LUONNOLLISEN KIELEN KONEELLINEN KÄÄNTÄMINEN

TURUN YLIOPISTO
Informaatioteknologian laitos
Tietotekniikka

TkK-tutkielma

Joulukuu 2009

Tuomas Tähti

TURUN YLIOPISTO
Informaatioteknologian laitos / Matemaattis-luonnontieteellinen tiedekunta

TÄHTI, TUOMAS: Luonnollisen kielen koneellinen kääntäminen

TkK-tutkielma, 27 s.
Tietotekniikka
Joulukuu 2009

Luonnollisen kielen koneellinen kääntäminen on laaja ongelma, jota on aktiivisesti pyritty ratkaisemaan yli puolen vuosisadan ajan. Koneellisten käännösmekanismien laajamittainen kehittäminen alkoi 1950-luvulla, jolloin alan kehitysmahdollisuudet kiinnostivat etenkin armeijaa ja tiedeyhteisöä. 1960-luvun kuluessa kävi kuitenkin selväksi, ettei laadukkaita konekäännöksiä saada aikaan kääntämällä tekstejä automaattisesti sanasta sanaan, vaan koneilla pitää sanaston ja syntaksin ohella olla tuntemusta myös semantiikasta.

1980-luvulla käännösongelmaan kehitettiin uusia lähestymistapoja, jotka poikkesivat merkittävästi aiemmista ratkaisuyrityksistä. Noista ajoista lähtien konekääntämisen metodit on voitu jakaa kolmeen pääryhmään: tietopohjaisiin, tilastollisiin ja esimerkkipohjaisiin.

Kansainvälistyvässä maailmassa kysyntä nopeasti ja tehokkaasti tuotettaville konekäännöksille ei ole ainakaan vähenemässä. Eräät uudenaikaiset ohjelmat voivat tulostaa jo 90-prosenttisen tarkkoja käännöksiä, mutta eivät yllä ammattitaitoisen ihmiskääntäjän tarkkuuteen. Alalla onkin odotettavissa vielä runsaasti menetelmiin kohdistuvaa kehitystyötä.

Tässä tutkielmassa luodaan aluksi lyhyt katsaus luonnollisen kielen koneellisen kääntämisen ongelman historiaan ja perusteisiin. Tämän jälkeen tarkastellaan yllämainittuja ratkaisutapoja käsitellen sekä niiden periaatteellista toimintaa että matemaattista pohjaa. Käännöstapojen eroavaisuudet ja vahvuusalueet tuodaan selkeästi esille. Tutkielma rajoittuu kirjoitetun tekstin kääntämiseen, joten puheentunnistusta ei käsitellä. Tutkielman ulkopuolelle rajataan myös hybridikääntäjät, jotka on toteutettu yhdistelemällä piirteitä eri käännöstavoista.

Asiasanat: kieli, kääntäminen, sanasto, syntaksi, semantiikka

Sisällys

1 Johdanto

Luonnollisen kielen koneellisten käännösmekanismien laajamittainen kehittäminen alkoi 1950-luvulla. Tuolloin konekääntämisen mahdollisuudet kiinnostivat etenkin armeijaa ja tiedeyhteisöjä, joiden odotukset olivat suuret. 1960-luvun kuluessa kävi kuitenkin selväksi, ettei laadukkaita konekäännöksiä saada aikaan yksinkertaisilla, automatisaatiota muistuttavilla metodeilla, vaan koneilla pitää sanaston ja sanajärjestyksen lisäksi olla tuntemusta myös semantiikasta. Käännöskoneille oli kuitenkin suuri kysyntä, sillä kylmän sodan vuosina varsinkin amerikkalaiset halusivat kyetä kääntämään suuria määriä venäjänkielistä tiedustelutietoa mahdollisimman nopeasti. Tästä huolimatta ala alkoi väliaikaisesti hiipua.

1980-luvulla yksityisyritykset alkoivat entistä enemmän kiinnostua koneellisen kääntämisen kaupallisesta potentiaalista, joten tutkimustyö ei ollut enää riippuvaista eri maiden hallituksista. Konekääntämiseen luotiin kokonaan uusia lähestymistapoja, ja syntyi nykyinen luokittelu sääntöpohjaisiin, tilastollisiin ja esimerkkipohjaisiin käännösmetodeihin.

2000-vuosikymmenellä luonnollisen kielen koneellisessa kääntämisessä on saavutettu rohkaisevia tuloksia, joissa käännös on jo 90-prosenttisen tarkka. Toisaalta tätä parempiin tuloksiin yltäminen on osoittautunut haastavaksi, joten kehitystyö jatkuu edelleen. Entistä parempien ohjelmistojen kehittämisen kannalta on oleellista, että alan ongelmat ymmärretään aiempaa kokonaisvaltaisemmin.

Konekääntämisessä hyödynnetään nykyään monenlaisia matemaattisia malleja, koneoppimista ja kognitiivista ymmärtämistä. Näitä soveltamalla ohjelmia räätälöidään erilaisiin tarpeisiin sen sijaan, että pyrittäisiin luomaan yhtä yleiskääntäjää. Viime aikoina konekääntäminen on Internetin ansiosta noussut suuren yleisön tietoisuuteen, mikä on vaikuttanut positiivisesti käännösohjelmien kysyntään ja kehityspaineeseen.

Tässä tutkielmassa tehdään aluksi luvussa 2 lyhyt katsaus konekääntämisen historiaan ja hahmotellaan yleisesti konekääntämisen erityispiirteitä ja alalla kohdattuja vaikeuksia. Luvuissa 3, 4 ja 5 tarkastellaan kolmea yllä mainittua käännösmetodia yksityiskohtaisesti. Sääntöpohjaisten, tilastollisten ja esimerkkipohjaisten konekäännösohjelmien väliset erot ja vahvuudet selvennetään lukijalle. Tutkielma käsittelee vain kirjoitetun tekstin kääntämistä ja sen ulkopuolelle rajataan uniikit hybridikääntäjät, joihin on yhdistetty piirteitä eri käännöstavoista.

2 Konekääntäminen

2.1 Historiaa

Luonnollisen kielen koneellinen kääntäminen edellyttää kielten analysoimista matemaattisesti, numeerisin menetelmin. Tällaista käännöstapaa ja siihen perustuvaa universaalikieltä, eräänlaista interlinguaa (ks. luku 3.3) hahmoteltiin jo 1600- ja 1700-luvuilla, kauan ennen kuin tehtävään soveltuvia laitteita oli saatavilla. Tuon ajan filosofeista mainittakoon René Descartes (1596–1650), Gottfried Leibniz (1646–1716) ja Emanuel Swedenborg (1688–1772). Heidän näkemystensä mukaan on mahdollista muuttaa mikä tahansa virke numeroilla tai numeroita vastaavilla merkkijonoilla ilmaistavaan muotoon, kunhan luonnollisten kielten rakennetta ja ympäröivää maailmaa tarkastellaan riittävän huolellisesti. Eräistä tieteellisistä erimielisyyksistään huolimatta Descartes ja Leibniz olivat yksimielisiä siitä, että matemaattis-koneellisen kielen konstruoiminen vaati entistä kehittyneempää logiikkaa ja matematiikkaa. [1]

Varsinkin Leibniz näki paljon vaivaa luodakseen matemaattisen mallin, jonka avulla luonnollista kieltä voitaisiin käsitellä laskukoneella. Hän lähestyi ongelmaa useista eri näkökulmista, mutta joutui lopulta hylkäämään suurimman osan kehitelmistään, koska ne kasvoivat kehitystyön edetessä laajoiksi ja monimutkaisiksi. Jälkikäteen tarkastellen Leibnizin idea, jossa kirjaimet korvattiin ennalta sovituilla numeroilla laskukonetta varten, enteili ehkä pikemminkin modernia tietotekniikkaa ja binäärilukuja kuin hankkeen varsinaista tavoitetta eli käännöskonetta.

Swedenborg oli edeltäjiään optimistisempi ja ryhtyi kirjoittamaan sana sanalta termistöä aakkoslyhenteiden käsitteelliseen yhdistelemiseen perustuneelle kielelle, jonka mekaanisen käsittelyn oli tarkoitus olla mahdollisimman vaivatonta ja yksiselitteistä. Hän aloitti työnsä ihmisen anatomiaa koskevista sanoista – ja myös lopetti niihin. Projekti oli liian mittava yhden miehen toteutettavaksi. Swedenborg esitti myös idean hieman toisenlaisesta, pelkästään numerosarjoja sisältävästä kielestä, jossa kunkin numerosarjan ensimmäinen luku ilmaisee, mistä yleisluontoisesta asiasta on kyse, ja jokainen seuraava numero

tarkentaa kyseessä olevaa asiaa vähän kerrallaan. [2] Eräässä mielessä tämä idea oli jo sellainen, jota nykyisissäkin konekäännöstoteutuksissa voitaisiin periaatteessa hyödyntää.

Yksinomaan koneiden tehtäväksi suunnitellun kielenkääntämisen teollinen historia alkaa 1950-luvulta. Alalla vallitsi aluksi varsin optimistinen ilmapiiri. Niinpä esim. suomalainen, tekniikasta kiinnostunut kirjailija Voitto Viro arveli vuonna 1960, että jo 1980-luvulla ”elektronilaskijat” korvaavat ihmiskääntäjät. [3] 1960-luvun edetessä koneellisen kääntämisen vaikeuksien monimutkaisuus oivallettiin kuitenkin kunnolla. Seuraavien 10–15 vuoden kuluessa konekääntämissovellusten kehittäminen väheni selvästi. Niiden käyttö tosin jatkui, koska koneet kykenivät kääntämään tekstiä ihmistä nopeammin. [4]

Vuoden 1975 päättyessä ei esim. Yhdysvalloissa ollut käynnissä enää yhtäkään valtiorahoitteista, pelkästään konekääntämiseen keskittynyttä kehitysprojektia, mutta ohjelmistosuunnittelu jatkui osana muita hankkeita, etenkin tekoälytutkimusta. Hyvä esimerkki tuonaikaisesta kehitystasosta ja tekoälyn käyttämisestä on vuosina 1974–1975 Yalen yliopistossa kirjoitettu ohjelma SAM (Script Applier Mechanism), joka lehtiartikkeleita tarkastellessaan kykeni käsitteellisellä tasolla päättelemään, miten peräkkäisten lauseiden merkitykset liittyivät toisiinsa. Englanninkielisen artikkelin luettuaan ohjelma kykeni laatimaan siitä tiivistelmän englanniksi, espanjaksi ja venäjäksi. Ohjelmalle voitiin lisäksi esittää lehtijuttuihin liittyviä kysymyksiä, joihin se vastasi tietojensa mukaisesti. [5] Nykyäänkin monet konekääntämisen asiantuntijat ovat sitä mieltä, että käännöksen onnistuminen edellyttää ohjelmistolta kognitiivista ymmärtämistä.

