πρωτόλεια περιγραφή του τί κάνει ένα «πρόγραμμα»: δίδει μια σχέση «**παράγει(π, δ, υ)**», τριών παραμέτρων:

- του προγράμματος π.
- των δεδομένων δ, και,
- του υπολογισμού υ.

$$\begin{aligned} & \text{παράγει}(\pi, \delta, \nu) \\ = & \text{«Το πρόγραμμα } \pi, \text{ με δεδομένα } \delta, \text{ παράγει τον υπολογισμό } \nu\text{»} \\ & \Leftrightarrow \\ & \text{(το } \nu_0 \text{ περιέχει τα } \delta) \wedge \forall k \geq 0 ((\nu_k, \nu_{k+1}) \in \text{βήματα}(\pi)) \end{aligned}$$

Συνοψίζουμε, στον επόμενο πίνακα, τον συμβολισμό και την ορολογία/έννοιες που χρησιμοποιήσαμε:

\mathbf{N}	οι φυσικοί αριθμοί.
Σ	το (πεπερασμένο) αλφάβητο που χρησιμοποιούμε.
Σ^*	όλες οι (πεπερασμένες) λέξεις του αλφαβήτου Σ .
σ, α, β	σύμβολα του αλφαβήτου.
λ	μια λέξη από το χρησιμοποιούμενο αλφάβητο: $\lambda \in \Sigma^*$.
$\lambda[k]$	το κ-οστό σύμβολο της λέξης λ .
$ \lambda $	το μήκος της λέξης λ .
\emptyset	η κενή λέξη (με μηδενικό πλήθος συμβόλων).
$\sigma^{(k)}$	μια λέξη αποτελούμενη από $k \in \mathbf{N}$ σύμβολα «σ» στη σειρά.
π	ένα πρόγραμμα : μια λέξη του $\pi \in \Sigma^*$, συχνά ένα σύνολο «οδηγιών».
$\text{βήματα}(\pi)$	η σημασία ενός προγράμματος βήμα-προς-βήμα: μια σχέση $\subseteq \Sigma^* \times \Sigma^*$.
ν	ο υπολογισμός : μια ακολουθία λέξεων: $\mathbf{N} \rightarrow \Sigma^*$.
$\nu_k, \nu[k]$	το κ-οστό στάδιο του υπολογισμού ν .
$\nu \downarrow, \nu \uparrow$	ο υπολογισμός συγκλίνει (περατούται) ή αποκλίνει .

⁴ Αξίζει να ελέγξει κάποιος εάν μια τέτοια εξήγηση έχει ποτέ δοθεί πλήρως, και εάν χωράει σε λιγότερες από αρκετές εκατοντάδες σελίδες. Επίσης, το πόσο «φυσική» και ευνόητη είτε είναι, είτε θα μπορούσε να γίνει...

4^ο Οι μηχανές Turing.

Αναφερόμενοι στην προηγούμενη ενότητα, η απλή συντακτική μορφή των εντολών $\lambda \rightarrow \lambda'$ είναι επιφανειακή. Η ανάλυση των δυνατοτήτων τους παραμένει εξαιρετικά περίπλοκη (αφού λ.χ. η προς αντικατάσταση λέξη λ μπορεί να εμφανίζεται «εδώ και εκεί» με ανεξέλεγκτο τρόπο). Η οδός που ακολούθησε η θεωρία υπολογισμού είναι να περιορίσει αυτή την μορφή, και να μελετήσει ακόμα απλούστερες μορφές. Τέσσερις μορφές έπαιξαν τελικά ρόλο, με τον εξής τρόπο:

- Επιλέγουμε ένα αλφάβητο *τερματικών* συμβόλων Σ , τα οποία είναι και το κυρίως αντικείμενο των γραμματικών χειρισμών μας. Θα τα γράφουμε ως *πεζά* σύμβολα, λ.χ. $\alpha, \beta, \gamma, \sigma$.
- Επιλέγουμε ένα αλφάβητο *παραγωγικών* συμβόλων Σ_K , με τα οποία «θυμόμαστε» τί είδους εργασία επιτελούμε και πού. Θα τα γράφουμε ως *κεφαλαία* σύμβολα, λ.χ. K, L, I, T .
- Εξετάζουμε προγράμματα φτιαγμένα από τα εξής είδη εντολών, $\lambda \rightarrow \lambda'$:

λ	λ'	
$K \rightarrow \emptyset$		Ένα παραγωγικό σύμβολο K είτε,
$K \rightarrow \lambda K'$		<ul style="list-style-type: none"> ▪ καταργείται, τερματίζοντας την παραγωγή. ▪ αντικαθίσταται από μια λέξη λ και ένα νέο παραγωγικό σύμβολο K'. Ίσως η απλούστερη ενδιαφέρουσα δυνατή περίπτωση. Αυτή θα μελετήσουμε στο A' μέρος αυτών των σημειώσεων.
$K \rightarrow \emptyset$		Ένα παραγωγικό σύμβολο K είτε,
$K \rightarrow \lambda K_1 K_2$		<ul style="list-style-type: none"> ▪ καταργείται, τερματίζοντας την παραγωγή. ▪ αντικαθίσταται από μια λέξη λ και δύο νέα παραγωγικά σύμβολα K_1, K_2. Μια ειδική περίπτωση πρακτικώς εξαιρετικά χρήσιμη, όπως θα φανεί στο B' μέρος αυτών των σημειώσεων.
$\alpha K \beta \rightarrow \alpha \beta$		Το ίδιο, περίπου, όπως και πριν, αλλά τώρα το τ και ρ θα συμβεί
$\alpha K \beta \rightarrow \alpha \lambda K_1 K_2 \beta$		εξαρτάται όχι από το K και μόνον, αλλά και από τα «συμφραζόμενα» α και β . Εδώ δεν θα εξετάσουμε αυτή την περίπτωση.

'Turing' (βλ. παρακάτω.)

Η 4^η περίπτωση ήταν και η... αρχική, αυτή στην οποία «πήδηξε» κατ' ευθείαν, (από διαίσθηση;) ο ίδιος Turing το 1936. Καθώς καλύπτει όλες τις άλλες, θα την παρουσιάσουμε στη συνέχεια, αν και για την ανάλυσή της θα επανέλθουμε στο Γ' μέρος αυτών των σημειώσεων.

Η γραμματική μορφή των μηχανών Turing.

Για να φτιάξουμε μια μηχανή Turing T , αρκούν τα εξής βήματα:

- Επιλέγουμε ένα αλφάβητο *τερματικών* συμβόλων Σ , τα οποία είναι και το κυρίως αντικείμενο των γραμματικών χειρισμών μας.
- Επιλέγουμε ένα αλφάβητο *παραγωγικών* (ή μη-τερματικών συμβόλων) συμβόλων Σ_K , με τα οποία κατά κάποια έννοια θυμόμαστε τί είδους «εργασία» έχουμε να επιτελέσουμε και «πού».
- Επιλέγουμε ως πρόγραμμα της μηχανής ένα *σύνολο οδηγιών*. Ως *οδηγία* θεωρούμε οποιαδήποτε οποιοδήποτε ζεύγος λέξεων (λ, λ') της παρακάτω ειδικής μορφής, (όπου τα σ, σ', τ είναι τερματικά σύμβολα, και το K παραγωγικό σύμβολο):

λ	\rightarrow	λ'
$\tau K \sigma$	\rightarrow	$\tau \sigma' K'$
$\tau K \sigma$	\rightarrow	$\tau K' \sigma'$
$\tau K \sigma$	\rightarrow	$K' \tau \sigma'$

(Αγνοείτε τα «κενά διαστήματα»: έχουν τεθεί απλώς για να διευκολύνουν την ανάγνωση.)
Μια ματιά στον τύπο αυτών των οδηγιών αποκαλύπτει τις προθέσεις μας:

- μεταβάλλουμε μόνο ένα τερματικό σύμβολο σε κάθε βήμα, από σ σε σ' , και συγκεκριμένα το σύμβολο που το παραγωγικό σύμβολο K «βλέπει» στα δεξιά του.
- μεταβάλλουμε το σύμβολο κατάσταση K σε ένα νέο K' .
- μετακινούμε το σύμβολο κατάσταση K/K' κατά $0, \pm 1$ θέσεις.

Συντακτικοί κανόνες:

Θα γράφουμε μία οδηγία μηχανής Turing ως μία «5άδα»-στοιχείο του $(\Sigma_K \times \Sigma) \times (\Sigma_K \times \Sigma \times \{-1, +1, 0\})$, και συγκεκριμένα ως δύο τμήματα, ένα «αριστερό» και ένα «δεξιό»:

$$(K, \sigma) \rightarrow (K', \sigma', \delta) \text{ όπου } \delta = -1, 0, +1$$

Ως πρόγραμμα Turing π , θεωρούμε ένα σύνολο τέτοιων οδηγιών, δηλαδή:

$$\pi \subseteq (\Sigma_K \times \Sigma) \times (\Sigma_K \times \Sigma \times \{-1, +1, 0\})$$

όπου ταυτόχρονα διακρίνονται προς ειδική χρήση τρεις καταστάσεις:

- μια αρχική, $I_\pi \in \Sigma_K$, και...
- δύο τελικές, Y_π και N_π , (αντίστοιχες μιας «θετικής» (Yes) και μιας «αρνητικής» (No) κατάληξης του προγράμματος π).

Για τεχνικούς λόγους που θα φανούν αργότερα, θα απαιτούμε (συντακτικός κανόνας!) προς αποφυγή συγχύσεως τα εξής:

- το αρχικό σύμβολο I_π να είναι πράγματι «αρχικό», δηλαδή να μην εμφανίζεται στο δεξιό τμήμα $(K', -, -)$ καμμιάς οδηγίας του π , και..
- τα δύο τελικά σύμβολα Y_π και N_π , να είναι πράγματι «τελικά», δηλαδή να μην εμφανίζονται στο αριστερό $(K, -)$ τμήμα καμμιάς οδηγίας του π .

Αυτά θα είναι τα **συντακτικά** συστατικά στοιχεία του προγραμματικού συμβολισμού που θα εξετάσουμε για λίγο στη συνέχεια και εκτενέστερα στο Γ' μέρος αυτών των σημειώσεων.

Σημασιολογικοί κανόνες:

Για να ορίσουμε τί κάνει ένα πρόγραμμα π «πρέπει & αρκεί» να εξηγήσουμε ποιά βήματα υπολογισμού βήματα(π), είναι επιτρεπτά (ή και επιβαλλόμενα) από το πρόγραμμα π . Η σχέση των βημάτων(-) ορίζεται επί δύο λέξεων ϕ και ϕ' ως εξής:

- Τα βήματα(π) του υπολογισμού είναι όλα τα ζεύγη λέξεων (ϕ, ϕ') που ορίζονται ως εξής:
Εάν $\phi = \alpha \tau K \sigma \beta$, (όπου α, β οποιεσδήποτε λέξεις), και η $(K, \sigma) \rightarrow (K', \sigma', \delta)$ είναι οδηγία του π , τότε στο επόμενο στάδιο ϕ' το σ τρέπεται σε σ' , το K σε K' , και το K' μετακινείται κατά δ θέσεις:
για $\delta = -1$: $\phi' = \alpha K' \tau \sigma' \beta$,
για $\delta = 0$: $\phi' = \alpha \tau K' \sigma' \beta$,
για $\delta = +1$: $\phi' = \alpha \tau \sigma' K' \beta$,
αλλιώς (αν λ.χ. το κατάσταση K είναι τελικό σύμβολο: $K \in \{Y_\pi, N_\pi\}$), τότε $\phi' = \emptyset$.

Για ένα πρόγραμμα-Turing π , είναι λοιπόν δυνατόν, σύμφωνα με τα όσα έχουμε εξηγήσει να έχουμε υπολογισμούς $v = \langle v_0, v_1, \dots, v_k, v_{k+1}, \dots \rangle$ αρχίζοντας από μια λέξη-δεδομένο λ , σύμφωνα με την εξής σημασιολογία:

- Αρχικό στάδιο v_0 :
Για $k = 0$, $v_k = I_\pi \lambda$, όπου I το αρχική παραγωγικό σύμβολο του προγράμματος π .
- Βήμα υπολογισμού:
Για κάθε δύο διαδοχικά στάδια (v_k, v_{k+1}) πρέπει $(v_k, v_{k+1}) \in \text{βήματα}(\pi)$.
- Τέλος υπολογισμού:
Όταν και μόνον $v_{k+1} = \emptyset$.

Η μηχανική μορφή των «μηχανών-Turing».

Δώσαμε τον παραπάνω ορισμό κυρίως για να καταστεί σαφές ότι ο τρόπος αυτός υπολογισμού είναι «συμβολικός» ή «γραμματικο-συντακτικός». Είναι όμως πιο παραστατικό να δούμε τα παραπάνω προγράμματα ως προγράμματα μιας μηχανής η οποία αποκαλείται *μηχανή-Turing*.

Σύνταξη...

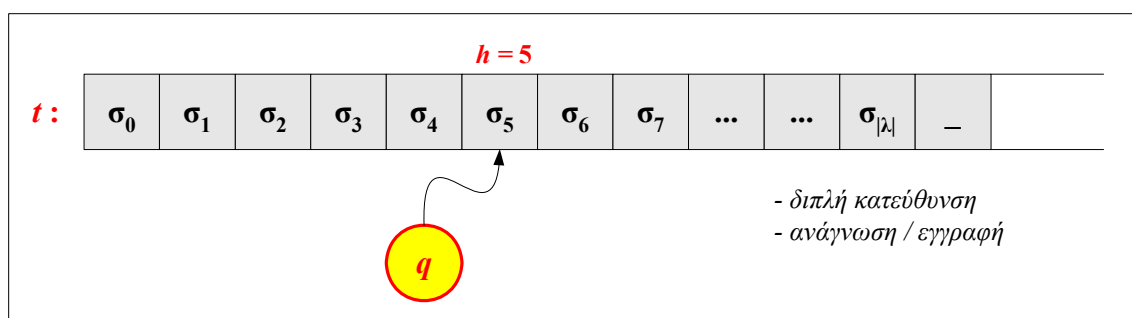
Σε μία μηχανή Turing διακρίνουμε τα ίδια συντακτικά στοιχεία, με τον ίδιο ρόλο, όπως πριν, μόνον που εδώ αποκαλούμε τα παραγωγικά σύμβολα (και) *καταστάσεις*.

... και σημασία:

Για την νοηματοδότηση των προγραμμάτων κατά «μηχανικό» τρόπο, θεωρούμε τα εξής:

- την *ταινία* της μηχανής, $t[0] t[1] t[2] \dots t[h] \dots$, δηλ. μια ακολουθία από θέσεις ή κυψέλες, στις οποίες είναι καταχωρισμένα κάποια τερματικά σύμβολα, τα οποία από ένα σημείο και μετά είναι ίσα με «_» (συμβατικά, το σύμβολο του «διαστήματος» ή της κενής θέσης).
- την *κεφαλή* της μηχανής, δηλαδή έναν φυσικό αριθμό $h \in \mathbf{N}$, ο οποίος δηλώνει σε ποιά θέση της ταινίας ευρίσκεται η κεφαλή,
- και τέλος την εκάστοτε *κατάσταση* $q \in \Sigma_k$.

Θα αποκαλούμε μια τριάδα (t, h, q) ως την *(καταστατική) περιγραφή* του αυτόματου. Θα σχεδιάζουμε ένα αυτόματο όπως παρακάτω, σχεδιάζοντας την «ταινία» σαν μια σειρά από κυψέλες που περιέχουν το αντίστοιχο σύμβολο και είναι αριθμημένες από αριστερά προς τα δεξιά, και την «κεφαλή» σαν μια κυψέλη που δείχνει με ένα βέλος τη θέση της και η οποία περιέχει την «κατάσταση» του αυτόματου.



Σχήμα: Η «ταινία» t , μιας μηχανής-Turing.

Ορίζουμε έναν υπολογισμό της μηχανής-Turing ως μία ακολουθία καταστατικών περιγραφών στην οποία η κάθε επόμενη περιγραφή ϕ' προκύπτει από την προηγούμενη ϕ , σύμφωνα με μία κάποια οδηγία του προγράμματος π . Συγκεκριμένα:

- Η σχέση βήματα(π) του υπολογισμού είναι όλα τα ζεύγη λέξεων (ϕ, ϕ') που ορίζονται ως εξής:
Μια οδηγία $(K, \sigma) \rightarrow (K', \sigma', \delta)$ είναι εφαρμόσιμη επί της $\phi = (t, h, q)$, εάν:
 - η κεφαλή «βλέπει» το σύμβολο μ , εάν δηλαδή: $t[h] = \sigma$, και,
 - η κατάσταση του αυτομάτου είναι η $K: q = K$,και εάν είναι εφαρμόσιμη, τότε η καταστατική περιγραφή ϕ' , είναι η εξής:

$$\phi' = \langle t', h', q' \rangle, \text{ όπου } t' = t, q' = K' \text{ και } h' = h + |\mu|.$$

(δηλαδή: το περιεχόμενο της ταινίας δεν μεταβάλλεται, η νέα κατάσταση είναι η K' , και η κεφαλή μετακινείται τόσες θέσεις όσα είναι τα σύμβολα της λέξης μ που «ανέγνωσε».)

Για μια μηχανή-Turing, σύμφωνα με τα όσα έχουμε εξηγήσει, έχουμε υπολογισμούς $v = \langle v_0, v_1, \dots, v_k, \dots \rangle$ αρχίζοντας από μια λέξη-δεδομένο λ , σύμφωνα με την εξής σημασιολογία:

- *Αρχικό στάδιο v_0 :*
Κάθε λέξη $\lambda = \sigma_1 \sigma_2 \sigma_3 \dots \sigma_n \in \Sigma^*$ μπορεί να «φορτωθεί» στην ταινία του αυτόματου ορίζοντας ως $t[k] = \sigma_k$, για $k = 1, \dots, |\lambda|$, (και κενές τις υπόλοιπες θέσεις). Η μηχανή αρχίζει με την αρχική καταστατική περιγραφή:

$$\langle t = \lambda, h = 1, q = I_\pi \rangle$$

-
- *Βήμα υπολογισμού:*
Για κάθε δύο διαδοχικές περιγραφές (u_k, u_{k+1}) πρέπει $(u_k, u_{k+1}) \in \text{βήματα}(\pi)$.
 - *Τέλος υπολογισμού:*
Όταν και μόνον η κατάσταση q είναι τελική: $q \in \{Y_\pi, N_\pi\}$.

Η αντιστοιχία μια μηχανής Turing με τις γραμματικές a-la Turing είναι προφανής. Η «μηχανή» είναι απλώς ένας λίγο πιο παραστατικός τρόπος παρουσίασης της γραμματικής.

Τί μπορούμε να υπολογίσουμε με τέτοιες μηχανές; Ο Turing ισχυρίστηκε το 1936, (και πολλοί άλλοι τα επιβεβαίωσαν στη συνέχεια), ότι αυτές οι μηχανές αρκούν για να υπολογίσουμε οτιδήποτε θα μπορούσαμε να υπολογίσουμε με οποιοδήποτε άλλο τρόπο. Αλλά ο δρόμος μέχρι εκεί είναι μακρύς.

5^ο Ομαλές γραμματικές και Πεπερασμένα αυτόματα.

Θα αρχίσουμε την ανάλυση των θεμελιακών ιδιοτήτων αυτών των τρόπων υπολογισμού, από την 1^η και απλούστερη περίπτωση τις «ομαλές γλώσσες». Μια «ομαλή γλώσσα» μπορεί να θεωρηθεί από 4 (τουλάχιστον) διαφορετικές οπτικές γωνίες.

1^ο: Ομαλές γραμματικές – η παραγωγική μορφή.

Ορίζουμε ως *ομαλή γραμματική*⁵ μία 4άδα, $G = \langle \Sigma, \Sigma_K, I, \Delta \rangle$ όπου:

- Σ Το «αλφάβητο»: ένα πεπερασμένο σύνολο *συμβόλων*, τα *τερματικά* σύμβολα. (Θα τα γράφουμε συνήθως με πεζά γράμματα, λ.χ. α, β, γ.)
- Σ_K Τα *παραγωγικά* σύμβολα ή *καταστάσεις*, ένα πεπερασμένο σύνολο συμβόλων. (Θα τα γράφουμε συνήθως με κεφαλαία γράμματα, λ.χ. K, Λ, Μ.)
- I Ένα επιλεγμένο *αρχικό* παραγωγικό σύμβολο $I \in \Sigma_K$.
- Δ Ένα σύνολο *κανόνων*. Στην παραγωγική μορφή μιας ομαλής γραμματικής οι κανόνες έχουν την εξής μορφή:
 - είτε $K \rightarrow \emptyset$.
 - είτε $K \rightarrow \lambda K'$.όπου τα K, K' είναι παραγωγικά σύμβολα, και το λ είναι μια λέξη από τερματικά σύμβολα: $\lambda \in \Sigma^*$.

Μια ομαλή γραμματική προσφέρει κανόνες «επαναγραφής» μιας λέξης $u_k \rightarrow u_{k+1}$: για ένα στάδιο $u_k \in (\Sigma \cup \Sigma_K)^*$, και κατά τα όσα έχουμε αναφέρει στις προηγούμενες ενότητες ισχύει το εξής:

«εάν $u_k = \mu K$ τότε ως επόμενο στάδιο μπορούμε να λάβουμε τη λέξη $u_{k+1} = \mu \lambda K'$ »

Το ζεύγος « $u_k \rightarrow u_{k+1}$ » θα το αναφέρουμε ως *βήμα παραγωγής* μιας λέξης. Ένας «υπολογισμός» είναι λοιπόν εδώ μια ακολουθία βημάτων:

$$u_0 \rightarrow u_1 \rightarrow u_2 \rightarrow \dots \rightarrow u_{n-1} \rightarrow u_n$$

στην οποία κάθε ζεύγος διαδοχικών σταδίων είναι ένα βήμα παραγωγής ως προς κάποιον από τους κανόνες της γραμματικής μας. Ειδικότερα μας ενδιαφέρουν οι εξής υπολογισμοί, που θα αποκαλούμε *παραγωγές*:

- το πρώτο στάδιο u_0 , είναι η λέξη I (το αρχικό παραγωγικό σύμβολο), και
- το τελευταίο στάδιο, u_n , αποτελείται μόνο από τερματικά σύμβολα $\lambda \in \Sigma^*$.

Θα συμβολίζουμε μια παραγωγή της λέξης λ από την γραμματική G με $I \Rightarrow_G \lambda$, (ή απλώς $I \Rightarrow \lambda$). Το σύνολο των λέξεων $\lambda \in \Sigma^*$ για τις οποίες υπάρχει μία τουλάχιστον παραγωγή, $I \Rightarrow \lambda$, θα το ονομάζουμε η *γλώσσα* $L(G)$ που ορίζεται από την γραμματική G .

(Προσέξτε ότι έχουμε έναν ιδιόμορφο υπολογισμό αφού,

- ουσιαστικά δεν έχουμε δεδομένα...
- ο υπολογισμός είναι «διακλαδωτικός» αφού είναι δυνατόν να έχουμε εναλλακτικές επιλογές αν λ.χ. $K \rightarrow \mu_1 K_1$ αλλά και $K \rightarrow \mu_2 K_2$.)

2^ο: Αυτόματα και ομαλές γραμματικές: ένα παραστατικό διάγραμμα.

Στα όσα θα ακολουθήσουν θα μας φανεί πολύ χρήσιμη μια παραστατική περιγραφή των οδηγιών μιας ομαλής γραμματικής (στη παραγωγική της μορφή). Αυτή θα είναι το *διάγραμμα μεταβάσεων*, που ορίζεται και αποτελείται από τα εξής:

⁵ «regular grammar»: συνήθίζεται και ο όρος «κανονική γραμματική», αν και η λέξη «κανονικός» μεταφράζει συνήθως την σημαντική μαθηματική έννοια του όρου *normal*, ενώ ο σχετικός αγγλικός όρος εδώ είναι *regular*.

