

Μηχανές Turing και Υπολογισιμότητα

Διδάσκοντες: **Σ. Ζάχος, Δ. Φωτάκης**
Επιμέλεια διαφανειών: **Δ. Φωτάκης**

Σχολή Ηλεκτρολόγων Μηχανικών
και Μηχανικών Υπολογιστών

Εθνικό Μετσόβιο Πολυτεχνείο



Θεωρία Υπολογισμού

- Γιατί κάποια προβλήματα **δεν λύνονται** από υπολογιστές;
- **Hilbert** (1900): **πληρότητα** και **αυτοματοποίηση** των μαθηματικών.
 - 10^ο πρόβλημα: Αλγόριθμος για **Διοφαντικές εξισώσεις**.
- Αλγόριθμος: **διατύπωση** και απόδειξη ορθότητας.
- Δεν υπάρχει αλγόριθμος;
 - **Ορισμός** «αλγόριθμου» μέσω **υπολογιστικού μοντέλου**, και απόδειξη ότι **ύπαρξη αλγόριθμου οδηγεί σε αντίφαση**.
- **Gödel**: μαθηματική **δεν είναι πλήρη!**
- **Turing**: μαθηματικά **δεν αυτοματοποιούνται!**
 - Υπάρχουν προβλήματα που δεν είναι υπολογίσιμα.
- **Matijasevic** (1970): Όχι αλγόριθμος για Διοφαντικές εξισώσεις.
 - Για κάθε αλγ. A , υπάρχει εξίσωση που ο A απαντά λάθος!

Υπολογιστικό Πρόβλημα και Αλγόριθμος

- (Υπολογιστικό) πρόβλημα: ορίζει μετασχηματισμό δεδομένων εισόδου σε δεδομένα εξόδου.
 - Διαισθητικά: ορίζεται από **ερώτηση** για **στιγμιότυπα εισόδου**.
- **Στιγμιότυπο**: αντικείμενο που αντιστοιχεί σε δεδομένα εισόδου.
 - Διατυπώνουμε **ερώτηση** και περιμένουμε **απάντηση**.
 - Άπειρο σύνολο στιγμιότυπων.
- Αλγόριθμος: **σαφώς** ορισμένη διαδικασία για την **επίλυση** προβλήματος σε **πεπερασμένο** χρόνο από υπολογιστική **μηχανή** (Turing).
 - Υπολογίζει **μηχανιστικά** την **σωστή απάντηση** σε **πεπερασμένο χρόνο**.

Προβλήματα Βελτιστοποίησης

- Πρόβλημα βελτιστοποίησης Π :
 - Σύνολο στιγμιότυπων Σ_{Π}
 - Σύνολο αποδεκτών λύσεων: $\forall \sigma \in \Sigma_{\Pi}, \Lambda_{\Pi}(\sigma)$
 - Αντικειμενική συνάρτηση: $\forall \sigma \in \Sigma_{\Pi}, f_{\sigma} : \Lambda_{\Pi}(\sigma) \mapsto \mathbb{R}$
- Δεδομένου στιγμιότυπου σ , ζητείται $\lambda_{\sigma}^* \in \Lambda_{\Pi}(\sigma)$:
 - $\forall \lambda \in \Lambda_{\Pi}(\sigma), f_{\sigma}(\lambda_{\sigma}^*) \geq f_{\sigma}(\lambda)$ πρόβλημα **μεγιστοποίησης**
 - $\forall \lambda \in \Lambda_{\Pi}(\sigma), f_{\sigma}(\lambda_{\sigma}^*) \leq f_{\sigma}(\lambda)$ πρόβλημα **ελαχιστοποίησης**
 - λ_{σ}^* βέλτιστη λύση και $f_{\sigma}(\lambda_{\sigma}^*)$ βέλτιστη αντικειμενική τιμή**

Προβλήματα Απόφασης

- Πρόβλημα απόφασης Π :
 - Σύνολο στιγμιότυπων Σ_{Π}
 - Σύνολο (αποδεκτών) λύσεων: $\forall \sigma \in \Sigma_{\Pi}, \Lambda_{\Pi}(\sigma)$
 - Δεδομένου $\sigma \in \Sigma_{\Pi}, \Lambda_{\Pi}(\sigma) \neq \emptyset$;
- Επιδέχεται μόνο δύο απαντήσεις: **ΝΑΙ ή ΌΧΙ.**

Παραδείγματα Προβλημάτων

- **Πρόβλημα Προσπελασιμότητας:**
 - **Στιγμιότυπο:** Κατευθυνόμενο γράφημα $G(V, E)$, κορυφές $s, t \in V$.
 - **Ερώτηση:** Υπάρχει $s - t$ μονοπάτι;

