Lec 18

→ Chomsky Normal Form (CNF)

$G = (N, \Sigma, P, S)$ is in CNF if all prod^n are of the form

$A \to BC$ $\quad A \to a$ $\quad A, B, C \in N$ $\quad a \in \Sigma$

- one step progress < # terminals ++
  # non terminals ++

(no epsilon or unit productions)

→ **Theorem**: for any CFG G, there is a CFG G' in CNF st $L(G') = L(G) - \{\varepsilon\}$

1) Claim 1: For any CFG $G = (N, \Sigma, P, S)$ there is a CFG G' with no ε or unit prod^n st $L(G') = L(G) - \{\varepsilon\}$

2) Claim 2: For any non-null $x \in \Sigma^*$, any derivation $S \xrightarrow[G]{*} x$ of min length does not use ε or unit prod^n

→ Converting into CNF

i) for each terminal $a \in \Sigma$ introduce a new non terminal A_a and add prod^n $A_a \to a$

ii) Replace all occur^n of a on RHS with A_a (except $B \to a$). Then all are of form

$A \to a$ $\quad A \to B_1 B_2 \ldots B_K \quad K \geq 2$

iii) for $K \geq 3$, introduce new non terminals

$A \to B_1 C \quad C \to B_2 \ldots B_K$ & so on.

Lecture 19

→ ## Linear Grammar

i) A CFG G is right linear if all prodⁿ are of the form

$A \to xB$, $A \to x$ $\quad A, B \in N \quad x \in \Sigma^*$

(at most one non terminal on the RHS and it must be right most)

ii) A CFG G is left linear if all prodⁿ are of the form

$A \to Bx$, $A \to x$ $\quad A, B \in N \quad x \in \Sigma^*$

(at most one non terminal on the RHS and it must be left most)

iii) Regular Grammar ⇒ Right or Left linear either

iv) Linear Grammar: at most one non terminal can occur on RHS in any prodⁿ irresp. of position.

Regular ⟹ Linear

Theorems:

1. Let G be a right ~~(or left)~~ linear grammar then $L(G)$ is regular.
2. Let $A \subseteq \Sigma^*$ be a regular set then $\exists$ a right ~~(left)~~ linear grammar G st $L(G)=A$
3. $A \subseteq \Sigma^*$ is regular iff $\exists$ a regular grammar G st $L(G)=A$.

→ ## Closure Properties

Kleene Star ✓

Union ✓

Concatenation ✓

Intersection ✗

i) **Union**: Let A and B be CFL and $G_1, G_2 \rightarrow$ CFG st $L(G_1)=A$ $L(G_2)=B$

$G_1 = (N_1, \Sigma_1, P_1, S_1)$ $G_2 = (N_2, \Sigma_2, P_2, S_2)$

- if $N_1 = N_2$, rename one.
- $G = (N_1 \cup N_2, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2, S)$

$S \rightarrow S_1$, $S \rightarrow S_2$

ii) **Concatenation**: $L(G_1)=A$ $L(G_2)=B$ with S_1 & S_2 as start symbols,

$L'(G) = AB = \{xy \mid x \in A, y \in B\}$

- Combine G_1 and G_2
- Add $S \rightarrow S_1 S_2$

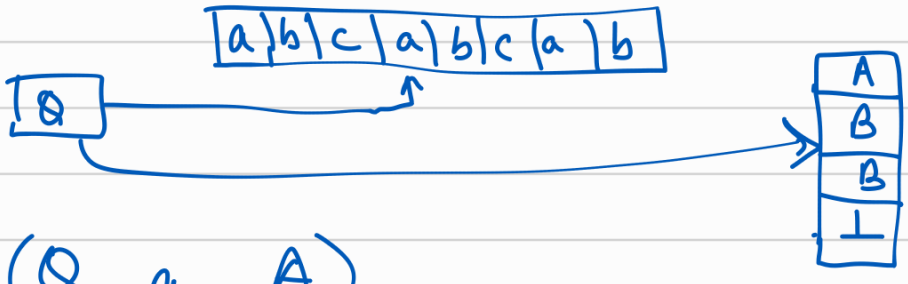
iii) **Kleene Star:** $L(G_1) = A$ S_1 = start sym
then $L(G) = A^*$

