1.1 Στο προηγούμενο επεισόδιο

- **Θεωρία Πολυπλοκότητας**

  Ταξινομεί τα επιλύσιμα προβλήματα σε εύκολα ή δύσκολα

- **Turing Machine**

  → Τώρα $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ — transition function $\delta: Q \times \Gamma \to Q \times \Gamma \times \{R, L\}$

  states · input alphabet · tape alphabet · initial state · accepting state · rejecting state

  → **Configuration** snapshot της κατάστασης της T.M.
  περιγραφή πλήρως σε κάθε βήμα της λειτουργίας της βρίσκεται

  ορίζεται από i) state q της T.M
  ii) περιεχόμενα της ταινίας
  iii) θέση της κεφαλής της T.M.

- **Γλώσσα της T.M M**

  - Στην $L(M)$ ανήκουν όλες οι λέξεις που η T.M M αποδέχεται
  - Μια γλώσσα λέγεται **αναγνωρίσιμη**, αν $\exists$ T.M M η οποία την αναγνωρίζει, δηλαδή για είσοδο α, $\begin{cases} \text{αν } \alpha \in L(M), \text{ η M αποδέχεται} \\ \text{αν } \alpha \notin L(M), \text{ η M απορρίπτει ή βρόχος} \end{cases}$
  - Μια γλώσσα λέγεται **διαγνώσιμη**, αν $\exists$ T.M η οποία την διαγιγνώσκει, δηλαδή για είσοδο α, $\begin{cases} \text{αν } \alpha \in L(M), \text{ η M αποδέχεται} \\ \text{αν } \alpha \notin L(M), \text{ η M απορρίπτει} \end{cases}$

- **Διαγνωσιμότητα Γλωσσών**

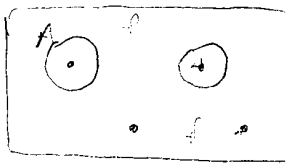
  - **Θεώρημα** Υπάρχουν μη-διαγνώσιμες γλώσσες και μη-αναγνωρίσιμες γλώσσες
    Απ: Halting Problem — Απ: 4.15, Sipser
  - Η L είναι διαγνώσιμη αν $\begin{cases} L \text{ είναι αναγνωρίσιμη} \\ \bar{L} \text{ είναι αναγνωρίσιμη} \end{cases}$

1.2 Εργαλεία

- Υπολογίσιμη Συνάρτηση $f: \Sigma^* \to \Sigma^*$ αν $\exists$ Τ.Μ που $\forall$ είσοδο w σταματάει έχοντας στην ταινία $f(w)$

- Απεικονιστική Αναγωγή

  Η γλώσσα A είναι απεικονιστικά αναγώγιμη στην B, $A \leq_m B$ αν υπάρχει υπολογίσιμη συνάρτηση $f: \Sigma^* \to \Sigma^*$ τ.ω: $\forall w \in A \iff f(w) \in B$

  ! το $\iff$ σημαίνει πως αν $w \notin A$ τότε $f(w) \notin B$

  Θ1 Αν $A \leq_m B$, B διαγνώσιμη τότε A διαγνώσιμη

  Θ2 Αν $A \leq_m B$, A μη-διαγνώσιμη τότε B μη-διαγνώσιμη

- Πολυωνυμική Ισοδυναμία Δύο υπολογιστικά μοντέλα είναι πολυωνυμικώς ισοδύναμα αν ένα από αυτά μπορεί να προσομοιώσει το άλλο πολυωνυμική αύξηση χρόνου

- Υπολογίσιμη σε πολυωνυμικό χρόνο Συνάρτηση $f: \Sigma^* \to \Sigma^*$ αν $\exists$ Τ.Μ πολυωνυμικού χρόνου που $\forall$ είσοδο w σταματάει έχοντας στην ταινία $f(w)$

- Απεικονιστική Αναγωγή Πολυωνυμικού Χρόνου

  Ο ορισμός είναι ίδιος με την Απεικονιστική Αναγωγή με μια διαφορά πως η f εδώ πρέπει να είναι υπολογίσιμη σε πολυωνυμικό χρόνο. Γράφουμε $A \leq_p B$ βλ. 7.25, Sipser

2.1 Χρονική Πολυπλοκότητα

- Χρονική Πολυπλοκότητα / Χρόνος Εκτέλεσης

  Έστω M μία Τ.Μ. Η χρονική πολυπλοκότητα της M είναι η συνάρτηση $f: \mathbb{N} \to \mathbb{N}$, που για κάθε n = μήκος εισόδου επιστρέφει $f(n)$ = το μέγιστο πλήθος βημάτων που μπορεί να εκτελέσει η M για είσοδο μήκους n

- Κλάση $TIME(t(n))$

  Έστω συνάρτηση $t: \mathbb{N} \to \mathbb{R}^+$. Η κλάση χρονικής πολυπλοκότητας $TIME(t(n))$ ορίζουμε το σύνολο όλων των γλωσσών που διαγιγνώσκονται από μηχανή Turing χρόνου $O(t(n))$

  π.χ $O(n^2)$ - όλες οι γλώσσες που μπορούν να διαγνωστούν σε $O(n^2)$

  ⚠ Αντίθετα με την θεωρία υπολογισιμότητας, όπου από το θέμα Church-Turing ξέραμε πως όλα τα υπολογιστικά μοντέλα είναι ισοδύναμα, η στην θεωρία πολυπλοκότητας η επιλογή μοντέλου επηρεάζει την χρονική πολυπλοκότητα μιας γλώσσας

  π.χ σε δικοταινιακή Τ.Μ. η B είναι χρόνου $O(n \log n)$
  σε μονοταινιακή Τ.Μ. η B είναι χρόνου $O(n^2)$

- Κλάση $NTIME(t(n))$

  Σε αυτήν την κλάση ανήκουν όλες οι γλώσσες που διαγιγνώσκονται από non-det Turing Machine χρόνου $O(t(n))$

2.2 Οι κλάσεις P και NP

• Η κλάση γλωσσών P

αποτελείται από όλες τις γλώσσες που μπορούν να διαγνωστούν σε πολυωνυμικό χρόνο από ιδανική αιτιοκρατική Τ.Μ

Άρα $P = \bigcup_k TIME(n^k)$

- Η P είναι αναλλοίωτη για όλα τα υπολογιστικά μοντέλα που είναι πολυωνυμικώς ισοδύναμα με την αιτιοκρατική φωνοσαντιακή Τ.Μ