Jo 1900-luvun puolivälissä saksalais-amerikkalainen Warren Weaver oli esitellyt hahmotelman tilastollisesta konekääntämisestä. [6] Kuitenkin vasta 1980-luvulla tietokoneet saavuttivat niin suuren tehokkuuden, että tilastollisuuteen perustuvia ohjelmia voitiin kehittää ja testata käytännössä. [7] Myös esimerkkipohjaisen kääntämisen ensiaskeleet otettiin 1980-luvulla, jolloin japanilainen tutkija Nagao Makoto laati siitä lyhyehkön julkaisun. [8] Voidaankin todeta, että viimeistään 1980-luvulla konekääntäminen eriytyi toisistaan selvästi eroaviksi metodeiksi, joita alettiin kehittää toisistaan riippumatta. Tutkimus sai myös lisää resursseja, kun entistä useammat yksityisyritykset alkoivat osoittaa kiinnostusta alaa kohtaan. Ohjelmistot olivat edelleen

puutteellisia, mutta odotuksetkin olivat maltillisempia ja tekniikkaan nähden realistisempia kuin 1950-luvulla.

1990-luvulta lähtien Internet on helpottanut laajojen, eri aloja käsittelevien sähköisten aineistokokoelmien eli oppimateriaalipankkien laatimista. Tästä on ollut hyötyä etenkin tilastolliselle konekääntämiselle, jota tutkitaan nykyään enemmän kuin mitään muuta koneellisen kääntämisen muotoa. Internet on edistänyt alaa myös siten, että se on tuonut nykyaikaisia konekäännössovelluksia suuren yleisön tietoisuuteen ja käyttöön.

2.2 Yleispiirteitä ja haasteita

Jokaisella konekääntämismenetelmällä on omat käyttötarkoituksensa ja vahvuusalueensa. Mitä rajatumpaan ja homogeenisempaan tekstijoukkoon käännösohjelmaa aiotaan käyttää, sitä erehtymättömämmäksi se voidaan yleensä hioa. Vastaavasti yleiskääntäjien kehittäminen luotettaviksi on haasteellista ja vaatii toisenlaisia ratkaisuja. Kaikilla konekääntäjillä on kuitenkin yksinkertaistetusti tarkastellen samat tehtävät: lukea käännettävä teksti, analysoida sitä ja tulostaa käännös. Metodien väliset yhtäläisyydet voidaan ymmärtää paremmin, jos ensin selvitetään luonnollisen kielen perusolemusta ja kääntämisessä kohdattavia ongelmia.

Luonnollinen kieli koostuu neljästä tasosta: fonologiasta, morfologiasta, syntaksista ja semantiikasta. Kirjoitettuja tekstejä käsiteltäessä fonologia voidaan käytännössä sivuuttaa, sillä se käsittelee äänteiden liittymistä toisiinsa. Morfologialla tarkoitetaan sanojen muotoja ja taivutuksia, syntaksi käsittelee sanajärjestyksiä sekä muita lauserakenteellisia piirteitä ja semantiikka tarkoittaa lauseiden merkityksiä. [9] Kielet eroavat toisistaan kaikilla näillä tasoilla, mutta nykyään semantiikka tuottaa käännösohjelmille eniten ongelmia. Yksinkertaisimmat toteutukset jättävät sen kokonaan vaille huomiota.

Vaikka luonnollisten kielten rakenne voidaan palauttaa edellä mainittuihin neljään yhteiseen tasoon, on eri kielten välillä mittavia eroja. Osa näistä eroista liittyy melko yksinkertaisiin ja helposti opittaviin asioihin, kuten sanajärjestykseen, mutta esim. sanojen

muodostamiseen liittyvät säännöt voivat vaikeuttaa kääntämistä huomattavasti. Jokainen luonnollinen kieli on jonkin ainutlaatuisen kulttuuriympäristön ja historiallisen kehityksen tuote, joten maailman eri puolilla puhuttavien kielten taustalla olevat ajattelutavatkin ovat osittain erilaisia. Käänteisesti tästä seuraa, että kääntäminen on usein melko suoraviivaista, jos kieltä käännetään jollekin sen läheisistä sukulaiskielistä, kuten suomesta viroon.

Tietokoneilla ei ole tietoisuutta eikä – pääsääntöisesti – myöskään tietoja ympäröivästä maailmasta, joten ne eivät voi ymmärtää tekstejä samalla tavalla kuin ihmiskääntäjä. Jos henkilöä, joka osaa vain muutamaa eurooppalaista kieltä, käsketään kääntämään jokin kirjoitus arabiasta japaniksi, hän on lähtökohtaisesti samanlaisessa tilanteessa kuin tietokone. Sekä ihmiselle että tietokoneelle voidaan antaa apuvälineiksi sanakirjoja ja jostakin tietystä aihepiiristä kertovia esimerkkitekstejä. Niistä on epäilemättä hyötyä varsinkin lyhyissä käännöstöissä, mutta yksinään ne eivät riitä tekemään sen paremmin ihmisestä kuin tietokoneohjelmastakaan pätevää kääntäjää.

Pelkkä sanaston ja kieliopin tunteminen ei riitä asiayhteyksien ymmärtämiseen ja monitulkintaisuuden karsimiseen. Mikäli lähdetekstissä on homonyymejä tai sanoja, joille kohdekielestä löytyy useita eri käännösvaihtoehtoja, on asiayhteyteen liittyvä päättelykyky tarpeen. Joissakin kielissä, kuten suomessa, monitulkintaisuusongelma liittyy myös sanojen taivutusmuotoihin ja yhdyssanoihin. Miten käännösohjelma voi ilman kontekstitietoja päätellä, onko ”häntä” eläimen osa vai hän-pronominin taivutettu muoto? Entä viittaako ”tietokoneistuminen” tietokoneiden kasvavaan käyttöön vai tietokoneen ääressä istumiseen?

Mainituista apuvälineistä ei ole hyötyä myöskään kielikohtaisten sanontojen ja idiomien kohdalla. Idiomit, kuten ”pitkin hampain” ovat muutaman sanan mittaisia metaforia, joiden merkitys ei vastaa kirjaimellista sisältöä. Yleisistä sanonnoista puolestaan on eri kielissä omat versionsa. Tällaiset sanonnat pitää tunnistaa muun tekstin seasta sujuvan käännöstuloksen saavuttamiseksi. Mitä useampaa eri kieltä ja kirjoitusteemaa käännöskone käsittelee, sitä enemmän asiantuntemusta sillä pitää olla.

Inhimillisten erehdysten aikaansaamat kirjoitusvirheet muodostavat oman ongelmansa. Mikäli väärin kirjoitettu sana muistuttaa kirjoitusasultaan tarkalleen yhtä olemassa olevaa sanaa, ihminen kykenee helposti tekemään päätöksen kyseisen sanan käyttämisestä, vaikka kirjoitusvirhe olisikin vieraskielisessä tekstissä. Ihminen voi tehdä ratkaisun sangen vaivattomasti myös silloin, jos virheellinen muoto muistuttaa kahta eri sanaa, mutta toinen on huomattavasti yleisemmin käytetty kuin toinen. Edes asiayhteyttä ei välttämättä tällöin tarvitse hahmottaa. Vanhanaikaisten sääntöpohjaisten käännösohjelmien tekijät ovat halutessaan voineet sisällyttää ohjelmiinsa valmiin listan tavallisimmista kirjoitusvirheistä ja niiden korjauksista, mutta täysin kattavan valmisluettelon laatiminen ei liene teoriassakaan mahdollista.

3 Sääntöpohjainen konekääntäminen

Sääntöpohjainen konekääntäminen oli aikoinaan konekääntämistutkimuksen ensimmäinen ja välttämätön kehitysvaihe. Vaikka siirtopohjaista ja interlinguaan perustuvaa kääntämistä on tutkittu 2000-luvullakin, ovat sääntöpohjaiset metodit pääasiassa menneisyyteen kuuluvia innovaatioita. Tietämys näistä menetelmistä kuitenkin helpottaa monimutkaisempien, tilastollisuutta ja kaksikielisiä esimerkkitekstejä hyödyntävien ohjelmien toiminnan ymmärtämistä.

3.1 Sanakirjapohjainen konekääntäminen

Sanakirjapohjainen konekääntäminen on sääntöpohjaisen konekääntämisen muodoista yksinkertaisin ja perinteisin. Eräissä lähteissä sitä ei epämatemaattisuutensa vuoksi edes lasketa sääntöpohjaiseksi menetelmäksi, vaan se mainitaan neljäntenä kääntämisen perusmetodina sääntöpohjaisten, tilastollisten ja esimerkkipohjaisten menetelmien ohella. [10]

Luvussa 2 verrattiin käännösohjelmaa ihmiseen, joka ei ymmärrä käännöstyössä käsiteltäviä kieliä, mutta jolle voidaan tarjota eräitä apuvälineitä. Ainoana työkaluna on nyt sanakirja. Käännösohjelma lukee lähdekielistä tekstiä sana sanalta ja etsii kohdekielisen käännöksen sähköisestä sanaparitietokannastaan. Vastinparien lista voi olla mutatoitumaton tai sitä voidaan tarpeen mukaan päivittää. Internetistä löytyy lukuisia ilmaisia sovelluksia, joihin käyttäjät voivat lisätä uusia käännöspareja. Yksi sanakirjapohjaisten käännösohjelmien hyvistä puolista onkin toteuttajien kannalta se, että sanastojen täydentäminen on teknisesti yksinkertaista ja se voidaan periaatteessa ulkoistaa kokonaan käyttäjille.