- Take G_1 along with a new start symbol S $S \rightarrow S_1 S \mid \varepsilon$

iv) **Intersection:** CFLs are not closed under intersection. $\{a^m b^m c^n\} \cap \{a^n b^m c^m\}$
$= \{a^n b^n c^n\}$ not CFL

Lecture 20

1) NPDA: Finite automata + Stack



(Q, a, A)

$$M = (Q, \Sigma, \Gamma, \delta, s, \perp, F)$$

$F \subseteq Q$
$s \in Q$

Q = finite set of states
Σ = finite set → input alphabet
Γ = finite set → stack alphabet
$\perp$ = initial stack symbol

$B_1 \ldots B_k$

$$\delta \subseteq (Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma) \times (Q \times \Gamma^*)$$

$$((p, a, A), (q_1, B_1 \ldots B_k)) \in \delta$$

Configuration of M: $\in Q \times \Sigma^* \times \Gamma^*$

(curr state, unread input string, curr stack)

start config = $(s, x, \perp)$

- if $((p, a, A), (q, \gamma)) \in \delta$ then
  $(p, ay, A\beta) \xrightarrow[M]{} (q, y, \gamma\beta)$
- if $((p, \varepsilon, A), (q, \gamma)) \in \delta$
  $(p, y, A\beta) \xrightarrow[M]{} (q, y, \gamma\beta)$

→ Acceptance by final state —

$(s, x, \perp) \xrightarrow[M]{*} (q, \varepsilon, \gamma)$ $q \in F$, $\gamma \in \Gamma^*$ (↓ any string)

→ Acceptance by empty stack —

$(s, x, \perp) \xrightarrow[M]{*} (q, \varepsilon, \varepsilon)$ $q \in Q$ (↳ any state)

→ NPDA

i) $\perp$: any other stack symbol used to denote start config^n. Need not always be at the bottom.

ii) If stack is empty, no transition applies.

iii) To accept by empty stack, everything must be popped out & nothing pushed back.

→ Intersection of CFL and regular set

CFLs are closed under intersection with regular sets. if $A \subseteq \Sigma^*$ is a CFL and $B \subseteq \Sigma^*$ is regular then $A \cap B$ is a CFL.

Lecture 22

→ Parse Trees

A parse tree satisfies –

i) Each interior node is labelled with an element of N.

ii) Each leaf node is labelled with Σ or ε.

iii) If an interior node is labelled A and its children are $B_1, \ldots B_k$ then $A \longrightarrow B_1 B_2 \ldots B_k \in P$

Depth = no. of edges in the longest path from root to leaf node.

For a grammar in CNF, no. of nodes can atmost double in every step. So in i^{th} level, atmost 2^i nodes $\Rightarrow$ for a string of length 2^n, depth must be atleast n $\Rightarrow$ Parse tree has $(n+1)$ levels.

Pumping Lemma

if $A \subseteq \Sigma^*$ is a CFL then $\exists\ k \geq 0$ st $\forall\ z \in A$ st $|z| \geq k$, can be split into 5 substrings $z = uvwxy$ st $vx \neq \varepsilon$ $|vwx| \leq k$ and $\forall i \geq 0$ $uv^i w x^i y \in A$.

To prove not CFL, contrapositive form

$\forall\ k \geq 0$ $\exists\ z \in A$ st $|z| \geq k$ and $\forall$ splits $uvwxy = z$ with $vx \neq \varepsilon$ and $|vwx| \leq k$ $\exists\ i \geq 0$ st $uv^i w x^i y \notin A$.

$\forall\ k$, $\exists\ z$, $\forall$ splits, $\exists\ i$.

Lecture 23

NPDAs and CFGs have equivalent expressive power.

Theorem 1: Given a CFG G we can construct an NPDA M st $L(M) = L(G)$.

