ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Γλώσσα και Αναδρομή: Οι αυτοαναφορικοί αλγόριθμοι στην τεχνητή νοημοσύνη

Κούστα Μαρία Νεφέλη

Επιβλέπων Καθηγητής: Αλέξανδρος Αρβανιτάκης

Αθήνα, Ιούνιος 2022

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΕΦΑΡΜΟΣΜΕΝΩΝ ΜΑΘΗΜΑΤΙΚΩΝ ΚΑΙ ΦΥΣΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Κούστα Μαρία Νεφέλη

Γλώσσα και Αναδρομή: Οι αυτοαναφορικοί αλγόριθμοι στην τεχνητή νοημοσύνη

Επιβλέπων:
Αλέξανδρος Αρβανιτάκης, Επίκουρος Καθηγητής, ΣΕΜΦΕ

Τριμελής Επιτροπή

Αλέξανδρος Αρβανιτάκης, Επίκουρος Καθηγητής, ΣΕΜΦΕ
Αντώνιος Χαραλαμπόπουλος, Καθηγητής, ΣΕΜΦΕ
Αριστείδης Παγουρτζής, Καθηγητής, ΗΜΜΥ

Αθήνα, Ιούνιος 2022

Ευχαριστίες

Θα ήθελα να ευχαριστήσω ιδιαίτερα τον επιβλέποντα καθηγητή Αλέξανδρο Αρβανιτάκη για τις πολύωρες συζητήσεις μας και την αμέριστη στήριξη και καθοδήγησή του καθόλη τη μακρά διάρκεια αυτής της διπλωματικής εργασίας. Επίσης, οφείλω ένα μεγάλο ευχαριστώ στους συναδέλφους Κωνσταντίνο Λειβαδά και Παναγιώτη Βασιλειάδη, με τους οποίους πορευτήκαμε σαν ομάδα. Τέλος, ευχαριστώ τους φίλους, τον αδελφό μου και τη μητέρα μου που ήταν πάντοτε εκεί με τον δικό τους μοναδικό τρόπο.

....
Κούστα Μαρία Νεφέλη

Περίληψη

Σκοπός της παρούσας διπλωματικής εργασίας είναι η μελέτη του φαινομένου της αναδρομής στα μαθηματικά και τη γλώσσα και η εξέτασή του εντός του πλαισίου μιας προτεινόμενης θεωρίας μάθησης, βασισμένης στο υπολογιστικό φαινόμενο της αυτοαναφοράς.

Στο κεφάλαιο 1. παραθέτουμε το ιστορικό πλαίσιο, στο οποίο αναπτύχθηκε η Θεωρία Αναδρομής και ορίζουμε την αναδρομή ως μαθηματικό φαινόμενο. Ασχολούμαστε με τις κλάσεις των πρωτογενώς αναδρομικών συναρτήσεων και των ελαχιστικά αναδρομικών μερικών συναρτήσεων.

Στο κεφάλαιο 2. εισάγουμε την έννοια της i-language και αναφερόμαστε στη σημασία της αναδρομής στη γλωσσική ικανότητα.

Στο κεφάλαιο 3. παρουσιάζουμε τους αυτοαναφορικούς αλγορίθμους και σχολιάζουμε την ικανότητά τους να εξελίσσονται με τη βοήθεια μιας μεθόδου εύρεσης μοτίβων, που θα καλούμε διαγωνιοποίηση.

Στο κεφάλαιο 4. αποδεικνύουμε, ότι οι αυτοαναφορικοί αλγόριθμοι είναι κλειστοί ως προς τη σύνθεση και συμπεριλαμβανομένων των αλγορίθμων ελέγχου, που θα παρουσιαστούν στο 3., καθίστανται κλειστοί ως προς την πρωταρχική αναδρομή και την ελαχιστοποίηση.

Στο κεφάλαιο 5. προτείνουμε ένα παράδειγμα, για το πως η γλωσσική κατάκτηση μπορεί να προσομοιωθεί από έναν αυτοαναφορικό αλγόριθμο.

Λέξεις Κλειδιά.
Αναδρομή, Αυτοαναφορικοί Αλγόριθμοι, Γλώσσα, Γλωσσική Κατάκτηση, Καθολική Γραμματική

Abstract

Recursion has been a valuable concept in mathematics for centuries. It reached its peak about 90 years ago when it was involved in the discussion about the foundations of mathematics, eventually leading to the definition of computability. In the meantime, the idea of algorithms was present since antiquity and, during the 1930s, blended with computability, as well.

In the Fifties, the theory of computability influenced a new approach to linguistics, which proposed that research in language should focus on its biological dimension. This view gave rise to a new method known as Generative Grammar, which serves as the research tool to unveil the common basis of all languages, or in technical terms, the Universal Grammar. Nowadays, the prominent belief is that the genetically encoded computational mechanism for the language faculty is based on recursion.

In this thesis, we present a learning theory concerning self-editing algorithms. We prove that the latter are closed under composition, primitive recursion, and the minimization operator thus, they compute exactly the class of computable functions. Therefore, we propose their potential ability to simulate the process of language acquisition.

Keywords. Recursion, Self-editing Algorithms, Language, FLN-FLB, Language Acquisition, Universal Grammar

Περιεχόμενα

Κατάλογος Σχημάτων

Προαπαιτούμενα

Θα αποπειραθούμε μια εισαγωγή στις θεμελιώδεις έννοιες που χρειάζονται για την ομαλή ανάγνωση της εργασίας. Κάποιες από αυτές θα οριστούν διαισθητικά και θα επανέλθουμε σε επόμενα κεφάλαια για περισσότερη αυστηρότητα.

Η ***αναδρομή*** σαν όρος απαντάται με πολλές αφορμές και είναι αντικείμενο μελέτης διαφορετικών επιστημονικών κλάδων. Εντασιακά και χωρίς να συνυπολογίζουμε ειδικούς ορισμούς που μπορεί να λαμβάνει, πρόκειται για μια διαδικασία, η οποία περιγράφεται μέσω του εαυτού της σε κάποιο προηγούμενο στάδιο, με αποτέλεσμα να παράγονται αντικείμενα που επαναλαμβάνονται με αυτο-όμοιο τρόπο. Εμφανίζει, δηλαδή, *αυτοαναφορική συμπεριφορά*. Η αναδρομή αποτελεί βασικό εργαλείο για τα Μαθηματικά και την Επιστήμη των Υπολογιστών, καθώς χρησιμοποιείται σαν μέθοδος ορισμού συναρτήσεων, όπου *η ίδια η συνάρτηση εφαρμόζεται κατά τον ορισμό της*. Παράλληλα, χρησιμοποιείται για τον ορισμό συνόλων και ακολουθιών. Για παράδειγμα, κατά τον ορισμό των φυσικών αριθμών, το 1 (μεταγενέστερα και το 0) ορίζεται ως ο πρώτος φυσικός αριθμός (δηλαδή εκείνος ο οποίος δεν είναι επόμενος κανενός) και κάθε φυσικός έχει επόμενο, ο οποίος είναι επίσης φυσικός. Αξιωσημείωτο και γνώριμο παράδειγμα αναδρομικού ορισμού ακολουθίας είναι η ακολουθία Fibonacci, η οποία μάλιστα κατασκευάστηκε για να μοντελοποιήσει την αύξηση του πληθυσμού των κουνελιών:

$$\begin{aligned} F_0 &= 0, \\ F_1 &= 1, \\ F_{n+1} &= F_n + F_{n-1},\ \forall n \geq 1. \end{aligned}$$

Η ιδέα πίσω από την έννοια του αναδρομικού ορισμού είναι πως κατασκευάζονται μαθηματικά αντικείμενα, δίνοντας τιμή σε κάποια ή κάποιες αρχικές περιπτώσεις (βάση) και στη συνέχεια, ορίζοντας τις επόμενες με τη βοήθεια των προηγούμενων, απλούστερων περιπτώσεων. Για παράδειγμα, ορίζουμε μια συνάρτηση f στο σημείο x χρησιμοποιώντας προηγουμένως ορισμένες τιμές της ($f(y)$ για $y < x$) και επίσης χρησιμοποιώντας "απλούστερες" συναρτήσεις g (ήδη ορισμένες). Στο επόμενο κεφάλαιο, θα μελετήσουμε πως η αναδρομή καθιερώθηκε σαν μέθοδος ορισμού, το θεώρημα που εξασφαλίζει την ύπαρξη και μοναδικότητα των αναδρομικά ορισμένων συναρτήσεων, καθώς και πως προέκυψε ένας ολόκληρος κλάδος στα μαθηματικά να ονομάζεται Θεωρία Αναδρομής.

Η ***μαθηματική επαγωγή*** είναι μια μέθοδος απόδειξης, άρρηκτα συνδεδεμένη με τους φυσικούς αριθμούς, καθώς χρησιμοποιείται για την απόδειξη ιδιοτήτων τους. Κάθε πρόταση P που αποδεικνύεται με μαθηματική επαγωγή εξαρτάται προφανώς από κάποιον φυσικό αριθμό, ο οποίος συμβολίζεται συνήθως με το γράμμα ν και επομένως γράφουμε $P(\nu)$. Η απόδειξη απαρτίζεται από τρια σκέλη:

1. την *βάση της επαγωγής*, όπου αποδεικνύουμε την πρόταση για $\nu = 0$ ή για $\nu = 1$ (αναλόγως την πρόταση),

2. την *επαγωγική υπόθεση*, όπου υποθέτουμε, ότι η πρόταση P ισχύει για κάποιον τυχαίο φυσικό αριθμό κ, δηλαδή υποθέτουμε $P(\kappa)$,

3. και το *επαγωγικό βήμα*, όπου αποδεικνύουμε, ότι αν $P(\kappa)$, τότε και $P(\kappa + 1)$.

Το καίριο σημείο κατανόησης είναι ο τρόπος με τον οποίο μια τέτοια απόδειξη διασφαλίζει, πως η ιδιότητα ισχύει *για όλους τους φυσικούς αριθμούς*. Αυτός ακριβώς ήταν και ο λόγος που οι μαθηματικοί εμπνεύστηκαν τη συγκεκριμένη μέθοδο. Το πρόβλημα που αντιμετώπιζαν ήταν το εξής: έχοντας αποδείξει κάποια ιδιότητα για συγκεκριμένους αριθμούς, με ποιόν τρόπο θα μπορούσε αυτή να αποδειχθεί για το σύνολό τους δεδομένης της απειρίας των περιπτώσεων; Αυτό που χρειάζονταν ήταν ένας ασφαλής τρόπος μετάβασης από μια περίπτωση στην επόμενη. Ακριβώς αυτή η μετάβαση διασφαλίζεται από το επαγωγικό βήμα. Έχοντας αποδείξει αυτό, γνωρίζουμε με σιγουριά, ότι αν η πρόταση ισχύει για κάποιον φυσικό αριθμό, τότε θα ισχύει και για τον επόμενό του. Το κλειδί, τώρα, βρίσκεται στη βάση της επαγωγής, καθώς ήδη έχουμε αποδείξει ότι η P ισχύει για τον πρώτο φυσικό αριθμό (είτε το 0 είτε το 1). Από το επαγωγικό βήμα θα ισχύει και για τον δευτερο και εφόσον ισχύει για τον δεύτερο, πάλι από επαγωγικό βήμα θα ισχύει και για τον τρίτο, κλπ.. Επομένως, η ιδιότητα θα ισχύει για κάθε φυσικό αριθμό.

Το παραπάνω πρόβλημα απασχολούσε τους μαθηματικούς για πολλούς αιώνες. Παράλληλα, πρώιμες μορφές επαγωγής χρονολογούνται ήδη από τον Ευκλείδη, ενώ εντοπίζονται και σε έργα των F. Maurolico (1494-1575) και Pierre de Fermat (1601-1665). Ωστόσο, καμιά από αυτές δεν ήταν αρκετά ικανοποιητική. Η πρώτη σαφής διατύπωση [32] οφείλεται στον Blaise Pascal, ο οποίος το 1654 στο βιβλίο του με θέμα τα αριθμητικά τρίγωνα (τα οποία φέρουν και το όνομά του) όρισε με σαφήνεια και χρησιμοποίησε την -πλέον γνωστή ως- μαθηματική επαγωγή, γράφοντας τα εξής (από αγγλική μετάφραση του D.E. Smith, 1959 [40]):

> Corollary 12: In every arithmetic triangle, if two cells are adjacent in the same row the left is to the right as the number of cells leftwards from the left is to the number of those rightwards from the right, inclusive. Although this proposition has an infinite number of cases, I will give a rather short demonstration, assuming two Lemmas.
> Lemma I: which is self-evident,that this proportion is met with in the second row.
> Lemma II: that if this proportion is found in any row, it will necessarily be found in the following row.
> From which it will be seen that this proportion is necessarily in all the rows: for it is in the second row by the first lemma; hence by the second, it is in the third row, hence in the fourth, and so on to infinity. It is necessary then only to prove the second lemma in this way. If this proportion is met with in any row... I say that the same proportion will be found in the following row... the same may be demonstrated in all the rest, since this proof is based only on the assumption that the proportion occurs in the preceding row, and that each cell is equal to the sum of the two preceding it, which is true in all cases.

Όπως τονίζει ο Paul Ernest στο άρθρο του [32], ο Pascal ορίζει και χρησιμοποιεί την μαθηματική επαγωγή, ακριβώς όπως έχουμε συνηθίσει να την συναντάμε. Φαίνεται να χρησιμοποιεί για λόγους ευκρίνειας δύο λήμματα, τα οποία και αποδεικνύει. Το πρώτο λήμμα αποτελεί τη βάση της επαγωγής, ενώ το δεύτερο το επαγωγικό βήμα και καταλήγει να αποδεικνύει το πόρισμα (Corollary 12) για όλες τις περιπτώσεις.

Έχοντας αποκτήσει μια πρώτη οικειότητα με τις έννοιες *αναδρομή* και *επαγωγή*, θα πάρουμε μια γεύση για το πως αυτές οι δυο συνδέονται οντολογικά:

Ορισμός 0.0.1. *Αναδρομή* ή *ορισμός μέσω επαγωγής* είναι η διαδικασία ορισμού που αντιστοιχεί σε απόδειξη μέσω επαγωγής (Kleene [31]).

Βλέπουμε, δηλαδή, ότι οι δυο έννοιες είναι άρρηκτα συνδεδεμένες, γεγονός που θα σχολιάσουμε εκτενώς στο επόμενο κεφάλαιο.

Μια επίσης καίρια έννοια για την περαιτέρω μελέτη μας είναι η ***υπολογισιμότητα***. Ο όρος αφορά εκείνα τα αντικείμενα, τα οποία μπορούν να καθοριστούν από υπολογισμούς, δηλαδή τι μπορεί να υπολογιστεί και τι όχι. Την έννοια της υπολογισιμότητας συνοδεύει αυτή του αλγορίθμου. Δοσμένο απλοϊκά, πρόκειται για μια συνταγή, ένα σύνολο βημάτων που οδηγεί στη λύση ενός προβλήματος. Αν θέλουμε να δώσουμε έναν πιο τυπικό όρισμό:

Ορισμός 0.0.2. *Αλγόριθμος* καλείται μια μέθοδος ή διαδικασία για την επίλυση κάποιου προβλήματος, η οποία καθορίζεται από ένα πεπερασμένο σύνολο κανόνων ή εντολών.

Παρόλη τη διαισθητική κατανόηση του τι συνιστά ο υπολογισμός, ας δούμε μια πρώτη, επίσης διαισθητική, προσέγγιση της έννοιας, για την οποία στη συνέχεια θα εξηγήσουμε εκτενώς τους τρόπους, με τους οποίους τυποποιήθηκε.

Ορισμός 0.0.3. *Υπολογισμός* ονομάζεται μια διαδικασία, όπου ξεκινώντας από δοσμένα αρχικά αντικείμενα (inputs) και με βάση ένα αυστηρά ορισμένο σύνολο κανόνων (πρόγραμμα, διαδικασία, αλγόριθμος), μετά από διαδοχικά βήματα, μεταβαίνουμε στην τελική κατάσταση με το τελικό αποτέλεσμα (output). [42]

Ορισμός 0.0.4. Κάθε μηχανή (ιδεατή ή πραγματική) ικανή να εκτελεί υπολογισμούς καλείται *υπολογιστικό σύστημα*.

Οι συναρτήσεις που θα μας απασχολήσουν θα είναι κυρίως οι αριθμοθεωρητικές και μάλιστα εκείνες που παρουσιάζουν τέτοια συμπεριφορά, ώστε να μπορούν να υπολογιστούν.

Ορισμός 0.0.5. *Αριθμοθεωρητική* (number-theoretic) ονομάζεται μια κ-μελής συνάρτηση $f : \mathbb{N}^\kappa \to \mathbb{N}$.

Ορισμός 0.0.6. Μια συνάρτηση f θα καλείται *μηχανιστικά υπολογίσιμη* (effectively calculable/effectively computable), αν οι τιμές της μπορούν να υπολογιστούν από μια πεπερασμένη μηχανιστική διαδικασία.

Δηλαδή αν υπάρχει αλγόριθμος που να υπολογίζει την τιμή $f(x)$ χρησιμοποιώντας την τιμή του x.

Αν τώρα έχουμε ένα κατηγόρημα $P(\alpha)$ ή περισσότερων μεταβλητών $P(\alpha_1, \alpha_2, \ldots, \alpha_\nu)$, μπορούμε να κατασκευάσουμε έναν αλγόριθμο για την άπειρη κλάση ερωτημάτων "Είναι το $P(\alpha)$ αληθές;". Για κάθε φυσικό αριθμό α ή για κάθε ν-αδα φυσικών αριθμών $\alpha_1, \alpha_2, \ldots, \alpha_\nu$, παίρνουμε ένα πρόβλημα της κλάσης. Οι πιθανές απαντήσεις τέτοιων ερωτήσεων είναι "ναι" ή "όχι". Έτσι, ο παραπάνω αλγόριθμος θα ονομάζεται *διαδικασία απόφασης* (decision procedure) για το κατηγόρημα P και όταν υπάρχει ένας τέτοιος αλγόριθμος, το κατηγόρημα ονομάζεται *αποκρίσιμο* [1] (decidable).

[1] Κάποιες φορές το συναντάμε και ως *αποφασίσιμο*. Ωστόσο, η μετάφραση *αποκρίσιμο* επικρατεί στη βιβλιογραφία.

Αντίστοιχα, για μια αριθμοθεωρητική συνάρτηση $\varphi(\alpha)$ ή $\varphi(\alpha_1, \alpha_2, \ldots, \alpha_\nu)$, ένας αλγόριθμος για την άπειρη κλάση ερωτημάτων "Ποιά είναι η τιμή της $\varphi(\alpha)$;" ή "Ποιά είναι η τιμή της $\varphi(\alpha_1, \alpha_2, \ldots, \alpha_\nu)$;" ονομάζεται *διαδικασία υπολογισμού* για τη συνάρτηση φ. Αν υπάρχει, η φ καλείται *υπολογίσιμη*.

Εφόσον σε κάθε κατηγόρημα P, μπορούμε να αντιστοιχήσουμε μια συνάρτηση φ, που να παίρνει τις τιμές 0 ή 1 (0 αν το P είναι ψευδές, 1 αν είναι αληθές), οι διαδικασίες απόφασης περιλαμβάνονται στις διαδικασίες υπολογισμού.

1 Θεωρία Αναδρομής

Σε αυτό το κεφάλαιο, αφού κάνουμε μια σύντομη ιστορική αναφορά στη γέννηση του κλάδου της Θεωρίας Αναδρομής, θα μελετήσουμε τους αναδρομικούς ορισμούς και τη σχέση τους με τις επαγωγικές αποδείξεις, καθώς και δυο κλάσεις συναρτήσεων στους φυσικούς αριθμούς, τις πρωτογενώς αναδρομικές και τις ελαχιστικά αναδρομικές.

1.1 Ιστορικά Στοιχεία

Παρακάτω, θα προσπαθήσουμε να παραθέσουμε με χρονολογική σειρά τα γεγονότα που οδήγησαν στη δημιουργία του κλάδου της Θεωρίας Αναδρομής και θα εξηγήσουμε τους λόγους που μετατράπηκε στη θεωρία του υπολογισμού. Ωστόσο, ο αναγνώστης σύντομα θα αντιληφθεί την πυκνότητα των γεγονότων, στα οποία συνέβαλαν εξέχουσες προσωπικότητες της μαθηματικής κοινότητας. Για τον λόγο αυτό, αρκετές φορές δεν θα είμαστε τόσο συνεπείς στη σειρά, αλλά θα δίνουμε βάση στα επεισόδεια και τον τρόπο που αυτά συνδέονται μεταξύ τους.

Πολύ πριν τον 19ο αιώνα, οι μαθηματικοί χρησιμοποιούσαν ευρέως τη μέθοδο του ορισμού συναρτήσεων μέσω επαγωγής. Η απαρχή της Θεωρίας Αναδρομής αποδίδεται από τον Stephen Kleene [31] στον Richard Dedekind, ο οποίος το 1888 εισήγαγε και απέδειξε το θεώρημά του *Satz der Definition durch Induktion* (Θεώρημα του Ορισμού μέσω Επαγωγής) [19], που εξασφαλίζει την ύπαρξη και μοναδικότητα των επαγωγικά ορισμένων συναρτήσεων στους φυσικούς αριθμούς. Ωστόσο, μετέπειτα διατηρήθηκε ο όρος *αναδρομικός ορισμός* (π.χ. [4]) για να μην υπάρξουν παρανοήσεις με την *απόδειξη μέσω ισχυρής επαγωγής*. Πλέον, χρησιμοποιείται κυρίως ο όρος αναδρομικός ορισμός και για τον λόγο αυτό, το θεώρημα ύπαρξης και μοναδικότητας του Dedekind ονομάζεται θεώρημα αναδρομής. Επηρεασμένος από τη δουλειά του Dedekind, ο Guiseppe Peano επιχείρησε να αναπτύξει μια κατάλληλη και ακριβή τυποποίηση με στόχο την φορμαλιστική αναδιατύπωση όλων των μαθηματικών. Τελικά, τα έργα του (1889, 1891) οδήγησαν στην αξιωματικοποίηση των φυσικών αριθμών, με πέντε αξιώματα, γνωστά ως αξιώματα Peano ή Peano - Dedekind. Σε αυτά, συμπεριλαμβανόταν και το αξίωμα της μαθηματικής επαγωγής, το οποίο συνοδευόταν από επαγωγικό ορισμό (που αργότερα ονομάστηκε πρωτογενής αναδρομή από την Rózsa Péter 1934).

Το 1899 ο Hilbert [26] έδωσε μια αξιωματικοποίηση της γεωμετρίας που αντικαθιστούσε τα αξιώματα του Ευκλείδη και έναν χρόνο αργότερα έδειξε πως το ερώτημα της συνέπειας της γεωμετρίας μπορεί να αναχθεί στο ίδιο για το σύστημα των πραγματικών αριθμών και αυτό με τη σειρά του στην αριθμητική (arithmetic), βασίζοντας την απόδειξή του σε αποτελέσματα του Dedekind. Η συνέπεια της αριθμητικής αποτελούσε και το δεύτερο από τα είκοσι τρια προβλήματα, που παρουσίασε ο Hilbert το 1900 στο διεθνές συνέδριο μαθηματικών του Παρισιού.

Ο Hilbert οραματιζόταν τον φορμαλισμό μιας μαθηματικής θεωρίας, έτσι ώστε να αρθεί κάθε αβεβαιότητα σχετικά με το τι αποτελεί αποδεκτή απόδειξη μέσα σε αυτήν. Το ζήτημα της αυστηρής απόδειξης τον απασχολούσε ιδιαίτερα. Μάλιστα, το 2000 βρέθηκε σε κάποιο σημειωματάριό του το 24ο πρόβλημα (σχετικό με την αυστηρότητα των αποδείξεων), το οποίο είχε αρχικά σκοπό να παρουσιάσει με τα υπόλοιπα είκοσι τρια, αλλά για λόγους άγνωστους έμεινε αδημοσίευτο [43]. Στο άρθρο του [25], το 1918, έθεσε το ζήτημα της *υπολογισιμότητας* στη βάση κάθε μαθηματικού προβλήματος, καθώς και το πρόβλημα της *αποφασισιμότητας* ενός μαθηματικού ερωτήματος μέσα από ένα πεπερασμένο πλήθος βημάτων [31]. Τη δεκαετία του 1920 ανέπτυξε τη λογική των κατηγορημάτων, σε μια προσπάθεια να αποσαφηνίσει τους κανόνες συναγωγής, έργο με το οποίο είχαν καταπιαστεί επίσης οι Frege και Russel. Ωστόσο, επηρεασμένος από την ανακάλυψη του παραδόξου στη θεμελίωση του Frege από τον Russel, ο Hilbert φαίνεται να είχε ένα βασικό κίνητρο για να διασφαλίσει, ότι η λογική των κατηγορημάτων δεν θα είχε αντίστοιχη μοίρα [20]. Σε αυτό το πλαίσιο, διατύπωσε το γνωστό *πρόβλημα απόφασης* (*Entscheidungsproblem*), το οποίο αποτελεί πρόβλημα εύρεσης ενός αλγορίθμου που να αποφαίνεται αν μια πρόταση είναι αποδείξιμη ή όχι εντός της λογικής των κατηγορημάτων. Έτσι, έστρεψε την προσοχή στον *υπολογισμό* που θα καταδύκνυει αν μια πρόταση είναι αληθής ή ψευδής και δεδομένου ότι ο αλγόριθμος που θα το αποφάσιζε αυτό μπορεί να δώσει μια μόνο απάντηση, η λογική θα ήταν εκ κατασκευής μη αντιφατική.

