Problem analysis of Greedy Algorithm(2)

YAO ZHAO

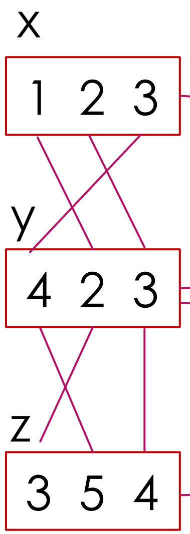
Nesting boxes

- A $\boldsymbol{d}$**-dimensional** box with dimensions $(x_1, x_2, \ldots, x_d)$ ***nests*** within another box with dimensions $(y_1, y_2, \ldots, y_d)$ if there exists a permutation π on $\{1, 2, \ldots, d\}$ such that $x_{\pi(1)} < y_1, x_{\pi(2)} < y_2, \ldots, x_{\pi(d)} < y_d$.
- (1) Argue that the nesting relation is transitive.
- (2) Describe an efficient method to determine whether one $\boldsymbol{d}$**-dimensional** box nests inside another.
- (3) Suppose that you are given a set of $\boldsymbol{n}$ $\boldsymbol{d}$**-dimensional** boxes $\{B_1, B_2, \ldots B_n\}$. Give an efficient algorithm to find the longest sequence $\langle B_{i1}, B_{i2}, \ldots B_{ik} \rangle$ of boxes such that B_{ij} nests within B_{ij+1} for $j = 1,2, \ldots k - 1$. Express the running time of your algorithm in terms of n and d.

Question 1: Argue that the nesting relation is transitive

- Suppose that box $\mathrm{x} = (x_1, x_2, \dots, x_d)$ nests with box $\mathrm{y} = (y_1, y_2, \dots, y_d)$ and box y nests with box $\mathrm{z} = (z_1, z_2, \dots, z_d)$ Then there exist permutations π and σ such that $x_{\pi(1)} < y_1, x_{\pi(2)} < y_2, \dots, x_{\pi(d)} < y_d$ and $y_{\sigma(1)} < z_1, y_{\sigma(2)} < z_2, \dots, y_{\sigma(d)} < z_d$. This implies $x_{\pi(\sigma(1))} < z_1, x_{\pi(\sigma(2))} < z_2, \dots, x_{\pi(\sigma(d))} < z_d$, so x nests with z and the nesting relation is transitive.

Sample:



exist permutations π : $\pi(1)$=3 $\pi(2)$=1 $\pi(3)$=2

$x_{\pi(1)} < y_1, x_{\pi(2)} < y_2, x_{\pi(3)} < y_3$

$x_3 < y_1, x_1 < y_2, x_2 < y_3$

exist permutations σ : σ (1)=2 σ (2)=1 σ (3)=3

$y_{\sigma(1)} < z_1, y_{\sigma(2)} < z_2, y_{\sigma(3)} < z_3$

$y_2 < z_1, y_1 < z_2, y_3 < z_3$

exist permutations $\pi\sigma$:

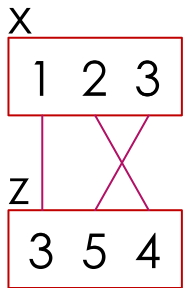
$\pi(\sigma(1))$= $\pi(2)$=1

$\pi(\sigma(2))$= $\pi(1)$=3

$\pi(\sigma(3))$= $\pi(3)$=2

$x_{\pi(\sigma(1))} < z_1, x_{\pi(\sigma(2))} < z_2, x_{\pi(\sigma(3))} < z_3$

$x_1 < z_1, x_3 < z_2, x_2 < z_3$



Question 2: Describe an efficient method to determine whether one d-dimensional box nests inside another

- Box x nests inside box y if and only if the increasing sequence of dimensions of x is component-wise strictly less than the increasing sequence of dimensions of y. Thus, it will suffice to sort both sequences of dimensions and compare them. Sorting both length d sequences is done in $O(d \log d)$, and comparing their elements is done in $O(d)$, so the total time is $O(d \log d)$.

Question 2:proof

- **Pf.** Box x nests inside box y if and only if the increasing sequence of dimensions of x is component-wise strictly less than the increasing sequence of dimensions of y.

the increasing sequence of dimensions of x is component-wise strictly less than the increasing sequence of dimensions of y.

By the definition of Nesting

Box x nests inside box y

Box x nests inside box y

?

the increasing sequence of dimensions of x **must be** component-wise strictly less than the increasing sequence of dimensions of y.

- **Claim.** the increasing sequence of dimensions of x must be component-wise strictly less than the increasing sequence of dimensions of y if Box x nests inside box y .