[illegible]

• Η κλάση γλωσσών NP

αποτελείται από όλες τις γλώσσες που είναι επαληθεύσιμες σε πολυωνυμικό χρόνο

- Επαληθευτής γλώσσας A είναι ένας αλγόριθμος V τέτοιος ώστε

$A = \{ w \mid$ ο V αποδέχεται τον $\langle w, c \rangle$, όπου c το πιστοποιητικό της ορθότητας της w στην $A \}$

π.χ
- Η γλώσσα $HAMPATH = \{ G \mid$ ο G περιέχει χαμιλτονιανή διαδρομή $\}$ είναι πολυωνυμικώς επαληθεύσιμη, καθώς αρκεί να μας δώσουν το πιστοποιητικό c = μια χαμιλτονιανή διαδρομή $\in G$ και εμείς μπορούμε απλώς να ελέγξουμε αν το c είναι hamilton path $\Rightarrow G \in HAMPATH$
- Η γλώσσα $\overline{HAMPATH}$ μπορεί να μην είναι καν πολυωνυμικά επαληθεύσιμη. Ακόμα και αν μας δίνουν G που ισχυρίζεται πως δεν έχει hamilton path, δεν έχουμε τρόπο να επαληθεύσουμε πως δεν έχει hampath χωρίς να χρησιμοποιήσουμε αλγόριθμο εκθετικού χρόνου.

Επίσης, $NP = \bigcup_k NTIME(n^k)$

2.3 Η κλάση NP-complete

- **Intuition** Θέλουμε μια κλάση γλωσσών που να αντιπροσωπεύει όλη την δυσκολία της NP

  Η NP-complete περιέχει τα δυσκολότερα προβλήματα στην NP

- **Ορισμός**

  Μια γλώσσα $B \in$ NP-complete αν 1) $B \in NP$
  2) $\forall A \in NP,\ A \leq_p B$

  - Έτσι αποδείχτηκε από τους COOK-LEVIN το πρώτο NP-πλήρες πρόβλημα, το SAT

- **Συνήθης Αποδεικτική Διαδικασία**

  Η γλώσσα B είναι NP-complete αν 1) $B \in NP$
  2) $A \in$ NP-complete
  3) $A \leq_p B$

2.4 Άλλες Κλάσεις

- **co-NP** Μια γλώσσα $A \in$ co-NP αν η $\bar{A} \in NP$

  π.χ. • $\overline{\text{HAMPATH}} \in$ co-NP εξ'ορισμού, αφού HAMPATH $\in$ NP

  • VALIDITY = { B | B είναι boolean formula που ικανοποιείται από όλες τις δυνατές αναθέσεις }

  VALIDITY $\in$ co-NP αφού $\overline{\text{VALIDITY}} \in$ NP $\rightsquigarrow$ αρκεί να φανταστούμε μία μη-ντετερμινιστική συσκευή

- **EXPTIME (ή EXP)**

  αποτελείται από όλες τις γλώσσες που μπορούν να διαγνωστούν από det. TM σε χρόνο $2^{p(n)}$, όπου $p(n)$ πολυώνυμο, δηλαδή

  $$\text{EXPTIME} = \bigcup_k \text{TIME}(2^{n^k})$$

- **NP-hard**

  Μια γλώσσα $A \in$ NP-hard αν $B \in$ NP-complete και $B \leq_p A$

  - Τα NP-hard προβλήματα είναι τουλάχιστον όσο δύσκολα όσο τα δυσκολότερα προβλήματα της NP
  - Δεν μπορώ να αποδείξω πως $A \in NP$ γιατί και $A \in$ NP-complete

3.1 Χωρική Πολυπλοκότητα

- Χωρική Πολυπλοκότητα / Χώρος Εκτέλεσης

  Έστω M μια Τ.Μ. Η χωρική πολυπλοκότητα της M είναι συνάρτηση $f: \mathbb{N} \to \mathbb{N}$ που για κάθε n=μήκος εισόδου επιστρέφει $f(n)$ = το μέγιστο πλήθος θέσεων ταινίας που είναι δυνατόν να διατρέξει η M όταν έχει είσοδο μήκους n

  ⚠ Για non-deterministic T.M M η $f(n)$ ορίζεται ως το μέγιστο πλήθος θέσεων ταινίας ανάμεσα σε όλους τους δυνατούς κλάδους υπολογισμού.

- Κλάση SPACE($f(n)$)

  Έστω $f: \mathbb{N} \to \mathbb{R}^+$. Τότε η SPACE($f(n)$) περιέχει όλες τις γλώσσες που διαγιγνώσκονται από ντετερμινιστική T.M χώρου $O(f(n))$

- Κλάση NSPACE($f(n)$)

  Έστω $f: \mathbb{N} \to \mathbb{R}^+$. Τότε η NSPACE($f(n)$) περιέχει όλες τις γλώσσες που διαγιγνώσκονται από non-deterministic T.M. χώρου $O(f(n))$

⚠ Ο χώρος είναι δραστικότερος πόρος από τον χρόνο, δηλαδή συνήθως (πάντα) θα χρειάζεται λιγότερος χώρος από ότι χρόνος. Αυτό δείχνει γεγονότως πως είναι επαναχρησιμοποιήσιμος, αντίθετα με τον χρόνο (when you where young, κτλ...)

Π.χ Ξέρουμε πως, SAT ε NP-complete. Αυτό είναι ενδεχόμενο πως βάλλον θέλει υπερπολυωνυμικό ~~πολυωνυμικό~~ ίσως και εκθετικό χρόνο για να λυθεί.

Παρ'όλα αυτά, επαναχρησιμοποιώντας τον χώρο μπορούμε να δώσουμε αλγόριθμο που το επιλύει σε γραμμικό χώρο.

Sketch of proof: Σε χώρο $O(n)$ αποθηκεύουμε την τρέχουσα εισόδοιά, ώστε να ξέρουμε και ποιά είναι η επόμενη

Σε χώρο $O(n)$ υπολογίζουμε αν η εισόδοια είναι αλήθεια

Άρα ανοδικά χώρος $O(n) \to$ γραμμικός.

- Πώς ακριβώς μετράμε την χωρική πολυπλοκότητα?