Γενικεύοντας, αν εφαρμόσουμε τα παραπάνω σε κάποιο τυπικό σύστημα (όχι απαραίτητα τη λογική των κατηγορημάτων), προκύπτει το πρόβλημα της εύρεσης μιας διαδικασίας απόφασης (Entscheidungsverfahren), όπου για οποιονδήποτε δοσμένο τύπο του συστήματος η αποδειξιμότητά του ή μη σε αυτό το σύστημα μπορεί να *αποφασιστεί σε πεπερασμένα βήματα*. Έτσι, προκύπτει το Entscheidungsproblem, το οποίο αποδίδεται μεν στον Hilbert, αλλά είχε εμφανιστεί προηγουμένως στους Schröder 1895 και Löwenheim 1915 [31]. Αναλυτικότερα, δοσμένης μιας απόδειξης και εφαρμόζοντας τους κανόνες που ορίζουν το τυπικό σύστημα θα πρέπει να είναι δυνατό να ελεγχθεί η ορθότητα της δοθείσας απόδειξης. Αν αγνοήσουμε τους διαφορετικούς τρόπους με τους οποίους μπορεί να γίνει ο φορμαλισμός μιας θεωρίας, ο στόχος της εμβάπτισης μιας μαθηματικής θεωρίας σε ένα τυπικό σύστημα είναι η εύρεση ενός αλγορίθμου για την έννοια της απόδειξης μέσα στη θεωρία. Το πρόβλημα απόφασης (decision problem), λοιπόν, για δοσμένο τυπικό σύστημα μεταφράζεται στο πρόβλημα εύρεσης ενός αλγορίθμου για την έννοια της αποδειξιμότητας (provability) [31]. Έτσι, η προσοχή στρέφεται στις έννοιες του υπολογισμού και των αλγορίθμων.

Οι τελευταίες απασχολούσαν τους μαθηματικούς από την εποχή των Βαβυλωνίων. Ήδη από τον Ευκλείδη παρατηρείται ανάπτυξη θεωρητικών αλγορίθμων (π.χ. η μέθοδος του Ευκλείδη για την εύρεση μέγιστου κοινού διαιρέτη). Το ίδιο το όνομα αλγόριθμος προέρχεται από τον Al-Khowarizmi, Άραβα μαθηματικό του 9ου αιώνα. Αργότερα, οι Pascal, Leibniz, κ.ά. επιδίωξαν την ανάπτυξη υπολογιστικών μηχανών. Ιδιαίτερα επιδραστική υπήρξε η μηχανή του Leibniz (*stepped reckoner*), ικανή να εκτελέσει πολλαπλασιασμό.

Το 1923 ο Thoralf A. Skolem στο άρθρο του *The foundations of elementary arithmetic established by the recursive mode of thought, without the use of apparent variables ranging over infinite domains.* έδωσε για πρώτη φορά μια διαισθητική περιγραφή των πρωτογενώς αναδρομικών συναρτήσεων [2]. Ακολούθησε ο David Hilbert, ο οποίος το 1926 επιδώθηκε στην προσπάθεια να αποδείξει την υπόθεση του σενεχούς του Georg Cantor, χρησιμοποιώντας περισσότερο περίπλοκες αναδρομικές διαδικασίες. Συγκεκριμένα, συμπεριέλαβε ένα παράδειγμα αμοιβαία αναδρομικής συνάρτησης, για την οποία ο Wilhelm Ackermann είχε αποδείξει [2], ότι δεν ανήκει στην κλάση των πρωτογενώς αναδρομικών συναρτήσεων. Παρότι ο Hilbert απέτυχε στο εγχείρημά του, κάποια σημεία της εργασίας του αξιοποιήθηκαν μετέπειτα από τον Kurt Gödel στην απόδειξη της συνέπειας της υπόθεσης του συνεχούς.

Το 1928 ο Hilbert και ο Ackermann στο βιβλίο τους [27] συμπεριλαμβάνουν την διατύπωση του Entscheidungsproblem για την πρωτοβάθμια κατηγορηματική λογική, καθώς και την

[2]Η ονομασία "πρωτογενώς αναδρομικές συναρτήσεις" πρωτοεμφανίστηκε στο άρθρο του Kleene το 1936 αντικαθιστώντας τον όρο "αναδρομικές συναρτήσεις", που χρησιμοποιούσαν οι Gödel (1931, 1934) και Péter (1934, 1935, 1936), ενώ ο όρος "αναδρομή" (αλλά όχι "αναδρομική συνάρτηση") εντοπίζεται για πρώτη φορά στους Skolem (1923) και Hilbert (1926).

ερώτηση αν αυτό το σύστημα είναι πλήρες. Το δεύτερο απαντήθηκε καταφατικά μόλις το 1929 από τον Kurt Gödel, ενώ το πρώτο, όπως θα δούμε παρακάτω, προκάλεσε εντατική έρευνα με αντικείμενο μελέτης την έννοια του υπολογισμού.

Η ανάπτυξη της συνολοθεωρίας του Cantor, καθώς και τα παράδοξα που ανέκυψαν, προκάλεσαν περί το 1930 έντονη συζήτηση πάνω στα θεμέλια των μαθηματικών [31]. Δημιουργήθηκαν, έτσι δυο σχολές: αυτή των κατασκευαστικιστών και αυτή των κλασικών. Οι μεν δε δέχονταν υπαρξιακές αποδείξεις παρά μόνον εάν συνοδεύονταν από κάποια μέθοδο κατασκευής του αντικειμένου, του οποίου το θεώρημα εγγυόταν την ύπαρξη. Οι δε, δέχονταν την ύπαρξη μαθηματικών αντικειμένων αν η απόδειξη ήταν σύμφωνη με τους κλασικούς λογικούς κανόνες [48]. Εκείνη την περίοδο για τους μαθηματικούς δεν ήταν καθόλου προφανές πως μπορεί κανείς να αποδείξει, ότι δεν υπάρχει αλγόριθμος για μια συγκεκριμένη κλάση ερωτημάτων, καθώς για κάτι τέτοιο θα χρειαζόταν ένας χαρακτηρισμός όλων των πιθανών αλγορίθμων σε ένα δοσμένο πεδίο. Το επόμενο βήμα για την θεωρία αναδρομής έγινε ακριβώς εστιάζοντας σε αυτό το κομμάτι, καθώς φαινόταν να είναι το μόνο βήμα για να αντιμετωπίσουν τα πιο κυρίαρχα προβλήματα που αφορούσαν στα θεμέλια των μαθηματικών.

Ταυτόχρονα, η ιδέα της αναδρομής συνδυάστηκε με αυτήν των υπολογίσιμων συναρτήσεων. Καταρχάς ήταν γνωστό, ότι η πρωτογενής και η k-fold αναδρομή ορίζουν αλγοριθμικά τις συναρτήσεις [3]. Ο Gödel στην απόδειξη των θεωρημάτων της μη πληρότητας [23], χρησιμοποίησε πρωτογενώς αναδρομικές συναρτήσεις, αλλά γρήγορα αντιλήφθηκε, ότι δεν εκφράζουν όλες τις μηχανιστικά υπολογίσιμες συναρτήσεις. Έτσι, η ιδέα ήταν να οριστούν όλοι οι πιθανοί αλγόριθμοι στους φυσικούς αριθμούς με τη βοήθεια γενικευμένων αναδρομικών συναρτήσεων. Αυτό τον οδήγησε, μετά από μια πρόταση του Herbrand, το 1934 να δώσει τον πρώτο αυστηρό ορισμό των γενικών αναδρομικών συναρτήσεων στους φυσικούς αριθμούς. Στη συνέχεια της δεκαετίας, ο Kleene απέδειξε τα θεμελιακά αποτελέσματα για τις αναδρομικές συναρτήσεις.

Από την άλλη μεριά, το πρόβλημα απόφασης του Hilbert έμενε ακόμα ανοιχτό. Για να μπορέσουν, λοιπόν, οι μαθηματικοί της εποχής να καταπιαστούν με το πρόβλημα, κατέστησαν αντικείμενο μελέτης τους την έννοια του υπολογισμού, γεγονός που τους οδήγησε σε ευθείς ορισμούς για την έννοια του αλγορίθμου και της υπολογίσιμης συνάρτησης. Σε αυτό το πνεύμα, ο Alonzo Church είχε σαν στόχο να χαρακτηρίσει όλες τις αριθμοθεωρητικές συναρτήσεις για τις οποίες υπάρχουν αλγόριθμοι. Έχοντας αναπτύξει τον λ-λογισμό στις εργασίες του ([15], [16]), ονόμασε τις αριθμοθεωρητικές συναρτήσεις, για τις οποίες υπάρχουν αλγόριθμοι και οι οποίες εκφράζονταν εντός του λ-λογισμού, λ-definable. Παράλληλα, ο Alan M. Turing ανέπτυξε την ομώνυμη μηχανή του, η οποία μοντελοποιεί την ιδέα ενός *φυσικού υπολογιστή* και κατά συνέπεια όλων των πιθανών υπολογισμών, τους οποίους αυτός μπορεί να εκτελέσει [4]. Το ερώτημα, τώρα, ήταν κατά πόσο οι τυποποιήσεις τους αυτές ταυτίζονταν πράγματι με όλες τις υπολογίσιμες συναρτήσεις και άρα δε θα ήταν δυνατό να εμφανιστούν άλλες τυποποιήσεις, όπου θα είναι εφικτή η έκφραση επιπλέον αλγορίθμων. Ο Church σε ένα γράμμα [18] του

[3]Η Rózsa Péter είχε προχωρήσει στην ιεράρχηση των επιπέδων των αναδρομικών συναρτήσεων (primitive recursive, double recursive, triple recursive, κτλ.) σε μια σειρά από άρθρα με αφετηρία το 1934 [34, 36, 33, 37], καθώς και στο βιβλίο της [35].

[4]Η μηχανή Turing είναι ένα μαθηματικό αντικείμενο, ένας αφηρημένος υπολογιστής, ο οποίος συγκροτείται από μια μονάδα ελέγχου, μια ταινία απεριόριστου (και όχι άπειρου) μήκους και μια κεφαλή που μπορεί να κινείται κατά μήκος της ταινίας (ως προς και τις δυο κατευθύνσεις), να διαβάζει και να γράφει σε κάθε κελί. Ο Turing ≪κατασκευάσε≫ τη μηχανή του, για να διερευνήσει τι είναι υπολογίσιμο και τι όχι, στην προσπάθειά του να λύσει το Entscheidungsproblem του Hilbert.

στον Gödel, του εξέφρασε την πεποίθησή του, ότι οι λ-definable συναρτήσεις ταυτίζονται με τις μηχανιστικά υπολογίσιμες, κάτι που βρήκε τον Gödel ιδιαίτερα σκεπτικό. Μάλιστα, ο ίδιος φαινόταν να μην είναι πεπεισμένος ούτε για τη δική του τυποποίηση [5]. Τελικά, ο Church έδειξε ότι οι λ-definable, τις οποίες ονόμασε αναδρομικές (recursive) [6], είναι ισοδύναμες με τις γενικές αναδρομικές συναρτήσεις (Herbrand-Gödel general recursive) και διατύπωσε τη θέση [7] του, ότι όλες οι συναρτήσεις για τις οποίες υπάρχουν αλγόριθμοι είναι λ-definable και ισοδύναμα Herbrand-Gödel general recursive, καταλήγοντας να δίνει αρνητική απάντηση στο πρόβλημα του Hilbert [17]. Παράλληλα, ο Turing διατύπωσε τη θέση του [8], ότι οι μηχανές Turing ταυτίζονται με τις μηχανιστικά υπολογίσιμες συναρτήσεις, καταλήγοντας, ανεξάρτητα από τον Church, επίσης σε αρνητική απάντηση για το Entscheidungsproblem [45]. Επίσης, στην ίδια εργασία συμπεριέλαβε την απόδειξη, ότι η τυποποίησή του ήταν ισοδύναμη με αυτή του Church. Αξίζει να αναφερθεί, ότι ο Gödel αποδέχθηκε τη θέση του Turing από την πρώτη στιγμή με ενθουσιασμό. Στη συνέχεια, ακολούθησαν κι άλλες αξιόλογες (όλες τους ισοδύναμες) τυποποιήσεις από τους Post [38], Markov και Smullyan [41].

Οι θέσεις των Church και Turing συχνά αναφέρονται συγχωνευμένα ως μια, την Church-Turing Thesis, που μεταφραζόμενη ελεύθερα δηλώνει, ότι η έννοια του υπολογισμού, όπως αυτή ορίζεται από τον λ-λογισμό, τις γενικές αναδρομικές συναρτήσεις, τις μηχανές Turing κλπ, είναι η σωστή έννοια του υπολογισμού [20].

Όπως είναι φανερό, από το αρνητικό αποτέλεσμα για το πρόβλημα απόφασης, οι μαθηματικοί είχαν πλέον καταφέρει να δώσουν έναν απόλυτο ορισμό για την έννοια του αλγορίθμου, δηλαδή της έννοιας *υπολογίσιμος*. Όλες οι διαφορετικές τυποποιήσεις που προτάθηκαν, αποτελούσαν στην ουσία διαφορετικές γλώσσες μέσα από τις οποίες μπορούν να εκφραστούν οι αλγόριθμοι. Το γεγονός, λοιπόν, ότι όλες τους εκ των υστέρων αποδείχθηκαν ισοδύναμες, έδειχνε πως η ιδέα του αλγορίθμου ήταν πλέον ανεξάρτητη της περιστασιακής τυποποίησης.

Το 1996, ο Robert Irving Soare πρότεινε να μετονομαστεί η Θεωρία Αναδρομής σε Θεωρία Υπολογισιμότητας. Στήριξε το επιχείρημά του στην εντασιακή διαφορά των εννοιών αναδρομή και υπολογισιμότητα και επισήμανε, ότι θα πρέπει να διαχωρίζουμε τη θέση του Church από αυτή του Turing [42].

1.2 Αναδρομικοί Ορισμοί και Επαγωγικές Αποδείξεις

Θα μας απασχολήσουν οι αναδρομικοί ορισμοί και η σχέση τους με τις επαγωγικες αποδείξεις. Το σύνολο, πάνω στο οποίο θα δουλέψουμε, είναι οι φυσικοί αριθμοί. Η διαίσθησή μας για αυτούς υπαγορεύει πως μπορεί κανείς να φτάσει οποιονδήποτε φυσικό αριθμό, ξεκινώντας από το μηδέν και επαναλαμβάνοντας ατέρμονα την πράξη του επομένου. Αυτή ακριβώς η διαίσθηση εκφράζεται στην *Αρχή Επαγωγής*. Η τελευταία, μαζί με το *Βασικό Λήμμα Αναδρομής*, αποτελούν θεμελιακές για το σύνολο των φυσικών. Πιο συγκεκριμένα, τα δεχόμαστε ως *αξιώματα*, δηλαδή δεν επιδέχονται απόδειξη, εκτός φυσικά αν βρισκόμαστε στο πλαίσιο κάποιας θεωρίας (π.χ. θεωρία συνόλων), όπου κάνουμε χρήση συγκεκριμένων ορισμών για τους αριθμούς [49].

---

[5] Σε ένα γράμμα στον Martin Davis στις 15 Φεβρουαρίου του 1965 εκφράζει τη δυσπιστία του [31]

[6] Φαίνεται να είναι η πρώτη φορά που εμφανίζεται ο όρος "αναδρομικές συναρτήσεις" να ταυτίζεται με τις "γενικές αναδρομικές" [42].

[7] Η ονομασία Church Thesis εμφανίστηκε πρώτη φορά σε άρθρο του Kleene [30] και πλέον έχει καθιερωθεί.

[8] Turing's Thesis [1936, §9]: "every intuitively computable function is computable by a Turing machine".

Αρχή Επαγωγής. Η χαρακτηριστική ιδιότητα του συνόλου των φυσικών αριθμών

$$\mathbb{N} = \{0, 1, 2, ...\}$$

είναι ότι *κάθε σύνολο* A *που περιέχει το 0 και είναι κλειστό για την πράξη του επομένου περιέχει όλους τους αριθμούς. Συμβολικά,*

$$\big[0 \in A \,\&\, (\forall n)\, [n \in A \Rightarrow n + 1 \in A]\,\big] \tag{1}$$

Είναι εύκολο να δειχθεί πως η Αρχή Επαγωγής είναι *ισοδύναμη* με το Βασικό Λήμμα Αναδρομής, δηλαδή αν υποθέσουμε οποιαδήποτε από τις δυο αρχές, συνάγουμε την άλλη.

Βασικό Λήμμα Αναδρομής. Για όλα τα σύνολα X, W και δοσμένες συναρτήσεις $g : X \to W,\ h : W \times \mathbb{N} \times X \to W$, υπάρχει ακριβώς μια συνάρτηση, τέτοια ώστε

$$\begin{aligned} f(0, x) &= g(x), \\ f(n + 1, x) &= h(f(n, x), n, x). \end{aligned} \tag{2}$$

Ειδικότερα, χωρίς την παράμετρο x, για κάθε $w_0 \in W$ και κάθε συνάρτηση $h : W \times \mathbb{N} \to W$, υπάρχει ακριβώς μια συνάρτηση $f : \mathbb{N} \to W$ που ικανοποιεί τις εξισώσεις

$$\begin{aligned} f(0) &= w_0, \\ f(n + 1) &= h(f(n), n). \end{aligned} \tag{3}$$

Ο λόγος, που η παραπάνω αρχή καλείται λήμμα είναι, επειδή συνήθως χρησιμοποιείται η Αρχή Επαγωγής για την απόδειξή της και όχι το αντίστροφο.

Η Αρχή Επαγωγής αποτελεί το εργαλείο για την απόδειξη, ότι όλοι οι φυσικοί αριθμοί έχουν κάποια ιδιότητα $P(\nu)$. Ξεκινάμε δείχνοντας τη βάση $P(0)$ και στη συνέχεια αποδεικνύουμε, ότι αν $P(\kappa)$ για κάποιο $\kappa \in \mathbb{N}$, τότε $P(\kappa + 1)$ και άρα $P(\nu),\ \forall \nu \in \mathbb{N}$. Έχοντας αποδείξει τα παραπάνω, από την Αρχή Επαγωγής έπεται, ότι το σύνολο $A = \{\nu \in \mathbb{N} \mid P(\nu)\}$ περιέχει όλους τους φυσικούς αριθμούς, δηλαδή το ζητούμενο.

Το Βασικό Λήμμα Αναδρομής εγγυάται την ύπαρξη και μοναδικότητα της αναδρομικά ορισμένης συνάρτησης - λύσης του συστήματος εξισώσεων (2). Εν γένει, τα θεωρήματα ύπαρξης και μοναδικότητας αποφέρουν και μια μέθοδο ορισμού του εν λόγω μαθηματικού αντικειμένου. Το Βασικό Λήμμα Αναδρομής αποφέρει ακριβώς τη μέθοδο του ***αναδρομικού ορισμού*** συναρτήσεων.

Οι αναδρομικοί ορισμοί είναι ιδιαίτερα σημαντικοί, καθώς ,όπως τονίζει ο Γ. Μοσχοβάκης [49], έχουν τρεις βασικές ιδιότητες. Καταρχάς, ένα μεγάλο μέρος των συναρτήσεων που συναντάμε στην αριθμητική και την πληροφορική μπορούν να οριστούν αναδρομικά. Παράληλα, φαίνεται να συμπεριφέρονται καλά ως προς την υπολογισιμότητα, καθώς μια συνάρτηση που ορίζεται αναδρομικά από τις *υπολογίσιμες* g και h, θα είναι επίσης *υπολογίσιμη*. Τέλος, όπως έγινε εμφανές και στην απόδειξη του Βασικού Λήμματος Αναδρομής, οι αναδρομικοί ορισμοί οδηγούν με φυσικό τρόπο στις ***επαγωγικές αποδείξεις***. Ας υποθέσουμε πως θέλουμε να αποδείξουμε κάποια ιδιότητα μιας συνάρτησης ορισμένης με αναδρομή. Σίγουρα, δεν μπορούμε να αποδείξουμε την ιδιότητα για κάθε διαφορετική τιμή του $\nu \in \mathbb{N}$ λόγω της απειρίας των περιπτώσεων. Θα ξεκινήσουμε αποδεικνύοντας τη βάση ($\nu = 0$) και στη συνέχεια θα δείξουμε, ότι για κάθε τιμή της ανεξάρτητης μεταβλητής, η αναδρομικά ορισμένη f διατηρεί την ιδιότητα. Επομένως, αν μια συνάρτηση ορίζεται αναδρομικά, αποδεικνύονται ιδιότητές της εύκολα με επαγωγή.

1.3 Πρωτογενώς αναδρομικές συναρτήσεις

Όπως αναφέρθηκε και στην υποενότητα 1.1, οι πρωτογενώς αναδρομικές συναρτήσεις αποτελούν μια κλάση αναδρομικά ορισμένων συναρτήσεων, οι οποίες περί το 1930 είχαν ήδη εξερευνηθεί και ιεραρχηθεί. Παρακάτω, θα μελετήσουμε αυτή την κλάση των συναρτήσεων, ακολουθώντας την πορεία που επιλέγει ο Γ. Μοσχοβάκης στις σημειώσεις του [49]. Ξεκινάμε με έναν βασικό ορισμό:

Ορισμός 1.3.1 (Α. Γραικιώτη)**.** *Πλειομελής συνάρτηση* (function of several variables) στο σύνολο M είναι η τυχαία συνάρτηση $f : M^n \to M$ μιας ή περισσοτέρων μεταβλητών στο M, και η *πλειομέλεια* (arity) της f είναι ο αριθμός n των μεταβλητών της.

Ορισμός 1.3.2. Ένα σύνολο F πλειομελών συναρτήσεων στους φυσικούς είναι *πρωτογενώς κλειστό* αν:

1. Η συνάρτηση $S(x) = x + 1$ του επομένου ανήκει στο F.

2. Για κάθε n και q, η σταθερή συνάρτηση n μεταβλητών

   $$C_q^n(x_1, ..., x_n) = q$$

   ανήκει στο F. Αν $n = 0$, τότε (συμβατικά) $C_q^0 = q$, δηλαδή ταυτίζουμε τη συνάρτηση 0 μεταβλητών με τη (μοναδική) τιμή της.

3. Για κάθε n και i, $1 \leq i \leq n$, η προβολή

   $$P_i^n(x_1, ..., x_n) = x_i$$

   ανήκει στο F. Παρατηρούμε, ότι η P_1^1 είναι η ταυτοτική συνάρτηση στο $\mathbb{N}$, $P_1^1(x) = x$.

4. *Κλειστότητα για σύνθεση.* Αν η m-μελής $g(u_1, ..., u_m)$ και οι m n-μελείς συναρτήσεις

   $$h_1(\vec{x}), ..., h_m(\vec{x})$$

   ανήκουν στο F, με $\vec{x} = (x_1, ..., x_n)$, τότε και η

   $$f(\vec{x}) = g(h_1(\vec{x}), ..., h_m(\vec{x})) \tag{4}$$

   ανήκει στο F.

