Aalto-yliopisto
Perustieteiden korkeakoulu
Tietotekniikan koulutusohjelma

Kaarivärityssongelma

Kandidaatintyö

29. huhtikuuta 2012

Sami J. Lehtinen

Aalto-yliopisto
Perustieteiden korkeakoulu
Tietotekniikan koulutusohjelma

KANDIDAATINTYÖN
TIIVISTELMÄ

Tekijä:	Sami J. Lehtinen
Työn nimi:	Kaariväritysongelma
Päiväys:	29. huhtikuuta 2012
Sivumäärä:	22
Pääaine:	Tietojenkäsittelytiede
Koodi:	IL3010
Vastuuopettaja:	Ma professori Tomi Janhunen
Työn ohjaaja(t):	Professori Petteri Kaski (Tietojenkäsittelytieteen laitos)

Työssä esitellään taustat tehokkaiden kaariväritysalgoritmien löytämisen vaikeudelle ja käydään läpi ongelman historiaa. Vizingin lause jakaa verkot luokkiin 1 ja 2 sen mukaan ovatko ne kaariväritettävissä $\Delta(G)$ vai $\Delta(G) + 1$ värillä. Vaikka jako onkin yksinkertainen, kyseessä on hyvin vaativa, NP-täydellinen ongelma.

Työssä tehdään katsaus NP-täydellisyyden historiaan ja esitellään Holyerin todistus kaariväritysongelman NP-täydellisyydestä. Todistuksessa kaariväritysongelma palautetaan 3SAT-ongelmasta – klausuulijoukosta C muodostetaan kääntö-, muuttujanasetus- ja tarkastuskomponenttien avulla verkko, joka on 3-väritettävissä jos ja vain jos klausuulijoukko C on toteutuva. Nykyisen tiedon perusteella ongelmalle ei siis tunneta tehokasta polynomisessa ajassa toimivaa ratkaisualgoritmia.

Työ esittelee tällä hetkellä nopeimman tunnetun eksaktin algoritmin kaariväritykselle. Ensin verkko G muunnetaan viivaverkoksi $L(G)$, minkä jälkeen lasketaan k-peittojen lukumäärä käyttäen dynaamista ohjelmointia. Jos n-kaariselle verkolle G on k-kaariväritys, se löydetään ajassa $O(2^n nk \operatorname{polylog} nk)$ ja tilassa $O(n2^n)$. Lopuksi käydään läpi kaariväritysongelman helpompia erikoistapauksia, joiden tarkastelu rajataan kaksijakoisiin verkkoihin ja tasoverkkoihin. Kaksijakoisille verkoille kaariväritys $\Delta(G)$ värillä voidaan ratkaista ajassa $O(m \log \Delta(G))$, jossa m on verkon kaarien lukumäärä. Tasoverkkojen kaariväritys voidaan ratkaista lineaarisessa ajassa $\max\{\Delta(G), 9\}$ värillä.

Avainsanat:	verkkoteoria, graafiteoria, kaariväritysongelma, NP-täydellisyys
Kieli:	Suomi

Aalto University
School of Science
Degree Program of Computer Science and Engineering

ABSTRACT OF
BACHELOR'S THESIS

Author:	Sami J. Lehtinen
Title of thesis:	Edge-coloring problem
Date:	April 29, 2012
Pages:	22
Major:	Information and Computer Science
Code:	IL3010
Supervisor:	Professor (pro tem) Tomi Janhunen
Instructor:	Professor Petteri Kaski (Department of Information and Computer Science)

In this thesis, we go through the background of the edge-coloring problem – the history of the problem and why it is hard to find efficient algorithms for edge-coloring. Vizing's theorem divides graphs into two distinct classes: graphs of class 1, which have an edge-coloring with $\Delta(G)$ colors, and graphs of class 2, which have an edge-coloring with $\Delta(G) + 1$ colors. Even though the division seems simple, we are dealing with a very hard problem – in fact, the problem is NP-complete.

We summarize the history of NP-completeness and present Holyer's proof on the NP-completeness of the edge-coloring problem. In the proof, the 3SAT-problem is reduced to the edge-coloring problem, thus at present there are no known efficient polynomial-time algorithms for solving the problem. A set of clauses C from the 3SAT-problem are used to create a graph with special components for inverting, setting variables, and checking satisfiability. The graph thus created is 3-colorable if and only if the set of clauses C is satisfiable.

We present the fastest presently known exact algorithm for edge-coloring. First, the graph G is transformed into a line graph $L(G)$, after which we compute the number of k-covers using dynamic programming. If a graph G with n edges has a k-coloring, it will be found in $O(2^n nk \operatorname{polylog} nk)$ time and in $O(n2^n)$ space. Finally, we go through some special cases for the edge-coloring problem. We limit our examination to bipartite graphs and planar graphs. Bipartite graphs can be edge-colored with $\Delta(G)$ colors in $O(m \log \Delta(G))$ time, where m is the number of edges in the graph. Planar graphs can be edge-colored in linear time with $\max\{\Delta(G), 9\}$ colors.

Key words:	graph theory, edge-coloring, NP-completeness
Language:	Finnish

Sisältö

1 Johdanto

Verkon kaarivärityksessä solmujen väliset kaaret väritetään siten, että yhdenkään kaaren vieressä ei ole toista samanväristä kaarta – yhdestäkään solmusta ei siis lähde kahta samanväristä kaarta. Verkkojen solmuväritys ja kaariväritys ovat ongelmina läheisiä – polynominen algoritmi solmuväritykselle ratkaisisi myös kaarivärityksen polynomisessa ajassa, koska verkon G kaariväritysongelman voi palauttaa viivaverkon (engl. *line graph*) $L(G)$ solmuväritysongelmaksi; tämä palautus esitetään työn luvussa 7.1. Tässä työssä keskitytään käsittelemään vain yksinkertaisia ja suuntaamattomia verkkoja. Työn kannalta keskeiset määritelmät käydään läpi luvussa 2.

Verkkojen kaarivärittäminen ratkaisee käytännön ongelman joukkueiden otteluaikataulujen luomisesta tai valokaapeliverkon taajuusalueiden jakamisesta. Jotta kaariväritystä voisi soveltaa aikataulutukseen, tulee sovelluskohteessa olla luonnollisesti pariutuvia asioita, esimerkiksi haastatteluja kahden ihmisen välillä, joukkueita joukkueaikataulussa tai kahden verkkolaitteen välisiä kommunikaatiotaajuuksia.

Nakano, Zhou, ja Nishizeki (1995) taustoittivat kaarivärityksen historiaa. Kaariväritysongelma esitettiin vuonna 1880 liittyen karttojen neliväritysongelmaan, joka tunnetaan nykyisin neliväritysteoreemana; sen todistivat vuonna 1976 Appel ja Haken (Diestel 2010, s. 143). Ensimmäisessä artikkelissa kaariväritysongelmasta, joka julkaistiin 1889, Tait todisti että karttojen neliväritysongelma on ekvivalentti kaikkien 3-kytkettyjen 3-säännöllisten tasoverkkojen kaariväritykselle kolmella värillä. König todisti vuonna 1916, että kaksijakoiset verkot voi aina kaarivärittää $\Delta(G)$ värillä (Diestel 2010, s. 125). Shannon todisti vuonna 1949, että kaikki verkot voidaan kaarivärittää $3\Delta(G)/2$ värillä. (Nakano et al. 1995)

Luvussa 3 käydään läpi Vizingin lause, jonka perusteella kaikki yksinkertaiset verkot voidaan jakaa kahteen luokkaan eli verkkoihin, jotka voidaan kaarivärittää verkon maksimiasteen $\Delta(G)$ ilmoittamalla määrällä värejä, ja verkkoihin, joille on kaariväritys $\Delta(G) + 1$ värillä. Vaikka jako kuulostaa yksinkertaiselta, optimaalisen kaarivärityksen löytäminen on hyvin vaativa ongelma. Itse asiassa ongelma kuuluu NP-täydellisten ongelmien luokkaan.

