TAMPEREEN TEKNILLINEN YLIOPISTO

ESA-PEKKA HAKULINEN

LINJA-AUTOISTA KERÄTTÄVÄLLE AVOIMELLE DATALLE SOVELTUVA TIETORAKENNE

Diplomityö

Tarkastajat: professori Ari Visa ja
professori Seppo Pohjolainen
Tarkastaja ja aihe hyväksytty
Luonnontieteiden tiedekunnan
kokouksessa 4.6.2014

TIIVISTELMÄ

TAMPEREEN TEKNILLINEN YLIOPISTO
Teknis-luonnontieteellinen koulutusohjelma
HAKULINEN, ESA-PEKKA: Linja-autoista kerättävälle avoimelle datalle soveltuva tietorakenne
Diplomityö, 48 sivua, 6 liitesivua
Lokakuu 2014
Pääaine: Matematiikka
Tarkastajat: professori Ari Visa ja professori Seppo Pohjolainen
Avainsanat: Avoin data, graafiteoria, liikennedata, tiedonlouhinta, tietorakenne

Tampereen kaupunki avasi syyskuussa 2013 Tampereen joukkoliikenteen reaaliaikaisen rajapinnan. Tämä kaikille avoin rajapinta tarjoaa kaupunkiliikenteen bussien sijaintitiedot sekunnin välein. Busseista kerättävää liikennedataa hyödyntämällä on mahdollista saada reaaliaikaista tietoa kaupungin liikenneverkon toiminnasta ja siten parantaa datan avulla liikenteen tilannekuvaa sekä lisätä tilannetietoisuutta. Tämän diplomityön tavoitteena on kehittää graafipohjainen tietorakenne, johon kerätään linja-autoista saatavaa dataa. Tietorakenne toimii datan tallentamisen sekä analysoinnin välineenä.

Diplomityön teoriaosassa käydään läpi työssä kehitetyn tietorakenteen taustalla olevaa graafi- ja matriisiteoriaa sekä esitetään Haversinen kaava, jolla voidaan laskea koordinaattipisteiden välisiä etäisyyksiä. Tietorakenteen testaamiseksi Tampereen keskustan ja kaupungin itälaidan välille rajattiin neljän kilometrin pituinen testireitti sekä kerättiin testidataa viiden arkipäivän ajalta. Työssä esitetään tietorakenteen suunnittelu ja toteutus sekä kuvataan algoritmien avulla testidatan prosessointi ja tietorakenteeseen tallentaminen. Diplomityön tulokset koostuvat pääosin tietorakenteeseen tallennetusta bussien ajoaikadatasta muodostetuista datamatriiseista sekä ajoaikahistogrammeista.

Diplomityössä kehitettiin menetelmä, jossa graafitietorakenteen kaarialkioon tallennetaan vain yksi datanäyte kunkin bussilinjan jokaisen linjalähdön bussista. Menetelmää hyödyntämällä bussidataa sisältävien muuttujien muistin käyttöä saatiin merkittävästi pienennettyä sekä päästiin eroon datassa havaituista ongelmista. Analyysit työn tuloksissa esitettävistä datamatriiseista osoittivat, että tietorakenteeseen tallennetusta datasta voidaan tuottaa luotettavaa tietoa bussiliikenteestä. Testidatasta muodostettujen ajoaikahistogrammien muotojen perusteella pystyttiin esittämään arvioita bussien etenemisestä testireitillä. Bussien ajoaikojen vaihtelut testireitillä noudattivat yleisiä työmatkaliikenteen ajankohtia ja ruuhka-aikoja. Työssä kehitettyä graafipohjaista tietorakennetta voidaan pitää hyvänä lähtökohtana reaaliaikaista datanhallintaa tavoiteltaessa.

ABSTRACT

TAMPERE UNIVERSITY OF TECHNOLOGY
Master's Degree Programme in Science and Engineering
HAKULINEN, ESA-PEKKA: A data structure appropriate for storing open data from buses
Master of Science Thesis, 48 pages, 6 Appendix pages
October 2014
Major: Mathematics
Examiner: Professor Ari Visa and Professor Seppo Pohjolainen
Keywords: Open data, graph theory, traffic data, data mining, data structure

The City of Tampere opened a data interface for public transport In September 2013. This open data interface provides the location information about the buses in city traffic. Real-time information is updated every second. With the traffic data collected from the buses it is possible to get real-time information about the traffic situation in the city and the current traffic flow. In that way the situational awareness of the traffic can be improved. The goal of this thesis is to develop a graph-based data structure for the data collected from the public transport buses. The data structure is a tool to store and analyze the data.

The theoretical basis of the thesis consists of the basic theory of graphs and matrices. The theory section presents the Haversine formula that is used for calculating the distance between two coordinate points. In order to test the data structure, a four kilometer long test route in the city of Tampere was created and a test data of a period of five working days was collected. The planning and implementation process of the data structure is described in the thesis. The processing of the test data and the storing of the data to the data structure are explained with algorithms. The results of the thesis consist of data matrices and bus driving time histograms formed from the stored data in the data structure.

In this thesis, a method where only one data sample per bus is stored in the one edge of the graph data structure was introduced. With this method the memory usage of the variables containing bus data was relevantly reduced. The analysis of the data matrices shown in the results indicated that it is possible to produce reliable information about the bus traffic from the data stored in the data structure. The differences in the bus driving times were equal to the normal traffic rush hours and peak commuting hours. With an analysis of the shapes of the bus driving time histograms, it was possible to estimate how the buses are moving along the test route. The graph-based data structure introduced in this thesis can be considered as a good basis for further development heading to real-time data management.

ALKUSANAT

Tämä diplomityö on tehty Tampereen teknillisen yliopiston signaalinkäsittelyn laitoksen multimedian ja tiedon louhinnan tutkimusryhmässä.

Diplomityön ohjaamisesta sekä neuvoista työn tekemisen aikana haluan kiittää multimedian ja tiedonlouhinnan tutkimusryhmän tutkijoita Juha Jylhää, Riitta Kermistä sekä Marja Ruotsalaista. Tampereen yliopiston puolelta työn edistymisessä ovat auttaneet professori Jyrki Nummenmaa sekä tutkija Paula Syrjärinne. Kiitos kuuluu myös professori Peter Thanischille, joka tarjosi työkalun diplomityössä käsitellyn datan keräämiseen. Lisäksi kiitän työn tarkastajia professori Ari Visaa sekä professori Seppo Pohjolaista.

Helsingissä 6.8.2014

Esa-Pekka Hakulinen

SISÄLLYS

MERKINNÄT JA LYHENTEET

G	Graafi
$V(G)$	Graafin solmujen joukko
$E(G)$	Graafin kaarien joukko
v_k	Solmu $k = 1, ...n$
e_k	Kaari $k = 1, ...m$
(a, b)	Pari
$d(v_i)$	Solmun v_i aste
$d^-(v_i)$	Solmun v_i tuloaste
$d^+(v_i)$	Solmun v_i lähtöaste
$\emptyset$	Tyhjä joukko
W	Graafin kulku
A	Matriisi
H	Graafin vieruspistematriisi
K	Graafin kytkentämatriisi
m	Matriisin vaakarivien lukumäärä
n	Matriisin sarakkeiden lukumäärä
a_{ij}	Matriisin vaakarivillä i sarakkeessa j oleva alkio
(ϕ_n, λ_n)	Koordinaattipiste
R	Maapallon säde
d	Pisteiden välinen etäisyys
f^{-1}	Funktion f käänteisfunktio

3G	Kolmannen sukupolven matkapuhelinteknologia.
C++	Ohjelmointikieli.
CEN	*Comité Européen de Normalisation* Euroopan standardoimisjärjestö.
CSV	*Comma-Separated Values* Tiedostomuoto taulukko- tai luetteloerotinmuotoisen tiedon tallentamiseen.
GPS	*Global Positioning System* Satelliittipaikannusjärjestelmä.
ITS	*Intelligent Transportation Systems* Yleistermi, jolla viitataan tieto- ja viestintätekniikan soveltamiseen liikenteessä ja liikenneverkoissa.
JSON	*JavaScript Object Notation* Avoimen standardin tiedostomuoto tiedonvälitykseen.
MAC	*Media Access Control* Tietoliikenneverkon osajärjestelmä.
SIRI	*Service Interface for Real Time Information* Yleiseurooppalainen rajapintastandardi liikennetiedon välittämiseen joukkoliikennetoimijoiden välillä.
TKL	*Tampereen Kaupunkiliikenne Liikelaitos* Tampereen kaupungin omistama liikelaitos, joka hoitaa linja-autoliikennettä.
XML	*Extensible Markup Language* Merkintäkieli rakenteisen tiedon tallentamiseen ja siirtämiseen.

1. JOHDANTO

Suomen hallitus teki 3.3.2011 periaatepäätöksen julkishallinnon digitaalisten tietoaineistojen avaamisesta [1]. Julkishallinnon vauhdittamana myös yritykset ja muut organisaatiot ovat yhä enemmän alkaneet avata datalähteitään. Näin saadulle avatulle datalle, tai yleisemmin avoimelle datalle, on eri lähteistä löydettävissä lukuisia määritelmiä. Vuonna 2004 perustettu kansainvälinen avointa dataa ja sisältöä edistävä organisaatio Open Knowledge Foundation määrittelee avoimen datan useiden erilaisten datan vaatimusten avulla. Näitä avoimen datan ominaispiirteitä ovat datan avoin saatavuus, datan uudelleenkäytettävyys ja -jakelu, datan yleinen rajoittamattomuus sekä datan koneluettavuus. [2] Helsingin seutua koskevaa tietoa avaava Helsinki Region Infoshare -hanke on määritellyt avoimen datan seuraavasti: ”Avoimella datalla tarkoitetaan julkishallinnolle, yrityksille, organisaatioille ja yksityishenkilöille kertynyttä jalostamatonta informaatiota, johon on avattu maksuton pääsy organisaation ulkopuolisille.” [3]

Tampereen kaupunki on linjannut kaupunkistrategiassaan yhdeksi strategiseksi painotukseksi tiedon avoimuuden [4, s. 9]. Kaupungin yksiköt ovatkin avanneet datalähteitä runsaasti – saatavilla on tietoa aina Tampereen talviliukumäistä asemakaavaindekseihin. Jo aiemmin julkisina olleiden bussien reitti- ja pysäkkiaikataulujen rinnalle avattiin syyskuussa 2013 Tampereen joukkoliikenteen reaaliaikainen SIRI-rajapinta (Service Interface for Real Time Information). SIRI on yleiseurooppalainen rajapintastandardi, jota käytetään julkisen liikenteen aikataulujen, ajoneuvojen ja yhteyksien reaaliaikaisten tietojen välittämiseen joukkoliikennetoimijoiden välillä.

Rajapinnan avulla saatavaa reaaliaikaista SIRI-bussidataa on aiemmin tutkittu ja hyödynnetty suhteellisen vähän. Aihetta käsitteleviä tutkimusartikkeleita on löydettävissä vain yksittäisiä kappaleita, jotka nekin reaaliaikaisen datan käsittelemisen sijaan liittyvät lähinnä aiemmin kerätyn SIRI-datapaketin käyttämiseen jonkin liikennesovelluksen testaamisessa.

Tämän diplomityön tavoitteena on kehittää graafipohjainen tietorakenne, johon kerätään linja-autojen SIRI-dataa. Tietorakenne toimii datan tallentamisen sekä analysoinnin välineenä. Lisäksi tietorakenteen on tarkoitus mahdollistaa tulevaisuudessa uusia aiempaa parempia avoimen liikennedatan sovelluksia.

Perinteisesti bussiliikennettä on tarkasteltu bussilinjojen tai yksittäisten ajoneuvojen tasolla, esimerkiksi aikataulun mukaisia bussien linjalähtöjä tutkimalla. Yk-

sittäisen bussilinjan tai linjanlähdön sijasta tässä työssä ollaan kiinnostuneita mitä bussiliikenteelle tapahtuu tien- ja kadunväleillä sekä esimerkiksi samalla kadulla olevien liikennevalojen välillä. Graafimallin tausta-ajatuksena oli, että tietorakennegraafin kaari- ja solmutietoalkioihin voitaisiin tallentaa bussien SIRI-dataa liikenteen analysoimista varten.

Tietorakenteen suunnittelun lähtökohdiksi muodostui kolme datan varastointiin sekä jatkokäyttöön liittyvää tavoitetta: 1) Suuren datamäärän prosessointi ja supistaminen, 2) merkityksellisen ja hyödyllisen tiedon louhiminen datan joukosta sekä 3) alustavien arvioiden muodostaminen menetelmän soveltuvuudesta reaaliaikaisen liikenneinformaation tuottamiseen sekä hyödyntämiseen. Tietorakenteen kehittämisen lisäksi työssä tutustutaan Tampereen joukkoliikenteen avoimen rajapinnan kautta saatavaan Tampereen bussien reaaliaikaiseen bussidataan. Työssä esitellään SIRI-standardia, Tampereen bussidatan rakennetta, datan keräämistä sekä prosessointia.

Kun kaikki kaupungin joukkoliikenteessä liikennöivät bussit lähettävät dataa sekunnin välein lähes ympäri vuorokauden, on kertyvän datan määrä suuri jo Tampereen kokoisen verrattain pienen kaupungin bussiliikenteessä. Tässä diplomityössä käytetyssä datankeräysmenetelmässä SIRI-datarivejä kertyi vuorokaudessa parhaimmillaan yli kahdeksan ja puoli miljoonaa kappaletta. Kun jokainen datarivi sisältää 24 alkiota, on esimerkiksi vuositasolla kertyvän datan määrä jo huomattava.

Tutustuttaessa Tampereen bussien SIRI-dataan diplomityön suunnitteluvaiheessa huomattiin, että kerätty data sisältää runsaasti hyödytöntä informaatiota, niin sanottua roskadataa. Roskadatana pidettiin muun muassa kopioitunutta dataa, eli kahta tai useampaa peräkkäistä saman ajoneuvon lähettämää samansisältöistä datariviä. Roskadataksi tulkittiin myös niin sanottu haamudata eli tapaukset, joissa liikennöintivuoronsa jo lopettanut ajoneuvo lähetti pahimmillaan useiden tuntien ajan sijaintidataansa muun muassa bussivarikolta. Tässä työssä käsiteltävää datan jatkohyödyntämistä ajatellen osa luvussa 3.3 esiteltävistä SIRI-datakentistä sisältää merkityksetöntä tai päällekkäistä tietoa. Esimerkiksi bussireittien alku- ja päätepysäkkien nimet eivät ole oleellista tietoa bussien ajoaikajakaumia tutkittaessa.

Joukkoliikenne- ja kaupunkisuunnittelun kannalta tallennetun bussidatan tiedonlouhinnasta sekä analysoinnista voidaan löytää useita merkittäviä hyötynäkökohtia. Konkreettisena esimerkkinä mainittakoon bussikaistojen sekä liikennevaloetuuksien vaikutus bussien ajoaikoihin kaupunkiliikenteessä. Myös uusien bussilinjojen- ja reittien suunnittelussa on mahdollista hyödyntää tarkkaa tietoa käytössä olevan linjaverkoston toiminnasta, esimerkiksi laadittujen aikataulujen paikkansapitävyydestä verrattuna toteutuneisiin ajoaikoihin eri katuväleillä. Muina ideoina ja esimerkkeinä mainittakoon ekologisempaan ajotapaan siirtymisen vaikutus bussien ajonopeuksiin ja ajoaikoihin, sekä sään ja talvikunnossapidon vaikutukset yksittäisten bussien ajokäyttäytymiseen.

Liikennedataa hyödyntämällä on mahdollista saada reaaliaikaista tietoa kaupungin liikenneverkon toiminnasta ja siten parantaa datan avulla liikenteen tilannekuvaa sekä lisätä tilannetietoisuutta. Yhtenä skenaariona on esitetty, että joukkoliikenteen bussit toimisivat eräänlaisina kaupungin liikenneverkon liikkuvina sensoreina ja näin muodostaisivat koko kaupungin kattavan sensoriverkoston. Bussien lähettämästä datasta voitaisiin havaita poikkeamia normaalista liikennetilanteesta. Poikkeavia tilanteita voisivat olla muun muassa liikenneonnettomuudet, bussien rikkoutumiset, poikkeukselliset säätilanteet, suurten yleisötapahtumien aiheuttamat liikenneruuhkat ja -järjestelyt ja niin edelleen.

Diplomityön luvussa 2 käydään läpi graafien ja matriisien perusteoriaa sekä -käsitteistöä. Kyseisessä luvussa esitellään myös graafien yleisimmät tietorakennetoteutukset. Luvun 3 alkuosassa tutustutaan SIRI-standardiin sekä Tampereen joukkoliikenteen bussien SIRI-dataan. Luvun loppuosassa kuvataan tietorakenteen testaamiseen käytettävän datan kerääminen. Luvussa 4 käydään läpi tietorakenteen suunnittelu ja toteutus sekä kuvataan algoritmien avulla testidatan prosessointi ja tietorakenteeseen tallentaminen. Luvussa 5 esitetään työn tulokset, jotka koostuvat pääosin tietorakenteeseen tallennetusta bussien ajoaikadatasta tuotetuista datamatriiseista sekä ajoaikahistogrammeista.

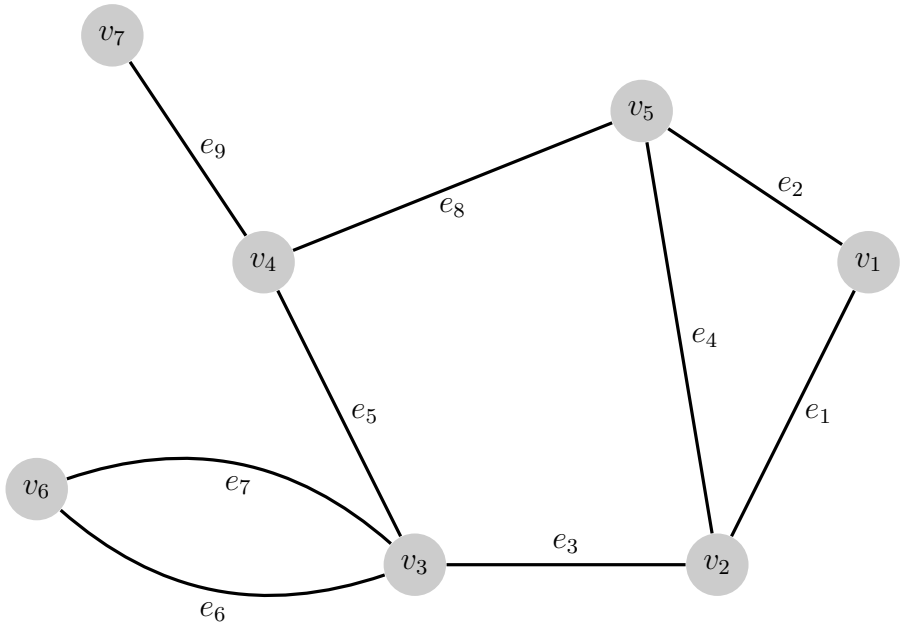
2. TEOREETTINEN TAUSTA

2.1 Graafi

Graafeja käsittelevä alaluku pohjautuu teoksiin Ruohonen (2013) *Graafiteoria* [5], Bapat (2010) *Graphs and Matrices* [6], sekä Thulasiraman, Swamy (1992) *Graphs: Theory and Algorithms* [7].