5. *Κλειστότητα για πρωτογενή αναδρομή.* Αν η n-μελής g και η $(n + 2)$-μελής h ανήκουν στο F, και αν η $(n + 1)$-μελής f ορίζεται από τις εξισώσεις

   $$\begin{aligned} f(0, \vec{x}) &= g(\vec{x}) \\ f(y + 1, \vec{x}) &= h(f(y, \vec{x}), y, \vec{x}), \end{aligned} \tag{5}$$

   τότε και η f ανήκει στο F. Συμβατικά, περιλαμβάνουμε σε αυτό το σχήμα την περίπτωση $n = 0$, οπότε το σχήμα παίρνει τη μορφή

   $$\begin{aligned} f(0) &= q \qquad (= C_q^0) \\ f(y + 1) &= h(f(y), y). \end{aligned}$$

Ορισμός 1.3.3. Η συνάρτηση f θα καλείται *πρωτογενώς αναδρομική* (primitive recursive) αν ανήκει σε κάθε πρωτογενώς κλειστό σύνολο συναρτήσεων, και *πρωτογενώς αναδρομική στο* Ψ, για τυχαίο σύνολο Ψ πλειομελών συναρτήσεων, αν ανήκει σε κάθε πρωτογενώς κλειστό σύνολο που περιέχει το Ψ. Συμβολικά,

$$\mathcal{R}_p = \{f \mid \text{η f πρωτογενώς αναδρομική}\}$$
$$\mathcal{R}_p(\Psi) = \{f \mid \text{η f πρωτογενώς αναδρομική στο } \Psi\}$$

Προκύπτει φυσικά η παρακάτω πρόταση:

Πρόταση 1.3.1. *Το σύνολο $\mathcal{R}_p(\Psi)$ των πρωτογενώς αναδρομικών συναρτήσεων στο Ψ περιέχει το Ψ και είναι πρωτογενώς κλειστό.*

Η έννοια της χαρακτηριστικής συνάρτησης αποδεικνύεται ιδιαίτερα σημαντική, καθώς μπορούμε να αποφανθούμε για μια σχέση $P(\vec{x})$ αν είναι πρωτογενώς αναδρομική, εξετάζοντας την χαρακτηριστική της συνάρτηση. Αντιστοίχως, για ένα σύνολο A. Αυτό σημαίνει, ότι πέρα από συναρτήσεις, θα χαρακτηρίσουμε ως πρωτογενώς αναδρομικά σύνολα και σχέσεις, με τον εξής τρόπο:

Ορισμός 1.3.4. Η χαρακτηριστική συνάρτηση μιας σχέσης $P(\vec{x})$ είναι η

$$\chi_P(\vec{x}) = \begin{cases} 1 & \text{, αν } P(\vec{x}) \\ 0 & \text{, αλλιώς} \end{cases} \tag{6}$$

και η σχέση $P(\vec{x})$ είναι πρωτογενώς αναδρομική αν η $\chi_P(\vec{x})$ είναι πρωτογενώς αναδρομική.

Με αντίστοιχο τρόπο, μπορούμε να ορίσουμε τη χαρακτηριστική συνάρτηση ενός συνόλου $A \subseteq \mathbb{N}$:

$$\chi_A(x) = \begin{cases} 1 & \text{, αν } x \in A \\ 0 & \text{, αλλιώς} \end{cases} \tag{7}$$

και το A είναι πρωτογενώς αναδρομικό αν η χ_A είναι πρωτογενώς αναδρομική.

Οι περισσότερες συναρτήσεις που συναντάμε τόσο στην καθημερινή μαθηματική ενασχόληση, όσο και στην Πληροφορική ανήκουν στην κλάση των πρωτογενώς αναδρομικών συναρτήσεων. Για παράδειγμα, η πρόσθεση ή η συνάρτηση μέγιστου κοινού διαιρέτη είναι πρωτογενώς αναδρομικές.

1.4 Ελαχιστικά αναδρομικές μερικές συναρτήσεις

Διαισθητικά, από τον τρόπο που ορίσαμε τις πρωτογενώς αναδρομικές συναρτήσεις, είναι φανερό πως είναι όλες τους υπολογίσιμες. Ωστόσο, εύκολα μπορούμε να δείξουμε, ότι δεν περιγράφουν όλες τις υπολογίσιμες συναρτήσεις:

> Έστω μια μηχανιστική απαρίθμηση (δηλαδή έχουμε έναν αλγόριθμο που με είσοδο (m, n) υπολογίζει την $f_m(n)$, βασιζόμενοι στο Θεώρημα Κανονικής Μορφής και Απαρίθμησης) όλων των συναρτήσεων στο $\mathcal{R}_p$ ως εξής:
>
> $$\varphi_0, \varphi_1, \varphi_2, \dots$$

Ορίζουμε, τώρα, μια συνάρτηση $f : \mathbb{N} \to \mathbb{N}$, τέτοια ώστε:

$$f(n) = \varphi_n(n) + 1.$$

Η f είναι προφανώς υπολογίσιμη. Έστω, τώρα, ότι η f είναι πρωτογενώς αναδρομική, δηλαδή εμφανίζεται στην παραπάνω αρίθμηση και έστω y ο δείκτης της στην αρίθμηση. Έχουμε:

$$\varphi_y(y) = f(y) = \varphi_y(y) + 1, \text{ άτοπο.}$$

Επομένως, η κλάση των πρωτογενώς αναδρομικών συναρτήσεων δεν περιλαμβάνει όλες τις υπολογίσιμες συναρτήσεις.

Σκοπός είναι να διευρύνουμε τον ορισμό μας σε μια κλάση συναρτήσεων που θα αποφεύγει το παραπάνω άτοπο. Επιλέγουμε, λοιπόν, να ορίσουμε *μερικές* συναρτήσεις, δηλαδή συναρτήσεις, οι οποίες δεν ορίζονται απαραίτητα για όλους τους φυσικούς αριθμούς. Κάτι τέτοιο ταιριάζει διαισθητικά και με την έννοια του υπολογισμού: ένα πρόγραμμα δεν είναι απαραίτητο, ότι τερματίζει για όλες τις εισόδους. Η μετάβαση από τις πρωτογενώς αναδρομικές συναρτήσεις (ολικές) συντελείται επιτρέποντας την απεριόριστη (ή μη φραγμένη) ελαχιστοποίηση.

Χρησιμοποιούμε τους παρακάτω συμβολισμούς:

$f(x)\uparrow$: η f αποκλίνει (diverges) για το όρισμα x ή το πρόγραμμα της f δε σταματά για είσοδο x.
$f(x)\downarrow$: η f συγκλίνει (converges) για το όρισμα x ή το πρόγραμμα της f σταματά για είσοδο x.

Ορισμός 1.4.1. Μερική συνάρτηση (partial function) από το πεδίο εισόδου A στο πεδίο τιμών ή εξόδου B, συμβολικά

$$f : A \rightharpoonup B,$$

είναι η τυχαία συνάρτηση

$$f : A_0 \to B\ , \ \ (A_0 = Domain(f) \subset A)$$

όπου το A_0 καλείται το πεδίο σύγκλισης (domain of convergence) της f.

Ωστόσο, με σκοπό να δοθεί έμφαση στο γεγονός, ότι η μερική συνάρτηση f ενδέχεται να μην ορίζεται για όλα τα στοιχεία ενός πεδίου εισόδου A, ορίζουμε ισοδύναμα:

$$f : A \to B \cup \{\perp\}$$

όπου το $\perp$ έχει επιλεχθεί να συμβολίζει ένα αντικείμενο που δεν ανήκει στο B. Έτσι, τα x για τα οποία $f(x) = \perp$ δεν ανήκουν στο $\mathrm{Domain}(f)$. Βέβαια, εδώ θα πρέπει να αναφερθεί, ότι υπολογιστικά δεν μπορούμε να γνωρίζουμε αν η f θα μας δώσει αυτό το αντικείμενο ή όχι. Δηλαδή, θεωρούμε, ότι σαν απάντηση θα μας δοθεί κάτι, αλλά στην πραγματικότητα ο υπολογιστής δεν απαντάει με αυτόν τον τρόπο. Επομένως, μπορούμε να χρησιμοποιήσουμε τους παραπάνω συμβολισμούς με τις εξής ισοδυναμίες:

$$f(x)\uparrow \iff x \notin \mathrm{Domain}(f) \iff f(x) = \perp$$
$$f(x)\downarrow \iff x \in \mathrm{Domain}(f) \iff f(x) \in B.$$

Φυσικά, κάθε συνάρτηση $f : A \to B$ (ολική) είναι μερική συνάρτηση με Domain(f)=A.

Σχολιάσαμε στην αρχή της υποενότητας, ότι η μετάβαση από ολικές σε μερικές συναρτήσεις συντελείται επιτρέποντας τη μη φραγμένη ελαχιστοποίηση. Η ίδια προϋποθέτει έναν τελεστή, ο οποίος στην πραγματικότητα επιδίδεται σε ≪βλακώδη≫ αναζήτηση (dumb search) [49]. Εκτενέστερα, ο τελεστής ≪δοκιμάζει≫ όλες τις περιπτώσεις ξεκινώντας από την πρώτη μέχρι να βρει εκείνη που ικανοποιεί τη συνθήκη. Παρόλο, που σαν διαδικασία φαντάζει χρονοβόρα, αποτελεί ένα πολύ χρήσιμο εργαλείο για την κατασκευή αλγορίθμων στους φυσικούς αριθμούς. Παραθέτουμε, εδώ, τον αυστηρό ορισμό της ελαχιστοποίησης μιας μερικής συνάρτησης.

Ορισμός 1.4.2. Η ελαχιστοποίηση της μερικής συνάρτησης $g(i, \vec{x})$ είναι η μερική συνάρτηση

$$\begin{aligned} f(y, \vec{x}) &= (\mu i \geq y)[g(i, \vec{x}) = 0] \\ &= \text{ ο ελάχιστος } w \geq y, \text{ τέτοιος ώστε} \\ &\quad g(w, \vec{x}) = 0 \;\;\&\;\; (\forall i)[y \leq i < w \implies g(i, \vec{x}) > 0]. \end{aligned} \tag{8}$$

Επομένως, θα θέλαμε να ονομάσουμε μια συνάρτηση ελαχιστικά αναδρομική, όταν:

Ορισμός 1.4.3. Η τυχαία μερική συνάρτηση $f : \mathbb{N}^n \to \mathbb{N}$ είναι ελαχιστικά αναδρομική (μ-αναδρομική, μ-recursive) στο σύνολο των μερικών συναρτήσεων Ψ, αν η f ανήκει σε κάθε σύνολο μερικών συναρτήσεων που περιέχει το Ψ και είναι πρωτογενώς κλειστό και κλειστό για ελαχιστοποίηση. Συμβολικά,

$$\mathcal{R}_\mu(\Psi) = \{f \mid \text{η f είναι μ-αναδρομική στο } \Psi\}$$
$$\mathcal{R}_\mu = \mathcal{R}_\mu(\varnothing)$$

Εντελώς αντίστοιχα με τις πρωτογενώς αναδρομικές συναρτήσεις, η τυχαία σχέση $R(\vec{x})$ είναι ελαχιστικά αναδρομική, αν η χαρακτηριστική της συνάρτηση είναι ελαχιστικά αναδρομική. Από τον ορισμό του $\mathcal{R}_\mu(\Psi)$ προκύπτει εύκολα η συμπερίληψη:

$$\mathcal{R}_p(\Psi) \subseteq \mathcal{R}_\mu(\Psi).$$

Η αντίστροφη αποδεικνύεται πως δεν ισχύει.

2 Αναδρομή και γλωσσική ικανότητα

Σε αυτό το κεφάλαιο, θα μας απασχολήσει η σχέση της *αναδρομής* με τη *γλωσσική ικανότητα*, όπως έχει προταθεί από τους Hauser et al. [24]. Για να μπορέσουμε, όμως, να μπούμε σε λεπτομέρειες, θα θέλαμε να παρουσιάσουμε το γλωσσολογικό πλαίσιο, στο οποίο θα κινηθεί η μελέτη μας, εξηγώντας τις βασικές έννοιες.

Η Θεωρητική Γλωσσολογία σαν επιστήμη μελετά τη γλώσσα, αναλύοντας τα διάφορα επίπεδά της. Περιλαμβάνει τη συστηματική (φωνολογία, σύνταξη, κ.ά.), την επικοινωνιακή και την βιολογική διάσταση της γλώσσας. Παρότι η γλώσσα είναι το βασικό αντικείμενο μελέτης της γλωσσολογίας, ο αυστηρός ορισμός της έννοιας δεν είναι δυνατός, αν θέλουμε να συνυπολογίσουμε όλες τις εκφάνσεις της, καθώς απαιτεί τη συμμετοχή πολλαπλών επιστημονικών τομέων. Έτσι, κατά το πέρασμα των χρόνων εμφανίστηκαν διάφορες προτάσεις για τον ορισμό της έννοιας. Όλες τους, ωστόσο, αποτύγχαναν να συμπεριλάβουν όλη την έκτασή της.

Η Γλωσσολογία πριν τον 20ο αιώνα είχε κυρίως ιστορικοσυγκριτική φύση και εμπλεκόταν με άλλες επιστήμες, όπως η φιλοσοφία και η φιλολογία, χωρίς σαφή όρια. Το πρώτο μισό του 20ου αιώνα ο Ferdinand de Saussure εισήγαγε την έννοια του δομισμού, θεμελιώνοντας τις αρχές της σύγχρονης γλωσσολογίας.[9] Λίγο μετά το 1950, ο Noam Chomsky πρότεινε μια νέα θεώρηση, πολύ διαφορετική από την καθιερωμένη ως τότε προσέγγιση, η οποία έγινε γνωστή ως *γενετική γλωσσολογία*. Σύμφωνα με τη θεώρηση αυτή, *είναι η βιολογική έκφανση της γλώσσας που θα πρέπει να απασχολεί την επιστήμη της γλωσσολογίας*. Οι λόγοι που τον οδήγησαν σε αυτήν την πρόταση ήταν αποτέλεσμα της συζήτησης γύρω από τη γλωσσική κατάκτηση. Το πρόβλημα ήταν η ικανοποιητική εξήγηση του πως ένα παιδί καταφέρνει να κατακτήσει τη μητρική του γλώσσα, δεδομένου ότι δεν την διδάσκεται με συστηματικό τρόπο. Οι παρατηρήσεις πάνω στον τρόπο με τον οποίο ένα παιδί κατακτά τη μητρική του γλώσσα οδήγησαν στο *επιχείρημα της ανεπάρκειας του ερεθίσματος*, το οποίο είναι πολυδιάστατο:

- Η γνώση της μητρικής είναι αδίδακτη, ασυνείδητη και κατακτάται με αποσπασματικά γλωσσικά ερεθίσματα.

- Τα παιδιά δεν επαναλαμβάνουν μόνο τυποποιημένες εκφράσεις, αλλά παράγουν λόγο δημιουργικά.

- Αποκτούν μια αντίληψη της εξάρτησης των κανόνων από τις δομές.

- Καταφέρνουν εμπειρικά να συγκροτήσουν το σύστημα της μητρικής τους γλώσσας και να αποκτήσουν διαισθήσεις για τη γλώσσα τους (γλωσσικές διαισθήσεις).

Τα παραπάνω συμπεράσματα οδήγησαν τον Chomsky να διατυπώσει τη θέση, ότι ο άνθρωπος γεννιέται με ένα σύστημα γλωσσικής γνώσης. Υποθέτει, δηλαδή, την ύπαρξη ενός έμφυτου μηχανισμού (μηχανισμός γλωσσικής κατάκτησης), που βασίζεται στην Καθολική Γραμματική, ένα σύνολο κανόνων που συμβάλει στη γλωσσική κατάκτηση, με τρόπο που θα μελετήσουμε στη συνέχεια. Επομένως, η ικανότητα του ανθρώπου για την κατάκτηση της γλώσσας είναι γενετικά προκαθορισμένη και στόχος της γλωσσολογίας είναι "η κατανόηση του τρόπου λειτουργίας του

[9] Στην πραγματικότητα, το βιβλίο του Saussure [39] εκδόθηκε με πρωτοβουλία δύο φοιτητών του, το 1916, τρια χρόνια μετά τον θάνατό του και βασιζόταν στις σημειώσεις των διαλέξεων του μαθήματος γενικής γλωσσολογίας που ο Saussure είχε αναλάβει από το 1907 έως και το 1911.

εγκεφάλου, όσον αφορά την ικανότητά του να αντιλαμβάνεται και να παράγει λόγο" (Σαβίνα Ιατρίδου, 2002).

Παράλληλα, προτάθηκε, ότι ο γλωσσολόγος θα πρέπει να εστιάσει στην ανάπτυξη γραμματικών που αποτελούνται από πεπερασμένους κανόνες, οι οποίοι εφαρμοζόμενοι επαναλαμβανόμενα θα μπορούν να αποδόσουν την απειρία των προτάσεων μιας γλώσσας. Επομένως, στο πλαίσιο της γενετικής γλωσσολογίας, η γραμματική θα πρέπει να νοείται ως ένας φορμαλιστικός μηχανισμός, ικανός να *γεννά* την γλώσσα. Η προσέγγιση αυτή έγινε γνωστή ως *γενετική γραμματική* και στην πραγματικότητα άλλαξε την αντίληψη των ερευνητών για το αντικείμενο μελέτης τους: σε πλήρη αναλογία με τα προηγούμενα η γραμματική θα πρέπει να αποτελεί τη θεωρία για τη γλώσσα ως εσωτερικό μηχανισμό.

Ο Chomsky αντιλήφθηκε, ότι ο ακριβής ορισμός των εννοιών αποτελεί τον μοναδικό τρόπο να αναπτυχθεί μια επιστήμη, καθώς έτσι καθορίζονται τα ερωτήματα που κανείς μπορεί να θέσει, αλλά και οι απαντήσεις που πρόκειται να λάβει. Αναγνωρίζοντας, λοιπόν, τον χαοτικό χαρακτήρα της έννοιας της γλώσσας, προσπάθησε να τυποποιήσει το αντικείμενο μελέτης της γλωσσολογίας, απαγκιστρώνοντάς το από εξωγλωσσικούς παράγοντες. Στο βιβλίο του [6], έδωσε την εξής περιγραφή (ελληνική μετάφραση από σημειώσεις του μαθήματος "Εισαγωγή στη γλωσσολογία" του Γ. Μικρού, Φιλοσοφική Σχολή, ΕΚΠΑ):

> Η γλωσσική θεωρία ασχολείται κυρίως με έναν ιδανικό ομιλητή-ακροατή σε μια ολότελα ομοιογενή γλωσσική κοινότητα, που γνωρίζει τη γλώσσα της τέλεια και είναι ανεπηρέαστος από τέτοιες γραμματικά μη σχετικές συνθήκες όπως περιορισμοί μνήμης, δυσκολίες συγκέντρωσης, αλλαγές προσοχής και ενδιαφέροντος και λάθη (τυχαία ή χαρακτηριστικά) όταν εφαρμόζει τη γνώση της γλώσσας του στην πραγματική επιτέλεση (actual performance).

Επίσης, περνά σε έναν περαιτέρω διαχωρισμό μεταξύ *γλωσσικής ικανότητας* (linguistic competence) και *γλωσσικής επιτέλεσης* (linguistic performance). Ο διαχωρισμός αυτός βασίζεται στην ιδέα, ότι η *γνώση* της γλώσσας δεν ταυτίζεται με τη *χρήση* της. Με τον όρο *γλωσσική ικανότητα*, αναφέρεται στη γνώση, που οι ομιλητές/ακροατές έχουν για τη γλώσσα τους, στο *εσωτερικευμένο σύστημα* των κανόνων που έχουν αποκτήσει *ασυνείδητα*, που τους επιτρέπει να παράγουν και να κατανοούν άπειρο αριθμό σωστών προτάσεων και ταυτόχρονα να αντιλαμβάνονται τη γραμματική και συντακτική τους ορθότητα [51]. Αντίθετα, ως *γλωσσική επιτέλεση* ορίζει την "πραγματική χρήση της γλώσσας σε συγκεκριμένες καταστάσεις" [6], η οποία δεν αντανακλά απαραίτητα τη γνώση του ομιλητή. Πώς, όμως, αυτή η τυποποίηση του αντικειμένου μελέτης της γλωσσολογίας υποδεικνύει την απάντηση στο ερώτημα "Τι είναι γλώσσα;";

2.1 Η έννοια της I-language

Εφόσον εντοπίσαμε την έκφανση της γλώσσας, στην οποία η επιστήμη της γλωσσολογίας θα πρέπει να εστιάσει, χρειάζεται να δώσουμε έναν τυπικό ορισμό για τη *γλώσσα*, εντός του πλαισίου της παραπάνω θεώρησης.

Όπως ήδη έχει τονισθεί, ο τρόπος με τον οποίο ο όρος «γλώσσα» χρησιμοποιείται και εντάσσεται στο λεξιλόγιό μας και κατά προέκταση στη σκέψη μας, εύκολα μας αποπροσανατολίζει λόγω των διαφορετικών διαστάσεών του και είναι δύσκολο να οριστεί με αυστηρότητα και χωρίς ανακρίβειες. Τον Οκτώβρη του 1950 στο άρθρο του [44], το οποίο σηματοδοτεί την αρχή της τεχνητής νοημοσύνης, ο A. M. Turing αντιμετώπισε ένα παρόμοιο πρόβλημα αναζητώντας απάντηση στο ερώτημα, αν οι μηχανές σκέφτονται. Δεδομένου, ότι οι λέξεις «μηχανές» και «σκέφτομαι» είναι μη αυστηρώς ορίσιμες, επέλεξε να αναδιατυπώσει το ερώτημα με ένα καινούριο, σε μορφή παιχνιδιού με συγκεκριμένους κανόνες -το γνωστό *Imitation Game ή Turing Test*. Κατάφερε, έτσι, να συγκεκριμενοποιήσει το ερώτημα και άρα να είναι σε θέση να διερευνήσει την απάντηση. Είναι σημαντικό εδώ να τονίσουμε, ότι το εγχείρημά του ήταν επί της ουσίας η προσομοίωση και όχι ο αυστηρός ορισμός. Εφόσον, δηλαδή, δεν μπορούσε να δώσει έναν «πειστικό» ορισμό για την σκέψη, αποφάσισε να δημιουργήσει μια διαδικασία που την προσομοιάζει. Αυτή είναι πλέον και η κατεύθυνση στην οποία κινείται η τεχνητή νοημοσύνη και σύμφωνα με τον Noam Chomsky, θα πρέπει να εργαστούμε με τον ίδιο τρόπο για να απαντήσουμε στο ερώτημα «Τι είναι γλώσσα;».

Προσπαθώντας να προσεγγίσουμε την έννοια, διαπιστώνουμε πως -ως προς ένα υποκείμενο- η γλώσσα μπορεί να λογιστεί τόσο ως κάτι εσωτερικό (internal), όσο και ως κάτι εξωτερικό (external). Ένας από τους υπέρμαχους της πρώτης οπτικής ήταν ο Otto Jespersen, ο οποίος όρισε τη γλώσσα ως ένα σύστημα, του οποίου η ύπαρξη εντοπίζεται στον νού του ομιλητή βάσει της πεπερασμένης εμπειρίας του και επάγει μια έννοια δομής, με την οποία ο ομιλητής μπορεί να σχηματίζει δικές του προτάσεις, παράγοντας αυθεντικό λόγο. Με δικά του λόγια:

A particular language is a system that comes into existence in the mind of the speaker on the basis of finite experience. This internal system in the mind yields a notion of structure that is definite enough to guide the speaker in framing sentences of his own and crucially 'free expressions' that are typically new to the speaker and the hearer.

Ο Jespersen κάνει λόγο για τις 'free expressions', οι οποίες διαφέρουν από τις τυποποιημένες εκφράσεις, διότι απαιτούν διαφορετική νοητική δραστηριότητα [28]. Ο όρος 'free expressions' αναφέρεται σε προτάσεις, τις οποίες ο ομιλητής θα πρέπει να κατασκευάσει δημιουργικά, α-ντιλαμβανόμενος την εκάστοτε περίσταση. Εξαιρούνται, επομένως, τυποποιημένες προτάσεις που χρησιμοποιούνται ευρέως και σχετίζονται με συγκεκριμένες περιστάσεις. Παράλληλα, προσεγγίζοντας τη γλώσσα εκ των έσω, το κύριο ζήτημα που απασχολεί την επιστήμη της γλωσσολογίας είναι οι αρχές που διέπουν τις γραμματικές όλων των γλωσσών (innate principles ή με την σύγχρονη ορολογία Καθολική Γραμματική, Universal Grammar), δηλαδή το σύνολο των κανόνων που συναποτελούν ένα γλωσσικό σύστημα γενετικά κωδικοποιημένο που προσδίδει σε όλες τις γλώσσες την κοινή θεμελιώδη τους δομή. Είναι φυσικό να ζητάμε την κοινή ταυτότητα όλων των γλωσσών, δεδομένου ότι κατατάσσονται στην ίδια κλάση (αυτή των φυσικών γλωσσών) και άρα, παρόλες τις διαφορές στην προφορά, το αλφάβητο, το λεξιλόγιο, κ.ά., θα πρέπει να κατέχουν κάποιον κοινό πυρήνα, ο οποίος αναμένεται να εντοπίζεται σε ένα περισσότερο αφηρημένο επίπεδο.

Σύμφωνα με τη δεύτερη οπτική (στρουκτουραλιστική/μπιχεϊβιοριστική), η γλώσσα είναι κάτι το εξωτερικό για ένα υποκείμενο, είναι ένα κοινωνικό συμβόλαιο, μια συλλογή από λέξεις που αντιστοιχούν σε εικόνες στο μυαλό των ανθρώπων μιας κοινότητας ή -όπως ισχυριζόταν ο Leonard Bloomfield- το σύνολο του λόγου που εκφέρεται σε μια κοινότητα [5].