- ένα πεπερασμένο σύνολο **κόμβων**, καθένας των οποίων φέρει ως επιγραφή ένα σύμβολο «κατάστασης» επιλεγμένο από ένα σύνολο καταστάσεων Σ_k . Θα σχεδιάζουμε ένα κόμβο σαν ένα «μικρό» κύκλο που περιβάλλει το σύμβολο που περιέχει.
- ένα πεπερασμένο σύνολο κατευθυντών **ακμών** που φέρουν ως επιγραφή λέξεις $\mu \in \Sigma^*$ εξ ενός αλφαβήτου τερματικών συμβόλων Σ . Θα σχεδιάζουμε τις ακμές ως βέλη, με την επιγραφή τοποθετημένη πλησίον του μέσου της ακμής.
- ένα επιλεγμένο αφετηριακό κόμβο με επιγραφή $I \in \Sigma_k$. Θα διακρίνουμε τον αφετηριακό κόμβο σχεδιάζοντας ένα βέλος που καταλήγει σε αυτόν χωρίς να εκκινεί από κανέναν άλλο κόμβο.
- ένα επιλεγμένο σύνολο τερματικών κόμβων $T \subseteq \Sigma_k$. Θα σχεδιάζουμε τους τερματικούς κόμβους με κύκλους των οποίων η περιφέρεια έχει σχεδιαστεί με «διπλή» γραμμή. Θα αποκαλούμε στις καταστάσεις T ως αποδεκτικές καταστάσεις, και στις οι υπόλοιπες $\Sigma_k - T$ ως απορριπτικές.

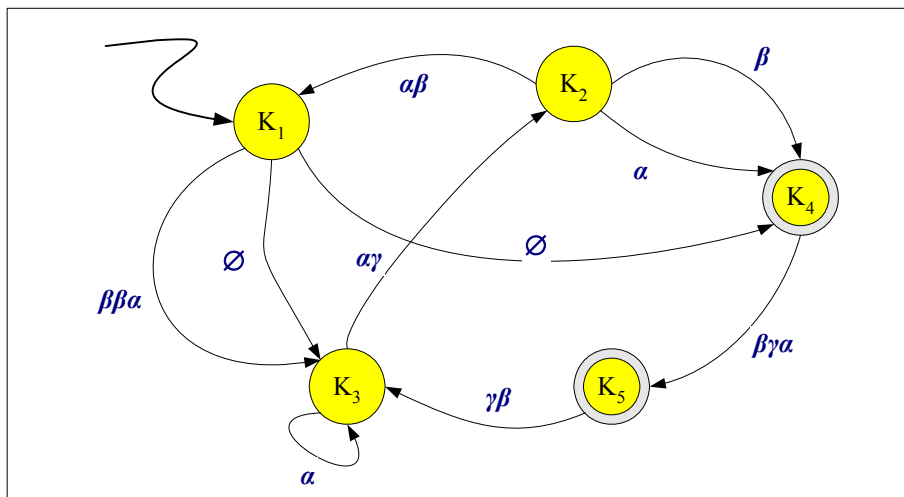
Σπεύδουμε εδώ να σχολιάσουμε την περίπου προφανή αντιστοιχία αυτών των διαγραμμάτων με τις ομαλές γραμματικές (παραγωγική μορφή):

Για κάθε ομαλή γραμματική $G = \langle \Sigma, \Sigma_k, I, \Delta \rangle$, σχεδιάζουμε το αντίστοιχο διάγραμμα D ως εξής:

- οι κόμβοι είναι όσοι τα παραγωγικά σύμβολα Σ_k , και επιγράφονται με αυτά.
- αφετηριακός κόμβος ορίζεται αυτό που επιγράφεται με το αρχικό σύμβολο I .
- για κάθε κανόνα $K \rightarrow \emptyset$, ο κόμβος K χαρακτηρίζεται ως τερματικός, και συλλέγεται στο σύνολο T .
- για κάθε κανόνα $K \rightarrow \lambda K'$, προσθέτουμε μια ακμή μετάβασης από τον κόμβο K στον κόμβο K' , και θέτουμε ως επιγραφή της ακμής αυτής τη λέξη λ .

Συμμετρικά, για κάθε διάγραμμα μεταβάσεων D , ορίζεται μια ομαλή γραμματική: $G = \langle \Sigma, \Sigma_k, I, \Delta \rangle$ στη παραγωγική της μορφή:

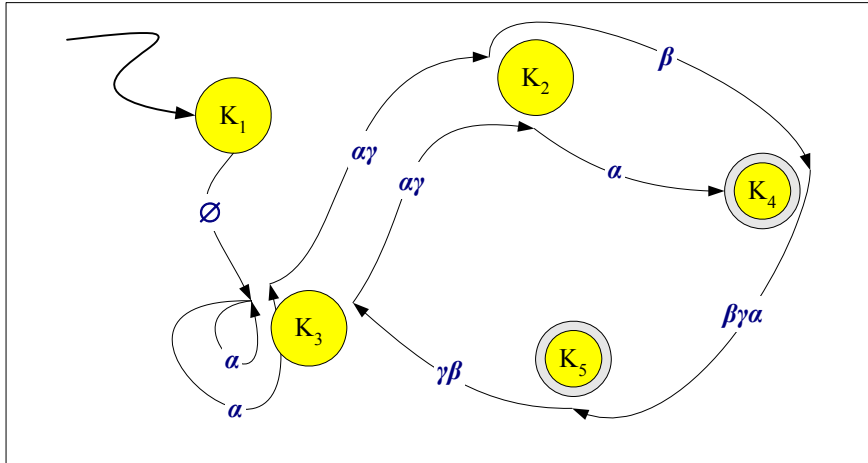
- το αλφάβητο Σ περιέχει όσα σύμβολα εμφανίζονται στις επιγραφές των ακμών του D .
- τα παραγωγικά σύμβολα είναι οι επιγραφές των κόμβων.
- αρχικό σύμβολο η επιγραφή I του αφετηριακού κόμβου
- για κάθε τερματικό κόμβο, προσθέτουμε τον κανόνα $K \rightarrow \emptyset$, στους κανόνες Δ .
- για κάθε ακμή (K, K') με επιγραφή λ , προσθέτουμε τον κανόνα $K \rightarrow \lambda K'$ στους κανόνες Δ .



Σχήμα: Ένα διάγραμμα μεταβάσεων D .

Μπορούμε να χρησιμοποιήσουμε ένα (οποιοδήποτε) διάγραμμα μεταβάσεων D για να ορίσουμε μια γλώσσα ως εξής: Θα ονομάζουμε **περίπατο** στο διάγραμμα μεταβάσεων κάθε ακολουθία διαδοχικών ακμών $w = \langle e_1, e_2, \dots, e_k, e_{k+1}, \dots, e_n \rangle$ – δηλαδή ακμών καθεμιά των οποίων καταλήγει στον κόμβο από τον οποίο εκκινεί η επόμενη. Ένας περίπατος θα είναι «αποδεκτικός» εάν (α) η αφετηρία του είναι ο αφετηριακός κόμβος, και (β) καταλήγει σε τερματικό κόμβο. Εάν διανύοντας ένα περίπατο w παραθέτουμε τις επιγραφές των ακμών του τότε σχηματίζουμε μια λέξη $\lambda(w)$. Η γλώσσα που ορίζει το διάγραμμα D είναι το σύνολο των λέξεων που παράγονται από όλους και μόνον τους αποδεκτικούς περιπάτους:

$$L(D) = \{\lambda(w): w \text{ αποδεκτικός περίπατος στο } D\}$$



Σχήμα: Ένας (αποδεκτικός) περίπατος στο διάγραμμα μεταβάσεων D που παράγει τη λέξη: $\alpha \alpha \alpha \gamma \beta \beta \gamma \alpha \gamma \beta \alpha \gamma \alpha$.

Είναι σχεδόν άμεσα προφανές ότι εάν η γραμματική G αντιστοιχεί στο διάγραμμα μεταβάσεων D , τότε οι αντίστοιχες γλώσσες που παράγονται είναι οι ίδιες: $L(G) = L(D)$.



Τι είναι μια «γλώσσα»;

Μια γλώσσα δεν είναι παρά ένα σύνολο λέξεων – υποσύνολο όλων των δυνατών λέξεων ενός αλφαβήτου Σ . Κάθε τέτοια λέξη δεν είναι παρά ο τρόπος που έχουμε διαλέξει για να παραστήσουμε ή περιγράψουμε ένα αντικείμενο που μας ενδιαφέρει. Αυτό που διακρίνει τις λέξεις μιας συγκεκριμένης γλώσσας από τις υπόλοιπες είναι ότι τα αντικείμενα που περιγράφουν έχουν μια «κοινή ιδιότητα» – ενώ εκείνα των υπολοίπων δεν την έχουν. Επομένως μια μηχανή που αποδέχεται μιιά γλώσσα μας επιτρέπει να αναγνωρίζουμε τα μέλη ενός συνόλου λέξεων, να «υπολογίζουμε» δηλαδή εάν ένα εκ των αντικειμένων που έχουμε περιγράψει, έχει ή δεν έχει μια «ιδιότητα».

3ο: Η «αναλυτική μορφή» μιας ομαλής γραμματικής.

Με την παραγωγική μορφή μιας ομαλής γραμματικής G μπορούμε να ορίζουμε μια γλώσσα $L(G)$. Αλλά τί να κάνουμε έναν ορισμό εάν δεν μπορούμε να τον εφαρμόσουμε, εάν δηλαδή δεδομένης μιας λέξης $\lambda \in \Sigma^*$ δεν είμαστε σε θέση να εξακριβώσουμε εάν αυτή η λέξη εμπίπτει ή όχι, σε αυτόν τον (γραμματικό) ορισμό;

Χρειαζόμαστε λοιπόν και κάποιου είδους κανόνες για να αναλύουμε μια λέξη λ ως προς μια γραμματική G , προς διαπίστωση του εάν $\lambda \in L(G)$. Η ανάλυση μιας λέξης ως προς την γραμματική G επιδιώκει την ανακάλυψη μιας παραγωγής $I \Rightarrow \lambda$. Το πώς μπορεί να γίνει αυτό δεν είναι προφανές εκ πρώτης όψεως (ούτε εκ δευτέρας...). Αυτό όμως που είναι φανερό εξ αρχής είναι ότι εάν (με κάποιο τρόπο) μας δινόταν η σειρά των παραγωγικών κανόνων που εφαρμόστηκαν για την παραγωγή, τότε θα έπρεπε να είμαστε σε θέση, τουλάχιστον, να **επιβεβαιώνουμε** ότι αυτή η σειρά κανόνων «ταιριάζει» με την υπό ανάλυση λέξη. Π.χ. εάν ο 1^{ος} κανόνας που εφαρμόστηκε ήταν $I \rightarrow \alpha \beta K$ θα πρέπει να είμαστε σε θέση να επιβεβαιώνουμε ότι το δύο πρώτα σύμβολα της λ είναι τα α και β . Εμείς, ως άνθρωποι, θα το κάναμε αυτό τοποθετώντας τα α και β κάτω από την αρχή της λέξης λ (είτε γραπτά είναι νοητά), και συγκρίνοντας αυτά ένα-προς-ένα, (ιδίως εάν ήσαν πολλά και όχι μόνον 2) με τα αντίστοιχα της λ .

Οι γραμματικοί κανόνες όμως δεν έχουν αυτή την «δισδιάστατη» μορφή, αλλά μπορούμε να έχουμε το ισοδύναμο αποτέλεσμα, εάν τα τοποθετούσαμε αριστερα της λ , με αντίστροφη σειρά σχηματίζοντας την λέξη $K \beta \alpha \Rightarrow \lambda$. Εάν η λ άρχιζε με α , εάν είχε δηλαδή την μορφή $\alpha \lambda'$, τότε θα είχαμε την λέξη

$K \beta \alpha \Rightarrow \alpha \lambda'$, και η διαπίστωση ότι η λ αρχίζει με 'α' λαμβάνει πια γραμματική μορφή, αφού εμφανίζεται η (υπο)λέξη $\alpha \Rightarrow \alpha$ – που μαρτυρεί κατά προφανή τρόπο ότι «βρήκαμε ότι ζητούσαμε».

Για την αναλυτική μορφή, λοιπόν, μιας ομαλής γραμματικής G θα χειριζόμαστε λέξεις την μορφή $\rho \Rightarrow \lambda$ (ώστε να συμβολίζουν υποτιθέμενες παραγωγές).

- Το αριστερό τμήμα – $\rho \in (\Sigma \cup \Sigma_k)^*$ – θα είναι ο κανόνας προς επιβεβαίωση ότι είναι εφαρμόσιμος.
- Το δεξιό τμήμα – $\lambda \in \Sigma^*$ θα είναι η λέξη προς ανάλυση (ή ότι υπόλοιπο έχει μείνει από αυτήν).

Αν το αριστερό τμήμα γράφεται ως $\rho = \rho' \sigma$, και το δεξιό ως $\lambda = \sigma \lambda'$, για κάποιο σύμβολο σ , τότε το σύμβολο σ πράγματι «επιβεβαιώνεται», το «διαβάζουμε» από τα δεδομένα, και «προχωρούμε» στο επόμενο. Αυτό είναι εκφράσιμο με τον απλό γραμματικό κανόνα: $\sigma \Rightarrow \sigma \rightarrow \Rightarrow$ (για κάθε $\sigma \in \Sigma$). Αν λοιπόν πρέπει να επιβεβαιώσουμε την λέξη $\phi = K \Rightarrow \lambda$, και υπάρχει στην G κάποιος κανόνας $K \rightarrow \sigma$ K' , αρκεί να επιβεβαιώσουμε την λέξη $\phi' = K' \sigma \Rightarrow \lambda$.

Οι προηγούμενες παρατηρήσεις εξηγούν το γιατί η παρακάτω «αναλυτική» μορφή G' μιας ομαλής γραμματικής G είναι σε θέση να επιβεβαιώνει σωστά ότι μια λέξη λ παράγεται από G :

Σ' Τερματικά σύμβολα: τα ίδιο με της G , συν ένα διαχωριστικό σύμβολο \Rightarrow .

Σ'_k Παραγωγικά σύμβολα: τα ίδια με εκείνα της G .

I' Το ίδιο το αρχικό σύμβολο της G .

Δ' Οι (νέοι) κανόνες:

- για κάθε κανόνα $K \rightarrow \emptyset$ της G , προσθέτουμε στο Δ' τον κανόνα: $K \Rightarrow \rightarrow \Rightarrow$
- για κάθε κανόνα $K \rightarrow \lambda K'$ της G , όπου $\lambda = \sigma_1 \sigma_2 \dots \sigma_n$, ($\sigma_k \in \Sigma$, $k = 1, \dots, n$), προσθέτουμε στο Δ' τον κανόνα: $K \Rightarrow \rightarrow K' \sigma_n \dots \sigma_2 \sigma_1 \Rightarrow$
- για κάθε τερματικό σύμβολο $\sigma \in \Sigma$, προσθέτουμε στο Δ' τον κανόνα «επιβεβαίωσης»: $\sigma \Rightarrow \sigma \rightarrow \Rightarrow$

Με αυτή την μορφή, μια «**συντακτική** (ή γραμματική ή λεκτική) **ανάλυση**» είναι μια ακολουθία βημάτων,

$$u_0 \rightarrow u_1 \rightarrow u_2 \rightarrow \dots \rightarrow u_n$$

όπου:

- το πρώτο στάδιο u_0 , είναι η λέξη $I \Rightarrow \lambda$, δηλαδή το αρχικό σύμβολο και η προς ανάλυση λέξη λ .
- καθε ζεύγος διαδοχικών σταδίων είναι ένα βήμα ανάλυσης ως προς κάποιον από τους κανόνες της γραμματικής G' μας, και,
- το τελικό στάδιο u_n , γράφεται ως \Rightarrow , δηλαδή, και στο δεξιό τμήμα η λέξη λ έχει αναγνωστεί πλήρως, και στο αριστερό τμήμα δεν έχει μείνει καμμία αναγνωριστική «εκκρεμότητα».

4ο: Μια «μηχανή» για μια ομαλή γραμματική.

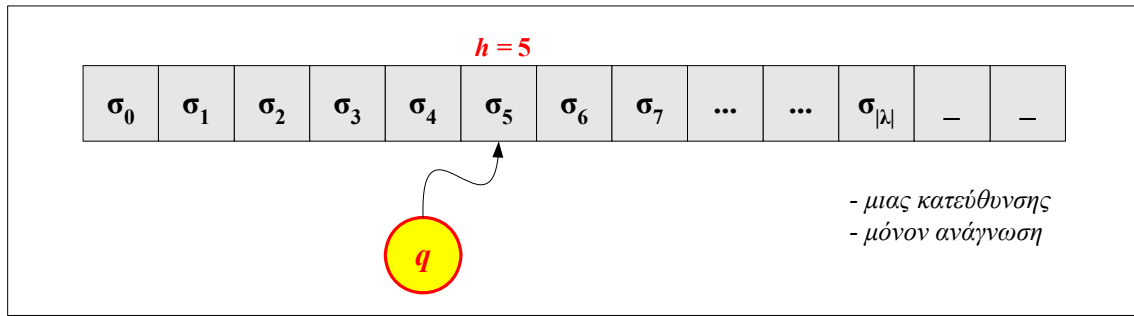
Και η αναλυτική μορφή των ομαλών γραμματικών επιδέχεται μια πιο παραστατική, και κυρίως μια πιο «μηχανική» περιγραφή, η οποία κατά κάποια έννοια είναι σε θέση να φέρει σε πέρας τους παραπάνω υπολογισμούς.

Θα ονομάζουμε (πεπερασμένο) **αυτόματο** μια 5άδα, $A = \langle \Sigma, \Sigma_k, I, T, \Delta \rangle$ όπου :

- το Σ είναι ένα οποιοδήποτε αλφάβητο.
- το Σ_k είναι κάποιο αλφάβητο, με σύμβολα που θα τα αποκαλούμε **καταστάσεις**,
- το $I \in \Sigma_k$ είναι μια διακεκριμένη κατάσταση που θα αποκαλούμε **αρχική**,
- το $T \subseteq \Sigma_k$, ένα (υπο)σύνολο διακεκριμένων καταστάσεων, οι **αποδεκτικές ή τελικές**, και
- οι κανόνες Δ είναι ένα σύνολο **οδηγιών** 3άδων της μορφής: $\langle K, \lambda, K' \rangle$, όπου:
 - τα K και K' είναι καταστάσεις, $K, K' \in \Sigma_k$, και,
 - το λ είναι μια λέξη από τερματικά σύμβολα, $\lambda \in \Sigma^*$.

Σε κάθε αυτόματο επισυνάπτουμε την «μηχανική» του όψη, και συγκεκριμμένα τα εξής:

- μια μεταβλητή κατάσταση q , $q \in \Sigma_k$.
- μια μεταβλητή h , (έναν φυσικό αριθμό $h \in \mathbb{N}$), ως την θέση μιας «κεφαλής».
- μια απεικόνιση, $t : \mathbb{N} \rightarrow \Sigma \cup \{ _ \}$, ως μια «ταινία» από κυψέλες με ένα τερματικό σύμβολο σε κάθε κυψέλη. (Θα σημειώνουμε μια «κενή» κυψέλη με το σύμβολο «κάτω παύλα» ' $_$ '.)



Σχήμα: Η εικόνα ενός (πεπερασμένου) αυτομάτου ως «μηχανής».

Ο σκοπός μας είναι η ταινία να φέρει επ' αυτής την υπό ανάλυση λέξη λ , η κεφαλή h να παίζει τον ρόλο του συμβόλου \Rightarrow προσδιορίζοντας την υπόλοιπη προς ανάλυση λέξη, και η κατάσταση K να αναπαριστά τον προς εφαρμογή αναλυτικό κανόνα. Ορίζουμε λοιπόν – κατά το γενικό πλαίσιο που δώσαμε στην εισαγωγή

- Η σχέση βήματα(π) της ανάλυσης είναι όλα τα ζεύγη λέξεων (ϕ, ϕ') που ορίζονται ως εξής: Μια οδηγία $\langle K, \lambda, K' \rangle$ είναι εφαρμόσιμη επί της περιγραφής $\phi = \langle t, h, q \rangle$, εάν:
 - η κεφαλή και για $|\lambda|$ σύμβολα προς τα δεξιά «βλέπει» την λέξη λ .
 - η κατάσταση του αυτομάτου είναι η K : $q = K$,
 και εάν είναι εφαρμόσιμη, τότε η καταστατική περιγραφή ϕ' , είναι η εξής:

$$\phi' = \langle t', h', q' \rangle, \text{ όπου } t' = t, q' = K' \text{ και } h' = h + |\lambda|.$$

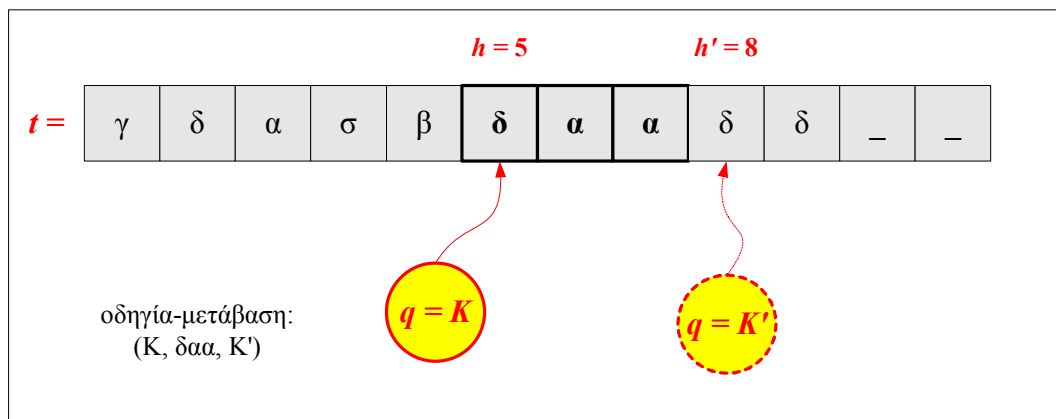
(δηλαδή: το περιεχόμενο της ταινίας δεν μεταβάλλεται, η νέα κατάσταση είναι η K' , και η κεφαλή μετακινείται τόσες θέσεις όσα είναι τα σύμβολα της λέξης λ που έχει πια «διαβάσει».)

Για ένα τέτοιο αυτόματο, σύμφωνα με όσα έχουμε εξηγήσει, έχουμε υπολογισμούς $v = \langle v_0, v_1, \dots, v_k, \dots \rangle$ αρχίζοντας από μια λέξη-δεδομένο λ , σύμφωνα με την εξής σημασιολογία:

- Αρχικό στάδιο v_0 : Κάθε λέξη $\lambda = \sigma_1 \sigma_2 \sigma_3 \dots \sigma_n \in \Sigma^*$ μπορεί να «φορτωθεί» στην ταινία του αυτομάτου ορίζοντας ως $t_i[k] = \sigma_k$, για $k = 1, \dots, |\lambda|$, (και κενές τις υπόλοιπες θέσεις). Η μηχανή αρχίζει με την αρχική καταστατική περιγραφή:

$$\langle t = \lambda, h = 1, q = I_\pi \rangle$$

- Βήμα υπολογισμού: Για κάθε δύο διαδοχικές περιγραφές (v_k, v_{k+1}) πρέπει $(v_k, v_{k+1}) \in \text{βήματα}(\pi)$.
- Τέλος υπολογισμού: Όταν και μόνον η κεφαλή έχει διαβάσει όλη την ταινία: $h = |\lambda| + 1$.



Σχήμα: Η οδηγία $\langle K, \delta\alpha\alpha, K' \rangle$ είναι εφαρμόσιμη.

Η εφαρμογή της θα προκαλούσε τις μεταβολές που εικονίζονται στο σχήμα.

Το αποτέλεσμα του υπολογισμού δίδεται από την τελική κατάσταση q , της μηχανής. Εάν αυτή είναι αποδεκτική, δηλ. $q \in T$, τότε λέμε ότι το αυτόματο *αποδέχθηκε την είσοδό του*, (την λέξη-δεδομένο λ), αλλιώς θα λέμε ότι την *απέρριψε*. Κάθε αυτόματο αποδέχεται, λοιπόν, μια γλώσσα $L(A) \subseteq \Sigma^*$ η οποία αποτελείται όλες και μόνον τις λέξεις του αλφαβήτου Σ , τις οποίες αυτό αποδέχεται.

