Lecture 14 +15+16+17

Core Algorithms

1, ... n $(x_1, \ldots x_n) \in \{0,1\}^n$

$p_1 \ldots p_n$

$w_1 \ldots w_n$

N-V algo for TSP
- $O(n^3 \phi)$
- in expectation.

maximize $p_1 x_1 + \ldots + p_n x_n$

subject to $w_1 x_1 + \ldots + w_n x_n \leq W$

$x_i \in \{0,1\} \quad \forall i$

Fractional knapsack
→ where fractions of items can be stored

Relaxation:

$x_i \in [0,1] \quad \forall i$

ie x_i - allowed to be a real value.

Greedy for frac. knaps

Sort items in descending order p_i/w_i ratio

1 2 ... n

$p_i/w_i \geq \frac{p_{i+1}}{w_{i+1}}$

Add items as long as weight does not exceed capacity.

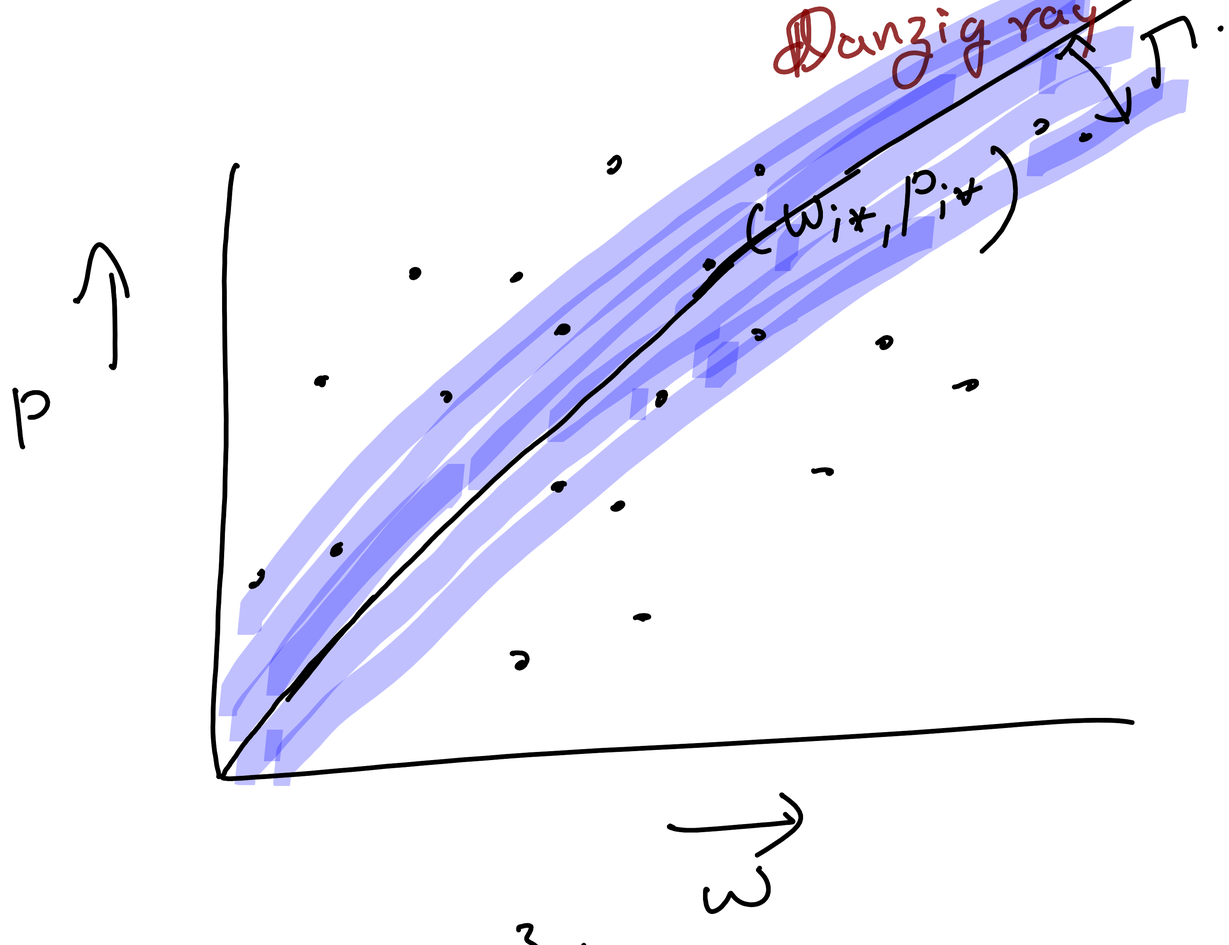
i^*-1

$x_1, x_2 \ldots x_{i^*-1}$

obs:- The solution is optimal & there is at one most one item i^* s.t $x_{i^*} \notin \{0,1\}$

(w_{i^*}, p_{i^*}).