Ωστόσο, η γλώσσα είναι ένα δυνητικά άπειρο σύνολο, για το οποίο -σύμφωνα με τον Noam Chomsky [8]- χρειαζόμαστε έναν εσωτερικό μηχανισμό που θα το χαρακτηρίζει και όχι μια λίστα όλων των δυνατών εκφωνημάτων της. Ο ίδιος υποστηρίζει, ότι αντικείμενο της γλωσσολογίας δεν μπορεί να είναι η γλώσσα που χρησιμοποιούν ομάδες ανθρώπων για επικοινωνία (π.χ. Τουρκικά, Γαλλικά, κ.ο.κ), διότι δεν αποτελεί μια καλώς ορισμένη έννοια. Ένα παράδειγμα που παραθέτει είναι τα Γερμανικά έναντι των Ολλανδικών. Σύμφωνα με την στρουκτουραλιστική/μπιχεϊβιοριστική προσέγγιση, θα έπρεπε να θεωρηθούν δύο διαφορετικές γλώσσες, αφού χρησιμοποιούνται από δυο διαφορετικές πολιτισμικές κοινότητες και έχουν βασικές μορφολογικές διαφορές. Ωστόσο, κάποιες γερμανικές διάλεκτοι μοιάζουν περισσότερο στα Ολλανδικά, απ' ότι σε άλλες διαλέκτους των Γερμανικών [9]. Φαίνεται, επομένως, ότι προκύπτουν επικαλύψεις, οι οποίες καθιστούν την ερευνητική διαδικασία μη αποδοτική.

Υιοθετώντας, λοιπόν, την πρώτη οπτική, ο γλωσσολόγος μελετά κάτι που βρίσκεται εσωτερικά (internally) στο μυαλό του ομιλητή (individual), δηλαδή την ώριμη κατάσταση που έχει αποκτηθεί μέσω της εμπειρίας του, καθώς και τη *γλωσσική ικανότητα* (faculty of language), η οποία καθορίζει τις αρχές (innate principles) και καθιστά ικανή τη μετάβαση από πεπερασμένα δεδομένα (εμπειρία) στην ώριμη κατάσταση.

Ο Chomsky ονομάζει αυτήν την κατάσταση *I-Language*. Το 'i' συμβολίζει τους χαρακτηρισμούς internal, individual και intensional. Οι δύο πρώτοι είναι σαφείς. Θα θέλαμε σε αυτό το σημείο να εστιάσουμε στον χαρακτηρισμό intensional (εδώ, επιλέγουμε να διατηρήσουμε την αγγλική ορολογία, ωστόσο στα ελληνικά συχνά μεταφράζεται ως "εντασιακός"). *Intension* χαρακτηρίζουμε την εσωτερική ερμηνεία, το ουσιαστικό περιεχόμενο ενός όρου ή μιας έννοιας, την πραγματική ιδέα που τη διέπει, η οποία αποτελεί και τον τυπικό ορισμό της. Αντίθετα, *extension* ονομάζουμε το εύρος εφαρμογής του όρου ή της έννοιας αυτής, καταγράφοντας συγκεκριμένα αντικείμενα, τα οποία περιγράφει. Μας ενδιαφέρει, επομένως, η πραγματική διαδικασία, ο αλγόριθμος και όχι το (δυνητικά άπειρο) σύνολο που παράγεται. Με άλλα λόγια, η intension και όχι η extension της γλώσσας. Όπως ακριβώς ο Chomsky αντιδιαστέλλει τους όρους γλωσσική ικανότητα και γλωσσική επιτέλεση, έτσι προβαίνει σε μια διάκριση ανάμεσα στα προϊόντα τους: το αφηρημένο γλωσσικό σύστημα, που ευθύνεται για την γλωσσική ικανότητα του ομιλιτή, καθιστά ικανή μετάβαση στην ώριμη κατάσταση της Internalized Language (I-language), ενώ η χρήση της γλώσσας σε συγκεκριμένες επικοινωνιακές συνθήκες αποδίδει την Externalized Language (E-language) [47].

Επομένως, ο όρος *γλώσσα*, με τον τρόπο που θα μας απασχολήσει στη συγκεκριμένη εργασία, θα ταυτίζεται με την I-Language, η οποία αναφέρεται σε κάποια εσωτερική συνιστώσα (internal component) του μυαλού/εγκεφάλου και αποτελεί το βασικό αντικείμενο μελέτης της γλωσσικής ικανότητας [24]. Στόχος μας είναι να κατανοήσουμε τη φύση και λειτουργία του συστήματος που γεννά τη γλώσσα (generative system in intension). Απλούστερα, *εστιάζουμε στους μηχανισμούς που θεμελιώνουν τη γλώσσα* και όχι στη θεώρησή της σαν σύστημα επικοινωνίας.

2.2 Γλωσσική Ικανότητα: Διαχωρισμός FLN - FLB

Η υπόθεση του Chomsky, ότι η γλωσσική ικανότητα του ανθρώπου είναι βιολογικά προκαθορισμένη, στηρίζεται από το γεγονός, ότι η μητρική γλώσσα αναπτύσσεται με φυσικό και αυθόρμητο τρόπο στον άνθρωπο. Θα πρέπει, λοιπόν, να κατανοήσουμε τον *βιολογικό μηχανισμό* που οφείλεται για τη γλώσσα (I-language). Τον γλωσσικό αυτό μηχανισμό επέλεξε να τον ονομάσει Καθολική Γραμματική (Universal Grammar) [10], μια θεωρία που αφορά στην αρ-

[10] Εδώ χρησιμοποιείται στην πραγματικότητα ένας παλιός όρος για να περιγράψει μια καινούρια έννοια. Σύμφωνα με την παραδοσιακή ερμηνεία, στόχος ήταν η αναζήτηση και διατύπωση ομοιοτήτων στις διάφορες γλώσσες.

χική κατάσταση της γλωσσικής ικανότητας του ανθρώπου: αυτή με την οποία γεννιέται [13]. Ζητάμε, δηλαδή, ένα θεωρητικό πλαίσιο, το οποίο να εξηγεί επαρκώς την υπολογιστική βάση του μηχανισμού αυτού. Λαμβάνοντας υπόψη την ευρύτητα των λειτουργιών που συμμετέχουν, κρίνεται απαραίτητη μια κατηγοριοποίηση της γλωσσικής ικανότητας σε δύο περισσότερο περιορισμένες συλλήψεις: την *FLB* (Faculty of Language - Broad Sense) και την *FLN* (Faculty of Language - Narrow Sense)[24].

Η FLB περιλαμβάνει μόνο εκείνα τα εσωτερικά συστήματα, τα οποία συνθέτουν την ικανότητα του ανθρώπου να κατακτά κάποια γλώσσα (i-language). Έτσι, αποκλείονται συστήματα, όπως το μυϊκό ή η μνήμη, που είναι μεν αναγκαία για την πραγμάτωση της γλώσσας, αλλά όχι ικανά για την κατάκτηση αυτής. Περιλαμβάνει, λοιπόν, τη *βιολογική ικανότητα* του ανθρώπου να κατέχει κάποια γλώσσα και αποτελείται από:

1. ένα εσωτερικό υπολογιστικό σύστημα (FLN)

2. τουλάχιστον δυο άλλα εσωτερικά συστήματα του οργανισμού, τα οποία θα καλούμε *sensory-motor* και *conceptual-intentional*.

Το sensory-motor (ή υποσύστημα φωνής/μορφής, όπως εντοπίζεται στην ελληνική βιβλιογρφία) αναφέρεται στην αισθητηριακή επεξεργασία και είναι ένα σύστημα που λαμβάνει πληροφορίες μέσω των αισθήσεων και ύστερα τις επεξεργάζεται και αντιδρά (αντίληψη και παραγωγή ήχου). Το conceptual-intentional σύστημα (ή υποσύστημα σημασίας) είναι υπεύθυνο για το νόημα και την ερμηνεία.

Η FLN ορίζεται ως το **αφηρημένο υπολογιστικό σύστημα** για τη γλώσσα, δηλαδή περιέχει ένα πεπερασμένο σύνολο κανόνων με τους οποίους καταφέρνει να συνδυάσει τις γλωσσικές μονάδες. Μια προτεινόμενη -και χωρίς λεπτομέρειες για τους ακριβείς μηχανισμούς-θεώρηση της εσωτερικής αρχιτεκτονικής του είναι η εξής: *παράγει* εσωτερικές αναπαραστάσεις και ύστερα τις *αντιστοιχίζει* στα υπόλοιπα συστήματα που περιλαμβάνει η FLB. Παρότι η FLN περιέχεται στην FLB και αλληλεπιδρά με τα υπόλοιπα συστήματά της, ορίζεται χωρίς να λαμβάνουμε τα τελευταία υπόψη μας.

Οι Hauser et al. (2002) [24] υποστηρίζουν πως η FLN είναι *αποκλειστικά ανθρώπινη* (ενώ οι υπόλοιποι μηχανισμοί της FLB εντοπίζονται και στα ζώα) και πως αποτελείται τουλάχιστον από τους κύριους υπολογιστικούς μηχανισμούς της αναδρομής. Με άλλα λόγια, η **αναδρομή** αποτελεί τον υπολογιστικό μηχανισμό που θεμελιώνει τη γλωσσική ικανότητα. Η FLN περιγράφεται, λοιπόν, ως ένα **αναδρομικό** υπολογιστικό σύστημα που παράγει και αντιστοιχίζει συντακτικά αντικείμενα στα sensory-motor και conceptual-intentional συστήματα.

Μπορούμε να κάνουμε δυο βασικές παρατηρήσεις για τη γλώσσα, που θα μας βοηθήσουν να κατανοήσουμε το παραπάνω συμπέρασμα:

1. κάθε πρόταση αποτελείται από διακριτές μονάδες (discrete units) και

2. δεν υπάρχει μη αυθαίρετο άνω φράγμα για το μήκος της.

Θα πρέπει εδώ να σημειώσουμε, ότι ο όρος πρόταση αναφέρεται σε "ακολουθίες που παράγονται από τους μηχανισμούς του αφηρημένου γλωσσικού συστήματος. Αυτές οι αφηρημένες

Η καινούρια προσέγγιση, όμως, χρησιμοποιεί τον όρο Καθολική Γραμματική για να περιγράψει το γενετικά καθορισμένο γραμματικό σύστημα που ευθύνεται για την προδιάθεση του ανθρώπου να αναπτύξει γλώσσα.

προτάσεις [είναι που εκ των υστέρων] μετατρέπονται σε εκφωνήματα, δηλαδή αποσπάσματα λόγου που πραγματώνονται σε συγκεκριμένες επικοινωνιακές συνθήκες" [47]. Φυσικά, με τον χαρακτηρισμό διακριτές μονάδες αναφερόμαστε στις λέξεις κάθε πρότασης, το πλήθος των οποίων είναι πάντα κάποιος θετικός ακέραιος αριθμός. Επίσης, αν αγνοήσουμε εξωγενείς παράγοντες, όπως η μνήμη ή η αναπνοή, το μήκος των εκφερόμενων προτάσεων θα μπορούσε να είναι άπειρο. Η γλώσσα (i-language), λοιπόν, αποτελεί την έξοδο ενός υπολογιστικού συστήματος (FLN), το οποίο αποδίδει την αριθμήσιμη απειρία, έννοια γνώριμη από τους φυσικούς αριθμούς. Όπως η συνάρτηση του επομένου εφαρμοζόμενη αναδρομικά, μας δίνει όλους τους φυσικούς αριθμούς, έτσι και η FLN αξιοποιώντας αναδρομικούς μηχανισμούς λαμβάνει ένα πεπερασμένο σύνολο από στοιχεία (εμπειρία) και παράγει ένα δυνητικά άπειρο σύνολο εκφράσεων. Επομένως, η FLN είναι ακριβώς ο μηχανισμός που καθιστά ικανή τη μετάβαση από πεπερασμένα δεδομένα στην γλώσσα (τονίζουμε και πάλι: I-Language).

2.3 Η αναδρομή της FLN

Θα προχωρήσουμε σε περαιτέρω ανάλυση του αφηρημένου υπολογιστικού συστήματος (FLN), με σκοπό να διακρίνουμε τους λόγους για τους οποίους προτάθηκε, ότι χαρακτηρίζεται από αναδρομικές διαδικασίες.

Προς το παρόν, μπορεί στον αναγνώστη να μην είναι εμφανής η σύνδεση του αντικειμένου μελέτης της γλωσσολογίας (με τον βιολογικό προσανατολισμό που περιγράψαμε) με την έννοια της αναδρομής, όπως την αντιλήφθηκαν οι Gödel, Turing και Church. Ωστόσο, η θεωρία αναδρομής και η -μάλλον φυσική- κατεύθυνση που πήρε προς τη θεωρία της υπολογισιμότητας αποδείχθηκαν ιδιαίτερα επιδραστικές στην ανάπτυξη της γενετικής [11] γραμματικής, μιας έννοιας υπό μελέτη από τα τέλη της δεκαετίας του 1940 [8]. Η Θεωρία Αναδρομής αποτέλεσε, επομένως, μια από τις κινητήριες δυνάμεις για να μετατοπιστεί το αντικείμενο μελέτης σε εκείνους τους εσωτερικούς μηχανισμούς που παράγουν προτάσεις και καθορίζουν τον ήχο και τη σημασία τους.

Στην αρχή της ενότητας, αναφερθήκαμε στον έμφυτο μηχανισμό που προορίζεται για τη γλωσσική κατάκτηση και επισημάναμε, ότι θα πρέπει να περιλαμβάνει ένα σύστημα, την Καθολική Γραμματική. Η ίδια, ως αντικείμενο μελέτης της γενετικής προσέγγισης (γενετική γραμματική), είναι ένα πεπερασμένο σύνολο κανόνων, υπεύθυνο για την εσωτερικευμένη γλωσσική ικανότητα του ανθρώπου, για την παραγωγή, δηλαδή, όλων των σωστών προτάσεων της γλώσσας και μόνον αυτών. Σύμφωνα με τη Θεωρία Αρχών και Παραμέτρων των Chomsky και Lasnik [11], η Καθολική Γραμματική στοιχειοθετείται από ένα σύστημα Αρχών και Παραμέτρων. Οι Αρχές είναι οι κανόνες που αντανακλούν εκείνα τα χαρακτηριστικά που παραμένουν ίδια σε όλες τις γλώσσες, ενώ οι Παράμετροι αποτελούν ακριβώς τους κανόνες που διαφοροποιούν τις γλώσσες μεταξύ τους. Η ερμηνεία για τη γλωσσική κατάκτηση που προτείνει η εν λόγω θεωρία είναι πως τα παιδιά γεννιούνται με το σύνολο των Αρχών και των Παραμέτρων, με τις τελευταίες όμως να μην έχουν ακόμη καθοριστεί ως προς τις τιμές τους. Το παιδί κατακτά τη μητρική του, εμπλουτίζοντας το λεξικό και ρυθμίζοντας τις παραμέτρους, έως ότου φτάσει στο τελικό στάδιο, όπου έχει καταφέρει να συγκροτήσει πλέον τη γραμματική της μητρικής του γλώσσας και άρα έχει φτάσει στην ώριμη κατάσταση της I-language.

[11] Η γενετική γραμματική (generative grammar), σε αντίθεση με άλλα είδη (ρυθμιστικές, περιγραφικές γραμματικές) δεν έχει μέλημα να περιγράψει τη γλώσσα, αλλά εστιάζει στους κανόνες που την παράγουν, που την γεννούν.

Η FLN θα θέλαμε να περιλαμβάνει την Καθολική Γραμματική: τις Αρχές και τις Παραμέτρους, δηλαδή ένα εξαρχής καθορισμένο σύνολο κανόνων και άλλο ένα το οποίο θα λάβει την ειδική μορφή του στα πρώτα χρόνια κατά τα οποία το παιδί έρχεται σε επαφή με τη μητρική του.

Γιατί, λοιπόν, η FLN στηρίζεται στην αναδρομή; Είναι σημαντικό να αναγνωρίσουμε τρεις βασικές ιδιότητες που συνοδεύουν την αναδρομή: υπολογισιμότητα, ορισμός μέσω επαγωγής και μαθηματική επαγωγή [46]. Έχουμε συζητήσει εκτενώς και για τις τρεις σε προηγούμενα κεφάλαια. Θα θέλαμε εδώ να σχολιάσουμε πως εντάσσονται στη μελέτη της γλώσσας.

Ξεκινώντας από την υπολογισιμότητα, θυμίζουμε, ότι η έννοια φορμαλίστηκε με επιτυχία και μάλιστα με αρκετούς ισοδύναμους τρόπους. Εκείνος, ο οποίος την περικλύει διαισθητικά σωστότερα, καθώς είναι εγγενώς πειστικός (γεγονός που έχει αναγνωριστεί από τον ίδιο τον Gödel), είναι ο φορμαλισμός που πέτυχε ο Turing. Ο ίδιος κατέληξε με πραγματικά κομψό τρόπο στον ορισμό της πεπερασμένης διαδικασίας (finite procedure). Για να γίνουμε κατανοητοί, θα πρέπει πρώτα να εξηγήσουμε το εξής: ο αναδρομικός ορισμός μιας συνάρτησης καθορίζει με ακρίβεια το σύνολο τιμών της, το οποίο άρα δεν μπορεί να είναι κάποιο τυχαίο σύνολο, αλλά θα περιέχει εκείνα τα στοιχεία και μόνο που καθορίζονται από τον ορισμό της. Με άλλα λόγια, η συνάρτηση αποτελεί τον κανόνα, ο οποίος ορίζει με ακρίβεια τη φύση των στοιχείων του παραγόμενου συνόλου. Είναι επόμενο, ότι δεν προϋποτίθεται η εκτενής κατασκευή ενός συνόλου, ώστε μια αναδρομικά ορισμένη συνάρτηση να μπορεί να το παράγει. Αυτό σε μια μηχανή Turing μεταφράζεται σε ένα πεπερασμένο σύνολο κανόνων, οι οποίοι εκφράζουν τις συνθήκες βάσει των οποίων ο υπολογισμός διακλαδώνεται (conditional branching) στις αντίστοιχες καταστάσεις (configurations). Μια μηχανή Turing θα παράγει ένα συγκεκριμένο σύνολο εξόδων, αν και μόνο αν εισέλθει στις κατάλληλες καταστάσεις (ανάλογα με τις εισόδους). Ωστόσο, το σύνολο όλων των δυνατών εξόδων της είναι *προκαθορισμένο* από το πεπερασμένο σύνολο κανόνων, το οποίο χρησιμοποιεί [12]. Αν θέλαμε να κάνουμε έναν παραλληλισμό, θα μπορούσαμε να φανταστούμε την E-Language σαν το σύνολο των εξόδων που υπολογίζονται ανάλογα με τις εισόδους που λαμβάνει η μηχανή, ενώ την I-Language σαν το σύνολο *όλων* των δυνατών εξόδων, δηλαδή όλων των αντικειμένων που πληρούν τις ορισμένες μέσω των κανόνων συνθήκες. Επομένως, ο προσδιορισμός ενός πεπερασμένου συνόλου κανόνων για τον υπολογισμό των τιμών μιας συνάρτησης δεν εμποδίζει την απειρία του παραγόμενου συνόλου τιμών. Μάλιστα, κατά τον Turing, οι υπολογίσιμες συναρτήσεις είναι εκείνες που μπορούν να υπολογιστούν με πεπερασμένα μέσα, γεγονός που επαληθεύει τη σύλληψη και λειτουργία της Καθολικής Γραμματικής: με πεπερασμένα μέσα κατορθώνουμε να παράγουμε ένα άπειρο σύνολο (i-language).

Είναι γεγονός, πως στην αρχική της μορφή, η γενετική γραμματική είχε παρουσιαστεί από τον Chomsky με τη μορφή κανόνων επανεγγραφής (rewrite rules), εντελώς ανάλογα με τα κανονικά συστήματα του Post ή τους κανόνες επανεγγραφής του Thue, τα οποια έχουν αποδειχθεί ισοδύναμα με τη μηχανή Turing και αντίστοιχα με τις αναδρομικές συναρτήσεις του Gödel. Η γλώσσα θεωρείται ως ένα σύνολο αλφαριθμητικών (strings), που παράγεται από τη γραμματική [Σ,F], όπου Σ ένα πεπερασμένο σύνολο αρχικών συμβόλων και F ένα επίσης πεπερασμένο σύνολο κανόνων επανεγγραφής [10]. Αναφερόμαστε σε κανόνες της μορφής: $\varphi \to \psi$, όπου αντικαθίσταται το φ από το ψ. Η δομή μιας πρότασης εξάγεται καθώς διαπερνάμε μια ακολουθία από τέτοιους κανόνες. Προβαίνοντας σε μια αναλογία με μαθηματική χροιά, η παραπάνω

[12] Αξίζει να σημειώσουμε, ότι η περατότητα αντικατοπτρίζεται και στην ταινία της μηχανής, η οποία μπορεί να είναι απεριόριστη, αλλά σε κάθε υπολογιστικό βήμα έχει χρησιμοποιηθεί πεπερασμένο μήκος της.

διαδικασία μπορεί να λογιστεί ως μια απόδειξη, όπου το σύνολο των αρχικών συμβόλων αντιστοιχίζεται σε ένα σύνολο αξιωμάτων και το πεπερασμένο σύνολο κανόνων επανεγγραφής στους κανόνες συναγωγής [10]. Επίσης, μπορούμε να προβούμε σε μια αναλογία με τους τύπους της λογικής, των οποίων ο ορισμός είναι κατ' εξοχήν αναδρομικός.

Συγκεκριμένα, οι Chomsky και Miller παρουσιάζουν τον εξής ορισμό για τη γραμματική, στο πλαίσιο της γενετικής θεώρησης [12]:

> by a grammar we mean a set of rules that (in particular) recursively specify the sentences of a language. In general, each of the rules we need will be of the form $\varphi_1, ..., \varphi_n \to \varphi_{n+1}$, [...] where each of the φ_i is a structure [...] and the relation '$\to$' is to be interpreted as expressing the fact that if our process of recursive specification generates the structures $\varphi_1, ..., \varphi_n$ then it also generates the structure φ_{n+1}.

Ο παραπάνω ορισμός αντικατοπτρίζει ακριβώς την ιδιότητα της μαθηματικής επαγωγής: δυνατότητα ασφαλούς μετάβασης από πεπερασμένα δεδομένα στην απεριοριστία, εφαρμόζοντας διαδοχικά το ίδιο βήμα και διασφαλίζοντας μια ισχυρή δομή. Πρόκειται ακριβώς για την ιδέα που αξιοποιείται στη λογική για τον ορισμό των τύπων, που προκύπτει αναδρομικά. Επομένως, η προσέγγιση που περιγράψαμε βασίζεται στον ορισμό του Gödel για τις αναδρομικές συναρτήσεις [23]:

> Eine zahlentheoretische Funktion φ heißt rekursiv, wenn es eine endliche Reihe von zahlentheoretischen Funktionen $\varphi_1, \varphi_2 ... \varphi_n$ gibt, welche mit φ endet und die Eigenschaft hat, daß jede Funktion φ_k der Reihe entweder aus zwei der vorhergehenden rekursiv definiert ist oder aus irgend welchen der vorhergehenden durch Einsetzung entsteht oder schließlich eine Konstante oder die Nachfolgerfunktion "$x + 1$" ist.

Ο Gödel εντοπίζει την αναδρομικότητα μιας συνάρτησης σε τρια στάδια. Καταρχάς, η περατότητα της ακολουθίας αντανακλά ακριβώς τον ορισμό του Turing για την υπολογισιμότητα: φτάνουμε στην επιθυμητή συνάρτηση φ έχοντας κάνει n βήματα. Ακόμη, εντάσει στον ορισμό του την έννοια του ορισμού μέσω επαγωγής: η συνάρτηση φ_k της ακολουθίας μπορεί να οριστεί από δύο προηγούμενες που επίσης είναι αναδρομικά ορισμένες ή από οποιαδήποτε από τις προηγούμενες με αντικατάσταση. Τέλος, ο Gödel θωρακίζει τον ορισμό του προσθέτοντας την επιλογή η φ_k να είναι η συνάρτηση του επομένου, εξασφαλίζοντας την ισχυρή δομή που παρέχει η μαθηματική επαγωγή.

Η στενή σχέση της αναδρομής -υπό το πρίσμα των Gödel, Turiing και Post- με την FLN, εξηγείται με εξαιρετική ακρίβεια από τους Watumull et al. [46]:

> The Post formalism represents explicitly the recursiveness of a generative grammar, with outputs recursed (returned) as inputs in the form of recursion applied by Gödel and represented in the stepwise computation of a Turing machine: i.e., definition by induction (definition by recursion) whereby a function f is defined for an argument x by a previously defined value (e.g., f (y), y ¡ x) so as to strongly generate increasingly complex structures carried forward on the tape.