Είναι φανερό από τα προηγούμενα ότι μια «ομαλή γλώσσα» έχει τέσσερις (τουλάχιστον) ισοδύναμους τρόπους να παρασταθεί: ως μια παραγωγική γραμματική ή διάγραμμα μεταβάσεων, και ως μια «αναλυτική» γραμματική ή πεπερασμένο αυτόματο. Λεπτομερέστερη «απόδειξη» περιττεύει: αυτή απλώς θα επαναλάμβανε πληκτικά τους παραπάνω ορισμούς.

Παράδειγμα: ένα αυτόματο που αναγνωρίζει ποιοι δυαδικοί αριθμοί διαιρούνται δια 3.

Έστω $\Sigma_2 = \{0,1\}$ το δυαδικό αλφάβητο. Το Σ_2^* μπορεί να ερμηνευθεί ως όλοι οι δυαδικοί αριθμοί. Εξ αυτών κάποιοι είναι πολλαπλάσια του 3. Έστω:

$$L_{3 \times} = \{ \lambda : \lambda \in \Sigma_2^*, \lambda - \text{ερμηνευμένη ως δυαδικός αριθμός} - \text{είναι πολλαπλάσιο του 3} \}$$

Είναι η $L_{3 \times}$ ομαλή γλώσσα; Γίνεται αποδεκτή από κάποιο πεπερασμένο αυτόματο; Ας υποθέσουμε ότι $\lambda = b_1 b_2 \dots b_n b_{n+1} \dots$, ότι έχουμε διαβάσει το αρχικό τμήμα της $\lambda_{[1..n]}$ έως και το n -οστό ψηφίο και ότι «θυμόμαστε» (ποιός; πώς;) το υπόλοιπο r_n της διαίρεσης δια 3 του αριθμού $\lambda_{[1..n]}$, (δηλαδή $\lambda_{[1..n]} = (3k_n + r_n)$). Εάν διαβάσουμε και λάβουμε υπόψη μας και το επόμενο δυαδικό ψηφίο b_{n+1} , ποιό θα είναι το νέο υπόλοιπο, αυτό δηλαδή του αριθμού $\lambda_{[1..n+1]}$; Προσέξτε ότι για τους αριθμούς $\lambda_{[1..n]}$ και $\lambda_{[1..n+1]}$ (δηλαδή τα αρχικά τμήματα της λέξης λ από 1 έως n σύμβολο, και από 1 έως $(n+1)$, ερμηνευμένα ως δυαδικούς) ισχύει:

$$\lambda_{[1..n+1]} = 2 \lambda_{[1..n]} + b_{n+1} = 2 (3k_n + r_n) + b_{n+1}$$

και ότι επομένως:

$$\lambda_{[1..n+1]} \bmod 3 = r_{n+1} = (2 r_n + b_{n+1}) \bmod 3$$

Το επόμενο υπόλοιπο, δηλαδή, είναι κάτι που μπορεί να υπολογιστεί,

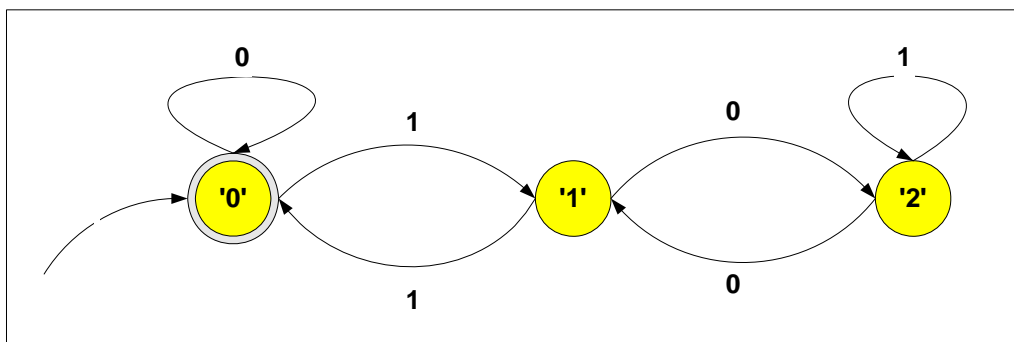
- από το προηγούμενο υπόλοιπο, και,
- το επόμενο σύμβολο-ψηφίο που διαβάζουμε.

Αρκεί επομένως να θυμόμαστε –με την κατάσταση του αυτομάτου μας– το εκάστοτε προηγούμενο υπόλοιπο r_n ! Ο παρακάτω πίνακας δίνει όλους τους 3×2 συνδυασμούς,

$$r_{n+1} = (2 r_n + b_{n+1}) \bmod 3 :$$

	$b_{n+1} = 0$	$b_{n+1} = 1$
$r_n = 0$	$r_{n+1} = 0$	$r_{n+1} = 1$
$r_n = 1$	$r_{n+1} = 2$	$r_{n+1} = 0$
$r_n = 2$	$r_{n+1} = 1$	$r_{n+1} = 2$

και ακολουθεί το αυτόματο εκπεφρασμένο ως διάγραμμα μεταβάσεων:



Σχήμα: Το διάγραμμα μεταβάσεων, για τις δυαδικές παραστάσεις των πολλαπλασίων του 3. ■

6^ο Πεπερασμένα αυτόματα: το «θεώρημα ισοδυναμίας» αιτιοκρατικών και μή.



Αιτιοκρατικές συσκευές: η υλοποιήσιμη εκδοχή μιας υπολογιστικής συσκευής.

Η παραγωγή και της ανάλυσης μιας λέξης ως προς μια ομαλή γραμματική μοιάζουν συμμετρικές διαδικασίες – αλλά αυτό είναι οπτική απάτη. Κατά την παραγωγή μιας λέξης έχουμε μια ελευθερία επιλογής κανόνων: λ.χ. σε κάποια φάση της παραγωγής ίσως να υπάρχουν δύο (τουλάχιστον) εφαρμόσιμοι κανόνες: $K \rightarrow \lambda_1 K_1$, και $K \rightarrow \lambda_2 K_2$. Ίσως μάλιστα να υπάρχουν και δύο κανόνες της μορφής $K \rightarrow \lambda K_1$, και $K \rightarrow \lambda K_2$, (δηλαδή $\lambda_1 = \lambda_2$). Ίσως μάλιστα να υπάρχει και κανόνας της μορφής: και $K \rightarrow K'$, (δηλαδή $\lambda = \emptyset$). Αυτό είναι επιθυμητό, διότι αυξάνει την εκφραστικότητα των κανόνων μας.

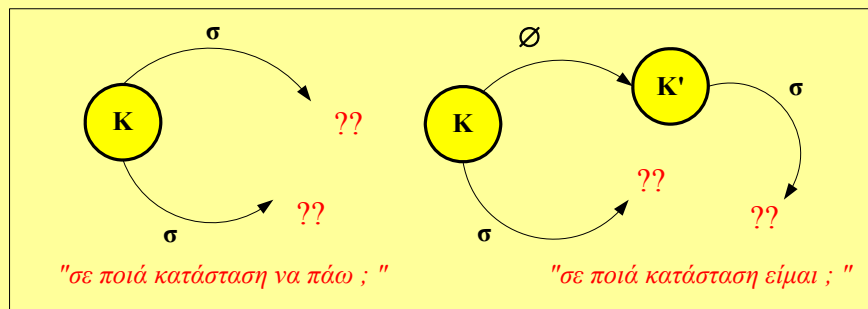
Κατά την παραγωγή μιας λέξης «πρέπει» λοιπόν να έχουμε πολλαπλές επιλογές, αλλά κατά την ανάλυση μιας λέξης, αυτό ακριβώς πρέπει να μην έχουμε. Ένα αυτόματο, εάν εγνώριζε τον εκάστοτε εφαρμοστέο κανόνα, θα μπορούσε να επιβεβαιώσει με «γραμματικό τρόπο», ότι αυτός πράγματι είναι εφαρμόσιμος· θα μπορούσε όμως και να εντοπίσει ποιός (αναλυτικός) κανόνας είναι ο εφαρμοστέος;

Εάν αναλύοντας την λέξη λ , το αυτόματο βρεθεί προ διλήμματος της μορφής: να εφαρμόσουμε τον $K \sigma_1 \rightarrow K_1$, ή τον $K \sigma_2 \rightarrow K_2$, τότε αρκεί να εξετάσει το 1^ο σύμβολο, έστω σ , της λέξης λ , να ελέγξει εάν $\sigma = \sigma_1$ ή $\sigma = \sigma_2$, ή ότι άλλο, και να πράξει αντιστοίχως. Εάν όμως ερχόταν η στιγμή να αντιμετωπίσει πολλαπλές επιλογές (όπως $K \sigma \rightarrow K'$ αλλά και $K \sigma \rightarrow K''$) τότε θα περιέπιπτε σε αμηχανία· σχεδόν κυριολεκτικά: πως θα αποφάσιζε σε μια τέτοια περίπτωση μια **μηχανή**;

Ας προσέξουμε εδώ, ότι ακόμα και η φαινομενικά αθώα οδηγία $K \rightarrow K'$ είναι ενδεχομένως μια συγκαλυμμένη πολλαπλή επιλογή καθώς εάν,

$$K \rightarrow K', \quad K \sigma \rightarrow K_1 \quad \text{και} \quad K' \sigma \rightarrow K_2$$

τότε ισοδύναμα $K \sigma \rightarrow K_1$ και $K \sigma \rightarrow K_2$. Στη 1^η περίπτωση «δεν ξέρουμε πού να πάμε», και στη 2^η περίπτωση «δεν γνωρίζουμε που βρισκόμαστε»...



Σχήμα: Διλήμματα στην «μηχανική» εκτέλεση οδηγιών.

Αυτή η ευελιξία είναι **πλεονέκτημα** από πλευράς (μαθηματικού) ορισμού, διότι διευκολύνει τον ορισμό ενός συνόλου από λέξεις μέσω μιας πιο εκφραστικής τεχνικής· (το είδαμε αυτό σε προηγούμενο παράδειγμα). Αποτελεί όμως **μειονέκτημα** από πλευράς «μηχανικού» υπολογισμού διότι μας φέρνει ενώπιον του διλήμματος:

«όταν βλέπω το σύμβολο σ , και έχω τις οδηγίες $K \sigma \rightarrow K'$, αλλά και $K \sigma \rightarrow K''$, πώς συνεχίζω; στη κατάσταση K' ή στην K'' ; »

Αυτή η διελευστίνδα ορισμού-υπολογισμού (εκφραστικότεροι ή συνοπτικότεροι ορισμοί απαιτούν δυσκολότερους υπολογισμούς για την διαπίστώσή τους) είναι

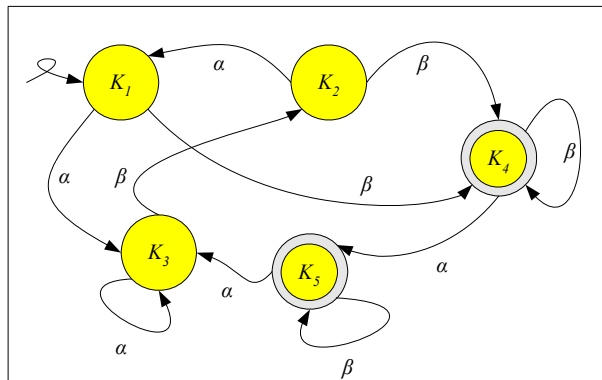
κεντρικό ζήτημα στην θεωρία υπολογισμού. Εμείς γνωρίζουμε ότι ενώπιον παρομοίων διλημάτων είναι δυνατόν να εξετάσουμε όλες τις πιθανές επιλογές – αλλά «εμείς», (οι σχεδιαστές, οι προγραμματιστές, οι μαθηματικοί, ή όποιος άλλος), όχι όμως κατ' ανάγκην και ο συγκεκριμένος *τύπος μηχανής* που αναλύουμε. Θα θεωρήσουμε λοιπόν μια ρεαλιστικά απλοποιημένη παραλλαγή των κανόνων γραμματικής ανάλυσης ώστε να αντιστοιχούν σε μια φυσικώς-τεχνολογικώς υλοποιήσιμη υπολογιστική συσκευή.

«Αιτιοκρατική» παραλλαγή ενός αυτομάτου.

Εάν...

- η σχέση-οδηγίες ενός αυτομάτου Δ , είναι συνάρτηση $\Sigma_K \times \Sigma \rightarrow \Sigma_K$, δηλαδή:
 - οι λέξεις «μ» των κανόνων αποτελούνται από ένα ακριβώς σύμβολο $K \sigma \rightarrow K'$, (οπότε δεν έχουμε κανόνες της μορφής $K \rightarrow K'$ όπου $\mu = \emptyset$)
 - για κάθε κατάσταση K , και κάθε σύμβολο σ έχουμε ένα και ακριβώς ένα κανόνα $K \sigma \rightarrow K'$, (οπότε η επόμενη κατάσταση $\Delta(K, \sigma)$ ορίζεται πάντοτε και μονότιμα, δηλαδή για κάθε K και κάθε σ , η μηχανή μας δεν περιέρχεται σε αμηχανία (sic), αλλά σε κάποια μοναδικά ορισμένη επόμενη κατάσταση).
- υπάρχει μία ακριβώς αρχική κατάσταση (όπως συνήθως επιτρέπει ο ορισμός των αυτομάτων),

τότε θα λέμε ότι η γραμματική είναι σε πρότυπη μορφή και ότι το αυτόματο είναι **αιτιοκρατικό**. Είναι προφανές ότι ένα αιτιοκρατικό αυτόματο έχει τα χαρακτηριστικά μιας φυσικής και υλοποιήσιμης συσκευής που έχει μηχανική συμπεριφορά – ενώ το μη αιτιοκρατικό δεν έχει τέτοια χαρακτηριστικά.



Σχήμα: Ένα αιτιοκρατικό αυτόματο, επί του αλφαβήτου $\Sigma = \{ \alpha, \beta \}$.

Θεώρημα: ισοδυναμία αιτιοκρατικών & μη-αιτιοκρατικών αυτομάτων

Ισχυρισμός: Για οποιοδήποτε μη-αιτιοκρατικό αυτόματο A , με n καταστάσεις, υπάρχει (και ορίζεται, και κατασκευάζεται) ένα αιτιοκρατικό αυτόματο A' , με $\leq 2^n$ καταστάσεις, το οποίο αποδέχεται την ίδια ακριβώς γλώσσα, δηλαδή: $L(A) = L(A')$.

Σχέδιο Απόδειξης: Υπάρχουν τέσσερις λόγοι για τους οποίους μπορεί ένα αυτόματο να μην είναι αιτιοκρατικό:

- υπάρχουν μεταβάσεις οφειλόμενες σε μια λέξη $\lambda \in \Sigma^*$ και όχι σε ακριβώς ένα σύμβολο $\sigma \in \Sigma$.
- υπάρχουν ζεύγη κατάσταση-σύμβολο χωρίς καμμία επόμενη κατάσταση.
- υπάρχουν κενές μεταβάσεις.
- υπάρχουν ζεύγη κατάσταση-σύμβολο με περισσότερες από μία επόμενες καταστάσεις.

Τα τρία πρώτα προβλήματα είναι δυνατόν να αντιμετωπιστούν σχετικά εύκολα.

- 1^ο βήμα: Αν έχουμε κάποιο κανόνα $K \rightarrow \lambda K'$, όπου το λ είναι μια λέξη $\lambda = \sigma_1 \sigma_2 \dots \sigma_n$, τότε αντικαθιστούμε αυτόν τον κανόνα με την σειρά των κανόνων: $K \rightarrow \sigma_1 B_1, K \rightarrow \sigma_2 B_2, \dots, K \rightarrow \sigma_n K'$, όπου τα $B_i, i = 1, 2, \dots, n-1$ είναι πρόσθετες βοηθητικές καταστάσεις (εκτός T φυσικά). Δεν είναι

δύσκολο να βεβαιώσει κάποιος ότι με μια τέτοια αλλαγή ούτε χάνουμε ούτε κερδίζουμε τίποτα – συνεχίζουμε να αποδεχόμαστε ακριβώς την ίδια γλώσσα όπως και πριν.

- 2^ο βήμα: Έχοντας – μετά το 1^ο βήμα – το πολύ ένα σύμβολο σε κάθε μετάβαση, παραμένει το ενδεχόμενο για κάποια κατάσταση $K \in \Sigma^*$, και σύμβολο $\sigma \in \Sigma$, να μην ορίζεται καμμία μετάβαση $(K, \sigma, K') \in \Delta$. Για αυτές τις περιπτώσεις χρησιμοποιούμε μία νέα βοηθητική κατάσταση $B_{\text{οχι}}$ (απορριπτική φυσικά, δηλαδή εκτός του συνόλου T των αποδεκτικών καταστάσεων), στην οποία οδηγούμαστε και «παγιδευόμαστε», από κάθε κατάσταση K και σύμβολο σ για τα οποία δεν ορίζεται μετάβαση, κατά τον εξής τρόπο:
 - εάν δεν ορίζεται επόμενη κατάσταση για K και σ , τότε ορίζουμε $\Delta(K, \sigma) = B_{\text{οχι}}$.
 - για κάθε $\sigma \in \Sigma$ ορίζουμε $B_{\text{οχι}} \sigma \rightarrow B_{\text{οχι}}$Αυτή η νέα βοηθητική κατάσταση δεν εμποδίζει την αποδοχή μιας λέξης που ήταν ήδη αποδεκτή, αλλά ούτε προκαλεί την αποδοχή κάποιων άλλων νέων λέξεων – επομένως συνεχίζουμε να αποδεχόμαστε ακριβώς την ίδια γλώσσα όπως και πριν.
- 3^ο βήμα: Οι «κενές» μεταβάσεις (μέσω δηλαδή της κενής λέξης) είναι δυνατόν επίσης να απαλειφούν, μία προς μία – αν και η απαλειφή αυτή απαιτεί λίγο πιο περίτεχνους χειρισμούς από ότι προηγουμένως.
 - Κατά πρώτο πρέπει να προσέξουμε ότι ίσως να υπάρχουν διαδοχικές κενές μεταβάσεις, λ.χ. $K \xrightarrow{\emptyset} K' \xrightarrow{\emptyset} K''$. Για κάθε τέτοιο ζεύγος μεταβάσεων προσθέτουμε την κενή μετάβαση $K \xrightarrow{\emptyset} K''$. Έτσι ούτε προσθέτουμε ούτε αφαιρούμε αποδεκτές λέξεις, και επί πλέον μπορούμε να ενδιαφερόμαστε μόνον για όσες λέξεις παράγονται χρησιμοποιώντας διαδοχικά μία μόνον κενή μετάβαση, (αν και ίσως χρησιμοποιούν πολλές τέτοιες, όχι όμως διαδοχικά).
 - Στη συνέχεια, για κάθε «κενή» μετάβαση, $K \xrightarrow{\emptyset} K'$, εξετάζουμε το πώς μπορεί να φθάσει το αυτόματο στη πρώτη κατάσταση K , μέσω μιας μη-κενής μετάβασης, έστω $X \xrightarrow{\sigma} K$. Αν προσθέσουμε τις νέες μεταβάσεις $X \xrightarrow{\sigma} K'$ για κάθε σχετική κατάσταση X , και αφαιρέσουμε την παλαιά (πλέον) κενή μετάβαση $K \xrightarrow{\emptyset} K'$, τότε δεν αφαιρούμε ούτε προσθέτουμε καμμία νέα αποδεκτή λέξη, διότι...
 - αν μια αποδεκτή λέξη λ χρησιμοποιεί την νέα μετάβαση, τότε αυτή η λ γινόταν και πριν αποδεκτή μέσω της παλαιάς κενής μετάβασης.
 - αν μια αποδεκτή λέξη χρησιμοποιούσε την παλαιά μετάβαση τότε στο προηγούμενο βήμα πρέπει να ακολουθούσε μια μη-κενή μετάβαση τύπου $X \xrightarrow{\sigma} K$, και επομένως γίνεται και τώρα αποδεκτή μέσω της νέας μη-κενής μετάβασης, $X \xrightarrow{\sigma} K'$.Με αυτό τον τρόπο μπορούμε να εξαλείψουμε όλες τις κενές μεταβάσεις μία-μία, τόσο τις αρχικές όσο και όσες «μεταβατικές» προσθέσαμε στη συνέχεια, χωρίς να αλλοιώσουμε το σύνολο των αποδεκτών λέξεων. Εκτός...
 - ...εάν υπάρχουν κενές μεταβάσεις από την αρχική κατάσταση $I: I \xrightarrow{\emptyset} I'$ (...). Τότε από την I μπορούμε με κενή μετάβαση να μεταβούμε σε ένα σύνολο καταστάσεων, έστω $\emptyset(I)$, κάθε μία των οποίων μπορεί να λειτουργήσει ως αρχική μετάβαση, – αφού μπορούμε να μεταβούμε σε αυτή χωρίς την ανάγνωση κανενός συμβόλου. Αρκεί λοιπόν να τις θεωρήσουμε όλες αυτές σαν αρχικές καταστάσεις, και να αναζητήσουμε ένα αιτιοκρατικό αυτόματο που θα αποδέχεται όσες λέξεις παράγονται στο αυτόματο που καταλήξαμε στο 3^ο βήμα, (το οποίο και θα συμβολίζουμε με A^+), αρχίζοντας από μια οποιαδήποτε αρχική κατάσταση και καταλήγοντας σε οποιαδήποτε αποδεκτική κατάσταση.
- 4^ο βήμα: Η κυρίως προσπάθειά μας εστιάζεται πλέον στο να αντιμετωπίσουμε το τελευταίο από τα προαναφερθέντα 4 ζητήματα: στο διάγραμμα ενός μη αιτιοκρατικού αυτομάτου, A , ίσως υπάρχουν μεταβάσεις, (έστω μη-κενές και έστω προκαλούμενες από ένα ακριβώς σύμβολο), οι οποίες δεν οδηγούν σε μία μόνο κατάσταση αλλά σε περισσότερες από μία καταστάσεις. Αυτό το γεγονός (όπως και η τελευταία παρατήρηση της προηγούμενης παραγράφου) μας αναγκάζει να θεωρήσουμε ως καταστάσεις του νέου ισοδύναμου αυτόματου, A' , ως σύνολα καταστάσεων του παλαιού. Από ένα σύνολο καταστάσεων του A μέσω ενός σύμβολου σ μπορούμε να μεταβούμε σε

ένα άλλο σύνολο καταστάσεων (πιθανά κενό!), και επομένως παρόμοιες πρέπει να είναι οι μεταβάσεις μας στο A' , εάν θέλουμε αυτό να εξομοιώσει το A . Όλες οι προηγούμενες παρατηρήσεις λαμβάνονται υπ' όψι στους εξής ορισμούς/χειρισμούς:

Ισοδύναμο A'	
Σ' αλφάβητο	το αλφάβητο Σ το αρχικού (μη-αιτιοκρατικού) αυτομάτου A^+ .
Σ'_k καταστάσεις	όλα τα υποσύνολα καταστάσεων του A^+ : $\{S : S \subseteq \Sigma_k\}$.
I' αρχική κατάσταση	το υποσύνολο $\emptyset(I)$.
T' αποδεκτικές καταστάσεις	όσα $S \subseteq \Sigma_k$ περιέχουν έστω μια τελική κατάσταση του A^+ .
Δ' οδηγίες	$\Delta'(S, \sigma) = \{K' : \text{για όλες τις } K \xrightarrow{\sigma} K' \in \Delta \text{ του } A^+\}$

Παρατηρείστε ότι...