$\hat{x}$ be an optimal solution.



Stmt: if $w_1 \cdots w_n$ & $p_1 \cdots p_n$ are ϕ-perturbed then $\exists$ Π s.t # of items within a distance of Π from the Danzig ray is $O(n^{\frac{1}{3}})$

$\therefore$ Expected running time = $Sort(n) + (n^{\frac{1}{3}})^3 \cdot \phi$.

$\leq O(n \log n)$.

Details see Röglin's notes.

A quick review of Complexity classes

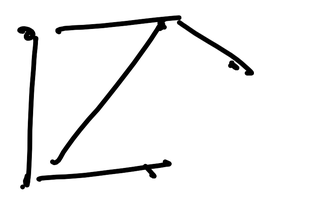
NP: non-deterministic polynomial time.

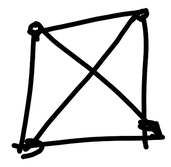
Set of all problems that have efficiently verifiable soln/witnesses.

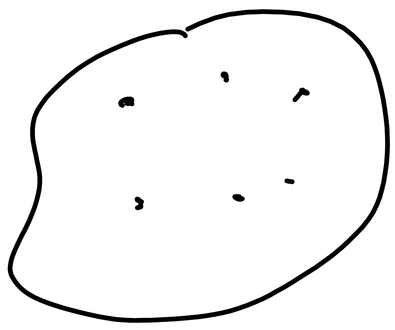
e.g. CLIQUE problem.

i/p = $G = (V, E)$ & $K > 0$

o/p: YES iff G has a clique on K vertices







witness for YES instance

A subset $S \subseteq V$, $|S| = K$
s.t $\forall\, i \neq j \in S$, $(i,j) \in E$.

$\therefore$ CLIQUE $\in$ NP

Vertex Cover
Dominating set
Hamiltonian cycle

Given G & K is there a path of length K

NP-hard

A problem B is NP hard if:
there is a poly time algo for B then for every problem $A \in NP$, we have a polynomial time algorithm.

Note:- CLIQUE, DomSet, VC, HamCycle, k-path, k-cycle
Knapsack, TSP, clustering.
All are NP hard problem. (in fact NP-complete).

For us:- No NP-complete problem is likely to have a polynomial time algorithm. ie $NP \neq P$.
P≠NP hypothesis

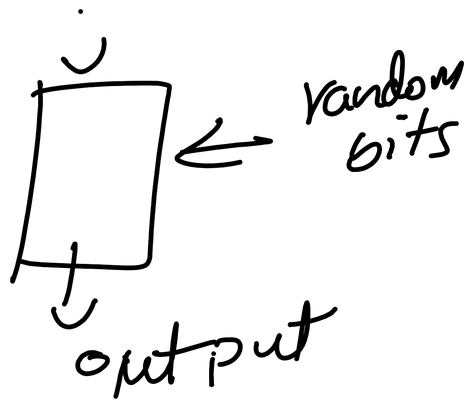
Suppose an algorithm A used random coin tosses-

Monte Carlo & Las Vegas.

runtime expectation.
o/p: guaranteed

runtime guaranteed
o/p: expection (ie possibility of errors).

TSP-implementations
Concorde-



Definition A problem B is said to be in ZPP (Zero error prob. poly time)
if it has a randomized algo which has expected poly time & outputs correct value.

Weakened P vs NP

Lecture - 15

Weakened P vs NP

NP ≠ ZPP.

Knapsack: has smoothed poly time algo

TSP ? - 2OPT - does not give exact soln.

Steiner tree ?

K-means ?