Δηλαδή, μια ακολουθία κανόνων υπό το πρίσμα της γενετικής γραμματικής διαρθρώνεται ως εξής: οι κανόνες λαμβάνουν αναδρομικά (κατά Gödel) σαν είσοδο ακριβώς τις εξόδους των κανόνων που προηγούνται στην ακολουθία, με το κάθε υπολογιστικό βήμα να αντανακλά τη λειτουργία της μηχανής Turing. Ο φορμαλισμός, λοιπόν, που πρότεινε ο Post επαληθεύει την αναδρομικότητα της γενετικής γραμματικής.

Η πρόταση των Hauser et al. (όπως σημειώνουν οι Watumull et al. [46]) έχει αναγνωσθεί εσφαλμένα επανειλλημένως. Η πιο συνηθισμένη παρανόηση είναι η ταύτηση της αναδρομής της FLN με την ικανότητά της -ως μηχανισμός- για εγκιβωτισμούς (syntactical embeddings). Αντίθετα, η θέση των συγγραφέων, όπως αναλύθηκε πιο πάνω, βασίζεται στην έννοια της αναδρομής, όπως διαμορφώθηκε τη δεκαετία του 1930 και αξιοποιεί τις θεμελιακές της ιδιότητες για την περιγραφή της λειτουργίας της FLN ως υπολογιστικό σύστημα, εναρμονίζοντας τη γλωσσολογική προσέγγιση του Chomsky με τα κατάλληλα μαθηματικά αποτελέσματα.

3 Αναδρομή και Εξέλιξη

Κεντρικό σημείο μελέτης της θεωρίας [1] είναι οι αυτοαναφορικοί αλγόριθμοι (self-editing algorithms) και η συμπεριφορά τους μέσα σε ένα περιβάλλον, με το οποίο μπορούν να αλληλεπιδρούν. Φαίνεται, πως η αυτοαναφορά (η δυνητική ικανότητα ενός αλγορίθμου να λαμβάνει σαν είσοδο το πρόγραμμά του) μπορεί σε συνδυασμό με έναν εσωτερικό μηχανισμό εύρεσης μοτίβων, τον οποίο θα καλούμε διαγωνιοποίηση, να συμβάλει στην εξέλιξη των αλγορίθμων.

3.1 Αλγόριθμοι και Αυτοαναφορά

3.1.1 Αλγόριθμοι και Κωδικοί

Στην ενότητα με τα προαπαιτούμενα δώσαμε έναν ορισμό της έννοιας του αλγορίθμου, ενώ στο κομμάτι της ιστορικής αναδρομής (1.1) αναφερθήκαμε στην προσπάθεια αυστηρής τυποποίησης της έννοιας του αλγορίθμου την δεκαετία του 1930 από τους Church, Gödel και Turing. Είναι, λοιπόν, πλέον εμφανές, πως οι αλγόριθμοι μπορούν να ταυτιστούν με τις υπολογίσιμες συναρτήσεις και άρα από τη θέση Church - Turing, ένας αλγόριθμος θα καλείται υπολογίσιμος, όταν μπορεί να υλοποιηθεί με κάποιον μηχανιστικό τρόπο.

Τώρα, μπορεί ένας αλγόριθμος να είναι, ουσιαστικά, τα βήματα για την επίλυση ενός προβλήματος, αλλά είναι λογικό πως υπάρχουν διαφορετικές υλοποιήσεις αυτών των βημάτων. Η υλοποίηση καθορίζεται από το υπολογιστικό μας μοντέλο. Διαφορετικές υλοποιήσεις ενός αλγορίθμου στο ίδιο υπολογιστικό σύστημα χαρακτηρίζονται ως *κωδικοποιήσεις*.

Γενικά, *κωδικοποίηση* ενός συνόλου A σε ένα σύνολο B είναι η τυχαία ένα-προς-ένα συνάρτηση c από το A στο B, που θεωρητικά μας επιτρέπει να ανακτήσουμε ένα τυχαίο στοιχείο $x \in A$ από τον *κωδικό* του $c(x)$ [49].

Οι πιο συχνές κωδικοποιήσεις αντιστοιχίζουν αλγορίθμους σε φυσικούς αριθμούς (π.χ. Gödel numbering). Ο Gödel αντιστοίχισε αρχικά δώδεκα σύμβολα στους φυσικούς αριθμούς από το 1 έως και το 12. Στη συνέχεια, τα γράμματα που αναπαριστούν μεταβλητές τα αντιστοίχισε σε πρώτους αριθμούς μεγαλύτερους του 12 (13,17,19,...). Αυτή η τακτική, του παρείχε τη δυνατότητα να αξιοποιήσει όλες τις επιτρεπτές πράξεις μεταξύ των φυσικών αριθμών για να κατασκευάσει τον μοναδικό αριθμό Gödel πιο σύνθετων λογικών εκφράσεων. Παραδείγματος χάρην, η έκφραση $0 = 0$ αποτελείται από σύμβολα με κωδικούς 6, 5, 6 αντιστοίχως. Για να αποδώσει έναν μοναδικό αριθμό σε αυτήν την έκφραση, ο Gödel χρησιμοποίησε τους τρεις αρχικούς πρώτους αριθμούς (2,3,5) και ύψωσε τον καθέναν στη δύναμη του αριθμού Gödel κάθε συμβόλου στην παραπάνω έκφραση, διατηρώντας τη σειρά τους. Τέλος, πολλαπλασιάζοντας τις δυνάμεις, κατέληξε σε έναν μοναδικό αριθμό για την έκφραση $0 = 0$: $2^6 \times 3^5 \times 5^6$. Αυτός, θα είναι ο μοναδικός κωδικός για το $0 = 0$ και τον ονομάζουμε *αριθμό Gödel* της έκφρασης.

Στην μελέτη μας, οι κωδικοποιήσεις θα αντιστοιχίζουν έναν αλγόριθμο σε μια περιγραφή του, τον κωδικό. *Κωδικό* θα ονομάζουμε ένα αλφαριθμητικό κάποιου αλφάβητου, που χρησιμεύει ως περιγραφή κάποιου αλγορίθμου.

3.1.2 Βασικές Έννοιες και Ορισμοί

Στην μετέπειτα μελέτη μας, θα θεωρούμε ότι ένας αλγόριθμος έχει σαν είσοδο και έξοδο κωδικούς. Είναι σημαντικό να παρατηρήσουμε σε αυτό το σημείο, ότι ένας κωδικός μπορεί να χρησιμεύει για δυο σκοπούς: περιγραφή ενός αλγορίθμου ή περιγραφή δεδομένων. Αν δεν είναι εμφανής ο διαχωρισμός, θα αναφερόμαστε σε *executable codes* και *data codes*, αντιστοίχως. Πολλές φορές, θα χρησιμοποιούμε τον όρο *πρόγραμμα* αντί για τον όρο κωδικός, όταν αναφερόμαστε σε executable codes. Επίσης, οι κωδικοί έχουν δομή, δηλαδή αποτελούνται από διακεκριμένα τμήματα, τα οποία θα ονομάζουμε *διευθύνσεις*. Έτσι, ένας κωδικός μπορεί να αποτελείται από executable και data τμήματα. Η δομή των κωδικών μπορεί να γίνει εμφανής μέσω παρενθέσεων, κάτι που διευκολύνει κατά πολύ τη χρήση διευθύνσεων. Για παράδειγμα, ας δούμε τον κωδικό:

$$((11, 01), (111, 010), (10, (110, 011, 101)))$$

Οι διευθύνσεις ενός κωδικού θα είναι ν-άδες φυσικών αριθμών (π.χ. η διεύθυνση (3,2,1) υποδεικνύει το πρώτο μέρος του δεύτερου μέρους του τρίτου μέρους του παραπάνω κωδικού, δηλαδή το 110) ή κάποια αλγοριθμική συνάρτηση που μας υπολογίζει κάποια διεύθυνση (π.χ. first(c) ή last(c), οι οποίες μας δίνουν την πρώτη και την τελευταία διεύθυνση ενός κωδικού c, αντιστοίχως).

Πριν προχωρήσουμε στη μελέτη των αυτοαναφορικών αλγορίθμων, θα χρειαστούμε έναν γενικότερο τρόπο να κατασκευάζουμε αλγορίθμους χρησιμοποιώντας επιμέρους (απλούστερους). Τους τελευταίους, που υπό μια έννοια θα είναι οι δομικοί λίθοι των αλγορίθμων υπό μελέτη, θα τους ονομάζουμε *εντολές* (instructions).

Έστω $\mathcal{B}$ το σύνολο των εντολών B_i. Ένας αλγόριθμος C θα είναι μια n-αδα εντολών $B_1, ..., B_n$, των οποίων η ροή εκτέλεσης ελέγχεται από έναν αλγόριθμο B, συμβολικά:

$$(B : B_1, ..., B_n)$$

Έχοντας κάνει μια τέτοια ανάλυση, μπορούμε εύκολα να παρακολουθήσουμε τον υπολογισμό, στον οποίο προβαίνει ένας αλγόριθμος δοσμένης κάποιας εισόδου (κωδικός) x. Ας θεωρήσουμε, ότι έχουμε ένα πρόγραμμα c, το οποίο αντιστοιχεί σε έναν αλγόριθμο $alg(c)$. Ο $alg(c)$, δοσμένης κάποιας εισόδου i υπολογίζει διαδοχικά:

$$i = i_1, i_2, ..., i_n,$$

όπου το i_n αποτελεί την έξοδο. *Υπολογιστικό βήμα* θα καλούμε οποιονδήποτε ενδιάμεσο υπολογισμό $i_k \mapsto i_{k+1}$, $k < n$, κατά το οποίο εκτελείται κάποια εντολή του alg(c). Αυτήν θα την ονομάζουμε *ενεργοποιημένη εντολή* (activated instruction). Έτσι, η B_1 είναι η πρώτη εντολή που εκτελείται και ο B ελέγχει ποια εντολή θα εκτελεστεί αμέσως μετά, αναλόγως τον δείκτη i και το στάδιο του υπολογισμού μέχρι εκείνη τη στιγμή. Δοσμένου, λοιπόν, ενός κωδικού x σαν είσοδο, ο αλγόριθμος $(B : B_1, ..., B_n)$ παράγει μια ακολουθία κωδικών

$$x = x_1 \mapsto x_2 \mapsto ...$$

όπου:

1. για κάθε $k = 1, 2, ...$ υπάρχει δείκτης j_k που τρέχει στο $1, ..., n$, τέτοιος ώστε $x_{k+1} = B_{j_k}(x_k)$ και ο B_{j_k} θα ονομάζεται *ενεργοποιημένη εντολή* (activated instruction) του *υπολογιστικού βήματος* $x_k \mapsto x_{k+1}$.

2. υποθέτουμε, ότι $j_1 = 1$, δηλαδή $B_1(x_1)$.

3. για κάθε $k = 1, 2, ...$ το j_{k+1} είναι συνάρτηση της αμέσως προηγούμενης ενεργοποιημένης εντολής και του τρέχοντα υπολογισμού σε αυτό το βήμα., δηλαδή $j_{k+1} = B(j_k, x_{k+1})$.

Οι έννοιες μεταφέρονται με φυσικό τρόπο και για τους κωδικούς που περιγράφουν τους εκάστοτε αλγορίθμους. Επομένως, επιτρέποντας σε κάθε εντολή ενός προγράμματος να είναι επίσης πρόγραμμα, δηλαδή να έχει η μορφή $(C : C_1, ..., C_m)$, οδηγούμαστε στην έννοια των *δομημένων προγραμμάτων* (structured programs), δηλαδή προγραμμάτων που αποτελούνται από εμφωλευμένα προγράμματα (εντολές). Δημιουργείται, έτσι, μια ιεραρχία, που μας επιτρέπει να αναλύουμε τους υπολογισμούς μας κάθε φορά στο επιθυμητό επίπεδο.

Εξηγούμε παρακάτω τους συμβολισμούς που θα χρησιμοποιηθούν:

- $c[b]$:ο κωδικός (η εντολή) b περιέχεται στον c
- $c[\bar{b}]$:η b είναι ενεργοποιημένη εντολή του κωδικού c
- $c[(\theta)b]$:ο κωδικός b περιέχεται στη διεύθυνση θ του c (όπου οι παρενθέσεις γύρω από το θ υποδεικνύουν, ότι το θ δεν είναι μέρος του κωδικού c)
- $c[(\theta)\varnothing]$:στον κωδικό c η διεύθυνση θ είναι άδεια
- $c[\varnothing]$:ο κωδικός c περιέχει μια άδεια διεύθυνση
- $c[(+)\bar{m}]$:στον κωδικό c έχει προστεθεί καινούρια διεύθυνση που περιέχει την ενεργοποιημένη εντολή $\bar{m}$

Στη θέση που υποδεικνύει μια διεύθυνση ενός κωδικού, θα βρίσκουμε κάποια εντολή του ή data τμήματα. Επιπλέον, θεωρούμε, ότι υπάρχουν δυο διευθύνσεις που ονομάζουμε *περιβαλλοντική είσοδος* (environmental input) και *περιβαλλοντική έξοδος* (environmental output), με τις οποίες ο κωδικός θα μπορεί να επικοινωνεί με το περιβάλλον. Στην πρώτη διεύθυνση, θα εμφανίζεται οποιαδήποτε είσοδος από το περιβάλλον, ενώ στη δεύτερη ο κωδικός θα αποθηκεύει τις εξόδους που προορίζει σαν απάντηση στο περιβάλλον. Ο λόγος που τις ονοματίζουμε κατ' αυτόν τον τρόπο είναι για να μην υπάρχει παρανόηση με το γεγονός, ότι οι κωδικοί μας θα λαμβάνουν ολόκληρο τον εαυτό τους σαν είσοδο.

Έστω θ_1 και θ_2 δύο διευθύνσεις. Θα ονομάζουμε *concatenation* των διευθύνσεων αυτών και θα συμβολίζουμε με $\theta_1^\frown\theta_2$, τη διεύθυνση θ_2 της θ_1. Για παράδειγμα $\theta_1 = (1, 2)$ και $\theta_2 = (1)$, τότε το concatenation τους είναι $\theta_1^\frown\theta_2 = (1, 2, 1)$.

Θα θέλαμε να επιτρέψουμε στα προγράμματα-εντολές (στην ουσία πρόκειται για υποπρογράμματα του αρχικού προγράμματος) να έχουν περισσότερες από μια εξόδους. Η πιο απλή περίπτωση που μπορούμε να μελετήσουμε για αρχή είναι μια εντολή R που αντιγράφει την είσοδό της, δηλαδή δοσμένου ενός κωδικού x, επιστρέφει δυο αντίγραφά του:

$$R : x \mapsto \{x, x\}.$$

Στο σημείο αυτό, θα πρέπει να τονίσουμε, ότι εν γνώση μας παραβιάζουμε τον συνήθη συμβολισμό των συνόλων. Επιλέγουμε, ωστόσο αυτή τη διατύπωση, καθώς μας διευκολύνει στη μελέτη μας. Γενικά, αν R η εντολή που αποτελείται από τις *υπο-εντολές* $\{R_1, ..., R_k\}$, τότε:

$$\{R_1, ..., R_k\} : x \mapsto \{R_1(x), ..., R_k(x)\}$$

Η R θα καλείται *proliferating*, τα $R_1(x), ..., R_k(x)$ *τέκνα* (children) ή *άμεσοι απόγονοι* (immediate successors) του x και το σύνολο που τα περιέχει θα συμβολίζεται ως im-suc(x). Όπως και πριν, η $R = \{R_1, ..., R_k\}$ θα ονομάζεται *ενεργοποιημένη εντολή* και αντίστοιχα, η R_i θα είναι η ενεργοποιημένη εντολή του επιμέρους υπολογιστικού βήματος $x \mapsto R_i(x)$ για $i = 1, ..., k$.

Είναι επόμενο, πως ο υπολογισμός που επάγουν οι proliferating κωδικοί θα έχει δενδρική μορφή, καθώς σε κάθε βήμα διακλαδώνεται ο υπολογισμός σε όσα τέκνα υπαγορεύει το πλήθος των υπο-εντολών της εντολής. Επομένως, τα υπολογιστικά βήματα μπορούν να αναπαρασταθούν με ένα δένδρο $(x_t)_{t \in T}$, όπου το σύνολο T των δεικτών επάγει στην πραγματικότητα την δενδρική δομή, για την οποία μιλάμε. Η είσοδος $x = x_\emptyset$ θα αποτελεί την *ρίζα* του δένδρου και κάθε στοιχείο του (δηλαδή κάθε κόμβος) μπορεί να έχει πεπερασμένο αριθμό (ακόμα και 0) τέκνων (άμεσων απογόνων). Ένα *κλαδί* στο υπολογιστικό δένδρο θα είναι μια ακολουθία

$$x_{t_1} \mapsto x_{t_2}, ...$$

τέτοια ώστε για κάθε k, το $x_{t_{k+1}}$ θα είναι τέκνο του x_{t_k}.

Τέτοιους υπολογισμούς θα τους καλούμε επίσης *proliferating* και παρόλο που ακόμα βρισκόμαστε σε πρώιμο στάδιο της μελέτης μας, μπορούμε να αντιληφθούμε, ότι αυξάνουν το πλήθος των κωδικών προσφέροντας, παράλληλα, μια ποικιλομορφία, καθώς τους διαφοροποιούν.

Πληθυσμό κωδικών θα ονομάζουμε, λοιπόν, ένα σύνολο από κωδικούς, οι οποίοι ενδέχεται να έχουν διαφοροποιηθεί μεταξύ τους μετά τη δράση κάποιας ενεργοποιημένης proliferating εντολής.

Τώρα, ας υποθέσουμε, ότι $c = c[b]$, δηλαδή με τον συμβολισμό που εισηγάγαμε ο c περιέχει σε κάποια διεύθυνση τον b σαν εντολή. Έστω ότι με είσοδο x, ο c προβαίνει στον παρακάτω υπολογισμό:

$$x_1 \mapsto x_2 \mapsto ...$$

και έστω, επίσης, ότι σε κάποιο υπολογιστικό βήμα $x_k \mapsto x_{k+1}$ η b χρησιμοποιείται σαν εντολή, δηλαδή η b είναι η ενεργοποιημένη εντολή του βήματος. Τότε, θα λέμε ότι ο $c[b]$ είναι στην κατάσταση $c[\bar{b}]$. Κωδικοί που βρίσκονται σε κάποια κατάσταση θα αναφέρονται ως *state-codes*. Ο αλγόριθμος υπεύθυνος για το επόμενο υπολογιστικό βήμα του state-code $c[\bar{b}]$ ονομάζεται *step-algorithm*, είναι ο αλγόριθμος που αντιστοιχεί στον κωδικό b και συμβολίζεται ως step-alg$(c[\bar{b}])$. Επομένως,

$$\text{step-alg}(c[\bar{b}]) = \text{alg}(b).$$

3.1.3 Αυτοαναφορικοί Υπολογισμοί

Σε αυτήν την υποενότητα, θα επανέλθουμε σε όσα είδαμε στην προηγούμενη με μόνη προσθήκη, ότι τα προγράμματά μας θα λαμβάνουν ως είσοδο τον ίδιο τους τον εαυτό. Η αυτοαναφορά, λοιπόν, εντοπίζεται ακριβώς στο ότι το πρόγραμμα μπορεί να παίρνει σαν είσοδο τον εαυτό του και να δρα πάνω του, τροποποιώντας κάποια μέρη και αφήνοντας άλλα ως έχουν. Η καινοτόμα παρατήρηση είναι πως οποιαδήποτε διαδικασία θέλουμε να εφαρμόσουμε εξωτερικά, η ίδια μπορεί να γίνει και εσωτερικά με τον εξής τρόπο: ας σκεφτούμε, πως έχουμε στα χέρια μας ένα πρόγραμμα, το οποίο απαρτίζεται από διάφορες εντολές και οι οποίες με τη σειρά τους αντιστοιχούν σε κάποιον αλγόριθμο. Αν θέλαμε να επιβάλουμε κάποια αλλαγή σε αυτό το πρόγραμμα, θα επιλέγαμε να το κάνουμε με ένα άλλο πρόγραμμα, το οποίο θα δεχόταν σαν είσοδό του προγράμματα και θα έφερνε εις πέρας τη συγκεκριμένη τροποποίηση. Αυτό είναι φυσικά μια εξωτερική διαδικασία. Τι θα γινόταν όμως αν το πρόγραμμα που θέλουμε να χρησιμοποιήσουμε για να κάνει την επιθυμητή αλλαγή βρισκόταν σαν εντολή σε κάποια διεύθυνση του αρχικού μας προγράμματος;

Ορισμός 3.1.1. *Αυτοαναφορικός* (self-editing) θα καλείται ένας αλγόριθμος C, όταν μπορεί να επεξεργάζεται τον εαυτό του.

Έχουμε ήδη τονίσει, πως πολλές φορές θα χρησιμοποιούμε τον όρο *πρόγραμμα* αντί για τον όρο κωδικός, όταν αναφερόμαστε σε executable codes. Επομένως,

Ορισμός 3.1.2. *Αυτοαναφορικό* (self-editing) θα καλείται ένα εκτελέσιμο πρόγραμμα (executable program) c, το οποίο λαμβάνει σαν είσοδο τον εαυτό του και δίνει σαν έξοδο ένα ή περισσότερα εκτελέσιμα προγράμματα.

Ορισμός 3.1.3. Έστω, ότι η ακολουθία $c_1, c_2, ...$ είναι μια ακολουθία από state-codes. Ένας υπολογισμός

$$c_1 \mapsto c_2 \mapsto ... \mapsto c_k \mapsto ... \tag{9}$$

θα καλείται *αυτοαναφορικός* (self-editing) αν για κάθε k, ο c_k αποτελεί ταυτόχρονα και τον state-code που πραγματοποιεί τον ενδιάμεσο υπολογισμό, αλλά και τον data κωδικό που λαμβάνεται σαν είσοδος. Συμβολικά, ο (9) θα καλείται αυτοαναφορικός, αν

$$c_{k+1} = \text{step-alg}(c_k)(c_k), \text{ για κάθε } k = 1, 2, ...$$

Για προφανείς λόγους, κάθε υπολογιστικό βήμα $c_k \mapsto c_{k+1}$ ενός αυτοαναφορικού υπολογισμού θα καλείται *αυτοαναφορικό βήμα* και θα συμβολίζεται ως self-ed(c_k), δηλαδή:

$$\text{self-ed}(c_k) = \text{step-alg}(c_k)(c_k) = c_{k+1}.$$

Θα πρέπει να σημειωθεί, ότι ο step-alg(c_k) θα είναι κάποια ενεργοποιημένη εντολή που βρίσκεται σε κάποια διεύθυνση του c_k.

Αν $c_k \mapsto c_{k+1}$ ένα αυτοαναφορικό βήμα, τότε και ο state-code c_k και ο step-algorithm step-alg(c_k) θα ονομάζονται αυτοαναφορικοί.

Έχουμε, πλέον, στήσει τα θεμέλια για να μπορέσουμε να αναφερθούμε στο κεντρικό αξίωμα της θεωρίας:

Θεμελιώδες Αξίωμα της Αυτοαναφοράς (Basic Self-Editing Principle)**.** Για κάθε αλγόριθμο B, υπάρχει ένας κωδικός b (στην πραγματικότητα ο κωδικός του), τέτοιος ώστε για κάθε κωδικό $c[\varnothing]$,

$$\text{self-ed}(c[\bar{b}]) = B(c[\bar{b}]) \tag{10}$$

Το παραπάνω αξίωμα μας διαβεβαιώνει στην πραγματικότητα, πως οποιαδήποτε διαδικασία, η οποία περιγράφεται από κάποιον αλγόριθμο B, θα θέλαμε να εφαρμόσουμε στον κωδικό c, μπορούμε να έχουμε το ίδιο αποτέλεσμα, αν συμπεριλάβουμε στις εντολές του c και κάποιον κωδικό b που αντιστοιχεί στον B και ο $c[\bar{b}]$ δράσει στον εαυτό του. Δηλαδή, το αξίωμα μας εξασφαλίζει πως οποιαδήποτε αλγοριθμική διαδικασία θα θέλαμε να εφαρμόσουμε εξωτερικά στον c, την ίδια μπορεί ο c να εφαρμόσει στον εαυτό του, αρκεί το πρόγραμμα που αντιστοιχεί στη διαδικασία να βρίσκεται σαν εντολή σε κάποια διεύθυνση του c και αυτή η εντολή να είναι ενεργοποιημένη.

Μια γενίκευση του παραπάνω αξιώματος προκύπτει αν ως B θεωρήσουμε κάποιον αλγόριθμο με περισσότερες από μια εξόδους και άρα ο κωδικός b που αντιστοιχεί στον B είναι είναι μια proliferating εντολή του c.