Formaalisti graafi G koostuu kahdesta äärellisestä joukosta: solmujen joukosta $V(G)$ sekä kaarien joukosta $E(G)$. Graafissa $G = (V, E)$, missä $V = \{v_1, ..., v_n\}$ ja $E = \{e_1, ..., e_m\}$, on näin ollen n solmua sekä m kaarta. Olkoon e_k kaari, jonka päätesolmut ovat v_i ja v_j. Tällöin kaarta merkitään solmujen parina $e_k = (v_i, v_j)$. Suuntaamattomassa graafissa kaaret (v_i, v_j) ja (v_j, v_i) ovat samat. Seuraavassa muutamia graafeihin liittyviä määritelmiä.

- Kaari $e_k = (v_i, v_i)$ on *silmukka*.
- Kaaret, joilla on samat päätepisteet, eli $e_k = (v_i, v_j)$ ja $e_l = (v_i, v_j)$ ovat *rinnakkaiset*.
- Kaaret, ovat *vierekkäiset*, jos niillä on yhteinen päätepiste.
- Solmut v_i ja v_j ovat *vierekkäiset*, jos niitä yhdistää kaari, eli (v_i, v_j) on kaari.
- Solmun v_i *aste* $d(v_i)$ on solmuun liittyneiden kaarien lukumäärä.
- Graafia, jossa ei ole silmukoita eikä rinnakkaisia kaaria nimitetään *yksinkertaiseksi graafiksi*.
- *Tyhjässä graafissa* $E(G) = \emptyset$ ei ole lainkaan kaaria.
- *Nollagraafissa* $V(G) = \emptyset$ ja $E(G) = \emptyset$ ei ole lainkaan solmuja.
- Mikäli graafi koostuu vain yhdestä solmusta, on graafi *triviaali*.



Kuva 2.1: Suuntaamaton graafi

Esimerkki 2.1.1. Kuvassa 2.1 on esitetty esimerkki suuntaamattomasta graafista. Graafissa on seitsemän solmua sekä yhdeksän kaarta. Graafin solmujen joukko $V = \{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$ ja graafin kaarien joukko $E = \{e_1, e_2, e_3, e_4, e_5, e_6, e_7, e_8, e_9\}$. Graafi ei ole yksinkertainen, sillä kaaret e_6 ja e_7 ovat rinnakkaiset.

2.1.1 Kulku, polku ja graafin yhtenäisyys

Graafin $G = (V, E)$ *kulku* W on äärellinen solmujen ja kaarien vuorotteleva jono $W = v_{i_0}, e_{j_1}, v_{i_1}, e_{j_2}, ...v_{j_{k-1}}, e_{j_k}, v_{i_k}$, missä $v_{i_{t-1}}$ ja v_{i_t} ovat kaaren e_{j_t} päätepisteet $(t = 1, ..., k)$. Kulku on *reitti*, jos kulun jokainen kaari esiintyy vain kerran. Kulku on *polku*, jos kulun jokainen solmu ja kaari esiintyvät vain kerran lukuun ottamatta päätesolmuja, jos ne ovat samat. Polku-käsitteen kautta voidaan määritellä graafin yhtenäisyys: Graafi $G = (V, E)$ on *yhtenäinen*, jos sen kaikkien solmuparien (v_i, v_j) välillä on polku.

Esimerkki 2.1.2. Kuvan 2.1 graafissa

$$v_1, e_2, v_5, e_8, v_4, e_9, v_7, e_9, v_4, e_5, v_3, e_6, v_6, e_7, v_3, e_3, v_2$$

on kulku,

$$v_1, e_2, v_5, e_8, v_4, e_5, v_3, e_6, v_6, e_7, v_3, e_3, v_2$$

on reitti,

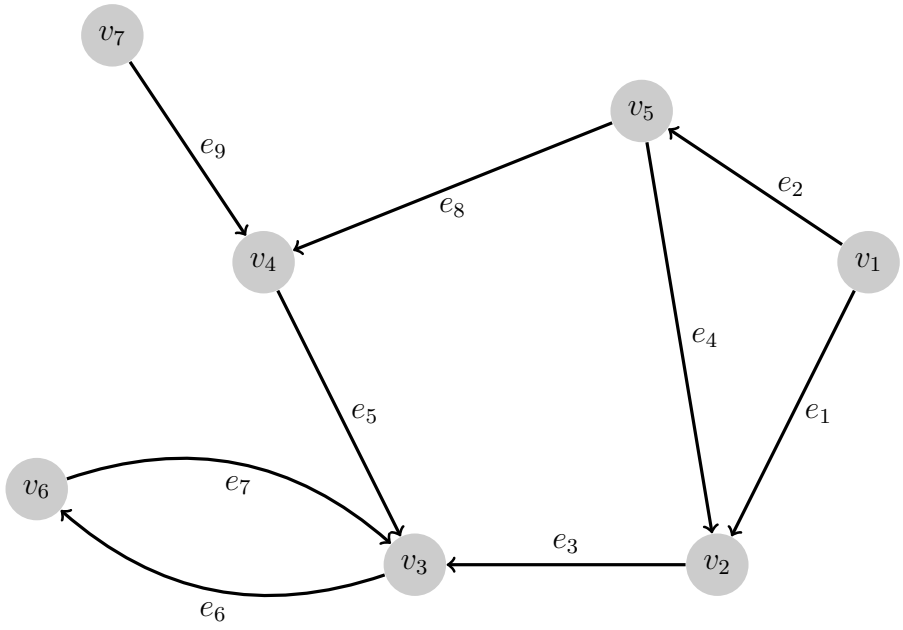
$$v_1, e_2, v_5, e_8, v_4, e_5, v_3, e_3, v_2, e_1, v_1$$

on polku. Graafi on yhtenäinen, sillä sen kaikkien solmuparien välille voidaan muodostaa polku.

2.1.2 Suunnattu graafi

Suunnattu graafi eli *digraafi* on graafi $G = (V, E)$, jonka solmuja yhdistävillä kaarilla on suunta. Näin ollen kaarien joukon $E(G)$ alkiot muodostuvat järjestetyistä pareista, jolloin kaari $e_k = (v_i, v_j)$ tarkoittaa nuolta solmusta v_i solmuun v_j ja kaari $e_l = (v_k, v_i)$ nuolta v_k:sta v_i:hun. Kaaressa $e_k = (v_i, v_j)$ v_i on *alkupiste* ja v_j on *loppupiste*.

- Suunnatun graafin $G = (V, E)$ *alusgraafi* on vastaava "tavallinen" suuntaamaton graafi.
- Alusgraafien kautta suuntaamattomien graafien käsitteet siirtyvät suunnatuille graafeille.
- Suunnatun graafin solmun v_i *tuloaste* $d^-(v_i)$ on solmuun tulevien kaarien lukumäärä ja vastaavasti *lähtöaste* $d^+(v_i)$ on solmusta lähtevien kaarien lukumäärä.



Kuva 2.2: Suunnattu graafi

Esimerkki 2.1.3. Kuvassa 2.2 on esitetty esimerkki suunnatusta graafista. Kyseisen graafin alusgraafi on kuvan 2.1 suuntaamaton graafi. Koska solmuun v_3 tulee kolme kaarta ja siitä lähtee yksi kaari sen tuloaste $d^-(v_3) = 3$ ja lähtöaste $d^+(v_3) = 1$. Vastaavasti solmulle v_7 pätee $d^-(v_7) = 0$ ja $d^+(v_7) = 1$.

2.2 Matriisi

Matriiseja esittelevässä alaluvussa lähteenä on käytetty niin ikään Bapatin kirjaa [6], sekä lisäksi teoksia Gross, Yellen (2006) *Graph Theory and Its Applications* [8], ja Pohjolainen (2008) *Matriisilaskenta* [9].

Matriisi on taulukko, joka sisältää alkioita. $m \times n$ -matriisissa on m vaakariviä ja n saraketta. Matriisin A vaakarivillä i sarakkeessa j olevaan alkioon viitataan notaatiolla a_{ij}.

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix} = (a_{ij})_{m \times n}$$

- Lukuja n ja m kutsutaan myös matriisin A *leveydeksi* ja *korkeudeksi*.
- Jos $n = 1$, niin A on $m \times 1$ -matriisi ja A:ta sanotaan *m-ulotteiseksi pystyvektoriksi*.
- Jos $m = 1$, niin A on $1 \times n$ -matriisi ja A:ta sanotaan *n-ulotteiseksi vaakavektoriksi*.
- Jos $m = n$, niin A on *neliömatriisi*.
- Matriisi, jonka alkioista suurin osa on nollia, on *harva matriisi*.

Olkoon $G = (V, E)$ suunnattu yksinkertainen graafi, jonka solmut ovat $V = \{v_1, ..., v_n\}$ ja kaaret $E = \{e_1, ..., e_m\}$. Graafin G *vieruspistematriisi* $H = (h_{ij})_{n \times n}$ on $n \times n$-matriisi, jonka alkio h_{ij} määritellään seuraavasti:

$$h_{ij} = \begin{cases} 1, & \text{jos } v_i\text{:stä on kaari } v_j\text{:hin} \\ 0, & \text{muulloin} \end{cases}$$

Suuntaamattoman yksinkertaisen graafin tapauksessa $h_{ij} = 1$, jos v_i:n ja v_j:n välillä on kaari.

Esimerkki 2.2.1. Kuvan 2.2 suunnatun graafin vieruspistematriisi on

$$\mathbf{H} = \begin{bmatrix} 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

2.3 Graafin tietorakennetoteutus

Graafin tietorakennetoteutus -osuus tukeutuu teoksiin Kokkarinen, Ala-Mutka (2000) *Tietorakenteet ja algoritmit* [10], sekä Cormen, Leiserson, Rivest (1990) *Introduction to Algorithms* [11].

Graafin tietorakennetoteutuksessa käytetään yleensä kahta vaihtoehtoista tapaa. Graafin *vierekkyyslistaesityksessä* graafin solmuja kuvaavat tietueet on liitetty toisiinsa linkitettynä listana, jolloin linkitetyn listan osoittimet ovat graafin kaaritietueita. Graafin *kytkentämatriisiesityksessä* graafi, jossa on n kappaletta solmuja, mallinnetaan $n \times n$ taulukkona tai matriisina, jonka alkiot vastaavat kaaritietueita.

Vierekkyyslistaesityksessä graafin jokaista solmua kohti muodostetaan yhteen suuntaan linkitetty lista. Graafista, jossa on n-kappaletta solmuja muodostuu siis n-kappaletta linkitettyjä listoja. Graafin jokaista solmua vastaa tietue, johon on linkitetty lista solmusta lähtevien kaarien päässä olevista solmuista. Kaaritietueen tulee sisältää osoitin sekä alku- että loppusolmuun. Vierekkyyslistaesitystä suositellaan käytettäväksi esimerkiksi silloin, jos mallinnettava graafi on harva, eli solmuista lähtee vähän kaaria. Vierekkyyslistaesitys mahdollistaa myös graafissa olevat rinnakkaiset kaaret.

Kytkentämatriisiesityksen edellytys on, että graafin solmut on numeroitu kokonaisluvuilla tai vaihtoehtoisesti identifioitu jollakin muulla yksikäsitteisellä tavalla. Kytkentämatriisin alkiot ovat kaaritietueita. Jos solmujen i ja j välillä on kaari, tieto kaaresta löytyy matriisin alkiosta $A(i, j)$. Olemattomia kaaria varten tarvitaan jokin erityisarvo, jotta tunnistetaan, että kyseisten solmujen välillä ei ole kaarta. Mikäli kytkentämatriisiesityksessä solmujen välisiä kaaria kuvataan totuusarvolla 1 ja olematonta kaarta totuusarvolla 0, on kytkentämatriisiesityksen määritelmä yhtenevä edellisen kohdan vieruspistematriisin kanssa. Kytkentämatriisiesitystä käytetään yleensä tiheiden graafien tapauksissa, eli silloin kuin graafin solmuista lähtee paljon kaaria. Kytkentämatriisista on yksinkertaista tarkastaa, onko kahden annetun solmun välillä kaari. Toisin kuin vierekkyyslistaesitys, kytkentämatriisiesitys ei salli mallinnettavassa graafissa rinnakkaisia kaaria.

Molemmissa toteutuksissa sekä solmu- että kaaritietueisiin voidaan lisätä sovelluksessa tarvittavaa informaatiota. Esimerkiksi mallinnettaessa lentoreittejä, lentokenttiä esittävät solmutietueet voivat sisältää lentokentän nimen ja sijainnin. Tällöin lentokenttien (solmujen) välisiä lentoa kuvaavaan kaaritietueeseen liittyviä tietoja ovat lennon numero ja lennon pituus.

Esimerkki 2.3.1. Kuvan 2.2 suunnatun graafin kytkentämatriisiesitys on

$$\mathbf{K} = \begin{bmatrix} 0 & \mathrm{e}_1 & 0 & 0 & \mathrm{e}_2 & 0 & 0 \\ 0 & 0 & \mathrm{e}_3 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathrm{e}_6 & 0 \\ 0 & 0 & \mathrm{e}_5 & 0 & 0 & 0 & 0 \\ 0 & \mathrm{e}_4 & 0 & \mathrm{e}_8 & 0 & 0 & 0 \\ 0 & 0 & \mathrm{e}_7 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathrm{e}_9 & 0 & 0 & 0 \end{bmatrix}$$

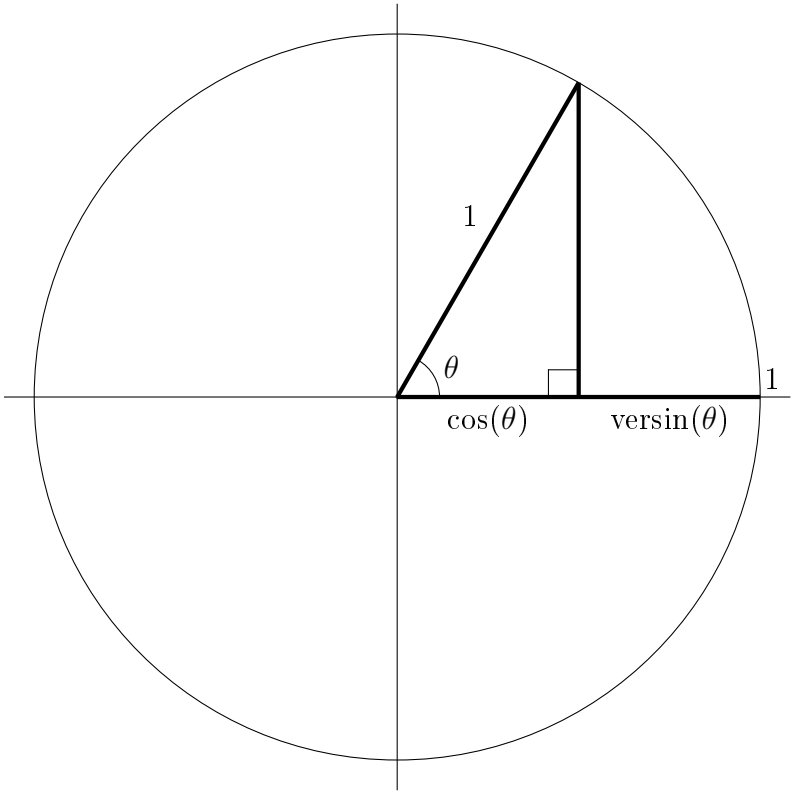
Olemattomat kaaret on tässä esimerkissä kuvattu luvulla 0.

2.4 Haversinen kaava

Haversinen kaavan on ensimmäisen kerran julkaissut Sinnott (1984) [12] *Sky and Telescope* -lehden julkaisussa *Virtues of the Haversine*.

Haversinen kaavaa käytetään pallon pinnalla olevien koordinaattipisteiden välisen etäisyyden laskemiseen. Yksikköympyrän suorakulmaisen kolmion trigonometrinen funktio versini (engl. versed sine) on kuvan 2.3 mukaisesti

$$\text{versin}(\theta) = 1 - \cos(\theta)$$



Kuva 2.3: Yksikköympyrän suorakulmainen kolmio.

Haversini (engl. half the versine) on nimensä mukaisesti puolet versinistä

$$\text{haversin}(\theta) = \frac{\text{versin}(\theta)}{2} = \frac{1 - \cos(\theta)}{2} = \sin^2\left(\frac{\theta}{2}\right)$$

Olkoon (ϕ_1, λ_1) ja (ϕ_2, λ_2) maapallon pinnalla olevien pisteiden leveys- ja pituuskoordinaatit, R maapallon säde sekä d koordinaattipisteiden välinen etäisyys. Tällöin pisteiden välisen keskuskulman haversini on

$$\text{haversin}\left(\frac{d}{R}\right) = \text{haversin}(\phi_2 - \phi_1) + \cos(\phi_1)\cos(\phi_2)\text{haversin}(\lambda_2 - \lambda_1)$$

Edellisestä ratkaistaan d käyttäen haversinin käänteisfunktiota haversin^{-1}

$$d = R \text{ haversin}^{-1}(\text{haversin}(\phi_2 - \phi_1) + \cos(\phi_1)\cos(\phi_2)\text{haversin}(\lambda_2 - \lambda_1)),$$

josta edelleen käyttämällä yhteyttä

$$\text{haversin}^{-1}(x) = 2\arcsin(\sqrt{x})$$

saadaan

$$\begin{aligned} d &= 2R\arcsin\left(\sqrt{(\text{haversin}(\phi_2 - \phi_1) + \cos(\phi_1)\cos(\phi_2)\text{haversin}(\lambda_2 - \lambda_1)}\right) \\ &= 2R\arcsin\left(\sqrt{(\sin^2\left(\frac{\phi_2 - \phi_1}{2}\right) + \cos(\phi_1)\cos(\phi_2)\sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right) \end{aligned} \tag{2.1}$$

Sovellettaessa Haversinen kaavaa maapallon pinnalla olevien koordinaattipisteiden väliseen etäisyyden mittaamiseen, on syytä huomioida että kaavan antama tulos on approksimaatio. Maapallo on muodoltaan ellipsoidi, eli se on hieman litistynyt navoiltaan. Tästä seuraa, että maapallon säde vaihtelee napojen 6356,755 kilometristä päiväntasaajan 6378,140 kilometriin [13, s. 112]. Maapallon muodosta johtuen myöskään maan kaarevuussäde ei ole vakio. Haversinen kaava ei myöskään huomioi maanpinnan muotoja, esimerkiksi mäkiä. Edellä mainituista seikoista aiheutuvan virheen on arvioitu olevan yleensä alle 0,3 prosenttia ja suurimmillaan 0,55 prosenttia pisteiden välisestä etäisyydestä [14].