- **Pf. (by induction)**
  - Base: let dimension = 1, if Box x nests inside box y , $x_1 < y_1$
  - Induction: Suppose that dimension = d, if Box x nests inside box y, there is an increasing permutations π for x and an increasing permutations σ for y, satisfy $x_{\pi(1)} < y_{\sigma(1)}, x_{\pi(2)} < y_{\sigma(2)}, \ldots, x_{\pi(d)} < y_{\sigma(d)}$
  - When dimension =d+1, if Box x nests inside box y, according the definition, there exist permutations λ :

- $x_{\lambda(1)} < y_1, x_{\lambda(2)} < y_2, \ldots, x_{\lambda(d)} < y_d, x_{\lambda(d+1)} < y_{d+1}$ -------(1)

Observe(1), the first d terms, $x_{\lambda(1)} < y_1, x_{\lambda(2)} < y_2, \ldots, x_{\lambda(d)} < y_d$, The first d terms satisfy the nested relationship of d dimensions, so for the first d terms, there will exist permutations π, σ

- $x_{\pi(1)} \leq x_{\pi(2)} \leq \ldots \leq x_{\pi(\mathrm{d})}$ --------(2)
- $y_{\sigma(1)} \leq y_{\sigma(2)} \leq \ldots \leq y_{\sigma(d)}$ --------(3)
- $x_{\pi(1)} < y_{\sigma(1)}, x_{\pi(2)} < y_{\sigma(2)}, \ldots, x_{\pi(d)} < y_{\sigma(d)}$ --------(4)

- **Pf. (continue)**

$x_{\lambda(d+1)}$ is inserted into Formula (2) (let the insertion position be j), thereby having a new permutation π' such that the x sequence remains increasing.

Then insert y_{d+1} into Formula (3) (let the insertion position be k), a new permutation σ' is created so that y sequence remains increasing.

$x_{\lambda(d+1)} = x_{\pi'(j)} < y_{d+1} = y_{\sigma'(k)}$ (5)

There are 3 cases:

- $j < k$: $x_{\pi'(1)} < y_{\sigma'(1)}, x_{\pi'(2)} < y_{\sigma'(2)}, \ldots x_{\pi'(j)} < x_{\pi'(j+1)} < y_{\sigma'(j)}, x_{\pi'(j+1)} < x_{\pi'(j+2)} < y_{\sigma'(j+1)},$ $\ldots x_{\pi'(k-1)} < x_{\pi'(k)} < y_{\sigma'(k)},$ $x_{\pi'(k)} < y_{\pi'(k-1)} < y_{\sigma'(k)}, x_{\pi'(k+1)} < y_{\sigma'(k+1)}, x_{\pi'(d+1)} < y_{\sigma'(d+1)}$

  So, for any $1 \le i \le d+1$, $x_{\pi'(i)} < y_{\sigma'(i)}$, the claim is true.

- $j = k$:

the insertion position: we have $x_{\lambda(d+1)} < y_{d+1}$ (see formula(1)), so $x_{\lambda(d+1)} = x_{\pi'(j)} < y_{d+1} = y_{\sigma'(k)}$(Formula(5)) = $y_{\sigma'(j)}$ (j=k)

The other positions hasn't changed. for any $1 \le i \le d+1$, $x_{\pi'(i)} < y_{\sigma'(i)}$, the claim is true.

- $j > k$: $x_{\pi'(1)} < y_{\sigma'(1)}, x_{\pi'(2)} < y_{\sigma'(2)}, \ldots x_{\pi'(k)} < x_{\pi'(j)} < y_{\sigma'(k)}$ (see formula(5)), $x_{\pi'(k+1)} < x_{\pi'(j)} < y_{\sigma'(k)} < y_{\sigma'(k+1)}, \ldots. x_{\pi'(j)} < y_{\sigma'(k)} < y_{\sigma'(j)}, x_{\pi'(j+1)} < x_{\pi'(j+1)}, x_{\pi'(d+1)} < y_{\sigma'(d+1)}$

So, for any $1 \le i \le d+1$, $x_{\pi'(i)} < y_{\sigma'(i)}$, the claim is true.

Sample:

	1	2	3	4
x	21	23	25	27
y	31	33	35	37

22
36

J=2 < k=4

	1	2	3	4	5
x	21	22	23	25	27
y	31	33	35	36	37

	1	2	3	4
x	21	23	25	27
y	31	33	35	37

22
32

j=k=2

	1	2	3	4	5
x	21	22	23	25	27
y	31	32	33	35	37

	1	2	3	4
x	21	23	25	27
y	31	33	35	37

26
32

J=4 > k=2

	1	2	3	4	5
x	21	23	25	26	27
y	31	32	33	35	37

Question 3: Suppose that you are given a set of n d-dimensional boxes $\{B_1, B_2, \ldots B_n\}$. Give an efficient algorithm to find the longest sequence $\langle B_{i1}, B_{i2}, \ldots B_{ik}\rangle$ of boxes such that B_{ij} nests within B_{ij+1} for $j = 1, 2, \ldots k-1$. Express the running time of your algorithm in terms of n and d.