Αξίωμα Αναπαραγωγής (Proliferating Principle). Έστω $B_1, ..., B_n$ αλγόριθμοι. Τότε, υπάρχει κωδικός b, τέτοιος ώστε για κάθε κωδικό $c[\varnothing]$,

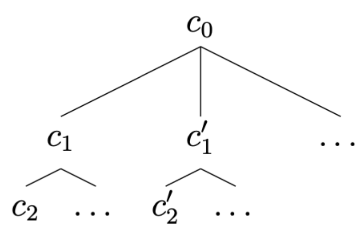
$$\text{self-ed}(c[\bar{b}]) = \{B_1(c[\bar{b}]), ..., B_n(c[\bar{b}])\}. \tag{11}$$

Αν, επιπλέον, οι $b_1, ..., b_n$ είναι κωδικοί για τους αλγορίθμους $B_1, ..., B_n$, αντιστοίχως, τότε μπορούμε να υποθέσουμε, ότι είναι υπο-κωδικοί του b.

Το Αξίωμα Αναπαραγωγής επάγει την έννοια του proliferating αυτοαναφορικού υπολογισμού, δηλαδή ο εν λόγω υπολογισμός οπτικοποιείται με ένα δένδρο $(c_t)_{t\in T}$, του οποίου η ρίζα θα είναι κάποιος state-code $c_\varnothing$ και τα τέκνα κάθε κόμβου c_t του δένδρου προκύπτουν από την εφαρμογή του step-alg(c_t) στον c_t. Συμβολικά,

$$\text{im-suc}(c_t) = \text{self-ed}(c_t) = \text{step-alg}(c_t)(c_t), \text{για κάθε } t \in T.$$

Επομένως, κάθε κλαδί του δένδρου αποτελεί έναν αυτοαναφορικό υπολογισμό.



Σχήμα 1: Δένδρο ενός proliferating αυτοαναφορικού υπολογισμού [50].

Όπως αναφέρθηκε και στην αρχή αυτής της ενότητας, θα θέλαμε οι κωδικοί να βρίσκονται μέσα σε ένα περιβάλλον, με το οποίο να αλληλεπιδρούν, ώστε να έχουν νόημα οι υπολογισμοί, τους οποίους προσπαθούμε να τυποποιήσουμε. Δεδομένου ενός δένδρου αυτοαναφορικών υπολογισμών, κάθε κόμβος του αποτελεί και έναν κωδικό, διαφοροποιημένο κατα έναν βαθμό σε σχέση με τον πρόγονό του. Αν πράγματι υπάρχει διαφοροποίηση (ίσως μετά από κάποιο υπολογιστικό βήμα το τέκνο να είναι αντίγραφο του προγόνου), θα φανεί και στη διεύθυνση του environmental output του εκάστοτε κωδικού. Με τον τρόπο αυτό, το περιβάλλον μπορεί να προβεί σε ενός είδους *επιλογή* (selection). Θα κρίνει, δηλαδή ποιός κωδικός επιβιώνει και ποιός όχι. Όταν ένας κωδικός επιβιώνει, το υπολογιστικό κλαδί δεν διακόπτεται και άρα ο συγκεκριμένος αυτοαναφορικός υπολογισμός συνεχίζεται. Αν όμως κάποιος κωδικός δεν επιβιώσει, το υπολογιστικό κλαδί σταματάει εκεί, δηλαδή ο υπολογισμός διακόπτεται και ο κωδικός δεν αποκτά τέκνα. Ένα κλαδί, του οποίου οι κωδικοί έχουν μέχρι στιγμής επιβιώσει, θα το ονομάζουμε *επιζώσα ακολουθία* (surviving sequence). Αργότερα, θα εισάγουμε και την έννοια της *επιτυχημένης υπακολουθίας* (successful sequence), η οποία παράγεται από μια επιλογή του αυτοαναφορικού κωδικού πάνω στην επιζώσα ακολουθία προγόνων του. Η επιλογή έγκειται στο ότι ο κωδικός θα διαλέξει (με τρόπο που θα δούμε στη συνέχεια) εκείνους τους προγόνους, οι οποίοι αποδείχθηκαν να συμπεριφέρονται καλά ως προς τις απαιτήσεις του περιβάλλοντος.

3.1.4 Αυτοαναφορικοί Υπολογισμοί Πλήρους Μνήμης

Μέχρι στιγμής έχουμε εξετάσει, πως ένας κωδικός c μπορεί ακολουθώντας αυτοαναφορικά υπολογιστικά βήματα να διαφοροποιεί κομμάτια του εαυτού του. Αν πρόκειται να προχωρήσουμε σκεπτόμενοι όχι απλώς τη διαφοροποίηση, αλλά την *εξέλιξη* ενός τέτοιου κωδικού (δηλαδή υπό μια έννοια τη διαφοροποίηση με κάποια αιτία), θα θέλαμε οι κώδικές μας να κατέχουν κάποια ικανότητα *μνήμης*. Να έχουν, δηλαδή, πρόσβαση σε προηγούμενες "εκδόσεις" του εαυτού τους. Με άλλα λόγια, να μπορούν να "κοιτάζουν" πίσω στο κλαδί υπολογισμού τους.

Έστω $(c_t)_{t \in T}$ ένας αυτοαναφορικός υπολογισμός. Για κάποιο c_t, θα καλούμε ιστορία (history) του την ακολουθία:

$$c_{t_1} \mapsto c_{t_2} \mapsto ... \mapsto c_{t_n} = c_t,$$

όπου $c_{t_1} = c_\varnothing$ η ρίζα του δένδρου και για κάθε $i < n$, $c_{t_{i+1}}$ είναι τέκνο του c_{t_i}.
Συμβολίζουμε την ιστορία του c_t με $\text{hist}(c_t)$.

Έχοντας τα παραπάνω κατά νου και δεδομένου ενός state-code $c = c_\varnothing$, θα ονομάζουμε αυτοαναφορικό υπολογισμό πλήρους μνήμης (complete memory self-editing computation), ένα δένδρο $(c_t)_{t \in T}$, τέτοιο ώστε για κάθε $t \in T$,

$$\text{im-suc}(c_t) = \text{step-alg}(c_t)(\text{hist}(c_t)).$$

Δηλαδή, κάθε state-code υπολογίζει τα τέκνα του εφαρμόζοντας τον κωδικό του, όχι μόνο στον εαυτό του, αλλά σε ολόκληρη την "ακολουθία" εαυτών του, δηλαδή σε αυτό που ονομάσαμε ιστορία του.

Εντολή πλήρους μνήμης θα ονομάζουμε έναν αλγόριθμο B, που υπολογίζει τα τέκνα ενός κωδικού c_t, λαμβάνοντας σαν είσοδο την ιστορία του c_t. *Πρόγραμμα πλήρους μνήμης* θα καλούμε ένα πρόγραμμα που χρησιμοποιεί εντολές πλήρους μνήμης.
Το Θεμελιώδες Αξίωμα της Αυτοαναφοράς μπορεί να αποδειχθεί και για την περίπτωση αυτοαναφορικών υπολογισμών πλήρους μνήμης. Παρακάτω, αρκούμαστε στο να το διατυπώσουμε:

Θεμελιώδες Αξίωμα της Αυτοαναφοράς (Έκδοση πλήρους μνήμης)**.** Έστω B τυχαίος αλγόριθμος με κωδικό b. Έστω, επίσης, $c_1, ..., c_n$ η ιστορία του c_n σε ένα υπολογιστικό δένδρο. Αν ο κωδικός b είναι ενεργοποιημένος στο c_n, τότε ο αυτοαναφορικός υπολογισμός πλήρους μνήμης του c_n ισούται με το $B(c_1, ..., c_n)$.

Σε ελεύθερη διατύπωση, το συγκεκριμένο αξίωμα μας διαβεβαιώνει, πως αν κοιτάζοντας την ιστορία ενός κωδικού c_n, αποφασίσουμε να κάνουμε οποιαδήποτε αλλαγή (που γίνεται αλγοριθμικά), την ίδια αλλαγή μπορεί ο c_n να την κάνει στον εαυτό του, αρκεί ο υπολογισμός να είναι αυτοαναφορικός, η επιθυμητή αλλαγή να υπάρχει σαν ενεργοποιημένη εντολή στον c και ο c_n να "βλέπει" την ιστορία του.

Αν και εύκολα δίνεται η εντύπωση, ότι ένας αυτοαναφορικός υπολογισμός πλήρους μνήμης είναι πιο ισχυρός από έναν απλό, κάτι τέτοιο δεν ισχύει στην πραγματικότητα. Ο λόγος είναι πως ένας αυτοαναφορικός υπολογισμός πλήρους μνήμης μπορεί να αναπαρασταθεί από έναν απλό ως εξής:

Ας θεωρήσουμε μια ένα προς ένα αντιστοιχία

$$\phi : (c_1, ..., c_n) \mapsto [c_1, ..., c_n],$$

η οποία λαμβάνει μια ακολουθία n κωδικών (την ιστορία του c_n) και την αντιστοιχίζει σε ένα πρόγραμμα που περιλαμβάνει τους κωδικούς αυτούς, διατηρώντας τη σειρά τους. Έτσι, αντί να θεωρούμε, ότι σε κάθε υπολογιστικό βήμα ο step-alg του c_n θα λαμβάνει την ιστορία του c_n σαν όρισμα, θα θεωρούμε, ότι λαμβάνει μόνο τον εαυτό του, ο οποίος όμως περιέχει αποθηκευμένες τις προηγούμενες εκδόσεις του, δηλαδή η ιστορία του c_n βρίσκεται αποθηκευμένη στον $[c_1, ..., c_n]$. Σε αυτόν τον κωδικό θα υποθέτουμε, πως καμία άλλη διεύθυνση πέρα από την τελευταία ($\text{last}([c_1, ..., c_n]) = c_n$) δεν είναι ενεργή. Επομένως, ένας αυτοαναφορικός υπολογισμός θα ονομάζεται *memory storing*, αν είναι της μορφής:

$$[c_1], [c_1, c_2], ..., [c_1, c_2, ..., c_n].$$

Για να γίνει κατανοητός ο υπολογισμός, ας αναλύσουμε ακριβέστερα το κάθε υπολογιστικό βήμα. Καταρχάς, λαμβάνουμε σαν είσοδο τον κωδικό c_1, ο οποίος μπορεί να θεωρήθεί ίδιος με τον $[c_1]$. Τώρα, ο step-alg$([c_1])([c_1])$ δίνει $[c_1, c_2]$, εφόσον πρόκειται για αυτοαναφορικό υπολογισμό. Αυτό σημαίνει, ότι ο c_1 περιέχει σε κάποια διεύθυνση έναν κωδικό m που λειτουργεί ως ενεργοποιημένη εντολή του βήματος. Δηλαδή, αντί για c_1, έχουμε $c_1[\bar{m}]$. Κατ' αντιστοιχία, στο επόμενο υπολογιστικό βήμα step-alg$([c_1, c_2])([c_1, c_2])$=$[c_1, c_2, c_3]$, όπου στην πραγματικότητα είναι $[c_1, c_2[\bar{m}]]$ κ.ο.κ. έως ότου φτάσουμε στο $[c_1, c_2, ..., c_n[\bar{m}]]$. Είναι σημαντικό να τονίσουμε, ότι ο $c_k[\bar{m}]$ εκτελείται κάθε φορά με είσοδο $[c_1, c_2, ..., c_k]$. Επίσης, θα μπορούσαμε να διατηρήσουμε για κάθε υπολογιστικό βήμα $c_k \mapsto c_{k+1}$ τον συμβολισμό

$$[c_1[\bar{m}], c_2[\bar{m}], ..., c_k[\bar{m}]],$$

αρκεί να έχουμε στον νού μας, πως η μόνη ενεργή διεύθυνση θα είναι η τελευταία και άρα ο συμβολισμός $\bar{m}$ στις προηγούμενες θα μας υπενθυμίζει απλώς, ποιά μέλη υπήρξαν ενεργά, δηλαδή χρησιμοποιείται καθαρά για μνημονικό σκοπό.

Παρατήρηση. *Το παρακάτω λήμμα μας εξασφαλίζει, ότι μπορούμε να εκφράσουμε οποιονδήποτε αυτοαναφορικό υπολογισμό πλήρους μνήμης, με έναν απλό αυτοαναφορικό, ο οποίος αποθηκεύει μνήμη.*

Λήμμα Μνήμης (Memory Lemma)**.** Έστω $(c_t)_{t\in T}$ αυτοαναφορικός υπολογισμός πλήρους μνήμης. Τότε, υπάρχει κωδικός m, τέτοιος ώστε η ρίζα $c_\varnothing[(+)\bar{m}]$ να δίνει τον υπολογισμό $(x_t)_{t\in T}$, όπου για κάθε $t \in T$, αν $c_1, c_2, ..., c_n = c_t$ η ιστορία του c_t στο $(c_t)_{t\in T}$, τότε

$$x_t = [c_1[(+)\bar{m}], ..., c_n[(+)\bar{m}]].$$

Το θεμελιώδες Αξίωμα της Αυτοαναφοράς για υπολογισμούς πλήρους μνήμης, σε συνδυασμό με το Λήμμα Μνήμης μας επιτρέπουν να μελετάμε αυτοαναφορικούς υπολογισμούς πλήρους μνήμης χρησιμοποιώντας μοντέλα απλών αυτοαναφορικών αλγορίθμων που αποθηκεύουν μνήμη, όπως είδαμε παραπάνω. Αυτό το αποτέλεσμα, θα το χρησιμοποιήσουμε στη μετέπειτα μελέτη μας κατά κόρον.

3.2 Διαγωνιοποίηση

Η διαγωνιοποίηση (diagonalization) αποτελεί *τον μηχανισμό για τη συστηματική εύρεση μοτίβων σε έναν πληθυσμό κωδικών*. Είναι το εργαλείο που φαίνεται να παρέχει στους κωδικούς τη δυνατότητα να εξελίσσονται, προσαρμοζόμενοι στις απαιτήσεις του περιβάλλοντος. Αν και διαβάζοντας τον όρο διαγωνιοποίηση, οι αρχικοί συνειρμοί του αναγνώστη τον κατευθύνουν προς την ομώνυμη αποδεικτική μέθοδο (βλ. απόδειξη του Cantor για την ύπαρξη διαφορετικών απείρων) που οδηγεί σε άτοπο, εδώ ο όρος επιλέχθηκε να περιγράφει μια "θετικά προσημασμένη διαδικασία", που συμβάλει στην εξέλιξη των αυτοαναφορικών αλγορίθμων. Αφού μελετήσουμε τον εν λόγω μηχανισμό, θα γίνει πιο κατανοητή η επιλογή της συγκεκριμένης ορολογίας. Ας δούμε, λοιπόν, με περισσότερη λεπτομέρεια τα διάφορα σημεία.

Η διαγωνιοποίηση θα εφαρμοστεί τόσο σε επιζώσες, όσο και σε επιτυχημένες ακολουθίες. Έστω, λοιπόν μια τέτοια ακολουθία:

$$c_1 \mapsto c_2 \mapsto ... \mapsto c_n.$$

Προφανώς, αν κανείς παρατηρήσει την ακολουθία, θα λάβει πληροφορίες ως προς την εξέλιξη και επιβίωσή της, χωρίς να χρειάζεται να γνωρίζει εκ των προτέρων τα κριτήρια της φυσικής επιλογής (δηλαδή της διαδικασίας που χρησιμοποιεί το περιβάλλον για να επιτρέψει την αναπαραγωγή μόνο εκείνων των κωδικών που πληρούν τις απαιτήσεις του περιβάλλοντος) ή της μεθόδου επιβράβευσης (κάποια μέθοδος που χρησιμοποιεί το περιβάλλον για να "βαθμολογήσει" την επίδοση των κωδικών ως προς την προσαρμοστικότητά τους στις απαιτήσεις του περιβάλλοντος). Αυτό που φαίνεται, λοιπόν, απαραίτητο για την εξέλιξη των αλγορίθμων σε ένα περιβάλλον είναι μια διαδικασία που θα αναγνωρίζει μοτίβα (pattern recognition) και -στο πλαίσιο των αυτοαναφορικών υπολογισμών- θα περιέχεται σαν εντολή στους κωδικούς. Ποιό είναι το όφελος;

1. Η διαδικασία αυτή θα επιτρέπει στον εκάστοτε κωδικό να αποκτά μια εικόνα του πως θα πρέπει να μεταβληθεί, ώστε να επιβιώσει, ψάχνοντας για στοιχεία στους προγόνους του.

2. Αν αυτή η διαδικασία περιέχεται ως εντολή στον κωδικό, τότε η ίδια κοιτάζοντας την ιστορία του, θα μπορεί να ψάχνει για μοτίβα και στον εαυτό της, γεγονός που προωθεί την εξέλιξη της ίδιας της διαδικασίας που ευθύνεται για την εξέλιξη. Αν θέλουμε να το θέσουμε άτυπα, ο κωδικός όχι απλώς θα μαθαίνει, αλλά και *θα μαθαίνει πως να μαθαίνει.*

Το φυσικό ερώτημα για τον αναγνώστη σε αυτό το σημείο είναι πως θα μπορούσε να μοιάζει ο αλγόριθμος της διαγωνιοποίησης. Η απάντηση σε αυτό είναι αρχικά ανησυχητική: οποιοσδήποτε αλγόριθμος που βρίσκει μοτίβα. Ωστόσο, η εξήγηση κρύβεται βαθιά στην καρδιά της συγκεκριμένης θεωρίας. Από τη στιγμη που η ίδια η διαγωνιοποίηση θα διαγωνιοποιεί τον εαυτό της, οποιονδήποτε αλγόριθμο (που αναγνωρίζει μοτίβα) και να χρησιμοποιήσει κανείς στην αρχή, τελικά αυτός θα εξελιχθεί με τρόπο κατάλληλο σύμφωνα με το περιβάλλον στο οποίο βρίσκονται οι κωδικοί. Έτσι, στην πραγματικότητα δεν έχει σημασία η αρχική επιλογή, αλλά το γεγονός, ότι η διαγωνιοποίηση εκτελείται αυτοαναφορικά.

Ο όρος *μοτίβο* (pattern) θα πρέπει να νοείται ως ένας αλγόριθμος R, τέτοιος ώστε για κάθε $k < n$, $R(c_k) \sqsubseteq c_{k+1}$. Δηλαδή ο c_n υπό μια έννοια θα "δοκιμάζει" αλγορίθμους πάνω στους κωδικούς c_k της ιστορίας του, για να ελέγχει κατά πόσο είναι αυτοί που ευθύνονται για τον υπολογισμό του υπολογιστικού βήματος $c_k \mapsto c_{k+1}$.

Έστω, λοιπόν, ότι έχουμε μια ακολουθία κωδικών που έχει παραχθεί από κάποιον αυτοαναφορικό υπολογισμό:

$$c_1[\delta_1], c_2[\delta_2], ..., c_n[\bar{\delta_n}] \tag{12}$$

και έστω Δ ο αλγόριθμος της διαγωνιοποίησης (δηλαδή ο αλγόριθμος που αναγνωρίζει μοτίβα), συμβολικά $\Delta = \mathrm{alg}(\delta_n)$. Λαμβάνοντας την (12) σαν είσοδο, ο Δ περιμένουμε να μπορεί να αναγνωρίσει εξελικτικά μοτίβα στην ακολουθία.

3.2.1 Αποφάσεις ενός αυτοαναφορικού αλγορίθμου

Η διαγωνιοποίηση θα ελέγχει με κάποια σειρά κωδικούς, ώστε να βρει εκείνον που ταιριάζει στην ακολουθία των προγόνων. Η απόφαση έγκειται στο πως ο αυτοαναφορικός αλγόριθμος c πρόκειται να χρησιμοποιήσει έναν κωδικό που η διαγωνιοποίηση αποφάσισε, ότι είναι ο κατάλληλος. Οι διαφορετικές επιλογές είναι τρεις:

1. *Προσωρινές Αποφάσεις* (Temporary Decisions). Ο c αποφασίζει να εφαρμόσει έναν κωδικό r στο περιεχόμενο μιας καινούριας διεύθυνσης θ, χωρίς όμως να καταχωρεί τον κωδικό σε αυτή τη διεύθυνση. Πρόκειται, δηλαδή, για μια απόφαση που δε χαρακτηρίζεται από μονιμότητα. Ο κωδικός r θα ονομάζεται temporary differentiating.

2. *Μόνιμες Αποφάσεις* (Permanent Decisions). Ο c αποφασίζει να καταχωρήσει τον κωδικό r σε μια καινούρια διεύθυνση θ και άρα πλέον ο r να αποτελεί εντολή του. Ο κωδικός r θα ονομάζεται permanent differentiating.

3. *ϕ-differentiating Αποφάσεις*. Ο c αποφασίζει να χρησιμοποιήσει έναν κωδικό r για να διαφοροποιήσει την δοσμένη ϕ διεύθυνσή του, δηλαδή να δράσει με τον $\mathrm{alg}(r)$ στο περιεχόμενο y της ϕ. Οι ϕ-differentiating αποφάσεις μπορεί να είναι είτε προσωρινές είτε μόνιμες. Ο κωδικός r θα ονομάζεται ϕ-differentiating.

3.2.2 Μορφές Διαγωνιοποίησης

Θα μελετήσουμε τρεις μορφές διαγωνιοποίησης: την ακολουθιακή (sequential), την στατιστική (statistical) και την παράλληλη (parallel).

Ακολουθιακή διαγωνιοποίηση. Όπως υποδεικνύει και το όνομά της, πρόκειται για διαγωνιοποίηση που εφαρμόζεται σε ακολουθίες κωδικών. Θα ονομάζουμε *σύστημα απόφασης* (decision system) έναν αλγόριθμο Δ, για τον οποίο ισχύει:

> ο Δ παίρνει μια απόφαση D και αποτελείται από δυο μέρη, τον *searcher* Δ_s και τον *tester* Δ_t.

Ο searcher είναι υπεύθυνος να προτείνει κωδικούς $r_1, r_2, \ldots$ στη σειρά, τους οποίους θα ονομάζουμε *προτεινόμενους κωδικούς* (proposed codes). Οι προτεινόμενοι κωδικοί κατασκευάζονται από ένα αρχικό σύνολο κωδικών, οι οποίοι συνδυάζονται χρησιμοποιώντας ένα πεπερασμένο σύνολο διαδικασιών κατασκευής νέων κωδικών από ήδη υπάρχοντες (π.χ. σύνθεση). Το σύνολο των προτεινόμενων κωδικών μπορεί να είναι είτε πεπερασμένο, είτε άπειρο. Ανάλογα με την προτεραιότητα (priority) των κωδικών, καθορίζεται και η θέση τους στην ακολουθία. Πιο σημαντικοί κωδικοί εμφανίζονται νωρίτερα από άλλους. Αν ένας κωδικός έχει μεγάλη προτεραιότητα και άρα εμφανίζεται νωρίς στην ακολουθία, θα ονομάζεται *απλός* (simple).
Δουλειά του tester Δ_t είναι να διαλέξει τον πιο απλό από τους κωδικούς που προτείνει ο

Δ_s, χρησιμοποιώντας ένα απλό αλγοριθμικό τεστ T (δηλαδή έναν αλγόριθμο που απαντάει true ή false), έτσι ώστε $T(r) = true$. Δηλαδή, ο T θα διαλέξει τον πρώτο r στην ακολουθία των προτεινόμενων κωδικών, για τον οποίο ισχύει $T(r) = true$. Τον T θα τον ονομάζουμε *αλγόριθμο ελέγχου του* Δ_t (testing algorithm).
Δεδομένου ενός αλγορίθμου S που λαμβάνει σαν είσοδο την ιστορία ενός κωδικού c και δίνει σαν έξοδο μια επιλεγμένη υποακολουθία αυτής, θα αναφερόμαστε στον εξής αλγόριθμο ελέγχου T:

> Έστω $c_1, c_2, ..., c_n$ η έξοδος του S. Διάλεξε τον πιο απλό κωδικό r της προτεινόμενης από τον searcher Δ_s ακολουθίας κωδικών $r_1, r_2, ...$, έτσι ώστε το r να ταιριάζει στην ακολουθία $c_1, c_2, ..., c_n$, δηλαδή
>
> $$\text{για κάθε } i < n, \text{alg}(r)(c_i) \sqsubseteq c_{i+1}$$
>
> και χρησιμοποίησέ τον σαν temporary ή permanent differentiating κωδικό.

Επομένως, ο όρος ακολουθιακή διαγωνιοποίηση χρησιμοποιείται για να περιγράφει τη λειτουργία ενός συστήματος απόφασης $\Delta = (\Delta_s, \Delta_t)$, όπου ο Δ_t χρησιμοποιεί το παραπάνω τεστ T. Η ακολουθία $c_1, c_2, ..., c_n$ θα ονομάζεται *ακολουθία ελέγχου* της διαγωνιοποίησης.