- **Πρόβλημα Συντομότερου Μονοπατιού:**
 - **Στιγμιότυπο:** Κατευθυνόμενο γράφημα $G(V, E)$, μήκη στις ακμές $w: E \rightarrow \mathbb{R}$, κορυφές $s, t \in V$.
 - **Ερώτηση:** Ποιο είναι το συντομότερο $s - t$ μονοπάτι;

Παραδείγματα Προβλημάτων

□ Πρόβλημα κύκλου Hamilton:

- **Στιγμιότυπο:** Γράφημα $G(V, E)$.
- **Ερώτηση:** Υπάρχει κύκλος Hamilton στο G ;

□ Πρόβλημα Πλανόδιου Πωλητή:

- **Στιγμιότυπο:** Σύνολο $N = \{1, \dots, n\}$ σημείων, αποστάσεις $d : N \times N \rightarrow \mathbb{R}_+$.
- **Ερώτηση:** Ποια περιοδεία ελαχιστοποιεί συνολικό μήκος ή ισοδύναμα, **ποια μετάθεση π** του N ελαχιστοποιεί το:

$$\sum_{i=1}^{n-1} d(\pi(i), \pi(i+1)) + d(\pi(n), \pi(1))$$

Προβλήματα και Τυπικές Γλώσσες

- Πρόβλημα βελτιστοποίησης → πρόβλημα **απόφασης** με φράγμα B .
 - **Ελαχιστοποίηση**: Ξεφικτή λύση με **κόστος** $\leq B$;
 - **Μεγιστοποίηση**: Ξεφικτή λύση με **κέρδος** $\geq B$;
 - Πρόβλημα βελτιστοποίησης **λύνεται σε πολυωνυμικό χρόνο ανν** αντίστοιχο **πρόβλημα απόφασης** λύνεται σε πολυωνυμικό χρόνο.
- Πρόβλημα απόφασης → **τυπική γλώσσα** με κωδικοποίηση.
 - Στιγμιότυπο: **συμβολοσειρά** αλφαβήτου Σ .
 - Πρόβλημα: **γλώσσα**, υποσύνολο Σ^* .
 - **Εύλογη** κωδικοποίηση, π.χ. δυαδική, χωρίς «σπατάλη» συμβόλων.
- Πρόβλημα Π και κωδικοποίηση e : **γλώσσα** $L(\Pi, e)$ με συμβ/ρές που αντιστοιχούν σε **ΝΑΙ-στιγμιότυπα** του Π .

$$L(\Pi, e) = \{e(x) \in \Sigma^* : x \in \Pi\}$$

Ντετερμινιστικές Μηχανές Turing

- Ντετερμινιστική Μηχανή Turing (DTM) $M \equiv (Q, \Sigma, \delta, q_0, F)$
 - Q σύνολο καταστάσεων.
 - Σ αλφάβητο εισόδου και $\Gamma = \Sigma \cup \{\sqcup\}$ αλφάβητο ταινίας.
 - $q_0 \in Q$ αρχική κατάσταση.
 - $F \subseteq Q$ τελική κατάσταση (συνήθως YES, NO, HALT).
 - $\delta : (Q \setminus F) \times \Gamma \mapsto Q \times \Gamma \times \{L, R, S\}$ συνάρτηση μετάβασης.
(κατάσταση q , διαβάζει a) \rightarrow
(νέα κατάσταση q' , γράφει a' , κεφαλή μετακινείται L, R ή S).
- Απεριόριστη ταινία ανάγνωσης / εγγραφής και κεφαλή που μετακινείται στις θέσεις τις ταινίας.
 - Διαβάζει είσοδο από ταινία.
 - Έξοδος: τελική κατάσταση και περιεχόμενο ταινίας.

Ντετερμινιστικές Μηχανές Turing

- Υπολογισμός DTM μπορεί **να μην τερματίζει!**
 - ... σε αντίθεση π.χ. με DFA.
 - Συνολική κατάσταση ή **διαμόρφωση** (configuration): $(q, \underline{\sigma} \alpha \sigma_r)$
 - (τρέχουσα κατάσταση q , συμβ/ρά αριστερά κεφαλής σ_l , σύμβολο σε θέση κεφαλής, συμβ/ρά δεξιά κεφαλής σ_r).
 - Αρχική διαμόρφωση με είσοδο $x = x_1 x_2 \dots x_n$: $(q_0, \underline{x_1 x_2 \dots x_n})$
 - Τελική διαμόρφωση με έξοδο $y = y_1 y_2 \dots y_m$: $(\text{HALT}, \underline{y_1 y_2 \dots y_n})$
 - Υπολογισμός DTM M : **συνάρτηση** $|-$ και σχέση $|-\ast$.
 - $|-$: διαμόρφωση που προκύπτει από τρέχουσα σε ένα βήμα.
 - $|-\ast$: διαμορφώσεις που προκύπτουν σε κάποιο #βημάτων.
- Για $(q_0, \underline{x_1 x_2 \dots x_n}) \vdash^\ast (\text{YES}, \dots)$ γράφουμε $M(x) = \text{YES}$
Για $(q_0, \underline{x_1 x_2 \dots x_n}) \vdash^\ast (\text{NO}, \dots)$ γράφουμε $M(x) = \text{NO}$
Για $(q_0, \underline{x_1 x_2 \dots x_n}) \vdash^\ast (\text{HALT}, \underline{y_1 y_2 \dots y_n})$ γράφουμε $M(x) = y$