Κάθε φορά που άφηνε Turing Machine, πρώτη πλέον αναπαριστάνεται μια T.M με 3 ταινίες:

(read only, write only) • input και output tape, απ'όπου η T.M παίρνει είσοδο κ'γράφει το αποτέλεσμα

(read/write) • work tape το "πρόχειρο" που χρησιμοποιεί για τις πράξεις της

⚠ Η space complexity μετράται από το πόσες θέσεις της work tape χρησιμοποίησε η T.M

⚠⚠ Αν αφήναμε την T.M να γράφει και στην input tape, τότε θα μπορούσαμε να μετρήσουμε ένα γραμμικές σε μέγεθος της εισόδου χωρικές πολυπλοκότητες

Παρατήρηση: Η γλώσσα $\overline{\text{ΠΛΗΡΟΤΗΤΑ/NFA}} = \{A \mid$ το A είναι NFA που δεν αποδέχεται όλες τις συμβολοσειρές$\}$
διαγωνιοποιείται από non-deterministic T.M χρησιμοποιώντας χώρο

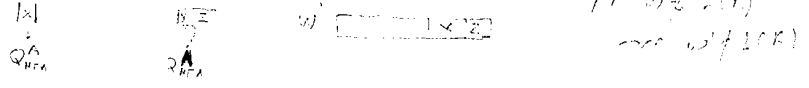
Απ: Θα χρησιμοποιήσουμε τον "αντιπροσωπευτικό μαντεμένιο" για να μαντέψουμε μία λέξη που απορρίπτεται από το NFA A.

Ως είσοδο παίρνουμε λοιπόν το DFA $A = (Q, \Sigma, \delta, q_0, F)$ (σύνολο καταστάσεων, αρχική κατάσταση, σύνολο αποδεκτικών καταστάσεων)

Θα χρησιμοποιήσουμε τον ορισμό της προηγούμενης σελίδας όπου μπορούσαμε να γράψουμε μόνο πάνω στο work tape (για απλότητα υποθέσαμε, θα μπορούσαμε να γράψουμε στο input tape αφού προηγουμένως να αποθέσουμε χώρο)

Για είσοδο $\langle A \rangle$, 1) Αντιγράφουμε τις καταστάσεις Q στο worktape
2) Τοποθετούμε έναν μαρκαδόρο στην initial state q_0
3) Επαναλαμβάνουμε 2^q φορές, όπου $q = |Q|$ = αριθμός καταστάσεων
4) Επιλέγουμε μη-ντετερμινιστικά ένα σύμβολο εισόδου και προσομοιώνουμε την αναζήτηση του από τον M
Ύστερα σημειώνουμε την κατάσταση όπου θα βρεθεί καταλήγοντας
5) Αν υπήρξε επανάληψη όπου όλοι οι καταστάσεις αποδοχής δεν είχαν πάνω τους κανένα μαρκαδόρο, τότε είχες κατασκευάσει έναν τέτοιο μη δεκτή λέξη $w \notin L(A)$ άρα $A \notin \overline{\text{ΠΛΗΡΟΤΗΤΑ/NFA}} \Rightarrow A \in \overline{\text{ΠΛΗΡΟΤΗΤΑ/NFA}}$
Άρα ΑΠΟΔΕΧΟΥ. Διαφορετικά απορρίπτουμε.

- Γιατί 2^q φορές;
(Και όπως το pumping lemma)
Γιατί υπάρχουν 2^q δυνατές configurations – συνδυασμοί σημαδεμένων/μη σημαδεμένων καταστάσεων
Άρα αν η λέξη με μήκος $\geq 2^q$ που απορρίπτεται τότε το configuration θα είχε επαναληφθεί σε κάποιο προηγούμενο βήμα, και έτσι αφαιρώντας το κομμάτι της λέξης που προσομοίωσε θα μπορούσαμε να φτιάξουμε μικρότερη μη αποδεκτή λέξη



- Γραμμικός Χώρος;
Ναι, χρειάζομαι: → Γράφω q καταστάσεις, χώρος $O(q) = O(n)$, όπου $n = |\langle A \rangle|$
→ Μετράω $0 \to 2^q$ άρα $\log_2 2^q$ bits $= q$ bits, άρα $O(q) = O(n)$

Άρα συνολικά χρειάζεται χώρος $O(n)$

3.2 Θεώρημα Savitch

Για $f: N \to R^+$ με $f(n) \geq n$ ισχύει $NSPACE(f(n)) \subseteq SPACE(f^2(n))$

! Ενώ ο μη-ντετερμινισμός πιστεύουμε πως προκαλεί εκθετική μείωση της πολυπλοκότητας σε σχέση με τον ντετερμινισμό όσο αφορά τον χρόνο, στο χώρο τα πράγματα είναι διαφορετικά. Μπορώ να προσομοιώσω μία non-det TM με μία Τ.Μ. απαιτώντας μόνο πολυωνυμικά περισσότερο χώρο (τετραγωνικό)

!! Το Θεώρημα ισχύει και για $f(n) \geq \log n$

Αλ: ο Πρόβλημα Μεταβατότητας "Υπάρχει μονοπάτι από τον s στον t με μήκος $\leq k$;"

Θα δώσουμε ντετερμινιστικό αλγόριθμο που θα λύνει το πρόβλημα της μεταβατότητας

```
ΜΕΤΑΒΑΣΗ (x, y, t)
    1) Αν t=1 τότε
        αν x=y ή από τον x μπορώ να πάω στον y σε 1 βήμα
        τότε αποδέξου. Αλλιώς απορριπτω.
    2) Αν t>1 τότε:
        Για κάθε συνδιάφορο κορυφή k ∈ G - {x,y} κάλεσε:
    3)      ΜΕΤΑΒΑΣΗ (x, k, t/2)
    4)      ΜΕΤΑΒΑΣΗ (k, y, t/2)
    5)      Αν 3), 4) δώσαν Αποδοχή, τότε αποδέξου
    6) Απορριψε
```

Ας δούμε πόσο χώρο χρειάζεται ο αλγόριθμος μας:

- Σε κάθε αναδρομική κλήση το t ελαττώνεται σε μισό. Άρα το βάθος της αναδρομής είναι $\log t$
- Σε κάθε αναδρομική κλήση πρέπει να αποθηκεύσουν τα ορίσματα και οι μεταβλητές συνάρτησης σε αυτό το επίπεδο αναδρομής, έτσι ώστε όταν επιστρέφουμε από τις αναδρομικές κλήσεις να μπορέσουμε να ανακτήσουμε αυτές τις πληροφορίες και να συνεχίσουμε τον έλεγχο.
  Άρα οι πληροφορίες είναι τα x, y, t_i οπότε θέλω $O(\log n)$
  ($\log n$ $\log n$ $t_i \leq n \Rightarrow \log n$)