Theorem 1: Given an NPDA M we can construct a CFG G st $L(M) = L(G)$

Lecture 24

→ Set Membership Question

Given a set $A \subseteq \Sigma^*$ and $x \in \Sigma^*$ is $x \in A$? $|x| = k$

1) **Regular set A:**

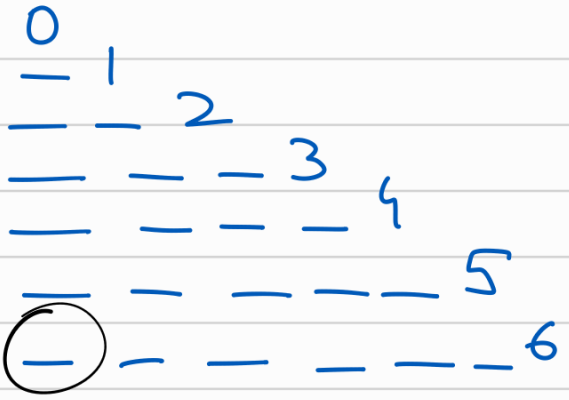
- DFA M st $L(M) = A \Rightarrow$ simulate M on x. iff we reach $q \in F_M$ then $x \in A$. $O(k)$
- NFA N st $L(N) = A \Rightarrow$ run x over all possible runs in $N \Rightarrow O(kn^2)$ $n = |Q|$ (we keep a set of union of all states we can go using that alphabet from current set of states ($\Rightarrow$ worst $n \to n$ i.e. kn^2)

2) **CFL A:** $O(k^3 n)$ $k = |x|$, $n = |P|$ } CKY algo

for each substring y of x, the set of all non terminals that generate y.

→ CKY Algo

$x = a\ a\ b\ b\ a\ b$
$0\ 1\ 2\ 3\ 4\ 5\ 6$



if $S \in T_{0,6}$ then $x \in L(G)$

$T_{i,j} \to$ loop over $K = 1$ to $j-1$
$P \to XY$ then $X = T_{i,i+K}$, $T_{i+K,j} = Y$ write P.

$T_{i,j}$: set of non-terminals that generate substring x_{ij} of x.

→ Cocke, Kasami & Younger Algo

(len = 1 strings)

```
for (i = 0 to n-1) begin
    T_{i,i+1} = φ
    For (A → a ∈ P)
        if a = x_{i,i+1} then T_{i,i+1} = T_{i,i+1} ∪ {A}
end

for m = 2 to n do
    for i = 0 to n-m do
        T_{i,i+m} = φ.
        for j = i+1 to i+m-1 do
            for (A → BC) ∈ P
                if B ∈ T_{i,j} and C ∈ T_{j,i+m}
                then T_{i,i+m} = T_{i,i+m} ∪ {A}
```

$O(n^3 p)$ $\quad |x| = n \quad |P| = p$

→ Ambiguous Grammar

A CFG G is ambiguous if $\exists\ x \in L(G)$ for which there are two different parse trees. (NOT 2 diff derivations, but 2 diff leftmost derivations).

(same parse tree can have 2 diff derivations) but that is not ambiguous.

A CFL $A \subseteq \Sigma^*$ is inherently ambiguous if $\forall$ CFG G st $L(G) = A$, G is ambiguous.

DPDA $\longleftrightarrow$ DCFL :

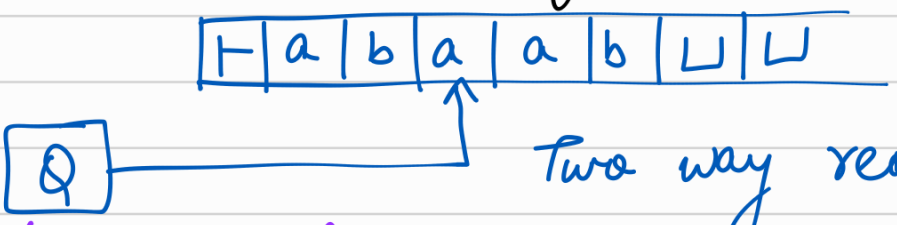
CFLs that can be accepted by a DPDA.

doubt → DCFLs always admit an unambiguous grammar.

DCFLs $\subset$ unambiguous CFLs.

$\forall G$ or $\exists G$ for that DCFL

→ Deterministic Turing Machine



Two way read/write head

finite set of states + semi-∞ tape on right. left end marker ⊢.

head → left right, read write symbols on the tape.

→ start in s and head at left end marker ⊢

→ read symbol on tape under head, & current state → write a new symbol on the tape. Move head ← or →.

→ accept → special accept state (t)
reject → special reject state (r)

$M = (Q, \Sigma, \Gamma, \vdash, \sqcup, \delta, s, t, r)$

$\Sigma \subseteq \Gamma \qquad \sqcup, \vdash \in \Gamma - \Sigma$

$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$

configuration $\in Q \times \{ y \sqcup^w \mid y \in \Gamma^* \} \times \mathbb{N}$

(p, z, n) → $n \geq 0$ current posⁿ of head on tape

state ← current content of tape

Start = $(s, \vdash x \sqcup^w, 0)$

$s_b^n(z)$ → in z, the n^{th} char replaced by b.

halt { accept ⇒ $(s, \vdash x \sqcup^w, 0) \xrightarrow[M]{*} (t, y, n)$

reject ⇒ $(s, \vdash x \sqcup^w, 0) \xrightarrow[M]{*} (r, y, n)$

- $\forall p \in Q, \exists q \in Q \quad \delta(p, \vdash) = (q, \vdash, R)$
- $\forall b \in \Gamma, \exists c, c' \in \Gamma$ and $d, d' \in \{L, R\}$

$\delta(t, b) \neq (t, c, d) \qquad \delta(r, b) = (r, c', d')$

Recursive: $A \subseteq \Sigma^*$ if $A = L(M)$ for some total TM M.

Recur. (11)

RE

RE: if $A = L(M)$ for some TM M.

- Recursive sets ⇒ closed under complementation
- $A = re \quad \bar{A} = re \Rightarrow A, \bar{A} =$ recursive

⊢	⊢	q	b		⊔	⊔
	⊢	b	b		b	⊔
	⊢	a	a		a	⊔

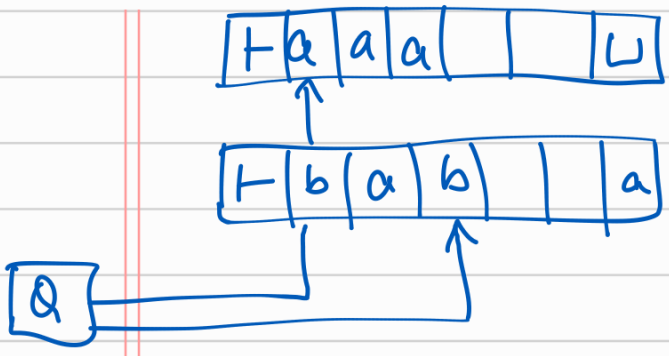
Tape symbol is a triple (c, d, e)

c
d
e

Multiple Tracks, single head

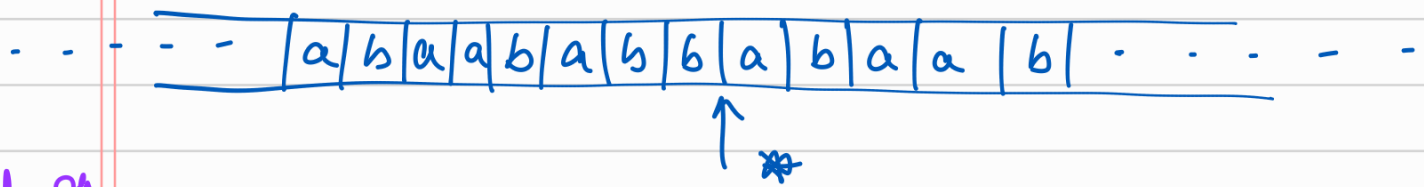
$\Gamma' = \{ \hat{a} \mid a \in \Gamma \}$ → for head posⁿ on each track

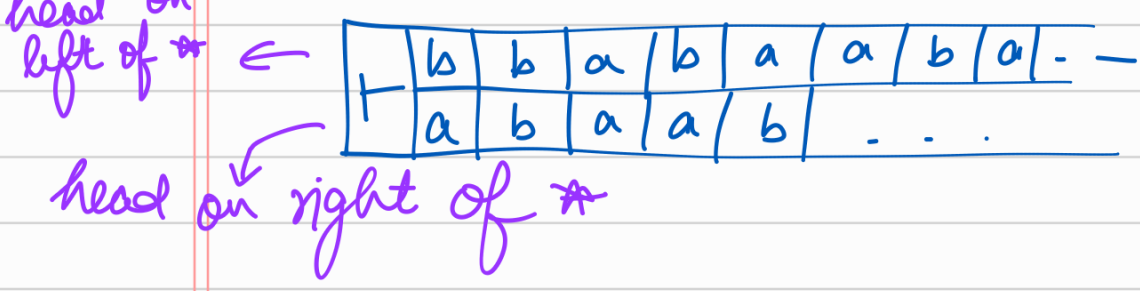
$\Sigma \cup \{\vdash\} \cup (\Gamma \cup \Gamma')$ → new tape symbol



