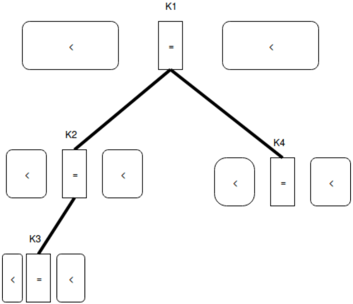
2η Σειρά Γραπτών Ασκήσεων

Αλγόριθμοι και Πολυπλοκότητα

ΣΗΜΜΥ, Εθνικό Μετσόβιο Πολυτεχνείο

Κλειδιά και κλειδαριές

- Θα χρησιμοποιήσουμε τα κλειδιά ως pivot για να ταξινομήσουμε τις πόρτες μας



Κλειδιά και κλειδαριές

- T=μέσος ολικός χρόνος

Κλειδιά και κλειδαριές

- T=μέσος ολικός χρόνος
- F=μέσος ολικός χρόνος εύρεσης συνόλων που διαμερίζονται

Κλειδιά και κλειδαριές

- T=μέσος ολικός χρόνος
- F=μέσος ολικός χρόνος εύρεσης συνόλων που διαμερίζονται
- D=μέσος ολικός χρόνος διαμερισμού

Κλειδιά και κλειδαριές

- T=μέσος ολικός χρόνος
- F=μέσος ολικός χρόνος εύρεσης συνόλων που διαμερίζονται
- D=μέσος ολικός χρόνος διαμερισμού
- $D = \Theta(nlogn)$ (Quicksort analysis)

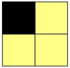
Κλειδιά και κλειδαριές

- T=μέσος ολικός χρόνος
- F=μέσος ολικός χρόνος εύρεσης συνόλων που διαμερίζονται
- D=μέσος ολικός χρόνος διαμερισμού
- $D = \Theta(nlogn)$ (Quicksort analysis)
- $F \leq nE[depth] = \Theta(nlogn)$

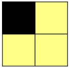
Κλειδιά και κλειδαριές

- T=μέσος ολικός χρόνος
- F=μέσος ολικός χρόνος εύρεσης συνόλων που διαμερίζονται
- D=μέσος ολικός χρόνος διαμερισμού
- D = $\Theta(nlogn)$ (Quicksort analysis)
- F $\leq nE[depth] = \Theta(nlogn)$
- $T = \Theta(nlogn)$

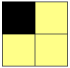
Puzzle

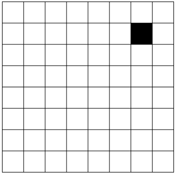
- E.B: n=1 

Puzzle

- E.B: n=1 
- E.Y: έστω ότι μπορώ σε $2^{n-1} \times 2^{n-1} \quad n \geq 2$

Puzzle

- E.B: n=1 
- E.Y: έστω ότι μπορώ σε $2^{n-1} \times 2^{n-1} \quad n \geq 2$
- Απόδειξη για $2^n \times 2^n \quad n \geq 2$



Puzzle

1. χωρίζω σε τέσσερα τετράγωνα μεγέθους $2^{n-1} \times 2^{n-1}$

Puzzle

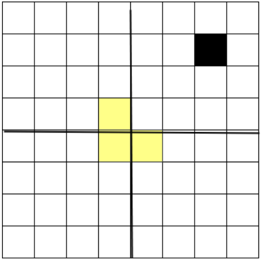
1. χωρίζω σε τέσσερα τετράγωνα μεγέθους $2^{n-1} \times 2^{n-1}$
2. γεμίζω κατάλληλα με ένα γάμμα το κεντρικό τετραγωνάκι

Puzzle

1. χωρίζω σε τέσσερα τετράγωνα μεγέθους $2^{n-1} \times 2^{n-1}$
2. γεμίζω κατάλληλα με ένα γάμμα το κεντρικό τετραγωνάκι
3. γεμίζω επαγωγικά το καθένα ξεχωριστά

Puzzle

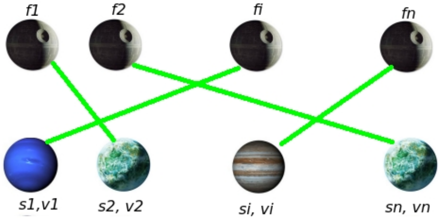
1. χωρίζω σε τέσσερα τετράγωνα μεγέθους $2^{n-1} \times 2^{n-1}$
2. γεμίζω κατάλληλα με ένα γάμμα το κεντρικό τετραγωνάκι
3. γεμίζω επαγωγικά το καθένα ξεχωριστά