Syntaksi ja semantiikka jäävät sanakirjapohjaisessa kääntämisessä joko vähälle huomiolle tai kokonaan vaille huomiota. Ohjelmat eivät välttämättä pyrikään valitsemaan sopivinta käännöstä kaikista tuntemistaan synonyymeistä, vaan käsittelevät vain yhtä sanaa kerrallaan asiayhteydestä välittämättä. Tällainen menetelmä ei käytännössä sovellu pitkien,

yleiskielisten tekstien kääntämiseen. Miksi näin alkeelliset konekääntämissovellukset eivät sitten ole poistuneet käytöstä?

Mitä yksinkertaisempaa tekstiä halutaan kääntää, sitä yksinkertaisempi käännösohjelma riittää työn tekemiseen. Mikäli tiedetään etukäteen, että kaikki käännettäviksi haluttavat kirjoitukset ovat luetteloita tai jotakin alaa koskevia standardimuotoisia dokumentteja, voidaan sanakirjapohjainen käännösohjelma – tai tarkemmin sanoen ohjelman käyttämä sanasto – optimoida suoriutumaan juuri niiden kääntämisestä. Tarkalleen yhtä tieteen tai elinkeinoelämän osa-aluetta varten spesifioitu käännösohjelmaversio kykenee käyttämään harvinaisiakin ammattitermejä sujuvasti: koska ohjelma käsittelee vain yhden aihepiirin tekstejä, se ei oikein käytettynä voi sekaantua asiayhteydestä. Luetteloiden kohdalla on kääntämistä helpottavana tekijänä myös se, ettei sanojen taivuttamisesta tarvitse välittää. Tällaisiin tehtäviin ei tarvita nykyajan monimutkaisia tilasto- ja esimerkkipohjaisia ohjelmia.

3.2 Siirtopohjainen konekääntäminen

Sanakirjapohjaisen käännösprosessin aikana tietokoneohjelma ei yleensä osoita minkääntasoista asiayhteyden ymmärtämistä. Siirtopohjaiset implementaatiot analysoivat tekstin sisältöä tarkemmin ja ovat sovellettavissa jonkin verran monimuotoisempiin teksteihin. Molempia metodeja käytetään erityisesti tapauksissa, joissa esimerkkipohjaiset ja tilastolliset menetelmät olisivat tehottomia, koska koneille tarjottavaa oppimateriaalia on vähän. Tällaiseen tilanteeseen voidaan joutua, jos toinen tai molemmat käsiteltävistä kielistä ovat harvinaisia.

Siirtopohjainen kääntäminen etenee vaiheittain. Ensin lähdekielistä tekstiä analysoidaan morfologisesti ja sanastollisesti, eli sanojen sanaluokat rekisteröidään ja sanoja ryhmitellään yleisluontoisiin kategorioihin. Jos tekstissä esiintyy esim. sanat ”koira” ja ”varis”, ohjelma voi jäsentää ne ”eläimet”-kategoriaan ja näin saada selville, että käännettävä tekstiosuus käsittelee eläimiä. Tällä tavalla ohjelma voi tunnistaa asiayhteyksiä ja valita käännökseen parhaiten sopivia kohdekielisiä termejä. Analyysin tuottama välitulos

riippuu kuitenkin ainoastaan lähdetekstistä, eikä siitä voida päätellä lopullisen käännöksen onnistumista.

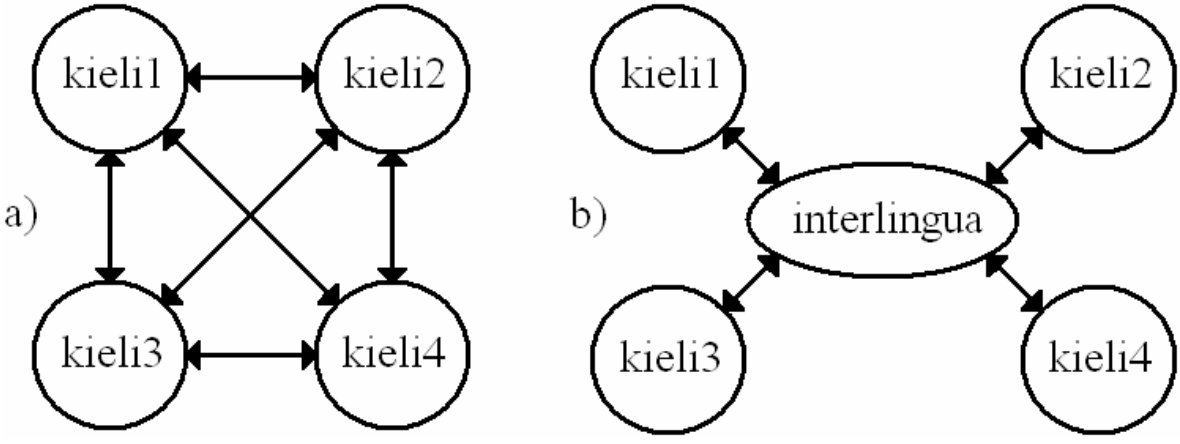
Seuraava vaihe on analyysin tuottaman välituloksen siirtäminen lähdekielestä kohdekieleen. Tämä muistuttaa sanakirjapohjaista kääntämistä, mutta hyödyntää sanastojen lisäksi kielioppisääntöjä ja korpuksia. Korpukset ovat laajoja kielellisiä tietovarastoja, joiden esimerkkilauseita tarkastelemalla voidaan saada varmistus kieleen liittyville hypoteeseille, kuten jonkin tietyn ilmaisun oikeaoppiselle käytölle. [11] Korpuksia hyödynnetään muissakin käännösmetodeissa kuin siirtopohjaisessa konekääntämisessä ja ne sisältävät avainsanoja käytettävyyden parantamiseksi. [12]

Kun teksti on sanastollisella tasolla siirretty kohdekieliseen muotoon, siinä saattaa olla vielä huonosti valittuja ilmaisuja ja suoranaisia virheitäkin, kuten ylimääräisiä tai puuttuvia päätteitä ja sanajärjestysvirheitä. Käännös saadaan sujuvammaksi formalisointioperaatioilla eli jälkikorjauksilla. Formalisointia kutsutaan joskus morfologiseksi generoinniksi. Ohjelmiston varhaisessa kehitysvaiheessa formalisointikykyä voidaan parantaa esim. ihmiskääntäjien ja funktiolaskennan yhteistoiminnalla. Ominaisuus on ohjelman laadun kannalta huomattavan tärkeä, joten sitä tarkastellaan seuraavassa yksityiskohtaisemmin.

Formalisointia varten kohdekielisestä käännöksestä, jossa on m virkettä, muodostetaan tietokoneen muistissa vektorit $KK_1 \ldots KK_m$. Vektorit ovat muotoa $KK_m = (S_1, \ldots S_i, \ldots S_n)$, missä S_1 on virkkeen ensimmäinen ja S_n sen viimeinen sana. Oletetaan, että käännöstä tarkastava ihmiskääntäjä havaitsee tekstissä virheellisen sanan ja syöttää koneelle korjatun virkkeen $KK_m' = (S_1, \ldots S_i', \ldots S_a, \ldots S_n)$. Käyttäjä myös osoittaa koneelle sanan S_a, joka oli virheen aiheuttaja. Merkitään, että virheen sisältänyt sana oli S_i ja oikea eli korvaava sana S_i'. S_a ilmentää jotakin attribuuttia, joka vaikuttaa käännökseen. Jos esim. englanninkielinen virke ”I ate two fish” (suom. ”Minä söin kaksi kalaa”) olisi ennen suomennoksen formalisointia muodossa ”Minä söin kaksi kala”, merkittäisiin S_a = ”kaksi”, S_i = ”kala” ja S_i'= ”kalaa”. Virheellinen attribuutti tässä esimerkissä oli sanan luku. Englannin kielessä sanan ”fish” yksikkö- ja monikkomuoto on sama, joten sanan S_a paikantaminen oli välttämätöntä käännöksen oikeellisuudelle.

Kun koneelle on annettu oikea käännös, se vertailee sanojen S_i ja S_i' ominaisuuksia, kuten sukua, lukua ja sijaa muodostamalla niiden eroista funktion $\delta(S_i, S_i')$. Mikäli funktio kuvaa tarkalleen yhtä eroavaisuutta, on virheen löytäminen yksinkertaista. Edellisen kappaleen esimerkissä funktion arvojoukko olisi ollut puuttuva monikon akkusatiivimuodon a-kirjain, joten merkintätavasta riippuen voidaan kirjoittaa δ(”kala”, ”kalaa”) = {–a} tai δ(”kala”, ”kalaa”) = {monikko, akkusatiivi}. Funktio tallentuu koneen muistiin, joten havaitessaan tekstissä seuraavan kerran sanan S_a ohjelma osaa välttää saman käännösvirheen. Ohjelmat suunnitellaan usein noudattamaan Occhamin partaveitsiperiaatetta eli jos samasta virkkeestä löytyy kaksi tai useampia virheitä, niiden oletetaan johtuvan samasta syystä. [13]

Siirtopohjaisessa konekääntämisessä sanastot ja kielioppitietokannat muodostavat kielipareja. Ne ovat toisistaan riippumattomia, samoin kuin sanakirjapohjaisten ohjelmien sanastot. Erillisyyden vuoksi niitä tarvitaan ohjelmistossa n * (n – 1) kappaletta, missä n on käytettävien kielien määrä. [14] Kuvassa 2.1a) mainitaan esimerkkinä neljä kieltä, joiden kääntämiseksi tarvitaan 4 * (4 – 1) eli 12 kieliparia. Jokainen nuolenkärki merkitsee kuvassa yhtä paria. On huomionarvoista, että esim. kieli1–kieli2-kielipari on eri asia kuin samoja kieliä käsittelevä kieli2–kieli1-pari. Mikäli ohjelmaa halutaan laajentaa käsittelemään uusia kieliä, kasvaa päivitystyön ja tarvittavan muistitilan määrä neliöllisesti. Kasvu voidaan pitää lineaarisena vain päättämällä, ettei mistään kielestä tarvita käännösmahdollisuutta kaikkiin muihin kieliin. Tällaista ohjelmistototeutusta voidaan kuitenkin – hyvästä syystä – pitää keskeneräisenä.



