Teknillinen korkeakoulu
Sähkö- ja tietoliikennetekniikan osasto
Tietoliikennetekniikan tutkinto-ohjelma

Liitevapaat koodit

Kandidaatintyö

23.5.2006

Mikko Malinen

Teknillinen korkeakoulu
Sähkö- ja tietoliikennetekniikan osasto
Tietoliikennetekniikan tutkinto-ohjelma

Kandidaatintyön tiivistelmä

Tekijä:	Mikko Malinen
Työn nimi:	Liitevapaat koodit
Päiväys:	23.5.2006
Sivumäärä:	27
Vastuuopettaja:	Prof. Raimo Kantola
Ohjaaja:	Prof. Patric Östergård, TKK Tietoliikennelaboratorio

Tämä työ on kirjallisuustutkimus liitevapaista koodeista. Liitevapaat koodit ovat englanniksi fix-free codes, bifix codes, biprefix codes tai reversible variable length codes RVLCs. Kirjoja liitevapaista koodeista ei ole julkaistu juuri ollenkaan, joten työ keskittyy julkaistuihin tieteellisiin artikkeleihin. Asia on varsin uutta, tutkitut artikkelit on julkaistu lähes poikkeuksetta vuosina 1995-2005. Työssä käsitellään tärkeätä $\frac{3}{4}$-konjektuuria, joka liittyy liitevapaiden koodien olemassaoloon. Useita konjektuurin erikoistapauksia on todistettu. Työssä käsitellään tarkemmin yhtä näistä erikoistapauksista, käydään läpi todistus riittävälle ehdolle liitevapaiden koodien olemassaololle. Tästä tuloksesta voidaan johtaa yläraja optimoitujen liitevapaiden koodien redundanssille. Alaraja redundanssille saadaan aikaisempien tulosten perusteella. Työssä esitetään, että täydellisisiä vaihtelevanpituisia liitevapaita koodeja on olemassa. Käytännön liitevapaiden koodien muodostaminen lähtee liikkeelle lähteen todennäköisyysjakaumasta. Työssä käydään läpi eräs tällainen tehokkaiden liitevapaiden koodien muodostusalgoritmi.

Avainsanat:	Liitevapaat koodit, redundanssi, Kraftin summa, entropia
Kieli:	Suomi

Helsinki University of Technology Abstract of Bachelor's Thesis
Department of Electrical and Communications Engineering
Communications Engineering degree program

Author:	Mikko Malinen
Name of Thesis:	Fix-free codes
Date:	23rd May, 2006
Pages:	27
Responsible teacher:	Prof. Raimo Kantola
Instructor:	Prof. Patric Östergård, TKK Communications Laboratory

This thesis is a literary survey about fix-free codes. Fix-free codes are also called bifix codes, biprefix codes or reversible variable length codes RVLCs. There are hardly any published books about fix-free codes, so this work concentrates on published scientific articles. This matter is quite new, the studied articles are published almost without exceptions in years 1995-2005. The work deals with the important $\frac{3}{4}$-conjecture, which is in connection with the existence of fix-free codes. Many special cases of this conjecture have been proved. The work deals more deeply with one of these special cases. It goes through a proof of a sufficient condition for the existence of fix-free codes. From this result can be derived an upper bound for the redundancy of optimized fix-free codes. The lower bound for the redundancy can be derived from earlier results. In this work it is presented that complete variable-length fix-free codes exist. Practical construction of fix-free codes starts with given probability distribution of the source. This work goes through one such construction algorithm of efficient fix-free codes.

Keywords:	Fix-free codes, redundancy, Kraft sum, entropy
Language:	Finnish

Kiitossanat

Haluan kiittää työni ohjaajaa professori Patric Östergårdia tämän aiheen ehdottamisesta minulle sekä runsaista kommenteista ja parannusehdotuksista työtä tehdessä.

Haluan kiittää myös vanhempiani tuesta.

Espoossa 29. toukokuuta 2006

Mikko Malinen

Sisältö

Luku 1

Johdanto

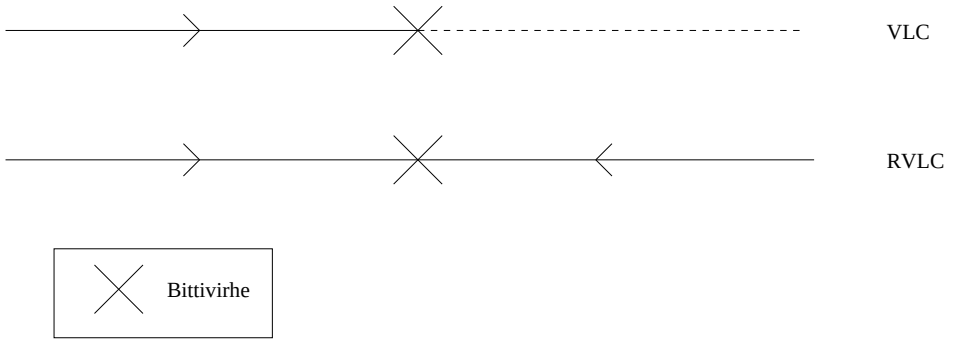
1.1 Koodit ja dekoodaus

Prefiksikoodi voidaan dekoodata ilman tulevien koodisanojen lukemista, koska koodisanan loppu on välittömästi tunnistettavissa. Täten, prefiksikoodin ollessa kyseessä, symboli x_i voidaan dekoodata heti kun tullaan sitä vastaavan koodisanan loppuun. Ei siis tarvitse nähdä koodisanoja, jotka tulevat myöhemmin. Prefiksikoodi on "itsepilkuttuva"koodi; voidaan lukea eteenpäin peräkkäisiä koodisymboleita ja lisätä pilkut koodisanojen erottelemiseksi ilman että katsotaan myöhempiä symboleita. Esimerkiksi binäärinen merkkijono 01011111010 voidaan jäsentää 0,10,111,110,10 jos nämä ovat koodisanoja. [2]

Liitevapailla koodeilla on joukko mielenkiintoisia ominaisuuksia. Esimerkiksi, sana joka on muodostettu liittämällä yhteen joukko koodisanoja liitevapaasta koodista, voidaan yksikäsitteisesti jäsentää kummasta tahansa päästä. Liitevapailla koodeilla on hyvät synkronisointiominaisuudet: jos koodi ei ole suffiksikoodi, se tarkoittaa, että jos kanava kadottaa bitin, niin vastaanotin ei voi helposti uudelleensynkronisoida. Vastaanotin jäsentää ensin lähetteen koodisanoiksi ja sitten dekoodaa ne. Kun koodi ei ole suffiksikoodi, vastaanotin jäsentää kadotetusta bitistä eteenpäin väärin ja jäsennetyn sanan loppu ei koskaan täsmää lähetetyn sanan lopun kanssa. Tätä tarkoitetaan uudelleensynkronisoinnin epäonnistumisella.

Lähetyksen idea on seuraava. Oletetaan, että käytetään binääristä liiteva-

paata koodia tiedonsiirtoon ja että sanoma on ryhmitelty N bitin lohkoihin. Koodisanan x dekoodauksessa havaitun virheen tapauksessa (koodit ovat epätäydellisiä ja siis on todennäköistä, että virhe havaitaan) lohkon dekoodauksessa siirrytään oikealta vasemmalle moodiin. Siten jos enintään yksi virhe tapahtuu lohkossa, korkeintaan yksi koodisana luetaan virheellisesti (ks. kuva 1.). Näitä koodeja käytetään kuvien siirtoon videostandardeissa H.263+ ja MPEG4 [9]. Videostandardeja H.263+ ja MPEG4 käsitellään viitteessä [11].



Kuva 1. Bittivirheestä toipuminen tavallisissa (VLC) ja ja liitevapaissa (RVLC) vaihtelevanpituisissa koodeissa.

1.2 Entropia

Entropia on satunnaismuuttujan epävarmuuden mitta. Olkoon X diskreetti satunnaismuuttuja aakkostolla D ja todennäköisyysmassafunktiolla $p(x) = \Pr\{X = x\}, x \in D$.

Määritelmä: Diskreetin satunnaismuuttujan X *entropia* $H(X)$ määritellään

$$H(X) = -\sum_{x \in D} p(x) \log p(x).$$

Entropiaa voidaan merkitä myös $H(p)$:llä. Logaritmi on 2-kantainen ja entropia ilmaistaan bitteinä. Entropia on satunnaismuuttujan kuvailussa tarvittava keskimääräinen määrä bittejä.

1.3 Prefiksikoodit

Määritelmä: Koodia sanotaan *prefiksikoodiksi* jos mikään koodisana ei ole toisen koodisanan prefiksi.

Lause 1 *(Kraftin epäyhtälö): Kaikille prefiksikoodeille yli D:n kokoisen aakkoston, koodisanan pituuksien $l_1, l_2, ..., l_m$ pitää toteuttaa epäyhtälö*

$$\sum_i D^{-l_i} \leq 1.$$

Kääntäen, kun annetaan koodisanan pituudet jotka toteuttavat tämän epäyhtälön, on olemassa prefiksikoodi näillä sananpituuksilla.