- 

Είσοδος: n Death Satellites δύναμης f_i
n πλανήτες αξίας v_i με δυναμη ασπίδας s_i

Εξοδος: Ένα ταίριασμα M των Satellites με πλανητες που μεγιστοποιεί το κέρδος της Αυτοκρατορίας:

$$max \sum_{i \in W} v_i, \;\; W = \{i \,|\, f_i > s_{M(i)}\}$$



Διαστημικές Μάχες - Αλγόριθμος

Greedy κριτήριο 1

Προσπάθησε να κατάστρεφεις πλανήτες με μεγάλη αξία

Greedy κριτήριο 2

Μη σπαταλάς μεγάλης δύναμης Satellites σε μικρής δύναμης ασπίδες

το μέγιστο δυνατό κέρδος με τη μικρότερη δυνατή "σπατάλη" δύναμης

Διαστημικές Μάχες - Αλγόριθμος

Greedy κριτήριο 1

Προσπάθησε να κατάστρεφεις πλανήτες με μεγάλη αξία

Greedy κριτήριο 2

Μη σπαταλάς μεγάλης δύναμης Satellites σε μικρής δύναμης ασπίδες

το μέγιστο δυνατό κέρδος με τη μικρότερη δυνατή "σπατάλη" δύναμης

Αλγόριθμος: Space

Ταξινόμησε τους πλανήτες σε φθίνουσα σειρά v_i
Για κάθε πλανήτη i αντιστοίχισε το Satellite με min f_j ώστε $f_j > s_i$
Αν $\nexists$ τέτοιο Satellite, αντιστοιχίζω min f_j

Έστω OPT η βέλτιστη λύση του προβλήματος, και Greedy η δική μας λύση. Εξετάζω τους πλανήτες σε φθίνουσα σειρά v_i, και η πρώτη φορά που διαφέρουν οι λύσεις είναι στον πλανήτη i: η OPT αντιστοιχίζει το Satellite f_{op} ενώ η Greedy το Satellite f_{gr}.
Διακρίνω τις περιπτώσεις:

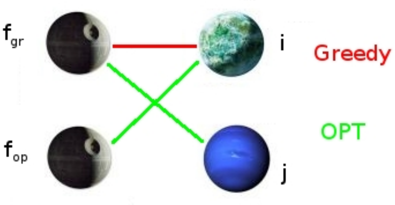
Διαστημικές Μάχες - Ορθότητα

Έστω OPT η βέλτιστη λύση του προβλήματος, και Greedy η δική μας λύση. Εξετάζω τους πλανήτες σε φθίνουσα σειρά v_i, και η πρώτη φορά που διαφέρουν οι λύσεις είναι στον πλανήτη i: η OPT αντιστοιχίζει το Satellite f_{op} ενώ η Greedy το Satellite f_{gr}.
Διακρίνω τις περιπτώσεις:

- Greedy χάνει, OPT κερδίζει: ($f_{gr} \leq s_i < f_{op}$) Άτοπο! ο greedy θα είχε διαλέξει f ώστε να κερδίζει
- Greedy κερδίζει, OPT χάνει: ($f_{gr} > s_i \geq f_{op}$) Μπορώ να αλλάξω το f_{op} με το f_{gr} στην OPT χωρίς να μειωθεί το κέρδος

Διαστημικές Μάχες - Ορθότητα

- Greedy κερδίζει, OPT κερδίζει: 2 περιπτώσεις
  - $f_{gr} > f_{op} > s_i$ Άτοπο!
    (ο greedy διαλέγει το min f_i ώστε να κερδίζει)
  - $f_{op} > f_{gr} > s_i$



Μπορώ να κάνω swap τα f_{op} και f_{gr} στην OPT χωρίς να χάσω αξία!