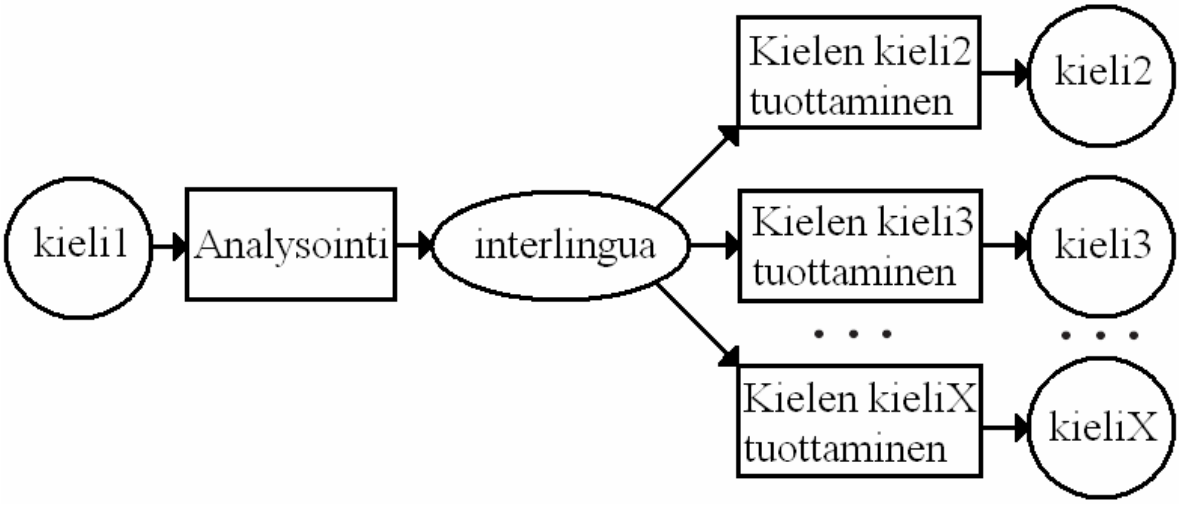
Kuva 2.1: a) Siirtopohjaisen ja b) interlinguaan perustuvan konekääntämisen käännöskaaviot.

3.3 Interlinguaan perustuva konekääntäminen

Siirtopohjaiset menetelmät johtavat ohjelmistojen resurssitarpeen eksponentiaaliseen kasvuun, koska käännösprosessit ovat kieliriippuvaisia. Ongelma on sitä pahempi, mitä useampaa kieltä ohjelman halutaan ymmärtävän. Ratkaisuksi on tarjottu apukieltä eli interlinguaa, jolle lähdekieliset tekstit käännetään ennen kohdekielisen version generointia. Kieliparien määräksi riittää 2 * n, kuten kuvasta 2.1b) käy ilmi. Kustakin kielestä tarvitaan kuitenkin samat sanasto- ja kielioppitiedot niin siirto- kuin interlinguakäännösohjelmistoissakin.

Ellei muuta mainita, tässä tutkielmassa käytetään termiä ”interlingua” yleisnimenä, joka viittaa mihin tahansa tietokoneiden käyttämään apukieleen. Interlingua tarkoittaa myös vuosina 1924–1951 kehitettyä, ihmisten puhuttavaksi tarkoitettua keinotekoista kieltä, joka perustuu pääasiassa latinaan, italiaan, espanjaan, portugaliin, ranskaan ja englantiin. [15]

Apukielenä interlingua voidaan määritellä universaalikieleksi, jolle jonkin kielijoukon kaikki lauserakenteet voidaan kääntää merkitystä muuttamatta. Interlinguan kieliriippumattomuus saadaan aikaan jakamalla ohjelmistokoodin toiminnalliset osat itsenäisiin moduuleihin, jotka joko analysoivat tai tuottavat tekstiä. Nämä toiminnalliset moduulit on kuvattu nelikulmioina kuvassa 2.2, josta käy ilmi myös käännösprosessin vaiheittainen eteneminen.



Kuva 2.2: Interlinguaa hyödyntävä käännösprosessi.

Käytännössä käännösmetodin suurimmaksi ongelmaksi on osoittautunut riittävän hyvän apukielen löytäminen tai keinotekoinen muodostaminen. Yleisesti käytettäväksi interlinguaksi on ehdotettu mm. englantia, mutta tulokset eivät ole olleet tyydyttäviä. Vuonna 2002 Microsoftin tutkijaryhmä selvitti kieliparin englanti-japani kääntämistä ja totesi, että olisi ”naiivia” pyrkiä esittämään englannin- ja japaninkielisten tekstien sisällöt täsmälleen samassa loogisessa muodossa. Kielissä havaittiin käytettävän ilmaisuja, jotka eivät – nykyisillä menetelmillä – ole käännettävissä yksiselitteisesti. Selvitys liittyi ensisijaisesti esimerkkipohjaiseen konekääntämiseen (ks. luku 4), mutta tulos on yleistettävissä myös interlinguaa koskevaksi ongelmaksi. [16]

Tietyissä yksittäistapauksissa luonnollista kieltä voidaan käyttää apukielenä. Esimerkiksi Leedsin yliopiston tutkijat käänsivät koneellisesti vuonna 2007 ukrainankielisiä tekstejä englanniksi käyttäen venäjää siltakielenä. [17] Tämä oli mahdollista kahdesta syystä. Ensinnäkin lähde- ja kohdekielinä ei ollut mitään muita kieliä kuin ukraina ja englanti. Toiseksi venäjän kielellinen ”etäisyys” näihin kieliin on pienempi kuin ukrainan ja englannin etäisyys toisiinsa. Ohjelman laajennettavuus oli hyvin rajallinen.

Nykyään interlinguatutkimuksessa ei yleensä pyritä käyttämään luonnollisia kieliä apukielinä. Sen sijaan oletetaan, että kartoittamalla erilaisten luonnollisten kielten piirteitä voidaan kehittää keinotekoisia kieliä, jotka ovat ilmaisukyvyltään niin monipuolisia, että niitä voidaan käyttää konekääntämisen apukielinä. Eräät tutkijat ovat tämän lisäksi esittäneet, että apukieli ei saa olla liian laaja, sillä tämä voisi johtaa monitulkintaisuuteen. Interlinguassa tulisi siis olla kaikki tarvittavat piirteet, mutta ei mitään ylimääräistä.

Kuten edellä esiteltyihin menetelmiin, myös interlinguaan perustuvan konekääntämisohjelmiston toteuttaminen on sitä haasteellisempaa, mitä monimutkaisempia ja laaja-alaisempia tekstejä halutaan kääntää. Usean eri kielen piirteitä yhdistävän apukielen tietokanta kasvaa erittäin laajaksi, mikäli ohjelmaa ei aiota soveltaa vain tarkasti rajatun alan teksteihin. Lisäksi tutkimuksia rahoittavilla tahoilla on yleensä kysyntää vain tietyntyyppisten tekstien käännöksille, joten monialaisen, kattavan interlinguan kehittämisessä ei ole toistaiseksi tapahtunut läpimurtoa. Apukieliä kehitetäänkin varta

vasten joillekin aloille ja jollekin suppealle kielijoukolle sopiviksi. Tällöin kehitystyö etenee alhaalta ylöspäin eli sanastosta kielioppiin ja semantiikkaan. [18]

Apukielten erilaisista kohdealueista ja kehityshistorioista seuraa, että niiden notaatiojärjestelmätkin poikkeavat jonkin verran toisistaan. Tämän tutkielman puitteissa ei ole mahdollista eikä järkevääkään esitellä kaikkia yksitellen. Seuraavassa esimerkissä selvitetään suomen kielen kääntämistä tunnetulle interlingualle nimeltä UNL (Universal Networking Language). Tämän kielen kehitystyö aloitettiin vuonna 1996 Yhdistyneiden Kansakuntien yliopistossa Tokiossa. [19]

UNL-interlinguassa, kuten muissakin tietokoneiden nykyään hyödyntämissä apukielissä, luonnollisten kielten sanoista muodostetaan universaalisanoja, jotka ilmaisevat pikemminkin konsepteja kuin tietynkielisiä sanoja. Esimerkiksi universaalisana `syödä(kl>tehdä, obj>ruoka)` merkitsee ”ruoan laittamista suuhun”. Notaatiossa kl merkitsee johonkin joukkoon kuulumista; syöminen kuuluu tekemisen joukkoon. Lyhenne obj merkitsee tekemisen kohdetta eli objektia. Vaikka tässä UNL-muodossa on vain suomenkielisiä sanoja ja lyhenteitä, kyseessä on kieliriippumaton universaalisana.

Kokonaisia lauseita käännettäessä universaalisanoihin liitetään lauseenjäsentiedot sekä attribuutteja, jotka ilmaisevat mm. aika- ja sijamuodon. Olkoon esimerkkilauseena ”Minä kirjoitan TkK-tutkielmaa kotona.” Suomalaistetulla UNL-kielellä ilmaistuna lause on

```
srt (kirjoittaa(kl>tehdä).@preesens.@tekeminen:01,
      minä(kl>ihminen).luku:Y1)
obj (kirjoittaa(kl>tehdä).@preesens.@tekeminen:04,
       TkK-tutkielma(kl>teksti).@luku:Y3
pkk (kirjoittaa(kl>tehdä).@preesens.@tekeminen:05,
      koti(kl>rakennus).@luku:Y3
```

Esimerkissä srt tarkoittaa suorittajaa (engl. agent) ja pkk paikka-attribuuttia. 01, 04, 05, Y1 ja Y3 ilmaisevat sijapäätteitä ja persoonamuotoja; esim. 01 tarkoittaa nominatiivia.