Strongly NP - hard

A problem B (a binary optimization) is said to B strongly NP hard if the restriction when $|c_i| \in [0, n^c]$ for some $c>0$ integers. remains NP hard.

$$\max_{\min} \; c_1x_1 + \cdots + c_nx_n$$

Subj: . . .

$x_1 \ldots x_n \in \{0,1\}$.

Example:- TSP when the edge weights are 1 or 2 remains NP hard.

∴ TSP is strongly NP-hard.

Ex:-

① Show that ~~TSP~~ can be formulated as a binary optimization problem.

② Suppose there is a poly time algorithm that solves TSP where weights are 1 or 2. Show that

this algo. can be used to solve the HamCycle problem.

Thm: Let Π be a binary opt. problem which is strongly NP-hard. N – input size n – # of variables.
Then there does not exist an algo for Π whose expected run. time is bounded by $poly(N, \phi)$ for ϕ – perturbed instances with coefficients from $[-1, 1]$ unless $NP \subseteq ZPP$.

Corollary:- TSP cannot have smoothed poly time algo unless $NP \subseteq ZPP$.

Pf:- A be an algo for ϕ- perturbed instances of Π

$\forall f_1 \cdots f_n$ $\qquad$ $f_1 \cdots f_n$

s.t $\mathbb{E}[\text{time}(A, y)] = poly(N, \phi)$

$y \leftarrow \phi$ perturb

Now, Π is strongly NP hard, Let Q be a poly s.t the special case of Π where $|c_i| \leq Q(N)$ remain NP-hard.

[eg: for TSP $Q(n) = 2$ suffices]

Π: minimize
$c_1 x_1 + \cdots + c_n x_n$
subj. to: . . .

I be an input instance of Π of length N and integer

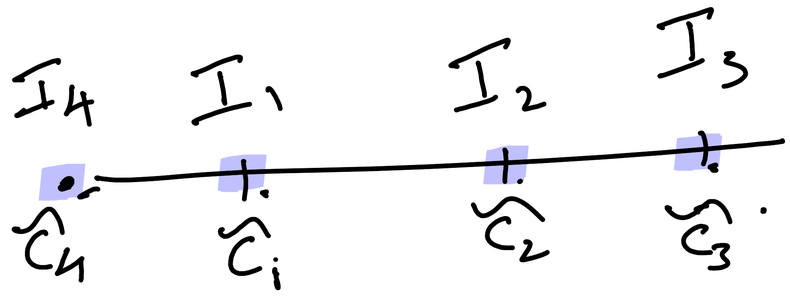
Coeff $c_1, \ldots c_n$, $|c| \le Q(N)$

integers

Step 1: Scale the coeff! $\hat{c_1} \ldots \hat{c_n}$ s.t $\hat{c_i} \in [-1, 1]$

$C = \max_i \{|c_i|\}$, $\hat{c_i} = c_i / C$.

Obs: Solutions remain the same



Lecture - 16

Set: $\phi = 2nC$

For each i, choose an interval I_i of length $\frac{1}{\phi}$ in $[-1, 1]$

s.t $\hat{c_i} \in I_i$

ϕ perturbed instance.

choose $c_1', c_2', \ldots, c_n'$ where c_i' is sampled uniformly at random from I_i

Density for c_i': $f_i(x) = \phi$.

Note: c_i' is ϕ-perturbed

I' be the new instance.

Algorithm

input I

① Obtain I'

② Run A on I', Let x^* be the soln.

③ Output x^*.

Time

$\rightarrow O(n)$

$\rightarrow$ Expected $p(n)$ time.

Expected poly

Correctness:- Show that x^* is also an optimal soln for I

pf:- Suppose not. Say x is an optimal soln for I

but x^* is not.

$c = (c_1, \cdots, c_n)$

$$cx^* - cx \geq 1 \quad (\text{as } c_i \text{ - integers})$$

$\therefore \quad \tilde{c}x^* - \tilde{c}x \geq \frac{1}{C}$ (by scaling). $\tilde{c} = (\tilde{c_1}, \cdots, \tilde{c_n})$ $\rightarrow$ (A)

we have $|\tilde{c_i} - c_i'| \leq \frac{1}{\phi} \quad \forall i.$

$c'x^* = \sum c_i' x_i^*$

$c'x = \sum c_i' x_i$

$$\therefore \quad c'x^* - c'x \geq \tilde{c}x^* - \tilde{c}x - \frac{n}{\phi}$$

$$= \tilde{c}x^* - \tilde{c}x - \frac{n}{2nC} = \tilde{c}x^* - \tilde{c}x - \frac{1}{2C} \geq \frac{1}{2C}$$ by (A)

ie x^* is not an optimal soln for I' — contradiction

$\therefore$ x^* must also be opt for I.