Garey ja Johnson (1979) esittelivät Cookin teoreeman (Cook 1971) toteutuvuusongelman NP-täydellisyydestä. Palauttamalla SAT-ongelma 3SAT-ongelmaan todistetaan myös jälkimmäinen NP-täydelliseksi (Garey ja Johnson 1979). Holyer (1981) todisti kaariväritysongelman NP-täydelliseksi palauttamalla sen 3SAT-ongelmasta. Tämän perusteella on todennäköistä, että kaariväritysongelmaan, kuten muihinkaan NP-täydellisiin ongelmiin, ei ole olemassa tehokasta algoritmia (Cormen, Leiserson, ja Rivest 1990). Todistusta tarkastellaan luvussa 6; lyhyt johdanto laskennan vaativuuden teoriaan ja 3SAT-ongelmaan esitellään luvuissa 4 ja 5.

Työ on kirjallisuustutkimus kaarivärityksen teoriasta ja historiasta. Tuoreimpien tutkimustulosten saralta työn luvussa 7 esitellään tällä hetkellä nopein tunnettu eksakti algoritmi kaariväritykselle. Algoritmin aikavaativuus k-väritykselle n-kaariselle verkolle on luokkaa $O(2^n nk \operatorname{polylog} nk)$ (Björklund, Husfeldt, ja Koivisto 2009); on helppo havaita, että esimerkiksi 100-kaarisen verkon eksaktin kaarivärityksen selvittäminen on täysin nykytietokoneiden kapasiteetin tavoittamattomissa. Luvussa 8 tehdään katsaus helpompiin erikoistapauksiin, joilla ongelma ratkeaa polynomisessa ajassa tai jopa lineaarisesti.

2 Työssä käytettäviä termejä

Verkko G muodostuu joukosta solmuja V ja joukosta kaaria $E \subseteq \{\{x, y\} | x, y \in V \text{ ja } x \neq y\}$ ja missä $V \cap E = \emptyset$. Joukon E alkiot ovat siis joukon V kaksialkioisia alijoukkoja (Diestel 2010, s. 2). Verkon G solmujoukkoon viitataan merkinnällä $V(G)$ ja kaarijoukkoon merkinnällä $E(G)$ (Diestel 2010, s. 2). Tämän määritelmän perusteella muodostetut verkot ovat aina *yksinkertaisia* (engl. *simple*) eli kahden solmun välillä on vain yksi kaari. Todetaan myös, että koska solmujen väliset kaaret kuvataan kahden solmun joukkoina, kuvattu verkko on myös suuntaamaton.

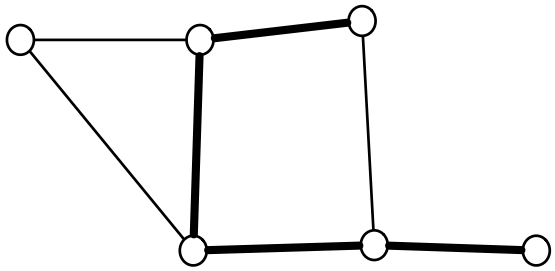
Solmun x *naapureita* (engl. *neighbor*) ovat kaikki ne solmut y, joihin on kaari xy joukossa $E(G)$. Kaaren xy naapureita ovat kaikki kaaret $yz \in E(G)$, toisin sanoen kaaret jakavat päätepisteen.

Verkon G *solmuväritys* on kuvaus $c : V(G) \to S$, missä S on joukko värejä. Merkinnällä $c(v)$ ilmoitetaan siis solmun v väri. Vastaavasti verkon G *kaarivärits* on kuvaus $c : E(G) \to S$ ja merkintä $c(xy)$ tarkoittaa kaaren xy väriä.

Verkon G *kromaattinen luku* $\chi(G)$ on vähimmäismäärä värejä, joilla verkon solmut voidaan värittää siten, että kullakin solmulla on vain erivärisiä naapureita. Vastaavasti verkon G *kaarikromaattinen luku* $\chi'(G)$ on vähimmäismäärä värejä, joilla verkon kaaret voidaan värittää siten, että kullakin kaarella on vain erivärisiä naapureita.

Verkon *polku* (engl. *path*) P on verkosta G muodostettu aliverkko solmuista V' ja kaarista E', jossa $V' = \{x_0, x_1, \ldots, x_k\} \subseteq V(G)$ ja $E' = \{x_0x_1, x_1x_2, \ldots, x_{k-1}x_k\} \subseteq E(G)$ ja jossa kaikki solmut x_i ovat eri solmuja. Joukon E' kaaret siis yhdistävät joukon V' solmut toisiinsa siten, että seuraamalla kaaria päätepisteistä x_0 tai x_k päädytään toiseen päätepisteeseen. Tässä työssä polulla tarkoitetaan aina yksinkertaista polkua, jossa ei ole syklejä. Kuvassa 1 on korostettu eräs verkon polku. (Diestel 2010, s. 6–7)

Solmun v *aste* (engl. *degree*) $d(v)$ on solmusta lähtevien kaarien lukumäärä. Verkon G maksimiaste merkitään $\Delta(G)$. Jos kaikille verkon solmuille pätee $d(v) = k$, sanotaan verkkoa *k-säännölliseksi* (engl. *k-regular*); esimerkiksi tässä työssä käsitellään 3-säännöllisiä verkkoja. (Diestel 2010, s. 5)



Kuva 1: Polku.

Joukko solmuja verkossa G muodostaa *riippumattoman joukon* (engl. *independent set*), jos mitkään joukon solmuista eivät ole toistensa naapureita.

Tasoverkko tai tasograafi (engl. *planar graph*) on verkko, joka voidaan upottaa tasoon siten, että verkon kaaren kohtaavat vain päätepisteissään, toisin sanoen yksikään verkon kaari ei risteä toisen kaaren kanssa.

Monikko (engl. *tuple*) esitetään notaatiolla $(a_1, a_2, .., a_k)$. Koska kyseessä olevassa monikossa on k alkiota, siitä voidaan käyttää myös nimitystä k-monikko. Monikon alkioiden järjestyksellä on merkitystä; esimerkiksi monikot (a_1, a_2, a_3) ja (a_1, a_3, a_2) ovat eri monikkoja.

Turingin kone (engl. *Turing machine*) on tietokoneen yksinkertaistettu matemaattinen malli, jolla on nauha eli muisti sekä tila. Toisaalta Turingin koneen muistia tai tilojen määrää ei ole rajattu mitenkään. Turingin kone on monikko $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}})$. Joukko Q kuvaa koneen eri tiloja ja $q_0 \in Q$ on koneen alkutila. Joukko Σ, koneen *syöteaakkosto*, on äärellinen joukko symboleja, jotka kone ymmärtää. Tyhjä symboli $\sqcup$ ei kuulu syöteaakkostoon Σ. *Nauha-aakkosto* Γ pitää sisällään tyhjän symbolin $\sqcup$ ja syöteaakkoston Σ. Symbolit $q_{\text{acc}} \in Q$ ja $q_{\text{rej}} \in Q$, missä $q_{\text{acc}} \neq q_{\text{rej}}$, kuvaavat koneen hyväksyvää ja hylkäävää lopputilaa. Kone saa syötteensä nauhan vasemmasta laidasta lukien nauhan lukumäärältään n ensimmäisessä "ruudussa", missä n on syötteen pituus. Koska tyhjä symboli $\sqcup$ ei kuulu syöteaakkostoon, ensimmäinen kohdattu tyhjä symboli merkitsee syötteen päättymistä. Siirtymäfunktio $\delta : Q \times \Gamma \to Q \times \Gamma \times \{\text{L}, \text{R}\}$ kuvaa siirtymät tilojen välillä. Siirtymäfunktion syöte on lukupään alla oleva nauhan symboli ja nykyinen tila. Tuloksena saadaan nauhalle kirjoitettava symboli, uusi tila ja suunta, johon lukupäätä siirretään. Lukupään liikkeet kuvataan symboleilla L ja R; L merkitsee siirtymää vasemmalle ja R oikealle. Jos lukupäätä yritetään siirtää nauhan vasemman laidan yli, lukupää pysyy paikallaan. Kone aloittaa suorituksen tilasta q_0 lukupää nauhan vasemmassa reunassa edeten tilasta toiseen siirtymäfunktion osoittamalla tavalla joko päätyen toiseen lopetustiloista q_{acc} tai q_{rej} tai jatkaen suoritusta ikuisesti. Jos kone M päätyy tilaan q_{acc}, kone hyväksyy syötteen. (Sipser 2006, s. 142–144)