4 Esimerkkipohjainen konekääntäminen

Edellä käsitellyt sääntöpohjaiset käännöstavat ovat käyttökelpoisia tilanteissa, joissa tekstiresurssit ovat liian niukat laajojen korpusten kokoamiseen. Sanasto ja kielioppi joudutaan niitä hyödynnettäessä tarjoamaan ohjelmistolle pala palalta. Jos taas sopivia lähde- ja kohdekielisiä tekstejä on runsaasti saatavilla, ohjelma voi käyttää niitä ajoaikaisena itseopiskelumateriaalina, jolloin yksitellen syötettyjä sanastotietoja tarvitaan vähemmän. Tällaista tapauskohtaisen koneoppimisen implementoimista kutsutaan esimerkkipohjaiseksi konekääntämiseksi.

4.1 Kaksikielinen korpus

Korpukset esiteltiin luvussa 3.2 kääntämisen apuvälineinä, joita siirtopohjaiset ohjelmat käyttävät ilmaisujen oikeellisuuden tarkistamiseen ja joiden rinnalla hyödynnetään merkittävästi myös sanastoa ja kielioppikokoelmaa. Esimerkkipohjaisissa käännösohjelmissa korpukset toimivat käännösprosessin tärkeimpänä perustana, joten niiden laajuuden merkitys korostuu. Tavallista sähköistä sanakirjaa käytetään lähinnä käännösten tarkistamiseen.

Korpukset sisältävät kokonaisia lauseita ja kirjoituksia niiltä aihealueilta, joihin käännösohjelmaa sovelletaan. Niitä voidaankin kutsua esimerkkikokoelmiksi, sillä korpuksissa esiintyvät lauserakenteet ovat samoja, joista käännettävä tekstikin koostuu. Tutkija Ralf Brown tiivisti korpusten ja samalla koko esimerkkipohjaisen konekääntämisen perusajatuksen vuosina 1997–2001 toteutetun tutkimusprojektin yhteenvedossa: ”Mikäli jo aiemmin käännetty lause havaitaan uudelleen, sitä vastaa todennäköisesti sama käännös kuin edelliselläkin kerralla.” [20]

Jotta korpuksia voitaisiin käyttää varsinaiseen kääntämiseen eikä vain käännösten tarkistamiseen, niiden on oltava kaksikielisiä. Kaiken tekstisisällön on sisällyttävä niihin sekä lähde- että kohdekielisenä. Myös tässä katsannossa on mielekästä kutsua korpuksia esimerkkikokoelmiksi; ne koostuvat valmiista mallikäännöksistä. Ohjelma suorittaa

käännöksen vertaamalla käännettävää kirjoitusta korpukseen. Sillä, kumpi korpuksen kielistä on lähde- tai kohdekieli, ei ole korpuksen rakenteen kannalta merkitystä.

Kaksikielisten korpusten ansiosta ohjelmien käyttöalueiden laajentaminen on yksinkertaista; korpuksen täydentäminen sopivilla kirjoituksilla riittää. Esimerkiksi biokemia-alan käännöstöissä on tarpeen, että ohjelma on erikoistunut sekä biologiaan että kemiaan. Lisäksi samaa ohjelmakoodia voidaan soveltaa eri kieliin, kunhan korpus korvataan jollakin toisella. Lyhyesti sanottuna kaksikieliset korpukset tekevät esimerkkipohjaisesta koodista monikäyttöistä. Toisaalta ohjelmat vaativat pitkiä korpuksia ja verrattain paljon laskenta-aikaa. [21]

Vaikka käytettävä korpus olisikin laaja, on epätodennäköistä, että siitä löytyisi täsmälleen samoja lauseita kuin käännettävästä tekstistä. Käännösohjelma parseroikin lauseita lyhyempiin osiin ja etsii lähdekielisestä tekstistä ja korpuksesta mitä tahansa yhtäläisyyksiä. Ne ovat pienimmillään yksittäisiä sanoja ja suurimmillaan kokonaisia virkkeitä. Kun joukko toisiaan vastaavia osia on löydetty, ne pitää enää yhdistää keskenään kokonaisiksi virkkeiksi (ellei käännettävä virke löytynyt korpuksesta sellaisenaan). Ohjelman tulee huomioida kohdekielen sanajärjestys, mutta muuten yhdistämisvaihe on yksinkertainen.

Esimerkiksi englanniksi käännettävä teksti voi sisältää lauseen ”Minä ostin hyvän kirjan eilen.” Oletetaan, että korpus sisältää seuraavat virkeparit:
1) **Minä ostin** uuden auton. / **I bought** a new car.
2) Mikä tuon kirjan hinta on? What is the price of that book?
3) Hän lainasi **hyvän kirjan** kirjastosta. / He borrowed **a good book** from the library.
4) Söimme herkullisen aterian **eilen**. / We had a delicious meal **yesterday**.
Yksikään mallivirkkeistä ei vastaa käännettävää lausetta, mutta esimerkkejä on silti riittävästi oikean käännöksen konstruoimiseen. Vertaamalla korpuksen lihavoituja kohtia saadaan oikea käännös: ”I bought a good book yesterday.”

Esimerkin lausetta 2 ei hyödynnetty käännöksessä, vaikka sana ”kirjan” esiintyy ensimmäisen kerran juuri siinä. Käydessään esimerkkilauseita ajoaikana läpi ohjelma valitsee lopulliseen käännökseen korpuksesta mahdollisimman pitkiä osia, mikä optimoi

lopputuloksen oikeellisuuden. Sanapari ”hyvän kirjan” sisältää enemmän lähdekielisen tekstin elementtejä kuin pelkkä sana ”kirjan”, joten lause 2 ei ole tarpeellinen tässä esimerkissä. Se tosin vaikuttaa käyttökelpoiselta siinä vaiheessa, kun ohjelma on lukenut vasta esimerkkilauseet 1 ja 2. Ohjelmaimplementaatiosta riippuen on tietysti mahdollista, ettei korpusta käydä läpi samassa järjestyksessä kuin tässä esimerkissä.

4.2 Lauseidenvälinen etäisyys

Kun ohjelma havaitsee korpuksessa useita osittain käyttökelpoisia lauseita, miten se käytännössä kykenee vertailemaan niitä keskenään? Nykyaikainen menetelmä tämän ongelman ratkaisemiseen on lauseidenvälisen etäisyyden laskeminen. Prosessissa hyödynnetään erityistä asiasanastoa, joka on tarpeen määritellä ensin.

Asiasanasto on tiettyyn aihepiiriin liittyvien termien lista, joka on ryhmitelty vastaavuussuhteiden ja hierarkkisten suhteiden mukaan. Asiasanastoissa merkitään, mihin aiheeseen termit liittyvät, mihin laajempaan ryhmään ne kuuluvat ja mitä aliryhmiä niihin mahdollisesti sisältyy. Asiasanastoihin voidaan kirjata myös termien synonyymit. Esimerkiksi Yleinen suomalainen asiasanasto YSA ryhmittelee sanan ”aaltoenergia” seuraavasti:
Kuuluu asiasanastoon: YSA - Yleinen suomalainen asiasanasto
Kuuluu ryhmään: 19 Energia. Polttoaineet
Laajempi termi: energia
Rinnakkaistermi: ekoenergia [22]

Lauseidenväliset etäisyydet ovat ei-negatiivisia reaalilukuja, jotka ilmaisevat kahden samaa kieltä olevan lauseen eroavaisuuksien määrän. Määrä suhteutetaan lauseiden pituuteen, koska mitä pidempiä sanajonoja vertaillaan, sitä merkityksettömämpi yksittäinen sana on. Identtisille lauseille etäisyys on nolla ja kaikissa muissa tapauksissa nollaa suurempi.

Alan keskeisiin kehittäjiin kuuluva Kōben yliopiston professori Eiichiro Sumita esitteli vuonna 2001 kaavan etäisyyksien laskemiseen. Laskentatapa perustuu Sumitan ja Hitoshi Iidan jo kymmenen vuotta aiemmin tekemään tutkimustyöhön. Etäisyyden laskemiseksi

tarvitaan sekä korpusta että asiasanastoa. Kun käännösohjelma käy korpuksen esimerkkejä läpi ja vertaa käännettävää lausetta niihin, se käytännössä vertaa lauseen pituutta korpuksen esimerkkien pituuteen samalla kun lauseen yksittäisiä sanoja verrataan asiasanastoon termi kerrallaan.

Kaavan mukaan käännettävän lauseen ja korpukseen kuuluvan mallilauseen välinen etäisyys LE on

$$LE = \frac{L + D + 2 * \sum SE}{M_s + M_e}$$

Luvut L ja D kertovat ”lisättyjen” sekä ”deletoitujen” sanojen määrän eli sen, kuinka paljon pidempi tai lyhyempi korpuksen mallilause on syötelauseeseen nähden. Sillä, kumpaa lausetta pidetään lähtökohtana ja kumpaa vertailukohteena, ei ole merkitystä, sillä L:n ja D:n summa on kummassakin tapauksessa sama. M_s ja M_e ovat syötelauseen ja korpuksen esimerkkilauseen sanamäärät.