3. LIIKENNEDATA

3.1 Liikennedatan taustaa

Laajamittainen datan kerääminen liikenteestä alkoi yleistyä 2000-luvun alussa. Tyypillisiä datankeräysmenetelmiä olivat aiemmin tiehen upotettavat induktiosilmukkailmaisimet sekä staattiset tienvarsikamerat. Matkapuhelimien kuluttajamäärien kasvettua datan keräämiseen ja liikenteen ennustamiseen ryhdyttiin käyttämään muun muassa matkapuhelinverkon solurakennetta hyödyntävää tekniikkaa. Menetelmän tausta-ajatuksena oli, että pääteiden liikennevirtoja voisi mallintaa keräämällä dataa matkapuhelinverkossa meneillään olevien puheluiden siirtymisestä soluverkon solusta toiseen. [15]

Matkapuhelinten tekniikan kehittyessä sekä muiden kannettavien elektronisten verkkoyhteydellä varustettujen laitteiden yleistyessä alettiin kiinnostua laajamittaisesta reaaliaikaisen liikennedatan tuottamisesta. Indianapoliksen kaupungissa Yhdysvalloissa kokeiltiin kannettavien elektronisten laitteiden MAC-osoitteisiin (Media Access Control) perustuvaa datan tallentamista Bluetooth-teknologiaa hyödyntäen. Liikkuvissa ajoneuvoissa olevien laitteiden MAC-osoitteiden tallentaminen perustui tien varteen sijoitettaviin antennikaappeihin. [16]

2010-luvun vaihteessa GPS-tekniikan kehittyminen sekä GPS-vastaanottimien nopea yleistyminen älypuhelimissa, navigaattoreissa sekä muissa mobiililaitteissa on avannut valtavasti uusia yksittäisiä datalähteitä. Dataa on perinteisesti ollut saatavilla vain tietyiltä, tarkasti ennalta määritellyiltä alueilta. Korkeat kustannukset ovat rajoittaneet huoltoa ja ylläpitoa vaativien, teiden varsiin tai autoihin sijoitettavien erillisten datankeräysjärjestelmien yleistymistä, eikä dataa ole näin ollen saatu kerättyä laajasti ja suuria määriä. Ihmisten ja ajoneuvojen mukana kaikkialla liikkuvien GPS-yhteydellä varustettujen laitteiden tuottama data on poistanut edellä mainittuja ongelmia. [17]

Liikennedatan saatavuuden ja määrän lisääntyminen on lisännyt kiinnostusta liikennevirtojen laajempaan tutkimiseen ja tätä kautta myös ennustamiseen. Etenkin suuret yksityiset teknologiayritykset, esimerkiksi Google ja Nokia, ovat alkaneet jakamaan navigointipalveluissaan yksityiskohtaista reaaliaikaista tietoa liikenteen sujuvuudesta sekä julkaisemaan liikenne-ennusteita. Viime aikoina myös julkisyhteisöt ovat alkaneet tuottaa ja julkaista dataa liikenteestä. Osana kaupungin tietoaineistojen julkaisemista Tampereen kaupunki avasi syyskuussa 2013 Tampereen joukkolii-

kenteen reaaliaikaisen SIRI-rajapinnan jo aiemmin julkisina olleiden bussien reitti- ja pysäkkiaikataulujen rinnalle.

3.2 SIRI

SIRI on CEN-hyväksytty (Comité Européen de Normalisation) rajapintastandardi julkisen liikenteen aikataulujen, ajoneuvojen ja yhteyksien reaaliaikaisten tietojen välittämiseen joukkoliikennetoimijoiden välillä. SIRI on kehitetty kahdeksan eri Euroopan maan asiantuntijoiden yhteistyönä ja päätoimijoina taustalla ovat olleet Saksan, Ranskan ja Iso-Britannian liikenneministeriöt. SIRI tarjoaa mahdollisuuden muun muassa yksittäisten ajoneuvojen reaaliaikaiseen GPS-pohjaiseen seurantaan, julkisten liikennevälineiden aikataulunäyttöjen ohjaukseen, reaaliaikaisen liikennetiedon tuomisen mobiililaitteille sekä vaihdollisten liikenneyhteyksien hallinnan.
SIRI on XML-pohjainen (Extensible Markup Language), eli se käyttää XML-muotoa tiedon välittämiseen. [18]

Yksi SIRI:n tarjoamista toiminnallisista moduuleista on Vehicle Monitoring Service (VM). Se antaa tietoa seurattavasta liikennevälineestä, kulkuneuvon reaaliaikaisesta sijainnista sekä vertaa liikennevälineen todellista sijaintia aikataulun mukaiseen oletettuun sijaintiin. Taulukossa 3.1 on esitetty SIRI:n VM-palvelun tuottamaa dataa.

Taulukko 3.1: SIRI:n VM-palvelun datarivejä

RecordedAtTime	LineRef	DirectionRef	DatedVehicle JourneyRef	VehicleLocation Longitude	VehicleLocation Latitude	Bearing	Delay	VehicleRef
1381821487001	16	2	0940	23.813181	61.502863	265.0	27	TKL_261
1381830832015	25	2	1240	23.749879	61.497546	79.0	98	OB_13001
1381835316011	10	2	1405	23.734823	61.496994	155.0	8	TKL_265
1381837512002	30	1	1430	23.846124	61.458428	0.0	15	TKL_009
1381841524003	31	1	1530	23.755170	61.497358	255.0	73	TKL_654

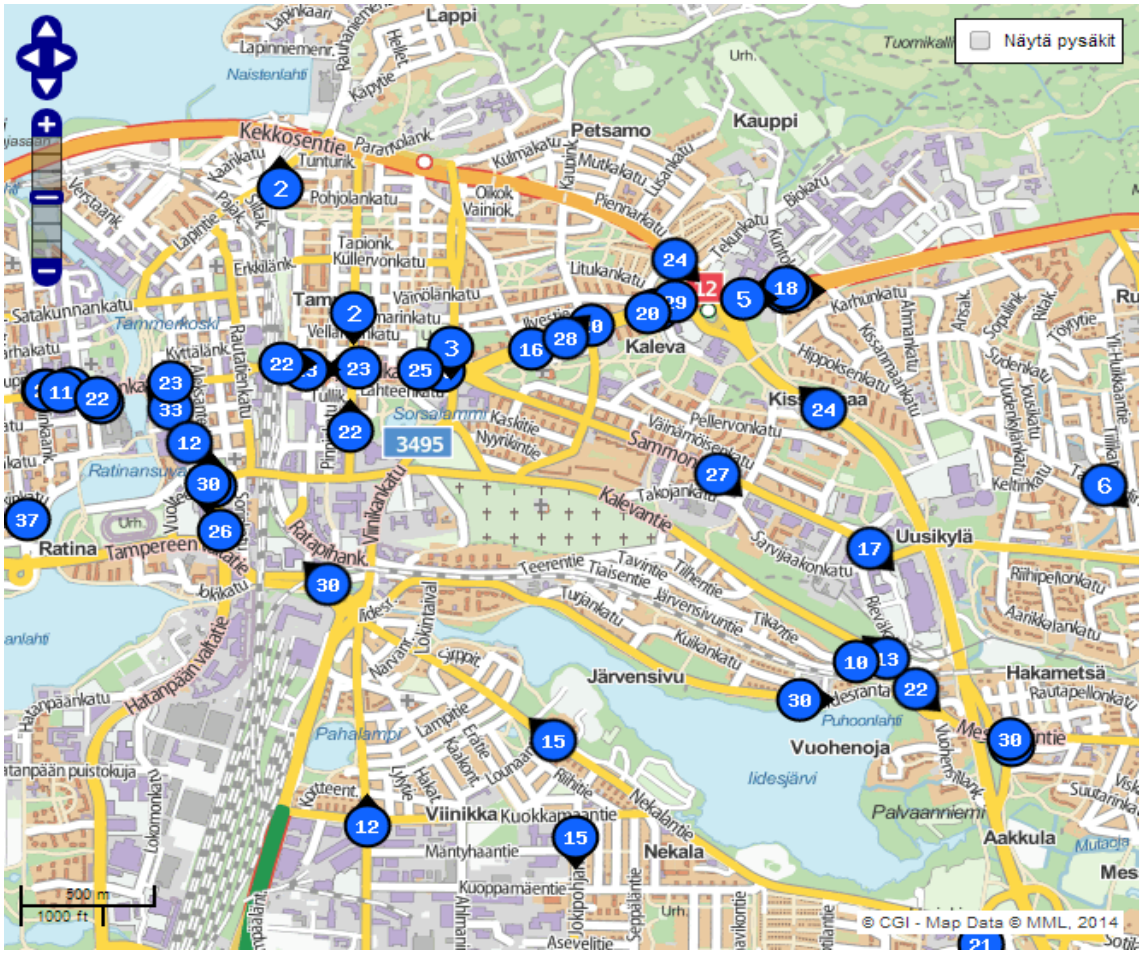
Aiemmissa SIRI-dataa sivuavissa tutkimuksissa on käytetty Dublinin kaupungin julkisen liikenteen busseista kerättyä SIRI-dataa. IBM Dublin Research Centerin sekä Dublin kaupungin tekemässä yhteistutkimuksessa [19] kehitettiin ITS-ratkaisua (Intelligent Transportation Systems) keräämään liikennedataa eri lähteistä ja tuottamaan datan pohjalta muun muassa reaaliaikaista tietoa liikennevirroista sekä tilastollista analyysia liikenteen sujuvuudesta. Tutkimuksessa käytettiin SIRI-dataa järjestelmän testaamiseen. Tutkimusdatana toimi noin tuhannen Dublinin julkisen

liikenteen bussin tuottamaa SIRI-data. Datapakettia kerättäessä näytteitä oli tallennettu keskimäärin 20 sekunnin välein.

Toisessa niin ikään IBM Research Centerin tekemässä tutkimuksessa [20] SIRI-dataa hyödynnettiin Dublinin kaupungin liikenneruuhkien diagnosoimisessa. SIRI-muodossa ollut Dublinin kaupungin bussien tuottama paikkatietodata oli yksi useista datalähteistä, jota käytettiin kaupungin liikennetilannetta käsittelevän mallin kehittämisessä.

3.3 Tampereen joukkoliikenteen SIRI-data

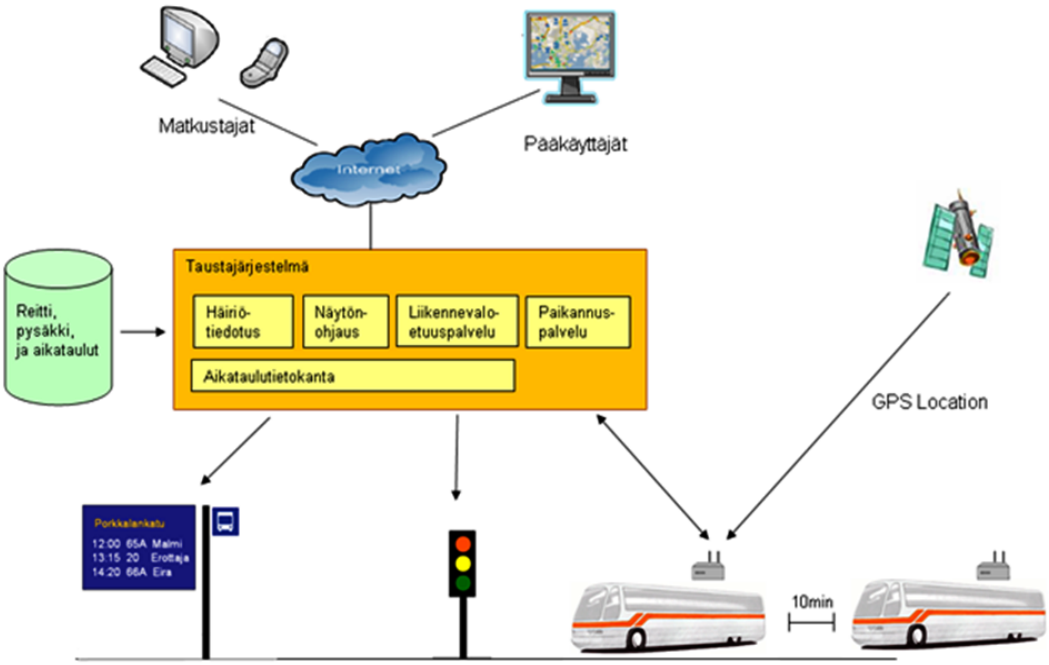
Tampereen joukkoliikenteen reaaliaikainen rajapinta avattiin syyskuussa 2013. Tämä tarkoittaa, että Tampereen joukkoliikenteen bussien reaaliaikaiset sijaintitiedot ovat avoimesti kaikkien saatavilla SIRI-formaatissa. Tampereen joukkoliikenne on jo aiemmin avannut bussien reitti- ja pysäkkiaikataulutietoja tarjoavan rajapinnan. [21]



Kuva 3.1: Tampereen joukkoliikenteen Lissu Liikenteenseuranta -palvelu. Siniset ympyrät kuvaavat kaupungissa liikennöiviä busseja.

Eräänä liikennedatan avaamisen tavoitteena Tampereen joukkoliikenteen sivuilla mainitaan: ”Rajapinnan avaus mahdollistaa entistä parempien sähköisten reittioppaiden suunnittelun ja bussien tarkan sijainnin huomioivien sovellusten kehittämisen älylaitteisiin.” [21] Esimerkkinä tästä on kuvassa 3.1 esitetty Tampereen joukkoliikenteen Lissu Liikenteenseuranta -palvelu. Lissu on internet-selaimella käytettävä sovellus, jonka avulla käyttäjä voi seurata busseja reaaliaikaisesti sekä saada päivittyvää tietoa bussin saapumisajasta pysäkille. Kaupallinen Lissu Liikenteenseuranta on otettu käyttöön ennen bussidatan avaamista vuonna 2011, mutta nyt avoimesti saatavilla oleva SIRI-data sisältää saman informaation, jonka perustalle Lissu on kehitetty.

Bussien paikkatietodatan tuottamisen perustana toimii IJ2010-järjestelmä, joka on Tampereen kaupungin tilaama reaaliaikainen matkustajainformaatiojärjestelmä. Järjestelmän toimintaperiaate on esitetty kuvassa 3.2.



Kuva 3.2: IJ2010-järjestelmän järjestelmäkuvaus [22].

Reaaliaikaisen datan tuottava järjestelmä toimii pääpiirteissään seuraavasti. Ajoneuvolaitteella varustettu kaupunkiliikenteen bussi lähettää 3G-datayhteydellä GPS-sijaintitietonsa IJ2010-informaatiojärjestelmän taustajärjestelmälle yhden sekunnin välein. Taustajärjestelmä yhdistää bussilta tulleen reaaliaikaisen sijaintitiedon sekä järjestelmässä olevan staattisen aikataulu- ja reittidatan. Näiden tietojen pohjalta taustajärjestelmä laskee arvion bussin etenemisestä aikataulun mukaisesti. Bussin sijaintitiedon sekä aikatauluennusteen lisäksi dataan yhdistetään muun muassa bussin

identifiointi- ja reittitietoja siten, että data on SIRI-standardin VM-moduulin mukaista. Data välitetään käyttäjärajapintaan, joka päivittyy yhden sekunnin välein. Rajapinta siis tarjoaa ideaalitilanteessa kaikkien sillä hetkellä liikennöivien bussien SIRI-datan joka sekunti. [22; 23]

Kuvassa 3.3 on SIRI-standardin mukainen bussidataa sisältävä XML-kutsun vastaus, joka on kerätty linjalla 29 liikennöivästä TKL:n (Tampereen Kaupunkiliikenne Liikelaitos) bussista.


```
{
  "Siri" : {
    "ServiceDelivery" : {
      "ResponseTimestamp" : 1396334975186,
      "ProducerRef" : {
        "value" : "IJ2010"
      },
      "Status" : true,
      "MoreData" : false,
      "VehicleMonitoringDelivery" : [ {
        "VehicleActivity" : [ {
          "ValidUntilTime" : 1396335005014,
          "MonitoredVehicleJourney" : {
            "LineRef" : {
              "value" : "29"
            },
            "DirectionRef" : {
              "value" : "1"
            },
            "FramedVehicleJourneyRef" : {
              "DataFrameRef" : {
                "value" : "2014-04-01"
              },
              "DatedVehicleJourneyRef" : "0910"
            },
            "OperatorRef" : {
              "value" : "TKL"
            },
            "OriginName" : {
              "value" : "Aitolahdentie 36",
              "lang" : "fi"
            },
            "DestinationName" : {
              "value" : "Korvenkatu 44",
              "lang" : "fi"
            },
            "Monitored" : true,
            "VehicleLocation" : {
              "Longitude" : 23.6703413,
              "Latitude" : 61.5082373
            },
            "Bearing" : 256.0,
            "Delay" : "P0Y0M0DT0H1M34.000S",
            "VehicleRef" : {
              "value" : "TKL_276"
            }
          },
          "RecordedAtTime" : 1396334975014
        } ],
        "version" : "1.3",
        "ResponseTimestamp" : 1396334975186,
        "Status" : true
      } ]
    },
    "version" : "1.3"
  }
}
```


Kuva 3.3: Yhden bussin SIRI-data tietyltä ajanhetkeltä

Seuraavassa on selitetty kuvan 3.3 SIRI-datan datakenttien sisältöä.

- *LineRef* määrittää bussin ajaman linjan linjanumeron.
- *DirectionRef* määrittää kumpaan suuntaan bussi on ajamassa linjalla. Mikäli arvo on 1, bussi ajaa erikseen määritellyltä alkupysäkiltä kohti päätepysäkkiä. Jos arvo on 2, bussin suunta on päätepysäkiltä alkupysäkille.
- *DataFrameRef* määrittää päivämäärän, jolloin bussi aloitti linjan ajamisen joko linjan alku- tai päätepysäkiltä.
- *DatedVehicleJourneyRef* määrittää ajan, jolloin bussi aloitti linjan ajamisen joko linjan alku- tai päätepysäkiltä.
- *OperatorRef* määrittää bussilinjaa liikennöivän operaattorin tunnuksen.
- *OriginName* määrittää bussilinjan ensimmäisen pysäkin nimen.
- *DestinationName* määrittää bussilinjan viimeisen pysäkin nimen.
- *Latitude, Longitude* määrittää bussin sijainnin leveys- ja pituuskoordinaatteina.
- *Bearing* määrittää suunnan (suuntakulman), johon bussi on ajamassa. Mikäli bussi ei näytteentallentamishetkellä liiku, on suuntakulman arvo 0.
- *Delay* määrittää bussin etenemisen aikatauluun verraten. Delayn arvo on sekuntimäärä, jonka bussi on edellä tai jäljessä aikataulusta.
- *VehicleRef* määrittää bussiajoneuvon identifioivan tunnuksen.
- *RecordedAtTime* määrittää datan tallennusajankohdan.

Edellä mainittujen lisäksi SIRI-data sisältää myös muun muassa dataa tallentavan palvelimen aikaleiman eli *ResponseTimestamp*-kentän. [24]

Tampereen joukkoliikenteen bussien SIRI-dataan tutustuttaessa havaittiin, että data sisältää niin sanottua roskadataa. Roskadataksi tulkittiin muun muassa kopioitunut data sekä haamudata. Datan ongelmiin perehdyttäessä havaittiin, että jopa 25 prosenttia kerätystä datasta voi olla roskadataa. Dataongelmien syiksi arveltiin sijaintitiedon tuottamiseen tarvittavan GPS-yhteyden katkeamista sekä bussin ja datan tallentavan palvelimen välisen 3G-datayhteden yhteysongelmia. Myös inhimilliset tekijät, esimerkiksi kuljettajan tekemät virheet bussikierron identifioivaan järjestelmään sisään- ja uloskirjauduttaessa, voivat olla syynä roskadatan ilmenemiseen.