Πολυπλοκότητα;

- Ταξινόμηση πλανητών: $O(n \log n)$
- Αναζήτηση σωστού s_j για κάθε f_i, ταξινόμηση ως προς s_j;

Διαστημικές Μάχες - Πολυπλοκότητα

Πολυπλοκότητα;

- Ταξινόμηση πλανητών: $O(n \log n)$
- Αναζήτηση σωστού s_j για κάθε f_i, ταξινόμηση ως προς s_j; Δεν αρκεί! Οι πλανήτες που αντιστοιχίζονται χαλάνε την αναζήτηση, θέλω δομή για ταξινόμηση με αφαίρεση στοιχείων$\rightarrow$ δέντρο δυαδικής αναζήτησης $O(\log n)$

Τελικά $O(n \log n)$

Ευθύγραμμό τμήμα με σπίτια x_i
κεραίες με εμβέλεια k

Θέλω min # κεραιών
Ιδέα 1: Κάθε σπίτι πρέπει να καλύπτεται $\rightarrow$ 1 κεραία σε κάθε σπίτι

Κεραίες i)

Ευθύγραμμό τμήμα με σπίτια x_i
κεραίες με εμβέλεια k

Θέλω min # κεραιών
Ιδέα 1: Κάθε σπίτι πρέπει να καλύπτεται $\rightarrow$ 1 κεραία σε κάθε σπίτι
Ιδέα 2: (Greedy) κάλυψε όσα περισσότερα μπορείς βάζοντας την κεραία στο δεξιότερο δυνατό σημείο

Κεραίες i)

Ευθύγραμμό τμήμα με σπίτια x_i
κεραίες με εμβέλεια k

Θέλω min # κεραιών

Ιδέα 1: Κάθε σπίτι πρέπει να καλύπτεται → 1 κεραία σε κάθε σπίτι

Ιδέα 2: (Greedy) κάλυψε όσα περισσότερα μπορείς βάζοντας την κεραία στο δεξιότερο δυνατό σημείο

Αλγόριθμος:

Σύνολο ακάλυπτων σπιτιών: S

while $S \neq \emptyset$ **do**

  Ξεκίνα από το πρώτο ακάλυπτο σπίτι x

  Βάλε κεραία όσο δεξιότερα μπορείς (ώστε x να καλύπτεται)

  Βγάλε απο το S τα σπίτια στα επόμενα k μέτρα

end

Πολυπλοκότητα: $O(n)$

Κεραίες i) Απόδειξη

$$\left.\begin{array}{l} \text{Λύση } OPT = \{p_1, \ldots, p_n\} \text{ μεγέθους } |OPT| = n \\ \text{Λύση } ALG = \{a_1, \ldots, a_m\} \text{ μεγέθους } |ALG| = m \end{array}\right\} \text{Θ.δ.ο. } m = n$$

Αρκεί η ALG να είναι πάντα "μπροστά" από την OPT: $a_i \geq p_i \; \forall i$
Επαγωγή στον αριθμό των κεραιών i.

Κεραίες i) Απόδειξη

$$\left.\begin{array}{l}\text{Λύση } OPT = \{p_1, \ \ldots, \ p_n\} \text{ μεγέθους } |OPT| = n \\ \text{Λύση } ALG = \{a_1, \ \ldots, \ a_m\} \text{ μεγέθους } |ALG| = m\end{array}\right\} \text{Θ.δ.ο. } m = n$$

Αρκεί η ALG να είναι πάντα "μπροστά" από την OPT: $a_i \geq p_i \ \forall i$
Επαγωγή στον αριθμό των κεραιών i.

- Βάση: $i = 1$, θα μπει 1 κεραία, $a_1 \geq p_1$ από αλγόριθμο

Κεραίες i) Απόδειξη

$$\left.\begin{array}{l}\text{Λύση } OPT = \{p_1, \ \ldots, \ p_n\} \text{ μεγέθους } |OPT| = n \\ \text{Λύση } ALG = \{a_1, \ \ldots, \ a_m\} \text{ μεγέθους } |ALG| = m\end{array}\right\} \text{Θ.δ.ο. } m = n$$

Αρκεί η ALG να είναι πάντα "μπροστά" από την OPT: $a_i \geq p_i \ \forall i$
Επαγωγή στον αριθμό των κεραιών i.