- Για κάθε κατάσταση του εξομοιωτικού αυτομάτου A' , $S \subseteq \Sigma_k$, και για κάθε σύμβολο $\sigma \in \Sigma$, ορίζουμε μία και ακριβώς μία νέα κατάσταση $S' = \Delta'(S, \sigma)$ ως εξής: ελέγχουμε κάθε μετάβαση $K \xrightarrow{\sigma} K'$ που υπάρχει στις αρχικές οδηγίες, και για κάθε τέτοια μετάβαση προσθέτουμε την κατάσταση K' στο σύνολο καταστάσεων S' , (ως ενδεχόμενη κατάσταση στην οποία θα έφθανε το αρχικό αυτόματο A^+).
- Το παραπάνω αυτόματο, A' , είναι αιτιοκρατικό, διότι...
 - για κάθε κατάστασή του, (σύνολο $S \subseteq \Sigma_k$), και κάθε σύμβολο $\sigma \in \Sigma$, ορίζεται ακριβώς μία επόμενη κατάσταση-σύνολο $S' = \Delta'(S, \sigma)$.
 - υπάρχει μία ακριβώς αρχική κατάσταση, το σύνολο $\emptyset(I) \subseteq \Sigma_k$.
- Θεωρούμε ένα σύνολο καταστάσεων $S \subseteq \Sigma_k$ ως αποδεκτική κατάσταση εάν και μόνον περιέχει έστω μία αποδεκτική κατάσταση του αρχικού αυτομάτου A^+ .
- Το κενό σύνολο \emptyset , λειτουργεί ως κατάσταση «οριστικής αποτυχίας»: ακόμα και εάν για κάποια κατάσταση S και κάποιο σύμβολο σ , δεν υπάρχει καμία μετάβαση $K \xrightarrow{\sigma} K'$ για κανένα $K \in S$, τότε $\Delta'(S, \sigma) = \emptyset$. Γι' αυτήν την «κατάσταση» και για κάθε σύμβολο σ ισχύει $\Delta'(\emptyset, \sigma) = \emptyset$, (αυτομάτως...) Επομένως το 2^ο βήμα επεξεργασίας της παραπάνω απόδειξης, είναι, τελικά, περιττό.

Είναι τελικά το A' ισοδύναμο με το A^+ (δηλαδή και με το A); Αποδέχεται δηλαδή ακριβώς την ίδια γλώσσα; Από τα προηγούμενα σχόλια, αρκεί να δείξουμε ότι κάθε λέξη του αυτομάτου A^+ στο οποίο καταλήξαμε στο 3^ο βήμα επεξεργασίας, (βλ. παραπάνω), η οποία αρχίζει από μια οποιαδήποτε αρχική του κατάσταση και καταλήγει σε μια αποδεκτική κατάσταση, γίνεται δεκτή στο εξομοιωτικό αυτόματο A' – και μόνον αυτές.

- Έστω ότι $\lambda = \sigma_1 \dots \sigma_n$ μια τέτοια λέξη, η οποία γίνεται αποδεκτή ως ο περίπατος:

$$K_0 \xrightarrow{\sigma_1} K_1 \xrightarrow{\sigma_2} K_2 \dots \xrightarrow{\sigma_n} K_n$$

όπου, λόγω της επεξεργασίας που έχουμε κάνει, τα σ_i είναι ένα σύμβολο, και ποτέ η κενή λέξη \emptyset . Η κατάσταση K_0 είναι μία από τις «αρχικές», φθάνουμε δηλαδή σε αυτή μέσω κενών μεταβάσεων από την αρχική I , και άρα ανήκει στο σύνολο $S_0 = \emptyset(I)$, δηλαδή ανήκει στην αρχική κατάσταση του A' . Από τον ορισμό των μεταβάσεων του A' , η διάδοχη κατάσταση $S_1 = \Delta(S_0, \sigma_1)$ θα περιέχει την K_1 αφού η S_0 περιέχει την K_0 , και έχουμε στη διάθεσή μας την μετάβαση $K_0 \xrightarrow{\sigma_1} K_1$. Συνεχίζοντας έτσι, (επαγωγικά αν θέλετε), θα καταλήξουμε ότι στο τέλος το εξομοιωτικό αυτόματο A' θα βρεθεί σε κατάσταση S_n , η οποία θα περιέχει την K_n . Αυτή όμως είναι αποδεκτική (για το A^+), και το ότι περιέχεται στην S_n , αρκεί για να καταστήσει και αυτή αποδεκτική (για το A'). Άρα και στο A' υπάρχει ένας περίπατος $S_0 \xrightarrow{\sigma_1} S_1 \xrightarrow{\sigma_2} S_2 \dots \xrightarrow{\sigma_n} S_n$, από την αρχική σε μια αποδεκτική κατάσταση που παράγει την λ .

- Το προηγούμενο ισχύει και αντιστρόφως: Έστω ότι η $\lambda = \sigma_1 \dots \sigma_n$ γίνεται αποδεκτή από τον περίπατο $S_0 \xrightarrow{\sigma_1} S_1 \xrightarrow{\sigma_2} S_2 \dots \xrightarrow{\sigma_n} S_n$ στο εξομοιωτικό αυτόματο A' , όπου S_0 είναι η αρχική κατάσταση του A' , και S_n είναι μια αποδεκτική κατάσταση. Η S_n – ως αποδεκτική του A' – πρέπει να περιέχει

μια τουλάχιστον αποδεκτική κατάσταση του A^+ , έστω την K_n . Αλλά αφού έχουμε την μετάβαση $S_{v-1} \xrightarrow{\sigma_v} S_v$ στο A' , υπάρχει μια μετάβαση $K_{v-1} \xrightarrow{\sigma_v} K_v$ στο αρχικό A^+ , αλλιώς η K_v δεν θα είχε συμπεριληφθεί στην S_v . (βλ. τον ορισμό του A'). Συνεχίζοντας έτσι, (επαγωγικά), θα καταλήξουμε ότι στο αρχικό αυτόματο A^+ υπάρχει μια μετάβαση $K_0 \xrightarrow{\sigma_1} K_1$, όπου $K_0 \in S_0$. Αλλά η S_0 είναι η αρχική κατάσταση του εξομοιωτικού αυτομάτου, δηλαδή η $\emptyset(I)$, δηλαδή το σύνολο των «αρχικών» καταστάσεων το A^+ , (όσες στις οποίες φθάνουμε από την αρχική κατάσταση I μέσω κενών μεταβάσεων). Άρα και στο A^+ υπάρχει ένας περίπατος $K_0 \xrightarrow{\sigma_1} K_1 \xrightarrow{\sigma_2} K_2 \dots \xrightarrow{\sigma_v} K_v$, από κάποια «αρχική» κατάσταση $K_0 \in \emptyset(I)$ σε μια αποδεκτική κατάσταση $K_v \in T$, που παράγει την λ . ■



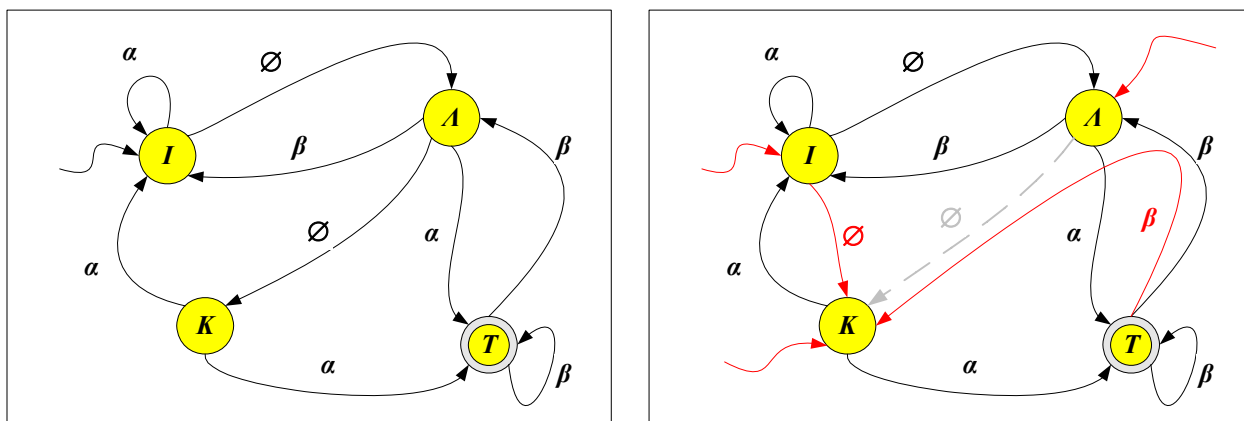
Η τεχνική και η έννοια της «εξομοίωσης».

Στο προηγούμενο θεώρημα βλέπουμε ένα οπωσδήποτε μη-τετριμμένο παράδειγμα εξομοίωσης – μιας τεχνικής που διαρκώς επανέρχεται στην επιστήμη του υπολογισμού και των αλγορίθμων: έχουμε δύο μηχανές (του ίδιου ή ακόμα και διαφορετικού τύπου) M και M' , και δείχνουμε ότι με μια εύλογη και φυσική αντιστοιχία των καταστατικών περιγραφών τους, είναι δυνατόν η 2^η από αυτές M' , όχι μόνον να επιφέρει από τα ίδια (= αντίστοιχα) δεδομένα τα ίδια (= αντίστοιχα) αποτελέσματα, αλλά κατά κάποια έννοια, τα επιφέρει **με τον ίδιο τρόπο**, ακολουθώντας δηλαδή «κατά βήμα», τα όσα θα έκανε η 1^η μηχανή M , για να επιφέρει τα δικά της αποτελέσματα από τα δικά της δεδομένα.

Αυτή η έννοια είναι κεντρικής σημασίας, διότι εάν ο τρόπος υπολογισμού μένει ο «ίδιος» (και όσο πιο «ίδιος» – αν μας επιτρέψετε την έκφραση – τόσο το καλύτερο), αυτό σημαίνει ότι οι δύο μηχανές «τρέχουν» τον ίδιο **αλγόριθμο**, και αυτή είναι η πιο βαρυσήμαντη (αλλά και σκιάδης) έννοια που διατρέχει όλη την θεωρία υπολογισμού και πολυπλοκότητας.

Παράδειγμα: Μετατροπή ενός μη-αιτιοκρατικού αυτομάτου σε αιτιοκρατικό.

Δίνουμε στη συνέχεια ένα παράδειγμα μετατροπής του μη-αιτιοκρατικού αυτομάτου του επόμενου σχήματος σε αιτιοκρατικό, υπολογίζοντας τις οδηγίες, $\Delta'(\sigma, S)$, του νέου αιτιοκρατικού αυτομάτου.



Σχήμα: Μετατροπή μη-αιτιοκρατικού αυτομάτου σε αιτιοκρατικό.

Το 1^ο βήμα δεν εμφανίζεται (και ούτως ή άλλως είναι το πιο απλό από όλα).

Το 2^ο βήμα είδαμε ότι τελικά είναι περιττό, (αφού υλοποιείται «αυτομάτως» από τα υπόλοιπα).

Το 3^ο βήμα παράγει εδώ μία σύνθετη μετάβαση (I, \emptyset, K) ως αποτέλεσμα των (I, \emptyset, A) και (A, \emptyset, K) , με αποτέλεσμα οι «αρχικές» καταστάσεις να είναι οι $\{I, K, A\}$. Στη συνέχεια απαλείφουμε μία-προς-μία τις κενές μεταβάσεις. Στο σχήμα δείχνουμε μόνο την απαλειφή της (A, \emptyset, K) : εξετάζοντας ποιές μη-κενές μεταβάσεις της μορφής $(-, -, A)$ υπάρχουν, εντοπίζουμε την (T, β, A) (μόνον), άρα προσθέτουμε την σύνθετη μετάβαση (T, β, K) (ερυθρή ακμή στο σχήμα). Με την απαλειφή των υπολοίπων κενών

μεταβάσεων (οι σχετικές ακμές δεν εικονίζονται στο σχήμα), καταλήγουμε στους παρακάτω δύο πίνακες μεταβάσεων, έναν μέσω α και έναν μέσω β . Οι «κίτρινες» μεταβάσεις είναι όσες προστίθενται λόγω κενών μεταβάσεων.

$\xrightarrow{\alpha}$	I	K	Λ	T
1	I	*	*	*
2	K	*	*	*
3	Λ			*
4	T			

$\xrightarrow{\beta}$	I	K	Λ	T
1	I			
2	K			
3	Λ	*	*	*
4	T		*	*

Το 4^ο βήμα, θέλει απλώς υπομονή: για κάθε υποσύνολο S του $\{I, K, \Lambda, M\}$ εξετάζουμε, για όλες τις καταστάσεις $X \in S$, όλες τις καταστάσεις Y στις οποίες οδηγούμαστε με μεταβάσεις (X, α, Y) , και τις περισυλλέγουμε σε μια νέα κατάσταση $S' = \Delta(S, \alpha)$. Το ίδιο και για τις μεταβάσεις μέσω β , τις $\Delta(S, \beta)$. Οι γραμμές 1 έως 4 ουσιαστικά επαναλαμβάνουν τους προηγούμενες πίνακες.

S:	I	K	Λ	T	$\Delta'(S, \alpha)$	$\Delta'(S, \beta)$
0	-	-	-	-	\emptyset	\emptyset
1	*	-	-	-	{I, K, Λ }	{}
2	-	*	-	-	{I, K, Λ }	{}
3	-	-	*	-	{T}	{I, K, Λ }
4	-	-	-	*	{}	{K, Λ , T}
5	*	*	-	-	{I, K, Λ }	{}
6	*	-	*	-	{I, K, Λ , T}	{I, K, Λ }
7	*	-	-	*	{I, K, Λ }	{K, Λ , T}
8	-	*	*	-	{I, K, Λ , T}	{I}
9	-	*	-	*	{I, K, Λ }	{K, Λ , T}
10		-	*	*	{T}	{I, K, Λ }
11	*	*	*	-	{I, K, Λ , T}	{I, K, Λ , T}
12	*	*	-	*	{I, K, Λ }	{K, Λ , T}
13	*	-	*	*	{I, K, Λ , T}	{I, K, Λ , T}
14	-	*	*	*	{I, K, Λ }	{I, K, Λ , T}
15	*	*	*	*	{I, K, Λ , T}	{I, K, Λ , T}

Η αφηρητική κατάσταση-σύνολο είναι χρωματισμένη με κίτρινο, και οι τερματικές με γκρι χρώμα. ■

7^ο Πεπερασμένα αυτόματα: τα θεωρήματα «κλειστότητας».



Η «θετική» προσέγγιση: από παλιές ομαλές γλώσσες σε νέες.

Η «κλειστότητα» είναι μια μαθηματική έννοια που επανέρχεται συνεχώς και είναι ιδιαίτερα βαρυσήμαντη. Το πεδίο εφαρμογής της είναι μια συλλογή «αντικειμένων» X επί των οποίων ορίζονται διάφορες κατασκευαστικές πράξεις: π.χ. επί των πραγματικών αριθμών ορίζεται η πράξη της διαίρεσης ή της κυβικής ρίζας.

Το επανερχόμενο ερώτημα είναι το εξής: εάν από διάφορα αντικείμενα της συλλογής X που έχουν μια πρόσθετη ενδιαφέρουσα ιδιότητα I κατασκευάσουμε μέσω μιας πράξης $\#$, ένα άλλο αντικείμενο της X , τότε αυτό θα έχει εκ νέου την ιδιότητα I ή ίσως όχι. Π.χ. οι ρητοί αριθμοί είναι «κλειστοί» ως προς την πράξη της διαίρεσης· δεν είναι όμως «κλειστοί» ως προς την πράξη της κυβικής ρίζας.

Επί των γλωσσών ορίζονται κατά φυσικό και εύλογο τρόπο διάφορες «πράξεις», είτε συνολοθεωρητικού, είτε γραμματικο-συντακτικού χαρακτήρα. Οι ομαλές γλώσσες είναι κλειστές ως προς αυτές τις πράξεις ή όχι; Αυτό θα εξετάσουμε στην τρέχουσα ενότητα.

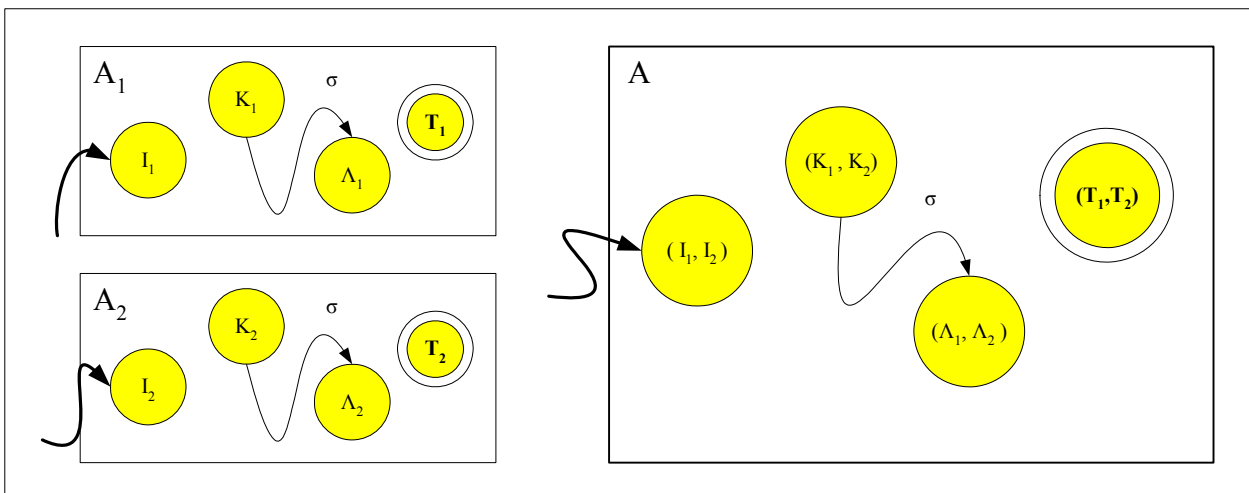
Να σημειώσουμε εδώ πως αυτή η ανάλυση θα έχει «θετικό» χαρακτήρα: θα μας δείξει ότι εάν έχουμε κάποιες ομαλές γλώσσες τότε είναι δυνατόν από αυτές να ορίσουμε ή/και συνθέσουμε νέες, επίσης ομαλές, γλώσσες.

Θεώρημα: Κλειστότητα ως προς συνολοθεωρητικές πράξεις.

Ισχυρισμός: Γιά κάθε αιτιοκρατικό αυτόματο A , (ή δύο αυτόματα A_1, A_2) που αποδέχονται τις γλώσσες $L(A), L(A_1), L(A_2)$ υπάρχει ένα αυτόματο που αποδέχεται τις εξής:

- (1) $L(A_1) \cap L(A_2)$ («τομή»)
- (2) $L(A_1) \cup L(A_2)$ («ένωση»)
- (3) $\Sigma^* - L(A)$ («συμπλήρωμα»)

Σχέδιο απόδειξης:



Σχήμα: Σύνθεση δύο αυτομάτων για την αποδοχή της τομής, $L(A_1) \cap L(A_2)$.

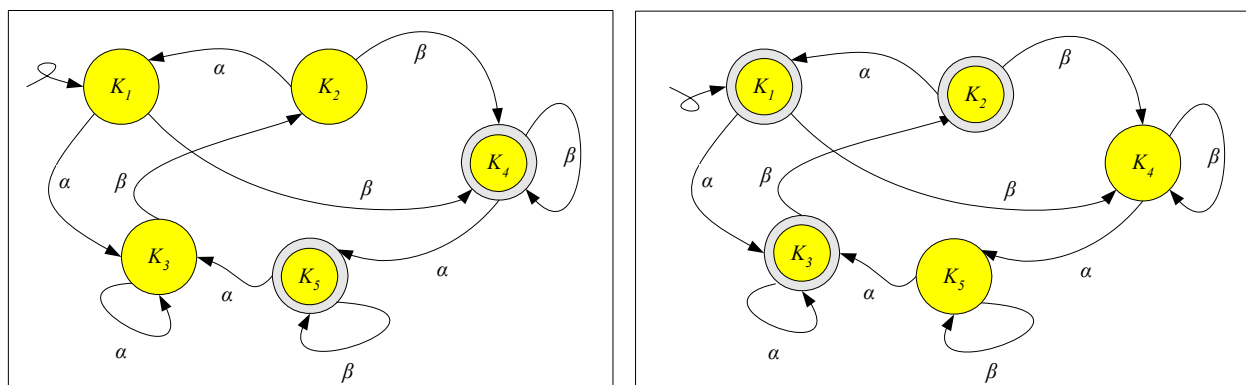
1) Ορίζουμε αυτόματο του οποίου οι καταστάσεις συμβολίζουν ζεύγη καταστάσεων (K_1, K_2) των A_1, A_2 . Για κάθε κατάσταση (K_1, K_2) και κάθε σύμβολο σ πηγαίνουμε στη κατάσταση - ζεύγος:

$$\Delta((K_1, K_2), \sigma) = (\Delta_1(K_1, \sigma), \Delta_2(K_2, \sigma))$$

δηλαδή ακολουθούμε τις μεταβάσεις Δ_1 του A_1 στο πρώτο μέλος του ζεύγους (K_1, K_2) , και τις μεταβάσεις Δ_2 του A_2 στο δεύτερο μέλος του ζεύγους. Η αρχική κατάσταση είναι η (I_1, I_2) και τερματικές καταστάσεις είναι όλα τα ζεύγη και τα δύο μέλη των οποίων είναι τερματικές καταστάσεις στα A_1 και A_2 (αντιστοίχως). Ένας περίπατος στο νέο αυτόματο αντιστοιχεί σε (δύο) περιπάτους στα A_1 και A_2 και έτσι γίνεται αποδεκτό ότι μόνο ότι θα γινόταν αποδεκτό και από τα δύο αυτόματα.

2) Φτιάχνουμε ένα αυτόματο όπως στη περίπτωση (1), μόνο που τώρα τερματικές καταστάσεις είναι όσα ζεύγη καταστάσεων (στο νέο αυτόματο) περιέχουν έστω μία τερματική κατάσταση των A_1, A_2 . Έτσι αποδεκτές γίνονται όσες και μόνον λέξεις γίνονται αποδεκτές από ένα τουλάχιστον των A_1, A_2 .

3) Το ίδιο αυτόματο, μετατρέποντας τις μη-τερματικές καταστάσεις σε τερματικές και αντίστροφα, αποδέχεται ότι το A δεν αποδέχεται άρα αποδέχεται την $\Sigma^* - L(A)$. Προσέξτε ότι αυτό οφείλεται στην αιτιοκρατικότητα του αρχικού αυτομάτου, χάρι στην οποία μια λέξη λ όταν γίνεται αποδεκτή γίνεται χάρι στην μία ακριβώς τερματική κατάσταση που καταλήγει. Εάν λόγω μη αιτιοκρατικότητας μια λέξη λ κατέληγε λ.χ. σε δύο καταστάσεις, σε μια K (αποδεκτική) και σε μια άλλη K' (απορριπτική), τότε η λ θα ήταν αποδεκτή στο A (λόγω της K), αλλά θα ήταν αδύνατον να αντιστρέψουμε τον ρόλο των K και K' διότι τότε θα η λ θα παρέμενε αποδεκτή στο νέο αυτόματο (λόγω της K' ...), και έτσι η λ θα ανήκε τόσο στην $L(A)$ όσο και στο (υποτιθέμενο) συμπλήρωμά της - πράγμα που θα ήταν προφανώς εσφαλμένο.



Σχήμα: Μετατροπή αιτιοκρατικού αυτομάτου για την αποδοχή του συμπληρώματος, $\Sigma^* - L(A)$.