Αξίωμα της Ακολουθιακής Διαγωνιοποίησης (Απλή Μορφή). Έστω Δ ένας αλγόριθμος ακολουθιακής διαγωνιοποίησης με κωδικό $\delta = \delta_n$ και $c = c[\bar{\delta}]$ αυτοαναφορικός κωδικός. Υποθέτουμε, ότι

$$c_1[\delta_1], c_2[\delta_2], ..., c_n[\bar{\delta_n}] = c[\bar{\delta}]$$

είναι η ιστορία του c. Τότε,

$$\text{self-ed}(c[\bar{\delta}]) = \Delta(c_1[\delta_1], c_2[\delta_2], ..., c_n[\bar{\delta_n}] = c[\bar{\delta}]).$$

Απόδειξη. Η απόδειξη είναι άμεση συνέπεια του Θεμελιώδους Αξιώματος της Αυτοαναφοράς (για υπολογισμούς πλήρους μνήμης) και του Λήμματος Μνήμης. □

Στατιστική διαγωνιοποίηση. Πρόκειται για μια γενικευμένη μορφή ακολουθιακής διαγωνιοποίησης, κατά την οποία επιλέγουμε το r εκείνο, το οποίο ταιριάζει σε κάποιο ποσοστό των μεταβάσεων $c_i \mapsto c_{i+1}$, για $i < n$ και το χρησιμοποιούμε με την ίδια συχνότητα που αυτό φαίνεται να ταιριάζει στην ακολουθία. Αυτό μπορεί να αφορά τόσο στη συχνότητα χρήσης του κατά τον υπολογισμό των απογόνων (για proliferating υπολογισμούς), όσο και στην πιθανότητα με την οποία ο r θα χρησιμοποιηθεί (non-proliferating υπολογισμοί). Συγκεκριμένα, δοσμένου ενός searcher Δ_s μπορούμε να αλλάξουμε τη λειτουργία του tester Δ_t να ελέγχει τη σχετική συχνότητα με την οποία ένας προτεινόμενος κωδικός r ταιριάζει στις μεταβάσεις $c_i \mapsto c_{i+1}$, $i < n$, της ακολουθίας $c_1, c_2, ..., c_n$ και να αποφασίζει να χρησιμοποιεί τον r με την ίδια σχετική συχνότητα ως differentiating κωδικού. Στην περίπτωση ενός proliferating υπολογισμού, η στατιστική διαγωνιοποίηση θα οδηγήσει σε διαφορετικές "μεταλλάξεις" του κωδικού με αποτέλεσμα την ποικιλομορφία στο σύνολο των άμεσων απογόνων.

Παράλληλη διαγωνιοποίηση. Σε αντίθεση με την ακολουθιακή διαγωνιοποίηση, όπου η σειρά στην ακολουθία παίζει σημαντικό ρόλο για να εντοπίζονται είτε εξελικτικά βήματα της ίδιας μορφής σε έναν αυτοαναφορικό κωδικό είτε ομαλές μεταβολές του περιβάλλοντος, στην παράλληλη διαγωνιοποίηση δεν λειτουργούμε πάνω σε ακολουθία κωδικών. Αντ' αυτού, η παράλληλη διαγωνιοποίηση αναζητά μοτίβα σε ένα *σύνολο μεταβάσεων*. Πιο συγκεκριμένα, ας υποθέσουμε, ότι έχουμε έναν αυτοαναφορικό κωδικό c και θέλουμε να αναζητήσουμε εξελικτικά μοτίβα στην επιζώσα ιστορία του $c_1, c_2, ..., c_n = c$. Θα συμβολίσουμε με θ_i τη διεύθυνση της περιβαλλοντικής εισόδου του κωδικού και θ_o της περιβαλλοντικής εξόδου. Επίσης, ο συμβολισμός

$$c \restriction \theta$$

υποδηλώνει την τιμή του περιεχομένου της διεύθυνσης θ του c. Ας υποθέσουμε, τώρα, ότι η απαίτηση του περιβάλλοντος είναι μια απλή σχέση R μεταξύ της εισόδου και της εξόδου ενός κωδικού και πως αυτή η σχέση παραμένει αμετάβλητη καθόλη τη διάρκεια του πειράματος. Εφόσον η $c_1, c_2, ..., c_n = c$ αποτελεί επιζώσα ακολουθία, θα πρέπει για κάθε $k < n$ το περιεχόμενο του c_k στη διεύθυνση θ_i να συνδέεται με το περιεχόμενο του c_{k+1} στη διεύθυνση θ_o με τη σχέση R. Συμβολικά,

$$R(c_k \restriction \theta_i) = c_{k+1} \restriction \theta_o, \text{ για κάθε } k < n.$$

Η αναζήτηση του κωδικού r θα πραγματοποιηθεί πάνω σε ένα σύνολο μεταβάσεων:

$$\{c_k \mapsto c_{k+1} : k < n\} = \{c_1 \mapsto c_2, c_2 \mapsto c_3, ..., c_{n-1} \mapsto c_n\},$$

όπου αναζητούμε κωδικό r, τέτοιον ώστε για όλα τα $k < n$, $\mathrm{alg}(r)(c_k) \sqsubseteq c_{k+1}$. Η έννοια της παράλληλης διαγωνιοποίησης παίζει ιδιαίτερα σημαντικό ρόλο στην εμφάνιση κοινών χαρακτηριστικών των κωδικών ενός συνόλου $\{s_1, ..., s_n\}$ και άρα να συμβάλει σε αφαιρετικές διαδικασίες. Ας υποθέσουμε, ότι όλοι οι κωδικοί περιέχουν μια εσωτερική αναπαράσταση του περιβάλλοντος. Σε αυτήν την περίπτωση, η παράλληλη διαγωνιοποίηση πάνω στα περιβαλλοντικά αντικείμενα $s_1, ..., s_n$ μπορεί να οδηγήσει στην ανάδυση της κοινής φύσης τους. Να τα κατατάξει, δηλαδή, σε μια κλάση, δημιουργώντας ένα καινούριο επίπεδο αφαίρεσης. Για παράδειγμα, αν κάθε ένα από τα $s_1, ..., s_n$ αποτελεί ένα ζεύγος αντικειμένων, τότε διαγωνιοποιώντας πάνω στο σύνολο $\{s_1, ..., s_n\}$, ο κωδικός μπορεί να δημιουργήσει την έννοια του αριθμού 2.

3.2.3 Διαγωνιοποίηση πάνω σε επιτυχημένες υπακολουθίες

Ορισμός 3.2.1. *Επιτυχημένη υπακολουθία* (successful sub-sequence) καλούμε την ακολουθία κωδικών που προκύπτει από μια εσωτερική επιλογή του αυτοαναφορικού αλγορίθμου πάνω στους επιτυχημένους (με βάση τη δική του κρίση) υπολογισμούς του.

Η διαγωνιοποίηση πάνω σε επιτυχημένες υπακολουθίες προϋποθέτει την ικανότητα του αυτοαναφορικού αλγορίθμου c να επιλέγει την ακολουθία ελέγχου (testing sequence) πάνω στην οποία δρα η διαγωνιοποίηση. Αυτό με τη σειρά του απαιτεί τη δυνατότητα να προσθέτει ο c στην ακολουθία ελέγχου οποιονδήποτε c_k από την ιστορία του. Κάτι τέτοιο αποτελεί μια καθαρά αλγοριθμική διαδικασία:

Έστω $\delta' = \delta[(+\theta_t)x_1, x_2, ..., x_m]$ ο κωδικός ενός αλγορίθμου διαγωνιοποίησης, όπου $x_1, x_2, ..., x_m$ η ακολουθία ελέγχου. (Υπενθυμίζουμε: Ο alg(δ') βρίσκει τον απλούστερο κωδικό r που ταιριάζει στην ακολουθία $x_1, x_2, ..., x_m$). Τώρα, αν $c_1, c_2, ..., c_n = c_n[(\theta)\delta']$ η ιστορία του c_n και $k \leq n$, τότε η συνάρτηση $c_1, ..., c_k, ...c_n = c_n[(\theta^\frown\theta_t)x_1, x_2, ..., x_m] \mapsto c_n[(\theta^\frown\theta_t)x_1, x_2, ..., x_m, c_k]$ προσθέτει το c_k στην ακολουθία και είναι προφανώς αλγοριθμική, άρα μπορεί να δώσει στον αλγόριθμο c την παραπάνω δυνατότητα που ζητήσαμε.

Για να μπορέσουμε να προχωρήσουμε τη μελέτη μας, θα χρειαστούμε την έννοια του *κύκλου* (cycle).
Έστω $x_1, x_2, ..., x_n = x$ η ιστορία του κωδικού x. Υποθέτουμε, ότι για κάθε k οι υπολογισμοί αποθηκεύουν μνήμη, δηλαδή για κάθε k, $x_k = [c_1, ..., c_k]$. Τώρα, αν R αλγοριθμική διαδικασία, τότε οι υπολογισμοί:

$$[c_1, ..., c_n] \mapsto R(c_1) \;\; \text{και} \;\; [c_1, ..., c_n] \mapsto [c_1, ..., c_n, R(c_1)]$$

είναι και οι δυο αλγοριθμικοί και άρα επικαλούμενοι το Θεμελιώδες Αξίωμα της Αυτοαναφοράς μπορούν να εκτελεστούν αυτοαναφορικά από τον c_n. Παρατηρούμε, ότι τελικά ο απόγονος θα υπολογιστεί σαν "μετάλλαξη" του c_1. Θα καλούμε και τους δυο παραπάνω υπολογισμούς *κύκλους*. Η μόνη τους διαφορά είναι, πως ενώ ο δεύτερος διατηρεί κανονικά τη μνήμη, ο πρώτος την διαγράφει.

Σε αυτό το σημείο θα εισάγουμε την έννοια της *σύνθετης ακολουθίας* ή *ακολουθίας από κύκλους*, η οποία έχει τη μορφή:

$$(c_1, c_1^1, \ldots, c_1^{n_1}), (c_2, c_2^1, \ldots, c_2^{n_2}), \ldots, (c_k, c_k^1, \ldots, c_k^{n_k}).$$

Μια σύνθετη ακολουθία αποτελείται, λοιπόν, από τους επιμέρους non-proliferating υπολογισμούς, για $i < k$:

$$c_i \mapsto c_i^1 \mapsto c_i^2 \mapsto \ldots \mapsto c_i^{n_i}$$

και ο $c_i^{n_i}$ λαμβάνει την απόφαση να κάνει έναν proliferating υπολογισμό, δίνοντας σαν άμμεσους απογόνους διαφοροποιήσεις του c_i, μια εκ των οποίων θα είναι ο c_{i+1}.
Οι τρεις βασικές υποθέσεις που κάνουμε είναι:

1. Για κάθε $i \leq k$ ο υπολογισμός $c_i \mapsto c_i^1 \mapsto c_i^2 \mapsto \ldots \mapsto c_i^{n_i}$ δεν περιλμβάνει proliferating υπολογιστικά βήματα.

2. Για κάθε $i < k$ το βήμα $c_i^{n_i} \mapsto c_{i+1}$ είναι ένας κύκλος.

3. Για κάθε $i < k$ το αυτοαναφορικό βήμα του $c_i^{n_i}$ είναι proliferating και άρα η $c_1, c_2, ..., c_k$ είναι μια επιζώσα ακολουθία.

Το μέλημά μας, εδώ, είναι να προσωμοιάσουμε τη διαδιακασία της μάθησης, όπως την αντιλαμβανόμαστε σε διάφορες πτυχές της ζωής μας. Μπορούμε να σκεφτόμαστε τον επιμέρους υπολογισμό (non-proliferating)

$$c_i \mapsto c_i^1 \mapsto c_i^2 \mapsto \ldots \mapsto c_i^{n_i}$$

ως μια αλληλουχία από *αβέβαια βήματα* (uncertain steps). Τα καλούμε, έτσι καθώς η διαγωνιοποίηση που συντελείται σε κάθε τέτοιο βήμα δεν είναι αποτέλεσμα φυσικής επιλογής και άρα μια “λανθασμένη απάντηση” δεν είναι καταδικαστική. Υπό μια έννοια, ο αλγόριθμος “δοκιμάζει” διαφορετικές απαντήσεις, λαμβάνοντας κάποιου είδους βαθμολόγηση από το περιβάλλον και διαγωνιοποιώντας κάθε φορά πάνω στην επιτυχημένη υπακολουθία, μέχρι να φτάσει ο $c_i^{n_i}$ να έχει τη σωστή έξοδο για το περιβάλλον. Μπορεί κανείς να φανταστεί την ανατροφοδότηση του περιβάλλοντος κατά τα αβέβαια βήματα περισσότερο σαν μια συμβουλευτική διαδικασία, κάποιου είδους καθοδήγηση, με σκοπό ο αλγόριθμος να προσεγγίσει τη σωστή απάντηση διαγωνιοποιώντας. Τώρα, το βήμα $c_i^{n_i} \mapsto c_{i+1}$ θα καλείται *βέβαιο* (certain), αφού σε αντίθεση με τα αβέβαια, είναι αποτέλεσμα φυσικής επιλογής. Είναι σημαντικό να τονίσουμε, ότι για αυτό το βήμα η διαγωνιοποίηση έχει νόημα να λαμβάνει χώρα τελικά στην επιτυχημένη υπακολουθία $c_0^{n_0}, c_1^{n_1}, ...$, εφόσον αυτοί είναι οι κωδικοί που πήραν θετική ανατροφοδότηση από το περιβάλλον. Τέλος, θα μπορούσαμε να σχολιάσουμε, ότι σε αντίθεση με τα αβέβαια βήματα, όπου μεμονωμένες λάθος απαντήσεις δεν είναι καταδικαστικές, στα βέβαια βήματα που αντιστοιχούν σε συμπεράσματα του αλγορίθμου, η λάθος διαγωνιοποίηση μπορεί να οδηγήσει σε πληθώρα λανθασμένων απαντήσεων και άρα καταδικάζεται από το περιβάλλον (φυσική επιλογή).

Επομένως, με αυτόν τον τρόπο εντοπίσαμε τα απαραίτητα συστατικά ενός *επιτυχημένου υπολογισμού* και έγινε αντιληπτό, ότι ένα αναδρομικό σύστημα ιεράρχισης (σύνθετη ακολουθία,) που χαρακτηρίζει την επιτυχία, θα πρέπει να έχει ως βάση την ικανότητα για επιβίωση.

3.2.4 Περαιτέρω δυνατότητες της διαγωνιοποίησης

Θα θέλαμε σε αυτό το σημείο να κάνουμε κάποιες παρατηρήσεις σχετικά με τις δυνατότητες της διαγωνιοποίησης:

1. Μπορεί να ελέγχει τις ίδιες της τις παραμέτρους.

    (a) Ένα είδος παραμέτρου της διαγωνιοποίησης είναι η προτεραιότητα των κωδικών στην αλληλουχία που προτείνει ο searcher. Μέχρι στιγμής αρκεστήκαμε να αναφέρουμε, ότι οι κωδικοί θα πρέπει να έχουν κάποια προτεραιότητα ως προς την σειρά με την οποία προτείνονται, γεγονός που μοιάζει διαισθητικά κατανοητό. Ωστόσο, αξίζει να προσθέσουμε, ότι μέσω της διαγωνιοποίησης η προτεραιότητα των κωδικών μπορεί να ρυθμίζεται και αυτή με βάση την εξέλιξη. Αυτό, διότι αν $B(r)$ είναι ένας αλγόριθμος που αλλάζει με κάποιο τρόπο την προτεραιότητα ενός κωδικού r σε κάποιον searcher δ_s, τότε από το Θεμελιώδες Θεώρημα της Αυτοαναφοράς, ο $B(r)$ μπορεί να υπάρχει σαν εντολή μέσα στον αλγόριθμο που χρησιμοποιεί τη διαγωνιοποίηση και άρα να υπόκειται σε αυτή. Επίσης, δεδομένου, ότι η προτεραιότητα ενός r βρίσκεται σε κάποια διεύθυνση του αυτοαναφορικού αλγορίθμου, η διαγωνιοποίηση μπορεί να εντοπίσει κάποιο μοτίβο σε αυτή κοιτάζοντας την ιστορία του αυτοαναφορικού κωδικού. Ας δούμε ένα αντιπροσωπευτικό παράδειγμα:
    Έστω θ η διεύθυνση με περιεχόμενο την προτεραιότητα w ενός κωδικού και έστω

    $$c[(\theta)w], c[(\theta)w+1], c[(\theta)w+2]$$

    μια ακολουθία που έχει διαπιστωθεί ως επιτυχημένη από τον c. Τότε, ο c διαγωνιοποιώντας πάνω σε αυτήν μπορεί να καταλήξει στο ότι κατά το επόμενο υπολογιστικό βήμα η προτεραιότητα θα πρέπει να συνεχίσει να αυξάνεται.

(b) Άλλο είδος παραμέτρου αποτελεί η παράμετρος που ελέγχει το μήκος k_d της ακολουθίας στο οποίο θα πρέπει ο προτεινόμενος κωδικός r να ταιριάζει για να ληφθεί η απόφαση από τον κωδικό d να εφαρμοστεί. Κάτι τέτοιο κρίνεται απαραίτητο, καθώς σε διαφορετική περίπτωση η πιθανότητα να εφαρμοστεί μια σωστή απόφαση $D(r)$ είναι αμελητέα. Μπορούμε σε αυτό το σημείο να κάνουμε άλλο ένα βήμα στη σκέψη μας λέγοντας, ότι ο αλγόριθμος

> Ενεργοποίησε το $d[r]$ οποτεδήποτε ο προτεινόμενος κωδικός r ταιριάζει τουλάχιστον σε k_d βήματα της πρόσφατης μνήμης.

θα έπρεπε ο ίδιος να ταιριάζει στην ακολουθία $c_1, c_2, ..., c_n$, έτσι ώστε, όντας αρκετά απλός, να μπορεί να προταθεί από την διαγωνιοποίηση.

2. Υποθέτοντας, ότι ο αυτοαναφορικός αλγόριθμος c περιέχει μιας μορφής αναπαράσταση του περιβάλλοντος, η διαγωνιοποίηση μπορει να οδηγήσει στην πρόβλεψη ομαλών περιβαλλοντικών αλλαγών (smooth environmental changes). Δεν χρειάζεται, επομένως, οι συνθήκες που επιβάλλει το περιβάλλον να είναι διαρκώς ίδιες. Ο Δ μπορεί με όμοιο τρόπο με αυτόν που περιγράψαμε πιο πάνω να λειτουργήσει σε ελαφρώς μεταβαλλόμενες συνθήκες περιβάλλοντος.

3. Ο tester περιγράφηκε ως ένας αλγόριθμος T, ο οποίος ελέγχει αν ένας προτεινόμενος κωδικός ταιριάζει στην ακολουθία ελέγχου ή όχι απαντώντας με true ή false. Θα πρέπει να σημειωθεί, ότι ενδέχεται να εμφανίζονται παραλλαγές του αλγορίθμου ελέγχου. Παραδείγματος χάρην, αντί για true/false μπορεί να δίνει έξοδο κάποια τιμή, με την οποία θα αξιολογεί το πόσο καλά ταιριάζει ο προτεινόμενος αλγόριθμος r στην ακολουθία κωδικών.

4. Μπορούμε να σκεφτούμε την περίπτωση, όπου κάποια απαίτηση του περιβάλλοντος έχει τοπικό χαρακτήρα. Θα περιμέναμε, μετά από όσα έχουμε πει, η διαγωνιοποίηση να μπορεί να αντιληφθεί κάτι τέτοιο. Πράγματι, ο τοπικός χαρακτήρας μπορεί να φανεί από το μήκος της ακολουθίας της πρόσφατης ιστορίας του c. Έτσι, η διαγωνιοποίηση μπορεί να ακολουθήσει μια πιο συντηρητική προσέγγιση, εφαρμόζοντας τον κωδικό που ταιριάζει σε κάποιο ποσοστό (ανάλογο του μήκους της ακολουθίας) των απογόνων του, αντί για το σύνολό τους.

5. Ακολουθώντας ακριβώς το ίδιο σκεπτικό, αν ένας κωδικός εντοπίζεται σε όλη την ακολουθία προγόνων, η διαγωνιοποίηση θα μπορούσε να αποφασίσει την εφαρμογή του στο σύνολο των απογόνων του. Ένα ταιριαστό παράδειγμα είναι το εξής: αν η διαγωνιοποίηση εντοπίσει κομμάτια του κωδικού που παραμένουν ίδια για μεγάλο χρονικό διάστημα (μήκος ακολοθίας απογόνων), να αποφασίσει να τα κληροδοτήσει σε όλους τους απογόνους του c.

Παρατήρηση. *Πέρα από τις δυνατότητες της διαγωνιοποίησης που τονίσαμε, θα θέλαμε να εστιάσουμε σε ένα ζήτημα, το οποίο κατά πάσα πιθανότητα έχει ήδη απασχολήσει τον αναγνώστη: το ερώτημα κατά πόσο ένας προτεινόμενος κωδικός r ταιριάζει στην ακολουθία ελέγχου είναι στην πραγματικότητα μη αποφασίσιμο (βλ. Πρόβλημα Τερματισμού). Ωστόσο, η διαγωνιοποίηση μπορεί να ξεπεράσει αυτό το πρόβλημα καθορίζοντας ένα άνω όριο για τα επιτρεπόμενα υπολογιστικά βήματα του υπολογισμού $alg(r)(c_k) \sqsubseteq c_{k+1}$. Κάτι τέτοιο, μοιάζει* *και λογικό, αφού αν για κάποιον απόγονο η παραπάνω διαδικασία δεν τερμάτιζε ποτέ, αυτό θα οδηγούσε στη μη επιβίωσή του.*

Έχοντας δει αρκετά στοιχεία, ας περιγράψουμε τώρα με έναν πιο τεχνικό και λεπτομερή τρόπο τον μηχανισμό της διαγωνιοποίησης σε έναν αλγόριθμο c. Το βασικό ζήτημα εδώ είναι να αναλύσουμε, καταρχάς, πως η διαγωνιοποίηση θα εφαρμόσει την απόφαση να χρησιμοποιήσει κάποιον προτεινόμενο r χρησιμοποιώντας τον κωδικό απόφασης $s[r]$. Όπως αναφέρθηκε και σε προηγούμενες παραγράφους, η απόφαση αυτή στηρίζεται σε έναν αλγόριθμο ελέγχου T, του οποίου τις προϋποθέσεις θα πρέπει να πληροί ο r. Θα δουλέψουμε πάνω σε μια σύνθετη ακολουθία:

$$(c_1, c_1^1, \ldots, c_1^{n_1}), (c_2, c_2^1, \ldots, c_2^{n_2}), \ldots, (c_k, c_k^1, \ldots, c_k^{n_k}). \tag{13}$$

Έστω, λοιπόν ο κωδικός $t[\varnothing, \varnothing]$ του αλγορίθμου ελέγχου και ο κωδικός $s[\varnothing]$ της απόφασης. Ο κωδικός t τονίζουμε, ότι έχει δυο κενές διευθύνσεις: η πρώτη θα χρησιμοποιηθεί για τον κωδικό r (όπως και στον s), ενώ η δεύτερη θα περιέχει την συνθήκη, στην οποία θα βασιστεί ο έλεγχος. Αυτή η συνθήκη μπορεί να είναι είτε εσωτερική, είτε εξωτερική. Και στις δυο περιπτώσεις, ωστόσο, θεωρούμε, ότι υπάρχει κωδικοποιημένη στον c και άρα η δεύτερη κενή διεύθυνση θα καταληφθεί από τον c.

Όπως έχουμε τονίσει σε όλη την ανάλυσή μας, η επιβίωση των αυτοαναφορικών αλγορίθμων στηρίζεται στη διαγωνιοποίηση και τις αποφάσεις που αυτή λαμβάνει. Επομένως, αν για έναν προτεινόμενο κωδικό r, ο αλγόριθμος ελέγχου δίνει έξοδο true, τότε θα πρέπει να εφαρμοστεί ο r (είτε συντηρητικά, είτε καθολικά με βάση τις παρατηρήσεις που κάναμε πιο πάνω) στο επόμενο υπολογιστικό βήμα (proliferating ή μη). Δηλαδή αν στην (13) για κάποιο $i \leq k$ και $j < n_i$, $\mathrm{alg}(t)(r, c_i^j) = \mathrm{true}$, τότε ο $\mathrm{alg}(s)(r)$ θα πρέπει να ταιριάζει στο υπολογιστικό βήμα $c_i^j \mapsto c_i^{j+1}$. Σίγουρα, στην αρχή υποθέτουμε, ότι κάτι τέτοιο θα συμβεί τυχαία, αλλά τελικά, ο κωδικός m για τον αλγόριθμο M:

Αν για κάποιο r, $\mathrm{alg}(t)(r, c) = \mathrm{true}$, τότε πάρε την απόφαση $\mathrm{alg}(s)(r)$

θα ταιριάζει στο υπολογιστικό βήμα $c_i^j \mapsto c_i^{j+1}$ για κάθε $i \leq k$ και $j < n_i$ και λόγω διαγωνιοποίησης θα μπορεί να διαπιστωθεί σε κάθε υπολογιστικό βήμα απο εκεί και έπειτα. Ουσιαστικά, λοιπόν, ανεβαίνουμε ένα επίπεδο, διότι κατά αυτόν τον τρόπο η διαγωνιοποίηση ελέγχει και δικές της συμπεριφορές στην επιζώσα ή επιτυχημένη ακολουθία. Απλούστερα, ο ίδιος ο κωδικός της απόφασης να εφαρμοστεί κάποιο r θα αποτελεί προτεινόμενο από τον searcher κωδικό και εφόσον θα ταιριάζει στην ακολουθία (για τους λόγους που εξηγήσαμε), η διαγωνιοποίηση θα αποφασίζει να τον εφαρμόζει.