3.4 Testidatan kerääminen

Tässä diplomityössä esiteltävän tietorakenteen testaamiseen käytetty SIRI-bussidata on kerätty Tampereen joukkoliikenteen reaaliaikaisesta rajapinnasta. Rajapinta löytyy osoitteesta: http://data.itsfactory.fi/siriaccess/vm/rest. Datan keräämiseen käytettiin noin kuudenkymmenen rivin pituista JSON-komentokielen (JavaScript Object Notation) skriptiä. Skripti kerää rajapinnassa sekunnin välein päivittyvän IJ2010-informaatiojärjestelmän taustajärjestelmän tuottaman bussidatan CSV-tekstitiedostoon (Comma-Separated Values). Skriptin tuottaman CSV-tiedoston jokainen rivi sisältää yhden bussin tietyn ajanhetken SIRI-datan. Datarivi koostuu yhteensä 21:stä pilkuilla erotetusta datakentästä, joiden sisältöä on esitelty kohdassa 3.3.

Dataa kerättiin yhteensä kuudelta päivältä - viisi päivää tietorakenteen testidataksi sekä lisäksi yksi vertailupäivä, esimerkiksi mahdollisten data-analyysin pohjalta tuotettujen ajoaikaennusteiden toteutumien vertailemiseksi. Taulukossa 3.2 on esitetty päivämäärät, jolloin dataa kerättiin, kyseisten päivien säätiedot, prosessoimattomien datatiedostojen koot sekä datarivien määrät kultakin datankeräyspäivältä.

Taulukko 3.2: Kerättyyn testidataan liittyviä tietoja.

Päivämäärä	Datan koko levyllä (GB)	Datarivien määrä	Sää
Ke 22.1.2014	1,21	5 025 900	Selkeää, lämpötila -14 °C – -22 °C
To 23.1.2014	1,86	7 683 000	Selkeää, lämpötila -14 °C – -21 °C
Pe 24.1.2014	2,08	8 580 900	Puolipilvistä, lämpötila -16 °C – -21 °C
Ma 27.1.2014	1,75	7 245 200	Pilvistä, lämpötila -9 °C – -12 °C
Ti 28.1.2014	1,20	4 966 200	Puolipilvistä, lämpötila -11 °C – -16 °C

Bussidataa kerättiin peräkkäisiltä arkipäiviltä. Arkipäivien väliin sijoittuva viikonloppu rajattiin pois datan keräyksestä, koska datan analysointia varten haluttiin bussivuorojen kannalta mahdollisimman samanlaisia päiviä. Kaikki bussilinjat eivät liikennöi viikonloppuisin ja viikonloppuisin liikennöivillä linjoilla bussilähtöjen määrä poikkeaa selvästi arkipäivien liikenteestä. Myös arkipäivien välillä liikenteessä on pieniä eroavaisuuksia, esimerkiksi perjantain ja lauantain välisen yön liikenne poikkeaa muista öistä. Datankeräyspäivien säätiedot tallennettiin, jotta mahdolliset sään ja ajokelin aiheuttamat muutokset bussiliikenteeseen voitaisiin huomioida dataa ja tuloksia analysoidessa.

Taulukossa 3.2 esitettävät vuorokausidatat kerättiin siten, että datan keräämisen suorittava skripti käynnistettiin ajastetusti kello 03.00 yöllä ja keskeytettiin ajastetusti 23 tunnin ja 45 minuutin kuluttua kello 02.45 seuraavana yönä. Näin siis kunkin vuorokauden data saatiin tallennettua erilliseen CSV-tiedostoon. Kello 02.45 muodostui luontevaksi datankeruun katkaisemisajaksi, koska Tampereen joukkoliikenteen aikataulutietojen perusteella arkipäivien ensimmäiset bussilähdöt alkavat kello 03.30 jälkeen ja viimeiset vuorot päättyvät noin kello 01.30 lukuun ottamatta perjantain ja lauantain välistä yötä [25]. Myös datan jatkokäytön kannalta oli hyödyllistä, että yhden valtavan datapaketin sijaan kunkin vuorokauden data oli omassa datatiedostossaan.

Datan tallentamisen jälkeen raakadataa prosessoitiin vielä siten, että vuorokauden bussidatan sisältävät CSV-tiedostot jaettiin 7-ZIP -nimisellä ohjelmalla 500 megatavun kokoisiksi tiedostoiksi. Tiedostokokoa pienennettiin, koska datan prosessointiin käytetyssä Matlab-ohjelmassa havaittiin tietokoneen keskusmuistin loppumisesta aiheutuvaa hidastumista ja jumiutumista, mikäli käsitellyt CSV-tiedostot olivat liian suuria.

4. MENETELMÄT

4.1 Tietorakenteen suunnittelu

Tyypillisenä lähestymistapana kaupungin bussiverkoston mallintamiseen voidaan pitää bussilinjojen tai yksittäisten ajoneuvojen tarkastelua. Tässä työssä esiteltävän tietorakenteen suunnittelussa päädyttiin tarkastelemaan bussiliikennettä spatiaalisesta näkökulmasta. Tietyn bussilinjan tai linjanlähdön sijasta ollaankin kiinnostuneita siitä, mitä busseille ja liikenteelle tapahtuu jollakin katuosuudella tai esimerkiksi kaksien liikennevalojen välillä.

Bussidatan tallentamiseen soveltuvan tietorakenteen suunnittelun edetessä heräsi ajatus kaupungin mallintamisesta graafiksi. Graafin solmuiksi päätettiin valita bussiliikenteen kannalta mielenkiintoisia kohtia, joita ovat esimerkiksi liikennevalot, bussipysäkit sekä risteykset, joissa bussilinjoja yhdistyy tai eroaa. Spatiaalisiksi alkioiksi muodostuivat solmuja yhdistävät kaaret, joita käytännössä ovat mielenkiintoisia pisteitä yhdistävät kadun- ja tienpätkät. Tausta-ajatuksena oli koko ajan, että kaari- ja solmutietoalkioihin voitaisiin tallentaa busseista saatua SIRI-dataa esimerkiksi liikenteen analysoimista varten.

Usein saman bussilinjan bussit ajavat samalla kadulla kahteen eri suuntaan, alkupysäkiltä kohti päätepysäkkiä ja päinvastoin. Tämän seurauksena voitaisiin ajatella, että kaupungin katujen mallinnuksen tuloksena muodostuvassa suunnatussa graafissa olisi rinnakkaisia kaaria kuvaamassa samalla kadulla eri suuntiin ajavia saman bussilinjan ajoneuvoja. Tämänkaltaisessa toteutuksessa ongelmia aiheuttaisivat esimerkiksi graafin solmuiksi valitut bussipysäkit. Monissa tapauksissa kadun eri puolien pysäkit ovat eri kohdissa katua. Kaupungeissa on usein myös erillisiä lähinnä bussien käyttöön tarkoitettuja kaistoja itse päätien eri puolilla, kuten Tampereen Teiskontien länsipäässä kuvan 4.1 mukaisesti. Tästä seuraa, että näennäisesti samalla kadulla eri suuntiin ajavat bussit liikennöivät todellisuudessa eri katuja pitkin.

Kuva 4.1: Eri suuntien bussit ajavat kuvan Teiskontiellä uloimmilla ajokastoilla.

Edellä mainituista syistä johtuen yhdeksi tietorakenteen suunnittelun kulmakiveksi muodostui periaate, että saman bussilinjan kahden eri suunnan liikennöintiä ei tallenneta samaan tietorakenteeseen. Myös SIRI-datan bussin ajosuunnan ilmaiseva DirectionRef-kenttä ohjasi suunnittelua bussien kahden eri ajosuunnan erottelemiseen. Kyseinen datakenttä jaottelee kaikkien bussilinjojen kaksi eri ajosuuntaa numeroilla 1 ja 2.

Rinnakkaisten kaarien lisäksi myös muita graafiteorian käsitteitä tulee esille mallinnettaessa kaupungin tieverkostoa suunnatuksi graafiksi. Bussien ajamien bussireittien voidaan ajatella olevan polkuja mallinnuksen lopputuloksena syntyvässä graafissa ja graafi on todennäköisesti yhtenäinen, koska pääsääntöisesti kaikkien solmuparien (katujen ja kadunosien) välille voidaan muodostaa polku. Muodostuvan suunnatun graafin solmujen tuloaste tai lähtöaste on nollaa suurempi, sillä jokaiseen solmuun tulee tai siitä lähtee vähintään yksi kaari.

Yleisimmät tavat graafin tietorakennetoteutukselle ovat vierekkyyslistaesitys sekä kytkentämatriisiesitys. Vierekkyyslistaesitystä suositeltiin käytettäväksi käsitellessä harvoja graafeja, kytkentämatriisiesitystä puolestaan tiheiden graafien tapauksissa. Tässä työssä käsiteltävässä tietorakenteessa päädyttiin käyttämään kytkentämatriisiesitystä, koska tietorakenteen toteutukseen ja testaamiseen käytetty Matlab-ohjelmisto soveltuu erinomaisesti matriisien käsittelyyn ja toisaalta Matlab ei juuri tue listatoteutustapaa verrattuna esimerkiksi C++-ohjelmointikieleen.

Mallinnettaessa kaupungin katuverkostoa graafiksi on lopputulos harva graafi, sillä solmuista lähtee verrattain vähän kaaria. Tällöin myös graafin kytkentämatriisi on harva. Näin käy varsinkin silloin, mikäli katuverkostoa jaetaan osiin risteyksien lisäksi myös esimerkiksi samalla kadulla olevien liikennevalojen ja bussipysäkkien kohdista. Matlab-ohjelmistossa on olemassa harvaa matriisia varten oma tietotyyppi *sparse matrix*. Harvan matriisin tallentaminen sparse matrix -muotoon kuluttaa vähemmän muistia verrattuna tavallisen matriisin tallentamiseen. Lisäksi sparse matrix -muuttujan käsittely on tavallista matriisimuuttujaa nopeampaa. Näin ollen runsaastikin tyhjiä alkioita sisältävän graafin kytkentämatriisin tallentamisen ja käsittelyn ei pitäisi tuottaa laskennallisia ongelmia käytettäessä Matlab-ohjelmaa. Tämän työn tietorakenteen tapauksessa katuverkostoa mallinnettaessa ei käytetty rinnakkaisia kaaria, joten kytkentämatriisi on myös tässä suhteessa soveltuva muodostuvan graafin tietorakennetoteutukseen.

Reaaliaikajärjestelmiä toteutettaessa törmätään usein muisti- ja laskentakapasiteetin rajallisuuteen. Tällöin joudutaan tekemään valinta muistinkäytön ja laskenta-ajan suhteen, koska resurssit eivät riitä kaikkeen. Ongelmaa voidaan yrittää ratkaista karsimalla tallennettavaa datamäärää tai vaihtoehtoisesti keventämällä reaaliaikaista datan prosessointia. Tässä työssä kehitettävää tietorakennetta suunnitellessa ydinkysymyksinä pohdittiin, tallennetaanko tietorakenteeseen kaikki mahdollinen hyödyllinen data ja tehdään analysoivasta taustalaskennasta mahdollisimman kevyttä, vai pyritäänkö datamäärä pitämään mahdollisimman pienenä ja vastaavasti kasvattamaan taustalaskennan osuutta. Koska suunnittelun taustalla oli ajatus reaaliaikaisesta datan hallinnasta, päädyttiin tietorakenteen datasisällöstä tekemään tässä vaiheessa mahdollisimman kevyt. Lisäksi myös analysoiva laskenta siirrettiin datan tietorakenteeseen tallentamisvaiheesta varsinaiseen datan analysointivaiheeseen. Toteutuksessa käytetyt tietotyypit pyrittiin valitsemaan siten, että ne mahdollistavat datan lisäämisen ja poistamisen tietorakenteesta sekä myös uudentyyppisen datasisällön lisäämisen tietorakenteeseen tulevaisuudessa.

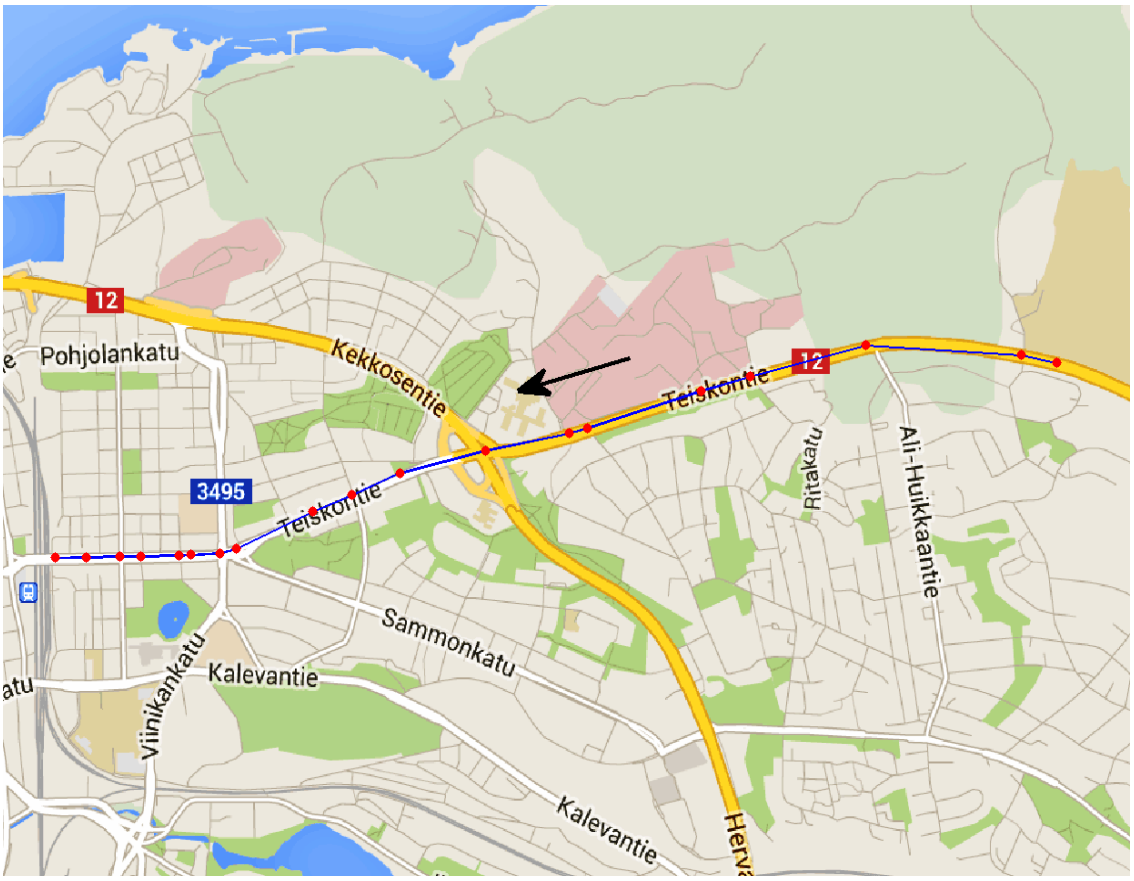
4.2 Tietorakenteen toteutus

Tietorakenteen toteutusalustaksi valittiin Matlab R2013a -ohjelmisto, josta käytettiin 64-bittistä versiota. Sekä graafin solmujen että kaarien tietotyypiksi valikoitui Matlabin *struct*-tietotyyppi. Matlabin structin kenttiin voidaan tallentaa eri tietotyyppiä olevia muuttujia ja struct mahdollistaa myös kenttien dynaamisen lisäämisen jälkikäteen. Solmutietueen kentiksi tallennettiin solmun leveys- ja pituuskoordinaatit (Latitude and Longitude). Kaaritietueen kentiksi valittiin kaaren alkusolmun leveys- ja pituuskoordinaatit (Latitude and Longitude), kaaren loppusolmun leveys- ja pituuskoordinaatit (Latitude and Longitude) sekä kaarta pitkin kulkevien bussilinjojen numerot (LineRef).

Graafin tietorakennetoteutuksen kytkentämatriisi päätettiin esittää neliömatriisina, jonka kaikki alkiot voidaan ensin alustaa nolliksi. Jos kytkentämatriisi on suuri, sen tietotyyppinä voidaan käyttää Matlabin sparse matrix -tietotyyppiä. Tietorakenteen toteutusvaiheessa kytkentämatriisin alkioiksi asetetaan indeksit (kokonaisluvut), joiden avulla solmuja yhdistävät kaaret ovat löydettävissä kaaristructista. Tämä toimenpide kuvataan tarkemmin luvussa 4.4.3. Pitämällä muistissa kaaritietueiden indeksit ja esittämällä ne kytkentämatriisissa kaarien alku- ja loppusolmujen määräämissä kohdissa, vältetään koko kaaristructin läpikäynti solmuja yhdistävän kaaren löytämiseksi. Suuressa tietorakenteessa kaaren etsiminen iteroimalla tietorakenteen läpi kaaren alku- ja loppukoordinaattien avulla voi olla hidasta. Toisaalta tähän työhön valittu toteuttamistapa edellyttää, että kaarien indekseistä pidetään kirjaa ja muutokset indekseissä päivitetään kytkentämatriisiin.

4.3 Testireitti

Tietorakenteen testaamista varten muodostettiin kuvan 4.2 mukainen testireitti Takahuhdin ja Tampereen keskustan itäpuolen välillä.



Kuva 4.2: Testireitillä seurattiin kuvassa olevan nuolen mukaisesti idästä länteen päin ajavia busseja. Testireitistä muodostetun graafin solmut ja kaaret indeksoitiin juoksevalla numeroinnilla itä-länsi suunnan mukaisesti.

Testireitillä seurattiin idästä länteen päin kohti keskustaa ajavia ajoneuvoja, jolloin kyseiseen suuntaan ajavien bussien SIRI-datan DirectionRef-kentän arvo oli 1. Testireitin valintaan vaikutti muun muassa se, että sitä pitkin kulkee vuorokaudessa verrattain paljon busseja sekä eri bussilinjoja. Reitillä on useita bussipysäkkejä sekä liikennevaloja, jotka erityisesti vaikuttavat liikenteen sujuvuuteen. Reittiä valittaessa huomioitiin myös, että kyseisellä välillä SIRI-datan GPS-sijaintitiedot ovat tarkempia verrattuna esimerkiksi Tampereen keskustan GPS-signaalien kannalta osittain haastavaan ympäristöön. SIRI-dataa käsiteltäessä huomattiin, että keskustan Hämeenkadulla SIRI-datasta saadut bussien GPS-sijaintitiedot olivat epätarkkoja verrattuna keskustan laitamiin ja keskustan ulkopuolisiin alueisiin.

Testireitistä muodostettiin 19 solmusta ja 18 kaaresta koostuva graafi. Taulukossa

4.1 on esitetty testireitistä muodostetun graafin solmujen indeksit, solmujen sijainnit sekä lyhyt perustelu, miksi kunkin solmun sijainniksi on valittu juuri kyseinen paikka.

Taulukko 4.1: Testireitistä muodostetun graafin solmuiksi valitut paikat.