Άρα ο αλγόριθμος μας χρειάζεται χώρο $O(\log t)\, O(\log n) \overset{t \leq n}{=} O(\log^2 n)$

- Χρήση του ΜΕΤΑΒΑΣΗ(x,y,n) στην απόδειξη

Θέλουμε δείξουν να προσομοιώσουμε αντιστρατευτικά μία non-det T.M. χώρου $O(f(n))$

Αφού η non-det T.M. χρησιμοποιεί χώρο $O(f(n))$, μπορεί να εκτελέσει εως και $2^{O(f(n))}$ βήματα. (και να έχει ως και $f(n)\,2^{O(f(n))}$ διαφορετικές φάσεις)

↳ Λόγω αυξημένης σημασίας στον $f(n)$ πάνω όσες και μπορεί να βρίσκεται η κεφαλή $2^{O(f(n))}$ διαφορετικές καταστάσεις και ταινίες $|\Gamma|^{f(n)}$ και $|Q|$ καταστάσεις της T.M

Συνοψίζουμε, έστω N η non-det T.M χώρου $O(f(n))$

Θα έχει το πολύ $2^{O(f(n))}$ βήματα

Έστω C_{INIT}, C_{ACCEPT} οι φάσεις έναρξης και αποδοχής της N για κάποια είσοδο w ($|w|=n$)

Η det T.M M που την προσομοιώνει λειτουργεί ως εξής:

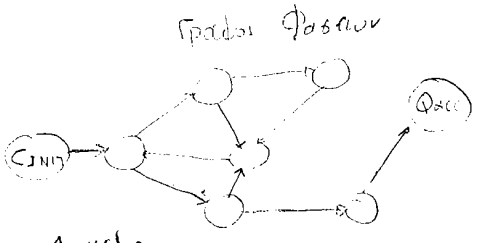
$M =$ "Για είσοδο w
1) Κάλεσε ΜΕΤΑΒΑΣΗ$(C_{INIT}, C_{ACCEPT}, 2^{O(f(n))})$

Ας δούμε πόσο χώρο χρησιμοποιεί η M

- Το βάθος της αναδρομής είναι $O(\log 2^{O(f(n))}) = O(f(n))$
- Σε κάθε επίπεδο αναδρομής χρειαζόμαστε χώρο για να αποθηκεύσουμε τις C_i, C_j, t_i άρα όσο $O(f(n))$

Συνολικά χρειάζεται χώρο $O(f^2(n))$

- Intuition



Η non det T.M. μπορεί να υπολογίσει απευθείας με μη-ντετερμινιστικό τρόπο δηλαδή το σωστό μονοπάτι υπολογισμού.

Η det T.M. πρέπει να ψάξει όλο τον γράφο των φάσεων του υπολογισμού!

Για να τα καταφέρει χάλλον θα χρειαστεί εκθετικό χρόνο, αλλά με τον έξυπνο τρόπο που είδαμε θα χρειαστεί μόνο πολυωνυμικά περισσότερο χώρο

3.3 Η κλάση PSPACE

- Η κλάση PSPACE

  αποτελείται από όλες τις γλώσσες που διαγιγνώσκονται από κάποια Τ.Μ πολυωνυμικού χώρου, δηλαδή $PSPACE = \bigcup_k SPACE(n^k)$

- Η κλάση NPSPACE

  αποτελείται από όλες τις γλώσσες που μπορούν να διαγνωσθούν από κάποια μη-ντετ Τ.Μ πολυωνυμικού χώρου, δηλαδή $NPSPACE = \bigcup_k NSPACE(n^k)$

- PSPACE = NPSPACE

  προκύπτει άμεσα από το Θ. Savitch αφού το τετράγωνο ενός πολυωνύμου είναι πολυώνυμο

3.4 Ιεραρχία Κλάσεων Γλωσσών

- $P \subseteq NP$ trivial

- $P \subseteq PSPACE$ Αφού μια μηχανή που λειτουργεί πολυωνυμικό χρόνο, το πολύ να χρησιμοποιεί πολυωνυμικό χώρο, σε χειρότερη περίπτωση στην οποία χρησιμοποιεί κάθε φορά διαφορετική θέση ταινίας / βήμα

- $NP \subseteq NPSPACE$ Για τον ίδιο λόγο

- $PSPACE = NPSPACE$ Θεώρημα Savitch

- $NP \subseteq PSPACE$

- $PSPACE \subseteq EXPTIME$

  Όπως είδαμε μηχανή που χρησιμοποιεί χώρο $f(n)$ μπορεί να έχει το πολύ $f(n)2^{O(f(n))}$ φάσεις.

  Αν βρεθεί δύο φορές σαν ίδια φάση κατά την διάρκεια του υπολογισμού, τότε η Τ.Μ βρίσκεται σε ατέρμονα βρόχο - δεν τερματίζει - δεν διαγιγνώσκεται.

  Άρα Τ.Μ που χρησιμοποιεί χώρο $f(n)$ δεν μπορεί να υπερβαίνει τα $f(n)2^{O(f(n))}$ βήματα

- ΙΕΡΑΡΧΙΑ

  Άρα $P \subseteq NP \subseteq PSPACE = NPSPACE \subseteq EXPTIME$

  Δεν ξέρω σε κάθε περίπτωση οι εγκλεισμοί είναι γνήσιοι.

3.5 Η κλάση PSPACE-complete

• Ορισμός

Μια γλώσσα $B \in$ PSPACE-complete αν: 1) $B \in$ PSPACE
2) $\forall A \in$ PSPACE, $A \leq_p B$

• Αν η B ικανοποιεί μόνο 2η συνθήκη μόνο, τότε η B ∈ PSPACE-hard

! Γιατί χρησιμοποιούμε ανοιχτή πολυωνυμικού χρόνου;

Και όχι πολυωνυμικού χώρου;

Είδαμε πως αν η ανοιχτή ήταν ανοιχτή πολυωνυμικού χώρου, τότε μπορεί να χρειαζόταν εκθετικό χρόνο για να την υπολογίσω αφού NP⊆PSPACE

Άρα αν είχαμε πως $\forall A \in$ PSPACE, $A \leq_{p^{\text{σε πολυωνυμικό χώρο}}} B$ τότε αυτόματα και αν βρίσκαμε αλγόριθμο πολυωνυμικού χρόνου για το B, δεν θα μπορούσαμε να λύσουμε εύκολα όλα τα άλλα προβλήματα ∈PSPACE.