$\therefore$ opt

Knapsack $\Longleftarrow$ psly (N, C) - algorithm, where $C = \max\{|c_i|\}$.

$W \rightarrow$ max wt

$O(W \cdot n^2)$

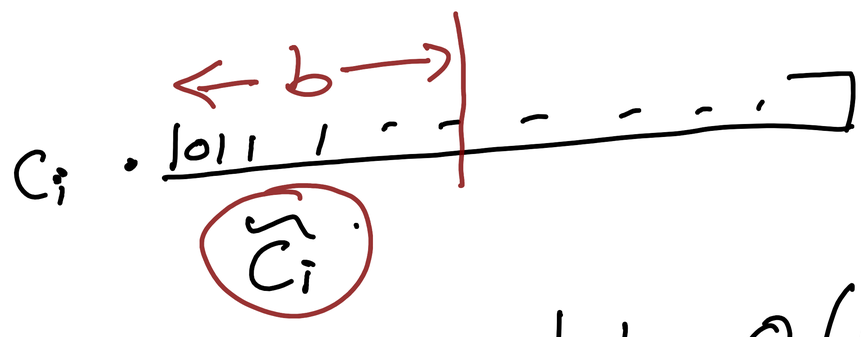
$\hookrightarrow$ pseudo linear time

Thm Let Π be binary optimization problem such that it has a pseudo linear time algorithm. Then Π has an expected poly time algo on ϕ-perturbed instances.

pf:- A_p is a pseudo linear time algo for Π

time$(A_p) = O(N^k \cdot C)$ → cont
L→ weight

Assumption: coeff are from $[-1\ 1]$. a/b - $\max\{|a|, |b|\}$



$c_1 \cdots c_n$

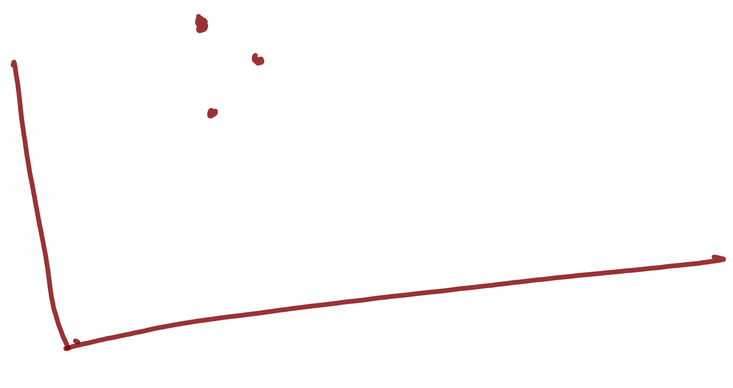
$\tilde{c}_1 \cdots \tilde{c}_n$ } - can be solved in tim $O(N^k \cdot 2^b)$

$b = O(\log N)$

Lecture-18

Let A be a pseudo linear time algorithm for Π.

Proposed dates for midsem:
Mar-19-21
5pm 11:59.



$c_1, \cdots c_n$ } I

I_b
↳ truncated instance.

$S \subseteq \{0,1\}^n$ - be the set of all feasible solutions.

b - to be chosen later.

I, $S \subseteq \{0,1\}^n$ obj: minimize $c_1 x_1 + \cdots + c_n x_n$.

$$x^* = \operatorname{argmin}\{cx \mid x \in S\}$$

$$x^{**} = \operatorname{argmin}\{(x)\ x \in S \setminus \{x^*\}\}.$$

Ref: Heiko Röglin

Define $\Delta = cx^{**} - cx^* \geq 0$

Lemma [Isolation Lemma] Let I be an instance of Π with $c_1 \cdots c_n$ - ϕ-perturbed. Then $\forall\ \epsilon > 0$.