Optimaalisia koodeja koskee seuraava lause:

Lause 2 *Olkoon L^* optimaalisen koodin koodisanan keskimääräinen pituus. Tällöin*

$$H(X) \leq L^* < H(X) + 1.$$

1.3.1 Huffman-koodit

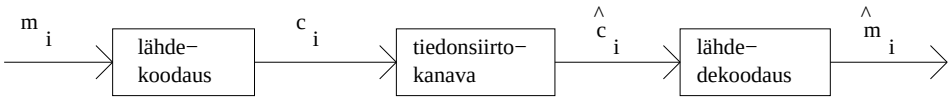
Optimaalinen prefiksikoodi annetulle jakaumalle voidaan muodostaa yksinkertaisella algoritmilla, jonka on kehittänyt Huffman. Millään muulla koodilla, kun käytetään samaa aakkostoa, ei voi ollla pienempää keskimääräistä pituutta kuin tällä algoritmilla muodostetulla.

1.4 Liitevapaat koodit

Liitevapaita koodeja (englanniksi fix-free codes, bifix codes, biprefix codes tai reversible variable length codes RVLCs) ovat koodit, jotka ovat sekä prefiksi- että suffiksikoodeja. Mikään koodisana ei ole toisen koodisanan alkuosa eikä toisen koodisanan loppuosa. Olkoon D aakkosto ja olkoot $w_i, i = [1..m]$ kaikki koodisanat, joiden pituudet ovat $l_i < l_{m+1}$. Olkoot $w_j, j = [(m+1)..n]$ koodisanat, joiden pituus on l_{m+1}. Tällöin koodisanan w_n sallittuja arvoja ovat

$$D^{l_{m+1}} - \sum_{i=1}^{m} w_i D^{l_{m+1}-l_i} - \sum_{i=1}^{m} D^{l_{m+1}-l_i} w_i - \sum_{j=m+1}^{n-1} w_j.$$

Tiedonsiirtojärjestelmän malli on kuvassa 2. Kuvasta on tahallaan jätetty pois kanavakoodaus, sillä sitä ei käsitellä tässä työssä. Liitevapaa koodi on lähdekoodi. Koodisanan c_i keskimääräinen pituus on pienempi kuin sanoman sanan m_i pituus (kompressointi).



Kuva 2. Tiedonsiirtojärjestelmän malli (ilman kanavakoodausta)

Luku 2

Teoria

2.1 $\frac{3}{4}$-konjektuuri ja sen todistettuja erikoistapauksia

2.1.1 $\frac{3}{4}$-konjektuuri

Liitevapaita koodeja muodostettaessa halutaan keskimääräisen koodisanan pituuden olevan mahdollisimman lyhyt. Tämän edellytyksenä on, että koodin Kraftin summa (ks. edempänä) on mahdollisimman suuri. Tämän vuoksi on tärkeätä tietää, kuinka suurella Kraftin summalla koodi on vielä olemassa.

Määritellään γ suurimmaksi vakioksi niin, että kaikille kokonaislukumonikoille $(l_1, l_2, ..., l_N)$ $\sum_{i=1}^{N} 2^{-l_i} < \gamma$ implikoi binäärisen liitevapaan koodin olemassaolon pituuksilla $l_1, l_2, ..., l_N$.

$\sum_{i=1}^{N} 2^{-l_i}$ tunnetaan $(l_1, l_2, ..., l_N)$:n Kraftin summana.

Tällainen γ on olemassa, koska Lemman 1 mukaan se ei voi olla yli $\frac{3}{4}$ ja lauseen 3 mukaan se on vähintään $\frac{5}{8}$. Lisäksi γ on rationaaliluku, sillä Kraftin summan termit ovat rationaalisia.

Lemma 1 $\gamma \leq \frac{3}{4}$.

Todistus: Mille tahansa $\gamma = \frac{3}{4} + \epsilon$, $\epsilon > 0$, valitaan k niin, että $2^{-k} < \epsilon$. Vektorille $(l_1, ..., l_N) = (1, k, ..., k)$ jossa $N = 2^{k-2} + 2$ saadaan

$$\sum_{i=1}^{N} 2^{-l_i} = \frac{1}{2} + 2^{-k}(2^{k-2} + 1) = \frac{3}{4} + 2^{-k} < \frac{3}{4} + \epsilon.$$

Kuitenkin, on olemassa tasan 2^{k-2} k-pituista sanaa, joissa koodisana c_1 (1-pituinen koodisana) ei ole prefiksinä eikä suffiksina ja, koska $1 + 2^{k-2} < N$, olemme osoittaneet, ettei halutuilla parametreilla olevaa koodia ole olemassa. □

Ahlswede *et al.* [1] esittivät seuraavan konjektuurin:

Konjektuuri 1 $\gamma = \frac{3}{4}$.

$\sum_{i=1}^{N} 2^{-l_i} < \gamma$ on riittävä ehto liitevapaan koodin olemassaololle. Kuitenkin, jos Kraftin summa on suurempi kuin γ, voi koodi silti olla liitevapaa.

2.1.2 Konjektuurin todistettuja erikoistapauksia

Olkoon $\mathbf{v}_n = (k_1, ..., k_n)$ vektori, jossa k_i:t ovat ei-negatiivisia kokonaislukuja. Merkitään $C(\mathbf{v}_n)$:llä binääristä vaihtelevanpituista koodia, joka sisältää k_i kpl i-pituista koodisanaa kullekin $i = 1, 2, ..., n$. Kraftin summa voidaan kirjoittaa myös vektorille $\mathbf{v}_n$ muodossa

$$S(\mathbf{v}_n) = \sum_{i=1}^{n} \frac{k_i}{2^i}. \tag{2.1}$$

Konjektuuri 1 on yhä avoin. Tässä listataan konjektuurista useita erikoistapauksia, joidenka tiedetään olevan tosia. Olkoon $p = \min\{i : k_i > 0\}$. Listataan ehdot, jotka implikoivat liitevapaan koodin $C(\mathbf{v}_n)$ olemassaolon.

1. $S(\mathbf{v}_n) \leq \frac{5}{8}$ [19] (ks. kappale 2.2)

2. $S(\mathbf{v}_n) \leq \frac{3}{4}$ ja $\forall i < j$ $k_i \neq 0$, $k_j \neq 0$ implikoi $2i \leq j$ [1]

3. $S(\mathbf{v}_n) \leq \frac{3}{4}$ ja $|\{i : k_i \geq 0\}| \leq 2$ [5]

4. $S(\mathbf{v}_n) \leq \frac{3}{4}$ ja $\forall i < n : k_i \leq 2^{p-2}$ [6]

5. $S(\mathbf{v}_n) \leq \frac{3}{4}$ ja $\forall i < n : k_i \leq 2$ [6]

6. $S(\mathbf{v}_n) \leq \frac{3}{4}$ ja $\frac{k_p}{2^p} + \frac{k_{p+1}}{2^{p+1}} \geq \frac{1}{2}$ [17]

7. $S(\mathbf{v}_n) \leq \frac{3}{4}$ ja $n \leq 8$ [17]

Kukorelly ja Zeger listaavat artikkelissa [7] lisäksi kaksi riittävää ehtoa binääristen liitevapaiden koodien olemassaololle: Olkoon L äärellinen monijoukko positiivisia kokonaislukuja, joiden Kraftin summa on korkeintaan 3/4. On osoitettu, että on olemassa liitevapaa koodi, jonka koodisanan pituudet ovat L:n alkiot jos kumpi tahansa seuraavista kahdesta ehdosta pätee. i) Pienin kokonaisluku L:ssä on vähintään 2, ja mikään kokonaisluku L:ssä, paitsi korkeintaan suurin niistä, ei ilmene yli $2^{\min(L)-2}$ kertaa. ii) Mikään kokonaisluku L:ssä, paitsi korkeintaan suurin niistä, ei ilmene yli kahta kertaa.

2.2 Yläraja ja alaraja liitevapaiden koodien redundanssille

2.2.1 Johdanto ylärajaan

Olkoon $p = \{p_1, ..., p_m\}$ lähteen todennäköisyysjakauma, ja olkoon C koodi lähteelle. Koodin C redundanssi R määritellään tämän koodin keskimääräisen koodisanan pituuden $L(C)$ ja lähteen entropian $H(p)$ välisenä erona. Merkitään optimaalisen liitevapaan koodin redundanssia R_f:llä.

Ahlswede *et al.* [1] ovat todistaneet, että $0 \leq R_f \leq 2$. He ovat myös osoittaneet, että alarajaa 0 ei voida parantaa (ks. alakappale 2.2.4). Myöhemmin Ye ja Yeung [15], [16] johtivat useita ylärajoja R_f:lle käyttäen osittaista tietoa lähteen jakaumasta. [19]:ssä parannetaan R_f:n ylärajaa 2:sta $4 - \log_2 5$:een, mikä on noin 1.678.

Olkoon koodi $C(\mathbf{v}_n)$ ja Kraftin summa $S(\mathbf{v}_n)$ kuten alakappaleessa 2.1.2.

Ahlswede *et al.* [1] todistivat, että konjektuuri 1 on tosi heikommassa ta-

pauksessa, kun Kraftin summa on enintään $\frac{1}{2}$. Jos konjektuuri on tosi, niin R_f:n yläraja voidaan parantaa $3 - \log_2 3$:een, mikä on noin 1.415 [16].