Μπορούμε, επίσης, να ορίσουμε νέες γλώσσες από παλιές και μέσω γραμματικών πράξεων. Οι τρεις πιο προφανείς τρόποι περιγράφονται παρακάτω:

$L(A_1) \parallel L(A_2)$	«παράθεση»	των $L(A_1) \parallel L(A_2)$	$L = \{ \lambda : \lambda = xy, x \in L(A_1) \text{ και } y \in L(A_2) \}$
$L(A)^*$	«επανάληψη»	της $L(A)$, ≥ 0 φορές	$L = \{ \lambda : \lambda = \lambda_1 \dots \lambda_n, n \geq 0, \text{ και } \lambda_k \in L(A), 0 \leq k \leq n \}$
$L(A)^+$	«επανάληψη»	της $L(A)$, ≥ 1 φορές	$L = \{ \lambda : \lambda = \lambda_1 \dots \lambda_n, n \geq 1, \text{ και } \lambda_k \in L(A), 0 \leq k \leq n \}$
$L(A)^R$	«κατοπτρική»	της $L(A)$	$L = \{ \lambda^R : \lambda \in L(A) \}$

(Ένας κομψός επαγωγικός ορισμός της κατοπτρικής λέξης λ^R για $\lambda \in \Sigma^*$ είναι:

$$\text{εάν } \lambda = \emptyset \text{ τότε } \lambda^R = \emptyset, \text{ αλλιώς } \lambda = \lambda' \sigma \text{ για } \lambda \in \Sigma^*, \sigma \in \Sigma, \text{ και } (\lambda)^R = (\lambda' \sigma)^R = \sigma (\lambda'^R).$$

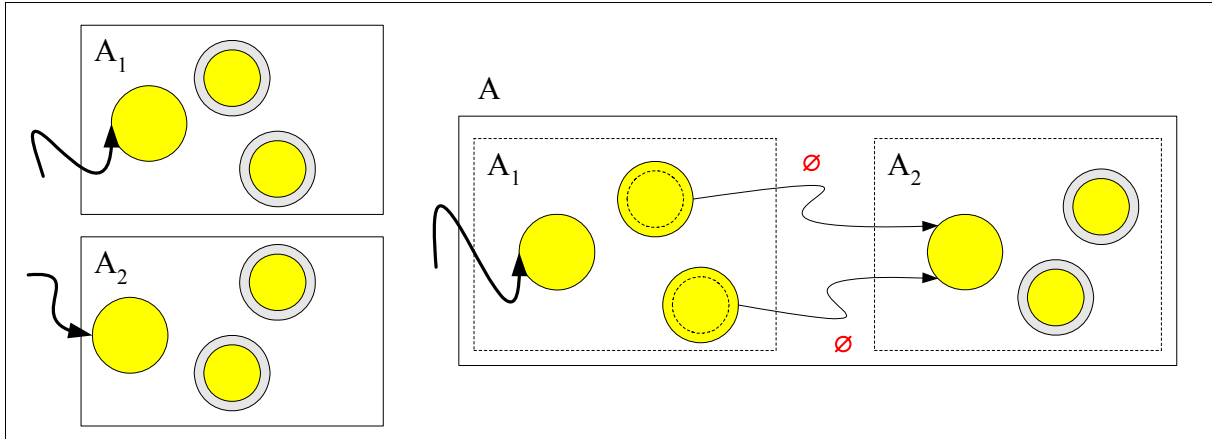
Θεώρημα: Κλειστότητα ως προς «γραμματικές» πράξεις).

Ισχυρισμός: Για οποιαδήποτε αυτόματα A, A_1, A_2 που αποδέχονται τις γλώσσες $L(A), L(A_1), L(A_2)$ υπάρχει αυτόματο, A' , το οποίο αναγνωρίζει τις γλώσσες:

- (1) $L(A_1) \parallel L(A_2)$ («παράθεση»)
- (2) $L(A)^*$ («αόριστη επανάληψη»)
- (3) $L(A)^R$ («κατοπτρική»)

Σχέδιο Απόδειξης:

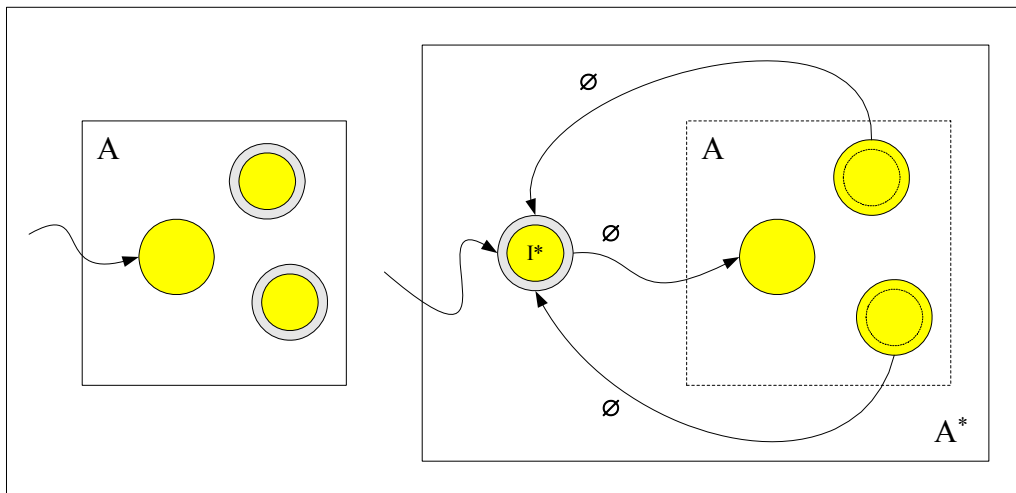
1) Υποθέτουμε χωρίς απώλεια γενικότητας ότι τα δύο αυτόματα δεν χρησιμοποιούν ίδια ονόματα για τις καταστάσεις τους ώστε να μην έχουμε σύγχυση – (αλλιώς απλώς τις μετονομάζουμε καταλλήλως ώστε να είναι όλες διαφορετικές μεταξύ τους). Ενώνουμε τις οδηγίες των δύο αυτομάτων σε ένα ενιαίο σύνολο. Νέα αρχική κατάσταση είναι η του A_1 , και νέες αποδεκτικές καταστάσεις είναι οι του A_2 . Οι αποδεκτές καταστάσεις του A παύουν να είναι αποδεκτές και οδηγούν με την κενή λέξη \emptyset στην αρχική κατάσταση του A η οποία παύει να θεωρείται φυσικά αρχική.



Σχήμα: Σύνθεση δύο αυτομάτων για την αποδοχή της παράθεσης, $L(A_1) \parallel L(A_2)$.

2) Η αρχική και οι αποδεκτικές καταστάσεις του A παύουν να είναι τέτοιες, και προσθέτουμε τα εξής:

- Μια νέα αρχική κατάσταση που είναι ταυτόχρονα και τερματική, ώστε να αποδεχθούμε την κενή λέξη (δηλαδή τις μηδέν επαναλήψεις της $L(A)$).
- Μια κενή μετάβαση από την νέα αρχική κατάσταση στην πρώην αρχική κατάσταση του A ώστε να μπορούμε να αποδεχθούμε μια λέξη που θα αποδεχόταν το A .
- Κενές μεταβάσεις από κάθε πρώην αποδεκτική κατάσταση στην νέα αρχική (και τερματική) κατάσταση ώστε μετά από μία λέξη αποδεκτή από το A να μπορούμε να αναγνωρίσουμε οσοδήποτε άλλες «εξ αρχής».

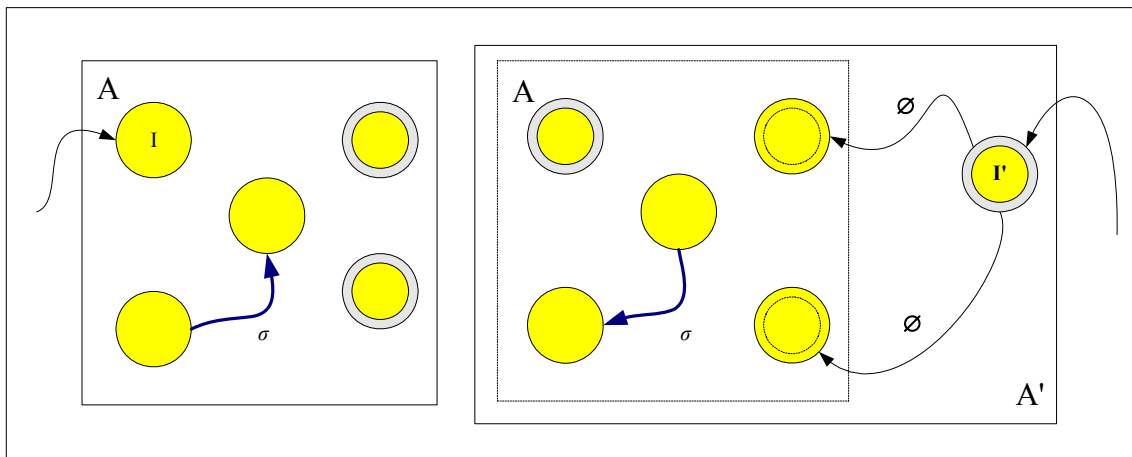


Σχήμα: Σύνθεση δύο αυτομάτων για την αποδοχή της αόριστης επανάληψης, $L(A)^*$.

Ο χειρισμός της $L(A)^*$ είναι εντελώς ανάλογος.

3) Αντιστρέφουμε τις φορές όλων των μεταβάσεων, κάθε μετάβαση $K \sigma \rightarrow K'$ αντιστρέφεται σε $K' \sigma \rightarrow K$. Η αρχική κατάσταση I , γίνεται η μόνη νέα τερματική κατάσταση $T' = \{I\}$, και οι πρώην τερματικές καταστάσεις T , παύουν να είναι πλέον τέτοιες και γίνονται «αρχικές» με την προσθήκη μία νέας αρχικής κατάστασης I' , η οποία μέσω της κενής λέξης οδηγεί σε αυτές. Κατ' αυτόν τον τρόπο κάθε

περίπατος στο νέο αυτόματο, εάν ληφθεί αντιστρόφως («κατοπτρικά»), αντιστοιχεί σε ταυτόσημο περίπατο (αντιστοιχεί στην ίδια λέξη) στο αρχικό αυτόματο.



Σχήμα: Μετατροπή ενός αυτόματου για την αποδοχή της κατοπτρικής, $L(A)^R$.



Η «αναλυτική» αξιοποίηση των θεωρημάτων κλειστότητας.

Τα θεωρήματα κλειστότητας έχουν κατασκευαστικό χαρακτήρα, καθώς μας λένε ότι εάν έχουμε κάποιες ομαλές/κανονικές γλώσσες, τότε μπορούμε από αυτές να ορίσουμε νέες και επίσης ομαλές, γλώσσες και μάλιστα να «κατασκευάσουμε» τα σχετικά αυτόματα. Στη πράξη όμως, πολλές φορές δεν θα χρειαστεί να ορίσουμε νέες γλώσσες, αλλά έχοντας ήδη κάποια γλώσσα L – ορισμένη με άλλους τρόπους και όχι μέσω ενός πεπερασμένου αυτομάτου – θα χρειαστεί συχνά να διαπιστώσουμε ακριβώς το εάν αυτή μπορεί να οριστεί ή όχι μέσω ενός αυτομάτου.

Σε τέτοιες περιπτώσεις περίπτωση η γενικότερη μέθοδος που καλείται σε χρήση είναι η διάσπαση του προβλήματος σε μικρότερα, και η κατασκευή της συνολικής λύσης από τις επί μέρους λύσεις. Η μέθοδος αυτή δεν αφορά μόνο στην θεωρία υπολογισμού, αλλά σε κάθε σχεδόν πρακτική, επιστημονική, τεχνολογική, μαθηματική, ή και απλά καθημερινή δραστηριότητα. Η διάσπαση ενός προβλήματος σε «μικρότερα» μπορεί να γίνει με όποιο τρόπο επιδέχεται η φαντασία μας, αλλά η ανησυχία που προκαλεί αυτή η μέθοδος είναι το εάν θα είναι δυνατή η σύνθεση των επί μέρους λύσεων.

Τα θεωρήματα κλειστότητας έρχονται να απαντήσουν αυτό το ερώτημα, διότι «μας λένε» ότι εάν για την αποσύνθεση μιας γλώσσας σε άλλες χρησιμοποιήσουμε τις πράξεις \cap , \cup , $-$, ή τις πράξεις $()^*$, \parallel , $()^R$, τότε η σύνθεση θα είναι δυνατή και μάλιστα κατά τον επιθυμητό τρόπο: εάν οι επί μέρους γλώσσες προκύψουν ομαλές, τότε και η αρχική γλώσσα θα είναι επίσης ομαλή και θα αναγνωρίζεται/παράγεται από κάποιο αυτόματο.

Παράδειγμα: ένα αυτόματο που καταλαβαίνει ποιοι δυαδικοί διαιρούνται δια 6.

Έχουμε δει προηγουμένως ότι υπάρχει αυτόματο που αποδέχεται την γλώσσα $L_{3x} \subseteq \{0,1\}^*$ των (δυαδικών) πολλαπλασίων του 3. Είναι πολύ απλό να δείτε ότι και η L_{2x} είναι επίσης ομαλή γλώσσα: πρέπει και αρκεί το τελευταίο ψηφίο να είναι '1'. Τί θα κάναμε όμως για την L_{6x} ; Μια σκέψη 2-3 δευτερολέπτων μας λύνει το πρόβλημα: η L_{6x} είναι η τομή των L_{2x} και L_{3x} – διότι ένας αριθμός διαιρείται δια 6 εάν και μόνον διαιρείται και δια 2 και δια 3, δηλαδή.

$$L_{6x} = L_{2x} \cap L_{3x}$$

Επομένως, από τα θεωρήματα κλειστότητας, και η L_{6x} είναι ομαλή γλώσσα.

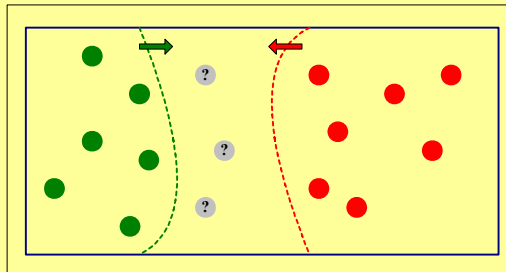
8^ο Πεπερασμένα αυτόματα: το «λήμμα άντλησης» και περιορισμοί.



Η «αρνητική» προσέγγιση: γλώσσες μη αποδεκτές από κανένα αυτόματο.

Το εξής ερώτημα έχει βαρύνουσα σημασία – και στη περίπτωση των αυτομάτων αλλά και σε (πολλές) ανάλογες περιπτώσεις: πώς μπορούμε να προσδιορίσουμε την υπολογιστική ισχύ αυτών των μηχανών που ονομάσαμε «αυτόματα»; Μήπως λ.χ. για κάθε γλώσσα υπάρχει κάποιο πεπερασμένο αυτόματο που την αναγνωρίζει; Θα δούμε πώς τα πράγματα δεν έχουν έτσι, αντιθέτως είναι πολύ εύκολο να ορίσει κάποιος γλώσσες που δεν γίνονται αποδεκτές από κανένα αυτόματο.

Να σημειώσουμε πως εδώ η ανάλυση θα έχει «αρνητικό» χαρακτήρα: θα μας δείξει ότι υπάρχουν γλώσσες που δεν γίνονται αποδεκτές από κανένα πεπερασμένο αυτόματο. Ας φανταστούμε ένα πλαίσιο, στο οποίο κάθε κουκίδα αναπαριστά μία γλώσσα επί αλφαβήτου έστω Σ . Αυτό που κάναμε στην προηγούμενη ενότητα είναι να χαράξουμε ένα «θετικό» όριο (βλ. επόμενο σχήμα, αριστερά) εντός του οποίου κάθε γλώσσα είναι ομαλή, και να «σπρώξουμε» αυτό το όριο όσο μπορούσαμε προς τα δεξιά.



Σχήμα: Γλώσσες $L \subseteq \Sigma^*$, και δύο όρια «χαρακτηρισμού» ως ομαλές ή μή.

Αυτό που θα κάνουμε στη τρέχουσα ενότητα είναι να χαράξουμε ένα «αρνητικό» όριο (βλ. σχήμα, δεξιά) εντός του οποίου κάθε γλώσσα δεν είναι ομαλή, και να «σπρώξουμε» αυτό το όριο όσο μπορούσαμε προς τα αριστερά.

Αλλά αυτό το έργο δεν είναι συμμετρικό με το προηγούμενο. Μια γλώσσα γ είναι ομαλή εάν υπάρχει έστω ένα αυτόματο α ικανό να την αποδέχεται, αλλά είναι μη-ομαλή, εάν κάθε αυτόματο είναι ανίκανο να την αποδεχθεί (αυτήν ακριβώς). Συγκεκριμένα, έστω ότι $\Phi(x, y)$ είναι η μαθηματική πρόταση που διατυπώνει τον ισχυρισμό ότι το (αυτόματο) x αποδέχεται την (γλώσσα) y .

■ Το ότι μια γλώσσα γ είναι ομαλή, γράφεται: $\exists x \Phi(x, \gamma)$ (#1)

■ Το ότι μια γλώσσα γ δεν είναι ομαλή, γράφεται: $\forall x \neg \Phi(x, \gamma)$ (#2)

Υπάρχει εδώ μια δομική-λογική διαφορά ανάμεσα στους δύο ισχυρισμούς.

■ Ο 1^{ος} είναι υπαρξιακός (\exists): Για να τον αποδείξουμε αρκεί να φέρουμε έστω ένα παράδειγμα- α , ότι δηλαδή για κάποιο $x \leftarrow \alpha$, αληθεύει ο ισχυρισμός $\Phi(\alpha, \gamma)$.

■ Ο 2^{ος} είναι καθολικός (\forall): Για να τον αποδείξουμε δεν αρκεί να φέρουμε όχι ένα, αλλά ούτε εκατομμύρια παραδείγματα α τέτοια ώστε $\neg \Phi(\alpha, \gamma)$.

Η πιο συνηθισμένη τεχνική για να δείξουμε ότι ισχύει μια πρόταση σαν την #2, είναι να «απαλείψουμε» το αυτόματο x , να δείξουμε ότι η ισχύς της $\Phi(x, \gamma)$ έχει μια λογική συνέπεια για τη γλώσσα γ – ότι δηλαδή, την αναγκάζει να έχει μια γλωσσική ιδιότητα $\Psi(\gamma)$: $\forall \alpha \forall \gamma (\Phi(\alpha, \gamma) \rightarrow \Psi(\gamma))$ (#3). Στη συνέχεια αρκεί και πάλι να δώσουμε απλώς ένα «αντι-παράδειγμα» μιας γλώσσας ξ που δεν έχει αυτή την ιδιότητα, ότι δηλαδή $\neg \Psi(\xi)$. Το ξ θα αποδείκνυε το ζητούμενο «με απαγωγή στο άτοπο».

Η απόδειξη μιας πρότασης σαν την (#3) φαίνεται λίγο πιο εύκολο καθήκον από την απόδειξη της (#2) διότι το συμπέρασμα αφορά σε ένα αντικείμενο (τη γλώσσα γ), και διότι η υπόθεσή της (#3) είναι κάπως πιο ισχυρή από εκείνη της (#2): άλλο να υποθέτεις απλώς ότι το x είναι ένα απλώς αυτόματο, και άλλο να υποθέτεις ότι το x

είναι αυτόματο και αποδέχεται την γλώσσα y (βλ. $\Phi(\alpha, \gamma)$).

Λέμε «λίγο πιο εύκολο» – διότι δεν πρέπει να μας διαφύγει το πόσο «βαρεια» σημασία έχει ο ισχυρισμός $\Phi(x, \gamma) \rightarrow \Psi(\gamma)$: λέει ότι υπάρχει μια «κοινή» ιδιότητα $\Psi(-)$ που έχουν όλες οι ομαλές γλώσσες – παρόλο που αυτές είναι άπειρες στο πλήθος, οι «περισσότερες» από αυτές έχουν άπειρο πλήθος λέξεων, και τα αλφάβητά τους, αν και πεπερασμένα μπορεί να έχουν οσοδήποτε μεγάλο μέγεθος. Το ερχόμενο «λήμμα άντλησης» θα μας προσφέρει ακριβώς μια τέτοια ιδιότητα.

Θεώρημα: «το λήμμα άντλησης»

Ισχυρισμός: Εάν μια γλώσσα $L(A)$, αναγνωρίζεται από αιτιοκρατικό αυτόματο $A = \langle \Sigma, \Sigma_k, I, T, \Delta \rangle$ τότε:

(α) κάθε λέξη επαρκώς μεγάλου μήκους λέξη $\lambda \in L(A)$, (συγκεκριμένα: $|\lambda| \geq |\Sigma_k|$), γράφεται ως:

$$\lambda = \alpha \mu \mu \mu \tau \dots \alpha \text{ ή } \lambda = \alpha \mu^{(k)} \tau$$

όπου τα α, μ, τ είναι λέξεις του Σ^* , και $\mu \neq \emptyset$, (ή $|\mu| \geq 1$).

(β) κάθε λέξη της μορφής $\alpha \mu^{(i)} \tau, i = 0, 1, 2, \dots$ ανήκει στην γλώσσα $L(A)$.

Σχέδιο Απόδειξης: Κάθε λέξη λ μήκους $n = |\lambda| \geq |\Sigma_k|$ που γίνεται αποδεκτή από το A , παράγεται από κάποιο περίπατο εντός του αυτομάτου, $I = K_0 \xrightarrow{\sigma_1} K_1 \xrightarrow{\sigma_2} K_2 \dots \xrightarrow{\sigma_n} K_n \in T$. Αυτές όμως έχουν πλήθος $(n+1) > |\Sigma_k|$, ενώ το πλήθος των καταστάσεων είναι μόνον $|\Sigma_k|$. Επομένως δύο από αυτές ταυτίζονται⁶ έστω οι K_i και K_j : $K_i = K_j$. Η λέξη λ λοιπόν αντιστοιχεί σε τρεις υπολέξεις α, μ, τ , (αρχή-μέση-τέλος):

- στη λέξη α που παράγεται από την διαδρομή μέχρι την πρώτη εμφάνιση της κατάστασης K_i .
- στη λέξη μ που παράγεται από την επανάληψη $k \geq 1$ φορές ενός «κυκλικού» περιπάτου $K_i \rightarrow K_i$, και
- στη λέξη τ που παράγεται από την τελευταία εμφάνιση της K_i έως την κατάσταση K_n .

Επομένως η λέξη λ έχει την μορφή $\alpha \mu^{(k)} \tau$. Ακολουθώντας την ίδια διαδρομή με μόνη διαφορά την επανάληψη του κυκλικού περιπάτου $K_i \rightarrow K_i$ οποιοδήποτε πλήθος φορών $k \geq 0$, θα έχουμε ότι κάθε λέξη της μορφής $\alpha \mu^{(k)} \tau$, για $k = 0, 1, 2, \dots$ γίνεται επίσης αποδεκτή.

Προσοχή: τα παραπάνω δεν εξασφαλίζουν ότι κάθε λέξη της $L(A)$ έχει την μορφή $\alpha \mu^{(k)} \tau$, αλλά μόνον το αντίστροφο: κάθε λέξη με αυτή τη μορφή είναι λέξη της $L(A)$. ■

Πόρισμα: μια «θεμελιακή» μη-ομαλή γλώσσα.

Ισχυρισμός: Δεν υπάρχει πεπερασμένο αυτόματο το οποίο να αποδέχεται την γλώσσα:

$$L_? = \{ x^{(v)} y^{(v)} : v \geq 0 \} \quad (x \neq y)$$

Σχέδιο Απόδειξης: Έστω ότι η $L_?$ είναι ομαλή γλώσσα. Τότε κατά το προηγούμενο λήμμα μία επαρκώς μεγάλη λέξη λ της $L_?$ (και υπάρχει τέτοια αφού η $L_?$ έχει λέξεις τόσο μακρές όσο θέλουμε), θα πρέπει να γράφεται ως $\lambda = \alpha \mu^{(k)} \tau$, και ότι όλες οι λέξεις της μορφής $\alpha \mu^{(k)} \tau$, για κάθε $k \geq 0$ θα πρέπει να ανήκουν στην $L_?$, θα είναι δηλαδή όλες της μορφής $\alpha^{(v)} \beta^{(v)}$ (για κάποιο v). Το επαναλαμβανόμενο όμως μεσαίο μέρος μ θα είναι $2+1$ ειδών:

- Αποτελείται μόνο από «x». Αυτό δεν είναι όμως δυνατόν διότι εάν η $\lambda = \alpha \mu^{(k)} \tau$ περιέχει ίσο πλήθος από «x» και «y» τότε η $\alpha \mu^{(k+1)} \tau$ (με ένα μεσαίο τμήμα μ επί πλέον) δεν μπορεί να έχει πια ίσο πλήθος «x» και «y» – θα έχει περισσότερα x από y.
- Αποτελείται μόνο από «y». Κατ' αναλογία με το προηγούμενο, ούτε αυτό είναι δυνατόν.
- Περιλαμβάνει και «x» και «y», είναι δηλαδή της μορφής $\mu = \dots xy\dots$, (ή $\mu = \dots yx\dots$). Αυτό δεν είναι όμως δυνατόν διότι τότε η λέξη $\alpha \mu^{(2)} \tau$ (που κατά το λήμμα άντλησης επίσης ανήκει στη γλώσσα $L(A)$), δεν έχει όλα τα «x» στην αρχή και όλα τα «y» στο τέλος, αφού $\mu\mu = \dots xy\dots xy\dots$ (ή $\dots yx\dots yx\dots$).