Παράδειγμα Μηχανής Turing

- Σε DTM με $\Sigma = \{0, 1\}$ μπορούμε να αντιστοιχίσουμε μερική **συνάρτηση** $f : \mathbb{N} \mapsto \mathbb{N}$ όπου είσοδος και έξοδος κωδικοποιούνται στο **δυναμικό σύστημα**.
- «Αλγόριθμος» DTM που με **είσοδο** x υπολογίζει **$x+1$** :
 - **Κάνε τρέχον** το κύτταρο με **τελευταίο σύμβολο** της εισόδου x ;
 - **repeat**
 - Αν τρέχον κύτταρο έχει \sqcup , γράψε **1** και **σταμάτα**;
 - Αν τρέχον κύτταρο έχει **1**, γράψε **0**, **κάνε τρέχον** το αμέσως **αριστερότερο** κύτταρο, και **κρατούμενο** $:= 1$;
 - Αν τρέχον κύτταρο έχει **0**, γράψε **1**, **κάνε τρέχον** το αμέσως **αριστερότερο** κύτταρο, και **κρατούμενο** $:= 0$;
 - **until** **κρατούμενο** = 0;
 - **Κάνε τρέχον** το κύτταρο με **πρώτο σύμβολο** του $x+1$ και **σταμάτα**;

Παράδειγμα Μηχανής Turing

- Μηχανή Turing που με είσοδο x υπολογίζει $x+1$:

$$Q = \{q_0, q_1, q_2, \text{HALT}\}$$

$$\Sigma = \{0, 1\}$$

q_0

$$F = \{\text{HALT}\}$$

	0	1	\sqcup
q_0	$(q_0, 0, R)$	$(q_0, 1, R)$	(q_1, \sqcup, L)
q_1	$(q_2, 1, L)$	$(q_1, 0, L)$	$(\text{HALT}, 1, S)$
q_2	$(q_2, 0, L)$	$(q_2, 1, L)$	(HALT, \sqcup, R)

- Παράδειγμα λειτουργίας με είσοδο $x = 1011$:

$$\begin{aligned}
 &(q_0, \underline{1}011) \vdash (q_0, 1\underline{0}11) \vdash (q_0, 10\underline{1}1) \vdash (q_0, 101\underline{1}) \vdash \\
 &(q_0, 1011\underline{\sqcup}) \vdash (q_1, 101\underline{1}) \vdash (q_1, 10\underline{1}0) \vdash (q_1, 1\underline{0}00) \vdash \\
 &(q_2, \underline{1}100) \vdash (q_2, \underline{\sqcup}1100) \vdash (\text{HALT}, \underline{1}100)
 \end{aligned}$$

Ντετερμινιστικές Μηχανές Turing

- **Ντετερμινισμός:** υπολογισμός $M(x)$ εξελίσσεται με προδιαγεγραμμένο τρόπο (\vdash είναι συνάρτηση).
 - $M(x)$ τερματίζει σε τελική κατάσταση ή δεν τερματίζει.
 - $M(x) = \text{YES}$: M αποδέχεται x . $L(M) = \{x \in \Sigma^* : M(x) = \text{YES}\}$
 - $M(x) = \text{NO}$: M απορρίπτει x .
 - $M(x) = y$: M υπολογίζει $y = f(x)$.
- **Καθολική Μηχανή Turing U :** $U(M; x) = M(x)$.
 - U προσομοιώνει κάθε άλλη DTM για οποιαδήποτε είσοδο.
 - U διαβάζει ως είσοδο περιγραφή DTM M και είσοδο x για M .
 - U προσομοιώνει υπολογισμό $M(x)$ και καταλήγει σε ίδιο αποτέλεσμα.

