

## 7.2 Luokka NP

- Luokka NP on:

$$\begin{aligned} NP &= \bigcup \{ \text{NTIME}(t) \mid t \text{ on polynomi} \} \\ &= \bigcup_{k \geq 0} \text{NTIME}(n^k + k) \end{aligned}$$

- Siis *polynomisessa* ajassa epädeterministisellä Turingin koneella tunnistettavien kielten joukko
- $P \subseteq NP$
- Luokan NP ongelmista puhuttaessa tarkoitetaan yleensä niitä ongelmia, jotka eivät kuulu P:hen
- Esim. Hamiltonin polun etsimiseen suunnatusta verkosta ei tunneta polynomista algoritmia, vaan verkossa on suoritettava kattava haku

- Seuraava epädeterministinen Turingin kone ratkaisee Hamiltonin polku ongelman polynomisessa ajassa
- Syötteellä  $\langle G, s, t \rangle$ , missä  $G$  on suunnattu verkko ja  $s$  sekä  $t$  sen solmuja,  $G$ :n solmujen lukumäärä on  $m$ :

1. Kirjoita  $m$  lukua  $p_1, \dots, p_m$ , missä kukin  $p_i$  on epä-deterministisesti valittu väliä  $1-m$
2. Jos listassa  $p_1, \dots, p_m$  on toistoja, hylkää
3. Jos  $p_1 \neq s$  tai  $p_m \neq t$ , hylkää
4. Jokaisella  $i \in \{1, \dots, m\}$  tarkista onko  $(p_i, p_{i+1})$  kaari  $G$ :ssä.

Jos yksikin testi epäonnistuu hylkää, muuten hyväksy

## Klikkiongelma

- Suuntamattoman verkon **klikki** (clique) on täysin kytketty aliverkko, s.o. kaikkien klikkiin kuuluvien solmujen välillä on kaari
- $k$ -klikki on  $k$ :n solmun aliverkko

$$\text{KLIKKI} = \{ \langle G, k \rangle \mid G \text{ on suuntaamaton verkko, jossa on } k\text{-klikki} \}$$

- Klikkiongelma voidaan ratkaista epädeterministisesti ensin valitsemalla (arvaamalla) annetun verkon  $k$  solmua
- Sen jälkeen deterministisesti tarkistetaan, että kaikkien valittujen solmujen välillä tosiaan on kaari
- Jos ei ole, niin syöte hylätään

## Osajoukkosumma

- Annettuna monijoukko lukuja  $S = \{x_1, \dots, x_k\}$  ja yksi erillinen luku  $t$ , onko  $S$ :llä ali(moni)joukkoa  $\{y_1, \dots, y_l\}$  s.e.  $\sum_i y_i = t$
- Esim. pari  $\langle \{4, 11, 16, 21, 27\}, 25 \rangle$  kuuluu tämän ongelman kieleen, koska  $4 + 21 = 25$
- Kattava haku: käy läpi kaikki  $2^m$  osajoukkoa ja tutki onko alkioiden summa  $t$
- Epädeterministisesti valitaan (arvataan) osajoukko lukuja annetusta kokoelmasta  $S$
- Deterministisesti tarkistetaan, että valittujen lukujen summa on  $t$

## P =?= NP

- P on nopeasti ratkaistavien ongelmien joukko
- NP on nopeasti verifioitavien ongelmien joukko
- Vaikuttaisi ilmiselvältä, NP on oleellisesti P:tä laajempi luokka, mutta ei ole kyetty todistamaan yhdellekään NP:n ongelmalle, ettei sillä ole polynomista ratkaisualgoritmia
- Täten on periaatteessa mahdollista, että P = NP
- Luokan  $NP \setminus P$  ongelmille tunnetaan vain eksponentiaalisen ajan vaativia deterministisiä algoritmeja  

$$NP \subseteq EXPTIME = \bigcup_k DTIME(2^{n^k})$$

## 7.3 NP-täydellisyys

- Funktio  $f: \Sigma^* \rightarrow \Gamma^*$  voidaan laskea polynomisessa ajassa, jos on olemassa Turingin kone  $M$  ja polynomi  $p$ , joilla
  - $f = f_M$  ja
  - $\text{time}_M(n) \leq p(n)$  kaikilla  $n$
- Olkoot  $A \subseteq \Sigma^*$ ,  $B \subseteq \Gamma^*$  formaaleja kieliä
- $A$  voidaan palauttaa polynomisesti  $B$ :hen, merk.
 