Άρα το PSPACE-complete πρόβλημα δεν θα αντιστοιχούσε στη δυσκολία όλων των PSPACE προβλημάτων

Συνοψίζοντας, όπως είχαμε πλήρη προβλήματα για μια κλάση πολυπλοκότητας, πρέπει η ανοιχτή θεωρεί να είναι υπολογιστικά αντίστοιχη από κάποιες υπολογιστές της ίδιας της κλάσης.

Παρακάτω θα δούμε 3 PSPACE-complete προβλήματα.

• Όπως και στα NP-complete, στο πρώτο PSPACE-complete πρόβλημα θα πρέπει να δείξουμε τη συνθήκη 2) γενικά, δηλ πως $\forall A \in$ PSPACE, $A \leq_p B$

Επειτα, αφού έχουμε πως B∈PSPACE-complete αρκεί να δείχνουμε πως $B \leq_p C$ για να αποδείξουμε πως C ∈ PSPACE-complete

3.5.1 TQBF

- Ορισμός $TQBF = \{ \langle \phi \rangle \mid \phi$ είναι αληθής, ποσοδεσμευμένος, λογικός τύπος $\}$

  Αληθής είναι πάντα true. Κάθε ποσοδεσμευμένος λογικός τύπος είναι πάντα ή TRUE ή FALSE

  Ποσοδεσμευμένος για κάθε μεταβλητή υπάρχει ποσοδείκτης

  $T_1 = \forall x \; (x \wedge y)$ $T_1 \notin TQBF$ γιατί δεν είναι ποσοδεσμευμένος

  $T_2 = \forall x \exists y \; (x \wedge y)$ $T_2 \in TQBF$

- Αποδείξη

  1) TQBF $\in$ PSPACE

  Θα κατασκευάσω αλγόριθμο που χάνει όλο τον χώρο λύσεων για να βρει τις τιμές αλήθειας του δοσμένου τύπου για μία εφικτή διαδοχή και έπειτα προσδιορίζει την τιμή αλήθειας του αρχικού τύπου.

  Αν προσπαθούσαμε να αποθηκεύσαμε όλες τις τιμές θα χρειαζόμασταν υπερπολυωνυμικό χώρο, καθώς εκθετικό είναι το μέγεθος των χώρων δύσεων. Αρα θα δοκιμάσουμε όπως και στο Θ Savitch έναν αναδρομικό αλγόριθμο.

  $T =$ "Για είσοδο $\langle \phi \rangle$, όπου ϕ μία quantified B.F. :

  1) Αν το ϕ δεν έχει ποσοδείκτες, τότε περιλαμβάνει μόνο σταθερές. Άρα τον υπολογίζω απ'ευθείας και αν είναι αληθής ACCEPT ενώ αν είναι ψευδής REJECT

  2) Αν το ϕ είναι της μορφής $\exists x \; (\phi')$ τότε καλώ αναδρομικά τον T για το ϕ'
  για $x = 0$ και
  για $x = 1$
  Αν μία από τις 2 κλήσεις αποδεχτεί, τότε αποδέχομαι. Αλλιώς ΑΠΟΡΡΙΠΤΩ

  3) Αν το ϕ είναι της μορφής $\forall x \; (\phi')$ τότε καλώ αναδρομικά τον T για το ϕ'
  για $x = 0$ και
  για $x = 1$
  Αν και οι δύο κλήσεις αποδέχτων, τότε ACCEPT. Αλλιώς REJECT

Ας δούμε πόσο χώρο χρειάζεται ο αλγόριθμός μας:

- Το βάθος της αναδρομής είναι όσο το πλήθος των μεταβλητών $m \leq O(m) = O(n)$
- Σε κάθε επίπεδο αναδρομής θέλω μόνο να αποθηκεύσω μία τιμή χώρο $O(1)$

Συνολικά, χώρος $O(m) = O(n)$

2) TQBF $\in$ PSPACE-hard $\Leftrightarrow$ $\forall A \in PSPACE$, $A \leq_p TQBF$

Sketch of proof

Έστω $M \in PSPACE$. Η αναγωγή απεικονίζει κάθε λέξη w σε αληθή QBFormula αν η M αποδέχεται την w

Έστω c_{INIT}, c_{ACCEPT} οι διαμορφώσεις αρχική και αποδοχής αντίστοιχα της M.
Από αδιαμφισβήτητα και προηγούμενως, ο υπολογισμός του M πρέπει να φτάσει από την $c_{INIT} \to c_{ACCEPT}$ σε το πολύ $t = 2^{f(n)}$ βήματα

Άρα θα κατασκευάσω τον $\Phi_{c_{INIT}, c_{ACCEPT}, t}$ που θα είναι αληθής αν ισχύει το παραπάνω

Το πως υπάρχει Sipser, σελ. 368-370

3.5.2 Νικηφόρες Στρατηγικές για Παίγνια

Έστω παίκτες $Y_{\text{παρχει}}$, $K_{\text{παντος}}$ οι οποίοι παίζουν το εξής παιχνίδι:

- τύπος $\phi_1 = \exists x_1 \forall x_2 \exists x_3 \forall x_4 \ldots ((x_1 \vee x_2) \wedge (\ldots) \wedge \ldots)$
- Οι παίκτες Y, K βλέπουν εναλλαξ τιμές για τα $x_1, x_2, \ldots$
- Αν η τιμοδοσία που προκύπτει είναι αληθής τότε Y wins. Αλλιώς K wins.
- Νικηφόρα Στρατηγική για τον παίκτη Y είναι κάποια στρατηγική η οποία αφήνει τον Y να κερδίζει, όπως και αν παίζει ο K
  Δηλαδή ο ϕ_1 γίνεται αληθής.