- Βάση: $i = 1$, θα μπει 1 κεραία, $a_1 \geq p_1$ από αλγόριθμο
- Έστω ότι ισχύει για n, θ.δ.ο. για $n + 1$
  Επ. υπόθεση $\Rightarrow$ οι πρώτες n κεραίες του ALG θα καλύπτουν τα σπίτια που καλύπτουν οι πρώτες n του OPT
  Αν βάλω την p_{n+1} στην ALG θα καλύπτει αναγκαστικά τα σπίτια στο $(a_n, \ p_{n+1})$
  Από αλγόριθμο $p_{i+1} \leq a_{i+1}$

Κεραίες ii)

Σπίτια στην περιφέρεια κύκλου

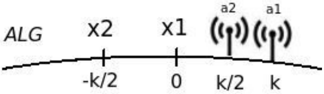
Δουλεύει η λύση του i) ξεκινώντας από τυχαίο σπίτι;

Κεραίες ii)

Σπίτια στην περιφέρεια κύκλου

Δουλεύει η λύση του i) ξεκινώντας από τυχαίο σπίτι; Όχι!
Παράδειγμα:

Ξεκινάω από το x_1, κόστος: 2

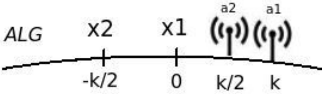


Κεραίες ii)

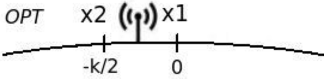
Σπίτια στην περιφέρεια κύκλου

Δουλεύει η λύση του i) ξεκινώντας από τυχαίο σπίτι; Όχι!
Παράδειγμα:

Ξεκινάω από το x_1, κόστος: 2



Κόστος optimal: 1



Περιττή κεραία αν ξεκινήσω απο x_1

Αλγόριθμος:
Τρέξε το α) ξεκινώντας από κάθε σημείο, και πάρε το min.
Πολυπλοκότητα $O(n^2)$

Κεραίες ii) Αλγόριθμος/Απόδειξη

Αλγόριθμος:
Τρέξε το α) ξεκινώντας από κάθε σημείο, και πάρε το min.
Πολυπλοκότητα $O(n^2)$
Απόδειξη/επιχείρημα ορθότητας:

- Αν υπάρχει instance που δεν έχω επικάλυψη της πρώτης με την τελευταία $\Rightarrow$ είναι OPT (ίδιο με α)
- Σε όλες εχω επικάλυψη και διαλέγω την min: έστω όχι η opt $\rightarrow$ προσπαθώ να μειώσω τις επικαλύψεις
  - Εστω η τελευταία κεραία, καλύπτει και k πρώτες
  - Μετακινώ την 1η κεραία a_1 για να αποφύγω επικάλυψη
  - Στο αριστερότερο άκρο της a_1 θα έχω κάποιο άλλο σπίτι $x_i \Rightarrow$ ισοδύναμο με το να ξεκινήσω από το $x_i \Rightarrow$ το έχω ήδη κάνει σε άλλη λύση!

Εργοστάσιο Ποτηριών i)

- $n = 100$, $k = 1$
  Βέλτιστη σειρά δοκιμών: 1εκ→2εκ→3εκ ... →(n-1)εκ
  Αν πετάξουμε το μοναδικό ποτήρι απο κάποιο ύψος χωρίς να δοκιμάσουμε όλα τα μικρότερα ύψη και αυτό σπάσει τότε δεν έχουμε βρει λύση.
- $n = 100$, $k = 2$
  binary search?
  - best case: Δοκιμή από τα 50 εκατοστά. Αν το ποτήρι δεν σπάσει δοκιμή από τα 75 εκατοστά κ.ο.κ. Αν είμαστε τυχεροί χρειάζονται 7 δοκιμές.
  - worst case: Δοκιμή από τα 50 εκατοστά. Αν το ποτήρι σπάσει θα πρέπει με το ποτήρι που μένει να δοκιμάσουμε από το 1εκ ως τα 49εκ. Χρειάζονται 50 δοκιμές. Κόστος $O(n)$

Εργοστάσιο Ποτηριών i)

Υπάρχει καλύτερη λύση! Ξεκινάμε από τα 14 εκατοστά.

- Αν το ποτήρι σπάσει τότε με το δεύτερο ποτήρι δοκιμάζουμε από το 1εκ ως τα 13εκ. Χρειάζονται 14 δοκιμές.
- Αν το ποτήρι δεν σπάσει δοκιμάζουμε τα 27εκ. Έτσι αν το ποτήρι σπάσει θα πρέπει να δοκιμάσουμε 15εκ - 26εκ και θα χρειαζόμαστε συνολικά 14 δοκιμές.

Ακολουθούμε το ίδιο σκεπτικό διατηρώντας τις δοκιμές ίσες με 14.

