Tilavaativuus I

Määritelmä

Olkoon M deterministinen Turingin kone, joka pysähtyy kaikilla syötteillä. M:n tilavaativuus on funktio $f : \mathbf{N} \longrightarrow \mathbf{N}$, missä $f(n)$ on M:n selaamien muistipaikkojen maksimilukumäärä, kun tarkastellaan kaikkia n:n pituisia syötteitä. Jos M:n tilavaatimus on $f(n)$, niin sanotaan myös, että M suorittaa (tehtävän) tilassa $f(n)$.

Jos M on epädeterminstinen Turingin kone, jossa kaikki haarat pysähtyvät kaikilla syötteillä, niin tilavaatimus $f(n)$ tarkoittaa M:n läpikäymien muistipaikkojen maksimilukumäärää, kun tarkastellaan kaikkia n:n mittaisia syötteitä ja kaikkia M:n laskennan haaroja.

Tilavaativuusluokat I

Määritelmä

Olkoon $f : \mathbf{N} \longrightarrow \mathbf{R}$ *funktio. Tilavaativuusluokat* $SPACE(f(n))$ *ja* $NSPACE(f(n))$ *määritellään seuraavasti:*

- $SPACE(f(n))$ *on niiden kielten* L *joukko, jotka voidaan tunnistaa tilassa* $\mathcal{O}(f(n))$ *deterministisellä Turingin koneella.*
- $NSPACE(f(n))$ *on niiden kielten* L *joukko, jotka voidaan tunnistaa tilassa* $\mathcal{O}(f(n))$ *epädeterministisellä Turingin koneella.*

Example (SAT-ongelman tilavaativuus)

SAT voidaan ratkaista lineaarisessa tilassa:

1. Käy läpi kaikki totuusarvoasetukset ϕ:n muuttujille $x_1, ..., x_n$.
2. Evaluoi ϕ annetulla totuusarvoasetuksella.
3. Jos ϕ tuli joskus todeksi, *hyväksy*, muuten *hylkää*.

Yleisesti kuitenkin uskotaan, ettei SAT-ongelmaa voida ratkaista polynomisessa ajassa, puhumattakaan lineaarisesta ajasta.

Savitchin lause

Lause

Kaikilla funktioilla $f : \mathbf{N} \longrightarrow \mathbf{R}^+$, *missä* $f(n) \geq n$,

$$\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n)).$$

Todistuksen idea I

- Todistus on simulointitodistus: täytyy näyttää, että epädeterminististä Turingin konetta, joka toimii tilassa $f(n)$, voidaan simuloida deterministisellä koneella tilassa $f^2(n)$.
- Varsinainen simulointi on aika teknistä, joten seuraavassa tyydytään vain antamaan simuloinnin idea.
- Simuloinnissa täytyy pitää kirjaa siitä, missä epädeterministisessä haarassa kulloinkin ollaan. Jokaisessa haarassa edetään aina yksi askel kerrallaan.
- Mutta haara, joka käyttää $f(n)$:n verran tilaa, voi kuluttaa $2^{\mathcal{O}(f(n))}$ aikaa, ja jokainen askel voi olla seurausta epädeterministisestä valinnasta.
- Jos haarat tutkitaan peräkäisessä järjestyksessä, kaikki epädeterministiset valinnat täytyy kirjata, jotta seuraava haara löydetään. Tämä saattaa vaatia tilaa $2^{\mathcal{O}(f(n))}$, mikä ylittää lauseessa sallitun.