Solmu	Sijainti	Solmun valintaperuste
1	Alkusolmu	
2	Teiskontien ja Jaakonmäenkadun risteys	Liikennevalot, pysäkki
3	Teiskontien ja Ali-Huikkaantien risteys	Liikennevalot, pysäkki
4	Teiskontien ja Ritakadun risteys	Liikennevalot, pysäkki
5	Teiskontie, sairaalan kohta	Liikennevalot
6	Teiskontien ja Kissanmaankadun risteys	Liikennevalot, pysäkki
7	Teiskontien ja Kuntokadun risteys	Liikennevalot
8	Teiskontien ja Hervannan valtaväylän risteys	Liikennevalot, pysäkki
9	Teiskontie 25	Pysäkki
10	Teiskontien ja Kaupinkadun risteys	Liikennevalot
11	Teiskontie 13	Pysäkki
12	Teiskontie 1	Liikennevalot
13	Teiskontien ja Kalevan puistotien risteys	Liikennevalot
14	Itsenäisyydenkatu 25	Pysäkki
15	Itsenäisyydenkadun ja Salhojankadun risteys	Liikennevalot
16	Itsenäisyydenkatu 13	Pysäkki
17	Itsenäisyydenkadun ja Yliopistokadun risteys	Liikennevalot
18	Itsenäisyydenkatu 3	Pysäkki
19	Loppusolmu	

Taulukossa 4.2 on esitetty testireitistä muodostetun graafin kaarien tiedot sekä kaarilla kulkevien bussilinjojen linjanumerot. Testitapauksen linjoista rajattiin pois muun muassa yölinjat, esimerkiksi linjat Y23 ja Y17, sekä useilla eri linjatunnuksilla liikennöivä linja 90.

Taulukko 4.2: Testireitistä muodostetun graafin kaarien tiedot.

Kaari	Alkusolmu	Loppusolmu	Kaaren pituus (m)	Kaaren linjat
1	1	2	140	16 18 28
2	2	3	600	16 18 28
3	3	4	450	16 18 28
4	4	5	200	16 18 28
5	5	6	450	16 18 28
6	6	7	72	16 18 28 29
7	7	8	350	16 18 28 29
8	8	9	350	16 18 20 28 29 39
9	9	10	200	16 18 20 28 29 39
10	10	11	170	16 18 20 28 29 39
11	11	12	300	16 18 20 28 29 39
12	12	13	67	16 18 20 28 29 39
13	13	14	120	3 16 17 18 20 25 27 28 29 37 39
14	14	15	44	3 16 17 18 20 25 27 28 29 37 39
15	15	16	140	3 16 17 18 20 23 25 27 28 29 37 39
16	16	17	83	3 16 17 18 20 23 25 27 28 29 37 39
17	17	18	130	2 3 10 13 16 17 18 20 22 23 25 27 28 29 37 39
18	18	19	110	2 3 10 13 16 17 18 20 22 23 25 27 28 29 37 39

Kuvassa 4.3 on esitelty testireitistä muodostetun graafin kytkentämatriisi.

$$\mathbf{K} = \begin{bmatrix}
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 2 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 7 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 9 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 11 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 12 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 13 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 14 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 15 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 16 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 17 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 18 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

Kuva 4.3: Testireitistä muodostetun graafin kytkentämatriisi. Olemattomat kaaret on kuvattu luvulla 0.

4.4 Testidatan prosessointi

Tietorakenteen testaamista varten kerätyn SIRI-datan prosessointi tietorakenteen testaamiseen soveltuvaan muotoon sekä datan tallentaminen tietorakenteeseen jakautui kolmeen vaiheeseen. Ensimmäisessä vaiheessa testidatasta eroteltiin valitulla testireitillä SIRI-standardin DirectionRef-kentän mukaisesti suuntaan 1 kulkevien bussilinjojen bussien data sekä poistettiin testidatasta sovelluksen kannalta tarpeettomat datakentät. Datan prosessoinnin toinen vaihe koostui testireitin kunkin bussilinjan aikataulun mukaisten linjalähtöjen erottelemisesta. Viimeisessä vaiheessa jokaisesta linjalähdöstä laskettiin sekä tallennettiin bussin ajoaika tietorakennegraafin osoittamilla kaarilla, eli testireitin eri osuuksilla.

4.4.1 Linjojen erottelu

Datan prosessoinnin ensimmäisen vaiheen aluksi määritettiin bussilinjat, jotka kulkevat testireittiä pitkin. Testireitin yhteensä 16 linjan linjanumerot esitetään taulukon 4.2 alimmilla riveillä. Kuten datan keräämistä käsittelevän alaluvun 3.4 lopussa

mainitaan, jokaisen viiden vuorokauden data jaettiin 500 MB:n kokoisiksi tiedostoiksi. Jokainen näistä 500 MB:n datapaketeista luettiin yksi kerrallaan CSV-tiedostosta Matlab-ohjelmaan. Tampereen joukkoliikenteen rajapinnasta kerätyssä SIRI-datassa on yhteensä 21 informaatiota sisältävää datakenttää. Näistä matriisimuotoon tallennettavaksi valittiin seitsemän datan prosessoinnin ja tietorakenteen kannalta oleellista dataa sisältävää kenttää. Kyseiset datakentät olivat: RecordedAtTime, LineRef, DirectionRef, DatedVehicleJourneyRef, Longitude, Latitude sekä VehicleRef. Edellä mainittujen datakenttien sisältämä data tallennettiin Matlab-ohjelmassa matriisimuotoon taulukon 4.3 mukaisesti.

Taulukko 4.3: Matriisimuotoon tallennetut SIRI-datakentät.

RecordedAtTime	LineRef	DirectionRef	DatedVehicle JourneyRef	VehicleLocation Longitude	VehicleLocation Latitude	VehicleRef
1390361392034	1	1	0505	23.728958	61.4986718	TKL_261
1390361392034	16	1	0510	23.767993	61.4981788	TKL_027
1390361392034	29	2	0510	23.771237	61.4984278	TKL_89
1390361392034	30	1	0510	23.781631	61.4902293	TKL_272
1390361392034	13	2	0515	23.625749	61.5087718	TKL_228
1390361392034	18	2	0520	23.663440	61.5081043	TKL_268
1390361392034	25	1	0520	23.793682	61.4979217	OB_13003
1390361392034	20	2	0520	23.789059	61.4991947	LL_33
1390361392034	21	1	0520	23.782316	61.4711287	TKL_011
1390361392034	5	1	0520	23.809557	61.4353428	TKL_012
1390361392034	15	1	0520	23.812673	61.4802742	Paunu_129

Matriisimuotoon tallennetusta datasta poimittiin kunkin testireitillä ajavan bussilinjan jokaisen testidatavuorokauden suunnan 1 data ja tallennettiin linjan omaksi vuorokausidataksi. Algoritmi 4.1 esittää linjadatan poiminnan vuorokausidatasta. Koska kerättyä testidataa oli viideltä vuorokaudelta ja testireitillä kulki 16 bussilinjaa, tämän vaiheen lopputuloksena syntyi yhteensä 80 vuorokausidatamatriisia.

Algoritmi 4.1 Poimi määrättyjen linjojen data vuorokausidatoista

```
1: Määritä valittujen bussilinjojen numerot
2: for all Valittujen bussilinjojen numerot do
3:     for all Vuorokausidata do
4:         Lue 500 MB:n CSV-muotoinen data levyltä
5:         Karsi tarpeettomat datakentät
6:         Tallenna data matriisimuotoon
7:         Poimi linjan suunnan 1 vuorokausidata
8:         if Vuorokauden viimeinen 500 MB:n CSV-tiedosto then
9:             Tallenna linjan vuorokausidatamatriisiksi
10:        end if
11:    end for
12: end for
```

Algoritmia tarkastellessa voi herätä kysymys, miksi kunkin testireitin linjan viiden eri päivän datoja ei yhdistetty, jolloin vuorokausidatojen määrä olisi vähentynyt 16 kappaleeseen testireitin bussilinjojen lukumäärän mukaisesti. Vaikka datakenttien lukumäärää karsittiin SIRI-datan alkuperäisestä 21 kentästä seitsemään, olivat vilkkaimmin liikennöityjen linjojen yhden vuorokauden datan sisältävien muuttujien koot matriisimuodossa lähes 300 MB. Vuorokausidatamatriisien lisäksi Matlab-ohjelmassa prosessoitiin samaan aikaan suuria CSV-tiedostoja, joten tietokoneen keskusmuistin loppumisesta aiheutuvien ongelmien minimoimiseksi jouduttiin jokaisen testireitin linjan kunkin vuorokauden datasta tekemään erillinen datamatriisi.

4.4.2 Linjalähtöjen erottelu

Seuraavassa testidatan prosessointivaiheessa edellisessä kohdassa muodostetuista vuorokausidatamatriiseista, joista jokainen sisälsi yhden bussilinjan yhden vuorokauden datan, eroteltiin aikataulunmukaiset linjalähdöt. Vuorokausidatamatriisin rivit järjestettiin SIRI-datan bussiajoneuvon identifioivan VehicleRef-kentän mukaisesti, jolloin saman linjalähdön datakentät saatiin datamatriisiin peräkkäin taulukon 4.4 mukaisesti. Tämän jälkeen SIRI-datan linjalähdön identifioivan DatedVehicleJourneyRef-kentän avulla datamatriisista poimittiin saman linjalähdön datarivit.

Taulukko 4.4: Matriisimuotoon tallennetut SIRI-datakentät järjestettynä VehicleRef-kentän mukaan, jolloin saman linjalähdön (DatedVehicleJourneyRef) datakentät saadaan peräkkäin.

RecordedAtTime	LineRef	DirectionRef	DatedVehicle JourneyRef	VehicleLocation Longitude	VehicleLocation Latitude	VehicleRef
1390572955012	16	1	1520	23.7099673	61.5287612	TKL_233
1390572956034	16	1	1520	23.7099118	61.5288005	TKL_233
1390572957001	16	1	1520	23.7098450	61.5288398	TKL_233
1390572958015	16	1	1520	23.7097783	61.5288833	TKL_233
1390572960012	16	1	1520	23.7096643	61.5289885	TKL_233
1390537930002	16	1	0630	23.9276573	61.4886257	TKL_235
1390537931001	16	1	0630	23.9274873	61.4887085	TKL_235
1390537932015	16	1	0630	23.9273100	61.4887860	TKL_235
1390537933013	16	1	0630	23.9271257	61.4888590	TKL_235
1390537934011	16	1	0630	23.9269365	61.4889277	TKL_235

Testidataa käsitellessä havaittiin, että datan joukossa oli vain muutaman datarivin sisältäviä linjalähtöjä. Nämä hyvin vähän dataa sisältävät linjalähdöt luokiteltiin virheelliseksi dataksi, joten ne hylättiin, kuten linjalähtöjen erottelun esittävästä algoritmista 4.2 voi havaita.

Algoritmi 4.2 Erottele linjalähtö

```
1: for all Linjan vuorokausidata do
2:     Lataa linjan vuorokausidata levyltä
3:     Järjestä linjan vuorokausidata VehicleRef:n mukaan
4:     for Järjestetty linjan vuorokausidata do
5:         Erottele yksittäinen linjalähtö DVJR:n avulla
6:         if Linjalähtö alle kymmenen datariviä then
7:             Hylkää linjalähtö
8:         else
9:             Tallenna eroteltu linjalähtö
10:        end if
11:    end for
12: end for
```

4.4.3 Datan tallentaminen tietorakenteen kaarialkioihin

Kolmannessa testidatan prosessointivaiheessa testidatasta erotelluista linjalähdöistä laskettiin bussien ajoajat testireitin osilla, eli kuvan 4.2 mukaisilla kaarilla, sekä tallennettiin muodostettu ajoaikadata tietorakenteen kaarialkioihin. Ajoaikojen tallentaminen jakautui kahteen vaiheeseen, ensimmäinen vaihe sisälsi haluttujen kaarien etsimisen tietorakenteesta sekä tietorakenteessa navigoimisen. Toisessa vaiheessa laskettiin sekä tallennettiin bussin ajoaikadata tietorakenteen kaarialkioon.

Ryhdyttäessä tallentamaan yksittäisen linjalähdön ajoaikoja oli tietorakenteesta aluksi etsittävä järjestyksessä ensimmäinen kaari, jota pitkin tarkasteltavan linjalähdön bussi ajaa. Valitusta testireitistä johtuen tässä testitapauksessa ensimmäinen tarkasteltavaa linjalähtöä sisältävä kaari etsitään iteroimalla läpi kaaritietoalkiot sisältävää structia. Ensimmäisen käsiteltävää bussilinjaa sisältävän kaaren löytymisen jälkeen seuraava vaihe oli etsiä kyseistä kaarta vastaava indeksi kytkentämatriisista. Tämä toteutettiin käymällä läpi rivi kerrallaan kytkentämatriisin rivien nollasta poikkeavia indeksialkioita ja vertaamalla niitä kaaristructista saatuun haluttua linjalähtöä sisältävän kaaren indeksiin. Vaihtoehtoisesti kaaren paikka kytkentämatriisissa olisi voitu selvittää kaarialkiossa olevien alku- ja loppukoordinaattien avulla, etsimällä kaaren alku- ja loppusolmun indeksit solmustructista ja niiden avulla selvittää kaarta vastaava kytkentämatriisin alkio.

Kunkin linjalähdön ensimmäisen kaaren tietorakenteesta etsimisen jälkeen seuraaviin kaariin siirrytään kytkentämatriisin indeksointia käyttäen. Jos käsiteltävän kaaren indeksi k on kytkentämatriisin kohdassa k_{ij}, löytyy seuraava kaaren indeksi rivin j nollasta poikkeavista alkioista. Mikäli rivillä on useampia nollasta poikkeavia alkioita, eli solmusta lähtee useita kaaria, on kaaristructista nollasta eroavia alkioita (kaaristructin indeksejä) käyttäen tarkastettava, kulkeeko kaarta pitkin kyseisen linjan busseja.

Testireitistä muodostetun graafin kytkentämatriisissa nollasta poikkeavia arvoja oli aina yksi, joten tämän työn tapauksessa seuraava kaaren indeksi löytyi alkiosta $k_{(i+1)(j+1)} = k_{j(j+1)}$. Tietorakenteen kaaria käytiin läpi siirtymällä kytkentämatriisin alkion jälkimmäisen indeksin mukaisesti seuraavalle riville aina niin kauan kunnes yksikään rivin indeksi-alkioiden kaarista ei sisältänyt kyseistä linjaa tai kunnes saavutettiin kytkentämatriisin viimeinen rivi. Algoritmi 4.3 esittää kaarien etsimisen.

Algoritmi 4.3 Etsi kaari

```
1: for all Linjalähtödata do
2:     for all Kaari do
3:         if Kaari sisältää linjaa then
4:             Aseta alkukaareksi
5:             break
6:         end if
7:     end for
8:     Etsi kaaren indeksi kytkentämatriisista
9:     Tallenna ajoaika kaareen
10:    Siirry kytkentämatriisin indeksin osoittamalle riville
11:    Valitse kaari kytkentämatriisin rivin nollasta poikkeavaa alkiota käyttäen
12:    while Rivin osoittamissa kaarissa linjaa do
13:        Tallenna ajoaika kaareen
14:        Siirry kytkentämatriisin indeksin osoittamalle riville
15:        Valitse kaari kytkentämatriisin rivin nollasta poikkeavaa alkiota käyttäen
16:    end while
17: end for
```

Datan tietorakenteeseen tallentamisen seuraavassa vaiheessa laskettiin bussin ajoaika kaarivälillä sekä tallennettiin bussia koskeva data kaaritietoalkioon. Linjalähdön datan jokaisen datapisteen etäisyys käsiteltävän kaaren alku- ja loppukoordinaateista laskettiin Haversinen kaavaa (2.1) käyttämällä ja tallennettiin sekä etäisyydet että etäisyyksiä vastaavat datapisteiden indeksit etäisyysvektoreihin. Näin saatujen kahden etäisyysvektorin etäisyysalkioista valittiin molemmista pienimmät, jotka määritettiin kaaren alku- ja loppusolmujen ohituspisteiksi. Mikäli algoritmissa 4.4 kuvatut hylkäysehdot eivät toteutuneet, bussin ajoaika tallennettiin kaareen. Ajoajan lisäksi kaaren dataan tallennettiin bussin linjanumero sekä kaaren alkupistettä lähimpänä olevan datarivin aikaleima. Algoritmi 4.4 esittää ajoaikojen tallentamisen.

Algoritmi 4.4 Tallenna ajoaika

```
1: Alusta kaksi etäisyysvektoria (d1, d2)
2: Laske Haversinella linjalähdön jokaisen näytteen etäisyys kaaren alkupisteestä
   ja tallenna d1:een
3: Laske Haversinella linjalähdön jokaisen näytteen etäisyys kaaren loppupisteestä
   ja tallenna d2:een
4: Järjestä d1
5: Järjestä d2
6: if Jos pienin etäisyys d1:ssa > 10 m then
7:     Hylkää
8: else if Jos pienin etäisyys d2:ssa > 10 m then
9:     Hylkää
10: else if Jos kellonaika kaaren alussa ja lopussa sama then
11:     Hylkää
12: else if Jos kaaren ajoaika negatiivinen then
13:     Hylkää
14: else
15:     Tallenna ajoaika kaareen
16:     Tallenna kellonaika kaareen
17:     Tallenna linjanumero kaareen
18: end if
```

Kuten algoritmista 4.4 on havaittavissa, ennen ajoajan, linjanumeron sekä kellonajan tallentamista datalle tehdään useita erilaisia tarkistuksia, jotka voivat aiheuttaa kaareen tallennettavan datan hylkäämisen. Dataan tutustuttaessa ja tietorakennetta kehitettäessä havaittiin lukuisia dataongelmia, joiden syiksi arveltiin muun muassa datan tallentumisen katkeamista sekä haamudataa. Mikäli etäisyysvektoriin tallennettu Haversinen kaavalla laskettu pienin etäisyys kaaren alku- tai loppupisteestä on yli kymmenen metriä, kaarta koskeva data hylätään. Syy liian suureen etäisyyteen voi johtua esimerkiksi siitä, että ajoneuvon datan lähettäminen on jostain syystä katkennut kaaren aikana, jolloin käsiteltävän kaaren loppupistettä koskevan etäisyysvektorin pienin etäisyys jää liian suureksi. Jos kellonaika kaaren alku- ja loppupisteen ohituspisteissä on sama, voi sama näyte olla kopioitunut koko kaarten välin tai vaihtoehtoisesti SIRI-datan aikaleima ei ole tallentunut oikein, minkä seurauksena kaaren ajoaikaa ei pystytä luotettavasti laskemaan.

Dataa käsiteltäessä havaittiin, että datan tallennusalgoritmi voi tuottaa myös negatiivisia kaaren ajoaikoja. Tämä voi johtua haamudatasta, joka tarkoittaa, että bussiajoneuvo on lopettanut bussireitillä liikennöinnin, mutta ajoneuvo lähettää edelleen SIRI-dataa esimerkiksi linjan päätepysäkin ja bussivarikon väliseltä siirtoajomatkalta. Tietorakenteeseen tallennetun datan avulla tuotettuja ensimmäisiä ajoaika-analyyseja käsiteltäessä havaittiin tietorakenteeseen tallentuvan myös hyvin pitkiä ja lyhyitä ajoaikoja. Pisimmät kaarien ajoajat olivat lähes tunnin mittaisia, kun taas lyhimmät saattoivat olla alle sekunnin pituisia. Myös nämä varsin epä-

määräiset ajoajat tulkittiin virheelliseksi dataksi ja ne eliminoitiin poistamalla 99,5 % fraktiilin ylittävät sekä 0,5 % fraktiilin alittavat ajoajat ennen varsinaista datan analysointia.

5. TULOKSET

Testidatasta tuotettiin luvussa esitettyjä tuloksia, jotta tietorakennetta käyttämällä saatujen bussien ajoaikojen luotettavuudesta voitaisiin tehdä alustavia arvioita ja tätä kautta pohtia työssä esitettyjen menetelmien jatkokehitystä. Osa datamatriiseista on tiivistetyssä muodossa tässä luvussa ja ne esitetään kokonaisina liitteessä A.