$$\Pr[\Delta \leq \epsilon] \leq 2n\phi\epsilon$$

$c_{i_1}, \cdots c_n \in [-1, 1]$.

c_i — $\lfloor c_i \rfloor_b$ — rounded to b-bits down

c_i — $\lceil c_i \rceil_b$ — rounded up to b-bits

$b = 3$

$\cdot 1\ 1\ 0\ 1\ 1\ 0$

$\lfloor c_i \rfloor_b$

$\lceil c_i \rceil_b : \cdot 1\ 1\ 1$

$\lfloor c_i \rfloor_b \leq c_i$

$\lceil c_i \rceil_b \geq c_i$

Obs:- $|\lfloor c_i \rfloor_b - c_i| \leq 2^{-b}$

$|\lceil c_i \rceil_b - c_i| \leq 2^{-b}$.

Let $\lfloor c \rfloor_b = (\lfloor c_1 \rfloor_b, \cdots, \lfloor c_n \rfloor_b)$

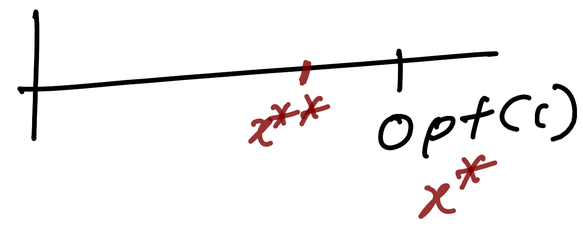
$\lceil c \rceil = (\lceil c_1 \rceil_b, \cdots, \lceil c_n \rceil_b)$

$Opt(c)$: optimum value.

Corrollary Let c' best $\forall i \quad c_i' \in \{\lfloor c_i \rfloor_b, \lceil c_i \rceil_b\}$.

Then $\Pr\left[\text{opt}(c) \neq \text{opt}(c')\right] \leq 2^{-b+2} \cdot n^2 \cdot \phi$

(c ϕ-perturbed)



pf:-

$\forall i \in \{1,\dots n\}$

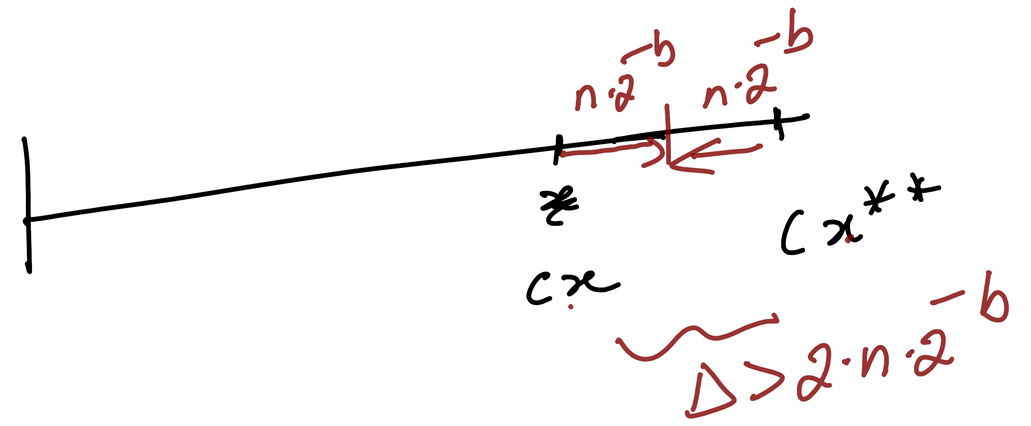
$|c_i - c_i'| \leq 2^{-b}$

$\therefore \forall x \in \{0,1\}^n$.

$|cx - c'x| \leq n \cdot 2^{-b}$.

$|cx^* - c'x^*| \leq n \cdot 2^{-b}$

$\therefore$ If $\Delta > 2n2^{-b}$ then x is an optimal soln w.r.t c, it remains so w.r.t c'. $|cx - c'x| \leq n \cdot 2^{-b}$



$\Pr\left[\text{opt}(c) \neq \text{opt}(c')\right]$

$\leq \Pr\left[\Delta \leq 2n2^{-b}\right] \leq 2n \cdot \phi \cdot 2n2^{-b}$

$= 4n^2 2^{-b} \phi$.

$\phi \leq n$.