Todistuksen idea II

- Tarkastellaan sen sijaan yleisempää ongelmaa. On annettu kaksi epädeterministisen koneen konfiguraatiota c_1 ja c_2 sekä luku t, ja täytyy ratkaista, päästäänkö c_1:stä c_2:een t:llä askeleella. Kutsutaan tätä ongelmaa saavuttamisongelmaksi.
- Jos saavuttamisongelma on ratkaistu, niin asettamalla c alkutilanteeksi, c_2 hyväksymistilanteeksi ja t sallituksi askelmääräksi voidaan ratkaista, hyväksyykö kone syötteen.
- Voidaan konstruoida deterministinen algoritmi, joka ratkaisee saavuttamisongelman. Se toimii etsimällä välikonfiguraation c_m ja rekursiivisesti ratkaisee, päästäänkö c_1:stä c_m:ään $t/2$:lla askeleella ja c_m:stä c_2:een $t/2$:lla askeleella.
- Rekursiivisiin kutsuihin kuluvaa tilaa käytetään uudestaan, jolloin saavutetaan huomattavaa säästöä tilan käytössä.
- Algoritmi tarvitsee tilaa varastoidakseen rekursiopinon. Rekursion jokainen taso käyttää $\mathcal{O}(f(n))$ tilaa konfiguraation tallettamiseen.

Todistuksen idea III

- Rekursion syvyys on $\log t$, missä t on maksimiaika, jonka epädeterministinen kone käyttää missään laskentahaarassaan.
- Nyt $t = 2^{\mathcal{O}(f(n))}$, joten $\log t = \mathcal{O}(f(n))$.
- Siten deterministinen simulointi käyttää $\mathcal{O}(f(n))$ tilaa. "$\square$"

PSPACE ja EXPTIME

Määritelmä

PSPACE on niiden kielten luokka, jotka voidaan tunnistaa polynomisessa ajassa deterministisellä Turingin koneella. Eli

$$\mathrm{PSPACE} = \bigcup_k \mathrm{SPACE}(n^k).$$

Huom: Savitchin lauseen johdosta ei ole tarpeen käsitellä epädeterminististä versiota luokasta PSPACE.

Määritelmä

EXPTIME on niiden kielten luokka, jotka voidaan tunnistaa eksponentiaalisessa ajassa deterministisellä tai epädetrministisellä Turingin koneella. Eli

$$\mathrm{EXPTIME} = \bigcup_k \mathrm{TIME}(2^{n^k}).$$

Luokkien keskinäiset suhteet I

Lause

$$\mathrm{P} \subseteq \mathrm{NP} \subseteq \mathrm{PSPACE} = \mathrm{NSPACE} \subseteq \mathrm{EXPTIME}.$$

Todistus. Ensimmäinen kuuluvuusrelaatio on selvä. Samoin tilaluokkien yhtäsuuruus Savitchin lauseen perusteella. Tarkastellaan muita lähemmin:

- $\mathrm{P} \subseteq \mathrm{PSPACE}$. Jos $t(n) \geq n$, niin jokainen ajan $t(n)$ kuluttava kone käyttää tilaa korkeintaan $t(n)$ muistipaikan verran, koska kone voi tutkia vain yhden muistipaikan yhdellä askeleella.
- Samoin NP$\subseteq$NSPACE, joten siis NP$\subseteq$PSPACE.
- Tutkitaan nyt luokkaan EXPTIME kuulumista. Jos Turingin koneen tilavaativuus on $f(n)$, missä $f(n)n$, niin sillä on korkeintaan $f(n)2^{\mathcal{O}(f(n))}$ erilaista konfiguraatiota. (Tämä vaatisi hieman tarkempaa perustelua.)

Luokkien keskinäiset suhteet II

- Jos Turingin kone pysähtyy, se ei välttämättä toista samaa konfiguraatiota. Siten aikaa kuluu pahimmillaan $f(n)2^{\mathcal{O}(f(n))}$, joten PSPACE $\subseteq$ EXPTIME. $\square$
- Ei tiedetä, ovatko edellisen lauseen luokkien kuuluvuudet aitoja vai ei.
- On kuitenkin todistettu, että P $\neq$ EXPTIME.
- Siten jonkin kuuluvuusrelaatioista on oltava aito, muttemme tiedä, mikä!
- Useimmat uskovat, että kaikki ovat aitoja.