$$A \leq_m^p B,$$
 jos on olemassa polynomisessa ajassa laskettava funktio  $f: \Sigma^* \rightarrow \Gamma^*$  s.e.

$$x \in A \Leftrightarrow f(x) \in B \quad \forall x \in \Sigma^*$$

**Lause 7.31** *Kaikilla kielillä  $A, B, C$  pätee*

- i.  $A \leq_m^p A$ , (refleksiivisyys)
- ii. jos  $A \leq_m^p B$  ja  $B \leq_m^p C$ , niin  $A \leq_m^p C$  (transitiivisuus),
- iii. jos  $A \leq_m^p B$  ja  $B \in \text{NP}$ , niin  $A \in \text{NP}$  ja
- iv. jos  $A \leq_m^p B$  ja  $B \in \text{P}$ , niin  $A \in \text{P}$ .

**Huom.:** palautusehdon osalta tämä lause on täsmälleen sama kuin lause 5.22. Muuttunut kohta on palautuksen polynominen laskettavuus.

**Todistus.**

- i. Valitaan palautusfunktiksi  $f(x) = x$ .
- ii. Yhdistetty funktio  $h(x) = g(f(x))$  on palautus  $A$ :sta  $C$ :hen,  $h: A \leq_m C$  (kts. 5.22).  
 $h$  voidaan laskea poly ajassa: Olkoon  $M_f(M_g)$  funktion  $f(g)$  polynomin  $p(q)$  rajoittamassa ajassa laskeva Turingin kone. Voidaan olettaa, että  $p$  ja  $q$  ovat kaikkialla ei-väheneviä.

Olkoon  $M_g$ :n,  $M_{\text{REW}}$ :n ja  $M_f$ :n toiminta kuten lauseen 5.22 todistuksessa.

Yhdistämällä koneet samoin kuin aiemmin saadaan funktion  $h$  laskeva kone  $M_h$ , jonka aikavaativuus syötteellä  $x$  on

$$\begin{aligned} \text{time}_{M_f}(x) + \text{time}_{M_{\text{REW}}}(f(x)) + \text{time}_{M_g}(f(x)) \\ \leq p(|x|) + 2p(|x|) + q(|f(x)|) \\ \leq 3p(|x|) + q(p(|x|)) \\ = O(q(p(|x|))), \end{aligned}$$

joka on polynominen  $x$ :n pituuden suhteen.

- iii. (ja iv.) Palautuksen  $f: A \leq_m^p B$  polynomin  $p$  rajoittamassa ajassa laskeva kone  $M_f$ , kielen  $B$   $q$ :n rajoittamassa ajassa tunnistava kone  $M_B$  ja  $M_{\text{REW}}$  yhdistämällä samaan tapaan kuin lauseen 5.22 todistuksessa, saadaan kielen  $A$  ajassa  $O(q(p(|x|)))$  tunnistava kone  $M_A$ . Se on deterministinen, jos kone  $M_B$ :kin on.  $\square$

## Lausekalkyylin toteutuvuus, SAT

Annettuna lausekalkyylin kaava  $\varphi$  (Boolean lauseke), joka koostuu muuttujista  $x_1, \dots, x_n$ , vakioista 0 (false) ja 1 (true), sekä konnektiiveista  $\vee$ ,  $\wedge$  ja  $\neg$ .

Onko  $\varphi$  toteutuva? Onko olemassa sellaista muuttujien arvoasetusta  $t: \{x_1, \dots, x_n\} \rightarrow \{0, 1\}$ , joilla

$$\varphi(t(x_1), \dots, t(x_n)) = 1$$

Arvataan arvoasetus  $t$  muuttujille ja tarkistetaan, että  $\varphi(t) = 1$ .