Ας ορίσουμε τώρα την γλώσσα:

ΠΑΙΓΝΙΟΛΟΓΙΚΕΣ = $\{ \langle \phi \rangle \mid$ το ϕ είναι παίγνιο λογικής όπου ο Y έχει νικηφόρα στρατηγική $\}$

- Παίγνιο Λογικής $\in$ PSPACE-complete

Απ: Προκύπτει άμεσα από το TQBF

Ο ϕ έχει νικηφόρα στρατηγική αν $\exists x_1$ τ.ω $\forall x_2$, $\exists x_3 \ldots$ ϕ αληθής

Ο $\phi \in TQBF$ αν $\exists x_1$ τ.ω $\forall x_2$, $\exists x_3 \ldots$ ϕ αληθής

3.5.3 Γενικευμένη Γεωγραφία

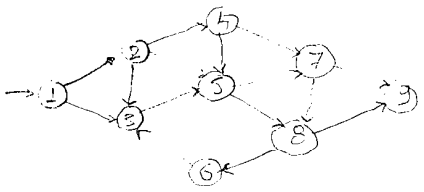
ο Ορισμοί

• Το παιχνίδι της Γεωγραφίας Παίκτες 1, 2 λένε ονόματα πόλεων. Σε κάθε γύρο ο παίκτης που παίζει πρέπει να πει όνομα πόλης που να αρχίζει με το γράμμα που τελειώνει η προηγούμενη πόλη. Δεν μπορούμε να πούμε την ίδια πόλη δύο φορές.

π.χ Pasadena → Anderlecht → Tokyo → Oslo → ...

• Γενικευμένη Γεωγραφία

$G = (V, E)$



Πλέον έχω γράφημα, και οι δυνατές επιλογές είναι οι κόμβοι. Σε κάθε γύρο μπορώ να διαλέξω τον v_{i+1} αν $\exists (v_i, v_{i+1}) \in E$

Αν ο παίκτης 1 παίζει πρώτος τότε έχει νικηφόρα στρατηγική αν

υπάρχει επιλογή του για κάθε επιλογή του Π2 να υπάρχει επιλογή του για ... να υπάρχει επιλογή

βλέπω την σύνθεση

$\exists x_1 \forall x_2 \exists x_3 \dots$

Έτσω δοθούν οι γλώσσες:

• Γεωγραφία = $\{ \langle G, b \rangle$ / στο παιχνίδι γενικευμένης γεωγραφίας με αφετηρία τον κόμβο b, ο παίκτης Π1 έχει νικηφόρα στρατηγική $\}$

ο ΓΕΩΓΡΑΦΙΑ ∈ PSPACE-complete

1) ΓΕΩΓΡΑΦΙΑ ∈ PSPACE

Θα δώσουμε αναδρομικό αλγόριθμο που θα τον διαγωνιώσουμε ως

Μ = " Για είσοδο $\langle G, b \rangle$

1) Αν out-deg$(b) = 0$, τότε ο Π1 χάνει. Άρα REJECT
2) Αφαιρούμε τον κόμβο b από το γράφημα και όλες τις ακμές που συνδέονται με αυτόν και καταγράφουμε το $G' = G - \{b\}$
3) Για κάθε κόμβο $b_1, \dots, b_k$ προς τους οποίους είχε ακμή ο b καλούμε αναδρομικά $M(G', b_i)$
4) Αν όλες οι κλήσεις απορρίπτουν τότε ο Π2 έχει νικηφόρα στρατηγική → ο Π1 δεν έχει → REJECT

   Αν υπήρχε τουλ. 1 κόμβος που δεν απορρίπτει, τότε ο Π2 δεν έχει νικηφόρα στρατηγική → ο Π1 έχει → ACCEPT

• Το βάθος αναδρομής είναι $|V| = n$ = Αριθμός κόμβων } συνολικά γραμμικός χώρος
• Σε κάθε επίπεδο αναδρομής πρέπει να αποθηκεύω μόνο έναν κόμβο

2) ΠΑΓΝΟΛΟΠΚΙΕΣ ΣΡ ΓΕΩΓΡΑΦΙΑ

Η αποδεξη απλοκωδικοποιουδηποτε σύνθετου σε ενα παιχνιδι γεωγραφίας παίρνει δυνατόν σύνθετο και κατασκευαζει γρήγορα (Sipser 376-8)

Από τα 1), 2) ΓΕΩΓΡΑΦΙΑ ∈ PSPACE-Complete.

- Αρα, εκτος αν P=PSPACE, δεν υπαρχει αλγοριθμος πολυωνυμικού χρονου για εύρεση νικηφόρας στρατηγικής.
- Σε πολλά παίγνια (π.χ. σκάκι) λόγω του τεράστιου κύρτου δένδρου, μπορούμε να βρούμε απλοποιήσεις όπως PSPACE-hardness, EXP-hardness κτλ
  Δηλαδή σκάκι, Sipser 378-379

3.6 Κλάσεις $\mathcal{L}$, $N\mathcal{L}$

- **Κλάση $\mathcal{L}$**

  Αποτελείται από όλες τις γλώσσες που μπορούν να διαγνωστούν σε λογαριθμικό χώρο από deterministic T.M., δηλαδή $\mathcal{L} = SPACE(\log(n))$

- **Κλάση $N\mathcal{L}$**

  Αποτελείται από τις γλώσσες που μπορούν να διαγνωστούν σε λογαριθμικό χώρο από non-deterministic T.M., δηλαδή $N\mathcal{L} = NSPACE(\log(n))$

- **Intuition**

  - Για λογαριθμικό χώρο και όχι χρόνο;

    Επειδή χρειαζόμαστε χρόνο τουλάχιστον $n = O(n)$ για να διαβάσουμε την είσοδο! Στον πραγματικό κόσμο η κλάση $\mathcal{L}$ βολεύει για ολοζώντανους όπου έχουμε πολλά μεγάλα μεγέθη δεδομένων εισόδου και μπορούμε να βάλουμε μόνο έναν λογαριθμό εξωτερικά στην RAM.