PSPACE-täydellisyys

Määritelmä

Kieli B on PSPACE-täydellinen, jos se täyttää seuraavat ehdot:

1. *B kuuluu PSPACE:hen ja*
2. *jokainen PSPACE:n kieli A on palautettavissa polynomisessa ajassa B:hen.*

Jos B täyttää vain ehdon 2, sanotaan, että B on PSPACE-kova.

Huom: PSPACE-täydellisyyden yhteydessä käytetään myös polynomisen ajan palautusta, ei polynomisen tilan palautusta. Näin siksi, että palautuksen on oltava helposti laskettavissa, mitä polynomisen tilan vaativa palautus ei automaattisesti ole.

TQBF-ongelma I

Määritellään ensimmäinen PSPACE-täydellinen ongelma, mutta ei käydä sen todistusta läpi tarkasti, vaan vain ideatasolla.

- Boolen kaava on kaava, jossa esiintyy Boolen muuttujia, vakiot 0 ja 1, sekä loogisia konnektiiveja $\wedge$, $\vee$ ja $\neg$.
- Boolen kaavojen yhteydessä esiintyvät myös kvanttorit $\exists$, $\forall$. Siten $\forall x\phi$ väittää, että kaikilla x:n arvoilla ϕ on tosi. Kaava $\exists x\phi$ puolestaan sanoo, että jollakin x:n arvolla ϕ on tosi.
- Muuttuja, joka seuraa välittömästi kvanttoria, on sidottu (bound) kvanttoriin.
- Jos tarkastellaan luonnollisia lukuja, niin kaava $\forall x[x + 1 > x]$ on ilmeisen tosi. Sen sijaan kaava $\exists y[y + y = 3]$ on epätosi. Siten kvanttoreita sisältävän kaavan totuus riippuu siitä universumista, josta muuttujat saavat arvonsa.

TQBF-ongelma II

- Kvanttorilauseet voivat sisältää useita kvanttoreita, kuten $\forall x \exists y [y > x]$. Kvanttoreiden järjestys on oleellista.
- Kvanttori voi esiintyä missä tahansa kohdassa lausetta. Kvanttorin vaikutusala (scope) voidaan rajata suluilla. Jos kaikki kvanttorit esiintyvät lauseen alussa ja niiden vaikutusala on koko lause, lauseen sanotaan olevan preneksissä normaalimuodossa. Jokainen lause voidaan helposti muuttaa preneksiin normaalimuotoon.
- Kvanttoreilla varustettuja Boolen kaavoja kutsutaan kvantifioiduiksi Boolen kaavoiksi. Tällaisten kaavojen universumi on $\{0, 1\}$. Esimerkiksi

$$\phi = \forall x \exists y \left[(x \vee y) \wedge (\bar{x} \vee \bar{y}) \right]$$

on kvantifioitu Boolen kaava.

TQBF-ongelma III

- Jos kaavan jokainen muuttuja kuuluu jonkin kvanttorin vaikutusalueeseen, sanotaan, että kaava on täysin kvantifioitu. Tällainen kaava on aina joko tosi tai epätosi.
- Esimerkiksi edellä oleva ϕ on täysin kvantifioitu. Jos sen sijaan $\forall x$ poistettaisiin kaavasta, kaava ei olisi enää täysin kvantifioitu eikä se olisi sen paremmin tosi kuin epätosikaan.

Määritelmä

TQBF on täysin kvantifioitujen tosien *Boolen kaavojen joukko.*

Lause

TQBF on PSPACE-täydellinen.

Todistus. Sivuutetaan.