Εργοστάσιο Ποτηριών i)

1ο Ποτήρι	Αν σπάσει: 2ο Ποτήρι	Δοκιμές
14	$1 \to 2 \to \ldots \to 13$	$1 + 13 = 14$
27	$15 \to 16 \to \ldots \to 26$	$2 + 12 = 14$
39	$28 \to 29 \to \ldots \to 38$	$3 + 11 = 14$
50	$40 \to 41 \to \ldots \to 49$	$4 + 10 = 14$
60	$51 \to 52 \to \ldots \to 59$	$5 + 9 = 14$
69	$61 \to 62 \to 63 \to 64 \to 65 \to 66 \to 67 \to 68$	$6 + 8 = 14$
77	$70 \to 71 \to 72 \to 73 \to 74 \to 75 \to 76$	$7 + 7 = 14$
84	$78 \to 79 \to 80 \to 81 \to 82 \to 83$	$8 + 6 = 14$
90	$85 \to 86 \to 87 \to 88 \to 89$	$9 + 5 = 14$
95	$91 \to 92 \to 93 \to 94$	$10 + 4 = 14$
99	$96 \to 97 \to 98$	$11 + 3 = 14$

Εργοστάσιο Ποτηριών i) Βελτιστότητα

Έστω x ο βέλτιστος αριθμός δοκιμών στην χειρότερη περίπτωση.

- Δοκιμάζουμε στα x εκατοστά - καλύπτουμε x βαθμίδες
- Δοκιμάζουμε στα $(x + (x - 1))$ εκατοστά - καλύπτουμε $x - 1$ βαθμίδες.
- κ.ο.κ

Συνολικά καλύπτουμε $x + (x - 1) + (x - 2) + \ldots + 2 + 1 = \frac{x(x+1)}{2}$ βαθμίδες. Πρέπει να καλυφθούν $n = 100$ βαθμίδες άρα:

$$\frac{x(x+1)}{2} \geq 100 \Rightarrow x = 14$$

Δυναμικός προγραμματισμός

Αναδρομή

$D[n, k] = 1 + min\{max\{D[i-1, k-1], D[n-i, k]\}\}, i = 1, 2, ..., n$
$D[1, k] = 1, D[0, k] = 0$
$D[n, 1] = n$

$D[n, k]$ ελάχιστος αριθμός δοκιμών στην χειρότερη περίπτωση για n εκατοστά (βαθμίδες) και k ποτήρια
Δοκιμάζουμε την i-οστή από μία ακολουθία n διαδοχικών βαθμίδων.

- Αν το ποτήρι σπάσει το πρόβλημα περιορίζεται σε $k-1$ ποτήρια και $i-1$ διαδοχικές βαθμίδες.
- Αν δεν σπάσει έχουμε k ποτήρια και $n-i$ διαδοχικές βαθμίδες.

Εργοστάσιο Ποτηριών ii) Πολυπλοκότητα

Πολυπλοκότητα

$\mathit{State\ Space} \times \mathit{Work\ for\ each\ step} = O(nk \times n) = O(n^2 k)$

Εργοστάσιο Ποτηριών iii) Με διωνυμικούς συντελεστές

Έστω $b(t,k)$ ο αριθμός των βαθμίδων που καλύπτονται από k ποτήρια με t δοκιμές.

- Αν το ποτήρι σπάσει καλύπτονται $b(t-1,k-1)$ βαθμίδες.
- Αν δεν σπάσει έχουμε καλύπτονται $b(t-1,k)$ βαθμίδες.
- $b(t,k) = 1 + b(t-1,k-1) + b(t-1,k)$

Ορίζουμε $g(t,k) = b(t,k+1) - b(t,k) = g(t-1,k) + g(t-1,k-1)$.
Η παραπάνω σχέση ισχύει για τους διωνυμικούς συντελεστές, επομένως $g(t,k) = \binom{t}{k}$
(εξαίρεση: $g(0,0) = 0$ λόγω αρχικών συνθηκών)

Εργοστάσιο Ποτηριών iii) Με διωνυμικούς συντελεστές

Επίσης ισχύει:

$$b(t,k) = g(t,k-1) + g(t,k-2) + \ldots + g(t,0) = \sum_{i=1}^{k} \binom{t}{i}$$

(τηλεσκοπικό άθροισμα)

Τώρα αρκεί απλώς να βρούμε ένα t τ.ω. $b(t,k) = \sum_{i=1}^{k} \binom{t}{i} \geq n$ ώστε να καλυφθούν όλες οι βαθμίδες.

Πολυπλοκότητα

Με binary search έχουμε $O(k \log(n))$