Jos  $\varphi$ :ssä on  $n$  muuttujaa, niin  $t$  voidaan esittää  $n$ -bittisenä binäärijonona ja tarkastaminen sujuu polynomisessa ajassa



- Stephen Cook ja Leonid Levin osoittivat 1970-luvun alkuvuosina, että on olemassa *NP-täydellisten* ongelmien luokka
- *NP*-täydellisen ongelman vaativuus on sidoksissa koko luokan *NP* ongelmien vaativuuteen
- Jos polynomiainen ratkaisualgoritmi on olemassa yhdellekin *NP*-täydelliselle ongelmalle, niin sellainen on olemassa kaikille luokan *NP* ongelmille
- Helpottaa ongelman  $P \stackrel{?}{=} NP$  käsittelyä ja vaikeiden ongelmien tunnistamista

**Lause 7.27**  $SAT \in P \Leftrightarrow P = NP$



- Monet lausekalkyylin kaavojen rajoitetut muodotkin ovat *NP*-täydellisiä
- Kaava  $\varphi$  on *konjunkttiivisessa normaalimuodossa* (cnf), jos se on muotoa

$$\varphi = C_1 \wedge C_2 \wedge \dots \wedge C_m,$$

missä kukin *tekijä*  $C_i$  on disjunktio

$$C_i = \alpha_{i1} \vee \alpha_{i2} \vee \dots \vee \alpha_{ir}$$

- Termit  $\alpha_{ij}$  ovat *literaaleja*: muuttujia tai niiden negaatioita
- CSAT on cnf-kaavojen toteutuvuusongelma:  
 $\{ \varphi \mid \varphi \text{ on toteutuva cnf-kaava} \}$
- Selvästi  $CSAT \in NP$ . Mv. Lausekalkyylin kaava voidaan muuntaa polynomisessa ajassa cnf-muotoon

- Rajoittamalla cnf-kaavan tekijöiden literaalien lukumäärä täsmälleen  $k$ :hon literaaliin saadaan  $k$ -konjunkttiivinen normaalimuoto ( $k$ -cnf)

- Kieliperhe

$$kSAT = \{ \varphi \mid \varphi \text{ on toteutuva } k\text{-cnf-kaava} \}$$

- Kieli 2SAT kuuluu luokkaan P

**Lause**  $CSAT \leq_m^P 3SAT$

#### Todistus.

- Annettu cnf-kaava  $\varphi$  voidaan muuttaa polynomisessa ajassa ekvivalentiksi 3-cnf-kaavaksi  $\varphi'$
- Olkoon  $\varphi = C_1 \wedge C_2 \wedge \dots \wedge C_m$

- Kukin tekijä  $C_k = \alpha_1 \vee \alpha_2 \vee \dots \vee \alpha_r$ ,  $r \geq 3$ , korvataan 3-cnf-kaavalla  

$$C_k' = (\alpha_1 \vee \alpha_2 \vee t_1) \wedge (\neg t_1 \vee \alpha_3 \vee t_2) \wedge \dots \wedge (\neg t_{r-3} \vee \alpha_{r-1} \vee \alpha_r),$$
missä  $t_1, \dots, t_{r-3}$  ovat uusia muuttujia. Kaava  $C_k'$  voidaan selvästi muodostaa polynomisessa ajassa tekijästä  $C_k$ .

Vielä on tarkistettava, että tämä muunnos täyttää palautusehdon  

$$\varphi \in CSAT \Leftrightarrow \varphi' \in 3SAT:$$

1.  $\varphi$  toteutuva  $\Leftrightarrow \varphi'$  toteutuva:

Kaikilla tekijöillä  $C_k$  täytyy  $\varphi$ :n toteuttavan totuusarvoasetuksen asettaa  $\alpha_i = 1$  jollakin  $\alpha_i \in C_k$ .