[19]:ssä todistetaan eräs erikoistapaus konjektuurista. Osoitetaan, että $S(\mathbf{v}_n) \leq \frac{5}{8}$ implikoi liitevapaan koodin $C(\mathbf{v}_n)$ olemassaolon (Lause 3). Tämä tulos tuottaa parantuneen ylärajan optimaalisten liitevapaiden koodien redundanssille (Lause 4).

2.2.2 Uusi riittävä ehto liitevapaiden koodien olemassaololle

Olkoon $\mathbf{w}$ mielivaltainen n-pituinen binäärinen vektori. Binääristä vektoria, joka on muodostettu ensimmäisistä {viimeisistä} $\mathbf{w}$:n p symbolista, sanotaan $\mathbf{w}$:n *p-prefiksiksi* {*p-suffiksiksi*} ja sitä merkitään ${}^p\mathbf{w}$ $\{\mathbf{w}^p\}$. Sanotaan, että vektorilla on muoto $\alpha \star \beta$, missä $\alpha, \beta \in \{0,1\}$, jos ${}^1\mathbf{w} = \alpha$ ja $\mathbf{w}^1 = \beta$.

Tarkastellaan binääristä vaihtelevanpituista liitevapaata koodia $C(\mathbf{v}_n)$, missä $\mathbf{v}_n = (k_1, ..., k_n)$.

Vektoria $\mathbf{w} \in \{0,1\}^n$ sanotaan *prefiksivapaaksi* {*suffiksivapaaksi*} yli koodin $C(\mathbf{v}_n)$ jos $C(\mathbf{v}_n)$ ei sisällä yhtään $\mathbf{w}$:n prefiksiä {suffiksia}.

Määritellään

$$
\begin{aligned}
{}^0\overrightarrow{F}(C) &= \{\mathbf{w} \mid \mathbf{w} \text{ on prefiksivapaa yli } C\text{:n ja } {}^1\mathbf{w} = 0\},\\
{}^1\overrightarrow{F}(C) &= \{\mathbf{w} \mid \mathbf{w} \text{ on prefiksivapaa yli } C\text{:n ja } {}^1\mathbf{w} = 1\},\\
\overrightarrow{F}(C) &= {}^0\overrightarrow{F}(C) \cup {}^1\overrightarrow{F}(C),\\
\overleftarrow{F}{}^0(C) &= \{\mathbf{w} \mid \mathbf{w} \text{ on suffiksivapaa yli } C\text{:n ja } \mathbf{w}^1 = 0\},\\
\overleftarrow{F}{}^1(C) &= \{\mathbf{w} \mid \mathbf{w} \text{ on suffiksivapaa yli } C\text{:n ja } \mathbf{w}^1 = 1\},\\
\overleftarrow{F}(C) &= \overleftarrow{F}{}^0(C) \cup \overleftarrow{F}{}^1(C).
\end{aligned}
$$

Olkoon M $\{0,1\}^n$:n mielivaltainen osajoukko.

Joukkoa M sanotaan *oikealta säännölliseksi* jos kaikki M:n sanojen $(n-1)$-suffiksit ovat pareittain erilliset, eli $\forall c_1, c_2 \in M, c_1 \neq c_2$ implikoi $c_1^{n-1} \neq c_2^{n-1}$.

Vastaavasti, joukkoa M sanotaan *vasemmalta säännölliseksi*, jos kaikki M:n sanojen $(n-1)$-prefiksit ovat pareittain erilliset, eli $\forall c_1, c_2 \in M, c_1 \neq c_2$ implikoi ${}^{n-1}c_1 \neq^{n-1} c_2$.

Selvästi, ${}^0\overrightarrow{F}(C)$ ja ${}^1\overrightarrow{F}(C)$ ovat oikealta säännöllisiä joukkoja. Samoin $\overleftarrow{F}{}^0(C)$ ja $\overleftarrow{F}{}^1(C)$ ovat vasemmalta säännöllisiä joukkoja.

Olkoot M_1 ja M_2 $\{0,1\}^n$:n mielivaltaisia osajoukkoja. Määritellään

$$M_1 \otimes M_2 = \{\mathbf{w} \in \{0,1\}^{n+1} | {}^n\mathbf{w} \in M_1 \text{ ja } \mathbf{w}^n \in M_2\}.$$

Seuraava lemma on ilmeinen.

Lemma 2 *Oletetaan, että $C(\mathbf{v}_n)$ on mielivaltainen liitevapaa koodi; tällöin $\overrightarrow{F}(C) \otimes \overleftarrow{F}(C)$ on kaikkien niiden $(n+1)$-pituisten sanojen joukko, jotka voidaan lisätä $C(\mathbf{v}_n)$:ään ilman koodin liitevapaan ominaisuuden rikkomista. Lisäksi, ${}^\alpha\overrightarrow{F}(C) \otimes \overleftarrow{F}{}^\beta(C)$ on kaikkien niiden muotoa $\alpha \star \beta$ ja pituudeltaan $n+1$ olevien sanojen joukko, jotka voidaan lisätä $C(\mathbf{v}_n)$:ään ilman koodin liitevapaan ominaisuuden rikkomista.*

Lemma 3 *Oletetaan, että M_1 on $\{0,1\}^n$:n oikealta säännöllinen osajoukko ja M_2 on $\{0,1\}^n$:n vasemmalta säännöllinen osajoukko; silloin*

$$|M_1 \otimes M_2| \geq |M_1| + |M_2| - 2^{n-1}.$$

Todistus: Merkitään $M_1^{(n-1)}$:llä M_1:n sanojen $(n-1)$-suffiksien joukkoa. Samaan tapaan, merkitään ${}^{(n-1)}M_2$:lla M_2:n sanojen $(n-1)$-prefiksien joukkoa. Koska M_1 on oikealta säännöllinen, seuraa että $|M_1^{(n-1)}| = |M_1|$. Samalla tavoin $|{}^{(n-1)}M_2| = |M_2|$. Koska $M_1^{(n-1)}$ ja ${}^{(n-1)}M_2$ ovat $\{0,1\}^{(n-1)}$:n osajoukkoja, seuraa että $|M_1^{(n-1)} \cup^{(n-1)} M_2| \leq 2^{n-1}$. Siksi $|M_1^{(n-1)} \cap^{(n-1)} M_2| \geq |M_1| + |M_2| - 2^{n-1}$. Merkittäköön $\mathbf{b}$:llä $M_1^{(n-1)} \cap^{(n-1)} M_2$:n mielivaltaista alkiota. Seuraa, että on olemassa $a, c \in \{0,1\}$ niin että $a\mathbf{b} \in M_1$ ja $\mathbf{b}c \in M_2$. Siis $a\mathbf{b}c \in M_1 \otimes M_2$. Siis $|M_1 \otimes M_2| \geq |M_1| + |M_2| - 2^{n-1}$. Tämä päättää todistuksen.

Lause 3 *Jos $S(\mathbf{v}_n) \leq \frac{5}{8}$, niin on olemassa liitevapaa koodi $C(\mathbf{v}_n)$.*

Todistus: Selvästi, riittää todistaa, että $S(\mathbf{v}_n) = \frac{5}{8}$ implikoi liitevapaan koodin $C(\mathbf{v}_n)$ olemassaolon. Tarkastellaan kolmea tapausta.
1) $k_1 = 1$
2) $k_1 = 0, k_2 = 2$
3) $k_1 = 0, k_2 \leq 1$

Jokaisessa tapauksessa muodostamme koodin $C(\mathbf{v}_n)$ n askeleessa. Askeleessa t lisätään k_t t-pituista sanaa koodiin. Askeleen t syöte on koodi $C(\mathbf{v}_{t-1})$, vaste on koodi $C(\mathbf{v}_t)$. Joten askeleella n muodostamme $C(\mathbf{v}_n)$:n.

Tapauksen 1 todistus: Todistamme, että $S(\mathbf{v}_n) \leq \frac{3}{4}$ ja $k_1 = 1$ implikoivat liitevapaan koodin $C(\mathbf{v}_n)$ olemassaolon. Tämä väite on vahvempi kuin lauseen väite. Olkoon $C(\mathbf{v}_1) = \{0\}$. Oletetaan, että muodostetaan liitevapaa koodi $C = C(\mathbf{v}_{t-1})$; todistamme, että askeleella t voimme lisätä k_t t-pituista sanaa koodiin ilman liitevapaan ominaisuuden rikkomista. Lemman 2 mukaan on riittävää, että todistetaan, että $|\overrightarrow{F}(C) \otimes \overleftarrow{F}(C)| \geq k_t$. Olkoon $\delta = S(\mathbf{v}_{t-1})$. Käyttäen (2.1):tä, saamme $\delta + \frac{k_t}{2^t} \leq \frac{3}{4}$. Siten

$$k_t \leq 3 \cdot 2^{t-2} - \delta \cdot 2^t. \tag{2.2}$$

Huomaa, että koska $0 \in C(\mathbf{v}_{t-1})$, seuraa että ${}^0\overrightarrow{F}(C) = \overleftarrow{F}{}^0(C) = \emptyset$. Siksi $\overrightarrow{F}(C)$ on oikealta säännöllinen ja $\overleftarrow{F}(C)$ on vasemmalta säännöllinen. Voidaan helposti tarkistaa, että $|\overrightarrow{F}(C)| = |\overleftarrow{F}(C)| = 2^{t-1}(1 - \delta)$. Käyttäen lemmaa 3 saadaan

$$|\overrightarrow{F}(C) \otimes \overleftarrow{F}(C)| \geq 3 \cdot 2^{t-2} - \delta \cdot 2^t. \tag{2.3}$$

Yhdistämällä (2.2) ja (2.3) saadaan $|\overrightarrow{F}(C) \otimes \overleftarrow{F}(C)| \geq k_t$. Tämä päättää lauseen 3 ensimmäisen tapauksen todistuksen.

Tapauksen 2 todistus: Todistamme, että $S(\mathbf{v}_n) \leq \frac{3}{4}, k_1 = 0$ ja $k_2 = 2$ implikoivat liitevapaan koodin $C(\mathbf{v}_n)$ olemassaolon. Taaskin, väitteemme on vahvempi kuin lauseen väite. Olkoon $C(\mathbf{v}_2) = \{00, 11\}$. Oletetaan, että muodostetaan liitevapaa koodi $C = C(\mathbf{v}_{t-1})$; todistamme, että $|\overrightarrow{F}(C) \otimes \overleftarrow{F}(C)| \geq k_t$. On riittävää todistaa, että molemmat epäyhtälöt (2.2) ja (2.3) toteutuvat. Epäyhtälön (2.2) todistus on täsmälleen sama kuin ylempänä, joten edetään epäyhtälöön (2.3).

Todistamme, että $\overrightarrow{F}(C)$ on oikealta säännöllinen. Oletetaan päinvastainen tapaus. Silloin on olemassa vektori $\mathbf{b} \in \{0,1\}^{t-2}$ niin että molemmat sanat $0\mathbf{b}$ ja $1\mathbf{b}$ ovat prefiksivapaita yli $C(\mathbf{v}_{t-1})$:n. Tarkastellaan kahta tapausta ${}^1\mathbf{b} = 0$ ja ${}^1\mathbf{b} = 1$ erikseen. Ensimmäisessä tapauksessa koodisana 00 on $0\mathbf{b}$:n prefiksinä. Toisessa tapauksessa koodisana 11 on $1\mathbf{b}$:n prefiksinä. Täten olemme tulleet ristiriitaan. Samalla argumentilla $\overleftarrow{F}(C)$ on vasemmalta säännöllinen. Kuten ylempänä, $|\overrightarrow{F}(C)| = |\overleftarrow{F}(C)| = 2^{t-1}(1-\delta)$. Lemman 3 käyttö tuottaa (2.3):n. Tämä päättää lauseen 3 toisen tapauksen todistuksen.

Tapauksen 3 todistus: Koska $k_1 = 0$ ja $k_2 \leq 1$, seuraa että vektori $\mathbf{v}_n$ voidaan yksikäsitteisesti esittää neljän vektorin $\mathbf{v}_n^1, \mathbf{v}_n^2, \mathbf{v}_n^3, \mathbf{v}_n^4$ summana niin että

$$\begin{cases} \mathbf{v}_n^i = \{k_1^i, ..., k_n^i\}, \quad i = 1,2,3,4, \\ S(\mathbf{v}_n^1) = \frac{1}{4} \\ S(\mathbf{v}_n^2) = S(\mathbf{v}_n^3) = S(\mathbf{v}_n^4) = \frac{1}{8} \\ \text{Jos } k_t^i \neq 0, \text{ niin } \forall i' > i, t' < t \quad k_{t'}^{i'} = 0. \end{cases}$$

Tarkastellaan seuraavaa esimerkkiä tällaisesta esityksestä.

$$\mathbf{v}_n = \{0,0,2,1,2,6,20\} \quad S(\mathbf{v}_n) = \frac{5}{8}$$

$$\mathbf{v}_n^1 = \{0,0,2,0,0,0,0\} \quad S(\mathbf{v}_n^1) = \frac{1}{4}$$

$$\mathbf{v}_n^2 = \{0,0,0,1,2,0,0\} \quad S(\mathbf{v}_n^2) = \frac{1}{8}$$

$$\mathbf{v}_n^3 = \{0,0,0,0,0,6,4\} \quad S(\mathbf{v}_n^3) = \frac{1}{8}$$

$$\mathbf{v}_n^4 = \{0,0,0,0,0,0,16\} \quad S(\mathbf{v}_n^4) = \frac{1}{8}$$

Muodostetaan koodi $C(\mathbf{v}_n)$ joka on neljän koodin $C(\mathbf{v}_n) = C^{00}(\mathbf{v}_n^1) \cup C^{01}(\mathbf{v}_n^2) \cup C^{10}(\mathbf{v}_n^3) \cup C^{11}(\mathbf{v}_n^4)$ unioni, missä jokainen koodi $C^{\alpha\beta}(\mathbf{v}_n^i)$ sisältää vain muotoa $\alpha \star \beta$ olevia koodisanoja.

Siten jokaiselle $t = 1, 2, ..., n$ t-pituisten koodisanojen joukko on muodostettu k_t^1 muotoa $0\star0$ olevasta koodisanasta, k_t^2 muotoa $0\star1$ olevasta koodisanasta, k_t^3 muotoa $1\star0$ olevasta koodisanasta ja k_t^4 muotoa $1\star1$ olevasta koodisanasta.

Aloitamme tyhjästä koodista $C(\mathbf{v}_1) = \emptyset$. Oletetaan, että muodostetaan liitevapaa koodi $C = C(\mathbf{v}_{t-1})$; todistamme, että askeleella t koodia voidaan laajentaa k_t^1 $0\star0$ koodisanalla, k_t^2 $0\star1$ koodisanalla, k_t^3 $1\star0$ koodisanalla ja k_t^4 $1\star1$ koodisanalla (t-pituisia) ilman liitevapaan ominaisuuden rikkomista. Lemman 2 mukaan on riittävää todistaa, että

$$|^0\overrightarrow{F}(C) \otimes \overleftarrow{F}^0(C)| \geq k_t^1,$$

$$|^0\overrightarrow{F}(C) \otimes \overleftarrow{F}^1(C)| \geq k_t^2,$$

$$|^1\overrightarrow{F}(C) \otimes \overleftarrow{F}^0(C)| \geq k_t^3,$$

$$|^1\overrightarrow{F}(C) \otimes \overleftarrow{F}^1(C)| \geq k_t^4.$$

Olkoon $\delta_i = S(\mathbf{v}_{t-1}^i)$. Huomaa, että rakenteen mukaan $\delta_i = 0$ ja $\delta_i < S(\mathbf{v}_n^i)$ molemmat implikoivat $\delta_{i+1} = 0$. Tarkastellaan neljää mahdollista tapausta:

1) $\delta_1 < \frac{1}{4}, \delta_2 = \delta_3 = \delta_4 = 0$
2) $\delta_1 = \frac{1}{4}, \delta_2 < \frac{1}{8}, \delta_3 = \delta_4 = 0$
3) $\delta_1 = \frac{1}{4}, \delta_2 = \frac{1}{8}, \delta_3 < \frac{1}{8}, \delta_4 = 0$
4) $\delta_1 = \frac{1}{4}, \delta_2 = \frac{1}{8}, \delta_3 = \frac{1}{8}, \delta_4 < \frac{1}{8}$

Kaikissa tapauksissa käytetään tietoa, että

$$k_t^i \leq 2^t(S(\mathbf{v}_n^i) - \delta_i). \tag{2.4}$$

Tapaus 3.1: $\delta_1 < \frac{1}{4}, \delta_2 = \delta_3 = \delta_4 = 0$. Käyttäen (2.4):a, saamme

$$k_t^1 \leq 2^{t-2} - \delta_1 \cdot 2^t, \quad k_t^2 \leq 2^{t-3},$$

$$k_t^3 \leq 2^{t-3}, \quad k_t^4 \leq 2^{t-3}.$$

Voidaan helposti tarkistaa, että

$$|^0\overrightarrow{F}(C)| = |\overleftarrow{F}^0(C)| = 2^{t-2} - \delta_1 \cdot 2^{t-1},$$

$$|^1\overrightarrow{F}(C)| = |\overleftarrow{F}^1(C)| = 2^{t-2}.$$

Käyttäen lemmaa 3 saadaan

$$|^0\overrightarrow{F}(C) \otimes \overleftarrow{F}^0(C)| \geq 2^{t-2} - \delta_1 \cdot 2^t \geq k_t^1,$$

$|{}^0\overrightarrow{F}(C) \otimes \overleftarrow{F}^1(C)| \geq 2^{t-2} - \delta_1 \cdot 2^{t-1} > 2^{t-3} \geq k_t^2$,
$|{}^1\overrightarrow{F}(C) \otimes \overleftarrow{F}^0(C)| \geq 2^{t-2} - \delta_1 \cdot 2^{t-1} > 2^{t-3} \geq k_t^3$,
$|{}^1\overrightarrow{F}(C) \otimes \overleftarrow{F}^1(C)| \geq 2^{t-2} > k_t^4$.

Tämä päättää tapauksen 3.1 todistuksen.

Tapaus 3.2: $\delta_1 = \frac{1}{4}, \delta_2 < \frac{1}{8}, \delta_3 = \delta_4 = 0$. Samalla argumentilla kuin ylempänä

$$k_t^1 = 0, \quad k_t^2 \leq 2^{t-3} - \delta_2 \cdot 2^t$$

$$k_t^3 \leq 2^{t-3}, \quad k_t^4 \leq 2^{t-3}.$$

Näemme, että

$|{}^0\overrightarrow{F}(C)| = 2^{t-2} - (\frac{1}{4} + \delta_2) \cdot 2^{t-1}$,
$|\overleftarrow{F}^0(C)| = 2^{t-3}$,
$|{}^1\overrightarrow{F}(C)| = 2^{t-2}$,
$|\overleftarrow{F}^1(C)| = 2^{t-2} - \delta_2 \cdot 2^{t-1}$.

Lemman 3 avulla saadaan

$|{}^0\overrightarrow{F}(C) \otimes \overleftarrow{F}^1(C)| \geq 2^{t-3} - \delta_2 \cdot 2^t \geq k_t^2$,
$|{}^1\overrightarrow{F}(C) \otimes \overleftarrow{F}^0(C)| \geq 2^{t-3} \geq k_t^3$,
$|{}^1\overrightarrow{F}(C) \otimes \overleftarrow{F}^1(C)| \geq 2^{t-2} - \delta_2 \cdot 2^{t-1} > 2^{t-3} \geq k_t^4$.

Tämä päättää tapauksen 3.2 todistuksen.

Tapaus 3.3: $\delta_1 = \frac{1}{4}, \delta_2 = \frac{1}{8}, \delta_3 < \frac{1}{8}, \delta_4 = 0$. Kuten ylempänä,

$$k_t^1 = 0, \quad k_t^2 = 0,$$

$$k_t^3 \leq 2^{t-3} - \delta_3 \cdot 2^t, \quad k_t^4 \leq 2^{t-3}.$$

Voidaan helposti osoittaa, että

$|\overleftarrow{F}^0(C)| = 2^{t-3} - \delta_3 \cdot 2^{t-1}$,
$|{}^1\overrightarrow{F}(C)| = 2^{t-2} - \delta_3 \cdot 2^{t-1}$,
$|\overleftarrow{F}^1(C)| = 3 \cdot 2^{t-4}$.

Käyttäen lemmaa 3 saadaan

$$|^1\overrightarrow{F}(C) \otimes \overleftarrow{F}^0(C)| \geq 2^{t-3} - \delta_3 \cdot 2^t \geq k_t^3,$$
$$|^1\overrightarrow{F}(C) \otimes \overleftarrow{F}^1(C)| \geq 3 \cdot 2^{t-4} - \delta_3 \cdot 2^{t-1} > 2^{t-3} \geq k_t^4.$$

Tämä päättää tapauksen 3.3 todistuksen.

Tapaus 3.4: $\delta_1 = \frac{1}{4}, \delta_2 = \frac{1}{8}, \delta_3 = \frac{1}{8}, \delta_4 < \frac{1}{8}$. Kuten ylempänä,

$$k_t^1 = 0, \quad k_t^2 = 0,$$
$$k_t^3 = 0, \quad k_t^4 \leq 2^{t-3} - \delta_4 \cdot 2^t.$$

Voidaan helposti nähdä, että

$$|^1\overrightarrow{F}(C)| = |\overleftarrow{F}^1(C)| = 2^{t-2} - (\frac{1}{8} + \delta_4) \cdot 2^{t-1}.$$

Lemman 3 avulla saadaan

$$|^1\overrightarrow{F}(C) \otimes \overleftarrow{F}^1(C)| \geq 2^{t-3} - \delta_4 \cdot 2^t \geq k_t^4.$$

Tämä päättää lauseen todistuksen.

2.2.3 Yläraja redundanssille

Lause 4 *Jokaiselle todennäköisyysjakaumalle* $p = \{p_1, ..., p_m\}$ *on olemassa binäärinen liitevapaa koodi* C *jossa koodisanojen keskimääräinen pituus* $L(C)$ *toteuttaa*

$$L(C) < H(p) + 4 - \log_2 5.$$

Todistus: Merkitään koodisanojen pituuksia $l_1, ..., l_m$:llä. Määritellään

$$l_i = \lceil -\log_2 p_i + 3 - \log_2 5 \rceil.$$

Seuraa, että

$$\sum_{i=1}^{m} 2^{-l_i} \leq \sum_{i=1}^{m} 2^{\log_2 p_i - 3 + \log_2 5} = \frac{5}{8} \sum_{i=1}^{m} p_i = \frac{5}{8}.$$

Lauseen 3 mukaan on olemassa liitevapaa koodi C koodisanan pituuksilla $l_1, ..., l_m$. Tämän koodin keskimääräinen pituus on

$$L(C) = \sum_{i=1}^{m} p_i \cdot l_i < \sum_{i=1}^{m} p_i(-\log_2 p_i + 4 - \log_2 5)$$

$$= H(p) + (4 - \log_2 5) \sum_{i=1}^{m} p_i = H(p) + 4 - \log_2 5.$$

Tämä päättää todistuksen.

2.2.4 Alaraja redundanssille

Lause 5 *Jokaiselle todennäköisyysjakaumalle $p = (p(1), ..., p(N))$ on olemassa binäärinen liitevapaa koodi C jossa keskimääräinen koodisanojen pituus toteuttaa*

$$H(p) \leq L(C).$$

Todistus: Lause on selvästi tosi, koska jokainen liitevapaa koodi on prefiksikoodi ja jokaiselle prefiksikoodille lause seuraa kohinattomasta koodauslauseesta (Noiseless Coding Theorem). Kohinaton koodauslause on esitetty [10]:ssa.

2.3 Täydelliset vaihtelevanpituiset liitevapaat koodit

Koodisanojen joukon $\{x_1, x_2, ..., x_n\}$ yli t-kirjaimisen aakkoston Σ sanotaan olevan *täydellinen* jos se toteuttaa Kraftin epäyhtälön yhtäsuuruudella, eli

$$\Sigma_{1 \leq i \leq n} t^{-|x_i|} = 1.$$

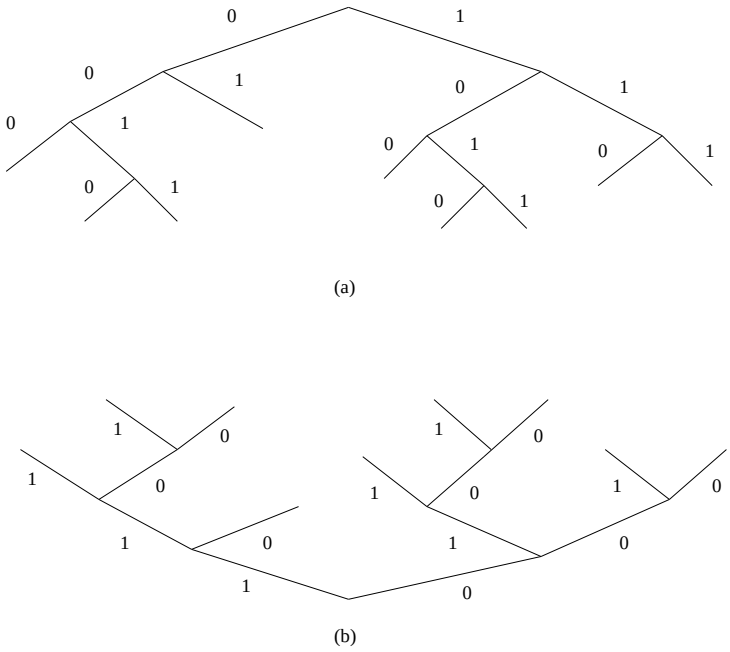
Kaikkien koodisanojen joukko Σ^k on selkeästi liitevapaa ja täydellinen. Osoitamme, että on muitakin esimerkkejä täydellisistä liitevapaista koodeista, sellaisista joiden koodisanat ovat vaihtelevanpituisia. Käsittelemme sellaisia vaihtelevanpituisia (täydellisiä) liitevapaita koodeja.

Väärä konjektuuri. Täydellisen liitevapaan koodin yli annetun t-kirjaimisen aakkoston Σ täytyy olla muotoa Σ^k jollakin kokonaisluvulla k. Eli ei ole olemassa vaihtelevanpituisia täydellisiä liitevapaita koodeja.

Artikkelin [4] kirjoittajat yrittivät todistaa tätä konjektuuria, mutta yllätykseksi huomasivat, että se on väärä. Tässä on vastaesimerkki, jonka he löysivät (yli binäärisen aakkoston):

$$A = \{01, 000, 100, 110, 111, 0010, 0011, 1010, 1011\}.$$

Prefiksipuu ja suffiksipuu koodille A ovat annettu kuvassa 3.



Kuva 3. (a) Prefiksipuu ja (b) suffiksipuu liitevapaalle koodille $A = \{01, 000, 100, 110, 111, 0010, 0011, 1010, 1011\}$.

Mikä tahansa vastaesimerkki konjektuurille luo perheen vastaesimerkkejä muodostamalla tuloja: olkoon A^k yhteenliitettyjen k:n sanan joukko A:sta. Koska A on liitevapaa, A^k:n täytyy myös olla liitevapaa.

Kun tiedetään, että konjektuuri on virheellinen, on luonnollista kysyä yleisiä menetelmiä liitevapaiden koodien muodostamiseksi, aivan kuten Huffman-koodejakin voidaan muodostaa. Liitevapaiden koodien muodostamisen ongelma vaikuttaa kuitenkin paljon vaikeammalta, emmekä tiedä kuinka muodostaa kaikki sellaiset koodit. Sovelluksissa vaikuttaisi hyödylliseltä olla koodeja joissa on mielivaltaisen suuri suhde pisimpien ja lyhyimpien koodisanojen välillä. Juuri kuvaillussa joukossa A pisimpien ja lyhyimpien sanojen suhde on 2. A on esimerkki yleisestä muodostamisesta jossa pisimmän ja lyhyimmän sanan suhde lähestyy 3:a. Artikkelissa [4] esitetään tällainen muodostusmenetelmä. Artikkelissa yleistetään tämä kehittämällä myös rekursiivinen muodostusmenetelmä, joka antaa mielivaltaisen suuren suhteen. Kirjoittajat jättävät avoimeksi kysymyksen tehokkaiden liitevapaiden koodien muodostamisesta, kun lähdeaakkoston todennäköisyysjakauma on annettu.

2.4 Algoritmi tehokkaiden liitevapaiden koodien muodostamiseksi

2.4.1 Muodostusalgoritmeista

Algoritmeja tehokkaiden liitevapaiden koodien muodostamiseksi käsitellään viitteissä [9], [3] ja [14]. Lisäksi [8]:ssa käsitellään sellaisen liitevapaan koodin muodostamista, jossa minimietäisyys $d_f = 2$. Tyypillisesti minimietäisyys on liitevapaissa koodeissa 1. Korkeammasta minimietäisyydestä on apua virheiden havaitsemisessa tiedonsiirrossa.

Seuraavassa käsitellään [9]:n sisältöä. Artikkelin kirjoittajat ehdottavat algoritmia liitevapaiden koodien muodostamiseksi, joka pitää sisällään uuden koodisanan valintamekanismin. Aloittaen lyhimmistä koodisanoista, ehdotettu algoritmi valitsee minkä tahansa i:n pituiset koodisanat maksimoiden $i + 1$:n pituisten valittavien koodisanojen määrän. Kasvaneen valittavissa olevien koodisanojen määrän ansiosta ehdotettu algoritmi muodostaa tehokkaampia koodeja suhteessa muihin kirjallisuudessa esiintyviin algoritmeihin.

2.4.2 Johdanto algoritmiin

Algoritmeja liitevapaiden koodien muodostamiseksi käsiteltiin viitteissä [12], [13]. Kun Takishima *et al.* [12] käsitteli symmetristen ja asymmetristen koo-

dien muodostamista erillisesti, Tsai ja Wu [13] esittelivät geneerisen algoritmin joka pitää sisällään valintamekanismin symmetrisille ja asymmetrisille liitevapaille koodeille. Symmetriset koodit ovat sellaisia, joissa koodisana on sama etuperin tai takaperin luettuna. Asymmetrisissä koodeissa koodisana voi olla eri etuperin tai takaperin luettuna. Molemmissa tapauksissa osoitettiin, että asymmetriset koodit ovat merkittävästi tehokkaampia; siksi kirjoittajat rajoittavat huomionsa näiden koodien muodostamiseen.

Liitevapaiden koodien muodostusalgoritmit yleisesti käyttävät Huffman-koodia lähtökohtana [12], [13]. Koska Huffman-koodien koodisananpituusjakauma on optimaalinen prefiksikoodijakauma, liitevapaan koodin muodostus aloitetaan valitsemalla lyhimmät liitevapaat koodisanat, joilla on sama pituus kuin lyhimmillä Huffman-koodisanoilla. Pidemmät liitevapaat koodisanat valitaan samanpituisiksi kuin Huffman-koodisanat, edellyttäen että on sen pituisia koodisanoja, jotka samanaikaisesti täyttävät sekä prefiksi- että suffiksiehdot. Siinä tapauksessa, että sellaisia koodisanoja ei ole, liitevapaan koodisanan pituutta täytyy lisätä, mikä pienentää liitevapaan koodin tehokkuutta.

Tietyn pituisten prefiksi- ja suffiksiehdon täyttävien koodisanojen lukumäärä riippuu aiemmin määrätyistä (lyhyemmistä) liitevapaista koodisanoista. Tätä riippuvuutta ei huomioitu liitevapaan koodin muodostusalgoritmin kehityksessä [12]:ssä, siksi tämä algoritmi tuottaa suhteellisen tehottomia liitevapaita koodeja. [13]:ssä esitettiin konjektuuri, että käytettävissä olevien (prefiksi- ja suffiksiehdon täyttävien) minkä tahansa pituisten koodisanojen lukumäärä riippuu metriikasta nimeltä minimi toistoväli (minimum repetition gap, MRG) assosioituna lyhyempien liitevapaiden koodisanojen kanssa. [13]:ssä esitetty algoritmi suorittaa liitevapaan koodisanan määräyksen MRG:n perusteella ja yleisesti tuottaa tehokkaampia koodeja kuin [12]:sä ehdotettu algoritmi. On mahdollista muodostaa formaali suhde käytettävissä olevien minkä tahansa pituisten koodisanojen lukumäärän ja lyhyempien liitevapaiden koodisanojen rakenteen välille. Seuraavassa alakappaleessa esitetään tämä suhde muodossa, jota voidaan helposti hyödyntää liitevapaan koodin muodostusalgoritmissa. Osoitetaan, että käytettävissä olevien liitevapaiden koodisanojen määrä ei riipu yksinomaan MRG:stä, ja käytetään tätä tulosta ehdotukseen uudesta liitevapaan koodin muodostusalgoritmista joka tuottaa tehokkaampia liitevapaita koodeja kuin algoritmit [12]:ssä ja [13]:ssä.

2.4.3 Affiksi-indeksit ja käytettävissä olevat liitevapaat koodit

Merkitään i:n pituista binääristä koodisanaa $w = (w_1 w_2 ... w_i)$. Määritellään prefiksijoukko $P_j(w)$ joukoksi j-pituisia koodisanoja, $j > i$, joiden prefiksi on w

$$P_j(w) = \{(x_1 x_2 ... x_j) | (x_1 x_2 ... x_i) = (w_1 w_2 ... w_i)\}.$$

Määritellään suffiksijoukko $S_j(w)$ joukoksi j-pituisia koodisanoja, $j > i$, joiden suffiksi on w

$$S_j(w) = \{(x_1 x_2 ... x_j) | (x_{j-i+1} x_{j-i+2} ... x_j) = (w_1 w_2 ... w_i)\}.$$

Prefiksijoukon kardinaliteetti $|P_j(w)|$ on selvästi yhtäsuuri suffiksijoukon kardinaliteetin kanssa

$$|P_j(w)| = |S_j(w)| = 2^{j-i}.$$

Oletetaan, että $W = \{w^1, w^2, ..., w^m\}$ on joukko liitevapaita koodisanoja, joiden pituudet ovat $i^1 \leq i^2 \leq ... \leq i^m$. Merkitään $n_a(j)$:llä j-pituisten koodisanojen määrää, $j > i^m$, joiden prefiksinä tai suffiksina ei ole mikään koodisana W:stä. Koska j-pituisten koodisanojen kokonaismäärä on 2^j, pätee

$$n_a(j) = 2^j - |(\bigcup_{w^k \in W} P_j(w^k)) \cup (\bigcup_{w^k \in W} S_j(w^k))|$$

$$= 2^j - |\bigcup_{w^k \in W} P_j(w^k)| - |\bigcup_{w^k \in W} S_j(w^k)| + |\bigcup_{w^k, w^l \in W} P_j(w^k) \cap S_j(w^l)|.$$

Muistutamme, että mikään liitevapaa koodisana ei ole toisen koodisanan prefiksi eikä suffiksi. Tämä implikoi, että kaikki prefiksijoukot, kuten suffiksijoukotkin, ovat erillisiä eli

$$|P_j(w^k) \cap P_j(w^l)| = |S_j(w^k) \cap S_j(w^l)| = 0, \quad w^k, w^l \in W, k \neq l.$$

Siksi

$$n_a(j) = 2^j - \sum_{w^k \in W} (|P_j(w^k)| + |S_j(w^k)|) + \sum_{w^k \in W} \sum_{w^l \in W} |P_j(w^k) \cap S_j(w^l)| \tag{2.5}$$

$$= 2^j - \sum_{k=1}^{m} 2^{j-i^k+1} + \sum_{w^k \in W} \sum_{w^l \in W} |P_j(w^k) \cap S_j(w^l)|. \tag{2.6}$$

Määritellään *affiksi joukko* $A_j(w^k, w^l)$ joukoksi j-pituisia koodisanoja, $j > \max(i^k, i^l)$, joiden prefiksi on w^k ja suffiksi on w^l

$$A_j(w^k, w^l) = P_j(w^k) \cap S_j(w^l). \tag{2.7}$$

Selvästi, $n_a(j)$ riippuu affiksijoukkojen $A_j(w^k, w^l), \quad w^k, w^l \in W$ kardinaliteeteista eli joukon W affiksi-indeksistä, joka määritellään

$$a_j(W) = \sum_{w^k \in W} \sum_{w^l \in W} |(A_j(w^k, w^l)|. \tag{2.8}$$

Voidaan osoittaa, että affiksijoukkojen $A_j(w^k, w^l)$ kardinaliteetit ovat seuraavat:
Tapauksessa, että $\max(i^k, i^l) < j < i^k + i^l$

$$|(A_j(w^k, w^l)| = \begin{cases} 1, & \text{jos } (w^k_{j-i^l+1} w^k_{j-i^l+2} ... w^k_{i^k}) = (w^l_1 w^l_2 ... w^l_{i^k+i^l-j}) \\ 0, & \text{jos } (w^k_{j-i^l+1} w^k_{j-i^l+2} ... w^k_{i^k}) \neq (w^l_1 w^l_2 ... w^l_{i^k+i^l-j}) \end{cases} \tag{2.9}$$

ja kun $i^k + i^l \leq j$

$$|(A_j(w^k, w^l)| = 2^{j-i^k-i^l}.$$

Havainnollistaaksemme suhdetta käytettävissä olevien liitevapaiden koodien ja lyhyempien koodisanojen rakenteen välillä, tarkastellaan kahta koodisanajoukkoa $W_1 = \{000, 001, 010, 100\}$ ja $W_2 = \{000, 010, 101, 111\}$. Affiksijoukot $A_4(w^k, w^l)$ koodisanoille W_1:stä ja W_2:sta on annettu taulukossa 1.

w^k	w^l	$A_4(w^k, w^l)$	$A_4(w^l, w^k)$	w_k	w^l	$A_4(w^k, w^l)$	$A_4(w^l, w^k)$
000	000	{0000}	{0000}	000	000	{0000}	{0000}
000	001	{0001}	∅	000	010	∅	∅
000	010	∅	∅	000	101	∅	∅
000	100	∅	{1000}	000	111	∅	∅
001	001	∅	∅	010	010	∅	∅
001	010	{0010}	∅	010	101	{0101}	{1010}
001	100	∅	{1001}	010	111	∅	∅
010	010	∅	∅	101	101	∅	∅
010	100	{0100}	∅	101	111	∅	∅
100	100	∅	∅	111	111	{1111}	{1111}

Taulukko 1. Esimerkkejä affiksijoukoista

Käyttäen taulukkoa 1, (2.8):a ja (2.6):a, voidaan laskea että $a_4(W_1) = 6$ ja $n_a(4) = 6$ W_1:lle, kun taas $a_4(W_2) = 4$ ja $n_a(4) = 4$ W_2:lle.

Tarkastellaan koodisanojen W_1 ja W_2 minimitoistovälejä. Minimitoistoväliä käytetään koodisanan valintakriteerinä Tsain algoritmissa [12], ja se voidaan tulkita seuraavasti: i-pituisen koodisanan w minimitoistoväli, jota merkitään $g(w)$:llä, on yhtäsuuri kuin minimaalinen positiivinen kokonaisluku g, jolle $|A_{i+g}(w, w)| > 0$. Algoritmi [13]:ssä valitsee koodisanat, joilla on pienin $g(w)$, mikä aiheuttaa kasvun $n_a(i + g)$:hen, pienin g ensimmäisenä, koska $n_a(i + g)$ riippuu $|A_{i+g}(w, w)|$:stä kuten (2.6):ssa. Kuitenkaan tämä algoritmi ei ota huomioon alkioita $|A_{i+g}(w^k, w^l)|, w^k \neq w^l$, joilla on merkittävä vaikutus $n_a(i + g)$:hen, ja vastaavasti koodauksen tehokkuuteen. W_1:n ja W_2:n välillä Tsain algoritmi valitsee W_2:n, koska $g(000) = g(111) = 1$ ja $g(010) = g(101) = 2$ kun taas $g(001) = g(100) = 3$. Kuitenkin joukko W_1 on parempi koodauksen tehokkuuden kannalta [$n_a(4) = 6$ ja $n_a(5) = 10$ W_1:lle kun taas $n_a(4) = 4$ ja $n_a(5) = 8$ W_2:lle]. Algoritmi jota ehdotetaan seuraavassa alakappaleessa suorittaa liitevapaiden koodisanajoukkojen valinnan niiden affiksi-indeksien perusteella, mikä aiheuttaa kasvaneen käytettävissä olevien pidempien koodisanojen määrän ja parantuneen koodaustehokkuuden.

2.4.4 Algoritmi tehokkaiden liitevapaiden koodien muodostamiseksi

Tarkastellaan liitevapaan koodin muodostamista M:ää eri kirjainta sisältävälle (M-ary) i.i.d. informaatiolähteelle $U = \{u_1, ..., u^M\}$, jolla on todennäköisyysmassafunktio $p_u = \{p^1, ..., p^M\}, p^1 \leq ... \leq p^M$. Ehdotetttu liitevapaan koodin muodostusalgoritmi sisältää seuraavat askeleet (askeleet 1, 2.2 ja 3 ovat samat kuin algoritmissa [13]:ssa, kun taas askel 2.1 sisältää uuden koodisanan valintamekanismin):

1) Muodosta Huffman-koodi C_H, joka kuvaa lähdesymbolit vastaaviksi binäärisiksi koodisanoiksi $\{c_H(u^1), ..., c_H(u^M)\} = \{w_H^1, ..., w_H^M\}$, joilla on pituus

$$\{l_H(u^1), ..., l_H(u^M)\} = \{i_H^1, ..., i_H^M\}$$

$$L_H^{\min} = i_H^1 \leq i_H^2 \leq ... \leq i_H^{M-1} = i_H^M = L_H^{\max}.$$

Merkitse C_H:n bittipituusvektoria $(n_H(1), ..., n_H(L_H^{\max}))$:llä, missä $n_H(i)$ esittää i-pituisten Huffman-koodisanojen pituutta. Anna alkuarvo määrättyjen

koodisanojen lukumäärälle, $m = 0$, ja liitevapaan koodin bittipituusvektorille $(n(1), ..., n(L_H^{\max})), n(i) = n_H(i), \quad i = 1, ..., L_H^{\max}$. Aloita liitevapaiden koodisanojen määrääminen tasolta $i = L_H^{\min}$.

2) Tunnista käytettävissä olevien i-pituisten koodisanojen joukko, jossa aikaisemmilla tasoilla määrätyt koodisanat eivät ole prefiksinä eikä suffiksina alkioille. Käytettävissä olevien koodisanojen määrä, jota merkitään $n_a(i)$:llä, on yhtäsuuri kuin 2^i tasolla $i = L_H^{\min}$. Tasoilla $i > L_H^{\min}$, $n_a(i)$ riippuu aikaisemmin määrätyistä RVLC koodisanoista $W = \{w^1, ..., w^m\}$
$= \{c(u^1), ..., c(u^m)\}$, kuten on annettu (2.6):ssa.

2.1) Jos $n_a(i) > n(i)$ (olen korjannut tämän $n_a(i) > n(i)$, alkuperäisessä artikkelissa se on $n_a(i) \geq n(i)$), niin $n(i)$ koodisanaa pitää määrätä. Olettaen, että $X_a(i, W) = \{x^1, ..., x^{n_a(i)}\}$:lla merkitään käytettävissä olevin koodisanojen joukkoa, kandidaattivalikoima liitevapaita koodisanoja on mikä tahansa $n(i)$-alkioinen osajoukko $X \subseteq X_a(i, W)$, ja kaikkien kandidaatti koodisanojen valikoimien joukko voidaan kirjoittaa

$$\underline{X}(i, W) = \{X | X \subseteq X_a(i, W), |X| = n(i)\}.$$

Olettaen, että liitevapaiksi koodisanoiksi on valittu koodisanat joukosta $X_s \in \underline{X}(i, W)$, niin (2.6)-(2.8):n mukaan käytettävissä olevien koodisanojen lukumäärä $n_a(j)$, tasolla $j > i$, on

$$n_a(j) = 2^j - \sum_{k=1}^{m} 2^{j-i^k+1} - n(i) \cdot 2^{j-i+1} + a_j(X_s \cup W).$$

Korkein $n_a(j)$ saadaan valitsemalla joukko, jossa on maksimaalinen $a_j(X_s \cup W)$, merkitään sitä

$$X_{s*}(j) = \arg \max_{X_s \in \underline{X}(i,W)} a_j(X_s \cup W).$$

Kuitenkaan, kuten voidan päätellä (2.9):stä, valikoima joka maksimoi $n_a(j_1)$:n ei välttämättä maksimoi $n_a(j_2)$:a eli yleisesti pätee, että $X_{s*}(j_1) \neq X_{s*}(j_2), i < j_1 < j_2$.

Siksi suoritamme valinnan seuraavaan tapaan: Tasolla i määräämme koodisanat joukosta

$$X_{s*}(i+1) = \arg \max_{X_s \in \underline{X}(i,W)} a_{i+1}(X_s \cup W).$$

Tämä valinta maksimoi käytettävissä olevien $i+1$-pituisten koodisanojen määrän. Kun algoritmi etenee tasolle $i+1$, suoritettu valinta maksimoi $n_a(i+2)$:n jne. Siten joka tasolla liitevapaiden koodien muodostamisalgoritmissa ehdotettu lähestymistapa suorittaa sellaisen koodisanojen valinnan, joka maksimoi käytettävissä olevien koodisanojen määrän seuraavalla tasolla, mikä tekee tästä lähestymistavasta tasolta-tasolle optimaalisen.

Huomaa, että on mahdollista olla useita kandidaattijoukkoja $X_{s_c} \in \underline{X}(i, W)$ joille $a_{i+1}(X_{s_c} \cup W) = a_{i+1}(X_{s*}(i+1) \cup W)$, jotka saavat saman $n_a(i+1)$. Siinä tapauksessa valitsemme joukon X_{s_c},jolla saadaan korkein $n_a(i+2)$, tarkastelemalla kandidaattivalikoimia $\underline{X}(i+1, W \cup X_{s_c})$ kaikilla kandidaattijoukoilla X_{s_c}, ja valitsemalla joukon X_{s_c} , jolla on suurin vastaava $\max_{X \in \underline{X}(i+1, W \cup X_{s_c})} a_{i+2}(X \cup X_{s_c} \cup W)$.

2.2) Jos $n_a(i) \leq n(i)$ (olen korjannut tämän $n_a(i) \leq n(i)$, alkuperäisessä artikkelissa se on $n_a(i) < n(i)$), niin kaikki käytettävissä olevat liitepapaat koodisanat määrätään ja bittipituusvektoria korjataan:

$$n(i+1) = n(i+1) + n(i) - n_a(i), \quad n(i) = n_a(i).$$

3) Määrättyjen koodisanojen lukumäärä päivitetään: $m = m + n(i)$. Jos $m < M$, niin koodisanan määrääminen (askel 2) jatkuu tasolla $i = i + 1$. Muutoin muodostus päättyy.

Tämä algoritmi on kompleksisuudeltaan korkeampi kuin liitevapaiden koodien muodostusalgoritmi [13]:ssa. Joka tasolla i, kun $n_a(i) \geq n(i)$, Tsain algoritmissa tutkitaan $n_a(i)$ MRG:tä, kun taas ehdotetussa algoritmissa tutkitaan $\binom{n_a(i)}{n(i)}$ affiksi-indeksiä.

Taulukko 2 vertailee liitevapaita koodeja Englannin kielen aakkostolle. Liitevapaat koodit on muodostettu käyttäen ehdotettua algoritmia ja algoritmeja, jotka on julkaistu [12]:ssa ja [13]:ssa. Parantuneen koodisanan valintamekanismin ansiosta ehdotetulla algoritmilla saadaan liitevapaa koodi, joka on merkittävästi tehokkaampi.

u	$p_U(u)$	Huffman-koodi	Takishiman RVLC	Tsain RVLC	Ehdotettu RVLC
E	0.14878	001	001	000	000
T	0.09351	110	110	111	001
A	0.08833	0000	0000	0101	0100
O	0.07245	0100	0100	1010	0101
R	0.06872	0110	1000	0010	0110
N	0.06498	1000	1010	1101	1010
H	0.05831	1010	0101	0100	1011
I	0.05644	1110	11100	1011	1100
S	0.05537	0101	01100	0110	1101
D	0.04376	00010	00010	11001	01110
L	0.04124	10110	10010	10011	01111
U	0.02762	10010	01111	01110	10010
P	0.02575	11110	10111	10001	10011
F	0.02455	01111	11111	001100	11110
M	0.02361	10111	111101	011110	11111
C	0.02081	11111	101101	100001	100010
W	0.01868	000111	000111	1001001	100011
G	0.01521	011100	011101	0011100	1000010
Y	0.01521	100110	100111	1100011	1000011
B	0.01267	011101	1001101	0111110	1110111
V	0.01160	100111	01110011	1000001	10000010
K	0.00867	0001100	00011011	00111100	10000011
X	0.00146	00011011	000110011	11000011	11100111
J	0.00080	000110101	0001101011	100101001	100000010
Q	0.00080	0001101001	00011010011	0011101001	1000000010
Z	0.00053	0001101000	000110100011	1001011100	1000000111
	Keskim.pituus	4.15572	4.36068	4.30678	4.25145
	Tehokkuus	0.99161	0.94500	0.95682	0.96928
	Kraftin summa	1	0.87867	0.91016	0.94531

Taulukko 2. Koodit Englannin kielen aakkostolle: Huffman-koodi, RVLC:t [12]:sta ja [13]:sta ja RVLC muodostettuna käyttäen ehdotettua algoritmia. (RVLC = liitevapaa koodi).

Kirjoittajat ovat siis ehdottaneet algoritmia tehokkaiden liitevapaiden koodien muodostamiseksi, mikä sisältää uuden koodisanan valintamekanismin. Ehdotettu algoritmi hyödyntää riippuvuutta valittuihin koodisanoihin liittyvien affiksi-indeksien ja pidempien käytettävissä olevien koodisanojen välillä. Tämä algoritmi sisältää joka tasolla korkeimman affiksi-indeksin omaavan koodisanajoukon valinnan, mikä johtaa lisääntyneeseen pidempien käytettävissä olevien liitevapaiden koodisanojen määrään, ja siten tuottaa tehokkaampia liitevapaita koodeja suhteessa muihin algoritmeihin kirjallisuudessa.

Luku 3

Johtopäätökset

Liitevapailla koodeilla on hyvät synkronointiominaisuudet. Bittivirheen tapahtuminen ei automaattisesti merkitse lohkon loppuosan menetystä. Tämä tekee niistä ylivoimaisen tavallisiin prefiksikoodeihin verrattuna tietyissä sovelluksissa. Tällainen sovellus on esimerkiksi videokuvan siirto. Siinä ei aina käytetä kanavakoodausta, mikä merkitsee, että bittivirheitä esiintyy siirretyssä datassa.

Liitevapaiden koodien redundanssi on tyypillisesti suurempi kuin tavallisissa prefiksikoodeissa. On kuitenkin osoittautunut, että redundanssin lisäys on marginaalinen, kun käytetään tehokasta liitevapaiden koodien muodostamisalgoritmia.

Viitteet

[1] R. Ahlswede, B. Balkenhol ja L. Khachatrian,"Some properties of fix-free codes", *Proceedings of first INTAS International Seminar on Coding Theory and Combinatorics*, Thahkadzor, Armenia, pp.20-33, 1996

[2] T. M. Cover ja J. A. Thomas, *Elements of Information Theory*, Wiley, New York, 1991

[3] W-H. Jeong, Y-S. Yoon ja Y-S. Ho, "Design of Reversible Variable-length Codes Using Properties of the Huffman Code and Average Length Function", *International Conference on Image Processing (ICIP)*, Vol. 2, pp. 817-820, 2004

[4] D. Gillman ja R. L. Rivest, "Complete Variable-length "Fix-Free" Codes", *Designs, Codes and Cryptography*, Vol. 5, pp.109-114, 1995.

[5] K. Harada ja K. Kobayashi, "A note on the fix-free property", *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, Vol. E82-A, pp.2121-2128, 1999

[6] Z. Kukorelly ja K. Zeger, "New binary fix-free codes with Kraft sum 3/4", *Proceedings of International Symposium on Information Theory*, Lausanne, Switzerland, p.178, 2002

[7] Z. Kukorelly ja K. Zeger, "Sufficient Conditions for Existence of Binary Fix-Free Codes", *IEEE Transactions on Information Theory*, Vol. 51, pp.3433-3444, 2005

[8] K. Laković ja J. Villasenor, "On Design of Error-Correcting Reversible Variable Length Codes", *IEEE Communications Letters*, Vol. 6, pp.337-339, 2002

[9] K. Laković ja J. Villasenor, "An Algorithm for Construction of Efficient Fix-Free Codes", *IEEE Communications Letters*, Vol. 7, pp.391-393, 2003

[10] C.E. Shannon ja W. Weaver, *The Mathematical Theory of Communication*, University of Illinois Press, Urbana, IL, 1949

[11] Y. Q. Shi ja H. Sun, *Image and Video Compression for Multimedia Engineering: Fundamentals, Algorithms, and Standards*, CRC Press, Boca Raton, Florida, 1999.

[12] Y. Takishima, M. Wada ja H. Murakami, "Reversible variable length codes", *IEEE Transactions on Communications*, Vol. 43, pp.158-162, 1995

[13] C. W. Tsai ja J. L. Wu, "On constructing the Huffman-code based reversible variable length codes", *IEEE Transactions on Communications*, Vol. 49, pp.1506-1509, 2001

[14] H-W. Tseng ja C-C. Chang, "A Branch-and-Bound Algorithm for the Construction of Reversible Variable Length Codes", *The Computer Journal*, Vol.47, pp.701-707, 2004

[15] C. Ye ja R.W. Yeung, "On fix-free codes", *Proceedings of International Symposium on Information Theory*, Sorrento, Italy, p.426, 2000

[16] C. Ye ja R. W. Yeung, "Some basic properties of fix-free codes", *IEEE Transactions on Information Theory*, Vol. 47, pp.72-87, 2001

[17] S. Yekhanin, "Sufficient conditions of existence of fix-free codes", *Proceedings of International Symposium on Information Theory*, Washington D.C., USA, p.284, 2001

[18] S. Yekhanin, "Improved Upper Bound for the Redundancy of Fix-Free Codes", *Proceedings of International Symposium on Information Theory*, Yokohama, Japan, p.80, 2003

[19] S. Yekhanin, "Improved Upper Bound for the Redundancy of Fix-Free Codes", *IEEE Transactions on Information Theory*, Vol. 50, pp.2815-2818, 2004