Kaavan muuttuja SE on semanttinen etäisyys, jonka määrittämiseen tarvitaan asiasanastoa. Semanttinen etäisyys mitataan kullekin termille erikseen ja tulokset summataan, jolloin saadaan koko sanajonon semanttinen etäisyys. Se on aina lukuvälillä [0, 1]. Kaavana on yksinkertainen jakolasku

$$SE = \frac{K}{N}$$

Muuttuja K kertoo vertailtavien sanojen matalimman yhteisen abstraktiotason asiasanastossa, kun taas N on abstraktiotasojen kokonaismäärä. Käännösohjelmien asiasanastot on nimittäin järjestetty hierarkkisesti termien tarkkuuden mukaan. Esimerkiksi sana ”kone” on hierarkiassa korkeammalla abstraktiotasolla kuin ”moottori”, joka puolestaan on korkeammalla kuin ”dieselmoottori”. Jos vertailtavat sanat ovat ”dieselmoottori” ja ”polttomoottori”, niiden matalin yhteinen taso – eli tarkin kumpaakin käsitettä koskeva määrittely – on ”moottori”. [23]

Verrataan lauseidenvälisen etäisyyden avulla yhtä käännettävää lausetta kahteen korpuksen lauseeseen:
SL) ”Minä kirjoitan TkK-tutkielmaa kotona.” (syötelause)

KL1) ”Minä murehdin TkK-tutkielmaa kotona joka päivä.” (korpuksen lause)
KL2) ”Minä kirjoitan TkK-tutkielmaa.” (toinen korpuksen lause)
Molemmat korpuksen lauseista ovat selvästi osittain käyttökelpoisia käännösprosessissa, mutta kumpi muistuttaa alkuperäistä syötelausetta enemmän?

Lauseessa KL1 on kahden sanan mittainen aikamääre, joka ei esiinny käännettävässä lauseessa SL missään muodossa. Näin ollen Sumitan kaavan muuttuja L saa arvon 2. D:n arvo on 0, koska SL ei sisällä yhtäkään lauseenjäsentä, joka puuttuisi korpuksen mallilauseesta KL1. Lauseen verbi on kuitenkin erilainen, joten sille määritellään semanttinen etäisyys. ”Kirjoittaminen” ja ”murehtiminen” ovat aivan erilaisia tekoja, joten niiden keskinäinen etäisyys on 1. Lauseiden SL ja KL1 sanamäärät ovat 4 ja 6. Sijoittamalla saadaan etäisyydeksi $(2 + 0 + 2 * 1) / (4 + 6) = 0{,}4$.

Lauseessa KL2 ei ole mitään ylimääräistä, mutta siitä puuttuu yksisanainen paikanmääre eli D:n arvo on 1. Lauseen pituus on 3 sanaa. Etäisyys syötelauseesta on vastaavasti $(0 + 1 + 2 * 0) / (4 + 3) \approx 0{,}14$. Etäisyys on nyt huomattavasti pienempi, joten lause KL2 muistuttaa syötelausetta enemmän kuin KL1. Niinpä mallilauseen KL2 käännös – joka sisältyy valmiina kaksikieliseen korpukseen – on hyvin käyttökelpoinen, kun konstruoidaan lauseen SL vieraskielistä käännöstä.

Oikeaa käännöstä varten korpuksesta tarvitaan KL2:n lisäksi vain jokin lause, johon sisältyy sana ”kotona”. Itse asiassa sitäkään ei tarvita, jos ohjelmaan on asiasanaston ja kaksikielisen korpuksen ohella liitetty perinteinen sähköinen sanakirja, joka tuntee tarvittavan sanan. Esimerkkipohjaisessa konekääntämisessä vanhanaikaiset sanastot ovat käyttökelpoisia juuri silloin, kun niitä käytetään yhden yksittäisen sanan kääntämiseen.

Etäisyyksien laskeminen ratkaisee vähimmäislaatuongelman. Esimerkkipohjaiset käännösohjelmat valitsevat tulostettavaksi käännökseksi ”vähiten huonon” vaihtoehdon, joka voi pahimmillaan olla hyvinkin harhaanjohtava tai kokonaan virheellinen. Lienee järkevää rakentaa ohjelma siten, että tietyn laaturajan alittavia käännöksiä ei palauteta lainkaan, vaan käyttäjälle ilmoitetaan, ettei sopivaa käännöstä löydy. Nyt vähimmäisraja voidaan helposti määrätä joksikin lauseidenväliseksi etäisyydeksi.

5 Tilastollinen konekääntäminen

Tilastollinen konekääntäminen perustuu asiasanastoihin, kielen piirteiden hierarkkisuuteen ja ennen kaikkea todennäköisyyslaskentaan. Warren Weaver hahmotteli menetelmää jo vuonna 1949, mutta vasta 1980-luvulla oli saatavilla riittävän tehokkaita tietokoneita ohjelmistokehittelyä varten. Tilastollinen metodi pyrkii jäljittelemään ihmiskääntäjän toimintaa korkealla abstraktiotasolla. Onkin mahdollista, että tilastolliset käännösohjelmistot kehittyvät entistä tarkemmiksi, kunhan ihmisaivojen toimintaa ymmärretään paremmin.

Tilastollinen konekääntäminen soveltuu tilanteisiin, joissa käännettävää tekstiä muistuttavaa vertailumateriaalia on runsaasti saatavilla. Tällainen materiaali toimii korpuksena (ks. luku 3.2), jota ohjelmat lukevat laskiessaan todennäköisyyksiä eri käännösten oikeellisuudelle. Korpusten ansiosta ohjelmistoille ei tarvitse syöttää manuaalisesti kaikkia kieliin liittyviä sääntöjä yksitellen. Esimerkkipohjaisten käännösohjelmien tavoin myös tilastolliset ohjelmistot lukevat kaksikielisiä korpuksia.

5.1 Kielimalli ja käännöksen todennäköisyys

Tilastollisen käännösmenetelmän lähtökohtana on kaksi tärkeää havaintoa. Ensinnäkin todetaan, että mikä tahansa kohdekielinen lause on jollakin todennäköisyydellä sopiva käännös lähdekielisen tekstin lauseelle. Merkitään suomennettavaa ruotsinkielistä lausetta kirjaimella r ja mielivaltaista suomenkielistä lausetta kirjaimella s. Todennäköisyys sille, että käännösohjelma tulostaa juuri lauseen s saatuaan syötteenä lauseen r, on P(s | r). Merkintä tarkoittaa s:n todennäköisyyttä silloin, kun r on tunnettu ja varma. Todennäköisyys on aina väliltä [0, 1], mutta mielivaltaisesti valitulle lauseelle yleensä lähellä nollaa. Erilaisista käännösvaihtoehdoista valitaan se, joka suurimmalla todennäköisyydellä on oikea. Tämä toimintatapa tosin kääntyy ohjelman heikkoudeksi, ellei käytössä oleva korpus ole riittävän laaja.

Metodin toinen lähtökohta pohjautuu edelliseen, mutta vaatii intuitiivisempaa päättelyä. Sen esitteli joukko IBM:n omistaman Thomas J. Watsonin tutkimuskeskuksen tiedemiehiä vuonna 1993. Tutkijaryhmä tiesi, että kohdekielisiin teksteihin voi jäädä huonosti valittuja ilmaisuja, jos ohjelmisto jäljittelee täysin realistisesti sellaista ihmiskääntäjää, joka puhuu käännöksen lähdekieltä äidinkielenään. IBM:n tutkijat oivalsivat, että käännökset ovat sujuvampia, jos käännöksen suunta voidaan jollakin tavalla vaihtaa kesken prosessin. Tätä varten esitetään varsin erikoinen mutta hyödyllinen hypoteesi.

Oletetaan, että kun ihminen muodostaa mielessään äidinkielellään olevan lauseen, hän itse asiassa muodostaa ensin vieraskielisen lauseen ja kääntää sen sitten päässään äidinkielelleen. Näinhän ajatusten muodostaminen ei käytännössä tapahdu, mutta oletus on ratkaisu käännösten sujuvuuden lisäämiseen. Tilastollisessa konekääntämisessä äidinkielenä ajatellaan lähdekieltä ja vieraana kielenä kohdekieltä. Käsitellään esimerkkinä tilannetta, jossa ruotsinkielinen lause r halutaan kääntää suomeksi. Käännösohjelman tehtäväksi jää nyt selvittää, mitä suomenkielistä lausetta s ihminen (oletetusti) ajatteli muodostaessaan päässään lauseen r, joka ohjelmalle syötettiin.

Käytännössä oletuksesta seuraa edellä käytettyjen merkintöjen mukaisesti, että käännösohjelman halutaan laskevan todennäköisyys P(s | r) käyttäen apuna todennäköisyyttä P(r | s). Tämä on mahdollista, kun sovelletaan 1700-luvulla kehitettyä Bayesin teoreemaa, joka on johdettu ehdollisen todennäköisyyden määritelmästä peruslaskutoimituksilla. Teoreema osoittaa mielivaltaisen ehdollisen todennäköisyyden ja sen oman vastakohdan välisen yhteyden. Toisen, erityisesti konekääntämiseen soveltuvan määritelmän mukaan teoreema on kuvaus todennäköisyyskäsitysten päivittämisestä, kun todistusaineisto (tässä tapauksessa korpuksen kulloinkin tarkasteltava lause) vaihtelee.

Teoreema esitetään muodossa

$$P(s \mid r) = \frac{P(r \mid s) * P(s)}{P(r)}$$

Kaavassa P(s) ja P(r) ovat kielimalleja. Kielimalli tarkoittaa sanajonon (esim. lauseen) esiintymisen todennäköisyyttä käytettävässä (kaksikielisessä) korpuksessa. Malli on

korpusriippuvainen eli yksi ohjelma voi tuottaa samalle lauseelle useita eri todennäköisyyksiä, jos korpusta muunnellaan. [24]

Jos lause s koostuu peräkkäisistä suomenkielisistä sanoista $ss_1 \ldots ss_m$, voidaan P(s) kirjoittaa muodossa $P(ss_1, \ldots, ss_m)$. Lauseen todennäköisyyttä ei ole järkevää laskea sana kerrallaan, ikään kuin sanat olisivat toisistaan erillisiä. Sen sijaan kielimalleja hyödynnetään n-grammeina, joissa jokaisen sanan esiintymätodennäköisyyttä laskettaessa otetaan huomioon n–1 sitä edeltävää sanaa. Kielimallin tuottama todennäköisyys on yksittäisten sanojen todennäköisyyksien kertoma

$$\prod_{i=1}^{m} P(s_i \mid s_{i-(n-1)}, \ldots, s_{i-1})$$

Mitä enemmän vertailuja tehdään, sitä parempi lopputulos saadaan, mutta toisaalta korkea n-muuttujan arvo johtaa suureen laskentatehotarpeeseen ja hidastaa ohjelman toimintaa.