Epädeterminististinen Turingin kone (engl. *non-deterministic Turing machine*) määritellään kuten Turingin kone yllä, mutta siirtymäfunktio on muotoa

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

Deterministisestä koneesta poiketen epädeterministiselle koneelle on siis mahdollista edetä usealla eri tavalla tietyn tila-syöte-parin tapauksessa. Koneen suoritus muodostaa puun, jonka haarat kuvaavat koneen mahdollisia suorituksen tiloja. Huomattava on, että suorituksen tila on eri käsite kuin koneen tila; jälkimmäinen on joku tila $q \in Q$, mutta ensimmäinen kuvaa koko tilaa, joka on koneen nauhan sisältö, lukupään paikka nauhalla ja koneen tila yhdessä. Jos jokin haaroista päätyy hyväksyvään lopetustilaan q_{acc}, kone hyväksyy syötteen. (Sipser 2006, s. 152))

Ongelma on *laskennallisesti työläs* (engl. *intractable*), jos ongelmalle ei ole polynomisessa ajassa toimivaa algoritmia. (Cormen et al. 1990, s. 916)

3 Kaarikromaattisen luvun raja-arvot

Lause 1 (Vizingin lause)**.** *Jokaiselle verkolle pätee*

$$\Delta(G) \leq \chi'(G) \leq \Delta(G) + 1.$$

Verkolla G on siis kaariväritys $\Delta(G)$ tai $\Delta(G) + 1$ värillä.

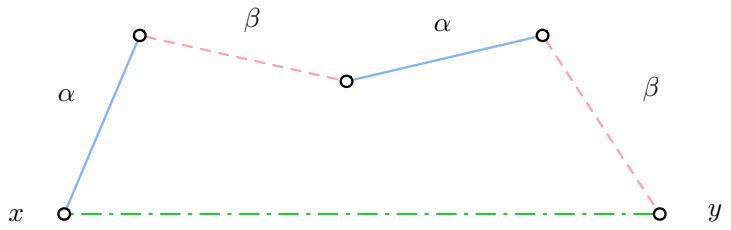
Todistus. Kaarikromaattinen luku $\chi'(G)$ on selvästi suurempi tai yhtä suuri kuin verkon maksimiaste $\Delta(G)$. Otetaan induktio-olettamaksi, että verkolle, jolla on maksimiaste $\Delta(G)$, löytyy kaariväritys $\Delta(G)+1$ värillä. Induktion perustapauksessa, jossa G on tyhjä, olettama pätee triviaalisti. Jatkossa käytetään Diestelin esimerkin (Diestel 2010, s. 126) mukaan kaarivärityksestä $\Delta(G) + 1$ värillä vain termiä *väritys*.

Jokaisesta verkon solmusta v *puuttuu* jokin väri, koska $d(v) < \Delta(G) + 1$. Olkoon tämä väri β; toisin sanoen solmusta v ei lähde kaarta, jolla olisi väri β. Kaikille muille väreille $\alpha \neq \beta$ solmusta v lähteville kaarille on maksimaalinen polku, jossa on vuoroin värejä α ja β.

Oletetaan, että verkolla G ei ole kaariväritystä $\Delta(G)+1$ värillä. Olkoon kaari $xy \in E(G)$ ja verkko $H = G - xy$, toisin sanoen verkosta G on poistettu kaari xy. Tästä seuraa apulause 2.

Apulause 2. *Mille tahansa kaariväritykselle $\Delta(G)+1$ värillä verkolle H, jossa solmusta x puuttuu väri α ja solmusta y puuttuu väri β, α/β-polun solmusta y pitää päättyä solmuun x.*

Muussa tapauksessa polulta voisi vaihtaa värit α ja β keskenään ja kaarelle xy voisi antaa värin α. Näin väritys olisi mahdollinen, mikä aiheuttaisi ristiriidan oletuksen kanssa. Eräs α/β-polku on esitetty kuvassa 2.



Kuva 2: α/β-polku solmusta x solmuun y.

Sanotaan verkkoa, josta on poistettu kaari xy_i, verkoksi G_i, toisin sanoen $G_i = G - xy_i$. Olkoon siis verkko $G_0 = G - xy_0$, jolle on annettu väritys c_0. Väritys on olemassa induktio-olettaman perusteella.

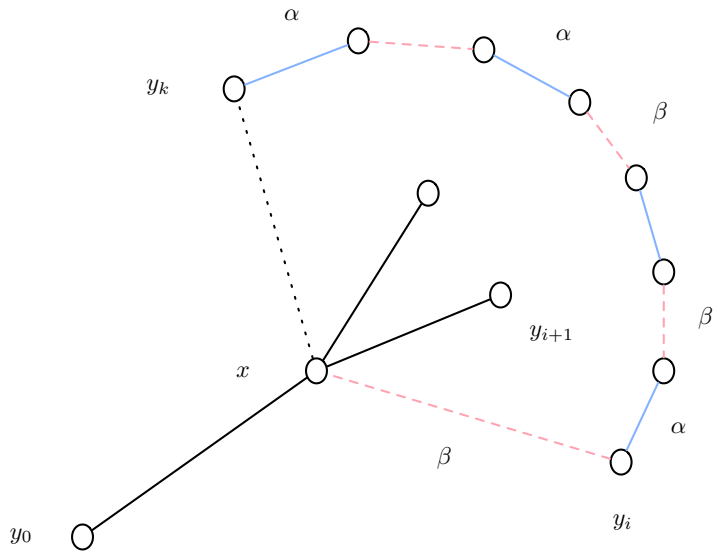
Tarkastellaan maksimaalista joukkoa solmuja $Y = y_0, y_1, \ldots, y_k$, joille pätee, että väri $c(xy_{i+1})$ puuttuu solmusta y_i värityksessä c_0. Joukon maksimaalisuuden perusteella solmulle y_k ei siis löydy määritelmän mukaista solmua y_{k+1}.

Muodostetaan nyt väritykset c_i verkoille G_i. Väritys c_i on muuten sama värityksen c_0 kanssa pois lukien kaari xy_{i+1}, joka saa saman värin kuin kaari xy_i värityksessä c_0. Tämä voidaan tehdä, koska kaari $xy_i \notin G_i$ ja solmusta y_{i+1} puuttuu kyseinen väri joukon määritelmän perusteella.

Merkitään solmusta y_k puuttuvaa väriä symbolilla β värityksessä c_0. Aliverkossa G_k pitää siis apulauseen 2 takia olla α/β-polku solmusta y_k solmuun x. Kaaren yx pitää olla väritetty värillä β ja siten polun viimeistä edellisen solmun y tulee olla joku solmuista $y_0, y_1, \ldots, y_{k-1}$ solmujoukon Y maksimaalisuuden perusteella. Merkitään $y_i = y$. Värityksen c_k määritelmän perusteella seuraa, että $c_k(xy_i) = c_0(xy_{i+1}) = \beta$.

Aliverkossa G_i on sama α/β-polku kuin aliverkossa G_k lukuun ottamatta poistettua kaarta xy_k; nämä on esitetty kuvassa 3. Koska solmusta y_k puuttuu väri β värityksistä c_i ja c_0, polku ei pääty solmuun x. Tämä on ristiriidassa apulauseen 2 kanssa, joten alkuperäinen väite on todistettu. (Diestel 2010, s. 125–127) □

Kaikki verkot voidaan siis jakaa kahteen luokkaan Vizingin lauseen perusteella – luokkaan 1, johon kuuluville verkoille on kaariväritys $\Delta(G)$ värillä ja luokkaan 2, johon kuuluville verkoille on kaariväritys $\Delta(G) + 1$ värillä. Jaon yksinkertaisuudesta huolimatta tämä on vaativa ongelma ja se kuuluu NP-täydellisten ongelmien luokkaan.