Καταλήγουμε λοιπόν σε όλες τις περιπτώσεις σε άτοπο, επομένως η $L_?$ δεν μπορεί να είναι ομαλή γλώσσα. Με εντελώς ανάλογο τρόπο μπορούμε να δείξουμε πως ούτε η γλώσσα $\{ x^{(v)} y^{(v)} : v \geq 0 \}$ είναι ομαλή. ■

⁶ Όπως γνωρίζουμε από τα διακριτά μαθηματικά και «την αρχή του περιστερώνα» ή «του Dirichlet».

Το λήμμα άντλησης καθίσταται πιο αποδοτικό και χρήσιμο να συνδυαστεί «έξυπνα» με τα θεωρήματα κλειστότητας. Δείτε τα εξής παραδείγματα:

Πόρισμα-παράδειγμα: η γλώσσα των «ισορροπημένων παρενθέσεων» είναι μη-ομαλή.

Ισχυρισμός: Η γλώσσα $\Gamma_{\square} \subseteq \{ [,] \}^*$ των «ισορροπημένων» παρενθέσεων δεν είναι ομαλή. (ισορροπημένες παρενθέσεις: μια σειρά παρενθέσεων, λ.χ. $[[[]]] [[[]]]$], όπου κάθε παρένθεση «ανοίγει» και «κλείνει» σωστά).

Σχέδιο απόδειξης: Η γλώσσα Γ_{\square} περιέχει το σύνολο των λέξεων $\{ [^{(v)}]^{(v)} : v \geq 1 \}$, γλώσσα που μόλις πριν είδαμε ότι δεν είναι ομαλή. Ας προσέξουμε όμως εδώ ότι αυτό δεν εξασφαλίζει την μη-ομαλότητα της Γ_{\square} : αν μια γλώσσα Γ περιέχει μια άλλη μη-ομαλή αυτό δεν σημαίνει και ότι η ίδια δεν είναι ομαλή... Π.χ. η γλώσσα $\Gamma = \{x, y\}^*$, που περιέχει όλες τις δυνατές λέξεις επί του αλφαβήτου $\{x, y\}$, είναι ομαλή, και το αυτόματο που την αναγνωρίζει είναι το πιο απλό από όλα... (ποιός). Και όμως αυτή η γλώσσα Γ περιέχει την ανώμαλη γλώσσα $L_2 = \{x^{(v)} y^{(v)}\}$ – (και για την ακρίβεια: περιέχει κάθε ανώμαλη γλώσσα επί του $\{x, y\}$!).

Αυτό που κάνει μια γλώσσα ανώμαλη δεν είναι το μέγεθος της αλλά η «μορφή» της – μιλώντας κάπως διαισθητικά. Γι’ αυτό θα χρησιμοποιήσουμε μια γλώσσα Λ που είναι ομαλή και που μας βοηθάει να διευκρινίσουμε με ακρίβεια και αυστηρότητα αυτό που φταίει στη γλώσσα των ισορροπημένων παρενθέσεων. Η γλώσσα αυτή έχει την μορφή $\Lambda = \{ [^{(k)}]^{(\lambda)}, k, \lambda \geq 0 \}$ και είναι μια γλώσσα με λέξεις από παρενθέσεις όπου στη αρχή υπάρχει ένα οποιοδήποτε πλήθος από αριστερές παρενθέσεις, και στο τέλος ένα οποιοδήποτε πλήθος από δεξιές παρενθέσεις – όχι κατ’ ανάγκη το ίδιο. Είναι εύκολο να βεβαιώσουμε ότι αυτή η γλώσσα είναι ομαλή. Αλλά η τομή της με την γλώσσα των ισορροπημένων παρενθέσεων Γ_{\square} είναι μια ανώμαλη γλώσσα (!) – η $\Gamma_2 = \{ [^{(v)}]^{(v)} \}$:

$$\Gamma_2 = \Gamma_{\square} \cap \Lambda$$

Και εδώ συνεργούν τα θεωρήματα κλειστότητας: η Λ είναι ομαλή, και εάν ήταν και η Γ_{\square} , τότε θα ήταν και η τομή τους, δηλαδή η Γ_2 , αλλά γνωρίζουμε ήδη ότι αυτή δεν είναι – άρα έχουμε άτοπο: ούτε η Γ_{\square} μπορεί να είναι ομαλή. Η γλώσσα Λ μας χρησίμευσε για να αποσπάσουμε από την υπό εξέταση γλώσσα ένα υποσύνολο που θα «έπρεπε» να είναι ομαλό διότι, ως προς ένα τουλάχιστον χαρακτηριστικό του, είναι ομαλό – (εδώ: όλα τα ‘[’ αριστερά και όλα τα ‘]’ δεξιά – ασχέτως πλήθους).

Παράδειγμα: Οι «πρώτοι» είναι μη-ομαλή γλώσσα.

Έστω το πολύ απλό αλφάβητο $\Sigma = \{ a \}$, με ένα μόνο σύμβολο. Σε αυτό είναι δυνατόν να παραστήσουμε αριθμούς, απλά επαναλαμβάνοντας το a τόσες φορές όσες μας λείει ο αριθμός αυτός, λ.χ. $a^{(4)} = aaaa$ για το τέσσερα, $a^{(7)} = aaaaaaa$ για το επτά, κττ. Έστω λοιπόν η γλώσσα $L_{\text{πρώτοι}} = \{ a^{(p)} : p \text{ πρώτος αριθμός} \}$. Είναι η γλώσσα αυτή ομαλή; Με το λήμμα άντλησης είναι εύκολο να αποδείξουμε πως όχι: καθώς υπάρχουν άπειροι πρώτοι αριθμοί ισχύει το λήμμα άντλησης, άρα η $L_{\text{πρώτοι}}$ περιέχει (μεταξύ άλλων) όλες τις λέξεις της μορφής (για κάθε $k \geq 0$, και για κάποιο $v_2 \geq 1$):

$$\underbrace{aa\dots a}_{v_1 \text{ φορές}} \underbrace{(aa\dots aa)^k}_{v_2 \text{ φορές}} \underbrace{a\dots aaa}_{v_3 \text{ φορές}}$$

Ο αριθμός p που παριστάνεται από αυτή την λέξη είναι το πλήθος των a αυτής της λέξης, δηλαδή ο $p = v_1 + k v_2 + v_3$. Δεν μπορεί όμως αυτός ο αριθμός να είναι πρώτος για όλα τα k , διότι για $k = (v_1 + v_3)$ έχουμε ήδη $p = (v_1 + v_3)(v_2 + 1)$, που δεν είναι πρώτος. (Πώς αντιμετωπίζουμε την περίπτωση $(v_1 + v_3) = 1$;) Από το άτοπο που προκύπτει έχουμε ότι η $L_{\text{πρώτοι}}$ είναι μη-ομαλή γλώσσα – τα αυτόματα δηλαδή δεν μπορούν να ανιχνεύσουν την ιδιότητα «πρώτος αριθμός».

Παράδειγμα: τα αυτόματα και ο πολλαπλασιασμός.

Χρησιμοποιώντας το δυαδικό αλφάβητο $\{ 0, 1 \}$ και τα σύμβολα ‘x’, ‘=’, είναι δυνατόν να παραστήσουμε με μια λέξη λ τους δύο παράγοντες ενός γινομένου και το γινόμενο, κατά τον προφανή τρόπο του εξής παραδείγματος, όπου $1011_2 = 11_{10}$, $110_2 = 6_{10}$, $1000010_2 = 66_{10}$, και $1011_2 \times 110_2 = 1000010_2$:

$$\lambda = 1011 \times 110 = 1000010$$

Μπορούμε να ορίσουμε λοιπόν την γλώσσα L_x των «σωστών πολλαπλασιασμών»:

$$L_x = \{ \lambda : \lambda \in \{0, 1, x, =\} \text{ και } \lambda \text{ παριστά (όπως στο παράδειγμα) σωστό πολλαπλασιασμό δυαδικών} \}.$$

Είναι η γλώσσα αυτή ομαλή; Όχι διότι κάποιοι πολλαπλασιασμοί παράγουν ένα φαινόμενο παρόμοιο με εκείνο της μη-ομαλής γλώσσας $\{x^{(v)} \vee x^{(v)} : v \geq 0\}$. Οι πολλαπλασιασμοί αυτοί είναι όσοι έχουν την μορφή $101 \times 1001 = 101101$, $1001 \times 10001 = 10011001$ και γενικά $10^{(k)}1 \times 10^{(k+1)}1 = 10^{(k)}110^{(k)}1$, και οι οποίοι αντιστοιχούν στις λέξεις:

$$\begin{array}{ccccccc} 10 \dots 01 & \times & 10 \dots 01 & = & 10 \dots 0110 \dots 01 \\ \text{--}\kappa\text{--} & & \text{--}\kappa+1\text{--} & & \text{--}\kappa\text{--} & & \text{--}\kappa\text{--} \end{array}$$

Το ότι αυτές οι λέξεις απλώς περιέχονται στη γλώσσα των σωστών πολλαπλασιασμών δεν αρκεί, όπως σε προηγούμενο παράδειγμα, για να διαπιστώσουμε ότι αυτή είναι μη-ομαλή. «Πρέπει» να δείξουμε ότι αυτές μπορούν να αποσπαστούν από την L_x μέσω τομής με μια ομαλή γλώσσα. Και αυτό πράγματι είναι δυνατόν μέσω της γλώσσας,

$$\Lambda = \{ 10 \dots 01 \times 10 \dots 01 = 10 \dots 0110 \dots 01 \}$$

$$\begin{array}{ccccccc} & & \text{--}\kappa\text{--} & & \text{--}\lambda\text{--} & & \text{--}\mu\text{--} & & \text{--}\nu\text{--} \end{array}$$

($\kappa, \lambda, \mu, \nu \geq 0$), η οποία εύκολα διαπιστώνεται ότι είναι ομαλή. Μια λέξη της Λ αναπαριστά όμως έναν σωστό πολλαπλασιασμό εάν και μόνον εάν $|\kappa - \lambda| = 1$ και $\mu = \nu$, όπως προκύπτει ελέγχοντας τις τρεις περιπτώσεις: $|\kappa - \lambda| = 0$ ή $|\kappa - \lambda| = 1$ ή $|\kappa - \lambda| \geq 2$. Αν εξετάσουμε την τομή $\Gamma = L_x \cap \Lambda$, θα έχουμε:

$$\Gamma = \{ \text{οι λέξεις της μορφής } 10^{(\kappa)}1 \times 10^{(\lambda)}1 = 10^{(\mu)}110^{(\mu)}1, \text{ όπου } |\kappa - \lambda| = 1, \mu = \min(\kappa, \lambda) \}$$

Με το λήμμα άντλησης προκύπτει, (κατά εντελώς παρόμοιο τρόπο με τα προηγούμενα παραδείγματα), ότι η γλώσσα Γ είναι μη-ομαλή, και άρα ούτε η γλώσσα L_x των σωστών πολλαπλασιασμών είναι ομαλή. Τα αυτόματα, δηλαδή, δεν μπορούν να κάνουν πολλαπλασιασμό· ούτε καν να αναγνωρίσουν έναν ήδη εκτελεσμένο σωστό πολλαπλασιασμό. (Μήπως γι' αυτό το λόγο οι μικροεπεξεργαστές για να κάνουν πολλαπλασιασμό εκτελούν (μικρο)κώδικα;)

■

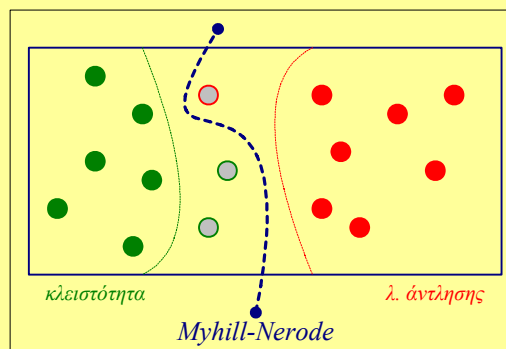
9^ο Πεπερασμένα αυτόματα: ο χαρακτηρισμός των *Myhill-Nerode*.



Ακριβείς χαρακτηρισμοί και η αξία τους.

Δεν είναι εντελώς προφανές γιατί η γλώσσα λ.χ. των πολλαπλασίων του 6 είναι ομαλή γλώσσα, ενώ μια τόσο απλή, φαινομενικά, γλώσσα όσο η $\{a^{(k)}b^{(k)} : k \geq 0\}$ δεν είναι ομαλή: εκ πρώτης (και μόνον) όψεως τα αυτόματα φαίνεται να είναι σε θέση να κάνουν διαίρεση (δια 6, ή δια 3 κοκ), ενώ δεν είναι σε θέση να συγκρίνουν δύο αριθμούς, (εδώ, το πλήθος των a ώστε να αποφασίσουν εάν είναι ίσο με το πλήθος των b ή όχι).

Το επόμενο θεώρημα έρχεται να δώσει την οριστική απάντηση σε τέτοιου είδους απορίες, με τον εξής τρόπο: Ανακαλώντας το σχήμα της προηγούμενης ενότητας, το θεώρημα αυτής της ενότητας θα χαράξει «οριστική» γραμμή στο σύνολο των γλωσσών επί ενός αλφαβήτου Σ , «αριστερά» της οποίας όλες οι γλώσσες θα είναι ομαλές, και «δεξιά» της οποίας όλες θα είναι μη-ομαλές.



Σχήμα: Ένας ακριβής χαρακτηρισμός των ομαλών γλωσσών.

Προφανώς ένας τέτοιος χαρακτηρισμός, με το να είναι απολύτως ακριβής, θα πρέπει να είναι σε θέση να «εξηγεί» το γιατί μια οποιαδήποτε γλώσσα είναι ή δεν είναι ομαλή.

Θεώρημα: ο χαρακτηρισμός *Myhill-Nerode* των ομαλών γλωσσών.

Ισχυρισμός: Έστω μια γλώσσα $L \subseteq \Sigma^*$ και έστω η εξής σχέση \approx_L μεταξύ των λέξεων του Σ^* προσδιορισμένη με βάση την γλώσσα L : $(\lambda \approx_L \lambda') \Leftrightarrow (\forall \tau \in \Sigma^* (\lambda\tau \in L \Leftrightarrow \lambda'\tau \in L))$. Τότε

- Η σχέση \approx_L είναι σχέση ισοδυναμίας επί των λέξεων του Σ^* .
- Η γλώσσα L αναγνωρίζεται από κάποιο αυτόματο εάν και μόνο εάν το πλήθος των κλάσεων ισοδυναμίας της \approx_L είναι πεπερασμένο.

Σχέδιο απόδειξης: Αν η γλώσσα L γίνεται αποδεκτή από ένα αυτόματο, θα γίνεται δεκτή και από ένα αιτιοκρατικό αυτόματο A με καταστάσεις Σ_K . Στα αιτιοκρατικά αυτόματα όμως κάθε λέξη λ οδηγεί το αυτόματο A σε μία ακριβώς συγκεκριμένη την οποία θα συμβολίζουμε με $K_A(\lambda)$. Δεν περιμένουμε φυσικά κάθε λέξη να οδηγεί το αυτόματο σε διαφορετική κατάσταση – εξ άλλου αυτό για γλώσσες με αρκετά πολλές λέξεις ή με άπειρο πλήθος λέξεων αυτό είναι αδύνατον. Με βάση λοιπόν το αυτόματο ορίζεται μία σχέση ως των λέξεων του Σ^* , η \approx_A ως εξής::

$$(\lambda \approx_A \lambda') \Leftrightarrow \text{«οι } \lambda \text{ και } \lambda' \text{ οδηγούν το αυτόματο στην ίδια κατάσταση, ή } K_A(\lambda) = K_A(\lambda')\text{»}$$

Είναι σχεδόν τετριμμένο να ελέγξουμε ότι η σχέση \approx_A είναι σχέσης ισοδυναμίας⁷, όπως επίσης και η σχέση \approx_L . Αυτό που μας ενδιαφέρει εδώ είναι η σχέση των δύο σχέσεων. Εάν $\lambda \approx_A \lambda'$, δηλαδή $K_A(\lambda) = K_A(\lambda')$, τότε οι λέξεις λ και λ' οδηγούν το A σε μία και την αυτή κατάσταση K . Επομένως όπως και εάν επεκτείνουμε αυτές τις λέξεις με ένα τελικό τμήμα $\tau \in \Sigma^*$, η διαδρομή τους εφεξής εντός του A θα είναι η ίδια, (το A είναι αιτιοκρατικό!) και επομένως είτε και οι δύο ($\lambda\tau$ και $\lambda'\tau$) θα καταλήξουν σε αποδεκτική κατάσταση, είτε και οι δύο σε απορριπτική, επομένως είναι ισοδύναμες και κατά την σχέση \approx_L .

$$(\lambda \approx_A \lambda') \Rightarrow (\lambda \approx_L \lambda')$$

Αφού αν δύο λέξεις είναι ισοδύναμες ως προς την σχέση \approx_A θα είναι και ως προς την σχέση \approx_L , οι κλάσεις ισοδυναμίας της 1^{ης} σχέσης περιέχονται σε κλάσεις ισοδυναμίας της 2^{ης}, δηλαδή:

$$\text{πλήθος-κλάσεων}(\approx_A) \geq \text{πλήθος-κλάσεων}(\approx_L)$$

Αλλά το πλήθος των κλάσεων ισοδυναμίας της σχέσης \approx_A είναι, (σχεδόν εξ ορισμού), όσες και οι καταστάσεις του A , δηλαδή: $\text{πλήθος-κλάσεων}(\approx_A) = |\Sigma_K|$. Όλα αυτά μας δίδουν ότι:

$$\text{πλήθος-κλάσεων}(\approx_L) \leq \text{πλήθος-κλάσεων}(\approx_A) = |\Sigma_K| < \infty$$

ότι δηλαδή το πλήθος των κλάσεων ισοδυναμίας της σχέσης \approx_L είναι πεπερασμένο – και έχουμε το 50% του θεωρήματός μας.



Τί πληροφορία «θυμάται» ένα αυτόματο;

Η ιδέα πίσω από όλα αυτά είναι πως οτιδήποτε είναι δυνατόν να «θυμάται» ένα αυτόματο A για κάποιο αρχικό τμήμα α μιας λέξης $\lambda = \alpha\tau$, το οποίο έχει διαβάσει μέχρι τώρα, δεν είναι δυνατόν να «καταχωρίζεται» παρά στην κατάσταση K , στην οποία η λέξη α το έχει οδηγήσει. Εάν λοιπόν η L περιέχει άπειρο πλήθος λέξεων για τις οποίες ανα δύο, λ και λ' , υπάρχει «τέλος» τ , για το οποίο η $\lambda\tau$ γίνεται αποδεκτή, η $\lambda'\tau$ απορρίπτεται, τότε δεν είναι δυνατόν οι λ και λ' να οδηγούν το αυτόματο A στην ίδια κατάσταση, διότι συνεχίζοντας με την τ το αυτόματο μέλλεται να σφάλλει είτε για την $\lambda\tau$ (εάν απορρίψει) είτε για την $\lambda'\tau$ (εάν αποδεχθεί). Μετά είτε την λ είτε την λ' το αυτόματο «θυμάται» πλέον την ίδια μία κατάσταση «πληροφορίας», ενώ επίκεινται δύο διαφορετικές απαντήσεις! Σε μια τέτοια περίπτωση, λοιπόν, το αυτόματο θα πρέπει να διαθέτει άπειρες καταστάσεις, και αυτό είναι αδύνατον.

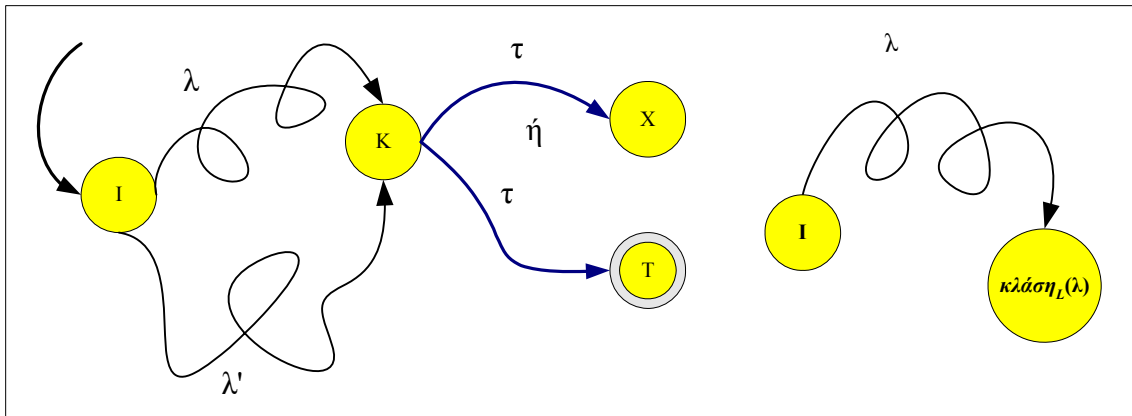
Αντιστρόφως, εάν το πλήθος των κλάσεων ισοδυναμίας της σχέσης \approx_L είναι πεπερασμένο τότε κάθε μία από αυτές μπορεί να αποτελέσει κατάσταση ενός αυτόματου. Μια τέτοια κατάσταση είναι να σαν σημαίνει: «με όποιο τρόπο και αν έφθασες εδώ 'ξέχνα-τον' - η συνέχεια έχει σημασία», διότι ασχέτως του αρχικού τμήματος της λέξης η οποιαδήποτε συνέχεια τ θα καθορίσει το εάν η λέξη θα γίνει αποδεκτή ή όχι – αυτό ακριβώς απαιτεί ο ορισμός της σχέσης \approx_L ! Ορίζουμε, λοιπόν, για την L το εξής αυτόματο A :

Σ	αλφάβητο	τα σύμβολα του αλφαβήτου Σ της γλώσσας L .
Σ_K	καταστάσεις	οι κλάσεις ισοδυναμίας της σχέσης \approx_L .
I	αρχική κατάσταση	η κλάση \emptyset : η κλάση της \approx_L όπου ανήκει η κενή λέξη \emptyset .
T	αποδεκτικές καταστάσεις	οι κλάσεις των λέξεων της L : { κλάση $_L(\lambda)$: για $\lambda \in L$ }.
Δ	οδηγίες	$\Delta(\text{κλάση}_L(\lambda), \sigma) = \text{κλάση}_L(\lambda\sigma)$

Είναι επίσης απλό να δει κάποιος ότι εάν $\lambda \approx_L \lambda'$ τότε για κάθε σύμβολο σ , ισχύει επίσης ότι $\lambda\sigma \approx_L \lambda'\sigma$: διότι μια επέκταση κατά τ των λέξεων $\lambda\sigma$ και $\lambda'\sigma$ δεν είναι παρά μια επέκταση (κατά $\sigma\tau$) των λ και λ' , και τα υπόλοιπα ισχύουν εξ ορισμού. Επομένως από μία κατάσταση κλάση $_L(\lambda)$ μπορούμε – μέσω του

⁷ Έχει δηλαδή τις ιδιότητες: ανακλαστική, συμμετρική και μεταβατική.

συμβόλου σ – να μεταβούμε σε μία και μοναδική επόμενη κατάσταση την κλάση(λς). Η κατάσταση αυτή ουσιαστικά σημαίνει: «έχω διαβάσει μέχρι τώρα λσ – ή κάτι ισοδύναμο». Έχουμε λοιπόν ορίσει ένα, αιτιοκρατικό μάλιστα, αυτόματο A. Γιατί όμως αποδέχεται ακριβώς τη γλώσσα L;



Σχήμα: Κάθε λέξη $\lambda \in \Sigma^*$ οδηγεί το A στην κλάση-κατάσταση $\kappa\lambda\sigma\eta_L(\lambda)$.

Ας προσέξουμε ότι κάθε λέξη $\lambda \in \Sigma^*$ οδηγεί το A στην κατάσταση $\kappa\lambda\sigma\eta_L(\lambda)$ – (πού αλλού!). Αυτό μπορούμε να αποδείξουμε επαγωγικά (επί του μήκους της λ):

- εάν $\lambda = \emptyset$ τότε η λ «οδηγεί» το A στην αρχική κατάσταση που είναι εξ ορισμού η $\kappa\lambda\sigma\eta_L(\emptyset)$.
- εάν $\lambda = \alpha \sigma$, $\alpha \in \Sigma^*$, $\sigma \in \Sigma$, τότε το πρώτο τμήμα α οδηγεί (από επαγωγική υπόθεση) το A στην κλάση(α), και από εκεί μέσω σ το οδηγεί εξ ορισμού στη κατάσταση $\Delta(\kappa\lambda\sigma\eta_L(\alpha), \sigma) = \kappa\lambda\sigma\eta_L(\alpha\sigma)$.