$C_k'$  toteutuu kun asetetaan literaalien arvot samoin kuin  $C_k$ :n toteuttavassa arvoasetuksessa ja uusien muuttujien arvot

seuraavasti

$$t_j = \begin{cases} 1, & \text{jos } j \leq i-2 \\ 0, & \text{jos } j > i-2 \end{cases}$$

2.  $\varphi$  toteutuva  $\Leftrightarrow \varphi'$  toteutuva:

Tällöin myös  $\varphi$ :n tekijöitä  $C_k$  vastaavat alikaavat  $C_k'$  toteutuvat.

Tällöin joko

- jokin literaali  $\alpha_i = 1$ ,  $\alpha_i \in C_k$ , ja  $C_k$  toteutuu sen ansiosta tai
- jollakin  $i < r-3$ :

$$t_i = 1 \wedge t_{i+1} = 0,$$

jolloin on oltava  $\alpha_{i+2} = 1$  ja tekijä  $C_k$  toteutuu.

Jos  $r \leq 3$ , niin

$$C_k = \alpha_1 \vee \alpha_2 \vee \alpha_3 \Leftrightarrow$$

$$C_k' = C_k$$

$$C_k = \alpha_1 \vee \alpha_2 \Leftrightarrow$$

$$C_k' = (\alpha_1 \vee \alpha_2 \vee t) \wedge (\alpha_1 \vee \alpha_2 \vee \neg t)$$

$$C_k = \alpha \Leftrightarrow$$

$$C_k' = (\alpha \vee t_1 \vee t_2) \wedge (\alpha \vee t_1 \alpha_2 \vee \neg t_2) \\ \wedge (\alpha \vee \neg t_1 \vee t_2) \wedge (\alpha \vee \neg t_1 \alpha_2 \vee \neg t_2)$$

Kaavojen toteutuvuuden ekvivalenssi säilyy.  $\square$

## Solmupeite, VC

Annettuna suuntaamaton verkko  $G$  ja luonnollinen luku  $k$ .

Onko  $G$ :ssä enintään  $k$ :n solmun solmujoukkoa, joka peittää kustakin verkon kaaresta vähintään toisen pään?

VC:n esittäminen formaalina kielenä edellyttää verkkojen koodaamista merkkijonoiksi. Voidaan käyttää samantapaisia koodaustekniikoita kuin Turingin koneiden kohdalla.

Arvataan annetusta verkosta  $G$  annettu määrä  $k$  solmuja ja tarkistetaan verkon koon suhteen polynomisessa ajassa, että valitut  $k$  solmua peittävät  $G$ :n kaaret

### Lause $3SAT \leq_m^p VC$

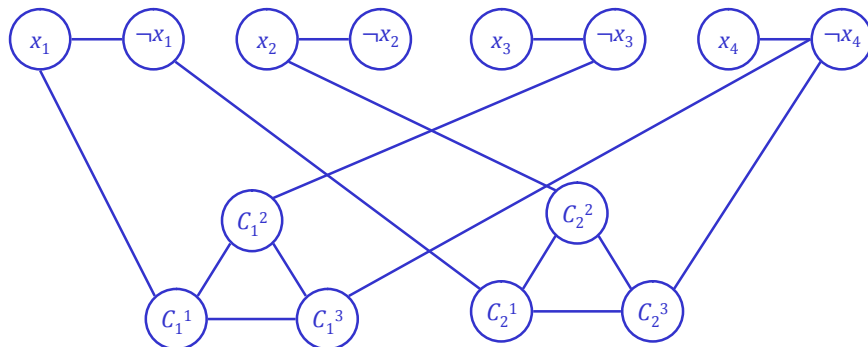
**Todistus.** Olkoon  $\varphi = C_1 \wedge C_2 \wedge \dots \wedge C_m$  3-cnf-kaava, jossa esiintyvät muuttujat  $x_1, \dots, x_n$ .

Vastaava solmupeiteongelman tapaus  $\langle G, k \rangle$  muodostetaan seuraavasti.