$b = 4 \log n$.

Suppose $b = 4 \log n$.

then $\Pr\left[\Delta \leq 2n2^{-b}\right] \leq \frac{4}{n}$ assuming $\phi \leq n$.

First Algo:-

i/p: $C = (c_1 \ldots c_n)$

Step1:- truncate c_i to b-bits to get c_i'

Step2: Run Ap on $C' = (c_1', \ldots c_n')$. to get a solution x

Step3:- ~~o/p~~ x.

step4: Check if x is opt if not got to step1:

$\Pr[\text{Algo fails}] \leq \Pr[\Delta \leq 2n \cdot 2^{-b}] \leq \frac{4}{n}$.

Runtime
$= \text{time for trunc} + \text{time (Ap)}$
$= O(N) + O(N^k \cdot 2^b)$
$= O(N^k \cdot 2^b)$
$= O(N^{k+4})$ $b = 4\log n$
$n \leq N$.

Lecture 18

TI: $\min \; c_1 x_1 + \cdots + c_n x_n$

Subj to: $x \in S \subseteq \{0,1\}^n$

$c_1 \ldots c_n$ are ϕ-perturbed $\quad c_i \sim f_i : [-1,1] \to [0,\phi]$
↳ density fn.

$x^* = \arg\min \{ cx \mid x \in S \}$ → optimal soln

$x^{**} = \arg\min \{ cy \mid y \in S \setminus \{x^*\} \}$ → next best
$cx^* \leq cx^{**}$

Winner gap $\Delta = cx^{**} - cx^* \geq 0$

Obtaining an error free algo (expected pdy runtime)

Certifying optimality.

Let I be a ϕ-perturbed instance of Π.
$x \in S$ be a feasible solution.

task:- Given x & $b \in \mathbb{N}$, check if x is an optimal solution or not.

$x = x_1 \cdots x_n$

Certify (c, x, b)

1 For $i \in \{1, \cdots n\}$ set $\hat{c_i} = \begin{cases} \lfloor c_i \rfloor_b & \text{if } x_i = 0 \\ \lceil c_i \rceil_b & \text{if } x_i = 1 \end{cases}$

$\hat{c} \in (\hat{c_1}, \cdots \hat{c_n})$

2 Compute $y = opt(\hat{c})$ using A_P.

3. if $x = y$ then return True
else return False

Lemma:- If certify (c, x, b) returns true, then x is an optimal solution to the instance I with $c = (c_1 \cdots c_n)$

proof:-

For the sake of contradiction suppose that certify (c, x, b) returns true, but x is not optimal. Let z be an optimal soln. ($cost\ of\ z < cost\ of\ x$)

$cz < cx$

Consider any index i s.t $x_i \neq z_i$

x_i	z_i
1	0
0	1

Case1:- $x_i = 0$ & $z_i = 1$

$$\hat{c_i} x_i - \hat{c_i} z_i = c_i x_i - \hat{c_i} z_i = c_i x_i - \lfloor c_i \rfloor_b z_i$$
$$\geq c_i x_i - c_i z_i$$

Case 2: $x_i = 1$ & $z_i = 0$

$$\tilde{c}_i x_i - \hat{c}_i z_i = \hat{c}_i x_i - c_i z_i = \lceil c_i \rceil_b x_i - c_i z_i \geq c_i x_i - c_i z_i$$

$$c_1 x_1 + \cdots + c_n x_n \geq c_1 z_1 + \cdots + c_n z_n$$

$$\tilde{c}_1 x_1 + \cdots + \tilde{c}_n x_n \quad \tilde{c}_1 z_1 \cdots + \tilde{c}_n c_n$$

$\therefore \hat{c} x - \hat{c} z \geq c x - c z > 0$

$\therefore z$ is a cheaper soln than x to the instance $\hat{c} = (\hat{c}_1, \ldots \hat{c}_n)$ as well. - Contradiction to the fact that $opt(\hat{c}) = x$ (since certify returns True)

$\therefore$ If certify returns True, then x is an optimal soln.

c_i : $0.\alpha_1 \alpha_2 \cdots \Theta \cdots$ (b)

Final Algo

Adaptive Rounding (II)