Υπολογισιμότητα

- Γλώσσα (πρόβλημα) L **αποκρίσιμη** (decidable),
 - ή **υπολογίσιμη** (computable),
αναδρομική (recursive),
επιλύσιμη (solvable):
 $\exists \text{DTM } M: \begin{cases} \forall x \in L, M(x) = \text{YES} \\ \forall x \notin L, M(x) = \text{NO} \end{cases}$
- Γλώσσα (πρόβλημα) L **αποδεκτή** (acceptable),
 - ή **ημιαποκρίσιμη** (semidecidable),
αναδρομικά απαριθμήσιμη
(recursively enumerable),
καταγράψιμη (listable):
 $\exists \text{DTM } M: \begin{cases} \forall x \in L, M(x) = \text{YES} \\ \forall x \notin L, M(x) = \nearrow \end{cases}$
- (Μερική) συνάρτηση f **υπολογίσιμη**:
 $\exists \text{DTM } M: \begin{cases} M(x) = y & \text{αν } f(x) = y \\ M(x) = \nearrow & \text{αν } f(x) \text{ δεν ορίζεται} \end{cases}$

Υπολογισιμότητα

- Να δείξετε ότι:
 - Κάθε αποκρίσιμη γλώσσα είναι και ημιαποκρίσιμη.
 - Αν γλώσσα L αποκρίσιμη, τότε συμπληρωματική αποκρίσιμη.
 - Γλώσσα L αποκρίσιμη ανν L και συμπληρωματική της L ημιαποκρίσιμες.

Θέση Church – Turing

- DTM πολύ **ισχυρό** υπολογιστικό μοντέλο!
- Προσπάθεια **ενίσχυσης** με επιπλέον δυνατότητες. Π.χ.
 - Πολλαπλές ταινίες.
 - Ταινία δύο (ή γενικότερα d) διαστάσεων.
 - Πολλαπλές κεφαλές.
 - Μη ντετερμινισμός (σχέση μετάβασης).
- Μπορεί ευκολότερος σχεδιασμός και «μικρή» επιτάχυνση.
 - «Ενισχυμένες» $(D, N)TM$ **προσομοιώνονται από τυπικές DTM.**
- **Δεν ενισχύεται** το υπολογιστικό **μοντέλο!**
 - **Ίδια κλάση** αποκρίσιμων και ημιαποκρίσιμων γλωσσών.

Θέση Church – Turing

- Δεν υπάρχει υπολογιστικό μοντέλο **ισχυρότερο** από DTM.
 - Π.χ. αναδρομικές συναρτήσεις, RAM, Church, Post, Markov, ...
- **Θέση των Church – Turing:**
Υπολογίσιμο αν DTM αποκρίσιμο!
 - Διαισθητικά: **αλγόριθμος** είναι DTM που τερματίζει πάντα (με σωστό αποτέλεσμα).
 - Θέση Church – Turing **δεν** μπορεί να αποδειχθεί.
 - Είναι θεωρητικά δυνατόν, αλλά **πρακτικά απίθανο** να διατυπωθεί στο μέλλον ισχυρότερο υπολογιστικό μοντέλο.

Μη-Υπολογισιμότητα

- Υπάρχουν **μη επιλύσιμα προβλήματα** (μη αποκρ. γλώσσες).
 - Γλώσσες μη αριθμήσιμες, DTM αριθμήσιμες!
- **Πρόβλημα τερματισμού** (halting problem):
 - Δεδομένης DTM M και συμβ/ράς x , $M(x)$ τερματίζει;
 - Υπάρχει(;) DTM / πρόγραμμα H που δέχεται ως είσοδο (οποιαδήποτε) DTM / πρόγραμμα M και την είσοδο x για M , και απαντά **YES** αν $M(x)$ τερματίζει και **NO** αν $M(x)$ δεν τερματίζει.
- Πρόβλημα τερματισμού είναι **μη επιλύσιμο!**
 - Απόδειξη με **διαγωνιοποίηση**.

Μη-Υπολογισιμότητα

- Πρόβλημα τερματισμού είναι **μη επιλύσιμο!**
 - Έστω ότι **υπάρχει DTM H** που για κάθε DTM M και είσοδο x, H(M; x) αποφασίζει αν M(x) **τερματίζει ή όχι**.
$$H(M; x) = \begin{cases} \text{YES} & \text{αν } M(x) \text{ τερματίζει} \\ \text{NO} & \text{αν } M(x) \text{ δεν τερματίζει} \end{cases}$$
 - Με βάση DTM H, κατασκευάζουμε **DTM D(M)**:
if **H(M; M) = YES** then **run forever** else **halt**
 - Εκ κατασκευής, D(M) τερματίζει αν M(M) **δεν** τερματίζει!
 - Δοκιμάζουμε D με είσοδο τον εαυτό της:
 - D(D) τερματίζει αν D(D) **δεν** τερματίζει! **Άτοπο!!!**
- Υπάρχουν πολλά άλλα μη επιλύσιμα προβλήματα.
 - Π.χ. επίλυση Διοφαντικών εξισώσεων.