Pelien voittostrategiat

- Olkoon

$$\phi = \exists x_1 \forall x_2 \exists x_3 \cdots Q x_k [\psi]$$

  kvantifioitu Boolen kaava preneksissä normaalimuodossa. Q tarkoittaa joko kaikki- tai olemassaolokvanttoria.
- Pelaajat A ja E pelaavat kaavapeliä.
- He valitsevat vuorollaan muuttujille $x_1, ..., x_k$ totuusarvon.
- A valitsee muuttujia, jotka on sidottu $\forall$-kvanttoriin, ja E muuttujia, jotka on sidottu $\exists$-kvanttoriin.
- Valintajärjestys on sama kuin kvanttorien järjestys kaavan alussa.
- E voittaa, jos valintojen lopussa ψ on tosi. Muuten A voittaa.

Pelien voittostrategiat II

Example

Tarkastellaan kaavaa

$$\phi_1 = \exists x_1 \forall x_2 \exists x_3 \left[(x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (\bar{x_2} \vee \bar{x_3}) \right].$$

- E antaa arvon x_1:lle, A x_2:lle ja lopuksi E x_3:lle.
- Jos $x_1 = 1$, $x_2 = 0$ ja $x_3 = 1$, niin alikaava

$$(x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (\bar{x_2} \vee \bar{x_3})$$

on tosi ja E voittaa.

Pelien voittostrategiat III

Example (jatkuu)

- Itse asiassa valitsemalla $x_1 = 1$ E voittaa aina, kun hän asettaa lopuksi x_3:n päinvastoin kuin x_2. E:llä on siis voittostrategia.
- Sen sijaan A:llä on voittostrategia kaavan

$$\phi_2 = \exists x_1 \forall x_2 \exists x_3 \left[(x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (x_2 \vee \bar{x_3}) \right]$$

tapauksessa, kun hän valitsee $x_2 = 0$.

Pelien voittostrategiat IV

Määritelmä

Formula-Game on niiden kvantifioitujen Boolen kaavojen joukko, joille E:llä on voittostrategia.

Lause

Formula-Game on PSPACE-täydellinen.

Pelien voittostrategiat V

Todistus.

- Formula-Game on itse asiassa sama probleema kuin TQBF. Osoitetaan tämä täsmällisesti.
- Kaava $\phi = \exists x_1 \forall x_2 \exists x_3 \cdots [\psi]$ on *tosi*, jos löytyy sellainen x_1:n totuusarvo, että kaikilla x_2:n totuusarvoilla löytyy x_3:n totuusarvo jne joilla kaava ψ on tosi. Mutta tämä on täsmälleen sama asia kuin että E:llä on voittostrategia.
- Sama päättely sopii myös tilanteseen, jossa kaikki- ja olemassaolokvanttorit eivät vuorottele jokaisen muuttujan jälkeen.
- Esimerkiksi kaavan

$$\phi = \forall x_1, x_2, x_3 \exists x_4, x_5 \forall x_6 [\psi]$$

  tapauksessa A valitsee totuusarvot muuttujille x_1, x_2 ja x_3, sen jälkeen E käsittelee muuttujat x_4 ja x_5 ja lopuksi A muuttujan x_6.
- Siis $\phi \in$TQBF täsmälleem silloin kun $\phi \in$Formula-Game. $\square$

Yleistetty maantieto

Kysymyksessä on seuraava peli:

- On annettu suunnattu verkko G ja yksi sen aloitussolmu s.
- Kaksi pelaajaa merkkaavat solmuja vuorotellen lähtien solmusta s. Seuraavan solmun täytyy muodostaa *yksinkertainen polku* jo valittujen solmujen kanssa (eli sama solmu ei saa toistua polulla).
- Se pelaaja, joka ei voi jatkaa peliä, häviää.

Yleistetty maantieto II

Määritelmä

GG koostuu sellaisista yleistetyn maantiedon tapauksista (G, s), joissa pelaajalla I on voittostrategia.

Lause

GG on PSPACE-täydellinen.