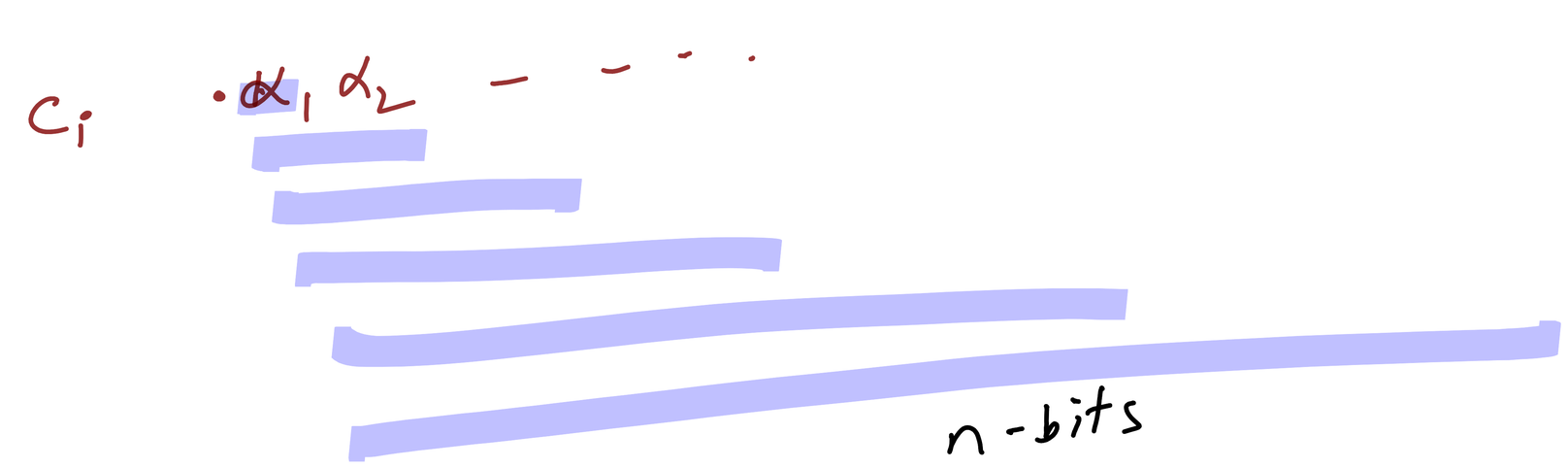
1 For $b = 1$ to $\ldots n$ do

2 Compute $x = opt(\lfloor c \rfloor_b)$ using A_P } $O(N^k \cdot 2^b)$

3. If certify $(c, x, b) =$ True then o/p x & halt $O(N^k \cdot 2^b)$

4 Compute $x = opt(c)$ by brute force. $O(2^N)$.

5. return x.



Run time : $\sum_{b=1}^{n} \Pr[\#\text{ iterations} = b] \cdot N^K \cdot 2^b + \Pr[\#\text{ iterations} > n] \cdot N^K \cdot 2^n$.

X: be the random variable representing the # of iterations.

$\Pr[X \geq j] = \Pr[$ certify $(c, x, 1)$ returns false $\wedge$
certify $(c, x, 2)$ $\wedge$
certify $(c, x, j-1)$ ... false$]$

$\leq \Pr[$ certify $(c, x, j-1)$ returns false$]$.

To estimate: $\Pr[$ certify $(c, x, j-1)$ returns false$]$.

Suppose $\Delta > 2^{-b+1} \cdot n$. then $Opt(c') = Opt(c)$

for any c' s.t $c'_i = \lfloor c_i \rceil_b$ or $\lceil c_i \rceil_b$.

$\therefore \Pr[$ certify returns false$]$
$\leq \Pr[\Delta \leq 2^{-b+1} \cdot n] \leq 2^{-b+2} \cdot n^2 \cdot \phi$.

$$\mathbb{E}[\text{time}] \leq \sum_{b=1}^{n} N^K \cdot 2^b \cdot \Pr[X \geq j] + O(2^n) \cdot \Pr[X > n]$$

$$\leq \sum_{b=1}^{n} O(N^K \cdot 2^b) \cdot 2^{-b+2} \cdot n^2 \phi + O(2^n) \cdot 2^{-n+2} \cdot n^2 \phi$$

$$\leq O(N^{k+3} \cdot \phi)$$