Taulukossa 5.1 on esitetty algoritmia 4.1 käyttämällä saatujen datamuuttujien muistin käyttö kunkin testidatapäivän osalta. Algoritmin 4.1 tuloksena saadut 80 vuorokausidatamuuttujaa yhdistettiin testidatapäivien mukaisesti siten, että taulukon riveillä esitettävät muuttujien muistin käytöt sekä datarivien määrät sisältävät testireitin yhteensä 16 bussilinjan SIRI-datan mukaisen suunnan 1 datan.

Taulukko 5.1: Matlab-ohjelmassa käsiteltyjen muuttujien muistin käyttö datakenttien karsimisen, tietorakenteeseen tallennettavien linjojen sekä ajosuunnan 1 valinnan jälkeen.

Päivämäärä	Muuttujien muistin käyttö (GB)	Datarivien määrä
Ke 22.1.2014	1,19	1 522 900
To 23.1.2014	1,80	2 292 100
Pe 24.1.2014	1,96	2 494 500
Ma 27.1.2014	1,61	2 054 500
Ti 28.1.2014	1,14	1 459 300
Yhteensä	**7,70**	**9 823 300**

Algoritmin 4.4 tuloksena saadun kaarimuuttujan muistin käyttö oli 9,903 MB eli 0,0097 GB. Kaarimuuttuja sisältää graafin kaaret sekä kaiken testireitiltä kaariin tallennetun datan. On syytä huomioida, että taulukossa 5.1 esitettävä datamuuttujien muistin käyttö sekä kaarimuuttujan muistin käyttö eivät ole täysin vertailukelpoisia. Tämä johtuu siitä, että taulukon datamuuttujat sisältävät bussien ajoaikadatan koko bussireitin osalta, kun taas kaarimuuttujassa on ajoaikadataa ainoastaan bussien testireittiä koskevalta osuudelta.

Kuvassa 5.1 on esitetty datamatriisi testireitin kaarien 6-18 ajoaikojen keskiarvoista sekunteina vuorokauden jokaiselta tunnilta. Datamatriisin arvot muodostettiin siten, että viiden testipäivän ajoajat yhdistettiin vuorokauden tuntien mukaisesti ja tämän jälkeen laskettiin kunkin tunnin keskiarvo. Mikäli arvoa ei ole, on käytetty merkintää –. Kyseinen datamatriisi kaikkien kaarien osalta esitetään liitteessä A.1.

klo/kaari	6	7	8	9	10	11	12	13	14	15	16	17	18
00–01	6.0	24.0	30.0	46.0	12.0	23.0	5.9	11.4	10.8	15.0	15.0	21.2	10.3
01–02	25.0	29.0	25.0	14.0	12.0	25.0	5.0	11.3	15.0	13.4	24.0	15.0	9.3
02–03	–	–	–	–	–	–	–	–	–	–	–	–	–
03–04	–	–	–	–	–	–	–	–	–	–	–	–	–
04–05	25.3	31.2	34.5	16.7	13.2	27.2	6.0	11.8	7.7	13.9	9.7	14.7	10.2
05–06	26.5	33.1	33.1	27.4	16.3	31.1	8.4	13.3	11.7	17.4	25.8	16.1	10.1
06–07	35.3	30.8	38.7	34.3	26.0	39.5	8.8	17.7	21.1	20.6	32.0	16.3	10.1
07–08	44.2	36.2	53.6	42.9	30.0	46.7	10.0	21.9	24.4	26.1	39.8	17.5	10.2
08–09	46.3	39.3	59.1	43.1	28.2	52.4	10.2	26.4	25.7	26.5	40.1	16.8	10.6
09–10	49.3	42.4	52.8	42.0	29.4	43.3	9.5	23.1	27.2	26.9	36.1	16.1	10.2
10–11	55.8	43.1	54.2	40.1	25.1	46.7	8.2	23.1	24.5	24.9	38.0	15.7	9.8
11–12	63.8	43.2	63.9	43.8	27.0	40.0	7.7	24.9	25.0	25.7	42.1	17.4	10.1
12–13	59.0	38.7	62.3	43.7	30.0	43.2	9.3	23.9	28.3	27.9	39.8	16.4	10.1
13–14	62.0	43.1	68.1	46.3	32.2	48.0	8.8	26.4	28.2	25.7	41.3	17.2	10.1
14–15	79.0	49.3	68.6	50.3	28.9	55.0	10.4	23.9	32.5	25.8	38.1	17.1	10.5
15–16	99.8	43.8	72.4	43.7	24.4	51.0	8.9	25.3	27.5	27.6	44.8	18.1	10.6
16–17	66.1	45.0	66.3	40.8	26.2	50.8	8.9	23.8	25.6	28.9	43.9	18.1	10.7
17–18	58.0	40.4	61.6	45.0	30.4	52.2	8.5	21.1	27.5	25.4	39.3	17.1	10.5
18–19	44.1	38.3	58.6	46.9	28.8	45.9	8.5	20.7	24.7	22.4	34.4	17.1	10.2
19–20	41.9	41.3	55.0	40.9	25.9	42.0	9.7	21.1	23.7	24.6	34.5	16.7	9.8
20–21	47.1	42.5	55.5	49.5	28.5	39.6	7.8	20.2	22.1	23.2	35.4	17.3	9.9
21–22	59.2	36.9	41.7	33.4	23.1	36.5	9.3	19.3	19.3	24.2	33.0	17.7	10.2
22–23	34.8	37.0	37.1	30.4	22.0	35.6	7.9	15.8	13.6	21.2	31.0	15.9	9.8
23–24	30.2	32.0	40.9	26.5	25.1	33.2	9.3	11.9	14.3	17.1	25.2	18.7	10.7

Kuva 5.1: Bussien ajoaikojen keskiarvot sekunteina kaarilla 6-18 vuorokauden jokaiselta tunnilta.

Tarkasteltaessa bussien ajoaikojen keskiarvoja tietorakenteen kaarilla havaitaan, että suurin osa kunkin kaaren kolmesta pisimmästä ajoajasta sijoittuu kello 7–9 ja kello 14–17 välisiin ajankohtiin. Tämä on tyypillisesti ihmisten työmatkaliikenteen ajankohta, toisin sanoen aamu- ja iltaruuhka-aika. Ajoaikojen keskiarvoista löytyy myös poikkeuksia, sillä esimerkiksi kaaren 17 pisin ajoaika on kello 00–01. Tämä voi johtua siitä, että yöaikaan bussiliikenne on verrattain vähäistä, joten yksittäiset pitkät tai lyhyet ajoajat voivat heilauttaa keskiarvoa yöaikojen pienestä näytemäärästä johtuen. Myöskään kaarella 1 yksikään kolmesta pisimmästä ajoajasta ei sijoitu kello 7–9 tai 14–17 välille. Pääosin kuitenkin ilta-, yö- sekä aamuyöaikaan, jolloin osa liikennevaloista on pois päältä sekä bussien matkustajamäärät että liikennemäärät ovat pienempiä kuin päiväaikaan, ajoajat ovat huomattavasti lyhyempiä kuin ihmisten työmatka-ajankohtina kello 7–9 sekä kello 14–17.

Kuvassa 5.2 on esitetty datamatriisi testireitin kaarien 6-18 ajoaikojen keskihajonnoista sekunteina vuorokauden jokaiselta tunnilta. Datamatriisin arvot muodostettiin siten, että viiden testipäivän ajoajat yhdistettiin vuorokauden tuntien mukaisesti ja tämän jälkeen laskettiin kunkin tunnin keskihajonta. Mikäli arvoa ei ole, on käytetty merkintää –. Kyseinen datamatriisi kaikkien kaarien osalta esitetään liitteessä A.1.

klo/kaari	6	7	8	9	10	11	12	13	14	15	16	17	18
00–01	0	0	0	0	0	0	0	2.8	15.2	3.5	9.7	12.7	2.0
01–02	0	0	0	0	0	0	0	2.3	9.6	1.5	0	1.7	0.6
02–03	–	–	–	–	–	–	–	–	–	–	–	–	–
03–04	–	–	–	–	–	–	–	–	–	–	–	–	–
04–05	31.0	9.4	11.6	1.2	1.2	2.1	0.6	1.9	10.3	1.1	2.5	4.2	1.5
05–06	13.7	10.8	10.6	12.3	7.1	10.4	6.2	3.4	9.7	4.5	17.0	8.1	2.1
06–07	14.4	8.3	13.3	12.0	10.5	13.1	8.7	8.6	14.0	7.8	17.2	8.5	2.0
07–08	20.0	12.4	19.6	15.5	10.6	18.1	9.3	12.2	14.9	12.5	18.2	9.0	2.2
08–09	22.2	14.4	16.1	15.0	12.5	18.9	8.6	15.3	15.3	12.1	21.0	7.8	2.6
09–10	21.5	14.9	14.4	15.4	11.5	18.1	7.4	14.2	15.8	12.2	20.1	7.6	2.3
10–11	25.2	15.1	17.8	13.7	11.9	16.6	6.5	14.5	14.3	11.4	20.5	7.6	2.0
11–12	28.1	15.8	22.7	16.9	11.6	14.3	4.5	16.1	15.3	12.4	22.2	9.0	1.9
12–13	19.9	12.5	20.4	15.6	11.3	15.4	7.9	15.7	16.2	14.0	21.7	8.1	2.0
13–14	23.0	15.8	22.2	16.5	13.2	17.3	8.8	17.9	17.2	12.1	23.3	8.7	2.5
14–15	30.6	20.0	21.1	21.4	14.4	19.9	9.6	15.9	19.4	11.9	21.4	9.3	2.5
15–16	28.5	14.4	22.2	15.1	10.7	19.0	6.6	17.1	17.0	13.9	21.3	9.1	2.2
16–17	33.1	18.2	19.8	18.1	10.3	21.3	7.3	13.4	14.9	13.4	21.1	9.2	2.2
17–18	31.3	15.3	19.7	14.2	10.4	20.7	6.8	11.6	14.9	11.1	19.4	9.0	2.2
18–19	21.1	12.4	24.6	20.8	12.1	17.6	6.0	12.5	14.0	9.6	17.3	9.1	2.0
19–20	26.3	15.9	21.0	16.3	13.3	13.6	7.8	13.1	15.2	11.5	18.9	8.2	1.5
20–21	20.3	13.0	17.0	27.1	13.9	15.0	6.1	10.5	13.2	10.3	20.2	9.1	1.4
21–22	19.4	12.9	14.4	14.7	13.3	11.5	6.9	10.9	13.5	12.3	19.6	9.5	1.7
22–23	21.7	12.7	12.5	13.9	11.3	13.4	6.0	8.0	13.2	8.5	15.7	8.3	1.6
23–24	21.8	8.4	13.1	14.4	13.4	6.4	11.0	2.4	10.8	5.7	15.9	11.3	3.0

Kuva 5.2: Bussien ajoaikojen keskihajonnat sekunteina kaarilla 6-18 vuorokauden jokaiselta tunnilta.

Tarkasteltaessa bussien ajoaikojen keskihajontoja tietorakenteen kaarilla suurimmat keskihajonta-arvot sijoittuvat aikavälille kello 13–18. Toisin kuin ajoaikojen keskiarvoissa, aamun työmatkaliikenteen aikaan eli kello 7–9 keskihajontojen arvot eivät kasva yhtä selvästi kuin iltapäiväliikenteen aikaan. Keskihajonta-datamatriisista nähdään, että aikavälillä kello 00–02 joidenkin kaarien keskihajonnan arvo on nolla. Tästä voidaan päätellä, että tietorakenteeseen on tallennettu näiden kaarien ja kellonaikojen osalta vain yhden bussin data. On syytä arvioida olisiko yöajan liikennettä kannattanut tarkastella erillään tai rajata ajoaikojen tarkastelu esimerkiksi aikavälille kello 04–22, sillä on kyseenalaista arvioida ajoaikojen keskiarvoja sekä keskihajontoja jonkin tunnin aikana, mikäli tunnin ajalta on tallentunut vain yksi näyte.

Kuvassa 5.3 on esitetty datamatriisi testireitin ajoajoista sekunteina vuorokauden jokaiselta tunnilta testidatan keräyspäiviltä. Kokonaisajoaika on summa kaarien ajoaikojen keskiarvoista kyseiseltä tunnilta. Tulokset laskettiin siten, että mikäli bussi ohitti osan testireitin kaarista ennen tasatunnin vaihtumista ja osan kaarista tasatunnin vaihtumisen jälkeen, kohdistui tasatunteja edeltävien kaarien ajoaika tasatuntia edeltävälle tunnille ja vastaavasti tasatuntia seuraavien kaarien ajoaika jälkimmäiselle tunnille.

klo/pvm	*ke* 22.1.	*to* 23.1.	*pe* 24.1.	*ma* 27.1.	*ti* 28.1.
04–05	343.7	341.0	404.7	330.9	283.8
05–06	421.4	417.5	405.0	399.5	414.2
06–07	547.6	545.8	512.5	527.7	549.1
07–08	654.4	659.9	639.6	706.4	675.0
08–09	702.8	646.8	678.0	681.6	686.7
09–10	632.1	636.9	651.1	648.2	636.2
10–11	625.3	613.9	625.3	659.6	640.2
11–12	659.9	658.2	649.7	706.3	683.2
12–13	637.4	649.6	686.6	640.7	645.8
13–14	695.3	684.6	692.8	704.6	671.6
14–15	711.2	736.7	780.7	703.0	686.6
15–16	744.7	775.6	736.8	723.6	770.5
16–17	721.4	698.2	605.1	697.3	693.8
17–18	698.0	687.3	656.4	687.9	698.8
18–19	644.4	644.7	585.9	579.5	630.2
19–20	594.9	607.1	587.4	600.7	638.4
20–21	610.4	575.1	626.8	588.2	584.5
21–22	573.2	544.3	583.8	520.8	552.5
22–23	473.8	547.8	487.5	449.5	443.4
23–24	401.1	530.0	499.7	423.5	401.7

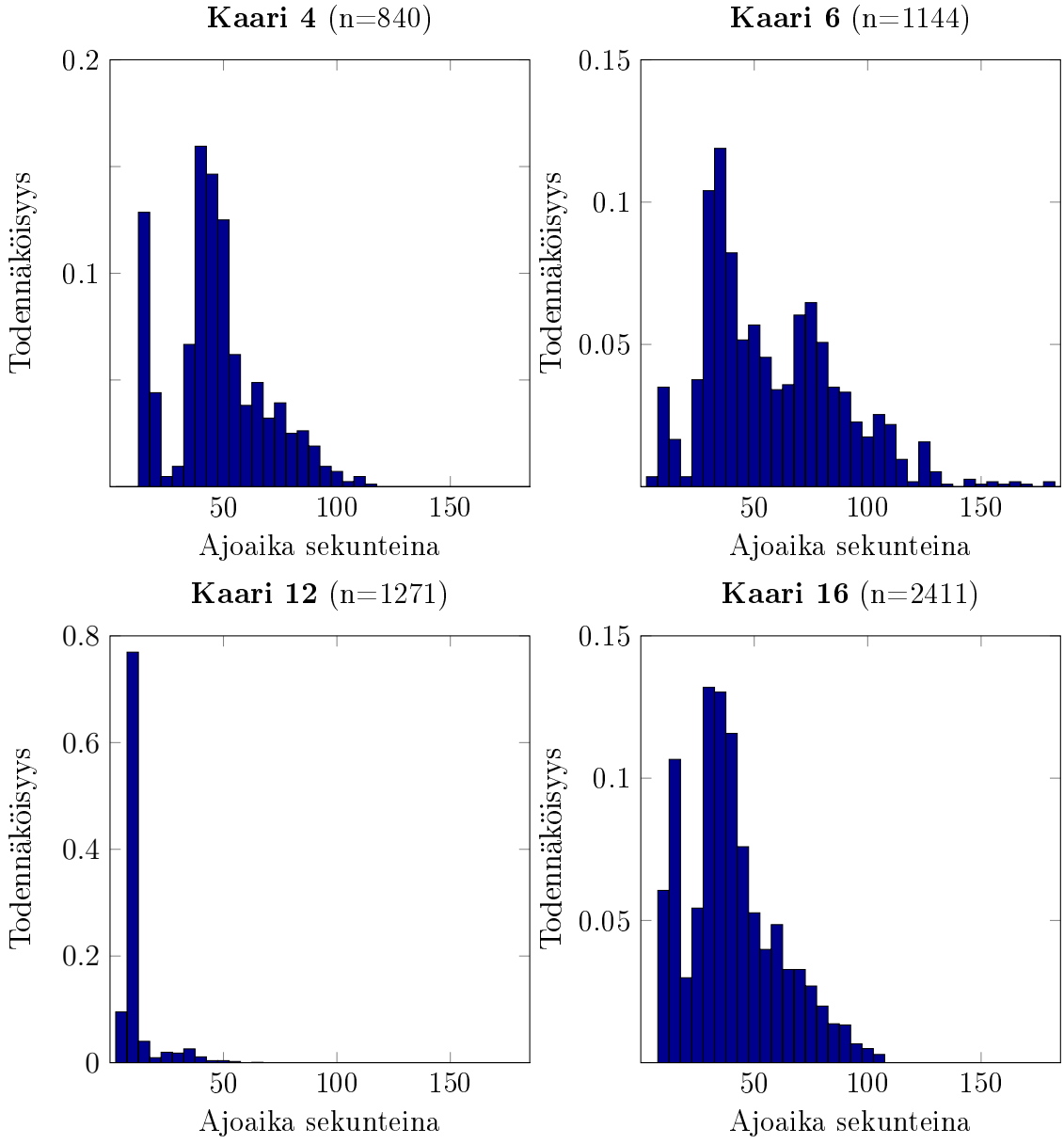
Kuva 5.3: Testireitin kokonaisajoaika sekunteina vuorokauden jokaiselta tunnilta testidatan keräyspäiviltä.

Testireitin kokonaisajoajan esittävästä datamatriisista nähdään, että perjantaita lukuun ottamatta jokaisen vuorokauden suurin ajoaika sijoittuu ajankohtaan kello 15–16. Perjantaina pisin ajoaika on kello 14–15. Muutenkin suurimmat ajoajat sijoittuvat pääsääntöisesti iltapäivän työmatkaliikenteen ajankohtaan, eli kello 14–17. On mahdollista, että perjantain pisimmän ajoajan muita arkipäiviä aikaisempi ajankohta johtuu siitä, että ihmiset lähtevät aikaisemmin töistä viikonlopun viettoon. Datamatriisista nähdään myös, että tiistaina 28.1. pisin ajoaika (klo 15–16) on yli 2,7-kertainen verrattuna lyhimpään ajoaikaan (klo 04–05).

Vaikka työn varsinaisena tavoitteena ei ollut ennustaa bussien ajoaikoja tietorakenteeseen tallennetun ajoaikadatan pohjalta, mainittakoon, että vertailupäivän

datasta eroteltiin muutamia linjan 16 busseja, jotka liikennöivät koko testireitin läpi. Esimerkiksi kello 11.07 linjalähdön aloittaneen bussin ajoaika testireitillä oli 663 sekuntia. Tarkasteltaessa kokonaisajoaikamatriisin kello 11–12 ajoajat sisältävää riviä, voidaan havaita, että ennustamalla vertailupäivän bussin ajoaikaa pelkästään edellisten päivien kaikkein testireitin bussien ajoaikojen avulla, oltaisiin päästy varsin hyvään ennusteeseen. Kyseessä on kuitenkin vain yksittäistapaus, jonka pohjalta ei voi tehdä pidemmälle meneviä johtopäätöksiä testidatan soveltuvuudesta ajoaikojen ennustamiseen.