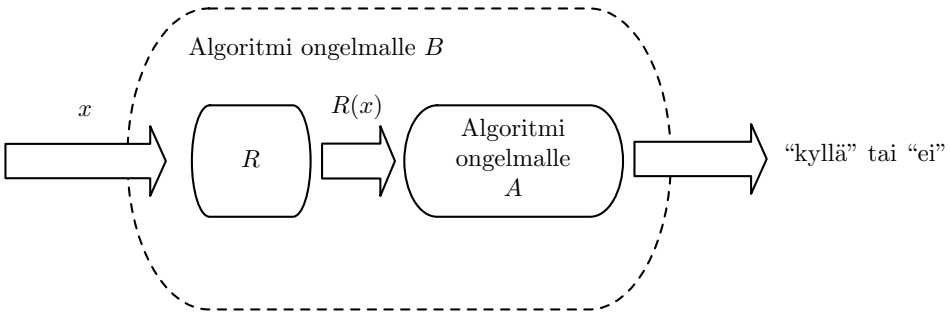


Kuva 3: α/β-polku solmusta y_k solmuun x verkossa G_k.

4 Päätösongelmat ja NP-täydellisyys

Päätösongelman ratkaisu on aina “kyllä” tai “ei”. Esimerkkejä tällaisista ongelmista ovat toteutuvuusongelma (SAT) ja kauppamatkustajan ongelma budjetilla (TSP(D)). Normaali kauppamatkustajan ongelma ei ole päätösongelma, koska vastauksena on polku, jolla on kustannus – päätösongelma tästä saadaan kysymällä onko mahdollista käydä kaikki paikat läpi ylittämättä annettua kattokustannusta.

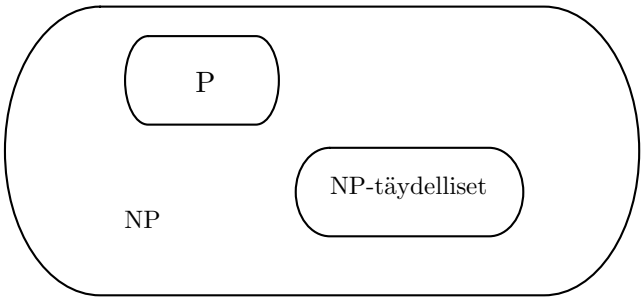
Päätösongelman A *palauttaminen* (engl. *reduction*) toiseksi ongelmaksi B tapahtuu etsimällä polynomisessa ajassa tapahtuva kuvaus R ongelman B syötteestä ongelman A syötteeksi. Kuvassa 4 palautus on esitetty graafisesti. (Papadimitriou 1994, s. 159–160)



Kuva 4: Palautus päätösongelmasta B päätösongelmaan A.

NP-täydellisyyden määrittelemiseksi tarvitsee vielä määritellä luokkien P ja NP ongelmat. Luokkaan P kuuluvat päätösongelmat ovat määritelmän mukaan ne ongelmat, jotka

deterministinen Turingin kone ratkaisee polynomisessa ajassa. Epädeterministinen Turingin kone ratkaisee luokkaan NP kuuluvat ongelmat polynomisessa ajassa: kone arvaa ratkaisun, tarkastaa sen ja siirtyy lopputilaan. Mikäli P $\neq$ NP, deterministinen Turingin kone joutuu todennäköisesti käyttämään samaan eksponentiaalisen määrän askelia annetun syötteen pituuteen verrattuna. Vaativuusluokkien suhteita toisiinsa on esitetty kuvassa 5.



Kuva 5: Päätösongelmien luokittelu.

Määritelmä 3. Päätösongelma on *NP-täydellinen*, jos ongelma kuuluu luokkaan NP ja kaikki muut NP-täydelliset ongelmat voidaan palauttaa tähän ongelmaan.

Vuosikymmenten uutterasta tutkimuksesta huolimatta ei ole saatu lopullista selvyyttä ovatko P ja NP todella eri luokkia. Yleisesti tietojenkäsittelyteorian tutkijat ovat sitä mieltä, että P $\neq$ NP lähinnä siksi, että on olemassa NP-täydellisten ongelmien luokka (Cormen et al. 1990, s. 929). Jos yhteenkin NP-täydelliseen ongelmaan löytyisi polynomisessa ajassa toimiva algoritmi, sitä voitaisiin käyttää kaikkien NP-täydellisten ongelmien ratkaisemiseen. Tällainen tulos olisi luonnollisesti ihmiskunnalle loistava – erilaisiin optimointiongelmiin voitaisiin kehittää tehokkaita ratkaisuja epätäydellisten heuristiikkojen tai erikoistapausten etsimisen sijaan (Fortnow 2009).

5 3SAT-ongelman NP-täydellisyys

SATISFIABILITY, lyhyesti SAT, eli toteutuvuusongelma oli ensimmäinen NP-täydelliseksi todistettu ongelma. Todistuksessa käytetään hyväksi sitä, että luokan NP ongelmalle on määritelmän mukaan olemassa epädeterministinen Turingin kone, joka ratkaisee ongelman polynomisessa ajassa syötteen koon suhteen. Syöte ja koneen suoritus palautetaan joukoksi toteutuvuusongelman klausuuleja, joka on toteutuva jos ja vain jos kyseessä oleva epädeterministinen Turingin kone annetulla syötteellä ainakin yhdellä suorituksella päätyy hyväksyvään lopputilaan. (Garey ja Johnson 1979, s. 38–44)

3SAT-ongelmassa tarkastellaan joukkoa klausuuleja $C = \{c_1, c_2, \ldots, c_k\}$, jossa jokainen klausuuli koostuu kolmesta literaalista ($|c_i| = 3$, kaikilla $i \in \{1, 2, \ldots, k\}$). Klausuuli-

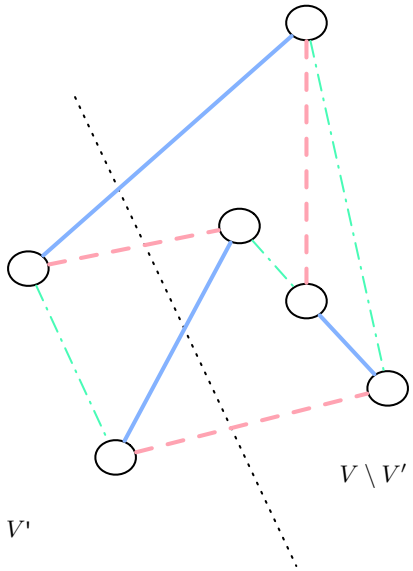
joukko C on toteutuva, jos on olemassa totuusjakelu $\mathcal{A}$, jolla kaikki annetut klausuulit toteutuvat.

3SAT on NP-täydellinen. Todistuksessa SAT-ongelman klausuulit jaetaan uudeksi joukoksi kolmiliteraalisia klausuuleja ottamalla mukaan joukko apuliteraaleja. (Garey ja Johnson 1979, s. 48–50)

6 Kaarivärityksen NP-täydellisyys

Kaarivärityksongelma kuuluu selvästi luokkaan NP, koska kaariväritys voidaan tarkastaa polynomisessa ajassa ja epädeterministinen Turingin kone voi arvata ratkaisun. Holyer (1981) todisti kaarivärityksen NP-täydelliseksi palauttamalla 3SAT-ongelman kaarivärityksongelmaan.

Todistuksessa rakennetaan 3-säännöllinen verkko komponenteista, jotka johdetaan 3SAT-ongelman klausuuleista. Komponenttien muodostamiseen tarvitaan pariteettiehtoa.



Kuva 6: Graafin jakaminen kahteen osaan.

Apulause 4 (Pariteettiehto)**.** *Olkoon verkko G 3-säännöllinen ja kaariväritetty kolmella värillä. Olkoon $V' \subseteq V(G)$ osajoukko verkon solmuja ja $E' \subseteq E(G)'$ ne kaaret, joilla solmut V' on kytketty verkon muihin solmuihin. Merkitään värillä i värjättyjen kaarien lukumäärää joukossa E' symbolilla k_i, missä $i \in \{1, 2, 3\}$. Tällöin*

$$k_1 \equiv k_2 \equiv k_3 \pmod 2$$

Todistus. Olkoon joukko $E_{12} \subseteq E(G)$ ne kaaret, jotka on värjätty värein 1 tai 2. Nämä kaaret muodostavat niihin kytkettyjen solmujen kanssa 2-säännöllisen verkon, joka on

joukko syklejä. E' ja E_{12} jakavat parillisen määrän kaaria, koska jokaista kaarta kohden, joka on joukon $V \setminus V'$ solmusta joukon V' solmuun, pitää löytyä toinen kaari, joka kytkeytyy joukon V' solmusta joukon $V \setminus V'$ solmuun. Yksi tällainen jako on esitelty kuvassa 6.