  - Τι μοντέλο χρησιμοποιώ;

    Input/output tape και work tape, όπως το ορίσαμε στην ενότ. 7

- **Φασικός και λογαριθμικός χώρος**

  Πλέον που ο χώρος $f(n)$ είναι $> \log n$, δεν μπορώ να πω πως μια μηχανή που χρησιμοποιεί χώρο $f(\log n)$ χρειάζεται χρόνο $2^{O(f(n))}$

  Πλέον έχουμε $O(\underbrace{c^{n f(n) g^{f(n)}}}_{|Q|} \cdot \underbrace{n}_{\text{|w|, φίλτρο στην θέση κεφαλή-μας T.M. μπορεί να βρίσκεται και στη}} \cdot \underbrace{f(n)}_{\text{τι μπορεί να είναι βρεθεί στην ταινία}} \cdot g^{f(n)}) = n\, 2^{O(f(n))}$

  Άρα όταν χρησιμοποιούμε χώρο $O(\log n)$ δεν μπορούμε να πούμε

  Όταν όμως έχω χώρο $f(n) \geq \log n$, τότε $n\, 2^{O(f(n))} = 2^{O(f(n))}$

- **Θεώρημα Savitch για λογαριθμικό χώρο**

  Το θεώρημα του Savitch μπορεί να επεκταθεί και για λογαριθμικο χώρο, δηλ έχω:

  $$N\mathcal{L} = NSPACE(\log(n)) = SPACE(\log^2(n))$$

Παράδειγμα $A = \{0^k 1^k \mid k \geq 0\} \in L$

Απ: Δεν μπορούμε να οδηγούμαστε ένα από από τα τυπία αποδοχής, γιατί τότε θα είχαμε χρησιμοποιήσει χώρο $O(n)$

Αρα πρέπει να σβήσω χρόνο ακόμη! Μπορώ να το βλέπω ως εξής:

1) Βλέπω αν είναι σωστή μορφή $0^a 1^b$. Αν όχι Reject
2) Καταγράφω # 0 σε έναν μετρητή. Αφού υπήρχαν $O(n)$ μηδενικά, ο μετρητής χρειάζεται χώρο $O(\log n)$
3) Το ίδιο για τα 1's.
4) Συγκρίνω. Αν counter 1 = counter 2 Accept. Αλλιώς reject

Έχω χώρο $O(\log n) + O(\log n) = O(\log n)$

Παράδειγμα $\text{ΔΙΑΔΡΟΜΗ} = \{\langle G, s, t\rangle \mid \exists \text{διαδρομή } s \to t \text{ στο κατευθυνόμενο } G\} \in NL$

! Είχαμε δείξει πως ΔΙΑΔΡΟΜΗ $\in P$ (Sipser σελ 305)

Θα δώσω αλγόριθμο λογαριθμικού χώρου:

1) Σε κάθε βήμα επιλέγω ~~πιθανός~~ διαθέσιμους γείτονες του επόμενου κόμβου της διαδρομής σου, ξεκινώντας από τον s
2) Σε έναν counter κρατάω τον αριθμό των επιλογών σου.
3) Αν ο counter ~~ξεπεράσει~~ φτάσει τον αριθμό $n = |V|$ τότε απορρίπτω
   Αν φτάσουμε στον t τότε ACCEPT

Βλέπουμε πως χρειαζόμαστε έναν μόνο counter $O(\log n) = O(\log(n))$
αφού μαντεύουμε τους κόμβους 1 προς 1.

3.7 Η κλάση NL-complete

- **Ορισμός**

  Μια γλώσσα $B \in NL$-complete αν 1) $B \in NL$
  2) $\forall A \in NL,\ A \leq_L B$

‼ Χρησιμοποιούμε $\leq_L$, δηλαδή αναγωγή πολυωνυμικού χώρου

Η συνάρτηση f δηλαδή που πραγματοποιεί την αναγωγή πρέπει να είναι υπολογίσιμη σε λογαριθμικό χώρο Τ.Μ.

‼‼ Γιατί δεν μπορώ να χρησιμοποιήσω $\leq_p$;

Ίδια επιχειρηματολογία με την σελίδα 19.
Θα δούμε πως $NL \subseteq P \Rightarrow$ οποιοδήποτε πρόβλημα $\in NL$ μπορεί να λυθεί σε πολυωνυμικό χρόνο $\Rightarrow$ μπορώ να βρω $\leq_p$ μεταξύ οποιωνδήποτε 2 προβλημάτων της $NL \Rightarrow$ η $\leq_p$ είναι υπολογιστικά πιο ισχυρή από τις γλώσσες της NL.

Ας δούμε αν ισχύει το "αν $A \leq_L B$ και $B \in L$ τότε $A \in L$"

- **Αν $A \leq_L B$ και $B \in L$ τότε $A \in L$**

Βλέπω μια ανάλογία με το "Αν $A \leq_p B$ και $B \in P$ τότε $A \in P$"
Εκεί υπολογίζω για την w [redacted] το $f(w)$ [redacted] σε πολυωνυμικό χρόνο, και μετά αφού $B \in P$ ελέγχω σε πολυωνυμικό χρόνο αν $f(w) \in B$ και συνεπώς αν $w \in A$.

Εδώ όμως δεν μπορώ να δουλέψω έτσι!

↓
Αν λειτουργήσει έτσι, θα πρέπει να υπολογίσω την $f(w)$ και να την αποθηκεύσω στην work tape → κάτι που ίσως χρειαστούμε χώρο $O(n)$

✓ επίσης ένας καλός χρόνος
άλλα σίγουρα χωρικά τρόπος.

Έστω M_A η Τ.Μ λογαριθμικού χώρου που πρέπει να κατασκευάσω για να δείξω $A \in L$
και Τ.Μ M_B που διαγιγνώσκει την B σε λογαριθμικό χώρο.

Η M_A θα υπολογίζει τα σύμβολα της $f(w)$ ένα προς ένα.
Δεν θα αποθηκεύει εισόδους αλλά θα δίνει στην M_B το επόμενο σύμβολο που αυτή χρειάζεται
Αν η M_B ζητάει επόμενο/προηγούμενο σύμβολο, τότε η M_A υπολογίζει ξανά την $f(w)$ μέχρι εκείνο το σύμβολο, και δεν αποθηκεύει ποτέ προηγούμενα

Άρα η M_A χρειάζεται δύο counter για να ξέρει σε ποιο σύμβολο βρίσκεται η M_B, άρα θέλει $\log(|f(w)|) = \log(n^2)$ χώρο.