Kuvassa 5.4 on esitetty histogrammit testireitin kaarien 4, 6, 12 sekä 16 ajoajoista. Yhteensä 18 kaaren joukosta valittiin esitettäväksi kyseiset neljä histogrammia, koska niistä voidaan erottaa erilaisia histogrammien muotoja. Histogrammit kaikkien kaarien osalta esitetään liitteessä A.1. Histogrammien y-akseli normalisoitiin jakamalla pylvään korkeus kaaren näytteiden kokonaislukumäärällä.



Kuva 5.4: Histogrammit testireitin kaarien 4, 6, 12 sekä 16 ajoajoista.

Kaaren neljä histogrammilla on melko selkeä kaksihuippuinen rakenne ja huipuista jälkimmäisellä on ”häntä” oikealle. Kyseisen kaaren alkuosassa on bussipysäkki, joten histogrammista voidaan erottaa ne bussit, jotka eivät pysähdy pysäkille (ensimmäinen huippu) sekä bussit, jotka seisahtuvat pysäkille (toinen huippu ja häntä).

Kaarta kuusi koskevasta histogrammissa nähdään, että ajoajat vaihtelevat 10 sekunnista aina lähes 200 sekuntiin, joten ajoaikojen hajonta on melko suurta. Kyseinen kaari sijaitsee eräänlaisessa sumppukohdassa – kaksien liikennevalojen lyhyessä välissä, jossa on vielä lisäksi bussipysäkki. On mahdollista, että liikenne jonoutuu kyseisessä kohdassa aamuruuhkan aikaan. Edellä mainitut seikat selittävät histogrammista nähtävät suuret erot bussien ajoajoissa kaarella.

Kaaren 12 histogrammista nähdään selkeä yksihuippuinen rakenne. Kyseisen kaaren alkusolmun kohta on bussipysäkki ja loppusolmun kohta liikennevaloristeys, jonka valot pääsääntöisesti näyttävät vihreää valoa bussin ajosuuntaan. Histogrammin yksihuippuisesta muodosta voidaan päätellä, että suurin osa busseista ei pysähdy liikennevaloihin. Toisaalta histogrammista on vaikea päätellä sijoittuuko bussin alkusolmupysäkille pysähtymisestä aiheutuva aika tähän vai edelliseen kaareen.

Kaaren 16 histogrammista voidaan erottaa kaksi huippua, joista jälkimmäisellä on ”häntä” oikealle. Kahden huipun välinen kuilu ei kuitenkaan ole niin suuri kuin kaaren neljä histogrammissa. Kaaren 16 alkusolmukohtana on bussipysäkki ja loppusolmuna liikennevaloristeys. Risteys on vilkkaasti liikennöity ja bussikaistalta oikealle kääntyvät autot joutuvat usein odottamaan kevyen liikenteen kadunylitystä, joka hidastaa bussien liikennettä. Näin ollen histogrammin muodosta voidaan päätellä, että osa busseista pääsee risteyksestä sujuvasti läpi (ensimmäinen huippu), kun taas osalla busseista kaarivälin ajaminen kestää selvästi pidempään (toinen huippu ja häntä) joko liikennevaloristeyksessä odottamisesta tai bussipysäkillä pysähtymisestä johtuen.

6. POHDINTA

Yhdeksi tässä diplomityössä kehitettävän tietorakenteen tavoitteista asetettiin suuren datamäärän prosessointi ja supistaminen siten, että datan joukosta voidaan samalla saada tehokkaasti ja kätevästi hyödyllistä tietoa - tässä tapauksessa informaatiota bussien ajoaika-analyyseihin. Verrattaessa taulukon 5.1 muuttujien muistin käyttöä ja testireitin ajoaikadatan sisältävää kaarimuuttujan muistin käyttöä, Matlab-ohjelmassa käsiteltyjen muuttujien muistin käytön kertaluokkaa saatiin pienennettyä lähes tuhannesosaan. Kuten luvussa 5 mainitaan, muuttujien muistin käytön ilmaisevat arvot eivät ole täysin vertailukelpoisia, koska taulukossa esitettävät datamuuttujat sisältävät bussien ajoaikadatan koko bussireitin osalta, kun taas kaarimuuttujassa on ajoaikadataa ainoastaan bussien testireittiä koskevalta osuudelta. Tästä huolimatta datan sisältävien muuttujien muistin käytön pienentymistä voidaan pitää merkittävänä ajatellen tietorakenteen jatkokehittämistä reaaliaikaisen datanhallinnan suuntaan.

Työssä kehitetyllä menetelmällä, jossa kunkin bussilinjan jokaisen linjalähdön bussista tallennetaan vain yksi datanäyte tietorakennegraafin kaarta kohti, sekä luvussa 4 esitettävillä datanhallinta-algoritmeilla päästään eroon SIRI-datassa havaituista ongelmista. Näitä alkuperäisessä datassa havaittuja ongelmia olivat esimerkiksi jo liikennöintinsä lopettaneiden bussien lähettämä haamudata, sekä useita kertoja kopioituneet samansisältöiset datarivit. Mikäli SIRI-dataa analysoitaisiin jatkossa reaaliaikaisesti, voitaisiin algoritmin 4.4 datan hylkäysehtoja kehittämällä mahdollisesti havaita ajoneuvon rikkoutumisen takia pysähtyneenä oleva bussi. Nykytilanteessa datan tietorakenteeseen tallentamisen yhteydessä karsitaan yli 99,5 % fraktiilin ylittävät ajoajat analysoimatta tarkemmin, johtuuko pitkä ajoaika esimerkiksi datan tallentamisen katkeamisesta tai bussin pysähtymisestä paikoilleen. Myös algoritmiikan tuottamasta negatiivisesta ajoajasta kaarilla voisi mahdollisesti havaita esimerkiksi yhteysongelmat bussin sekä taustajärjestelmän välillä, tai että ajoneuvo lähettää dataa, vaikka se on lopettanut vuoroliikenteen ja menossa bussivarikolle.

Testireitistä muodostetun graafin solmukohdat valittiin siten, että ne sijaitsevat keskellä liikennevaloristeystä tai keskellä bussipysäkkiä. Sekä pysäkit että liikennevalot ovat kohtia, joissa bussit ovat usein pysähtyneinä. Jos bussi pysähtyy liikennevaloristeykseen, liikennevaloista aiheutuva odotusaika tallentuu liikennevaloja edeltävään kaareen, kun kaarien välinen solmukohta on keskellä liikennevaloristeystä.

Mikäli bussi pysähtyy bussipysäkille ja vierekkäiset kaaret erottava solmu on keskellä pysäkkiä, on epäselvää, kumpaan bussipysäkkisolmun viereisistä kaarista pysähtymisestä aiheutuva odotusaika tallentuu. Tämä vaikuttaa myös ajoaikojen keskiarvoihin ja -hajontoihin, sillä on mahdollista, että bussipysäkille pysähtymisestä aiheutuva odotusaika tallentuu eri busseilla eri kaariin. Jatkossa solmut kannattaa sijoittaa bussipysäkkiä ennen sekä pysäkin jälkeen ja vastaavalla tavalla myös liikennevaloille. Näin saadaan muodostettua liikennevalo- sekä bussipysäkkikaaria ja tätä kautta voidaan tuottaa tarkempia arvioita liikennetilanteesta bussien ajoaikoihin perustuen.

Tietorakenteen testaamista varten muodostettu testireitti Tampereen keskustan ja Takahuhdin välillä oli käytännössä suora tienpätkä, eli testireitti ei sisältänyt risteäviä teitä. Tämän seurauksena testireitistä muodostetussa graafissa solmujen tuloja lähtöaste oli suurimmillaan yksi. Mikäli työssä esitettyä algoritmiikkaa haluttaisiin jatkokehittää, tai hyödyntää graafeille kehitettyjä algoritmeja, tulisi testireitistä tehdä monimutkaisempi esimerkiksi siten, että reitti sisältäisi risteäviä teitä ja kattaisi kokonaisia bussilinjoja. Tällöin solmuista lähtisi useampia kaaria jolloin myös muodostuvan graafin solmujen tulo- ja lähtöasteet olisivat suuremmat kuin yksi. Kokonaisia bussilinjoja sisältävä testireitti mahdollistaisi puolestaan kokonaisvaltaisemmat analyysit bussien ajoajoista.

Sää ja ajokeli vaikuttavat oleellisesti liikenteen sujuvuuteen etenkin talviaikaan. Testidatan keräyspäivinä sää oli talvinen, mutta poutainen. Yhtenäkään päivänä ei ollut lumisadetta. Tämän työn tapauksessa sään vaikutusta bussien ajoaikoihin kaarilla sekä testireitin kokonaisajoaikaan voidaan pitää pienenä, sillä sää oli hyvin samanlainen jokaisena testidatan keräyspäivänä.

7. JOHTOPÄÄTÖKSET

Tässä diplomityössä kehitettiin graafipohjainen tietorakenne, johon tallennettiin SIRI-bussidataa. Testidataa käsiteltäessä sekä tietorakennetta kehittäessä tutustuttiin Tampereen joukkoliikenteen reaaliaikaiseen rajapintaan sekä SIRI-dataan.

Kehitetty graafipohjainen tietorakenne on varteenotettava menetelmä suuren datamäärän pienentämiseksi sekä SIRI-datassa havaittujen dataongelmien ratkaisemiseksi. Työssä kehitetyllä menetelmällä, jossa kunkin bussilinjan jokaisen linjalähdön bussista tallennetaan vain yksi datanäyte jokaista tietorakennegraafin kaarta kohti, voidaan merkittävästi pienentää SIRI-dataa sisältävien muuttujien muistin käyttöä sekä päästä eroon SIRI-datassa havaituista dataongelmista.

Tietorakenteen testaamiseksi kerättiin testidataa viiden arkipäivän ajalta sekä rajattiin neljän kilometrin pituinen testireitti Tampereen keskustan ja kaupungin itälaidan välille. Testidataa prosessoitaessa sekä datan tietorakenteeseen tallennusvaiheessa ei havaittu sellaisia ongelmia, joiden takia tietorakennetta ja taustalla olevaa algoritmiikkaa ei voisi jatkokehittää ja testata suuremmalla datamäärällä. Tietorakennetta jatkokehitettäessä seuraavaan testireittiin tulee valita risteäviä teitä, jolloin testireitistä muodostettava graafi on tässä työssä esitettyä graafia monimutkaisempi.

Tietorakenteeseen tallennetuista bussien ajoajoista testireitillä muodostettiin erilaisia datamatriiseja. Analyysit datamatriiseista osoittavat, että tietorakenteeseen tallennetusta datasta voidaan tuottaa luotettavaa tietoa bussiliikenteestä. Bussien ajoaikojen vaihtelut noudattivat yleisiä työmatkaliikenteen ajankohtia ja ruuhkaaikoja. Ajoajoissa havaittiin suuria vaihteluita vuorokaudenaikojen välillä. Testireitin datasta muodostettujen ajoaikahistogrammien muotojen perusteella pystyttiin esittämään arvioita bussien etenemisestä tietorakenteen kaarilla.

Yhtenä työn tavoitteena oli arvioida tietorakenteen soveltuvuutta reaaliaikaiseen datanhallintaan. Koska datamuuttujien muistin käyttöä onnistuttuiin keventämään, työssä kehitettyä tietorakennetta voidaan pitää hyvänä lähtökohtana reaaliaikaista datanhallintaa tavoiteltaessa. Tässä vaiheessa datankäsittelyn hoitava algoritmiikka on vielä testivaiheessa, joten reaaliaikaista datan käsittelyä ajatellen algoritmeja tulisi kehittää tehokkaammiksi.

Tietorakenteen seuraavassa kehitysvaiheessa tulee kiinnittää huomioita etenkin tietorakennegraafin solmukohtien valintaan. Tämän työn testireitillä osa kaarien vä-

lisistä solmuista sijaitsi keskellä bussipysäkkiä, jonka seurauksena ei voitu tarkasti määrittää, kumpaan viereisistä kaarista pysäkillä pysähtymisen aiheuttama lisäaika tallentuu. Tietorakenne suunniteltiin helposti muunneltavaksi, joten solmujen uudelleensijoittaminen tai solmujen lisääminen ja poistaminen eivät tuota vaikeuksia. Tietorakennegraafin solmuja siirtämällä voidaan saavuttaa erilaista diagnostiikkaa. Esimerkiksi tarkastelemalla tietorakennegraafia yhden liikennevaloristeyksen alueelta on mahdollista tutkia bussien seisomis- ja odotusaikoja risteysalueella.

Tämän diplomityön todennäköisimpänä konkreettisena jatkokehityssuunnitelmana on hyödyntää työssä kehitettyä graafipohjaista tietorakennemallia Big Data -tyyppisessä tietokantalaskennassa. Yhdistämällä työn graafimalli sekä suuri tietokanta voidaan tässä työssä esitettyä liikenteen diagnostiikkaa jatkokehittää merkittävästi. Harkinnassa on myös erityyppisen datan, esimerkiksi liikennevalojen signaalidatan, tallentaminen tietorakenteeseen. Tällöin eri datamuotoja yhdistämällä saadaan lisää informaatiota liikennetilanteesta.

LÄHTEET

[1] Valtioneuvoston periaatepäätös julkisen sektorin digitaalisten tietoaineistojen saatavuuden parantamisesta ja uudelleenkäytön edistämisestä 3.3.2011. Helsinki 2011. Saatavissa: http://www.lvm.fi/c/document_library/get_file?folderId=1551281&name=DLFE-11828.pdf&title=Ehdotus%20valtioneuvoston%20periaatepaatokseksi%20-%20Julkinen%20tietoaineisto%20(3.3.2011).pdf

[2] Open Knowledge Foundation. What is Open Data? [WWW]. [viitattu 25.3.2014]. Saatavissa: http://okfn.org/opendata/

[3] Helsinki Region Infoshare. Mitä on avoin data. [WWW]. [viitattu 25.3.2014]. Saatavissa: http://www.hri.fi/fi/mita-on-avoin-data/

[4] Tampereen kaupunki. Yhteinen Tampere – näköalojen kaupunki Tampereen kaupunkistrategia 2025. Tampere 2014. Saatavissa: http://www.tampere.fi/material/attachments/k/6IoZ2as0k/DK_TRE_strategia_suomi_kevyt.pdf

[5] Ruohonen, K. Graafiteoria. Tampereen teknillisen yliopiston opetusmoniste. 2013. 108 s.

[6] Bapat, R.B. Graphs and Matrices. Intia 2010, Hindustan Book Agency. 171 s.

[7] Thulasiraman, K., Swamy, M.N.S. Graphs: Theory and Algorithms. Yhdysvallat 1992, John Wiley & Sons, Inc. 460 s.

[8] Gross, J.L., Yellen, J. Graph Theory and Its Applications. Second edition. Yhdysvallat 2006, Chapman & Hall7/CRC. 779 s.

[9] Pohjolainen, S. MAT-31090 Matriisilaskenta 1. Tampereen teknillisen yliopiston opetusmoniste. TTY Matematiikan laitos 2008. 158 s.

[10] Kokkarinen, I., Ala-Mutka, K. Tietorakenteet ja algoritmit. Suomi 2000, Satku. 293 s.

[11] Cormen, T., Leiserson, C., Rivest, R., Introduction to Algorithms. Yhdysvallat 1990, The MIT Press 1028 s.

[12] Sinnott, R.W. Virtues of the Haversine, Sky and Telescope, vol. 68, no. 2 (1984) pp. 159.

[13] Matemaattisten aineiden opettajien liitto. MAOL-taulukot. Suomi 2005, Otava. 167 s.

[14] Movable Type Scripts. Calculate distance, bearing and more between Latitude/Longitude points. [WWW]. [viitattu 31.3.2014]. Saatavissa: http://www.movable-type.co.uk/scripts/latlong.html

[15] Alger M. Real-time traffic monitoring using mobile phone data. 2005 University of Bristol, Vodafone Pilotentwicklung GmbH. 12 s.

[16] Wasson J., Sturdevant J., Bullocks D. Real-Time Travel Time Estimates using Media Access Control Address Matching. ITE Journal. June 2008, pp. 20-23.

[17] Tao S., Manalopoulos V., Rodriguez S., Rusu A. Real-Time Urban Traffic State Estimation with A-GPS Mobile Phones as Probes. Journal of Transportation Technologies 2012 2, pp 22-31.

[18] SIRI Management Overview - White Paper. CEN TC 278 Working Group 3 Sub Group 7. Version 1.0. 2005

[19] Gasparini L., Bouillet E., Calabrese F., Verscheure O., O'Brien B., O'Donnel M. System and Analytics for Continuously Assessing Transport Systems from Sparse and Noisy Observations: Case Study in Dublin. IEEE Conference on Intelligent Transportation Systems, Washington DC USA, 2011 pp.1827-1832.

[20] Lecue F., Schumann A., Sbodio M. Applying semantic web technologies for diagnosing road traffic congestions. ISWC12 The 11th international conference on The Semantic Web, Boston, 2012. pp.114-130.

[21] Tampereen joukkoliikenne. Reaaliaikaisen rajapinnan avaus mahdollistaa entistä tarkempien reittioppaiden kehittämisen. [WWW]. [viitattu 13.2.2014]. Saatavissa: http://joukkoliikenne.tampere.fi/fi/asiakaspalvelu/ajankohtaista/reaaliaikaisen-rajapinnan-avaus-mahdollistaa-entista-tarkempien-reittioppaiden-kehittamisen.html

[22] Myyryläinen T. Informaatiojärjestelmä. Tampere 30.10.2013. Tampereen joukkoliikenne. PowerPoint-kalvot.

[23] Ruotsalainen S. Tampereen joukkoliikenteen ICT-palvelujen kehittäminen. Diplomityö. Tampere 2013. Tampereen teknillinen yliopisto. Tuotantotalouden ja rakentamisen tiedekunta. 67 s.