- $G$ :ssä on solmu kutakin literaalia kohden
- $G$ :ssä on kolme solmua  $C_j^1, C_j^2, C_j^3$  kutakin  $\varphi$ :n tekijää  $C_j$  kohden
- $G$ :ssä on kaaret:
  - $(x_i, \neg x_i)$ ,
  - $(C_j^1, C_j^2), (C_j^2, C_j^3), (C_j^3, C_j^1)$  ja
  - Jos  $C_j = \alpha_1 \vee \alpha_2 \vee \alpha_3$ , niin  $(C_j^1, \alpha_1), (C_j^2, \alpha_2), (C_j^3, \alpha_3)$
- $k = n + 2m$

Selvästi verkko  $G$  voidaan muodostaa kaavasta  $\varphi$  polynomisessa ajassa.

Kaavan  $F = (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee x_2 \vee \neg x_4)$  verkko



1.  $\varphi$  toteutuva  $\Leftrightarrow$

$G$ :llä on enintään  $k = n + 2m$  solmun solmupeite:

- Otetaan totuusarvoasetusta vastaavaan solmupeitteeseen kutakin muuttujaa vastaten sitä literaalia, joka saa arvon 1, vastaava solmu ( $n$  solmua)
- Jokaista tekijää  $C_j$  vastaavasta kolmiosta on yhdestä kulmasta alkava kaari  $(C_j^i, \alpha_i)$  nyt peitetty
- Otetaan solmupeitteeseen vielä kolmion kaksi muuta kulmaa (kaikkiaan  $2m$  solmua)

2.  $G$ :llä on enintään  $k$ :n solmun solmupeite  $\Leftrightarrow \varphi$  toteutuva:

- Olkoon  $V'$ ,  $|V'| \leq k$ ,  $G$ :n solmupeite
- Jotta  $V'$  voisi peittää kaikki  $G$ :n kaaret, on sen sisällettävä yksi solmu kutakin muuttujaa kohden ja vähintään kaksi solmua kustakin  $C_j$ -kolmiosta
- Näin ollen  $|V'| = k$
- Asetetaan

$$t(x_i) = \begin{cases} 1, & \text{jos } x_i \in V' \\ 0, & \text{jos } \neg x_i \in V' \end{cases}$$

- Jokaisen  $C_j$ -kolmion kärjistä alkavista kaarista yhden peittää literaalisolmu  $\alpha \in V'$
- Tällöin  $t(\alpha) = t(C_j) = 1$  □

- Siis  $SAT \leq_m^p CSAT \leq_m^p 3SAT \leq_m^p VC$
- Kieli  $B$  on  $NP$ -täydellinen, jos
  1.  $B \in NP$  ja
  2.  $A \leq_m^p B$  kaikilla  $A \in NP$
- $NP$ -täydellinen kieli voidaan tunnistaa deterministisesti polynomisessa ajassa jos ja vain jos kaikki muutkin luokan  $NP$  kielet voidaan tunnistaa deterministisesti polynomisessa ajassa

**Lause 7.35** *Olkoon  $B$   $NP$ -täydellinen kieli ja  $B \in P$ . Tällöin  $P = NP$ .*

**Lause 7.36** Olkoon  $B$   $NP$ -täydellinen kieli,  $C \in NP$  ja  $B \leq_m^p C$ .

Tällöin myös  $C$  on  $NP$ -täydellinen.

**Todistus.** Koska  $B$  on  $NP$ -täydellinen, niin määritelmän mukaan  $A \leq_m^p B$  kaikilla  $A \in NP$ . Toisaalta  $B \leq_m^p C$ , joten polynomisen palautuksen transitiivisuuden perusteella (lause 7.31) pätee  $A \leq_m^p C$  kaikilla  $A \in NP$ . Oletuksen perusteella  $C \in NP$ , joten väite pätee.  $\square$

- Jos siis halutaan osoittaa  $C$   $NP$ -täydelliseksi kieleksi, riittää muodostaa polynominen palautus tunnetusta  $NP$ -täydellisestä kielestä  $B$  kieleen  $C$  ja lisäksi todeta, että  $C \in NP$
- Ensin pitäisi todeta jokin kieli  $NP$ -täydelliseksi