- **ΔΙΑΔΡΟΜΗ ∈ NL-complete**

1) **ΔΙΑΔΡΟΜΗ ∈ NL** το δείξαμε, σελ. 18

2) **∀A ∈ NL, A ≤_L ΔΙΑΔΡΟΜΗ**

A ∈ NL άρα χρησιμοποιεί χώρο $O(\log n)$.
Έστω η T.M. που την διαγιγνώσκει, M_A.
Η M_A σε είσοδο w έχει το πολύ $O(n)$ φάσεις → για να τις διαπεραστούν θέλω χώρο $O(\log n)$

Άρα η T.M που υπολογίζει το $\leq_L$ λειτουργεί ακόμα σε λογαριθμικό χώρο γιατί:
- Οι κόμβοι του G μπορούν να αναπαρασταθούν σε logspace, αφού για κάθε πιθανό κόμβο v_i, ελέγχω αν αυτός αποτελεί φάση c_i της M_A

Η αναγωγή λειτουργεί ως εξής:
1) Για κάθε φάση της M_A, προσθέτει στον γράφο G έναν κόμβο
2) Αν από την φάση c_i παω στην c_j σε ένα βήμα τότε $(c_i, c_j) \in E$
3) Αρχικός κόμβος s του G η αρχική φάση και κόμβος t η φάση ACCEPT

Άρα αν υπάρχει $s \to t$ μονοπάτι στο γράφημα, τότε M_A accepts w
Αντίστροφα αν $\nexists$ s-t path στο G, τότε M_A rejects w

Η T.M. που υπολογίζει το $A \leq_L$ ΔΙΑΔΡΟΜΗ δουλεύει σε λογαριθμικό χώρο γιατί:
- Κάθε δυνατός κόμβος του G, v_i, αναπαριστάται από $\log n$ bits
  Ελέγχω για κάθε v_i αν αυτός υπάρχει c_i της M_A επί της w, έρχονται των M_A επί της w βήμα και βήμα..
- Μπορώ να ελέγξω όλους τους δυνατούς συνδυασμούς c_i, c_j ώστε να επαληθεύσω αν $(v_i, v_j) \in E$, με τον ίδιο τρόπο.

- **NL ⊆ P**

- Δείξαμε πως ∀A ∈ NL, $A \leq_L$ ΔΙΑΔΡΟΜΗ
- Όμως δείξαμε πως ΔΙΑΔΡΟΜΗ ∈ P
- Η T.M. που υπολογίζει το $\leq_L$ λειτουργεί σε χώρο $\log(n)$
  Όμως τότε δείξαμε πως λειτουργεί σε χρόνο $2^{O(\log(n))} = 2^{O(n)}$ = πολυωνυμικός

Άρα η T.M που υπολογίζει το $\leq_L$ λειτουργεί σε πολυωνυμικό χρόνο.
Συνεπώς δείξαμε πως ∀A ∈ NL, $A \leq_p$ ΔΙΑΔΡΟΜΗ
Αφού όμως ΔΙΑΔΡΟΜΗ ∈ P, τότε A ∈ P

Άρα NL ⊆ P

3.8 NL = coNL

Αφού ξέρω πως ΔΙΑΔΡΟΜΗ ∈ NL-complete, αρκεί να δείξω πως $\overline{\text{ΔΙΑΔΡΟΜΗ}} \in NL$

Έστω η non-det T.M M που διαγιγνώσκει την $\overline{\text{ΔΙΑΔΡΟΜΗ}}$ σε λογαριθμικό χώρο.

Η M πρέπει να αποδέχεται αν το G δεν έχει διαδρομή $s \to t$

Απόδειξη: (σελ. 387-89, Sipser)

3.9 Θεώρημα Ιεραρχίας

- **Intuition**
  - Περισσότεροι πόροι → περισσότερη ισχύς
  - Άρα μία T.M πρέπει να μπορεί να διαγιγνώσκει περισσότερες γλώσσες σε χρόνο (αντίστοιχα χώρο) n^3 παρά σε n
  - Ισχύει;
- **Ιεραρχία Χώρου**

  Έστω $f: N \to N$ συνάρτηση τ.ω $f(n) \geq \log n$ και χωρικά κατασκευάσιμη

  Υπάρχει γλώσσα που διαγιγνώσκεται σε χώρο $O(f(n))$ αλλά όχι σε $o(f(n))$,

  δηλαδή $SPACE(o(f(n))) \subset SPACE(f(n))$
- **Ιεραρχία Χρόνου**

  Έστω $t: N \to N$ συνάρτηση τ.ω. $t(n) \geq O(n \log n)$ και χρονικά κατασκευάσιμη

  Υπάρχει γλώσσα που διαγιγνώσκεται σε χρόνο $O(t(n))$ αλλά όχι σε $o(t(n)/\log(t(n)))$

  δηλαδή $TIME(o(t(n)/\log t(n))) \subset TIME(t(n))$

Επίσης μπορούν να αποδειχτούν τα ακόλουθα:

- $SPACE(n^{\varepsilon_1}) \subset SPACE(n^{\varepsilon_2})$ για $0 \leq \varepsilon_1 < \varepsilon_2$, $\varepsilon_1, \varepsilon_2 \in \mathbb{R}$
- $TIME(n^{\varepsilon_1}) \subset TIME(n^{\varepsilon_2})$ για $1 \leq \varepsilon_1 < \varepsilon_2$, $\varepsilon_1, \varepsilon_2 \in \mathbb{R}$

4 Ιεραρχία

- **NL ⊂ PSPACE** Από Savitch $NL \subseteq SPACE(\log^2 n)$
  Από Θ.Ιεραρχίας Χωρών $SPACE(\log^2 n) \subset SPACE(n)$
  Άρα $NL \subset PSPACE$

- **PSPACE ⊂ EXPSPACE**

  $SPACE(n^k) \subset SPACE(2^{\log n})$, όπου για αρκετά μεγάλα n $\quad k \leq 2^{\log n}$

  $SPACE(n^{\log n}) \subset SPACE(2^n)$

  Άρα $PSPACE \subset EXPSPACE$

  ! Απόδειξη του ότι υπάρχουν προβλήματα τα οποία είναι δυσεπίλυτα
  Αν και είναι τεχνική

- **P ⊂ EXPTIME**

  Με τον ίδιο τρόπο.

- **Ιεραρχία**

  Ξέραμε πως:

$$L \subseteq NL = coNL \subseteq P \subseteq NP \subseteq PSPACE = NPSPACE \subseteq EXPTIME$$

Όμως είδαμε πως $P \subset EXPTIME$
άρα κάποιος τουλάχιστον εκ των παραπάνω πρέπει να είναι γνήσιος.

Η σωστή σειρά είναι ότι όλοι οι εγκλεισμοί είναι γνήσιοι: