ΑΛΓΟΡΙΘΜΟΙ ξ ΠΟΛΥΠΛΟΚΟΤΗΤΑ

«Σχέσεις μεταξύ Κλάσεων Πολυπλοκότητας»

ΨΑΡΡΗ ΔΗΜΗΤΡΑ

Συνάρτηση Πολυπλοκότητας

Μια συνάρτηση f:N→N θεωρείται κατάλληλη συνάρτηση πολυπλοκότητας, αν είναι αύξουσα και ισχύει :

- Υπάρχει μια k-strings (ή k ταινιών) TM M=(K,Σ,δ,s) με input x μεγέθους n και output:

$(s, \triangleright, x, \triangleright, \in, \ldots, \triangleright, \in) \to M^t\ (h, x, \triangleright, \square j^2, \ldots, \triangleright, \square j^{k-1}, \triangleright, \Pi^{f(|x|)})$:

$t=O(n+f(n))$ και $j^i=O(f(|x|))$ για $i=2,\ldots,k-1$

Δηλ. η M τερματίζει σε $O(|x|+f(|x|))$ steps και χρησιμοποιεί $O(f(|x|))$ space

Μορφές συνάρτησης

$f(n)=c$, $f(n)=n$, $f(n)=\lceil \log n \rceil \ldots$

Βασικές Κλάσεις Πολυπλοκότητας

- P= $\cup_{k>0}$TIME(n^k)
- NP= $\cup_{k>0}$ NTIME(n^k)
- PSPACE= $\cup_{k>0}$ SPACE(n^k)
- NPSPACE= $\cup_{k>0}$ NSPACE(n^k)
- EXP= $\cup_{k>0}$ TIME(2^n) ^{k}
- L= SPACE(logn)
- NL= NSPACE(logn)

Λήμμα 1: H $\in$ TIME($(f(n))^3$)

H = { M;x : M αποδέχεται για input x μετά από το πολύ $f(|x|)$ steps }

Έστω U μια TM με 4 strings που αποφασίζει την H σε χρόνο $(f(n))^3$ που λειτουργεί ως εξής:

<u>1ο Στάδιο:</u>

• Αρχικοποιεί το **<u>4ο str.</u>** με ένα "alarm clock" $\Pi^{f(|x|)}$ που χρησιμ. στην προσομ. της Μ(η οποία λειτουργεί σε χρόνο $O((f|x|))$

•Αντιγράφει την περιγραφή της Μ στο **<u>3ο str.</u>**

•Μετατρέπει το x στην κωδικοποίηση $\triangleright$x στο **<u>1ο str.</u>**

•Αρχικοποιεί το **<u>2ο str.</u>** με την κωδικοποίηση της αρχικής κατάστασης **s**

•Ελέγχει αν η περιγραφή της Μ είναι μια γνήσια TM και **<u>αποδέχεται</u>** αν είναι, αλλιώς **<u>απορρίπτει</u>**

- ➢Αυτό απαιτεί γραμμικό χρόνο, χρησιμ. το 4ο και το 1ο str., ο συνολικός χρόνος είναι $O((f|x|) + n) = O(f(n))$

<u>2ο Στάδιο:</u>

•Προσομοιώνει τη Μ για input x στο 1ο str. (όπου βρίσκονται όλες οι κωδικοποιήσεις των περιεχομένων των strings της Μ), με δύο scans στο 1ο str. της U

- **Το 1ο scan**:
  - ➢Η U συλλέγει όλες τις σχετικές πληροφορίες με τα σύμβολα της M και τις γράφει στο 2ο str. που περιέχει και την τρέχουσα κατάσταση
  - ➢Συσχετίζει τα περιεχόμενα 2ου και 3ου str.(περιγραφή της M) για να βρει την κατάλληλη μετάβαση της M
- **Το 2ο scan**:
  - ➢Η U εκτελεί τις κατάλληλες αλλαγές στο 1ο str. και προχωράει κατά 1 στο “alarm clock”

-Συνολικός χρόνος προσομοίωσης $O(f^2(n))$ και για τα δύο scans

-Η U για είσοδο <M;x>

  - ➢Αποδέχεται : αν η M αποδέχεται για είσοδο x σε $f(|x|)$ steps
  - ➢Απορρίπτει: αν η M απορρίπτει για είσοδο x ή αν το “alarm clock” ξεπεραστεί

<u>Επομένως:</u> η U πάντα **τερματίζει** και ο **συνολικός χρόνος** (από 1ο στάδιο $O(f(n))$ και από 2ο στάδιο $O(f^2(n))$) θα είναι **$O(f(n)^3)$**.

Λήμμα 2: $H \notin TIME(f\lfloor n/2 \rfloor)$

- M αποφασίζει H σε χρόνο $f(\lfloor n/2 \rfloor)$

- Κατασκευάζω μηχανή D (M):
  - Αποδέχεται αν η M (<M;<M>)='no'
  - Απορρίπτει αν η M (<M;<M>)= 'yes'
- η D για είσοδο <M> εκτελείται

  και M (M;<M>) εκτελείται επίσης σε χρόνο $f(\lfloor 2n+1/2 \rfloor)=f(n)$
- Αν D (<D>)='yes' τότε M (D;<D>)='no' $\Rightarrow$ D(<D>) $\notin$ H

  Άρα η **D αποτυχαίνει να αποδεχτεί την <D>**, δηλ. την περιγραφή της σε f(n) steps, και επομένως D(<D>)='no'
- Αν D(<D>)='no' $\Rightarrow$ D (D)='yes'

  Δηλ. και στις δύο περιπτώσεις η D έχει τιμή **αντίθετη του εαυτού της**, πράγμα **άτοπο** και επομένως δεν μπορεί να ισχύει η υπόθεση: H $\in$ TIME($f\lfloor n/2 \rfloor$)

Συμπεράσματα από τα σχετικά Λήμματα

1) $H \in \mathrm{TIME}((f(n))^3)$

2) $H \notin \mathrm{TIME}(f(\lfloor(n/2 \rfloor)$

Συγκρίνοντας τα συμπεράσματα προκύπτουν:

1) The Time Hierarchy Theorem: Αν $f(n) \geq n$, τότε θα ισχύει:

$$\mathbf{TIME(f(n)) \subset TIME((f(2n+1))^3)}$$

Σημαντικό χαρακτηριστικό: η $f(2n+1))^3$ είναι ένα πολυώνυμο, οποτεδήποτε η f(n) είναι πολυώνυμο. Από αυτό προκύπτει: $\mathbf{P \subset EXP}$

<u>Απόδειξη:</u>

Κάθε πολυώνυμο θα γίνεται μικρότερο του $2^n \Rightarrow$

$P \subseteq \mathrm{TIME}(2^n)$ και $P \subseteq \mathrm{EXP}$ -Για να αποδείξουμε γνήσιο εγκλεισμό:

Αλλά $\mathrm{TIME}(2^n) \subset \mathrm{TIME}((2^{2n+1})^3) \subseteq \mathrm{TIME}(2^{(n)2}) \subseteq \mathrm{EXP}$ από το θεώρημα

Και επειδή $P \subseteq \mathrm{TIME}(2^n) \Rightarrow P \subset \mathrm{EXP}$

2) **The Space Hierarchy Theorem:** Αν f(n) μια κατάλληλη συνάρτηση, τότε ισχύει:

$$\mathbf{SPACE(f(n)) \subset SPACE(f(n)logf(n))}$$

Αποδεικνύεται με παρόμοιο τρόπο με το προηγούμενο

Θεώρημα Gap

Υπάρχει μία αναδρομική συνάρτηση f(n) :$N \rightarrow N$, τέτοια ώστε να ισχύει

TIME(f(n)) = TIME($2^{f(n)}$). Αν f(n) δεν είναι κατάλληλη, απρόβλεπτα αποτελέσματα.

<u>Απόδειξη:</u>

- Ορίζουμε την f με τέτοιο τρόπο ώστε καμία TM, input length n, να μην τερματίζει με αριθμό βημάτων μεταξύ f(n) και $2^{f(n)}$

  - Έστω όλες TM $M_0,\ldots,M_i$, ορίζονται με το παραπάνω χαρακτηριστικό ή δεν τερματίζουν ποτέ. Ακόμα και έτσι μπορούν να αποφασιστούν αν προσομοιωθούν για αρ. βημάτων πάνω από 2^k+1. Χαρακτηριστικό P(i,k) : για input length l, halts in fewer k steps or after than 2^k steps.

  - $K_1=2i$, $j=2,3,\ldots,k_j=2^{k_{j-1}}+1$, ορίζουμε f(i), $i \geq 0$, αν $P(i,k_j)$ false τότε πρέπει να υπάρχει ένας ακέραιος $l \leq N(i)$ έτσι ώστε $P(i,k_l)$ true. H f(i) : $f(i)=k_l$

- Έστω τώρα μια **<u>L $\in$ TIME($2^{f(n)}$)</u>**, αποφασίσιμη από Mj, in time $2^{f(n)}$, τότε:

  - για input x , $|x| \geq j$, τότε Mj αδύνατον να τερματίζει ανάμεσα σε f($|x|$) και $2^{f(x)}$ steps, λόγω κατασκευής της Mj και των τιμών της $n \geq j$

  - για input x, $|x| < j$, δεν ξέρουμε πότε τερματίζει η Mj, αλλά μπορούμε να τροποποιήσουμε την Mj στην Mj', επαυξάνοντας τις καταστάσεις της ώστε να αποφασίζει για όλα τα input in time < 2i. Αυτό οδηγεί στο ότι η **<u>L $\in$ TIME(f(n))</u>**

Θεωρήματα Ιεραρχίας

- Πως οι κλάσεις του ίδιου είδους (DTM time & DTM space) συσχετίζονται μεταξύ τους, όταν μεταβάλλεται η συνάρτηση πολυπλοκότητας
- Τα ίδια αποτελέσματα ισχύουν και για NDT classes αλλά αποδεικνύονται πιο δύσκολα
- Πιο ενδιαφέρον όμως η σχέση μεταξύ διαφορετικών κλάσεων (όπως P εναντίον NP)

Τι ισχύει:

1) SPACE(f(n) $\subseteq$ NSPACE(f(n) και TIME(f(n) $\subseteq$ NTIME(f(n)

<u>Επειδή:</u> Κάθε DTM TM είναι επίσης NDT (με μια επιλογή για κάθε βήμα) και έτσι κάθε γλώσσα στο SPACE(f(n) είναι και στο NSPACE(f(n). Παρόμοια για TIME(f(n) και NTIME(f(n)

2) NTIME(f(n)) $\subseteq$ SPACE(f(n))

Απόδειξη:

-Έστω L $\in$ NTIME(f(n)), υπάρχει NDT TM decides L in time f(n). Θα σχεδιάσουμε DTM M' decides L in space f(n).

▪M' δημιουργεί τις ND choices της M, (0…d-1) choices, d=max numb για κάθε (state,symbol) συνδυασμό της M

▪M' προσομοιώνει λειτουργία της M σε space f(n) (σε time f(n) μόνο O(f(n)) χαρακτήρες γράφονται)

▪M' δημιουργεί εκθετικά πολλές προσομοιώσεις για να βρει ποια αποδέχεται και τις εκτελεί μία προς μία (**erase previous-reuse space**)

▪M' ίχνος ακολουθίας τρεχ.προσομοιώσεων + δημιουργία νέας = space O(f(n))

3) NSPACE(f(n)) $\subseteq$ TIME($k^{\log n+f(n)}$)

<u>Απόδειξη:</u> Reachability Method for simulating space-bounded machines

- Έστω k_string NDT M, (input,output), decides L in space f(n)
  - Κάθε configuration ένα στιγμιότυπο ενός υπολογισμού της M για input x:

    (q,i,w2,u2,…w_{k-1},u_{k-1}), πλήθος 2k-2, state+head positions για τα k strings,

    0 ≤ I ≤ n=|x| position first cursor on input string($\triangleright$x)
  - Σύνολο configurations: K επιλογές(state), n+1 επιλογές(i), $|\Sigma|^{(2k-2)f(n)}$ επιλογές(όλα τα εναπομείναντα strings) $\rightarrow$ $nc_1^{f(n)} = c_1^{\log n+f(n)}$
- Ορίζουμε G(M,x), κόμβοι=configurations και κάθε ακμή μεταξύ 2 configurations C1 και C2

  ανν C1 $\rightarrow^{M}$ C2, και <u>x$\in$L ανν path C0=(s,0, $\triangleright$, ϵ,...,ϵ) $\rightarrow$ C=(‘yes’,I,…)</u>
  - Δηλ. έχουμε αναγάγει το πρόβλημα απόφασης αν x$\in$L, σε πρόβλημα REACHABILITY ενός γράφου με $c_1^{\log n+f(n)}$ κόμβους, που έχει αλγόριθμο πολυωνυμικό (με bound $c=c_2n^2$) $\rightarrow$ $c^{\log n+f(n)}$
- Δύο τρόποι υλοποίησης αλγόριθμου:

1) Adjacency matrix γράφου G(M,x) και εκτελείται ο αλγόριθμος
2) Αποφασίζει αν (C,C’) ακμή του G(M,x) αν C’ παράγεται από C

➪ P $\subseteq$ NP αλλά NP $\subseteq$ PSPACE και L $\subseteq$ NL $\subseteq$ P $\Rightarrow$

L $\subseteq$ NL $\subseteq$ P $\subseteq$ NP $\subseteq$ PSPACE

- Από (3) άμεσα συνάγεται:

$$\mathbf{NSPACE(f(n)) \subseteq SPACE(k^{logn+f(n)})}$$

- Από την προσομοίωση NDT space, από εκθετικό DT space, είναι το καλύτερο αποτέλεσμα;
- Θα αποδείξουμε ότι η προσομοίωση DT space μπορεί να είναι τετραγωνική:
  - ❖ χρήση REACHABILITY method(middle first search), επειδή breadth first & depth first: space n
  - ❖ Θεώρημα Savitch: REACHABILITY SPACE($\log^2$)
  - ❖ Συνέπεια θεωρήματος Savitch: NSPACE(f(n)) $\subseteq$ SPACE(f^2(n)), όταν f(n) $\geq$ logn

Θεώρημα Savitch: REACHABILITY SPACE($\log^2 n$)

- G graph, n nodes, PATH(x,y,i) ισχύει αν $\exists$ path (x,y) length 2^i, και εφόσον any path in G μπορεί να είναι **το πολύ** n $\Rightarrow$ PATH(x,y, $\lceil \log n \rceil$)
- Σχεδιάζουμε M, 2 working strings και 1 input, decides PATH(x,y,i)
  - Adjacency matrix του G δίνεται στο input str.
  - Πρώτο work. Str. περιέχει διάφορες τριάδες (x,y,i) και το δεύτερο scratch space
  - work str: O(logn) χώρου
- M αποφασίζει για PATH(x,y,i):
  - Αν i=0, $2^0=1 \Rightarrow$ x=y ή x,y adjacent at the input
  - Αν i$\geq$ 1, υπολογίζω PATH(x,y,i) από αλγόριθμο:
    - for all nodes z test whether PATH(x,z,i-1) and PATH(z,y,i-1)
    - Κάθε μονοπάτι 2^i από x στο y έχει ένα Z midpoint, x και y το πολύ 2^{i-1} από z
    - Generate all z, reuse space, add new (x,z,i-1) to frst str, recursively
      - Negative answer P(x,z,i-1), erase this, move next z
      - Positive answer P(x,z,i-1), erase this, write (z,y,i-1)
      - Negative P(z,y,i-1), erase this, move next z
      - Positive P(z,y,i-1) compare with P(x,y,i) 2nd recursive call $\Rightarrow$

$\Rightarrow$επιστρέφει positive answer στο PATH(x,y,i)

Το work str. περιέχει $\lceil \log n \rceil$ τριάδες και κάθε μια μεγέθους το πολύ 3logn. Άρα η μηχανή λειτουργεί σε χώρο $O(\log^2 n)$.

Τώρα θα χρησιμοποιήσουμε θεώρημα του Savitch για να αποδείξουμε:

$NSPACE(f(n)) \subseteq SPACE(f^2(n))$, όταν $f(n) \geq \log n$

<u>Απόδειξη:</u>

- Προσομοιώνουμε NDT f(n)-space bounded M, input x, $|x|=n$
  - Τρέχουμε τον προηγούμενο αλγόριθμο σε ένα configuration fraph G της M με input x
  - Ελέγχει αν 2 κόμβοι συνδέονται, εξετάζοντας το input και transition function της M
  - Επειδή $f(n) \geq \log n$, ο G έχει $c^{f(n)}$ κόμβους, επομένως χρειάζεται **$O(f^2(n))$ space**
- Επειδή όμως το τετράγωνο του πολυωνύμου είναι επίσης πολυώνυμο

$\Rightarrow$ **PSPACE = NPSPACE**

Θεώρημα Immerman-Szelepscènyi: G graph και ένας κόμβος x, ο αριθμός των κόμβων που προσπελάζονται από το x, μπορεί να υπολογιστεί από NDT M σε **space logn**

Περιγραφή αλγόριθμου που εκτελεί η M:

1)Υπολογίζουμε |S(1)|, |S(2)|,..., |S(n1)|, όπου S(k) set nodes of G, reached from x in path ≤ k

|S(0)|=1;

compute |S(k)|

For k:=1,2,...,n-1 do: from |S(k-1)|;

2)Εξετάζει κόμβους με αριθμητική σειρά, reusing space, αν ανήκουν στο S(k):

l:=0;

if u $\in$ S(k)

for u:=1,2,…,n do: then l:=l+1;

3)Πως αποφασίζει αν u $\in$ S(k) :

m:=0; reply:=false;

f epeat:

if v $\in$ S(k-1)

then m:=m+1;

if G(v,u) then reply:=true;

if in the end m< |S(k-1)| then “no” (give up) else return reply;

4)

4)Πως αποφασίζει αν v $\in$ S(k-1): ND ξεκινά από x και μαντεύει k-1nodes

w_0:=x;

for i:=1,...,k-1

guess a node w_i ;

if G(w_{i-1}, w_i) then reply:=true

else give up;

end_for;

if w_{k-1} = v then report v $\in$ S(k-1) else give up;

➢Ο αλγόριθμος κάθε φορά επαναχρησιμοποιεί το χώρο και υλοποιείται από M logn space-bounded

- 1o string: variables k, S(k-1), l, u, m, v, p, w_p, w_{p-1}
- 2o string: input
- 3o string: output

Όλοι οι ακέραιοι αυξάνονται κατά 1, συγκρίνονται μεταξύ τους και με nodes in input. Όλοι είναι bounded n $\Rightarrow$ logn space

Θα δείξουμε ότι ο αλγόριθμος είναι σωστός, δηλ. ότι σωστά υπολογίζει το $|S(k)|$, με επαγωγή:

1) Για k=0 ισχύει

2) Υποθέτουμε ότι υπολογίζει σωστά το $|S(k\text{-}1)|$

3) Θα αποδείξουμε ότι το ίδιο ισχύει και για το $|S(k)|$. Αρκεί να δείξουμε ότι αποφασίζει σωστά αν $u \in S(k)$ σε όλα τα υπολογιστικά μονοπάτια, που επιστρέφουν αποτέλεσμα

- Έστω ότι $u \in S(k)$ και v η προτελευταία κορυφή σε μονοπάτι μήκους k από τη x στη u

  i) **Ένα τουλάχιστον** υπολογιστικό μονοπάτι θα σχηματίσει το μονοπάτι k-1 από τη x στη v και θα διαπιστώσει ότι $u \in S(k)$, επιστρέφει reply=true, και τότε μόνο l:=l+1

  ii) **Κάθε υπολογιστικό** μονοπάτι που θα έχει αποτύχει να σχηματίσει ένα μονοπάτι k-1 από τη x στη v και θα έχει βρει τιμή $m < |S(k\text{-}1)|$, επιστρέφει reply=false, και give up

  Από i) και ii) προκύπτει ότι υπάρχει μονοπάτι μήκους k από τη x στη u και άρα αποφασίζει σωστά ότι $u \in S(k)$ και επομένως υπολογίζει σωστά το $|S(k)|$

- Αν αντίθετα $u \notin S(k)$ καμία ακολουθία μη ντετερμινιστικών επιλογών δεν

Θα χρησιμοποιήσουμε το θεώρημα για να αποδείξουμε ότι αν $f \geq \log n$ μια κατάλληλη συνάρτηση πολυπλοκότητας, τότε **NSPACE(f(n)) = coNSPACE(f(n))**

Απόδειξη:

Έστω $L \in$ NSPACE(f(n)), αποφασίζεται από NDT M που είναι space-bounded f(n). Θα δείξουμε ότι υπάρχει μια f(n) space-bounded NDT ¬M, που αποφασίζει τη ¬ L.

- Για input x η ¬M εκτελεί τον αλγόριθμο στο configuration graph της M
- Ο αλγόριθμος αποφασίζει αν 2 configurations συνδέονται και η ¬M αποφασίζει με βάση το x και την transition function της M:
  - ¬M απορρίπτει αν βρει μία accepting configuration $u \in S(k)$, $\forall$ k
  - ¬M αποδέχεται αν υπολογίζει το $|S(n-1)|$ και δε βρει καμία accepting configuration

<u>Θ.ΙΕΡ.ΧΡ</u>: $TIME(f(n)) \subset TIME(f(2n+1)^3)$	1΄)L $\subseteq$NL (από 2)
<u>Θ.ΙΕΡ.ΧΩΡ</u>: $SPACE(f(n)) \subset SPACE(f(n)\log f(n))$	2΄)NL $\subseteq$ P (από 7)
1) $TIME(f(n)) \subseteq NTIME(f(n))$	3΄)P $\subseteq$ NP (από 1)
2) $SPACE(f(n)) \subseteq NSPACE(f(n))$	4΄)NP $\subseteq$ PSPACE (από 5)
3) $TIME(f(n)) = coTIME(f(n))$	5΄)PSPACE =NPSPACE (από 8)
4) $SPACE(f(n)) = coSPACE(f(n))$	6΄)L $\subseteq$ NL $\subseteq$ P $\subseteq$ NP $\subseteq$ PSPACE=NPSPACE
5) $NTIME(f(n)) \subseteq SPACE(f(n))$	7΄)PSPACE $\subseteq$ EXP (από 8)
7) $NSPACE(f(n)) \subseteq TIME(k^{\log n+f(n)})$	8΄)P = coP (από 3)
8) $NSPACE(f(n)) \subseteq SPACE(f^2(n))$, $f(n) \geq \log n$, Savitch	9΄)PSPACE = coPSPACE (από 4)
9) $NSPACE(f(n)) = coNSPACE(f(n))$, $f(n) \geq \log n$, Immerman-Szelepscènyi	10΄)NPSPACE = coNPSPACE (από 9)
10) NTIME(f(n)) ? coNTIME(f(n)), Ανοικτό πρόβλημα	11΄)NL = coNL (από 9)
	12΄)P $\subset$ EXP (από Θ.ΙΕΡ.ΧΡ.)
	13΄)NL $\subset$ PSPACE (από Θ.ΙΕΡ.ΧΩΡ. + 8)

ΙΕΡΑΡΧΙΑ ΚΛΑΣΕΩΝ