Yksi yleisimmin käytetyistä n-grammeista on trigrammi. Siinä n saa arvon kolme, joten kunkin sanan todennäköisyydessä otetaan huomioon kaksi sitä välittömästi edeltävää sanaa oikeassa järjestyksessä. Sanan s_{i+2} todennäköisyys riippuu siis siitä, kuinka monta kertaa sekvenssit $\{s_i, s_{i+1}, s_{i+2}\}$ ja $\{s_i, s_{i+1}\}$ esiintyvät korpuksessa. Esiintymiskertojen suhdelukua $\frac{\{s_i, s_{i+1}, s_{i+2}\}}{\{s_i, s_{i+1}\}}$ kutsutaan trigrammitaajuudeksi f_{i+2}. [25]

Ennen Bayesin teoreemaan palaamista sovelletaan trigrammia lauseeseen ”Minä kirjoitan TkK-tutkielmaa kotona”. Virkkeen ensimmäistä ja toista sanaa ei voida verrata kahteen edelliseen sanaan, joten otetaan käyttöön merkintä <va>, jonka ohjelma tunnistaa virkkeen aluksi. Kielimalli palauttaa lauseelle todennäköisyyden P(Minä | <va>, <va>) * P(kirjoitan | <va>, Minä) * P(TkK-tutkielmaa | Minä, kirjoitan) * P(kotona | kirjoitan, TkK-tutkielmaa). Bayesin teoreeman $P(s \mid r) = \frac{P(r \mid s) * P(s)}{P(r)}$ kielimallit P(s) ja P(r) voidaan laskea kaksikielisestä korpuksesta tällä menetelmällä.

Lause r on ohjelmalle annettu syöte, joka ei muutu ajoaikana. Teoreeman nimittäjä P(r) on siis vakio, joten todennäköisyys P(s | r) on suoraan verrannollinen kaavan osoittajaan.

Ainoa, mitä ohjelma voi tehdä saavuttaakseen mahdollisimman suuren P(s | r) -todennäköisyyden, on kokeilla korpuksen eri lauseiden sijoittamista s:n paikalle.

Merkitään, että ŝ on lauseen r tarkin saatavilla oleva suomenkielinen vastine. Lauseelle ŝ voidaan verrannollisuuden nojalla kirjoittaa

$$\hat{s} = \arg\max_{s}(P(s) * P(r \mid s))$$

Yhtälön kertolaskun osat täydentävät toisiaan: luvun P(s) arvo on suuri, kun s on kielellisesti moitteeton ja samankaltaisia lauseita löytyy korpuksesta runsaasti. Todennäköisyys ei riipu r:stä lainkaan. Sen sijaan P(r | s) on korkea kaikille sellaisille suomenkielisille lauseille, jotka sisältävät suunnilleen ruotsinkielistä lausetta r vastaavat ilmaisut. Vertailussa ei kiinnitetä huomiota sanajärjestykseen ja kielioppiin. Keskenään kerrottuina nämä kaksi todennäköisyyttä antavat tietoa siitä, onko lause s sekä sisällöllisesti että kieliopillisesti kelvollinen käännös.

Mahdollisimman suurien todennäköisyyksien ja optimaalisen käännöksen löytämiseksi vertailuja joudutaan tekemään huomattavan paljon. Määrään vaikuttaa myös luettavan kaksikielisen korpuksen koko. Laskutoimitukset ovat sinänsä varsin yksinkertaisia, mutta määränsä vuoksi työläitä ja hitaita.

5.2 Hierarkkiset fraasit

Yhtälön $\hat{s} = \arg\max_{s}(P(s) * P(r \mid s))$ merkintätapa antaa ymmärtää, että kaikki käännösohjelman koskaan tarvitsemat lauseet s ja r löytyvät korpuksesta sellaisinaan, valmiina vertailua varten. Toisin sanoen korpuksen tulisi sisältää käännettävien tekstien kaikki mahdolliset lauseet tyhjentävänä listana. Näin laaja korpus ei ole mahdollinen – ja tekisi olemassaolollaan käännösohjelmista tarpeettomia. Käännettävät lauseet on pilkottava pienempiin vertailuosiin, fraaseihin. Suppeimmillaan fraasi voi olla yhden sanan mittainen.

Yksi tapa lauseiden osittamiseen on esitelty esimerkkipohjaisen konekääntämisen yhteydessä luvussa 4.1. Kyseinen metodi kuitenkin käsittelee jokaista fraasia yhtä tärkeänä

ja on paremmin yhteen sovitettavissa lauseidenvälisten etäisyyksien vertailemiseen kuin todennäköisyyslaskentaan. Tilastollisen konekääntämisen tutkimuksessa on 2000-luvulla kehitetty osittamismenetelmä, jolla saavutetaan parempia tuloksia ja joka samalla täydentää asiasanastojen hierarkkisuutta.

Käännettävät virkkeet rakentuvat usein asteittain siten, että jokin fraasi sisältyy yhtenä osana toiseen fraasiin. Mitä pidempiä virkkeet ovat, sitä enemmän niissä on tämänkaltaista kerroksittaisuutta. Hierarkkisia fraaseja esiintyy kuitenkin myös yhden lauseen pituisissa virkkeissä. Fraasit jaetaan ei-terminaaleihin ja terminaaleihin (engl. nonterminal, terminal). Ei-terminaalit ovat fraaseja, jotka voidaan jakaa vielä pienempiin osiin, eli terminaaleihin ja mahdollisesti toisiin ei-terminaaleihin. Niissä on yksi tai useampia tuntemattomia tekijöitä. Vastaavasti terminaalit ovat virkkeen yksiselitteisimpiä ja pienimpiä osia, jotka eivät ole jaettavissa.

Esitellään virke ”Minä olen yksi niistä opiskelijoista, joilla on ongelmia tutkielman kanssa”. Esimerkiksi pronominissa ”minä” ei ole mitään sellaista piirrettä, joka voidaan korvata toisella piirteellä, joten se on terminaali. Sen sijaan rakenne ”on [jotakin] [jonkin] kanssa” on ei-terminaali, jossa hakasulkeiden sisällöt voidaan korvata useilla erilaisilla sanoilla tai sanajoukoilla. Tässä tapauksessa on käytetty terminaaleja ”ongelmia” ja ”tutkielman”.

Ei-terminaalien korvattavat osat merkitään tuntemattomiksi: ”on X_1 X_2 kanssa”. Jos edellä esitelty virke halutaan kääntää esimerkiksi englanniksi, ei-terminaali merkitään muodossa
$X \rightarrow$ <on X_1 X_2 kanssa; have X_1 with X_2>

Olivatpa lähdekielisen lauseen piirteet X_1 ja X_2 mitä tahansa, niiden käännökset voidaan sijoittaa ei-terminaalin englanninkieliseen osaan siten, että käännös on yhtä oikeellinen ja järkevä kuin alkuperäinenkin suomenkielinen fraasi.

Kirjainta X käytetään sekä yksittäisten tuntemattomien piirteiden (X_1 ja X_2) että koko ei-terminaalin (X) merkitsemiseen, koska ei-terminaali saattaa kokonaisuudessaan sisältyä

johonkin laajempaan ei-terminaaliin ja toimia sen tuntemattomana piirteenä. Tällä tavalla fraasit muodostavat keskinäisen hierarkian.

Hierarkiassa alimpina ovat terminaalit. Välittömästi niiden yläpuolella ovat ne ei-terminaalit, joihin sisältyy yksi tai useampia tuntemattomia osia ja lisäksi yksi tai useampia terminaaleja. Kaikkein ylimpinä ovat erityiset sidossäännöt, joihin ei sisälly lainkaan terminaaleja. Sidossääntöjä merkitään yleensä kirjaimella S ja niitä voidaan pitää ei-terminaalien erikoistapauksina. Kaikkien näiden hierarkkisten rakenteiden kokoelmaa kutsutaan synkronoiduksi kontekstiriippumattomaksi kieliopiksi. Sen käyttöönotto konekääntämisessä tekee käännösprosessista entistä syntaksikeskeisemmän. [26]

Esimerkkivirkkeen fraasien hierarkia ylhäältä alaspäin on seuraava:
$<S_1, S_1> \rightarrow < S_2\ X_3; S_2\ X_3>$
$\rightarrow < X_6\ X_5\ X_3; X_6\ X_5\ X_3>$
$\rightarrow$ <Minä olen X_3; I am X_3>
$\rightarrow$ <Minä olen yksi X_4, joilla X_7; I am one of X_4 who X_7>
$\rightarrow$ <Minä olen yksi X_4, joilla on X_1 X_2 kanssa; I am one of X_4 who have X_1 with X_2>
$\rightarrow$ <Minä olen yksi niistä opiskelijoista, joilla on ongelmia tutkielman kanssa; I am one of those students who have problems with the thesis>

Käytettyjen alaindeksien suuruusjärjestys on epäoleellinen, sillä niitä käytetään vain piirteiden erottelemiseen toisistaan.

Fraaseille voidaan laskea todennäköisyydet luvussa 5.1 kuvatulla tavalla. Hierarkkisten fraasien suurimpana etuna pidetään sitä, että menetelmä ei käsittele virkkeiden fraaseja samantasoisina, mitä ne eivät olekaan. Metodilla on saavutettu hyviä tuloksia ja viime vuosina konekääntämistutkimuksen painopiste onkin alkanut siirtyä esimerkkipohjaisista tilastollisiin ohjelmistoihin.

Kun jokin fraasi- tai lausekäännös on kerran tehty, ohjelma voi tallentaa sen, joten käännösprosessia ei tarvitse tehdä uudestaan alusta alkaen, jos sama rakenne tulee myöhemmissäkin teksteissä vastaan. Tämä voi nopeuttaa ohjelman toimintaa, mikä tosin

riippuu siitä, onko tehokkaampaa etsiä valmis käännös muistista vai laskea se kokonaan uudestaan. Toinen merkittävä hyötynäkökohta – jota alan kirjallisuudessa ehkä hieman yllättäen korostetaan edellä mainittua enemmän – on, että koneoppimisen ansiosta ohjelma kääntää saman virkkeen aina samalla tavalla.

Oppimisprosessia kutsutaan hallituksi koneoppimiseksi, koska toteuttaja päättää, mitä lähdeaineistoa ohjelmalle annetaan oppimista varten. Oppimistehokkuuden kannalta keskeisintä onkin korpuksen laatu ja laajuus. Eräitä uusia käännösohjelmistoja on saatu koneoppimisen ansiosta hyvään toimintakuntoon muutamissa viikoissa tai jopa päivissä. [27]

2000-luvulla on alettu kehittää myös graafiteoriaan pohjautuvia koneoppimismenetelmiä, jotka eivät käsittele tekstin virkkeitä yksi kerrallaan, vaan kykenevät havaitsemaan laajojakin teemakokonaisuuksia. Kilpailevia algoritmeja on useita ja tällä hetkellä on liian aikaista sanoa, mitkä niistä osoittautuvat kehityskelpoisimmiksi ja vakiintuvat laajaan käyttöön. [28]

6 Yhteenveto

Luonnollisen kielen koneellinen kääntäminen on osoittautunut oletettua laajemmaksi ongelmakokonaisuudeksi. Ala sivuaa kielitiedettä, tekoälytutkimusta, matematiikan eri osa-alueita ja tulevaisuudessa luultavasti yhä enemmän myös psykologiaa ja ihmisaivojen tutkimusta. Pelkkä ohjelmointiasiantuntemus ei riitä ohjelmistojen kehittämiseen.

Ideaalinen käännösohjelmaimplementaatio on täysautomaattinen, korkealaatuinen, universaali ja laaja-alainen. Universaalilla tarkoitetaan tässä kaikkien kielien tuntemista ja laaja-alaisuus merkitsee kaikenlaisten tekstien ammattitermien ja kirjoitustyylien tuntemista. Kaikkien neljän ominaisuuden toteuttaminen samassa ohjelmistossa on merkittävä haaste eikä siihen olla 1950-luvun jälkeen juuri pyrittykään. On mielekkäämpää laatia useita erilaisia ohjelmistoja, jotka on optimoitu käyttäjiensä tarpeiden mukaan.

Esimerkkipohjaisessa konekääntämisessä oletetaan, että ohjelman ei tarvitse ymmärtää käännettävän tekstin sisältöä, vaan että ihmiskääntäjäkin pilkkoo mielessään virkkeitä osiin ja kääntää yhden osan kerrallaan. Tilastollisessa käännösmetodissa puolestaan hyödynnetään virkkeiden rakenteiden hierarkkisuutta sekä sanasto- että fraasitasolla, joten elementtejä käsitellään toisiinsa kytkettyinä eikä eristettyinä. Jälkimmäinen lähestymistapa on vallannut alaa jo usean vuoden ajan ja tämä kehitys tulee luultavasti tulevaisuudessa korostumaan entisestään. Toisaalta vaikuttaa siltä, että kaikkein yksinkertaisimmatkaan eli sääntöpohjaiset käännösohjelmat eivät tule täysin katoamaan, sillä niiden käyttö on yksinkertaista ja nopeaa.

Tässä tutkielmassa esitellyistä kolmesta käännöstavasta kaksi on kehitetty alle 30 vuotta sitten. Tänä aikana maailman tiedonvälitys, kaupankäynti ja hallinto ovat voimakkaasti kansainvälistyneet, mistä Euroopan Unioni ja G20-talousryhmä ovat hyviä esimerkkejä. Samalla myös käännösten kysyntä on kasvanut. Tulevaisuudessa tullaan odotettavasti kehittämään aivan uudenlaisia, entistä monimutkaisempia konekäännöstapoja. Niiden rakenteesta ja matemaattisesta perustasta ei kuitenkaan tämän tutkielman puitteissa voida esittää päteviä ennusteita.

Viitteet

[1] Roinila, Markku. 1996. ”Descartes, Leibniz ja universaalikielen mahdollisuus.” *niin & näin* 4/1996.

[2] Swedenborg, Emanuel (johdanto: Siukonen, Jyrki). 2000. *Clavis Hieroglyphica: hieroglyfinen avain ja muita filosofisia tekstejä*. Tampere: Gaudeamus Kirja, s. 63–74, 132–139.

[3] Viro, Voitto. 1960. *Huomispäivä on teidän.* Porvoo: Werner Söderström Osakeyhtiö.

[4] Slocum, Jonathan. 1985. “A Survey of Machine Translation: its History, Current Status, and Future Prospects.” *Computational Linguistics*, 11. vuosikerta, s. 1–17. MIT Press, Cambridge, MA, USA.

[5] Schank, Roger. 1984. *Tekoälyn mahdollisuudet.* Espoo: Weilin+Göös.

[6] Hutchins, John. 2000. *Early Years in Machine Translation.* Amsterdam: John Benjamins Publishing Company.

[7] Honkonen, Janos. 2007. “Tietokone kääntäjänä: Kone menee koulunpenkille.” *Tiede* 9/2007.

[8] Nagao, Makoto. “A Framework of a Mechanical Translation between Japanese and English by analogy principle.” Teoksessa: Elithorn, Alick; Banerji, Ranan (toim.). 1984. *Artificial and Human Intelligence*. Amsterdam: Elsevier Science Publishers.

[9] Honkonen, Janos. 2007. ”Tietokone kääntäjänä: Kieli sujuu, vaikka ymmärrys pätkii.” *Tiede* 9/2007.

[10] Kovacheva, Todorka; Mitev, Koycho; Dimitrov, Nikolay. ”Text-to-Text Machine Translation System (TTMT System) – A Novel Approach for Language Engineering.”

Teoksessa: Markov, Krassimir; Ivanova, Krassimir (toim.). 2006. *Proceedings of the Fourth International Conference of Information Research and Applications.* Sofia: FOI-COMMERCE.

[11] Crystal, David. 1992. *An Encyclopedic Dictionary of Language and Languages*. Oxford: Blackwell, s. 85

[12] Rao, Durgesh. 1998. "Machine translation: A gentle introduction." *Resonance,* 3. vuosikerta, numero 7/1998, s. 61–70.

[13] Llitjós, Ariadna; Carbonell, Jaime; Lavie, Alon. 2005. "A Framework for Interactive and Automatic Refinement of Transfer-based Machine Translation." *EAMT 2005 Conference Proceedings*, s. 87–96.

[14] Bender, Howard. 2002. *Natural Intelligence in a Machine Translation System. Machine Translation: From Research to Real Users.* Heidelberg: Springer Berlin.

[15] Kiviaho, Allan. Suomen interlinguayhdistyksen kotisivusto. <http://www.interlingua.fi/indexfi.htm> Viitattu 29.9.2009

[16] Brockett, Chris; Aikawa, Takako; Aue, Anthony; Menezes, Arul; Quirk, Chris. "English-Japanese Example-Based Machine Translation Using Abstract Linguistic Representations." <http://acl.ldc.upenn.edu/coling2002/workshops/data/w07/w07-04.pdf> Viitattu 2.10.2009

[17] Babych, Bogdan; Hartley, Anthony; Sharoff, Serge. 2007. "Translating from under-resourced languages: comparing direct transfer against pivot translation." *Proceedings of MT Summit XI.* Kööpenhamina, s. 29–35.

[18] Leavitt, John; Lonsdale, Deryle; Franz, Alexander. 1994. "A Reasoned Interlingua for Knowledge-based Machine Translation." *Proceedings of the 1994 Canadian Conference of Artificial Intelligence*. Banff, Alberta, Kanada.

[19] Dave, Shachi; Parikh, Jignashu; Bhattacharyya, Pushpak. 2001. “Interlingua-based English–Hindi Machine Translation and Language Divergence.” *Machine Translation*, 16. vuosikerta, numero 4/2001, s. 251–304.

[20] Brown, Ralf. *Generalized Example-Based Machine Translation*. <http://www.cs.cmu.edu/afs/cs.cmu.edu/user/ralf/pub/WWW/ebmt/> Viitattu 10.10.2009.

[21] Yeh, Yu-Hui. 2002. *Example-Based Machine Translation (Chinese to English)*. The University of Queensland. The School of Information Technology and Electrical Engineering. Submission for the degree of Bachelor of Engineering.

[22] Yleinen suomalainen asiasanasto YSA. <http://www.yso.fi/onto/ysa/Y94503> Viitattu 12.10.2009

[23] Sumita, Eiichiro. ”Example-based machine translation using DP-matching between word sequences.” <http://acl.ldc.upenn.edu/W/W01/W01-1401.pdf> Viitattu 12.10.2009.

[24] Brown, Peter; Della Pietra, Stephen; Della Pietra, Vincent; Mercer, Robert. 1993. “The mathematics of statistical machine translation: parameter estimation.” *Computational Linguistics*, 19. vuosikerta, numero 2/1993, s. 263–311.

[25] Roukos, Salim. 1997. *Survey of the State of the Art in Human Language Technology: Language Representation*. New York: Cambridge University Press.

[26] Chiang, David. 2005. “A Hierarchical Phrase-Based Model for Statistical Machine Translation.” *Proceedings of the 43rd Annual Meeting of the ACL*. Ann Arbor, Michigan, Yhdysvallat, s. 263-270.

[27] Lopez, Adam. 2008. “Statistical Machine Translation.” *ACM Computing Surveys*, 40. vuosikerta, numero 3/2008, s. 8:1–8:49.

[28] Alexandrescu, Andrei; Kirchhoff, Katrin. 2009. “Graph-based Learning for Statistical Machine Translation.” *Proceedings of the 2009 Annual Conference of the North American Chapter of the ACL*. Boulder, Colorado, Yhdysvallat, s. 119–127.