Τέλος, θα αναφερθούμε στην ικανότητα της διαγωνιοποίησης να εξειδικεύεται. Ας υποθέσουμε την εξής κατάσταση: έστω c αυτοαναφορικός αλγόριθμος, όπου

$$c = c[\delta[t'], (\theta)\varnothing, (+\phi)\varnothing].$$

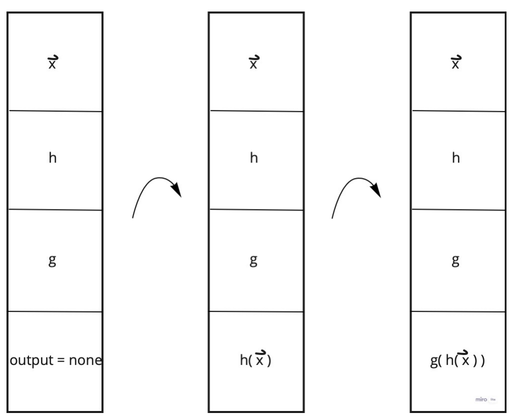
Δηλαδή ο c περιέχει έναν αλγόριθμο απόφασης με κωδικό $\delta[t'] = [\delta_s, \delta_t[t']]$ (όπου t' ο κωδικός του αλγορίθμου ελέγχου) και δυο κενές διευθύνσεις: μια προϋπάρχουσα (θ) και μια που προσθέτουμε (ϕ). Αν ο c αντιγράψει τον $\delta[t] = [\delta_s, \delta_t[t]]$ στην διεύθυνση ϕ, τότε ο searcher δ_s θα έχει τη δυνατότητα να εξειδικευθεί στην αναζήτηση θ-differentiating κωδικών. Προσέχουμε, ότι ο tester t' έχει αντικατασταθεί από τον t.

4 Αυτοαναφορικοί αλγόριθμοι και αναδρομικές συναρτήσεις

Σε αυτήν την ενότητα αποδεικνύουμε, ότι οι αυτοαναφορικοί αλγόριθμοι είναι κλειστοί ως προς τη σύνθεση, το σχήμα της πρωτογενούς αναδρομής και -με την προσθήκη των αλγορίθμων ελέγχου- ως προς τον τελεστή ελαχιστοποίησης. Άρα, λαμβάνοντας υπόψη την θέση των Hauser et al., ένας αυτοαναφορικός αλγόριθμος πρέπει να μπορεί να κατακτήσει μια γλώσσα, με τον τρόπο που η θεωρία της αναδρομικότητας της FLN εντείνεται, ότι το καταφέρνει (από καθαρά υπολογιστική άποψη) ένας άνθρωπος. Θα σκιαγραφήσουμε την πορεία της απόδειξης.

Έστω c αυτοαναφορικός αλγόριθμος. Υποθέτουμε, ότι ο c περιέχει τη συνάρτηση του επομένου $S(n) = n + 1$ σαν εντολή. Επίσης, ότι έχει τη δυνατότητα να διαπερνά τις διευθύνσεις του, το οποίο μπορεί υπό μια έννοια να μεταφραστεί ως ότι περιέχει την συνάρτηση της προβολής. Ακόμη, δύναται να δέχεται σαν είσοδο το περιεχόμενο μιας διεύθυνσής του, να γράφει σαν έξοδο σε κάποια διεύθυνσή του και τέλος, να ενεργοποιήσει κάποια εντολή που περιέχεται σε κάποια διεύθυνσή του.

Κλειστότητα ως προς τη σύνθεση. Έστω g και h συναρτήσεις που περιέχονται στις διευθύνσεις θ_g και θ_h του c, αντίστοιχα και έστω οι διευθύνσεις θ_x , όπου ο c έχει αποθηκευμένη την είσοδο $\vec{x}$ και θ_o, όπου ο c αποθηκεύει τα αποτελέσματα από τους υπολογισμούς των g και h. Αν ο c υπολογίσει την h για την τιμή $\vec{x}$, θα αποθηκεύσει το αποτέλεσμα στην διεύθυνση θ_o. Έπειτα, η g μπορεί να υπολογιστεί λαμβάνοντας σαν είσοδο $h(\vec{x})$ από την θ_o και το αποτέλεσμα (δηλαδή η σύνθεση $g(h(\vec{x}))$ να αντικαταστήσει τελικά το προηγούμενο περιεχόμενο της θ_o. (Φυσικά, λαμβάνουμε τις g και h να είναι συμβατές για τη σύνθεση.)



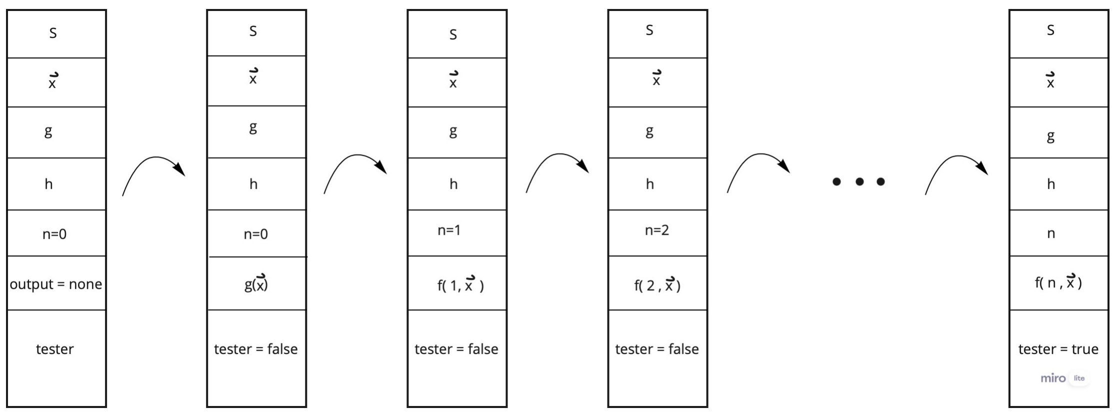
Σχήμα 2: Υπολογισμός σύνθεσης συναρτήσεων από έναν αυτοαναφορικό αλγόριθμο.

Κλειστότητα ως προς την πρωτογενή αναδρομή. Έστω, τώρα, ότι ο c περιέχει επιπρόσθετα μια διεύθυνση, όπου αποθηκεύονται οι τιμές μιας παραμέτρου $n \in \mathbb{N}$, μια διεύθυνση για την είσοδο $\vec{x}$ και δυο διευθύνσεις για τις συναρτήσεις g και h, με βάση τις οποίες ορίζεται η f ως εξής:

$$f(0, \vec{x}) = g(\vec{x})$$
$$f(n+1, \vec{x}) = h(f(n, \vec{x}), n, \vec{x}).$$

Θα δείξουμε, ότι ο c μπορεί να υπολογίσει την f. Φανταζόμαστε σε μια αρχική κατάστασή του ο c να περιέχει σε αντίστοιχες διευθύνσεις τα: $S, \vec{x}, g, h, n$. Επίσης, θα περιέχει μια διεύθυνση θ_o, όπου θα αποθηκεύει τις τιμές της f που θα υπολογίζει. Τέλος, θα πρέπει να συμμετέχει και ο tester, ο οποίος θα δίνει έξοδο false, έως ότου ο c να φτάσει να υπολογίσει την τιμή της f για το επιθυμητό n, όπου και θα απαντήσει true για να σταματήσει ο υπολογισμός.

Στο πρώτο βήμα με $n = 0$ η g με είσοδο $\vec{x}$, υπολογίζει το $g(\vec{x}) = f(0, \vec{x})$ και το αποθηκεύει στη διεύθυνση θ_o. Στο επόμενο βήμα, λειτουργούν δύο εντολές: η h, λαμβάνοντας σαν είσοδο το $f(0, \vec{x})$, το $n = 0$ και το $\vec{x}$, υπολογίζει το $f(1, \vec{x})$ και το αποθηκεύει στην θ_o και η S αυξάνει το n κατά ένα. Αξίζει σε αυτό το σημείο να σχολιάσουμε, ότι κάτι τέτοιο είναι εφικτό, καθώς σε κάθε στάδιο ο υπολογισμός θα "βλέπει" την προηγούμενη εκδοχή του και θα "γράφει" την καινούρια. Το σκεπτικό της απόδειξης μπορεί να φανεί στο Σχήμα 3:



Σχήμα 3: Υπολογισμός μιας πρωτογενώς αναδρομικής συνάρτησης από έναν αυτοαναφορικό αλγόριθμο.

Κλειστότητα ως προς την ελαχιστοποίηση. Έστω f η ελαχιστοποίηση της μερικής συνάρτησης g. Δηλαδή, έχουμε $f(y, \vec{x}) = (\mu i \geq y)[g(i, \vec{x}) = 0]$. Ένας αυτοαναφορικός αλγόριθμος μπορεί να υπολογίσει την f για κάποιο ζεύγος $(y, \vec{x})$ αξιοποιώντας τους searcher και tester της διαγωνιοποίησης. Συγκεκριμένα, ο searcher προτείνει φυσικούς αριθμούς (οι οποίοι αντιστοιχούν σε κωδικούς) ξεκινώντας απ' το y και ο tester θα πρέπει να βρει τον ελάχιστο κωδικό i που ικανοποιεί τη ζητούμενη συνθήκη $g(i, \vec{x}) = 0$.

5 Γλώσσα και αυτοαναφορά: ένα παράδειγμα

Τέλος, βασιζόμενοι στη θεώρηση της UG, θα προσπαθήσουμε να δώσουμε ένα παράδειγμα για το πως η κατάκτηση της γλώσσας μπορεί να προσομοιωθεί από έναν αυτοαναφορικό αλγόριθμο.

Καταρχάς, ο c θα πρέπει να περιλαμβάνει εκ κατασκευής ακριβώς το σύνολο των κανόνων που αποδίδονται στην UG, εφόσον θεωρούμε, ότι αυτοί είναι γενετικά προκαθορισμένοι. Επομένως, ο c θα περιλαμβάνει κάποια εντολή g η οποία στις διευθύνσεις της θα έχει αποθηκευμένους τους κανόνες που αντιστοιχούν στην UG (τις Αρχές και τις -αρρύθμιστες ακόμη- Παραμέτρους). Ταυτόχρονα, θα πρέπει να περιλαμβάνει δυο ακόμη εντολές που θα αντιστοιχούν στα συστήματα sensory-motor και conceptual-intentional. Το sensory-motor μπορεί να αναφέρεται στην διεύθυνση της περιβαλλοντικής εισόδου, απ' όπου ο c θα μπορεί να λαμβάνει αποσπασματικά δεδομένα, όπως λέξεις, φράσεις ή και ολόκληρες προτάσεις, τις οποίες θα επεξεργάζεται για να απαντήσει κατάλληλα στο περιβάλλον. Το conceptual-intentional σύστημα θα αντιστοιχεί σε κάποια εντολή υπεύθυνη για το νόημα και την ερμηνεία, δηλαδή με κατάλληλες διαδικασίες (βλ. παράλληλη διαγωνιοποίηση) ο c θα είναι σε θέση να ανακαλύπτει την κοινή φύση των γλωσσικών ερεθισμάτων, δημιουργώντας τις αντίστοιχες αφαιρετικές έννοιες. Επίσης, θεωρούμε, ότι το περιβάλλον θα διαθέτει κάποια μορφή επιβράβευσης, όπως ακριβώς συμβαίνει και με τα παιδιά κατά τη διαδικασία της γλωσσικής κατάκτησης. Φυσικά, στην αρχή εναποθέτουμε τις ελπίδες μας στην τύχη, που σημαίνει, ότι περιμένουμε πως τυχαία κάποια στιγμή ο c θα δώσει μια ικανοποιητική απάντηση. Από εκεί και πέρα, σειρά έχει η διαγωνιοποίηση, με βάση την οποία οι σωστές απαντήσεις θα λαμβάνονται «συνειδητά».

Η πρότασή μας είναι, πως μια τέτοια διαδικασία μπορεί να προσομοιωθεί με τη χρήση σύνθετων ακολουθιών που αξιοποιούν τους κύκλους (cycles). Κατά τα αβέβαια βήματα, ο c θα μπορεί να «δοκιμάζει» απαντήσεις και να τις βαθμολογεί ανάλογα με την επιβράβευση του περιβάλλοντος. Με αυτή τη διαδικασία και παίρνοντας τις κατάλληλες ανατροφοδοτήσεις, θα εμπλουτίζει το λεξικό του και θα ρυθμίζει τις παραμέτρους του, ακριβώς όπως περιγράφει η Θεωρία Αρχών και Παραμέτρων. Κάθε φορά που ο c θα προβαίνει σε ένα βέβαιο βήμα, σημαίνει πως θα έχει λάβει τη σωστή απόφαση και άρα έχει απαντήσει ικανοποιητικά στο περιβάλλον. Συνεχίζοντας την τροφοδότηση με γλωσσικά δεδομένα από το περιβάλλον, ο c θα προχωράει στα αβέβαια βήματα του επόμενου «σταδίου», όπου θα συνεχίσει να προσαρμόζει τις παραμέτρους και να κάνει χρήση των αρχών, μέχρι το επόμενο βέβαιο βήμα. Έτσι, περιμένουμε μετά από κατάλληλο αριθμό τέτοιων κύκλων να καταφέρει να συγκροτήσει τη γραμματική της γλώσσας, στην οποία τον έχουμε εκθέσει.

6 Επίλογος

Συνοψίζοντας, παρακολουθήσαμε την πορεία της έννοιας της αναδρομής μέσα στα μαθηματικά και αντιληφθήκαμε την εξέλιξή της ως τυποποίηση της υπολογισιμότητας. Στη συνέχεια, μελετήσαμε την επιρροή που άσκησε η Θεωρία της Υπολογισιμότητας στον κλάδο της Γλωσσολογίας, όπου το αντικείμενο μελέτης μετατοπίστηκε στον γενετικά κωδικοποιημένο εσωτερικό μηχανισμό (FLN) που οφείλεται για τη γλώσσα. Εργαλείο των γλωσσολόγων για την μελέτη αυτού του μηχανισμού είναι η γενετική γραμματική, οι κανόνες της οποίας έχουν αναπτυχθεί κατ' αντίστοιχα με τους τύπους της λογικής. Η πρόταση, λοιπόν, ότι η FLN βασίζεται σε αναδρομικές διαδικασίες απορρέει ακριβώς από τον αναδρομικό ορισμό των τύπων στη λογική. Παράλληλα, παρουσιάσαμε το υπολογιστικό σύστημα των αυτοαναφορικών αλγορίθμων και αποδείξαμε, ότι είναι ικανό να υπολογίσει την κλάση των υπολογίσιμων συναρτήσεων. Σε αυτό το πλαίσιο, προτείναμε, ότι θα πρέπει να είναι ικανό να προσομοιώσει την γλωσσική κατάκτηση με τον τρόπο που περιγράφει η Γενετική Γλωσσολογία.

Θα θέλαμε σε αυτό το σημείο να τονίσουμε, ότι η παραπάνω μελέτη έγινε σε ένα καθαρά θεωρητικό πλαίσιο. Περαιτέρω ενασχόληση με το αντικείμενο θα μπορούσε να αφορά, καταρχάς, σε προσπάθειες ανάπτυξης του συστήματος των αυτοαναφορικών αλγορίθμων σε περιβάλλον υπολογιστή. Έχοντας αυτή τη βάση και λαμβάνοντας υπόψη τις εξελίξεις στον τομέα της Γλωσσολογίας, θα είχε ενδιαφέρον να αποπειραθεί κανείς την προσομοίωση της γλωσσικής κατάκτησης σε ένα πρακτικό επίπεδο εφαρμογής.

Βιβλιογραφία

[1] Arvanitakis A.D. *Recursion, Evolution and Conscious Self.* May 2021.

[2] Wilhelm Ackermann. "Zum Hilbertschen Aufbau der reellen Zahlen". In: *Mathematische Annalen* 99 (1928), pp. 118–133.

[3] Rudolf Berghammer. "A Relation-Algebraic Treatment of the Dedekind Recursion Theorem". In: *Relational and Algebraic Methods in Computer Science*. Ed. by Uli Fahrenberg, Peter Jipsen, and Michael Winter. Cham: Springer International Publishing, 2020, pp. 15–30. ISBN: 978-3-030-43520-2. DOI: 10.1007/978-3-030-43520-2_2.

[4] Paul Bernays. "Hilberts Untersuchungen über die Grundlagen der Arithmetik". In: *Dritter Band: Analysis · Grundlagen der Mathematik · Physik Verschiedenes: Nebst Einer Lebensgeschichte*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1935, pp. 196–216. ISBN: 978-3-662-38452-7. DOI: 10.1007/978-3-662-38452-7_14.

[5] Leonard Bloomfield. "A Set of Postulates for the Science of Language". In: *Language* 2.3 (1926), pp. 153–164. DOI: 10.2307/408741.

[6] Noam Chomsky. *Aspects of the Theory of Syntax*. 50th ed. The MIT Press, 1965.

[7] Noam Chomsky. "LECTURE I: WHAT IS LANGUAGE?" In: *The Journal of Philosophy* 110.12 (2013), pp. 645–662. ISSN: 0022362X.

[8] Noam Chomsky. "Of Minds and Language". In: *Biolinguistics* 1 (2007), pp. 009–027.

[9] Noam Chomsky. *Rules and Representations*. Columbia University Press, 1980, pp. 217–227. ISBN: 9780231132718.

[10] Noam Chomsky. "Three Models for the Description of Language". In: *IRE Transactions on Information Theory* 2.3 (1956), pp. 113–124.

[11] Noam Chomsky and Howard Lasnik. "The Theory of Principles and Parameters". In: *1. Halbband: An International Handbook of Contemporary Research*. Ed. by Joachim Jacobs, Arnim von Stechow, Wolfgang Sternefeld, and Theo Vennemann. De Gruyter Mouton, 2008, pp. 506–569. DOI: doi:10.1515/9783110095869.1.9.506.

[12] Noam Chomsky and George A. Miller. "Introduction to the Formal Analysis of Natural Languages". In: 1968.

[13] Noam Chomsky and Neil Smith. *New Horizons in the Study of Language and Mind.* Cambridge University Press, 2000. DOI: 10.1017/CBO9780511811937.

[14] Alonzo Church. "A Note on the Entscheidungsproblem". In: *Journal of Symbolic Logic* 1.1 (1936), pp. 40–41. DOI: 10.2307/2269326.

[15] Alonzo Church. "A set of postulates for the foundation of logic". In: *Annals of Mathematics Studies* 33.2 (1932), pp. 346–366.

[16] Alonzo Church. "A set of postulates for the foundation of logic (second paper)". In: *Annals of Mathematics Studies* 34.2 (1933), pp. 839–864.

[17] Alonzo Church. "An Unsolvable Problem of Elementary Number Theory". In: *American Journal of Mathematics* 58.2 (Apr. 1936), pp. 345–363.

[18] Martin Davis. *Computability and Unsolvability*. New York: McGraw-Hill, 1958.

[19] Richard Dedekind. *Was sind und was sollen die Zahlen?* Braunschweig: Vieweg, 1888.

[20] Gilles Dowek. *Οι μεταμορφώσεις του λογισμού*. Trans. by Τεύκρος Μιχαηλίδης. Αθήνα: ΕΚΚΡΕΜΕΣ, 2012.

[21] V. Fromkin, R. Rodman, and N. Hyams. *2008*. Trans. by Γ. Ξυδόπουλος. Αθήνα: Πατάκης.

[22] K. Gödel, S. Feferman, J.W. Dawson, S.C. Kleene, G. Moore, R. Solovay, and J. van Heijenoort. *Kurt Gödel: Collected Works: Volume I: Publications 1929-1936*. Collected Works of Kurt Godel. OUP USA, 1986. ISBN: 9780195039641.

[23] Kurt Gödel. "Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I". In: *Monatshefte für Mathematik und Physik* 38 (1931), pp. 173–198.

[24] Marc D. Hauser, Noam Chomsky, and W. Tecumseh Fitch. "The Faculty of Language: What Is It, Who Has It, and How Did It Evolve?" In: *Science* 298.5598 (2002), pp. 1569–1579. DOI: 10.1126/science.298.5598.1569.

[25] David Hilber. "Axiomatisches Denken". In: *Mathematische Annalen* 78 (), pp. 405–415.

[26] David Hilbert. *Grundlagen der Geometrie*. Leipzig: Teubner, 1899.

[27] David Hilbert and Wilhelm Ackermann. *Grundzüge der theoretischen Logik (Principles of Mathematical Logic)*. Springer-Verlag, 1928. ISBN: 0-8218-2024-9.

[28] Otto Jespersen. *The Philosophy of Grammar*. New York: Henry Holt and Company, 1924.

[29] Stephen C. Kleene. "Origins of Recursive Function Theory". In: *Annals of the History of Computing* 3.1 (Jan. 1981). DOI: 10.1109/MAHC.1981.10004.

[30] Stephen C. Kleene. "Recursive predicates and quantifiers". In: *Transactions of the American Mathematical Society* 53 (1943), pp. 41–73.

[31] Stephen C. Kleene. "The Theory of Recursive Functions, Approaching its Centennial". In: *Bulletin (New Series) of the American Mathematical Society* 5.1 (July 1981), pp. 43–61.

[32] Ernest Paul. "Mathematical Induction: A Recurring Theme". In: *The Mathematical Gazette* 66.436 (1982), pp. 120–125. ISSN: 00255572.

[33] Rózsa Péter. "Konstruktion nichtrekursiver Funktionen". In: *Mathematische Annalen* 111 (1935), pp. 42–60.

[34] Rózsa Péter. "Rekursive Funktionen". In: *Verhandlungen des Internationalen Mathematiker-Kongresses Zürich*. Vol. 2. 1932, pp. 336–337.

[35] Rózsa Péter. *Rekursive Funktionen*. Akademiai Kiado', 1951.

[36] Rózsa Péter. "Über den Zusammenhang der verschiedenen Begriffe der rekursiven Funktion". In: *Mathematische Annalen* 110 (1934), pp. 612–632.

[37] Rózsa Péter. “Über die mehrfache Rekursion”. In: *Mathematische Annalen* 113 (1936), pp. 489–527.

[38] Emil L Post. “Formal reductions of the general combinatorial decision problem”. In: *American Journal of Mathematics* 65.2 (1943), pp. 197–215.

[39] Ferdinand de Saussure. *Μαθήματα Γενικής Γλωσσολογίας*. Trans. by Φ. Δ. Αποστολόπουλος. Athens: Παπαζήση, 1979.

[40] D.E. Smith. *A sourcebook in mathematics*. Vol. 1. Dover, 1959.

[41] Raymond M. Smullyan. “Theory of formal systems”. In: *Annals of Mathematics Studies* 47 (1961).

[42] Robert Irving Soare. “Computability and Recursion”. In: *Bulletin of Symbolic Logic* 2 (1996), pp. 284–321.

[43] Rüdiger Thiele. “Hilbert’s Twenty-Fourth Problem”. In: *The American Mathematical Monthly* 110.1 (2003), pp. 1–24. DOI: 10.1080/00029890.2003.11919933.

[44] A. M. Turing. “I.—COMPUTING MACHINERY AND INTELLIGENCE”. In: *Mind* LIX.236 (Oct. 1950), pp. 433–460.

[45] A. M. Turing. “On Computable Numbers, with an Application to the Entscheidungsproblem”. In: *Proceedings of the London Mathematical Society* s2-42.1 (Jan. 1937), pp. 230–265. ISSN: 0024-6115. DOI: 10.1112/plms/s2-42.1.230.

[46] Jeffrey Watumull, Marc D. Hauser, Ian G. Roberts, and Norbert Hornstein. “On recursion”. In: *Frontiers in Psychology* 4 (Jan. 2014). ISSN: 1664-1078. DOI: 10.3389/fpsyg.2013.01017.

[47] Έλενα Αναγνωστοπούλου. *Σημειώσεις Σύνταξη Ι-ΓΛΩΦ 165*. 2013.

[48] Γ. Κολέτσος. *Μαθηματική Λογική*. 2015.

[49] Γιάννης Ν. Μοσχοβάκης. *Αναδρομή και Υπολογισιμότητα*. June 2012.

[50] Νικήτας Σταθάτος. “Αναδρομή και εξέλιξη συστημάτων τεχνητής νοημοσύνης”. Διπλωματική Εργασία. Ε.Μ.Π.

[51] Χ. Χαραλαμπάκης. *Νεοελληνικός λόγος: μελέτες για τη γλώσσα, τη λογοτεχνία και το ύφος*. Αθήνα: Νεφέλη, 1992. ISBN: 9789602111338.