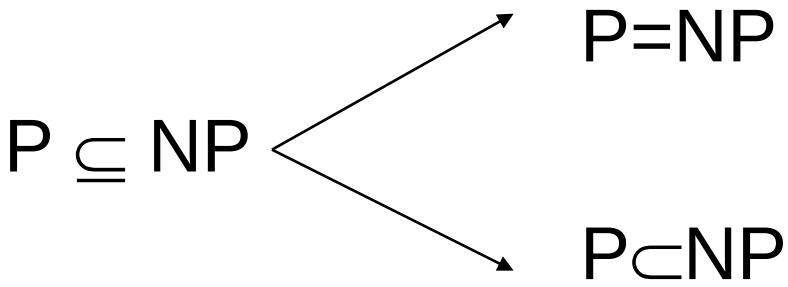
NP-completitudine

1. P versus NP
2. NP-completitudine

NP-completitudine



⇒ există argumente atât în favoarea <u>tezei</u> ***P ≠ NP*** cât şi în favoarea <u>tezei</u> ***P = NP*** .

NP-completitudine

$P \neq NP \Leftrightarrow$ <u>nu există</u> un algoritm rapid care să inlocuiască căutarea directă.

Cea mai bună metodă cunoscută în prezent pentru a rezolva probleme din clasa *NP* în mod determinist necesită timp de lucru exponenţial

$\Rightarrow$ putem demonstra că: $NP \subseteq EXPTIME = \bigcup_{k \in N} TIME(2^{n^k})$

<u>Terminologie</u>

P = clasa problemelor rezolvabile algoritmic în timp de lucru **polinomial** (determinist),

NP = clasa problemelor rezolvabile algoritmic în timp de lucru **exponenţial** (determinist).

= clasa problemelor rezolvabile algoritmic în timp polinomial **nedeterminist**

= clasa problemelor **verificabile** algoritmic în timp polinomial determinist.

.

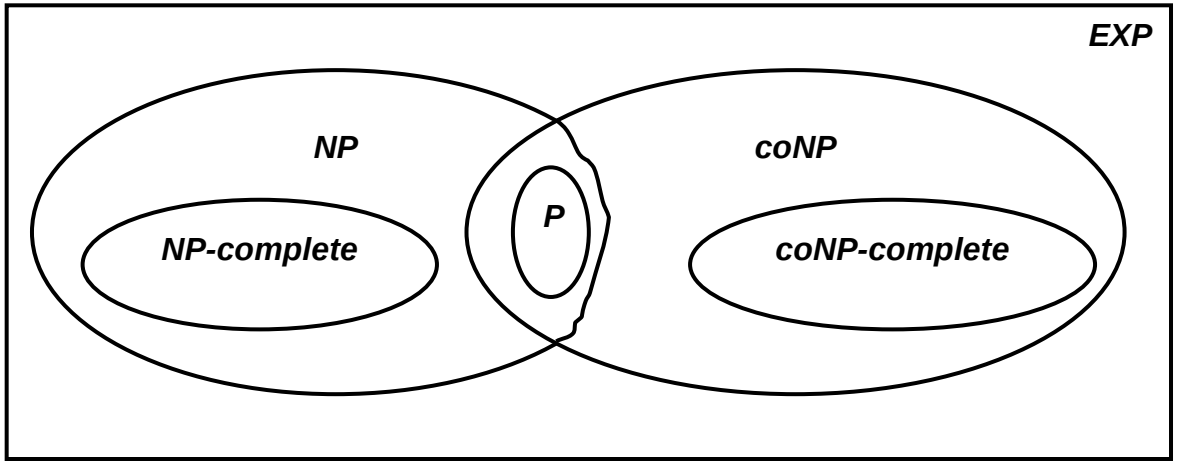
NP-completitudine

Teorema 1

$P \subseteq NP \cap coNP$

Observatia 1

$P \subset NP \cap coNP$.



Observatia 2

$P \subset EXPTIME$.

NP-completitudine

1. P versus NP
2. NP-completitudine

NP-completitudine

1970: Stephen COOK şi Leonid LEVIN:

Probleme din clasa *NP* a căror complexitate proprie este strâns legată de complexitatea întregii clase.

Aceste probleme se numesc **NP-complete**.

NP-completitudine

Fenomenul *NP*-completitudinii este important din punct de vedere:

- teoretic:
- practic:

Cercetatorii considera că $P \neq NP$

$\Rightarrow$ demonstrarea faptului că o problemă este *NP*-completă este o dovadă puternică a caracterului ei nepolinomial.

NP-completitudine

Exemplul 1: Problema evaluarii = satisfiabilitatii (Satisfiability Problem)

Definiţia 1

O formulă booleană se numeşte satisfezabila=evaluabilă

$\Leftrightarrow$ există o combinaţie de valori de adevăr care, date variabilelor booleene din formulă, evaluează formula la valoarea 1.

Problema evaluării (satisfiabilităţii) constă în a verifica dacă o formulă booleeană oarecare este evaluabilă (satisfezabilă) şi se codifică prin:

$SAT = \{ < \phi > \mid \phi$ este o formulă booleeană evaluabilă $\}$

NP-completitudine

Teorema 2 (Cook-Levin)

$SAT \in P \Leftrightarrow P = NP$

demonstraţie

Metoda folosită: reductibilitatea polinomială a timpului de lucru.

NP-completitudine

Definiţia 2

Funcţia $f: \Sigma^* \to \Sigma^*$ se numeşte polinomial calculabila

$\Leftrightarrow$ există o MT cu timp de lucru polinomial: $\forall\ w \in \Sigma^*$, se opreşte, având pe bandă secvenţa $f(w)$.

Definiţia 3

Fie limbajele $A, B \subseteq \Sigma^*$.

Limbajul A se numeşte polinomial reductibil la limbajul B

$\Leftrightarrow\ \exists\ f : \Sigma^* \to \Sigma^*$ polinomial calculabilă astfel încât $\forall w \in \Sigma^*$:

$w \in A \leftrightarrow f(w) \in B$.

Funcţia f se numeşte reducerea polinomială a lui A la B.

Notatia 1

$A \leq_P B$

NP-completitudine

Teorema 3

Fie limbajele $A, B \subseteq \Sigma^*$.

Daca $A \leq_P B$ si $B \in P \Rightarrow A \in P$.

demonstratie

Cf. ip.: $\exists$ M = algoritm cu timp de lucru polinomial care decide B

Cf. ip.: $\exists$ f = reducere polinomiala a lui A la B

Construim urmatorul algoritm N care decide limbajul A:

N = " Fie secventa de intrare $w \in \Sigma^*$:

1. Se calculeaza f(w).
2. Se ruleaza M pe intrarea f(w); N returneaza exact ceea ce returneaza M."

Cf. ip. $A \leq_P B$: $w \in A \leftrightarrow f(w) \in B \Rightarrow$ M accepta $f(w) \leftrightarrow w \in A$. (*)

N ruleaza in timp polinomial pt ca: (**)

- etapa 1: reducerea f este polinomiala => timp polinomial
- etapa 2: avem o compunere de 2 polinoame => timp polinomial

Cf. (*) si (**): $A \in P$.

NP-completitudine

Definitia 4

Fie un limbaj $B \subseteq \Sigma^*$.

Limbajul B se numeşte NP-complet $\Leftrightarrow$

1. $B \in NP$;
2. $\forall A \in NP: A \leq_P B$.

Teorema 4

Fie un limbaj $B \subseteq \Sigma^*$.

Daca B este NP-complet si $B \in P$ => P=NP.

demonstratie

Rezulta imediat din definitia reductibilitatii polinomiale

NP-completitudine

Teorema 5

Fie un limbaj $B \subseteq \Sigma^*$.

Daca B este NP–complet si

$\forall C \in NP: B \leq_P C$

=> C este NP–complet.

demonstratie

Cf. ip.: B este NP-complet → (cf. Def.4) $\forall A \in NP: A \leq_P B$,

Cf. ip.: $\forall C \in NP: B \leq_P C$,

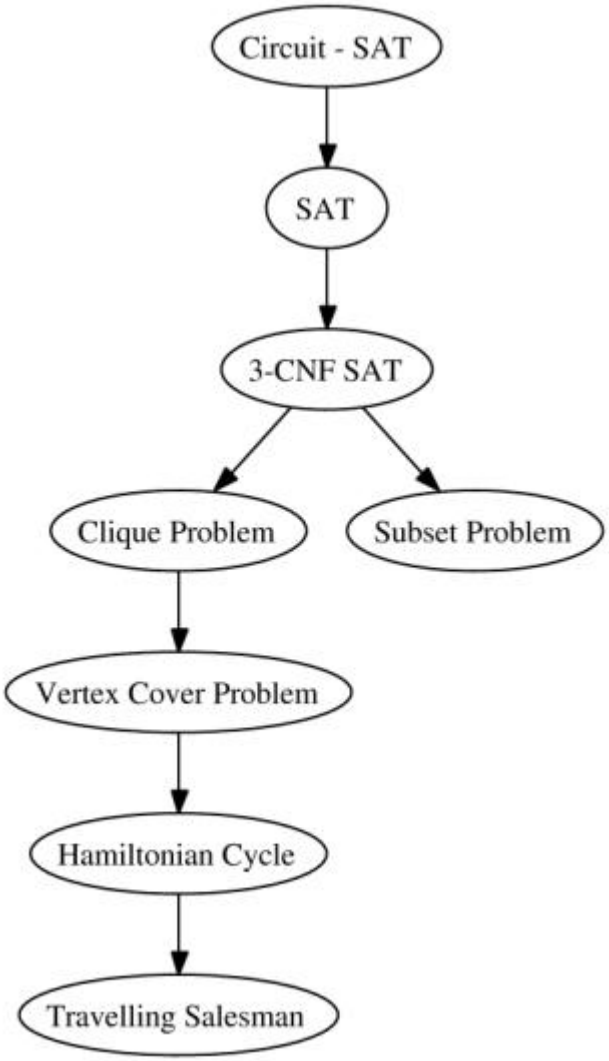
compunerea a 2 polinoame este tot un polinom

$\Downarrow$

$A \leq_P C$

=> (cf. Def.4) C este NP-complet.

NP-completitudine



NP-completitudine

1. P versus NP
2. NP-completitudine