Tästä seuraa $k_1 + k_2 \equiv 0 \pmod 2$, toisin sanoen

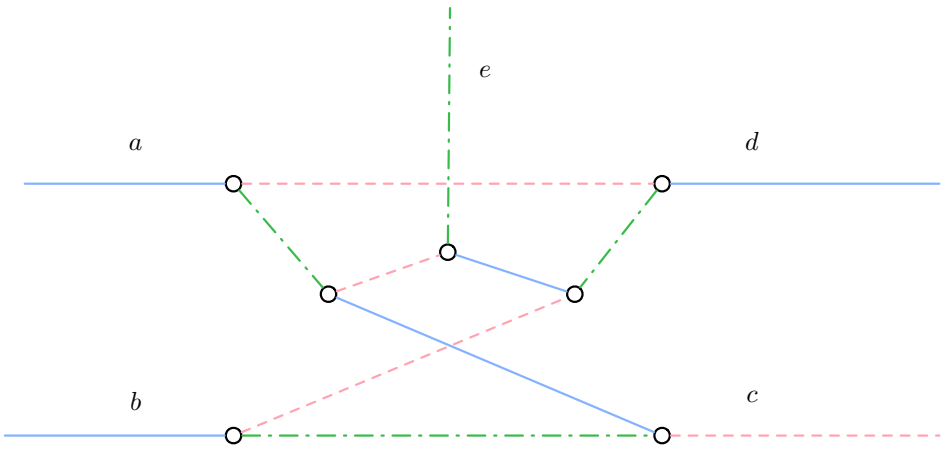
$$k_1 \equiv k_2 \pmod 2.$$

Sama perustelu pätee väreillä 2 ja 3, joten väite on todistettu. □

Palautus 3SAT-ongelmasta etenee rakentamalla 3SAT-ongelmasta verkko G osa kerrallaan. Nämä osat ovat literaalien totuusjakelussa muuttujan arvoa kuvaava asetuskomponentti (engl. *variable setting component*), negaatiota kuvaava kääntökomponentti (engl. *inverting component*) ja klausuulin toteutuvuutta kuvaava tarkastuskomponentti (engl. *satisfaction testing component*).

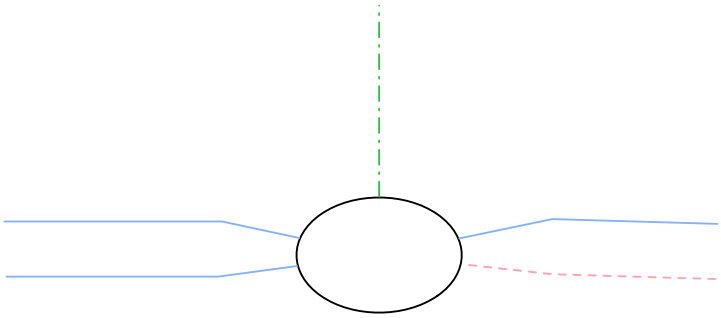
Informaatio kulkee verkossa kaariparien avulla. Jos kaariparin kaaret ovat keskenään eri väriä, kaariparin arvo on F (epätosi). Jos kaaret ovat keskenään samaa väriä, kaariparilla on arvo T (tosi).

Kääntökomponentti on esitetty kuvassa 7. Jos verkolla G on kaarien 3-väritys, kääntökomponentin kahdesta kaariparista toisen kaarien tulee olla väritetty keskenään samalla värillä. Toisin sanoen kaarien a ja b tai kaarien c ja d on oltava keskenään samanvärisiä ja loppujen kolmen on kunkin oltava keskenään eri väriä. Kääntökomponentti saa siis "syötteenä" arvoa kuvaavan kaariparin, jonka se kääntää vastakkaiseksi. Kääntökomponentti on varsin monimutkainen rakenne ja siitä käytetään kuvan 8 mukaista yksinkertaistusta niissä kuvissa, joissa se esiintyy useaan kertaan.



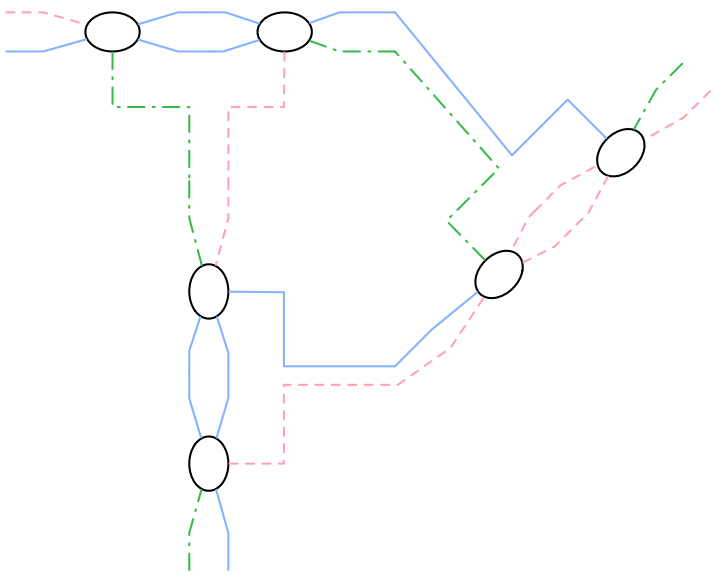
Kuva 7: Kääntökomponentti.

Totuusjakelun muuttujat kuvataan muuttujan asettavalla asetuskomponentilla. Tästä komponentista lähtee yksi kaaripari kutakin kuvattua muuttujaa käyttävää klausuulia



Kuva 8: Kääntökomponentin yksinkertaistus.

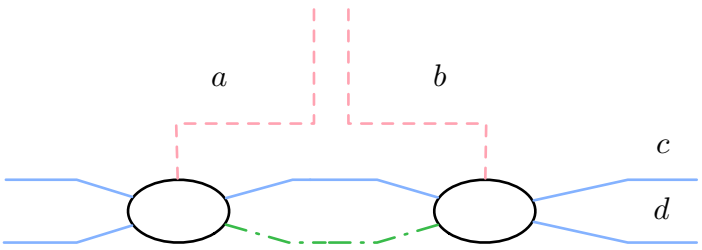
kohden. Jos klausuulijoukossa C on n klausuulia, joissa esiintyy muuttuja u_i, asetuskomponentissa on $2n$ kääntökomponenttia ja siitä lähtee n kaariparia. Eräs asetuskomponentti on esitetty kuvassa 9.



Kuva 9: Muuttujan asettava komponentti.

Jos verkolla G on kaariväritys kolmella värillä, kaikki asetuskomponentista lähtevät kaariparit kuvaavat samaa arvoa F tai T. Asetuskomponentin alkiosta, joka siis muodostuu kahdesta kääntökomponentista, lähtee kolme kaariparia. Jotta asetuskomponentilla olisi kaarien 3-väritys, kaikkien näiden kaariparien on kuvattava samaa arvoa. Tarkastellaan arvon T asettavan komponentin osaa kuvassa 10. Riippumatta keskimmäisen kääntökomponentit yhdistävän kaariparin värien valinnasta, kaarien a ja b värien täytyy olla samat. Pariteettiehdon perusteella kumpikaan kääntökomponenttien välisistä kaarista ei ole väritetty värillä $c(a)$. Jotta pariteettiehto täyttyisi oikeanpuoleisen kääntökomponentin tapauksessa, täytyy jonkun tai kaikkien kaarista b, c ja d olla väritetty värillä $c(a)$. Jos kaikki kaaret tai vain kaari b on väritetty värillä $c(a)$, väitteemme on todistettu. Jos puolestaan väritämme toisen kaarista c tai d värillä $c(a)$, täytyisi myös sen parin olla

väritetty kyseisellä värillä, koska oikeanpuoleisen kääntökomponentin sisääntulo kuvaa arvoa F. Tässä tapauksessa on pakko värittää myös kaari b värillä $c(a)$ pariteettiehdon täyttämiseksi. Kaikki kolme kaariparia kuvaavat siis samaa arvoa T. Todistuksesta seuraa myös, että asetuskomponentin kuvatessa arvoa F kaikkien kaariparien on kuvattava arvoa F, koska muusta seuraisi ristiriita.



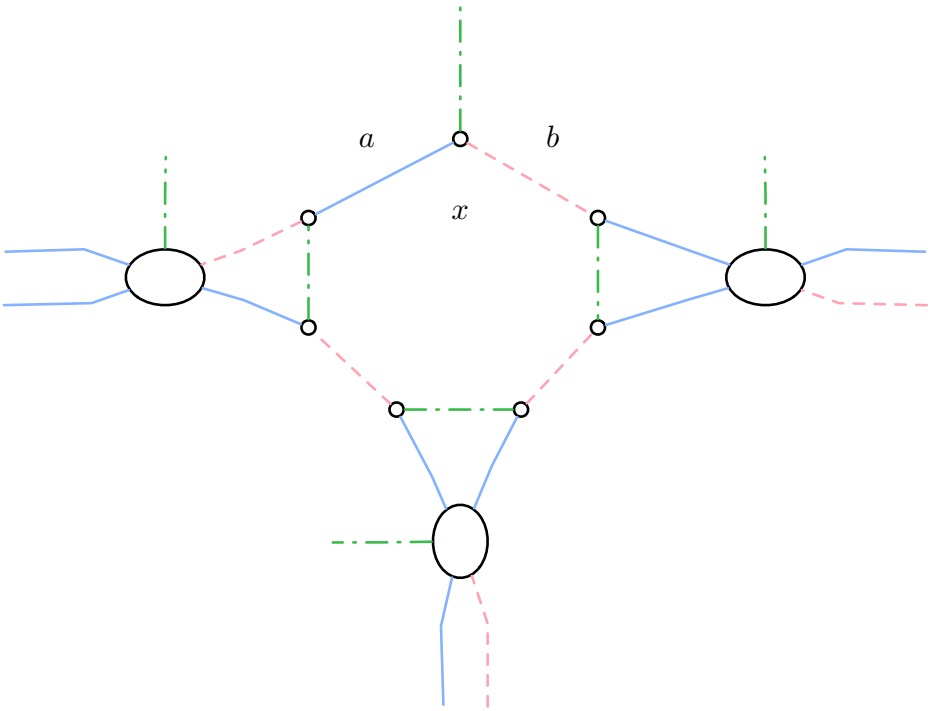
Kuva 10: Osa muuttujan arvolle T asettavaa komponenttia.

Kuvan 11 tarkastuskomponentin syötteenä toimiville kaaripareille ei ole värien suhteen vaatimuksia ja merkityksellistä on vain ovatko kaaret keskenään samaa vai eri väriä. Tarkastuskomponentin kääntökomponentteja hyödyntäen mitkä tahansa värit voidaan yhtenäistää komponentin sisäosan väritystä varten. Tarkastuskomponentin kaariväritys kolmella värillä ei onnistu jos ja vain jos kaikki komponentin sisääntuloparit kuvaavat arvoa F. Kääntökomponenttien jälkeen kaikkien kaarien on ennen sisäsolmuja oltava väritetty samalla värillä, jotta väritys kolmella värillä olisi mahdollinen, mutta nyt solmun x molemmat sisäkaaret a ja b olisi väritetty samalla värillä. Toisin sanoen tässä tapauksessa komponentilla ei ole kaarivärityst kolmella värillä.

Lause 5. *Sen selvittäminen, kuuluuko 3-säännöllinen verkko G luokkaan 1 vai 2, toisin sanoen onko verkon kaarikromaattinen luku $\chi'(G)$ 3 vai 4, on NP-täydellinen ongelma.*

Todistus. Jokaista klausuuleissa esiintyvää muuttujaa u_i tai $\bar{u}_i$ kohden otetaan asetuskomponentti U_i. Jokaista klausuulijoukon C klausuulia c_j kohti otetaan tarkastuskomponentti C_j. Olkoon klausuulin c_j paikan k literaali $l_{j,k}$ muuttuja u_i. Kiinnitetään nyt yksi asetuskomponentin U_i kaaripari tarkastuskomponentin syötekaariparii k. Mikäli literaali $l_{j,k}$ on muotoa $\bar{u}_i$, asetuskomponentin U_i ja tarkastuskomponentin C_j väliin liitetään kääntökomponentti.

Näin on muodostettu verkko H, jossa on kaaria, joilta puuttuu toinen päätepiste. Kopioimalla verkko H ja liittämällä päätepisteettömät vastinkaaret toisiinsa saadaan verkko G. Klausuulijoukosta C muodostettu verkko G on verkon muodostavien komponenttien ominaisuuksista johtuen väritettävissä kolmella värillä jos ja vain jos klausuulijoukko C on toteutuva. Kuvattu palautus tapahtuu polynomisessa ajassa, joten väite on todistettu. (Holyer 1981) □



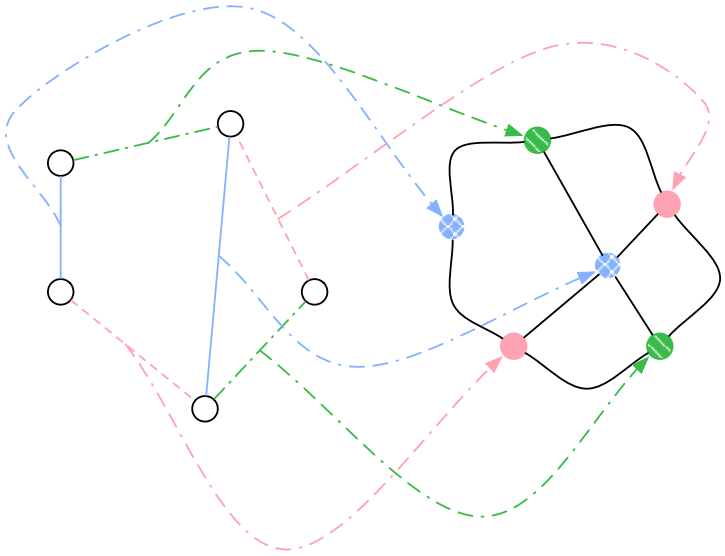
Kuva 11: Klausuulin toteutuvuuden tarkastava komponentti.

7 Nopein tunnettu algoritmi kaariväritykselle

Björklund, Husfeldt, ja Koivisto (2009) esittävät tällä hetkellä nopeimman tunnetun algoritmin solmuväritykselle. Jotta algoritmia voitaisiin käyttää kaariväritykselle, verkosta G täytyy muodostaa ensin viivaverkko $L(G)$; tämä muunnos esitetään luvussa 7.1. Luvussa 7.2 todistetaan, että algoritmi todella ratkaisee ongelman ja luvussa 7.3 perustellaan algoritmin vaatima aika.

7.1 Viivaverkon muodostaminen

Viivaverkko muodostetaan lisäämällä jokaista verkon G kaarta kohden solmu verkkoon $L(G)$. Kahden kaaren ollessa naapureita – eli kun ne jakavat päätepisteen – viivaverkkoon lisätään kaari näitä kaaria kuvaavien solmujen välille. Muunnos etenee seuraavasti: merkitään verkon G kaarta $x_ix_j \in E(G)$ vastaavaa solmua viivaverkossa $L(G)$ symbolilla v_k, missä $k \in \{1, 2, \ldots, |E(G)|\}$. Jokaista verkon G kaarta x_ix_j, jota kuvaa verkossa $L(G)$ solmu v_a, ja sen naapuria x_jx_m, jota kuvaa verkossa $L(G)$ solmu v_b, kohden lisätään kaari v_av_b verkkoon $L(G)$. Esimerkki viivaverkon muodostamisesta on esitetty kuvassa 12. Viivaverkon $L(G)$ muodostaminen verkosta G tapahtuu selvästi polynomisessa ajassa, koska kukin verkon G kaari käsitellään vain kerran ja kullakin kaarella on korkeintaan $2(\Delta(G) - 1)$ naapuria.



Kuva 12: Viivaverkon muodostaminen.

7.2 Algoritmin periaate

Solmuväritys ratkaistaan käyttämällä joukkopeittoja, joiden laskemiseen puolestaan tarvitaan inkluusio-ekskluusioperiaatetta.

Olkoon B äärellinen joukko, jolla on osajoukot $A_1, \ldots, A_n \subseteq B$. Käytetään konventiota $\bigcap_{i \in \emptyset} A_i = B$. Merkinnällä $\overline{A_i}$ tarkoitetaan joukon A_i käänteisjoukkoa joukossa B, toisin sanoen joukkoa $B \setminus A_i$.

Apulause 6 (Inkluusio-ekskluusioperiaate)**.** *Joukon B alkioiden, jotka eivät ole yhdessäkään joukoista A_i, lukumäärä on*

$$\left| \bigcap_{i=1}^{n} \overline{A_i} \right| = \sum_{X \subseteq 1,\ldots,n} (-1)^{|X|} \left| \bigcap_{i \in X} A_i \right|.$$

Todistus. Jos alkio $a \in B$ ei kuulu mihinkään joukoista A_i, se kasvattaa yhtälön vasenta puolta yhdellä. Samoin se kasvattaa oikeaa puolta yhdellä termin $X = \emptyset$ takia. Jos a kuuluu joukkoon A_i kaikilla $i \in I \neq \emptyset$, se ei vaikuta yhtälön vasempaan puoleen. Koska a kuuluu joukkoon $\bigcap_{i \in X} A_i$ kaikille $X \subseteq I$, alkion a vaikutus yhtälön oikealle puolelle on

$$\sum_{X \subseteq I} (-1)^{|X|} = \sum_{i=0}^{|I|} \binom{|I|}{i} (-1)^i.$$

Tämä summa on nolla binomilauseen perusteella.

$$\sum_{i=0}^{|I|} \binom{|I|}{i} (-1)^i = \sum_{i=0}^{|I|} \binom{|I|}{i} (-1)^i 1^{|I|-i} = (-1+1)^{|I|} = 0.$$

Tämä todistaa väitteen. (Björklund et al. 2009, s. 550) □

Tarkastellaan n-alkioista joukkoa N, jonka osajoukot kuuluvat joukkoon joukkoja $\mathcal{F}$. Monikko $(S_1, \ldots, S_k)$ on *k-peitto* joukolle N siten, että $S_1 \cup \ldots \cup S_k = N$.

Määritellään luku $c_k(\mathcal{F})$, joka on k-peittojen lukumäärä.

Apulause 7 (Joukkopeittojen lukumäärä)**.** *Olkoon $a(X) = |\{S \in \mathcal{F} : S \cap X = \emptyset\}|$ niiden joukkojen lukumäärä joukossa $\mathcal{F}$, joiden yksikään alkio ei ole joukossa X. Tällöin*

$$c_k(\mathcal{F}) = \sum_{X \subseteq N} (-1)^{|X|} a(X)^k.$$

Todistus. Käytetään apulausetta 6, jossa B on joukko k-monikoita $(S_1, \ldots, S_k)$ joukon $\mathcal{F}$ alkioista ja olkoot $A_i \subseteq B$ ne k-monikot, joiden joukkoihin ei kuulu alkio i, toisin sanoen $i \notin S_1 \cup \cdots \cup S_k$. Nyt $c_k(\mathcal{F})$ on niiden k-monikoiden lukumäärä, jotka kuuluvat joukkoon B ja jotka eivät ole missään joukoista A_i; toisin sanoen $\left|\bigcap_{i \in X} \overline{A_i}\right|$. Kun lisäksi otetaan huomioon, että monikon alkioiden permutaatiot ovat merkitseviä, ja monikon pituus on k, tästä seuraa $\left|\bigcap_{i \in X} A_i\right| = a(X)^k$. (Björklund et al. 2009, s. 551) □

Määritellään joukko $\mathcal{R}$, johon kuuluvat kaikki verkon G riippumattomat joukot, joilla on vähintään yksi alkio. Verkon $L(G)$ solmujen k-väritys voidaan esittää myös jakamalla $V(L(G))$ värien mukaan eri joukkoihin. Jokainen näistä joukoista muodostaa riippumattoman joukon solmuja, jotka kuuluvat joukkoon $\mathcal{R}$. Toisin sanoen, jos verkon $L(G)$ solmujoukolle löytyy joukkopeitto siten, että riippumattomia joukkoja on k, samalla on löydetty myös verkon $L(G)$ solmuvärityks. Käytetään seuraavassa apulausetta 7 asettaen $\mathcal{F} = \mathcal{R}$ ja $N = V(G)$.

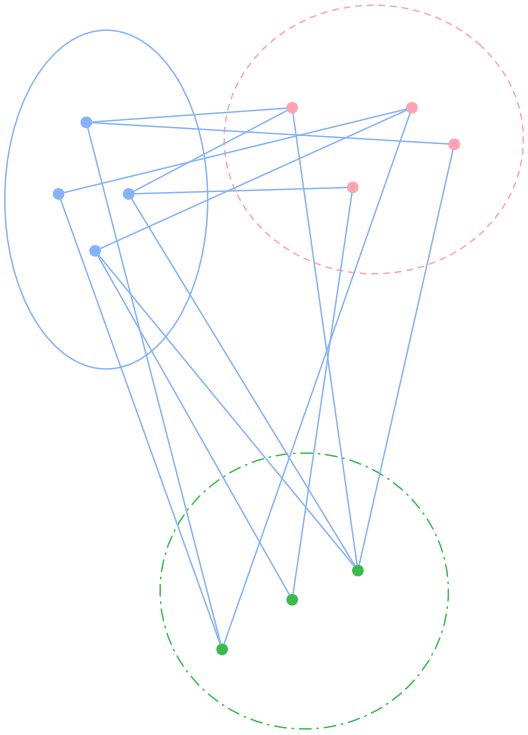
Apulause 8 (Solmuvärityksen olemassaolo)**.** *Verkon $L(G)$ solmujen k-väritys on mahdollinen jos ja vain jos $c_k(\mathcal{R}) > 0$.*

Todistus. Verkon k-solmuväritys tarkoittaa solmujoukon $V(L(G))$ jakamista riippumattomiin joukkoihin, joita on k kappaletta. Jos tällainen jako on olemassa, on $c_k(\mathcal{R}) > 0$. Toisaalta, jos on olemassa joukkopeitto $S_1 \cup \ldots \cup S_k = V(L(G))$, jossa alkio a saattaa olla useammassa kuin yhdessä joukossa $S_1, \ldots, S_k$, väritys tästä saadaan käyttämällä kullekin alkiolle väriä $c(a) = \min\{t : a \in S_t\}$, toisin sanoen otetaan alkion a väri indeksiltään ensimmäisen joukon mukaan. (Björklund et al. 2009, s. 556). Eräs väritys ja samalla jako riippumattomiin joukkoihin on esitetty kuvassa 13. □

7.3 Algoritmin laskennallinen vaativuus

Olkoon n verkon $L(G)$ solmujen lukumäärä.

Lause 9 (Solmuväritysongelman laskennallinen vaativuus)**.** *Verkon solmujen k-värityksen olemassaolo verkolle $L(G)$ voidaan ratkaista ajassa $O(2^n nk \operatorname{polylog} nk)$.*



Kuva 13: Solmujen jako riippumattomiin joukkoihin, mikä on samalla solmuväritys.

Todistus. Olkoon joukko $N(v)$ solmu v ja sen naapurit, toisin sanoen $N(v) = \{v\} \cup \{u \in V(L(G)) : uv \in E(L(G))\}$. Osoitetaan, että $a(X)$ toteuttaa palautuskaavan

$$a(X) = a(X \cup \{v\}) + a(X \cup N(v)) + 1 \qquad (v \notin X). \tag{1}$$

Tutkitaan riippumattomia joukkoja $S \neq \emptyset$, jotka eivät leikkaa joukon X kanssa. Nämä joukot kuuluvat kahteen luokkaan: joko ne sisältävät alkion v, jolloin kyseisiä joukkoja merkitään symbolilla S_v, tai joukot eivät sisällä alkiota v, jolloin niitä merkitään symbolilla $S_{\bar{v}}$. Joukot $S_{\bar{v}}$, joihin ei kuulu alkio v, lasketaan termillä $a(X \cup \{v\})$. Lasketaan nyt joukot $S_v \setminus \{v\}$, joita on sama määrä kuin jäljellä olevia joukkoja S_v. Koska S_v on riippumaton joukko ja se pitää sisällään alkion v, siinä ei voi olla muita alkioita joukosta $N(v)$. Tästä seuraa, että joukko $S_v \setminus \{v\}$ ei leikkaa joukkoja $N(v)$ tai X. Joukko S_v on siis joko joukko $\{v\}$ vastaten termiä $+1$ tai $S_v \setminus \{v\}$ on epätyhjä riippumaton joukko, joka lasketaan termillä $a(X \cup N(v))$. (Björklund et al. 2009, s. 556)

Palautuskaava (1) muodostaa algoritmin, jolla $a(X)$ voidaan laskea halutulle solmujoukolle. Rekursion perustapaus on $a(V) = 0$. Operaatiot suoritetaan $O(n)$-bittisillä kokonaisluvuilla. Arvot talletetaan taulukkoon, jonka koko on luokkaa $O(n2^n)$, joka on myös algoritmin aikavaatimus. Koska taulukon elementit on korotettava eksponenttiin k, tästä seuraa aikavaatimus $\log k$ kertolaskuille ja summille nk-bittisillä kokonaisluvuilla. (Björklund et al. 2009, s. 556) □

8 Kaariväritys helpommille erikoistapauksille

Esimerkkinä huomattavasti helpommasta erikoistapauksesta käytetään kaksijakoisia verkkoja (engl. *bipartite graph*). Kaikille kaksijakoisille verkoille on kaariväritys $\Delta(G)$ värillä (Diestel 2010, s. 125). Esimerkiksi tiedonsiirto-ongelma, sopivasti rajattuna, ratkeaa kaksijakoisten verkkojen tavoin (Cole, Ost, ja Schirra 2001). Tiedonsiirto-ongelmassa siirretään suuri joukko tiedostoja tietokoneverkon eri koneiden välillä: ongelmaa kuvataan tietokonenoodien välisistä verkkoyhteyksistä muodostetulla verkolla, jossa solmut ovat tietoverkon noodien kommunikaatioportteja ja kaaret kuvaavat näiden porttien välisiä tiedonsiirtoja. Yleinen tapaus on NP-täydellinen, koska se palautuu kaarivärityksongelmaan (Nakano, Zhou, ja Nishizeki 1995). Myös monissa tilanteissa, joissa ongelma muuten ratkeaisi polynomisessa ajassa, mutta joissa tiedostot pitää edelleenlähettää noodien välillä, ongelma on NP-täydellinen. Rajattu ongelma ratkeaa polynomisessa ajassa. (Nakano ja Nishizeki 1991).

Cole et al. (2001) esittelivät algoritmin kaksijakoisen verkon kaarivärityksen $\Delta(G)$ värillä ajassa $O(m \log \Delta(G))$, jossa m on verkon G kaarien lukumäärä. Tulos koskee monimutkaisempia verkkoja, niin kutsuttuja multigraafeja, joilla kahden solmun välillä voi olla useampia kaaria, mutta se pätee siis myös yksinkertaisille verkoille. (Cole et al. 2001)

Cole ja Kowalik (2008) kuvasivat algoritmin, jolla tasoverkot voidaan värittää lineaarisessa ajassa $\max\{\Delta(G), 9\}$ värillä. Jos verkon maksimiaste on suurempi tai yhtäsuuri kuin 9, algoritmi antaa optimaalisen värityksen.

Kaarikromaattisen luvun $\chi'(G)$ selvittäminen on rajatumpi ongelma kuin kaarivärityksen selvittäminen, joka näin ollen on siis myös NP-täydellinen ongelma. Joskus kaariväritykseen riittää epätäydellinenkin ratkaisu, jolloin kaariväritykseen käytetään optimaalista enemmän värejä, mutta värityksen muodostaminen on tehokasta. Esimerkiksi tasoverkoille, joiden maksimiaste $\Delta(G)$ on 4, 5, 6 tai 7, on kaariväritykset $\Delta(G) + 2$ värillä lineaarisessa ajassa. (Cole ja Kowalik 2008)

9 Jälkisanat

Kaarivärityksongelma, kuten sisarensa solmuvärityksongelma, on tärkeä algoritmiikassa. Koska kyseessä on NP-täydellinen ongelma, on mahdollista, että tehokasta ratkaisua perusongelmaan ei ikinä löydetä (Fortnow 2009). On myös mahdollista, että ongelmaa ei koskaan saada todistettua laskennallisesti työlääksi (Fortnow 2009). Tällaisia ongelmia joudutaan ratkaisemaan suurillakin verkoilla, joten ongelman luontainen vaikeus asettaa ohjelmoijalle erityisiä haasteita löytää sopiva algoritmi kulloinkin käsiteltävälle sovellukselle. Jos sovellus ei kuulu tunnettuihin erikoistapauksiin, joihin löytyy täsmällinen

nopea algoritmi, voidaan joutua käyttämään erilaisia heuristiikkoja. Niissä on omat ongelmansa: heuristiikat voivat yleisessä tapauksessa toimia mainiosti, mutta pahimmassa tapauksessa algoritmin suoritus ei pääty koskaan tai saadaan muulla tavalla epäoptimaalinen tulos. Vaikka perusongelmalle ei ratkaisua löytyisikään, on jatkossakin mahdollista parantaa sovellusten suorituskykyä erilaisille erikoistapauksille räätälöityjen algoritmien muodossa.

Lähteet

Björklund, A. & Husfeldt, T. & Koivisto, M. Set partitioning via inclusion-exclusion. *SIAM Journal on Computing*, 39(2):546–563, 2009. DOI: 10.1137/070683933.

Cole, R. & Kowalik, L. New linear-time algorithms for edge-coloring planar graphs. *Algorithmica*, 50:351–368, 2008. ISSN 0178-4617. DOI: 10.1007/s00453-007-9044-3.

Cole, R. & Ost, K. & Schirra, S. Edge-coloring bipartite multigraphs in $O(E \log D)$ time. *Combinatorica*, 21:5–12, 2001. ISSN 0209-9683. DOI: 10.1007/s004930170002.

Cook, S. A. The complexity of theorem-proving procedures. *Proceedings of the Third Annual ACM Symposium on Theory of Computing*, STOC '71, sivut 151–158, New York, NY, USA, 1971. ACM. DOI: 10.1145/800157.805047.

Cormen, T. H. & Leiserson, C. E. & Rivest, R. L. *Introduction to Algorithms*. MIT Press, Cambridge, Massachusetts, USA, 1990. ISBN 978-0-262-53091-0.

Diestel, R. *Graph Theory*, osa 173 sarjasta *Graduate Texts in Mathematics*. Springer, Heidelberg, neljäs painos, 2010. ISBN 978-3-642-14278-9.

Fortnow, L. The status of the P versus NP problem. *Communications of the ACM*, 52 (9):78–86, 2009. DOI: 10.1145/1562164.1562186.

Garey, M. R. & Johnson, D. S. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. A Series of Books in the Mathematical Sciences. W. H. Freeman and Company, New York, 1979. ISBN 0-7167-1045-5.

Holyer, I. The NP-completeness of edge-colouring. *SIAM Journal on Computing*, 10(4): 718–720, 1981.

Nakano, S.-i. & Nishizeki, T. Scheduling file transfers under port and channel constraints. Teoksessa *ISA'91 Algorithms*, Hsu, W.-L. & Lee, R., toimittajat, osa 557 sarjasta *Lecture Notes in Computer Science*, sivut 43–51. Springer Berlin / Heidelberg, 1991. ISBN 978-3-540-54945-1. DOI: 10.1007/3-540-54945-5_48.

Nakano, S.-i. & Zhou, X. & Nishizeki, T. Edge-coloring algorithms. Teoksessa *Computer Science Today*, van Leeuwen, J., toimittaja, osa 1000 sarjasta *Lecture Notes in Computer Science*, sivut 172–183. Springer Berlin / Heidelberg, 1995. ISBN 978-3-540-60105-0. DOI: 10.1007/BFb0015243.

Papadimitriou, C. H. *Computational Complexity*. Addison Wesley Longman Publishing Co., Inc., Redwood City, California, USA, 1994. ISBN 0-201-53082-1.

Sipser, M. *Introduction to the Theory of Computation*. Course Technology, Boston, Massachusetts, USA, 2006. ISBN 978-0-619-21764-8.