Το αυτόματο A αποδέχεται λοιπόν όλες και μόνον τις λέξεις της L για τους εξής δύο λόγους:

- Έστω ότι η λέξη $\lambda \in L$. Η λ οδηγεί το παραπάνω αυτόματο A στην κατάσταση $\kappa\lambda\sigma\eta_L(\lambda)$ που είναι εξ ορισμού αποδεκτική για το A, επομένως το A αποδέχεται την λ, δηλαδή $\lambda \in L(A)$, και $L \subseteq L(A)$.
- Έστω ότι το αυτόματο A αποδέχεται την λέξη $\lambda \in L(A)$. Αυτό σημαίνει ότι η λ καταλήγει σε κλάση-κατάσταση που είναι αποδεκτική, δηλαδή είναι της μορφής $\kappa\lambda\sigma\eta_L(\lambda')$ για κάποια λέξη $\lambda' \in L$. Αλλά, όπως είδαμε και πριν, μια λέξη ανήκει στην κλάση-κατάσταση στην οποία αυτή οδηγεί το A, δηλαδή $\lambda \in \kappa\lambda\sigma\eta_L(\lambda')$. Δύο λέξεις της ίδια κλάσης είναι όμως ισοδύναμες, δηλαδή είτε και οι δύο ανήκουν, είτε και οι δύο δεν ανήκουν στην L· και εδώ, αφού $\lambda' \in L$, συμπεραίνουμε ότι $\lambda \in L$. Επομένως $L(A) \subseteq L$ και συνολικά $L(A) = L$. ■



Το «μικρότερο» αυτόματο που αποδέχεται μια ομαλή γλώσσα!

Προσέξτε ότι το πλήθος των καταστάσεων του αυτομάτου αυτόματο του θ. Myhill-Nerode ισούται ακριβώς (εκ κατασκευής) με το πλήθος των κλάσεων της σχέσης \approx_L . Έχουμε όμως δείξει ότι για καθε αυτόματο A με Σ_k καταστάσεις που αποδέχεται την γλώσσα L ισχύει ότι:

$$|\approx_L| \leq \Sigma_k$$

Τι «μας λέει» αυτή η σχέση; Ότι οποιοδήποτε αυτόματο αποδέχεται την γλώσσα L έχει πλήθος καταστάσεων μεγαλύτερο ή ίσο από εκείνο που μας παρέχει το θεώρημα Myhill-Nerode! Έχουμε λοιπόν ανακαλύψει την δομή του ελάχιστου αυτομάτου το οποίο αποδέχεται μία δεδομένη ομαλή γλώσσα! Αυτό το θεώρημα απαντά λοιπόν – (στην ειδική βέβαια περιοχή των αυτομάτων) – ένα σχεδόν απρόσιτο πρόβλημα της σχεδίασης αλγορίθμων:

«ποιός είναι ο «μικρότερος» αλγόριθμος-πρόγραμμα-μηχανή (πείτε όπως θέλετε...) που επιλύει ένα δεδομένο πρόβλημα;»

Μια δεύτερη ματιά σε δύο ήδη γνωστά παραδείγματα:

...γιατί η «διαιρετότητα δια $n=3$ » αντιστοιχεί σε ομαλή γλώσσα:

Έστω δύο λέξεις του $\{0, 1\}$ με τα λ και λ' ως «αρχικά τμήματα» και ένα «τελικό τμήμα», τ . Γράφοντας (εδώ) με λ την λέξη και με $[\lambda]$ τον αριθμό που περιγράφει (δυαδικά), έχουμε:

$$[\lambda \tau] = 2^{|\tau|} [\lambda] + [\tau], \quad \text{και} \quad [\lambda' \tau] = 2^{|\tau|} [\lambda'] + [\tau]^8$$

Εάν $[\lambda] = 3q + r$ και $[\lambda'] = 3q' + r'$, τότε:

$$[\lambda \tau] = 2^{|\tau|} [3q + r] + [\tau], \quad \text{και} \quad [\lambda' \tau] = 2^{|\tau|} [3q' + r'] + [\tau]$$

το οποίο μας δίνει:

$$\begin{aligned} [\lambda \tau] \bmod 3 &= (2^{|\tau|} r + [\tau]) \bmod 3 \\ [\lambda' \tau] \bmod 3 &= (2^{|\tau|} r' + [\tau]) \bmod 3 \end{aligned}$$

Εάν λοιπόν οι λ και λ' δίνουν το ίδιο υπόλοιπο $r = r'$, ως προς 3, τότε ανεξαρτήτως ενός κοινού τελικού τμήματος τ οι λέξεις/αριθμοί $\lambda\tau$ και $\lambda'\tau$ θα έχουν επίσης το ίδιο υπόλοιπο, δηλαδή είτε θα διαιρούνται και οι δύο δια 3 είτε δεν θα διαιρείται καμμία από τις δύο δια 3. Εάν δύο λέξεις λοιπόν είναι, ως αριθμοί, ισοϋπόλοιπες ως προς 3 τότε είναι ισοδύναμες ως προς την L_{3^*} . Η L_{3^*} έχει επομένως τόσες κλάσεις ισοδυναμίας όσα είναι τα δυνατά υπόλοιπα ως προς 3, δηλαδή μόνον τρεις (υπόλοιπα 0,1,2), και κατά το θεώρημα Myhill-Nerode, είναι ομαλή γλώσσα! (Η παραπάνω απόδειξη αυτή μάλιστα γενικεύεται για οποιοδήποτε k (τα δυνατά υπόλοιπα είναι 0 1 2... $k-1$) και φυσικά όχι μόνον για το δυαδικό αλφάβητο.)

...και γιατί τελικά η γλώσσα $L_? = \{ \alpha^{(k)} \beta^{(k)} : k \geq 0 \}$ δεν είναι ομαλή γλώσσα:

Προσέξτε ότι εάν $i \neq j$ τότε οι λέξεις $\alpha^{(i)}$ και $\alpha^{(j)}$ δεν είναι ισοδύναμες κατά την σχέση $\approx_{L_?}$, διότι για την τελική συνέχεια $\tau = \beta^{(i)}$ η λέξη $\alpha^{(i)}\tau = \alpha^{(i)}\beta^{(i)}$ ανήκει στην $L_?$, ενώ η λέξη $\alpha^{(j)}\tau = \alpha^{(j)}\beta^{(i)}$ δεν ανήκει, (αφού $i \neq j$). Κάθε λέξη $\alpha^{(i)}$, $i \geq 0$, ανήκει λοιπόν σε μια ξεχωριστή κλάση, επομένως οι κλάσεις ισοδυναμίας της σχέσης $\approx_{L_?}$ είναι απείρου πλήθους, και άρα, τελικά η γλώσσα $L_?$ είναι μια μη-ομαλή γλώσσα. ■

⁸ Το τελικό τμήμα τ «σπρώχνει» το τμήμα λ κατά μήκος(τ) = $|\tau|$ δυαδικά ψηφία προς αριστερά, επομένως πολλαπλασιάζει τον αριθμό $[\lambda]$ επί $2^{|\tau|}$.

10^ο Πεπερασμένα αυτόματα: τα θεωρήματα «διαγνωσιμότητας».



Ποιές πληροφορίες είναι εξαγωγίμες από ένα αυτόματο;

Είδαμε ότι τα πεπερασμένα αυτόματα (και οι ισοδύναμες τεχνικές), μας επιτρέπουν να προσδιορίζουμε μια γλώσσα, δηλαδή ένα σύνολο λέξεων που περιγράφουν «αντικείμενα» με μια κοινή ιδιότητα. Πόσο «διαφανής» είναι ένας τέτοιος ορισμός; Διότι αν κατέχουμε μέσω ενός αυτομάτου A τον πλήρη ορισμό μιας γλώσσας $L(A)$, η αξία αυτού του ορισμού πραγματώνεται όταν μέσω αυτού ερχόμαστε σε θέση να «βλέπουμε» και να διαπιστώνουμε κάποιες ιδιότητες για την γλώσσα αυτή. Υπάρχουν μερικά γενικότερες θεμελιακές ιδιότητες, που θα θέλαμε να είμαστε σε θέση να διαπιστώνουμε για μία L , ή δύο γλώσσες L_1, L_2 , λ.χ.:

- Είναι $L = \emptyset$;
- Είναι $L_1 = L_2$;

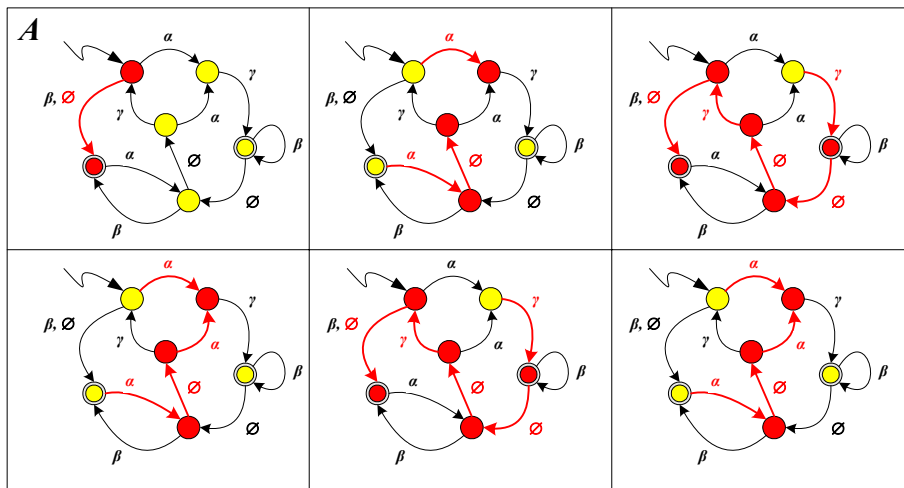
Σε αυτή την σύντομη ενότητα θα δείξουμε ότι ορισμένα θεμελιακά ζητήματα περι γλωσσών, είναι όντως ανιχνεύσιμα εάν αυτές ορίζονται μέσω αυτομάτων.

Θεώρημα: συσχέτιση ομαλών γλωσσών μέσω των αντιστοιχών αυτομάτων.

Ισχυρισμός: Δεδομένων αυτομάτων A, A_1 και A_2 , υπάρχει «αλγοριθμικός» τρόπος διάγνωσης των εξής:

1. εάν μια λέξη $\lambda \in \Sigma^*$, ανήκει στην γλώσσα $L(A)$ ή όχι.
2. εάν το A απορρίπτει πάντοτε, εάν δηλαδή $L(A) = \emptyset$.
3. εάν το A αποδέχεται πάντοτε, εάν δηλαδή $L(A) = \Sigma^*$.
4. εάν το A_1 αποδέχεται ειδικότερη γλώσσα από το A_2 : $L(A_1) \subseteq L(A_2)$.
5. εάν το A_1 αποδέχεται την ακριβώς την ίδια γλώσσα με το A_2 : $L(A_1) = L(A_2)$.

Σχέδιο απόδειξης:



Σχήμα: Η ανίχνευση της αποδοχής μιας λέξης $\lambda = αγαβα$, σε (μη-αιτιοκρατικό) αυτόματο A .

1. Αν το αυτόματο A είναι αιτιοκρατικό τότε όλα είναι εύκολα «εξ ορισμού»: εκτελούμε τις οδηγίες του ισοδύναμου αιτιοκρατικού αυτόματου και αποφασίζουμε με βάση το εάν η τελική του κατάσταση είναι αποδεκτή ή όχι. Αλλά ακόμα και εάν το A είναι μη αιτιοκρατικό, μπορούμε να «εξομοιώσουμε» το αντίστοιχο αιτιοκρατικό χωρίς πρώτα να το κατασκευάσουμε, αρκεί οι μεταβάσεις του να είναι απλές (δηλαδή να είναι κενές (\emptyset) ή να προκαλούνται ένα μόνον σύμβολο:
 - Διατηρούμε ένα σύνολο καταστάσεων $S \subseteq \Sigma^k$, ως την «κατάσταση» στην οποία βρισκόμαστε.
 - Η αρχική «κατάσταση» είναι η αρχική I μαζί με όλες τις καταστάσεις που είναι προσιτές από αυτήν μέσω κενών μεταβάσεων.

- Για κάθε «κατάσταση» S και για ένα-ένα τα σύμβολα $\sigma = \lambda[\kappa]$, $\kappa = 1, \dots, |\lambda|$, της λέξης λ , κατασκευάζουμε το σύνολο των καταστάσεων S' στις οποίες μεταβαίνουμε από οποιαδήποτε κατάσταση του S μέσω του συμβόλου σ . Συμπληρώνουμε την «κατάσταση» S' με όλες τις καταστάσεις του Σ_κ που είναι προσιτές από αυτήν μέσω κενων μεταβάσεων. Το σύνολο S' γίνεται η νέα «κατάσταση» S .
- Ελέγχουμε εάν το σύνολο-«κατάσταση» S περιέχει έστω μία αποδεκτική κατάσταση του T .

Στο προηγούμενο σχήμα εικονίζεται ένα παράδειγμα για την προτεινόμενη διαδικασία. Οι κόμβοι της εκάστοτε «κατάστασης» S σημειώνονται με ερυθρό χρώμα. Οι μεταβάσεις με ερυθρές επιγραφές (είτε κενές, είτε μέσω ενός συμβόλου), υποδεικνύουν τον λόγο και τρόπο σχηματισμού της επόμενης «κατάστασης» S' . Όπως φαίνεται στο σχήμα, το αυτόματο A απορρίπτει την λέξη **αγαβα**: η λέξη αυτή δεν το οδηγεί σε καμία αποδεκτική κατάσταση.

Ακολουθεί ο σχετικός «ψευδοκώδικας».

Αλγόριθμος: «Διάγνωση λέξης μέσω αυτομάτου».
<p>Συνάρτηση Διάγνωση (A: αυτόματο, λ: λέξη): ΝΑΙ/ΟΧΙ</p> <p>$\Sigma(A)$: το αλφάβητο του A $\Sigma_\kappa(A)$: οι καταστάσεις του A $I(A)$: η αρχική κατάσταση του A $T(A)$: οι αποδεκτικές καταστάσεις του A $\Delta(A)$: οι μεταβάσεις του A $S, S' \subseteq \Sigma_\kappa(A)$, $\sigma \in \Sigma(A)$</p> <p>Συνάρτηση Απλές-μεταβάσεις (S, σ, S') { $S' \leftarrow \emptyset$, Για $K \in S$, $(K, \sigma, K') \in \Delta(A)$ { $S' \leftarrow S' + K$ }, $S \leftarrow S'$ }</p> <p>Συνάρτηση Κενές-μεταβάσεις (S') { Επαναλαμβάνουμε $OK \leftarrow \text{ΑΛΗΘΕΣ}$ Για $K \in S'$, Για $(K, \emptyset, K') \in \Delta(A)$ Εάν $K' \notin S'$ τότε { $S' \leftarrow S' + K'$, $OK \leftarrow \PsiΕΥΔΕΣ$ } } έως-ότου OK }</p> <p>{ $S \leftarrow I(A)$ Κενές-Μεταβάσεις(S) Για $\kappa = 1$ έως μήκος(λ) { Απλές-Μεταβάσεις($S, \lambda[\kappa], S'$) Κενές-Μεταβάσεις(S') $S \leftarrow S'$ }</p> <p>Διάγνωση $\leftarrow ((S \cap T(A)) \neq \emptyset)$ }</p>

Ας προσέξουμε σε αυτό το σημείο ότι η κενή γλώσσα δεν περιέχει καμία λέξη – ούτε την κενή λέξη \emptyset (με μηδέν σύμβολα). Αντιθέτως η γλώσσα $\{\emptyset\}$ δεν είναι κενή – έχει μία λέξη (έστω την κενή...).

2. Η διάγνωση του εάν $L(A) = \emptyset$ ή όχι, φαίνεται να αποτελεί δύσκολο πρόβλημα διότι καλούμαστε να διαπιστώσουμε ότι κάθε περίπατος στο αυτόματο A οδηγεί σε απορριπτική κατάσταση, την στιγμή που οι δυνατοί περίπατοι ίσως να έχουν άπειρο πλήθος: οι καταστάσεις είναι μεν πεπερασμένες αλλά εάν υφίστανται «κύκλοι» στο αυτόματο, αυτοί μπορούν να διανυθούν οσοδήποτε φορές. Οι κύκλοι όμως δεν προσφέρουν τίποτε στην προσιτότητα από την αρχική κατάσταση μιας αποδεκτικής κατάστασης, έστω η εξέτασή τους μπορεί να παραληφθεί. Αρκεί δηλαδή να ελέγξουμε εάν υπάρχει έστω μία απλή διαδρομή (= περίπατος χωρίς κύκλους) από την αρχική κατάσταση σε κάποια αποδεκτική κατάσταση του T . Εάν βρούμε έστω μία τότε $L(A) \neq \emptyset$, αλλιώς $L(A) = \emptyset$. Αμέσως παρακάτω δίδεται ο σχετικός «ψευδοκώδικας».

Αλγόριθμος: «Διάγνωση τετριμμένου αυτομάτου», $L(A) = ?? \emptyset$.

Συνάρτηση ΚενήΓλώσσα (A: αυτόματο) : ΝΑΙ/ΟΧΙ

$\Sigma(A)$: το αλφάβητο του A
 $\Sigma_K(A)$: οι καταστάσεις του A
 $I(A)$: η αρχική κατάσταση του A
 $T(A)$: οι αποδεκτικές καταστάσεις του A
 $\Delta(A)$: οι μεταβάσεις του A
 $S, S' \subseteq \Sigma_K(A), \sigma \in \Sigma(A)$

Συνάρτηση **Μεταβάσεις (S)**

{ **Επαναλαμβάνουμε**

OK \leftarrow ΑΛΗΘΕΣ

Για $K \in S$, **Για** $(K, \lambda, K') \in \Delta(A)$ // για οποιαδήποτε $\lambda \in \Sigma(A)^*$

Εάν $K \notin S$ **τότε** { $S \leftarrow S+K$, $OK \leftarrow \PsiΕΥΔΕΣ$ } }

έως-ότου OK

}

{ $S \leftarrow I(A)$

Μεταβάσεις (S)

ΚενήΓλώσσα $\leftarrow ((S \cap T(A)) = \emptyset)$

}

3. Η διάγνωση του εάν $L(A) = \Sigma^*$ ή όχι, δεν αποτελεί πρόβλημα συμμετρικό εκ πρώτης όψεως με το προηγούμενο. Εδώ καλούμαστε να ελέγξουμε εάν όλες οι δυνατές λέξεις του αλφάβητου Σ οδηγούν το αυτόματο A σε αποδεκτική κατάσταση, και όλες οι λέξεις έχουν άπειρο πλήθος χωρίς να φαίνεται ποιές από αυτές θα μπορούσαμε να αποκλείσουμε από την εξέταση... Πλην όμως, η διάγνωση $L(A) = \Sigma^*$ ισοδυναμεί με την διάγνωση $\Sigma^* - L(A) = \emptyset$, και επειδή η $L(A)$ είναι ομαλή, το ίδιο είναι και η συμπληρωματική της, και μάλιστα μπορούμε να κατασκευάσουμε κάποιο αυτόματο A' που αναγνωρίζει την συμπληρωματική γλώσσα: $L(A') = (\Sigma^* - L(A))$, (βλ. σχετική ενότητα περί κλειστότητας). Αρκεί λοιπόν να εξετάσουμε εάν $L(A') = \emptyset$ ή όχι.
4. Από την συνολοθεωρία γνωρίζουμε ότι η σχέση $(X \subseteq Y)$ ισχύει εάν και μόνον εάν $(X - Y) = \emptyset$. Αφού οι γλώσσες $L(A_1)$ και $L(A_2)$ προέρχονται από τα αυτόματα A_1 και A_2 , υπάρχει (και έχουμε δείξει πως κατασκευάζεται) ένα αυτόματο $A^{(-)}$ που αποδέχεται την διαφορά τους: $L(A^{(-)}) = L(A_1) - L(A_2)$. Αρκεί λοιπόν να ελέγξουμε εάν $L(A^{(-)}) = \emptyset$ ή όχι.
5. Ελέγχουμε εάν $L(A_1) \subseteq L(A_2)$ και $L(A_2) \subseteq L(A_1)$.

■

11^ο Πεπερασμένα αυτόματα και «ομαλές εκφράσεις».

Ομαλές εκφράσεις:

Είναι δυνατόν να δώσουμε έναν ακόμα εξυπηρετικό και ενδιαφέροντα τρόπο να παράγουμε γλώσσες – μέσω ενός τύπου εκφράσεων που θα ονομάσουμε «ομαλές». Μέσω αυτών θα λάβουμε τελικά άλλον ένα πολύ ενδιαφέροντα χαρακτηρισμό των ομαλών γλωσσών. Ο ορισμός των ομαλών εκφράσεων είναι επαγωγικός και ακολουθεί:

- Για κάθε λέξη $\lambda \in \Sigma^*$, η $R' = \lambda$ θεωρείται ως ομαλή έκφραση.
- Εάν τα $R, R_k, k = 1, \dots, n$ είναι ομαλές εκφράσεις τότε και οι εξής εκφράσεις θεωρούνται ομαλές:
 - $R' = R_1 R_2 R_n$ («παράθεση»)
 - $R' = \{ R_1, R_2, \dots, R_n \}$ («ένωση»)
 - $R' = R^*$ («αόριστη επανάληψη»)
 - $R' = (R)$ («παρενθέσεις» για την αποφυγή συγχύσεων)

Η ιδέα είναι να κάνουμε κάθε ομαλή έκφραση R χρήσιμη για τον ορισμό μιας γλώσσα, της $L(R)$, η οποία ορίζεται επίσης επαγωγικά (κατ' ανάγκη..., αντίστοιχα με την δομή της έκφρασης R):

Βάση επαγωγής:

Εάν $R' = \lambda$, για $\lambda \in \Sigma^*$ τότε $L(R) = \{ \lambda \}$.

Βήμα επαγωγής:

Εάν $R' = R_1 R_2 \dots R_n$ τότε $L(R') = L(R_1) \parallel L(R_2) \parallel \dots \parallel L(R_n)$

Εάν $R' = \{ R_1, R_2, \dots, R_n \}$ τότε $L(R') = L(R_1) \cup L(R_2) \cup \dots \cup L(R_n)$

Εάν $R' = R^*$ τότε $L(R') = L(R)^*$

Εάν $R' = (R)$ τότε $L(R') = L(R)$

Παραδείγματα: μερικές ομαλές (κανονικές) εκφράσεις.

(1) επί του $\Sigma = \{ \alpha, \beta, \gamma \}$ οι εξής εκφράσεις θεωρούνται ομαλές:

$$\begin{aligned} & \{ \alpha\beta, \beta\alpha\alpha, \beta\gamma\gamma \} \\ & \{ \alpha\beta, \emptyset \} (\{ \alpha\beta, \alpha\beta^*\gamma \})^* \{ \alpha, \gamma \} \\ & (\{ \alpha, \gamma\beta \}^* \parallel \beta)^* \end{aligned}$$

(2) επί του $\Sigma = \{ +, -, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, E, . \}$ οι εξής ομαλές εκφράσεις μας δίνουν την γλώσσα των αριθμών γραμμένων κατά την «επιστημονική» γραφή (λ.χ. +3141592654E-9, 0.789654E38, κττ):

$$\begin{aligned} R_{\langle \text{πρόσημο} \rangle} &= \{ +, -, \emptyset \} \\ R_{\langle \text{ψηφίο} \rangle} &= \{ 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 \} \\ R_{\langle \text{φυσικός} \rangle} &= R_{\langle \text{ψηφίο} \rangle} R_{\langle \text{ψηφίο} \rangle}^* = R_{\langle \text{ψηφίο} \rangle}^+ \\ R_{\langle \text{εκθέτης} \rangle} &= \{ E \} R_{\langle \text{πρόσημο} \rangle} R_{\langle \text{φυσικός} \rangle} \\ R_{\langle \text{αριθμός} \rangle} &= R_{\langle \text{πρόσημο} \rangle} R_{\langle \text{φυσικός} \rangle} \{ . \} R_{\langle \text{φυσικός} \rangle} R_{\langle \text{εκθέτης} \rangle} \end{aligned}$$

Ο χαρακτηρισμός των ομαλών γλωσσών μέσω ομαλών εκφράσεων δίδεται από το εξής θεώρημα:

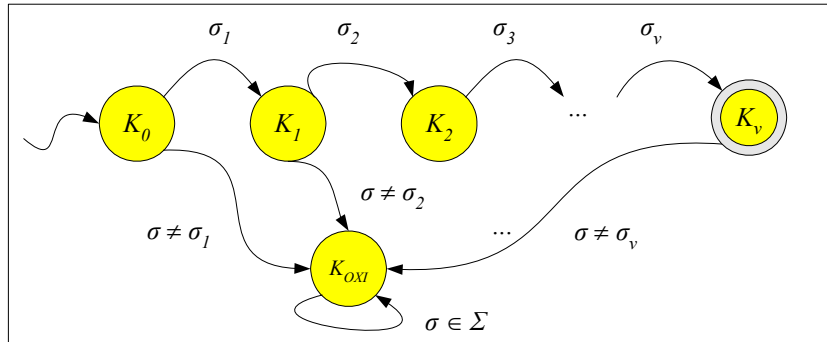
Θεώρημα: ισοδυναμία ομαλών εκφράσεων και πεπερασμένων αυτομάτων

Ισχυρισμός: Για κάθε ομαλή έκφραση R' υπάρχει ένα αυτόματο A το οποίο αναγνωρίζει την ίδια γλώσσα, δηλαδή $L(A) = L(R')$ και αντιστρόφως.