$\delta : Q \times \Gamma^2 \to Q \times \Gamma^2 \times d^2$

Multiple tapes





Lect 27

→ Universal TM

Fix some encoding of any TM & its δ as a string
- easy to interpret
- able to encode all TMs upto isomorphism.
- able to decode from string.

$$L(U) = \{ M \# x \mid x \in L(M) \}$$

M: encoded M
$\#$: Symbol in U's alphabet delimit M and x.
x: encoded x over input alphabet of M.

- check if M and x are valid encodings, if not reject.
- do a step by step simulation of M on x.

description of M
Contents of M's tape
State of M & pos^n of tape head

- M accepts x ⇒ U accepts M#x
- M rejects x ⇒ U rejects M#x.
- M loops on x ⇒ U loops on x

$HP = \{M\#x \mid M \text{ halts on } x\}$
re ✓ recursive ✗ → Cantor Diagonaln

$MP = \{M\#x \mid x \in L(M)\}$
re ✓ rec ✗

$\overline{HP} = \{M\#x \mid M \text{ does not halt on } x\}$
re ✗ rec ✗

$\overline{MP} = \{M\#x \mid x \notin L(M)\}$
re ✗ rec ✗

→ Properties of TM

For a TM is it decidable if

i) $\varepsilon \in L(M)$ ii) $L(M) = \phi$ iii) $L(M) = \Sigma^*$

```
N(y) {
 erases y;
 write x on tape;
 run (M,x);
 accept; }
```

$L(N) = \Sigma^*$ if M halts on x

$L(N) = \phi$ if M loops on x.

assume ∃ Total TM K for each quesn.
feed N in them.

For a TM is it decidable if

i) $L(M)$ is regular ii) $L(M)$ is CFL iii) $L(M)$ is recursive

$N(y)$ {
save y on one track;
write x on another track;
run (M, x);
run (M_A, y);
accept if M_A accepts; }

$A = HP$ (re, not rec)

feed N to each Total TM of these.

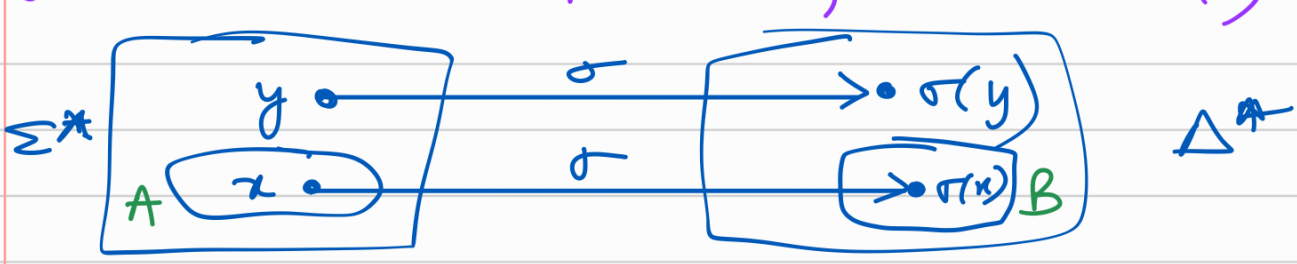
$$L(N) = \begin{cases} A & \text{if } M \text{ halts on } x \\ \phi & \text{if } M \text{ loops on } x \end{cases}$$

Lec 31

→ Reductions

Given $A \subseteq \Sigma^*$ $B \subseteq \Delta^*$ a many-one red^n of A to B is a computable func

$\sigma : \Sigma^* \to \Delta^*$ st $\forall x \in \Sigma^*$, $x \in A \iff \sigma(x) \in B$



$A \leq_m B \Rightarrow A$ reduces to B via a map σ.

