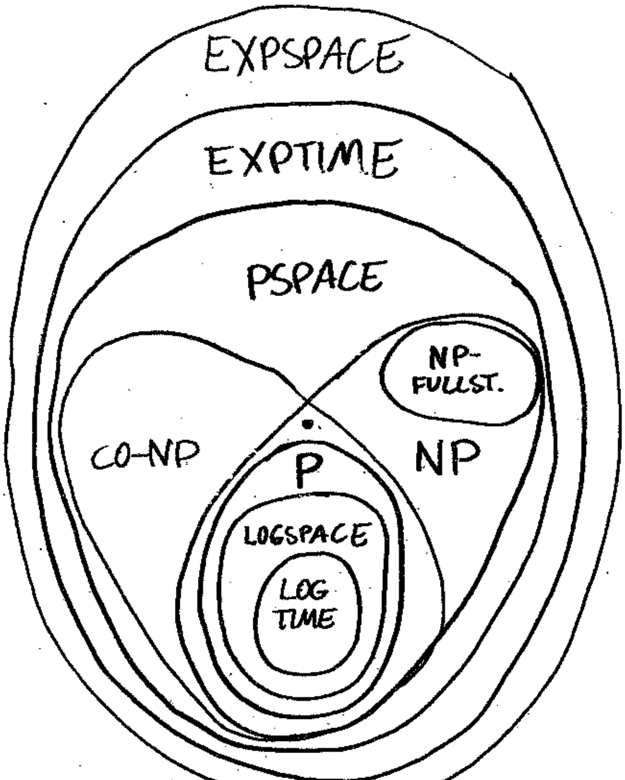
KOMPLEXITETSKLASSER



P – PROBLEM SOM KAN LÖSAS I POLYNOMISK TID

NP – PROBLEM SOM KAN LÖSAS I ICKE-DETERMINISTISK POLYNOMISK TID.

KOMPLEMENTKLASS:

CO-NP ETT PROBLEM $A \in$ CO-NP OM KOMPLEMENTPROBLEMET TILL $A \in NP$, TEX "FINNS DET INTE NÅGON HAMILTONSK CYKEL?"

ANDRA TIDSKLASSER:

LOGTIME – PROBLEM SOM KAN LÖSAS I LOGARITMISK TID

EXPTIME – PROBLEM SOM KAN LÖSAS I EXPONENTIELL TID

MINNESKLASSER:

LOGSPACE – PROBLEM SOM KAN LÖSAS MED TILLGÅNG TILL LOGARITMISKT MYCKET MINNE (OAVSETT TID)

PSPACE – PROBLEM ... POLYNOMISKT MINNE

EXPSPACE – ... EXPONENTIELLT MINNE

Monotona funktioner och co-NP En boolesk formel kallas *monoton* om den byggs upp av enbart onegerade variabler, konstanter, parenteser och operanderna $\wedge$ och $\vee$.

En funktion $f : \{0,1\}^n \to \{0,1\}$ kallas monoton om det för varje indata x gäller att om $f(x) = 1$ och vi bildar x' genom att ändra en eller flera variabler x_i från 0 till 1 så är $f(x') = 1$ (det vill säga hur vi än ändrar variabler från 0 till 1 i indata så kan funktionsvärdet aldrig ändras från 1 till 0). Man kan ganska enkelt visa att varje monoton boolesk formel beskriver en monoton funktion, men det är inte det denna uppgift går ut på.

En funktion som beskrivs av en formel som innehåller negationer kan fortfarande vara monoton, vilket gäller till exempel $(x_1 \wedge \overline{x}_2) \vee x_2$.

Visa att det är co-NP-fullständigt att avgöra ifall en given boolesk formel beskriver en monoton funktion!

Lösning

Kalla problemet MONOTONE. Visa först att MONOTONE tillhör co-NP. Givet en formel ϕ ska vi då kunna verifiera om formeln *inte* beskriver en monoton funktion. Om ϕ inte är monoton så finns det en variabeltilldelning x så att $\phi(x) = 1$ och $\phi(x') = 0$ där variabeltilldelningarna x och x' bara skiljer sig genom att en eller flera variabler som är 0 i x har fått värdet 1 i x'.

Vi gissar därför en variabeltilldelning x och en eller flera av dom nollvärda variablerna och verifierar att $\phi(x) = 1$ och att om vi sätter dom valda variablerna till 1 så blir $\phi = 0$.

Då återstår att reducera ett känt co-NP-fullständigt problem till MONOTONE. Det verkar rimligt att försöka reducera co-SAT som är co-NP-fullständigt eftersom SAT är NP-fullständigt.

Vi noterar först att om en formel ψ inte är satisfierbar så är den monoton (eftersom den alltid är 0 kan den aldrig ändra värde från 1 till 0). Vi måste därför se till att den formel vi konstruerar inte är satisfierbar just då ψ inte är satisfierbar samt att den inte kan bli monoton då ψ är satisfierbar. En formel som uppfyller detta är $\phi = \psi \wedge \neg y$ där y är en ny variabel som inte förekommer i ψ. Reduktionen blir då

```
co-SAT(ψ) =
    φ ← ψ ∧ ¬y
    return MONOTONE(φ)
```

Om ψ inte är satisfierbar så blir ϕ alltid 0, vilket gör att MONOTONE(ϕ) är sann. Om ψ är satisfierbar blir $\phi = 1$ om $y = 0$ och $\phi = 0$ om $y = 1$, varför MONOTONE(ϕ) är falsk. Därmed är reduktionen korrekt. Att den kan beräknas i polynomisk tid är uppenbart.

VISA ATT LOGSPACE $\subseteq$ P

$Q \in$ LOGSPACE $\Longleftrightarrow$ $\exists$ TURINGMASKIN SOM LÖSER Q PÅ ETT ARBETSBAND MED $O(\log n)$ RUTOR (n = ANTALET BITAR I INDATA)

RÄKNA ANTALET MÖJLIGA KONFIGURATIONER FÖR EN SÅDAN TURINGMASKIN!

ANTAL TILLSTÅND HOS TURINGMASKINEN: $O(1)$
POSITIONER FÖR LÄSHUVUDET PÅ INDATABANDET: n
POSITIONER FÖR HUVUDET PÅ ARBETSBANDET: $O(\log n)$
MÖJLIGA INNEHÅLL PÅ ARBETSBANDET: $3^{O(\log n)}$
OM ALFABETET ÄR $\{0,1,B\}$

TOTALT ANTAL MÖJLIGA KONFIGURATIONER: $O(n \log n)\cdot 3^{O(\log n)}$

TURINGMASKINEN KAN INTE VARA I SAMMA KONFIGURATION FLERA GÅNGER FÖR DÅ BLIR DET EN OÄNDLIG SLINGA.

ALLTSÅ: EN ÖVRE GRÄNS FÖR TIDSKOMPLEXITETEN ÄR

$$O(n \log n)\cdot 3^{O(\log n)} = O(n^{O(1)})$$

DVS POLYNOMISK TID.

PROBLEM I PSPACE

QBF – KVANTIFIERAD BOOLESK FORMEL

INDATA: EN BOOLESK FORMEL $\varphi(x_1, x_2, \ldots, x_n)$

FRÅGA: ÄR FÖLJANDE KVANTIFIERADE FORMEL SANN?

$$\exists x_1 \forall x_2 \exists x_3 \ldots Q x_n \ \varphi(x_1, x_2, \ldots, x_n)$$

DÄR Q ÄR $\exists$ OM n ÄR UDDA OCH $\forall$ OM n ÄR JÄMNT

BEVIS AV ATT QBF $\in$ PSPACE

```
EVALQBF(i, n, φ, {x1,...,x_{i-1}}) =
    IF i>n THEN RETURN φ(x1,...,xn)
    xi ← FALSE
    FVAL ← EVALQBF(i+1, n, φ, {x1,...,xi})
    xi ← TRUE
    TVAL ← EVALQBF(i+1, n, φ, {x1,...,xi})
    IF ODD(i) THEN RETURN FVAL ∨ TVAL    (∃)
             ELSE RETURN FVAL ∧ TVAL     (∀)
```

EVALQBF$(1, n, \varphi, \{\})$ LÖSER QBF I $O(n^2)$ MINNE.

(DET GÅR ATT VISA ATT QBF ÄR PSPACE-FULLSTÄNDIGT.)

PROBLEM I EXPTIME

GENERALISERAT SCHACK

INDATA: EN POSITION PÅ ETT $n \times n$-SCHACKBRÄDE MED EN SVART KUNG, EN VIT KUNG OCH ETT ANTAL SVARTA OCH VITA BÖNDER, HÄSTAR, LÖPARE, TORN OCH DAMER.

FRÅGA: FINNS DET NÅGON SÄKER VINNANDE STRATEGI FÖR VIT FRÅN DENNA POSITION OM MAN SPELAR ENLIGT VANLIGA SCHACKREGLER?

BEVIS AV ATT GENERALISERAT SCHACK $\in$ EXPTIME

VARJE RUTA PÅ SCHACKBRÄDET INNEHÅLLER ANTINGEN
- EN SVART PJÄS (6 OLIKA MÖJLIGHETER)
- EN VIT PJÄS (6 OLIKA MÖJLIGHETER)
- INGENTING (EN MÖJLIGHET)

ANTALET MÖJLIGA POSITIONER I ETT SPEL BEGRÄNSAS AV $2\cdot(6+6+1)^{n^2} = 2\cdot 13^{n^2}$.
DET ÄR ANTINGEN SVARTS ELLER VITS TUR

SAMMA POSITION ÅTERKOMMER ALDRIG (I EN VINNANDE STRATEGI).
STORLEKEN PÅ SPELTRÄDET BLIR ALLTSÅ INTE MER ÄN $2\cdot 13^{n^2}$.