⁹ Η συντομογραφία $R^+ = R R^*$, χρησιμοποιείται αρκετά συχνά, για να δηλώσει την επανάληψη της γλώσσας $L(R)$ μία ή περισσότερες φορές, (και όχι μηδέν ή περισσότερες φορές).

Σχέδιο απόδειξης: Η πρώτη κατεύθυνση μπορεί να αποδειχθεί αρκετά εύκολα, επαγωγικά, με χρήση των θεωρημάτων κλειστότητας.

Βάση επαγωγής: Εάν $R' = \lambda$ τότε το αυτόματο A που αποδέχεται ακριβώς και μόνον την λ κατασκευάζεται κατά άμεσο και προφανή τρόπο:



Σχήμα: Το αυτόματο που αποδέχεται την λέξη $\lambda = \sigma_1 \sigma_2 \dots \sigma_n$.

Βήμα επαγωγής: εξετάζουμε τους 4 τρόπους κατασκευής μιας σύνθετης ομαλής έκφρασης.

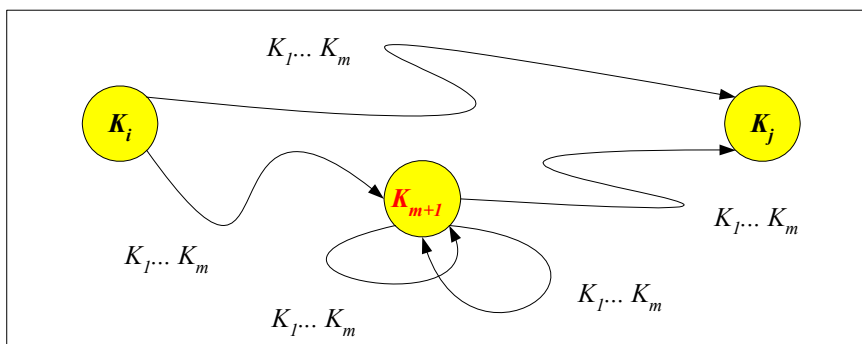
- Εάν $R' = R_1 R_2 R_n$ έχουμε (από επαγωγική υπόθεση) ότι άποια αυτόματα A_1, A_2, \dots, A_n αναγνωρίζουν τις γλώσσες $L(R_1), L(R_2), \dots, L(R_n)$. Το αυτόματο που αναγνωρίζει την $L(R')$ είναι το αυτόματο που αναγνωρίζει την παράθεση των γλωσσών $L(R_k)$, (βλ. θ. κλειστότητας).
- Εάν $R' = \{R_1, R_2, \dots, R_n\}$ έχουμε (επαγωγικά) ότι κάποια αυτόματα A_1, A_2, \dots, A_n αναγνωρίζουν τις $L(R_1), L(R_2), \dots, L(R_n)$. Το αυτόματο που αναγνωρίζει την $L(R')$ είναι το αυτόματο που αναγνωρίζει την ένωση των γλωσσών $L(R_k)$, (βλ. θ. κλειστότητας).
- Εάν $R' = R^*$ έχουμε (επαγωγικά) ότι κάποιο A' αναγνωρίζει την γλώσσα $L(R)$. Το αυτόματο που αναγνωρίζει την γλώσσα $L(R')$ είναι το αυτόματο που αναγνωρίζει την άοριστη επανάληψη $L(R)^*$, (βλ. θ. κλειστότητας).
- Εάν $R' = (R)$ τότε $L(R') = L(R)$, και το αυτόματο που αναγνωρίζει την 2^η αναγνωρίζει και την 1^η.

Για την απόδειξη της αντίστροφης κατεύθυνσης χρειάζεται κάτι πιο δραστικό: πρέπει να δείξουμε ουσιαστικά ότι όλες ακριβώς οι περιπάτοι από την αρχική κατάσταση ενός αυτομάτου A έως κάποια τελική περιγράφονται από κάποια ομαλή έκφραση έκφραση. Προς τούτο θεωρούμε το πώς μπορούμε να ορίσουμε σταδιακά και επαγωγικά το σύνολο όλων αυτών των περιπάτων. Έστω $K_i, i = 1, \dots, n$ οι καταστάσεις του αυτομάτου A . Ορίζουμε τις εξής γλώσσες, για $i, j = 1, \dots, n$:

$$\Lambda_m[K_i, K_j] = \{ \text{όλες οι λέξεις-περίπατοι από την κατάσταση } K_i \text{ έως και την } K_j \text{ που περνούν από κάποιες από τις καταστάσεις } K_1 \text{ έως } K_m, \text{ και μόνον} \}$$

Για $m = 0$ έχουμε μόνον τις κατευθείαν διαδρομές, επομένως για $i, j = 1, \dots, n$, ισχύει το εξής:

$$\Lambda_0[K_i, K_j] = \{ \lambda : \text{για όσες λέξεις } \lambda \text{ υπάρχει κατ' ευθείαν μετάβαση } K_i \xrightarrow{\lambda} K_j \}$$



Σχήμα: Δύο επιλογές: η K_{m+1} κατάσταση χρησιμοποιείται (≥ 1 φορές) – ή όχι.

Για $k = m+1$ θα εκφράσουμε τα $\Lambda_{m+1}[-, -]$ μέσω των $\Lambda_m[K_i, K_j]$ (για $i, j = 1, \dots, n$): ένας περίπατος από την κατάσταση K_i έως και την K_j είτε χρησιμοποιεί την κατάσταση K_{m+1} είτε όχι. Και εάν την χρησιμοποιεί, μία ή περισσότερες φορές, ενδιάμεσα, (ανάμεσα δηλαδή σε δύο διαδοχικές εμφανίσεις της K_{m+1}), θα περνά, φυσικά, μόνον από τις καταστάσεις K_1 έως K_m . Επομένως όλες και μόνον οι λέξεις-περίπατοι από την από K_i στην K_j που περνούν μόνον από (κάποιες από) τις καταστάσεις $1..m+1$ περιγράφονται ως εξής για $i, j = 1, \dots, n$:

$$\Lambda_{m+1}[K_i, K_j] = \{ \Lambda_m[K_i, K_{m+1}] \Lambda_m[K_{m+1}, K_{m+1}]^* \Lambda_m[K_{m+1}, K_j], \Lambda_m[K_i, K_j] \}$$

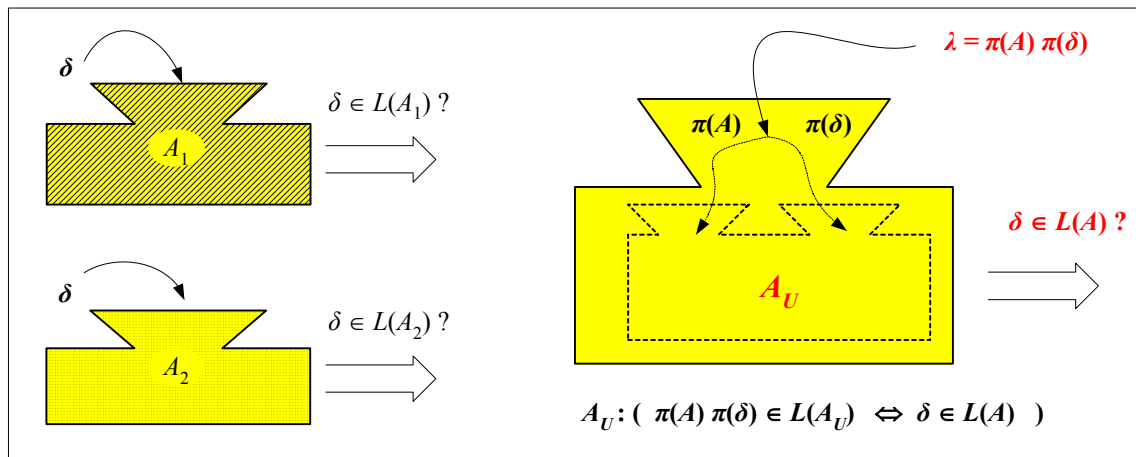
(Ας προσέξουμε στην παραπάνω έκφραση (και) τα τρία στοιχεία των ομαλών εκφράσεων: την παράθεση, την αόριστη επανάληψη, και την ένωση.)

Η γλώσσα του αυτομάτου A προκύπτει από όλους και μόνον τους περιπάτους που αρχίζουν από την αγετηριακή κατάσταση I , καταλήγουν σε κάποια τερματική κατάσταση T_j (για $j = 1, 2, \dots, v, v = |T|$), και ενδιάμεσα επιτρέπεται να περνούν από οποιαδήποτε κατάσταση, δηλαδή από τις $K_i, i = 1, \dots, m$, για $m = n$. Επομένως η έκφραση:

$$R = \{ \Lambda_n[I, T_1], \Lambda_n[I, T_2], \dots, \Lambda_n[I, T_v] \}$$

περιγράφει όλους τους αποδεκτικούς περιπάτους στο A . Παρατηρούμε στα παραπάνω ότι οι εκφράσεις $\Lambda_m[K_i, K_j]$ (για $m = 0$, και για $m = 1, 2, \dots, n$) είναι όλες ομαλές εκφράσεις, επομένως ομαλή είναι και η τελική έκφραση R που περιγράφει την γλώσσα του αυτομάτου A : $L(A) = L(R)$. ■

12^ο Πεπερασμένα αυτόματα: υπάρχει «καθολικό» αυτόματο;



Τα αυτόματα και μια εισαγωγή στον κόσμο του προγραμματισμού.

Είχαμε στις προηγούμενες ενότητες μια εισαγωγή στον κόσμο των αυτομάτων. Αποτελούν υπολογιστικών συσκευών, του γραμματικο-συντακτικού τρόπου, συσκευές που είναι μεν απλές όχι όμως και τετριμμένες.

Κάθε αυτόματο είναι υλοποιήσιμο· ακριβέστερα, για κάθε αυτόματο, η αιτιοκρατική του παραλλαγή (που υπάρχει πάντοτε), είναι φυσικώς υλοποιήσιμη: Για ένα αυτόματο A_1 μπορούμε να φτιάξουμε μια συσκευή που αν δεχθεί μια λέξη δ ως δεδομένο θα μας απαντήσει εάν αυτή ανήκει ή όχι στη γλώσσα $L(A_1)$. Ας προσέξουμε όμως εδώ ότι για ένα άλλο αυτόματο A_2 θα χρειαστεί να κατασκευάσουμε μια άλλη, νέα, συσκευή για την αναγνώριση της αντίστοιχης γλώσσας $L(A_2)$ – (βλ. στο παραπάνω σχήμα τις «μηχανές» A_1 και A_2).

Υπάρχει όμως μια άπειρη ποικιλία αυτομάτων ικανών να αναγνωρίσουν μια αντιστοίχως άπειρη ποικιλία γλωσσών, και επομένως «τεχνολογικά» θα πρέπει να είμαστε σε θέση να κατασκευάζουμε, να διαθέτουμε, και να χρησιμοποιούμε, μια άπειρη ποικιλία από (τέτοιες) υπολογιστικές συσκευές. Εκ πρώτης όψεως αυτό μοιάζει αναπόφευκτο από το γεγονός και μόνον ότι ένα αυτόματο χρησιμοποιεί αλφάβητα (Σ_k και Σ) τα οποία αν και πεπερασμένα, είναι δυνατόν να λάβουν οσοδήποτε μεγάλο μέγεθος.

Αν αυτό ήταν το πρόβλημα, τότε έχουμε μια λύση: αρκεί ένα πεπερασμένο αλφάβητο για να περιγράψουμε όλα τα αυτόματα (όσες καταστάσεις και εάν έχουν), και όλες τις λέξεις επί των οποίων καλούνται να λειτουργήσουν (όσα σύμβολα και αν έχει το αλφάβητό τους).

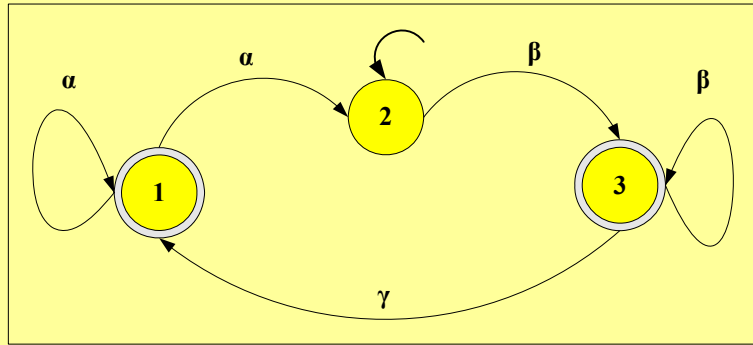
Λ.χ. αρκεί ένα αλφάβητο $\Sigma_U = \{ [, \bullet , | ,] \}$ με 4 σύμβολα για να αναπαραστήσουμε ή περιγράψουμε τα σύμβολα ενός αλφαβήτου $\Sigma = \{ \sigma_1, \sigma_2, \dots, \sigma_n \}$:

- σ_1 περιγράφεται από τη λέξη $\pi(\sigma_1) = [\bullet]$
- σ_2 περιγράφεται από τη λέξη $\pi(\sigma_2) = [\bullet \bullet]$
- ...
- σ_n περιγράφεται από τη λέξη $\pi(\sigma_n) = [\bullet \bullet \bullet \dots \bullet]$ (ν τελείεις)

Εντελώς παρόμοια μπορούμε να περιγράψουμε τις καταστάσεις του αλφάβητου Σ_k , και με λίγο πιο περίτεχνο τρόπο είμαστε σε θέση να περιγράψουμε τις οδηγίες $\Delta(K, \sigma) = K'$, ενός αυτομάτου. Λ.χ. εάν οι καταστάσεις K, K' περιγράφονται από τις λέξεις $\pi(K) = [\bullet \bullet]$, $\pi(K') = [\bullet]$, και το σύμβολο σ από την λέξη $\pi(\sigma) = [\bullet \bullet \bullet]$, τότε η οδηγία $\langle K, \sigma, K' \rangle \in \Delta$, περιγράφεται κατά μονοσήμαντο τρόπο από την λέξη:

$$\pi(\langle K, \sigma, K' \rangle) = [\bullet \bullet | \bullet | \bullet \bullet \bullet]$$

Αν έχουμε στη διάθεσή μας ένα αυτόματο A , με οδηγίες Δ , και παραθέσουμε όλες τις περιγραφές των οδηγιών $\{\pi(i): i \in \Delta\}$, θα λάβουμε μια λέξη $\pi(\Delta) \in \Sigma^*$, που περιγράφει επακριβώς τις οδηγίες του αυτομάτου. Με ένα ακόμα ανάλογο βήμα μπορούμε να περιγράψουμε και τα υπόλοιπα στοιχεία του A : την αρχική κατάσταση I και το σύνολο T των αποδεκτικών καταστάσεων, επιτυγχάνοντας έτσι μια πλήρη και ακριβή περιγραφή του A . Δίνουμε ένα μικρό παράδειγμα: ένα αυτόματο A και την περιγραφή του $\pi(A)$, καθώς και μια λέξη δ προς αποδοχή ή όχι και την περιγραφή της $\pi(\delta)$:



$$\pi(A) = \begin{array}{cccccccc} [\bullet] & [\bullet] & [\bullet] & [\bullet] & [\bullet] & [\bullet] & [\bullet] & [\bullet] \\ \text{---I---} & \text{---T---} & \text{---}\Delta(-,-)\text{---} & \text{---}\Delta(-,-)\text{---} & \text{---}\Delta(-,-)\text{---} & \text{---}\Delta(-,-)\text{---} & \text{---}\Delta(-,-)\text{---} & \text{---}\Delta(-,-)\text{---} \end{array}$$

$$\pi(\delta) = \begin{array}{cccccccc} [\bullet] & [\bullet] & [\bullet] & [\bullet] & [\bullet] & [\bullet] & [\bullet] & [\bullet] \\ \alpha & \alpha & \beta & \alpha & \gamma & \gamma & \alpha & \beta \end{array}$$

Η γλώσσα των περιγραφών όλων των δυνατών αυτομάτων L_A φαίνεται *prima vista* «ακατανόητη», αλλά δεν καθόλου τέτοια: είναι μάλιστα μια ομαλή γλώσσα (!) και μπορούμε να δώσουμε μια ομαλή περιγραφή της. Έχουμε $L_A = L(R_A)$, όπου:

$R_I = [\bullet]^+$	αρχικές καταστάσεις I
$R_T = [\bullet^+ (\bullet^+)^*]$	σύνολα αποδεκτικών καταστάσεων T
$R_\sigma = [\bullet^+]$	σύμβολα Σ ή καταστάσεις Σ_K
$R_\mu = [R_\Sigma R_\Sigma R_\Sigma]$	μεταβάσεις $(-, -, -)$
$R_\Delta = [R_\mu^*]$	οδηγίες Δ
$R_A = [R_I R_T R_\Delta]$	περιγραφή αυτομάτου A

Ακόμα πιο απλούστερη είναι η γλώσσα των περιγραφών L_δ όλων των δυνατών λέξεων: η ομαλή περιγραφή της είναι απλώς η...

$$R_\delta = \{ [], [\bullet^+ (|\bullet^+)^*] \}$$

και $L_\delta = L(R_\delta)$. Προφανώς εξ ίσου ομαλή είναι και η γλώσσα $L_\pi = L(R_A R_\delta)$ που προκύπτει από την παράθεση δύο τέτοιων περιγραφών: την περιγραφή $\pi(A)$ ενός οποιου αυτομάτου A , ακολουθούμενου από την περιγραφή $\pi(\delta)$ μιας οποιας λέξης δ προς αποδοχή ή απόρριψη – (βλ. την «μηχανή» A_π στο αρχικό σχήμα).

Το ερώτημα που έρχεται έχει κεφαλαιώδη σημασία και δείχνει με τον πιο καθαρό τρόπο πόσο κοντά – στα ζητήματα του υπολογισμού – ευρίσκεται η θεωρία με την πράξη. Το ερώτημα είναι: εφόσον αρκεί ένα και μόνον πεπερασμένο αλφάβητο που είναι «καθολικό», ικανό δηλαδή για να περιγράψουμε όλα τα δυνατά αυτόματα και όλες τις δυνατές λέξεις-εισόδους για αυτά, μήπως υπάρχει επίσης ένα «καθολικό» αυτόματο, ικανό να εκτελέσει το έργο οποιουδήποτε άλλου; Υπάρχει, δηλαδή, κάποιο αυτόματο A_π που αποδέχεται την εξής γλώσσα L_π :

$$L_U = L(A_U) = \{ \lambda : \lambda = \pi(A)\pi(\delta) \in L_\pi, \text{ και το αυτόματο } A \text{ αποδέχεται τη } \delta \}$$

Αν υπήρχε ένα τέτοιο «θαυματοργό» αυτόματο τότε το εξής θαυμάσιο θα συνέβαινε: για την υλοποίηση ενός αυτομάτου A δεν θα χρειαζόταν να κατασκευάζουμε (κυριολεκτικά...) κάθε φορά μια νέα συσκευή για την υλοποίηση του εκάστοτε A , αλλά θα αρκούσε να κατασκευάζαμε μόνον το καθολικό αυτόματο A_U , και στη συνέχεια απλώς να «γράφουμε» την περιγραφή $\pi(A)$ του επιθυμητού αυτομάτου A . Έτσι, κάθε φορά που θα θέλαμε να διαπιστώσουμε εάν το A αποδέχεται την λέξη δ ή όχι, θα γράφαμε την περιγραφή της $\pi(\delta)$, θα παραθέταμε τις δύο περιγραφές σε μια λέξη $\lambda = \pi(A)\pi(\delta)$, και θα δίναμε την είσοδο λ στο καθολικό αυτόματο A_U προς εκτέλεση.

Αυτό είπε ο Alan Turing (περί το 1930) ότι ισχύει για τις «μηχανές Turing», διαπιστώνοντας ότι υπάρχει μια καθολική μηχανή T_U , (Turing, όχι αυτόματο), και οδηγώντας στην τεχνολογική, και όχι μόνον, μεταμόρφωση του 20^{ου} αιώνα. Εν τέλει, με την θεωρία του, δημιούργησε τρεις πολύ πρακτικούς «ρόλους»:

- **κατασκευαστές υλισμικού:** φτιάχνουν την καθολική μηχανή T_U .
- **σχεδιαστές λογισμικού/προγραμματιστές:** γράφουν περιγραφές $\pi(T)$ για το T_U .
- **χρήστες:** περιγράφουν τα δεδομένα τους $\pi(\delta)$, και χρησιμοποιούν τα $\pi(T)$ και T_U .

Το εισαγωγικό σχόλιο ήταν λίγο μακρύ – αλλά χωρίς αυτό, το θεώρημα της τρέχουσας ενότητας (μια των 12 γραμμών...) ίσως να φαινόταν ακατανόητο και να περνούσε απαρατήρητο.

Θεώρημα: δεν υπάρχει καθολικό αυτόματο A_U .

Ισχυρισμός: Η γλώσσα $L_U = \{ \lambda : \lambda = \pi(A)\pi(\delta) \in L_\pi, \text{ και το αυτόματο } A \text{ αποδέχεται τη } \delta \}$ δεν είναι ομαλή.

Σχέδιο απόδειξης: Έστω ότι υπήρχε ένα αυτόματο A_U (αιτιοκρατικό, χ.α.γ.), αποδεχόμενο την γλώσσα L_U , και έστω $\nu_U < \infty$ το (πεπερασμένο) πλήθος των καταστάσεών του. Η γλώσσα $\Gamma = \{ \alpha^i : i \leq (\nu_U + 1) \}$ είναι ομαλή και γίνεται αποδεκτή από κάποιο αυτόματο A_Γ , αλλά από το θεώρημα Myhill-Nerode γνωρίζουμε ότι δεν γίνεται αποδεκτή από κανένα αυτόματο με $\leq \nu_U$ καταστάσεις, διότι η γλώσσα Γ έχει $(\nu_U + 1)$ κλάσεις στη σχέση ισοδυναμίας \approx_Γ , (βλ. 9^η ενότητα). Η περιγραφή $\pi(A_\Gamma)$ οδηγεί πάντοτε το A_U σε μία κατάσταση $K_\Gamma = K(\pi(A_\Gamma))$, και από εκεί η υπόλοιπη περιγραφή $\pi(\delta)$ το οδηγεί σε κατάσταση αποδεκτική (αν $\lambda \in \Gamma$) ή απορριπτική, (αν $\lambda \notin \Gamma$). Αν πάρουμε το παραλλαγμένο αυτόματο $A_U[I=K_\Gamma]$ στο οποίο η αρχική κατάσταση είναι η K_Γ , τότε θα έχουμε αυτόματο με ν_U καταστάσεις που αποδέχεται την γλώσσα Γ , ενώ γνωρίζουμε ότι αυτή απαιτεί τουλάχιστον $(\nu_U + 1)$ καταστάσεις – πράγμα άτοπο. ■