Todistus I

Seuraava algoritmi ratkaisee, onko pelaajalla I voittostrategiaa:

1. Jos s:stä ei lähde kaaria, hylkää, koska I häviää heti.
2. Poista solmu s ja kaikki siihen liittyvät kaaret, jolloin saadaan uusi verkko G_1.
3. Kutsu algoritmia rekursiivisesti kaikista solmuista $s_1, s_2, \ldots, s_k$, joihin s:stä oli kaari.
4. Jos kaikki nämä johtavat hyväksymiseen (verkossa G_1), *hylkää* (koska II voittaa tällöin). Muussa tapauksessa I:n täytyy voittaa, joten hyväksy.

Todistus II

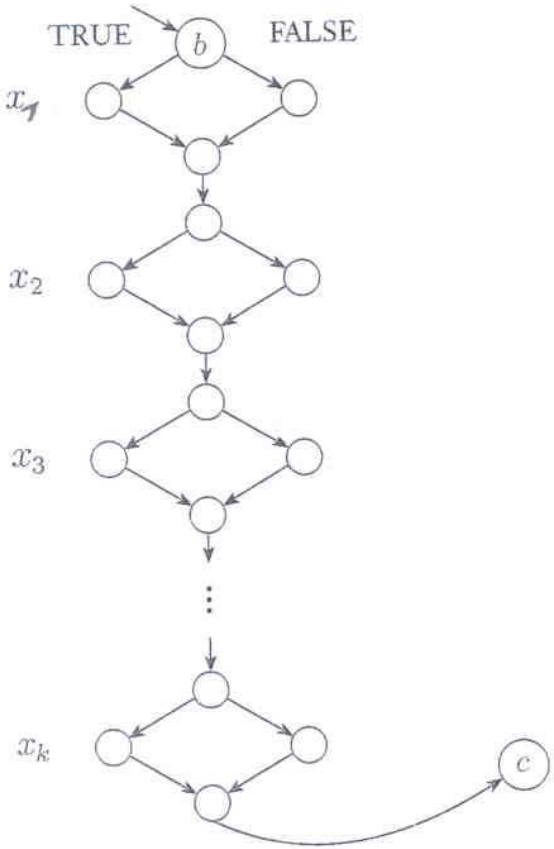
- Algoritmin tarvitsee tallettaa vain rekursiopino.
- Jokainen rekursiokutsu lisää yhden solmun pinoon. Siten pinon koko on korkeintaan m, missä m on G:n solmujen lukumäärä.
- Siten algoritmi toimii lineaarisessa tilassa.
- Seuraavaksi osoitetaan, että Formula-Game voidaan palauttaa polynomisessa ajassa GG:hen.
- Eli osoitetaan, että kaava

$$\phi = \exists x_1 \forall x_2 \exists x_3 \cdots Q x_k [\psi]$$

  voidaan kuvata GG:n tapaukselle (G, b).
- Oletetaan, että ϕ:n kvanttorit alkavat $\exists$:lla ja seuraavat vaihtuvassa järjestyksessä jokaisen muuttujan jälkeen. Jos kaava ei ole tällainen, se voidaan muuttaa ekvivalentiksi kaavaksi lisäämällä kvanttoreita ja apumuuttujia.
- Oletetaan lisäksi, että ψ on konjunktiivisessa normaalimuodossa.

Todistus III

Ryhdymme nyt konstruoimaan kaavaa ϕ vastaavaa verkkoa. Verkko koostuu kahdesta osasta, vasemmasta ja oikeasta. Seuraava kuva näyttää, minkälainen on vasen puolisko:



Todistus IV

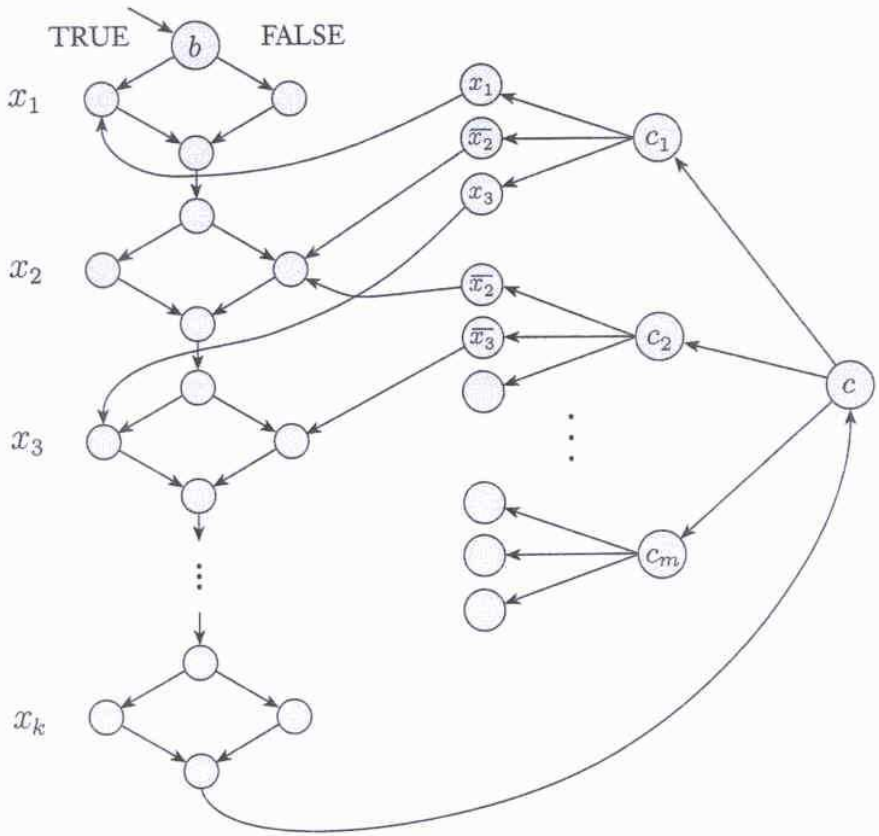
Pelin kulku etenee verkossa seuraavasti:

- Pelaaja I aloittaa solmusta b. Hänen täytyy valita jompikumpi kahdesta kaaresta, jotka edustavat E:n kahta mahdollista totuusarvovalintaa.
- Vasen kaari vastaa arvoa tosi, oikea arvoa epätosi.
- Tämän jälkeen II jatkaa ja hänellä on vain yksi mahdollisuus edetä.
- Samoin I:llä on vain yksi mahdollisuus. Nyt vuorostaan II:lla on kaksi mahdollisuutta valita I:n jälkeen.
- Peli etenee tällä tavalla I:n ja II:n vuorotellessa kunnes päädytään viimeiseen solmuun alhaalla. Tästä on siirtymä verkon oikeanpuoleiseen osaan, jonka ensimmäinen solmu on c.
- Solmussa c vuorossa on pelaaja II, sillä viimeinen kvanttori oli oletuksen mukaan eksistenssikvanttori.

Todistus V

Seuraava kuva näyttää verkon koko rakenteen, kun kaava on muotoa

$$\phi = \exists x_1 \forall x_2 \cdots Q x_k \left[(x_1 \vee \bar{x}_2 \vee x_3) \wedge (\bar{x}_2 \vee \bar{x}_3 \vee \cdots) \wedge \cdots \wedge (\ \ ) \right].$$



Todistus VI

- Solmussa c pelaaja II valitsee jonkin tekijän c_i.
- Jos kaava ϕ on epätosi, II voittaa valitsemalla tekijän, joka ei tule todeksi valitussa totuusarvoasetuksessa.
- Tällöin siis jokainen literaali, jonka I valitsee seuraavaksi, on epätosi ja se on yhdistetty siihen timantin puoliskoon, jota ei vielä ole pelattu. Siis II voi jatkaa, mutta II:n jälkeen I ei enää voi jatkaa.
- Jos ϕ on tosi, jokaisessa tekijässä on vähintään yksi literaali tosi, joten sellainen on yhdistetty timantin oikeaan puoliskoon, joka on jo pelattu. Siis I voi valita, muttei II.

□