[24] ITS factory. ITSFactory siriaccess developer-guide. [WWW]. [viitattu 19.2.2014]. Saatavissa: http://wiki.itsfactory.fi/index.php/ITSFactory_siriaccess_developerguide

[25] Tampereen joukkoliikenteen aikataulut 12.8.2013-1.6.2014. [WWW]. [viitattu 5.3.2014]. Saatavissa: http://aikataulut.tampere.fi/

A. LIITE: DATAMATRIISIT SEKÄ AJOAIKAHISTOGRAMMIT

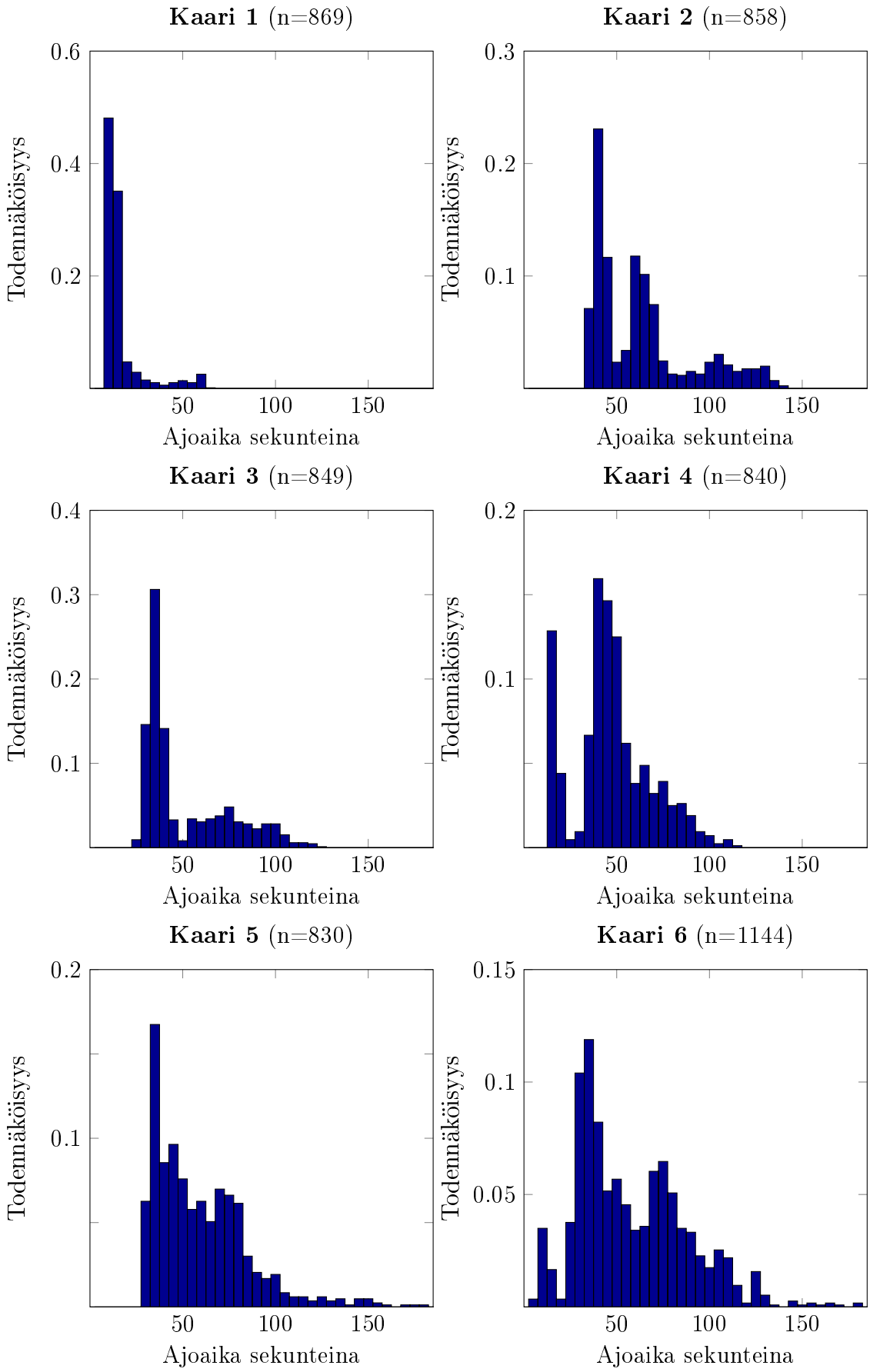
Liitteessä on esitetty datamatriisit bussien ajoaikojen keskiarvoista ja keskihajonnoista kaarilla 1–18 sekä histogrammit kaarien 1–18 ajoajoista.

klo/kaari	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
00–01	7.0	32.0	28.0	11.1	32.9	6.0	24.0	30.0	46.0	12.0	23.0	5.9	11.4	10.8	15.0	15.0	21.2	10.3
01–02	10.0	36.0	26.0	12.0	31.0	25.0	29.0	25.0	14.0	12.0	25.0	5.0	11.3	15.0	13.4	24.0	15.0	9.3
02–03	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
03–04	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
04–05	8.1	36.0	28.4	19.2	32.3	25.3	31.2	34.5	16.7	13.2	27.2	6.0	11.8	7.7	13.9	9.7	14.7	10.2
05–06	8.6	38.0	31.5	25.6	37.6	26.5	33.1	33.1	27.4	16.3	31.1	8.4	13.3	11.7	17.4	25.8	16.1	10.1
06–07	10.8	54.5	38.6	45.3	56.4	35.3	30.8	38.7	34.3	26.0	39.5	8.8	17.7	21.1	20.6	32.0	16.3	10.1
07–08	13.7	69.8	57.9	50.3	71.4	44.2	36.2	53.6	42.9	30.0	46.7	10.0	21.9	24.4	26.1	39.8	17.5	10.2
08–09	12.9	65.3	52.7	50.9	72.1	46.3	39.3	59.1	43.1	28.2	52.4	10.2	26.4	25.7	26.5	40.1	16.8	10.6
09–10	18.4	66.8	49.8	43.7	53.7	49.3	42.4	52.8	42.0	29.4	43.3	9.5	23.1	27.2	26.9	36.1	16.1	10.2
10–11	15.5	55.4	60.3	40.2	51.6	55.8	43.1	54.2	40.1	25.1	46.7	8.2	23.1	24.5	24.9	38.0	15.7	9.8
11–12	15.9	66.7	52.0	48.0	56.2	63.8	43.2	63.9	43.8	27.0	40.0	7.7	24.9	25.0	25.7	42.1	17.4	10.1
12–13	14.0	64.1	45.9	41.8	52.6	59.0	38.7	62.3	43.7	30.0	43.2	9.3	23.9	28.3	27.9	39.8	16.4	10.1
13–14	16.8	63.6	45.2	50.8	56.5	62.0	43.1	68.1	46.3	32.2	48.0	8.8	26.4	28.2	25.7	41.3	17.2	10.1
14–15	14.4	62.0	48.4	55.4	54.2	79.0	49.3	68.6	50.3	28.9	55.0	10.4	23.9	32.5	25.8	38.1	17.1	10.5
15–16	11.3	66.1	49.5	55.7	68.9	99.8	43.8	72.4	43.7	24.4	51.0	8.9	25.3	27.5	27.6	44.8	18.1	10.6
16–17	11.0	64.2	47.4	43.5	61.7	66.1	45.0	66.3	40.8	26.2	50.8	8.9	23.8	25.6	28.9	43.9	18.1	10.7
17–18	12.6	77.7	56.6	46.4	57.1	58.0	40.4	61.6	45.0	30.4	52.2	8.5	21.1	27.5	25.4	39.3	17.1	10.5
18–19	13.1	57.4	54.1	38.3	52.4	44.1	38.3	58.6	46.9	28.8	45.9	8.5	20.7	24.7	22.4	34.4	17.1	10.2
19–20	23.6	50.8	52.4	36.7	54.8	41.9	41.3	55.0	40.9	25.9	42.0	9.7	21.1	23.7	24.6	34.5	16.7	9.8
20–21	18.0	49.0	40.7	35.8	54.4	47.1	42.5	55.5	49.5	28.5	39.6	7.8	20.2	22.1	23.2	35.4	17.3	9.9
21–22	13.7	45.5	41.8	37.7	51.1	59.2	36.9	41.7	33.4	23.1	36.5	9.3	19.3	19.3	24.2	33.0	17.7	10.2
22–23	9.2	41.9	38.2	36.6	40.5	34.8	37.0	37.1	30.4	22.0	35.6	7.9	15.8	13.6	21.2	31.0	15.9	9.8
23–24	9.1	38.8	32.6	35.2	35.0	30.2	32.0	40.9	26.5	25.1	33.2	9.3	11.9	14.3	17.1	25.2	18.7	10.7

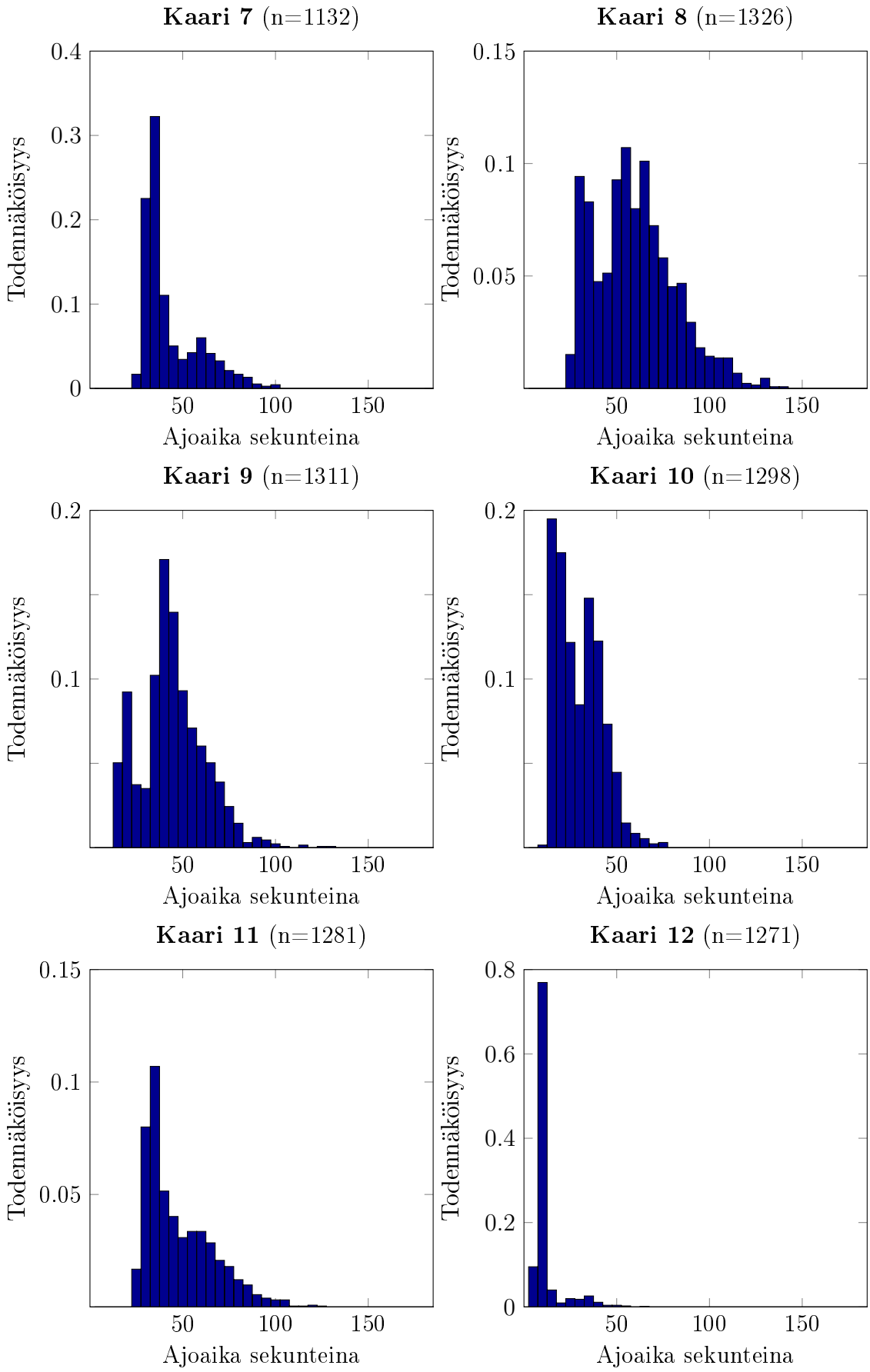
Kuva A.1: Bussien ajoaikojen keskiarvot sekunteina kaarilla 1–18 vuorokauden jokaiselta tunnilta.

klo/kaari	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
00–01	0	0	0	0	0	0	0	0	0	0	0	0	2.8	15.2	3.5	9.7	12.7	2.0
01–02	0	0	0	0	0	0	0	0	0	0	0	0	2.3	9.6	1.5	0	1.7	0.6
02–03	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
03–04	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–
04–05	0.7	2.0	2.5	15.6	2.5	31.0	9.4	11.6	1.2	1.2	2.1	0.6	1.9	10.3	1.1	2.5	4.2	1.5
05–06	3.5	6.5	6.9	14.3	13.5	13.7	10.8	10.6	12.3	7.1	10.4	6.2	3.4	9.7	4.5	17.0	8.1	2.1
06–07	4.7	17.3	18.7	15.5	23.3	14.4	8.3	13.3	12.0	10.5	13.1	8.7	8.6	14.0	7.8	17.2	8.5	2.0
07–08	10.7	29.0	25.0	19.6	37.1	20.0	12.4	19.6	15.5	10.6	18.1	9.3	12.2	14.9	12.5	18.2	9.0	2.2
08–09	8.7	24.1	23.4	20.1	39.3	22.2	14.4	16.1	15.0	12.5	18.9	8.6	15.3	15.3	12.1	21.0	7.8	2.6
09–10	16.6	22.7	20.0	17.5	16.9	21.5	14.9	14.4	15.4	11.5	18.1	7.4	14.2	15.8	12.2	20.1	7.6	2.3
10–11	14.5	15.7	23.6	17.7	18.8	25.2	15.1	17.8	13.7	11.9	16.6	6.5	14.5	14.3	11.4	20.5	7.6	2.0
11–12	12.9	25.1	24.0	19.7	17.8	28.1	15.8	22.7	16.9	11.6	14.3	4.5	16.1	15.3	12.4	22.2	9.0	1.9
12–13	11.0	25.1	19.2	18.8	16.7	19.9	12.5	20.4	15.6	11.3	15.4	7.9	15.7	16.2	14.0	21.7	8.1	2.0
13–14	13.7	25.7	20.6	17.8	16.3	23.0	15.8	22.2	16.5	13.2	17.3	8.8	17.9	17.2	12.1	23.3	8.7	2.5
14–15	11.3	28.3	27.1	22.4	19.7	30.6	20.0	21.1	21.4	14.4	19.9	9.6	15.9	19.4	11.9	21.4	9.3	2.5
15–16	4.9	33.3	25.5	24.7	29.1	28.5	14.4	22.2	15.1	10.7	19.0	6.6	17.1	17.0	13.9	21.3	9.1	2.2
16–17	4.1	30.7	25.9	22.6	26.3	33.1	18.2	19.8	18.1	10.3	21.3	7.3	13.4	14.9	13.4	21.1	9.2	2.2
17–18	6.0	36.2	26.4	23.6	24.8	31.3	15.3	19.7	14.2	10.4	20.7	6.8	11.6	14.9	11.1	19.4	9.0	2.2
18–19	10.8	21.0	23.8	17.6	16.6	21.1	12.4	24.6	20.8	12.1	17.6	6.0	12.5	14.0	9.6	17.3	9.1	2.0
19–20	18.6	18.2	25.0	13.8	17.9	26.3	15.9	21.0	16.3	13.3	13.6	7.8	13.1	15.2	11.5	18.9	8.2	1.5
20–21	16.2	15.4	17.1	13.1	16.7	20.3	13.0	17.0	27.1	13.9	15.0	6.1	10.5	13.2	10.3	20.2	9.1	1.4
21–22	10.2	12.7	15.4	13.6	16.8	19.4	12.9	14.4	14.7	13.3	11.5	6.9	10.9	13.5	12.3	19.6	9.5	1.7
22–23	2.7	12.8	18.9	20.4	15.4	21.7	12.7	12.5	13.9	11.3	13.4	6.0	8.0	13.2	8.5	15.7	8.3	1.6
23–24	4.3	8.7	6.7	21.2	8.1	21.8	8.4	13.1	14.4	13.4	6.4	11.0	2.4	10.8	5.7	15.9	11.3	3.0

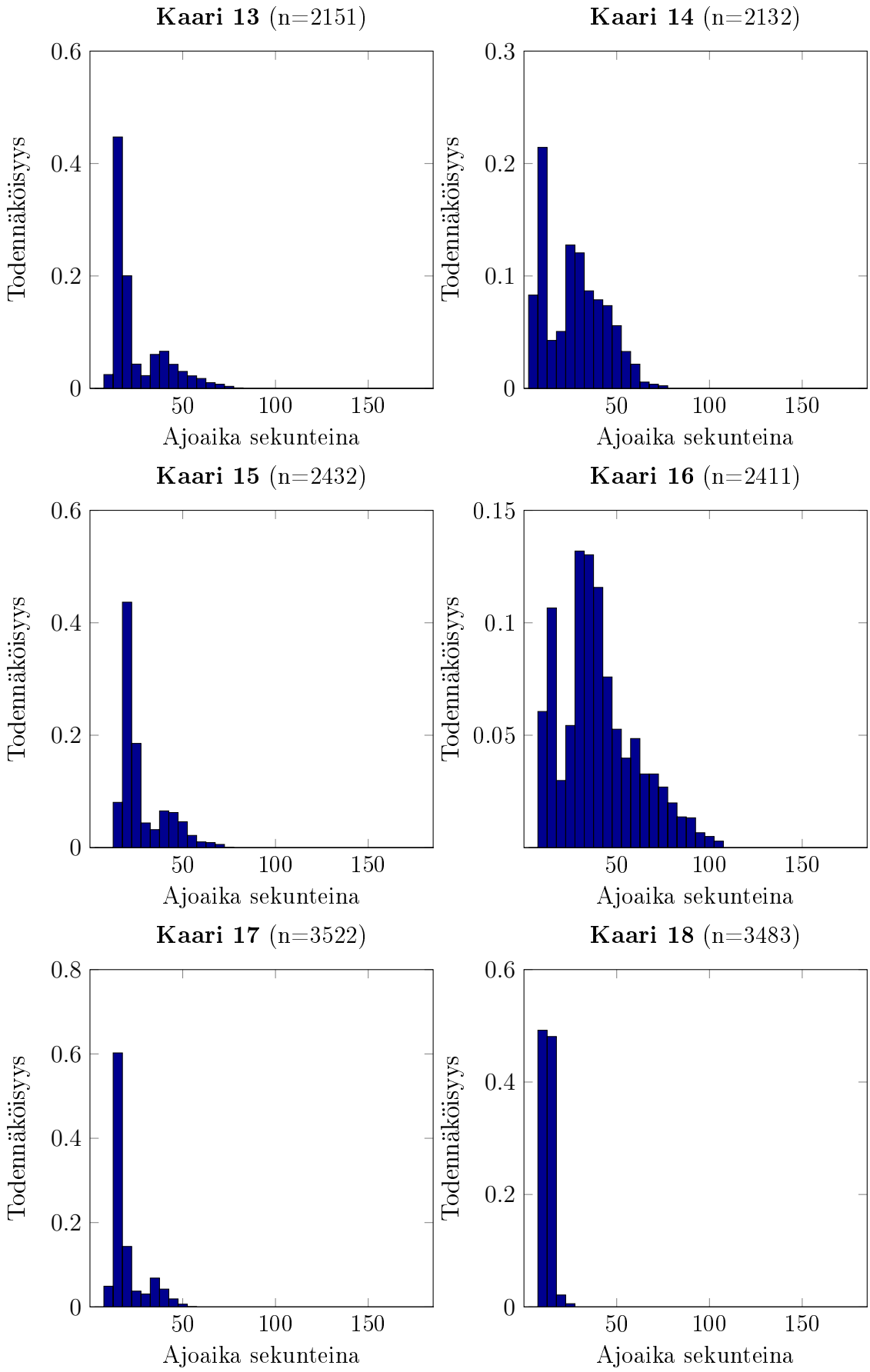
Kuva A.2: Bussien ajoaikojen keskihajonnat sekunteina kaarilla 1–18 vuorokauden jokaiselta tunnilta.



Kuva A.3: Histogrammit testireitin kaarien 1–6 ajoajoista.



Kuva A.4: Histogrammit testireitin kaarien 7–12 ajoajoista.



Kuva A.5: Histogrammit testireitin kaarien 13–18 ajoajoista.