- We will create a nesting-graph G with vertices $B_1, B_2, \ldots B_n$ as follows. For each pair of boxes B_i, B_j, we decide if one nests inside the other. If B_i nests in B_j, draw an arrow from B_i to B_j. If B_j nests in B_i, draw an arrow from B_j to B_i. If neither nests, draw no arrow. To determine the arrows efficiently, after sorting each list of dimensions in $O(nd \log d)$ we compare all pairs of boxes using the algorithm from part (2) in $O(n^2 d)$. By part (1), the resulted graph is acyclic, which allows us to easily find the longest chain in it in $O(n^2)$ in a bottom-up manner. This chain is our answer. Thus, the total time is $O(nd * max(\log d, n))$.

Sample:

8 boxes

5	2	20	1	30	10
23	15	7	9	11	3
40	50	34	24	14	4
9	10	11	12	13	14
31	4	18	8	27	17
44	32	13	19	41	19
1	2	3	4	5	6
80	37	47	18	21	9

sorting each list:

Node 1

1 2 5 10 20 30

Node 2

3 7 9 11 15 23

Node 3

4 14 24 34 40 50

Node 4

9 10 11 12 13 14

Node 5

4 8 17 18 27 31

Node 6

13 19 19 32 41 44

Node 7

1 2 3 4 5 6

Node 8

9 18 21 37 47 80

Create a nesting-graph

Node 1

1 2 5 10 20 30

Node 2

3 7 9 11 15 23

Node 3

4 14 24 34 40 50

Node 4

9 10 11 12 13 14

Node 5

4 8 17 18 27 31

Node 6

13 19 19 32 41 44

Node 7

1 2 3 4 5 6

Node 8

9 18 21 37 47 80

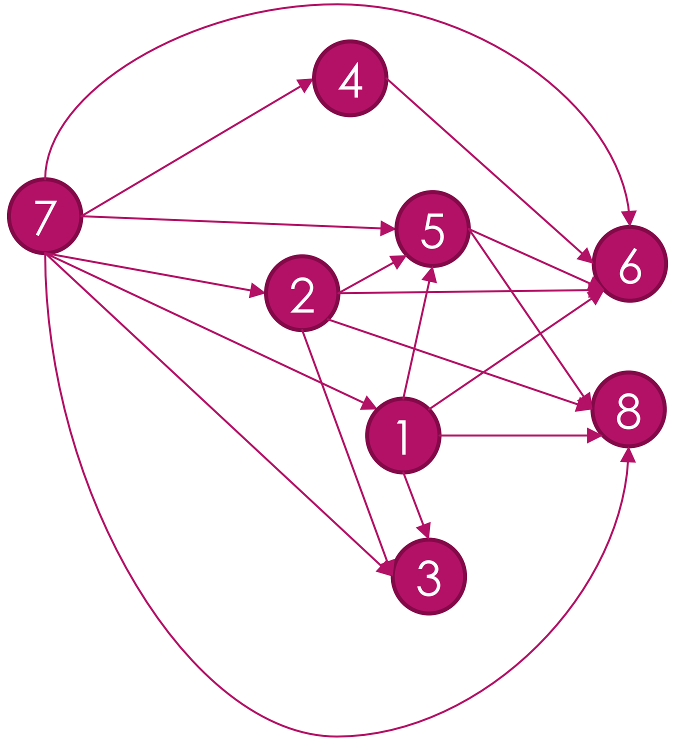
compare all pairs of boxes using the algorithm from part (2)

	1	2	3	4	5	6	7	8
1	0	0	1	0	1	1	0	1
2	0	0	1	0	1	1	0	1
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	1	0	0
5	0	0	0	0	0	1	0	1
6	0	0	0	0	0	0	0	0
7	1	1	1	1	1	1	0	1
8	0	0	0	0	0	0	0	0

We will create a nesting-graph G with vertices $\boldsymbol{B_1}, \boldsymbol{B_2}, \ldots \boldsymbol{B_n}$ as follows. For each pair of boxes $\boldsymbol{B_i}, \boldsymbol{B_j}$, we decide if one nests inside the other. If $\boldsymbol{B_i}$ nests in $\boldsymbol{B_j}$, draw an arrow from $\boldsymbol{B_i}$ to $\boldsymbol{B_j}$. If $\boldsymbol{B_j}$ nests in $\boldsymbol{B_i}$, draw an arrow from $\boldsymbol{B_j}$ to $\boldsymbol{B_i}$. If neither nests, draw no arrow.

Find the longest chain in the DAG in $O(n^2)$ in a bottom-up manner

	1	2	3	4	5	6	7	8
1	0	0	1	0	1	1	0	1
2	0	0	1	0	1	1	0	1
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	1	0	0
5	0	0	0	0	0	1	0	1
6	0	0	0	0	0	0	0	0
7	1	1	1	1	1	1	0	1
8	0	0	0	0	0	0	0	0



the longest chain :4

The following are the possible chain:
7 → 1 → 5 → 8
or
7 → 1 → 5 → 6
or
7 → 2 → 5 → 6
or
7 → 2 → 5→ 8