→ Transitive : $A \leq_m B$ (σ) and $B \leq_m C$ (τ)
$\Rightarrow A \leq_m C$ $\tau \circ \sigma$

$A \leq_m B \implies \overline{A} \leq_m \overline{B}$

i) B is re then A is re
ii) A is not re then B is not re

i) B is rec then A is rec
ii) A is not rec then B is not rec

→ **Rice Theorem**

Every non-trivial property of r.e. sets is undecidable.

$P: \{\text{re sets of } \Sigma^*\} \to \{T, \perp\}$

Non trivial → ∃ re sets A and B st $P(A) = T$ and $P(B) = \perp$

Lect 33

→ **VALCOMPS**

$$\#\alpha_0\#\alpha_1\#\alpha_2 \ldots \#\alpha_N\#$$

$$\Delta = \{\#\} \cup (\Gamma \times (Q \cup \{-\}))$$

VALCOMPS $(M, x) = \phi$ if M does not halt on x
$\overline{\text{VALCOMPS}}\,(M, x) = \Sigma^*$ if M " " "

↳ CFL

$\overline{HP} \leq \{G \mid L(G) = \Sigma^*$ i.e. VALCOMP$\}$

$\overline{HP}$ not re

$L(G) = \Sigma^*$ not re

Lect 34

→ Running Time of a TM M

i) $F: \mathbb{N} \to \mathbb{R}^+$ $F(N)$ is the max no. of steps M takes on any input of len n.

Asymptotic upper bound-

$f, g: \mathbb{N} \to \mathbb{R}^+$ $f(n) = O(g(n))$ if

$\exists c$ and n_0 st $\forall n \geq n_0$ $f(n) \leq c\, g(n)$

TM: $\forall x \in \Sigma^*$ M halts in atmost $O(t(|x|))$ steps $\Rightarrow t: \mathbb{N} \to \mathbb{R}^+$

$TIME(t(n)) = \{ L \mid \exists$ a deterministic TM having running time $O(t(n))$ st $L = L(M) \}$

Complexity class P

$$P = \bigcup_k TIME(n^k)$$

class of all languages that are decidable in poly. time on a DTM of single tape.

Examples $\in P$

i) $PATH = \{ G \# u \# v \mid G$ is a dir graph & $u \to \dots \to v \}$

ii) $CFL\text{-}M = \{G \# x \mid x \in L(G) \quad G = CFL\}$

→ NP class

poly. time verifier. ⇒ poly in len of inp

A verifier for a lang L is an algo V where $L = \{w \mid V \text{ accepts } w \# k$ for some c)

$SAT = \{\alpha \mid \exists$ a valⁿ V st $V \models \alpha\}$

$HPATH = \{G \# s \# t \mid G =$ dir with H path from s to t $\}$

→ Non Deter. TM

$\delta: Q \times \Gamma \to P(Q \times \Gamma \times \{L, R\})$

$NTIME(t(n)) = \{L \mid \exists$ a non-deter TM M running in time $t(n)$ st $L = L(M)\}$

$NP = \bigcup_k NTIME(n^k)$

P → class of lang in which memb. can be decided efficiently

NP → class of lang in which memb can be verified efficiently.

Lect 35

For every ND TM there is an equi DTM.

Every $t(n)$ time ND single tape TM has an equivalent $2^{O(t(n))}$ time DTM of single tape.

Running Time of NTM is $t: \mathbb{N} \to \mathbb{R}^+$ if $\forall x \in \Sigma^*$ M halts in atmost $O(t(|x|))$ steps
$\Rightarrow$ Max no. of steps M has in any comp $= O(t(|x|))$.

→ NP Completeness

Cook Levin Th : $SAT \in P$ iff $P = NP$

Poly Time Redn :

A func $f: \Sigma^* \to \Sigma^*$ is poly time comp func if some poly time TM exist that given x writes $F(x)$ on its tape.

$\forall w \in \Sigma^*$, $w \in A$ iff $F(w) \in B$
F is a poly time comp func. $A \leq_p B$

if $A \leq_p B$ and $B \in P \Rightarrow A \in P$

NP Complete

i) $A \in NP$

ii) every $B \in NP$ is poly time red to A
i.e. $\forall B \in NP, \exists B \leq_p A$.

→ A is NP comp. and $A \in P$ then $P = NP$.

→ A is NP comp & $A \leq_p B$ for $B \in NP$
$\Rightarrow B$ is NP comp.